Application of Blockchain Technology in Voting System

CS 656 Computer Networks Course Project

Jun Zhao
University of Waterloo
Student ID: 20683461
jun.zhao@uwaterloo.ca

ABSTRACT

Blockchain has emerged as a new technology that impacts our lives, and has inspired hundreds of innovative applications in financial, political and social areas. In voting application, blockchain technology introduces a new way to construct a more secure and transparent system with less inherent security vulnerabilities than currently-used systems. For the current project, we propose and implement a blockchain-based voting system on the Ethereum platform. Several experiments are carried to evaluate the proposed voting system in terms of reliability, security, and performance.

1 INTRODUCTION

Voting system has existed since hundreds of years. It serves as the foundation of any form of democracy and requires a secure and transparent approach to be successful. The traditional paper-based voting system is too much reliant on the procedural security of the officials to work correctly and honestly. This system is difficult to scale out and prone to corruption, resulting in the people's voice not being clearly heard[1]. Thus, the inevitable flaws of paper-based voting lead to a new form of system, electronic voting, to overcome these problems.

However, due to the uncertainty in the authenticity and integrity of the Electronic Voting Machines (EVMs), electronic voting system cannot be fully trusted to be used for large-scale elections[3][15]. Many electronic systems have been proposed and implemented, but a number of common security vulnerabilities, including unauthorized privilege escalation, incorrect use of cryptography, vulnerabilities to network threats, and poor software development processes, can be detected within these systems [12][20].

With the great success of Bitcoin[16][17], the emergence of blockchain technology introduces a new way to construct a more secure and transparent system which has less inherent security issues than the traditional paper-based and electronic voting system[21]. Furthermore, with custom functionality in the form of contracts, the generic blockchain-based platform, Ethereum[5], is able to provide a solution with high level of security and transparency, which could revolutionize the electronic voting system[21].

In this paper, we will first review some background information about blockchain technology and the Ethereum platform that our proposed prototype uses, as well as some currently existing blockchain-based voting applications, blockchain technology challenges and possible future directions in section 2. In section 3, we introduce our voting system features, design and implementation methodology in detail, and carried out several experiments to analyze the system reliability, security and performance. Finally, we conclude our work in section 4.

2 RELATED WORK

In this section, we first review the related blockchain technologies, with particular attention to the Ethereum platform. Then, we choose some currently existing blockchain-based voting applications to analyse their system characteristics. Finally, some open problems and potential future directions about blockchain technology are reviewed and discussed briefly.

2.1 Blockchain Technologies

Essentially, the blockchain is the decentralized database that is shared by all network nodes[7]. It serves as the transparent ledger with the transaction records which are stored in a linked chain of blocks. Each block contains a number of transactions and the hash code of its previous block, which forms an abstract chain of blocks. The transaction records are updated by miners, monitored by everyone in the network, and owned and controlled by nobody[21].

Blockchains are governed by a set of rules called the consensus protocol. Bitcoin and Ethereum are currently using the Proof of Work consensus protocol. Proof of Work is based on puzzles that are difficult to solve, but easy to verify the solution. This consensus requires solving the puzzle to add a new block to the chain, thus requiring computational power for creating new blocks. By accepting the longest chain as the correct chain, Proof of Work protocol makes attacks to the blockchain computationally impractical as long as a majority of CPU power is controlled by honest nodes [16].

With the successful application of Bitcoin in digital payment system, blockchain technology becomes more and more popular and have the capacity for reconfiguring all aspects of society and its operations[9][22]. The different kinds of existing and potential activities in the blockchain revolution can be broken down into three categories: Blockchain 1.0, 2.0 and 3.0[21]. Blockchain 1.0 means the deployment of cryptocurrencies in currency-related applications, while Blockchain 2.0 is contracts, which are more extensive than simple cash transactions. And Blockchain 3.0 refers to the applications beyond currency, finance, and markets[21].

In the blockchain context, we are now extending currency and payment transactions to the more complicated applications such as the recording and transfer of more complex assets like smart property and smart contracts, which needs a more robust scripting system [21]. Ethereum is a decentralized platform that runs smart contracts. It is also a Turing-complete programming language for building and publishing distributed applications[6]. It serves as a foundational general-purpose platform to implement the paradigm of a transactional singleton machine with shared-state[26].

Ethereum is so far the most well-known and used framework for smart contracts. With carefully design for the voting applications,

Ethereum can provide a reliable smart contract platform to achieve the required level of security and transparency, and maintain the necessary privacy of transactions[19][23].

2.2 Influential Blockchain-based Voting Applications

Before proposing our application for the state election, we select three typical implementations among currently existing applications, and briefly discuss their system characteristics:

(1) Votebook
In September 2016, Kaspersky Labs and The economist newspaper organized a case-study comptition on voting system implementation using blockchain technology. The NYU students offered an effective case study and proposed a new voting system, Votebook, based on the blockchain technology[11]. Votebook created a "permission blockchain" by granting encryption keys to nodes. The system uses voting machines resembling traditional Electronic Voting Machines (EVMs) which serves as nodes on a blockchain. This approach is more reliable and secure than the traditional electronic voting system using EVMs[8].

(2) Follow My Vote
Another implementation of blockchain voting system is Follow My Vote, which also allows for auditing, secure and tamper-proof counting of votes[24]. However, this system requires the voter to install the "voting booth" on the voter's personal device and submit identity documents to an organization holding the election, which results in many security flaws associated with the user authentication. This approach is clearly flawed and unrealistic because it relies on using voter's own devices.

(3) VoteWatcher
Currently, the most mature and tested blockchain voting application could be the VoteWatcher project, which is supported by Blockchain Technologies Corporation (BTC). VoteWatcher offers voting service not only for govenmental elections but also for an array of customizable elections. Based on their website[25], this system uses current methods to register and verify voter's identity, and generated ballots that contain unique tokens in the form of a QR code to prevent double voting. VoteWatch employs multiple methods to audit the election completely, including using paper ballots to compared with the electronic tally, burning all data from the election to a write-once DVD before connecting the voting machines, and using timestamped to create blockchain records for every scanned ballot. All these methods help create trustworthy voting experience with transparency for voters.

2.3 Challenges and Opportunities

As a new technology, blockchain is facing some open problems. We summarize three typical challenges that are related to our project:

(1) Scalability
As a decentralized database, blockchain includes a peer-to-peer network protocol. The blockchain database is maintained and updated by all nodes connected to the network. Each node in the network uses the same consensus protocol and executes the same instructions, which can be considered as a "world computer". However, this massive parallelization of computing across the entire network does not make computation more efficient[5]. With increasing amount of transactions, the blockchain becomes heavy and suffers the tough scalability problem to handle large-scale transaction processing in real-time faction[27].

(2) Security in programmable blockchain
In blockchain context, smart contract is a code fragment that could be executed by miners automatically[27]. Many smart contracts are now deployed on programmable blockchain such as Ethereum[5] and Hawk[13]. Similar to programming languages, the programmable blockchain itself is value-agnostic[5]. The security vulnerabilities in programmable blockchain mainly result from the smart contract code[2]. Bugs in smart contract could bring serious damages[10]. Therefore, smart contract attack analysis becomes more and more important in decentralized application development.

(3) Privacy Leakage
Another challenge of blockchain applications is about the user privacy. Even though blockchain users only make transactions with generated addresses instead of real identity, the transactional privacy cannot be guaranteed to prevent information leakage[14]. Besides, the recent study shows that Bitcoin users can be deanonymized, and presents a method to link user user pseudonyms to the IP addresses where the transactions are generated[4].

Blockchain is still a fast developing and evolving technology. There are now a number of efforts to address the above open problems. Nevertheless, blockchain also shows great potential in industry and academia. Some possible future directions are presented as follows:

- Big data analytics
  Blockchain can possibly be combined with big data. Transaction data could be stored in a distributed and secure way and used for further big data analytics[27].
- Artificial Intelligence
  The Blockchain technology can serve as an infrastructure to ensure that AI systems follow specified legal and safety regulations to be more integrated into human society[18].
- Application beyond currency, finance, and markets
  Another potential future direction of blockchain technology is about more extensive applications beyond current use cases such as currency, finance, and markets. Particularly, potential application areas can include government, health, science, literacy, culture, and art[21].

3 PROPOSED PROTOTYPE

In this section, we take into account the characteristics of currently existing voting applications, and propose a blockchain voting prototype which is implemented in the Ethereum smart contract framework.

First, we describe the application features that we need for the state elections. Second, we introduce our approach to this problem, and present some key points about implementation of this voting

system. Finally, we test and evaluate the proposed system in terms of validity, security and performance.

3.1 Problem Statement

Considering the currently existing blockchain-based voting applications, we collect some important use cases for the state election. For the current project, the voting system should include the following features:

- The system only allows the administrator to register new voter information;
- The system only allows the administrator to register candidates information;
- Individuals can check their personal voting information, which cannot be accessed by other voters;
- The system should not allow voters to vote twice, which we call "double voting";
- Voters can delegate once their vote to other voter in the same state;
- The system only allows the administrator to start and stop the election;
- The system can record and audit the election;
- The system can count the votes based on the state voting results, in a way like US presidential election, to obtain the winning candidate;
- The system can count the votes based on the absolute voting results (actually obtained votes for each candidate) to obtain the winning candidate.

3.2 Approach

With custom and reliable functionality of smart contract, the Ethereum platform is chosen to build our decentralized voting system. We would develop and test our decentralized application on a private test network. This is because the smart contracts deployment, transactions and calls in the public Ethereum netwrok (Homestead) do cost gas, paid by real *ether*, which is the value token of the Ethereum platform. Thus, making use of a private test network can avoid real public use cost. Our approach can be summarized as the following implementation steps:

(1) Set up a private Ethereum network with a number of test nods (3-5 nodes);
(2) Develop smart contracts using the Solidity language on the Ethereum platform;
(3) Deploy the smart contracts on the private network and the Ganache test network;
(4) Use the private network and the Truffle framework to test and evaluate the decentralized voting system.

In order to meet the functionality requirements, we maintain a candidate vote count table for each state, illustrated in Table 1. Whenever a voter gives his vote, a transaction will be added in the public ledge to update the corresponding vote count value in the candidate vote count table. When the election is finished, we can easily sum up the votes for each candidate to obtain the winning candidate based on the absolute voting results.

Moreover, if we want to count the votes based on the state voting results, like the US presidential election, we can first compute the winner for each state, and then add the votes of the state to the corresponding candidate. Finally, we select the maximum vote owner as the winner based on the state voting results.

Table 1: Candidate vote count table

State ID	Candidate_1	...	Candidate_n
1	<voteCount_1_1>	...	<voteCount_1_n>
2	<voteCount_2_1>	...	<voteCount_2_n>
...	...	...	...
m	<voteCount_m_1>	...	<voteCount_m_n>

As for vote delegation, we build an abstract delegation tree to represent the delegation relation. In the delegation tree, each voter represent a node. When the voter decides to delegate his vote to another voter, we create an edge from the voter to his delegate. When it comes to the case of multi-delegation in height, we compress the tree path and make the voter node directly point to the root delegate.

Figure 1 illustrates an example of the tree path compression. In the delegation tree, Node 1 and Node 2 delegate their vote to the Node 0, and Node 3 delegates its vote to Node 2. Then, Node 4 joins the delegation tree and delegates its vote to Node 3. By compressing the path from the beginning node, Node 3, to the root node, Node 0 (detailed implementation described in Algorithm 1), we can control the height of the delegation and keep the tree almost flat. In this way, we can greatly reduce the height of the delegation tree and improve the execution efficiency to find the root delegate.

Finally, by adding specific conditions to the execution functions in the contract, access control, such voter's personal information and administrative operations, can be achieved according to different function behavioral needs.

3.3 Implementation

Based on the Ethereum platform, we choose to use the *go-ethereum* client, usually called *geth*, which is implemented in the *Go* language. As for the contract development language, we use *Solidity* to implement smart contracts. The platform tools that we use in the project are listed in Table 2.

Table 2: Ethereum platform tools

	Tool	Version
Client	go-ethereum	1.7.3
Language	Solidity	0.4.18
Framework	Truffle	4.0.1
IDE	Remix	online

For the current project, we present some key points on implementation of the voting system from the following perspective:

(1) Setting up private Ethereum network

Before developing and deploying the smart contracts, we build up our own private Ethereum network. The main network configuration is presented in Table 3. Totally, we set up 1 bootnode and 3 member nodes, up to 4 of which can be miner nodes. When a

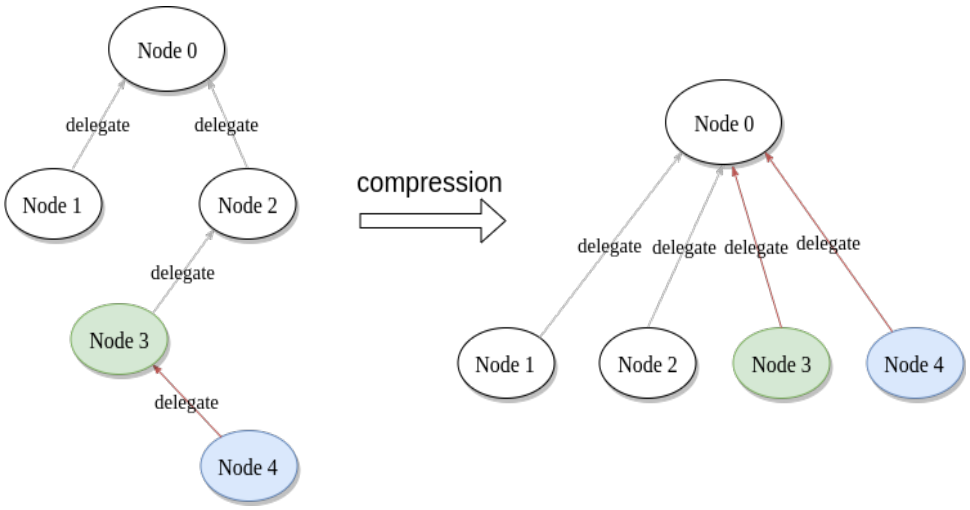

Figure 1: Tree path compression for multi-delegation in height

node connects to the network for the first time, it tries to connect the predefined bootnode. The bootnode can help the new member node to join the network and find other nodes.

Different from mining on the public Ethereum network, which is a complex task and only feasible using GPUs, mining in a private network setting, however, does not need heavy computational resources. We only use CPU instances to mine, and we can produce a stable stream of blocks at reasonably short intervals. Additionally, in order to facilitate block mining, we set relatively small value for the *difficulty* parameter in the genesis block.

Table 3: Private Ethereum network configuration

Parameter	Value
Networkid	1024
Bootnode number	1
Miner number	1-4
Block gaslimit	9000000000
Difficulty	40
rpc	enabled

(2) Contract development

In developing the voting contract, we combine the *mapping* type and the dynamic *Array* type to represent the candidate vote count table (Table 1), which is declared as *mapping (uint => Candidate[])* type. This table is stored as a state variable, and it can be accessed by all functions defined in the contract.

For the implementation of the delegation tree, we define a *struct* type called *Voter* to represent a tree node. In the *Voter* struct, an attribute of *address* type, named *delegate*, is used to store the voter's delegate address, serving as a pointer to the parent node in the delegation tree. Whenever a voter calls the *delegate* function to delegate his vote to another, we execute the procedure, described in Algorithm 1, to find his root delegate. While looking for the root delegate, we compress the tree path and make the voter node directly point to the root delegate. By compressing the tree path, we can keep the delegation tree almost completely flat, which helps to improve the execution efficiency.

Algorithm 1 Find the root delegate with path compression

```
1:  Input: the voter, voter, and his desired delegate, to.
2:  Output: the root delegate.
3:  next ← to.delegate
4:  while next ≠ NIL do
5:      if next.delegate ≠ NIL then
6:          to.delegate ← next.delegate
7:      to ← to.delegate
8:      next ← to.delegate
9:      require to ≠ voter
10: return to
```

When it comes to data access control, we use the function modifiers to restrict the access of different functions. In our smart contract, three function modifiers have been defined: *onlyCreator*, *onlyStarted*, and *onlyFinished*. The *onlyCreator* modifier only allows access for the creator of the contract to execute the applied functions, while the *onlyStarted* and *onlyFinished* modifiers control the access to the functions which are related to the start and end of the election. Other specific access conditions are defined in the function body according to the function behavioral needs.

(3) Contract deployment and testing

In the current project, we use the Truffle framework to deploy and test the smart contracts on the Ethereum networks. We configure two Ethereum networks: the private network that we set up and the Ganache test network which is part of the Truffle Suite. Both network are successful configured and contracts can be easily deployed to the target network by choosing different network parameter.

Deployment steps are run conditionally based on the network being deployed to. We define our own migration scripts, which are

JavaScript files that help to deploy contracts to the target Ethereum network. When the contracts are successfully deployed to the network, we first perform manual tests to the contract instances on the *Remix* IDE platform, which aims to discover and fix the major defects of the system. Moreover, in order to guarantee the functionality of the system and completeness of testing, we also create a test plan and a set of important test cases to make sure that the system functions as expected.

3.4 Experiments

(1) Experiment design

In order to evaluate the proposed voting system, we design an experiment to mock a small state election. For simplicity's sake, there are totally 2 candidates and 10 voters from 2 states in this election. When the elections starts, the voters can either *vote* for his candidate or *delegate* his vote to another desired voter. As the election ends, votes can be counted either based on the absolute amount for each candidate or based on the state voting results. One example of the experiment tests that we tested are described in Table 4.

Table 4: Experiment test example

Timestamp	Voter	State	Action
1	voter2	state1	delegate to voter4
2	voter10	state2	delegate to voter9
3	voter6	state1	delegate to voter2
4	voter4	state1	vote for candidate1
5	voter3	state1	delegate to voter1
6	voter5	state1	delegate to voter3
7	voter1	state1	vote for candidate2
8	voter9	state2	delegate to voter8
9	voter7	state1	delegate to voter6
10	voter8	state2	vote for candidate2

(2) Verification

To verify that the system meets required specifications and functions as expected, we create a set of test cases to check the voting status in each state. The final results of the candidate vote count table are shown in Table 5. Based on our testing results, the voting system fulfills the functional purpose.

Table 5: Experimental results

State ID	candidate1	candidate2	State winner
1	4	3	candidate1
2	0	3	candidate2
Total	4	6	-

In addition, the candidate vote count table is a public state variable of the contract, so the voting status is available to anyone, which can ensure the transparency of the election and make the system trustworthy.

(3) Security

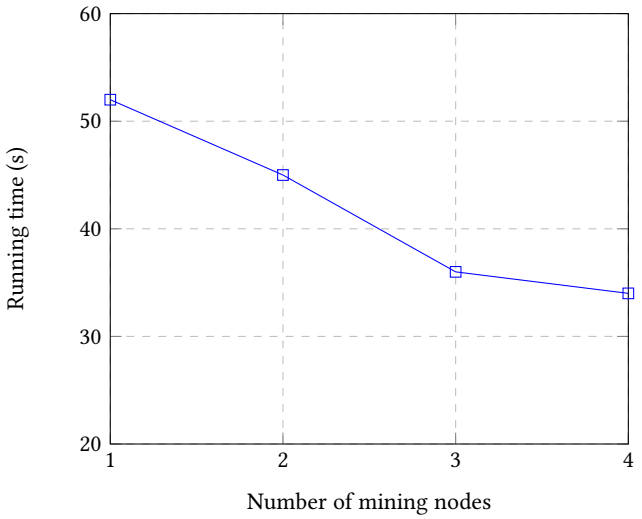

Figure 2: Testing running time with respect to mining resources

Since the proposed system is built and deployed on the Ethereum network, the basic security of the system can be guaranteed by the underlying blockchain technology and the Ethereum Virtual Machine (EVM). In the current project, we mainly test the system on the contract security level. Based on the requirements of the system features, some important security cases are tested as follows:

- The voter can have access to its voting information, but it cannot access other voter's information
- The voter can only vote or delegate its vote once in the election, "double voting" will be rejected
- The voter can take voting action only when the election is open
- Nobody can change the voting results when the election is finished

All the above tests have passed, the security requirements can be met on the smart contract level.

(4) Performance

On the Ethereum platform, a smart contract transaction is executed when the mining node validates and includes the transaction in a block it generates. Therefore, we also evaluate the system execution performance with respect to mining resources which are used over the whole network.

In our experiments, we select a set of typical test cases, totally 8 test cases, to execute. In the private Ethereum network that we set up previously, we measure the running time with respect to different number of miners, which varies from 1 to 4 mining nodes. Figure 2 shows the variation of testing running time with respect to the number of mining nodes. As the number of mining nodes increases from 1 to 3, we can observe that the testing running time goes down almost linearly. However, when the mining resources continue to grow from 3 nodes to 4 nodes, the performance difference becomes much less significant and the running time tends to be stable.

4 CONCLUSION

Blockchain technology introduces a new way to construct a secure and transparent voting system. In this paper, we implement a blockchain-base voting system in the Ethereum smart contract framework. Our system can be used for state elections with required level of security and transparency.

We believe that this work is a valuable exploration in blockchain voting area, and the concepts as well as the implementation of the proposed system could be helpful for the development of blockchain-based voting applications.

REFERENCES

[1] Ben Adida and Ronald L Rivest. 2006. Scratch & vote: self-contained paper-based cryptographic voting. In *Proceedings of the 5th ACM workshop on Privacy in electronic society*. ACM, 29–40.

[2] Nicola Atzei, Massimo Bartoletti, and Tiziana Cimoli. 2017. A Survey of Attacks on Ethereum Smart Contracts (SoK). In *International Conference on Principles of Security and Trust*. Springer, 164–186.

[3] Benjamin B Bederson, Bongshin Lee, Robert M Sherman, Paul S Herrnson, and Richard G Niemi. 2003. Electronic voting system usability issues. In *Proceedings of the SIGCHI conference on Human factors in computing systems*. ACM, 145–152.

[4] Alex Biryukov, Dmitry Khovratovich, and Ivan Pustogarov. 2014. Deanonymisation of clients in Bitcoin P2P network. In *Proceedings of the 2014 ACM SIGSAC Conference on Computer and Communications Security*. ACM, 15–29.

[5] Vitalik Buterin. 2014. Ethereum: A next-generation smart contract and decentralized application platform. *URL https://github. com/ethereum/wiki/wiki/% 5BEnglish% 5D-White-Paper* (2014).

[6] Vitalik Buterin et al. 2013. Ethereum white paper. (2013).

[7] Michael Crosby, Pradan Pattanayak, Sanjeev Verma, and Vignesh Kalyanaraman. 2016. Blockchain technology: Beyond bitcoin. *Applied Innovation* 2 (2016), 6–10.

[8] Kevin Kirby Fernando Maymi, Anthony Masi. 2016. Votebook: A proposal for a blockchain-based electronic voting system. (2016). https://www.economist.com/sites/default/files/nyu.pdf

[9] Marco Iansiti and Karim R Lakhani. 2017. The Truth About Blockchain. *Harvard Business Review* 95, 1 (2017), 118–127.

[10] Christoph Jentzsch. 2016. The history of the dao and lessons learned. (2016). https://blog.slock.it/the-history-of-the-dao-and-lessons-learned-d06740f8cfa5

[11] Eugene Kaspersky. 2016. Cyber Security Case Study Competition-Kaspersky. (2016). http://www.economist.com/whichmba/mba-case-studies/cyber-security-case-study-competition-2016

[12] Tadayoshi Kohno, Adam Stubblefield, Aviel D Rubin, and Dan S Wallach. 2004. Analysis of an electronic voting system. In *Security and Privacy, 2004. Proceedings. 2004 IEEE Symposium on*. IEEE, 27–40.

[13] Ahmed Kosba, Andrew Miller, Elaine Shi, Zikai Wen, and Charalampos Papamanthou. 2016. Hawk: The blockchain model of cryptography and privacy-preserving smart contracts. In *Security and Privacy (SP), 2016 IEEE Symposium on*. IEEE, 839–858.

[14] Sarah Meiklejohn, Marjori Pomarole, Grant Jordan, Kirill Levchenko, Damon McCoy, Geoffrey M Voelker, and Stefan Savage. 2013. A fistful of bitcoins: characterizing payments among men with no names. In *Proceedings of the 2013 conference on Internet measurement conference*. ACM, 127–140.

[15] Rebecca Mercuri. 2002. A better ballot box? *IEEE spectrum* 39, 10 (2002), 46–50.

[16] Satoshi Nakamoto. 2008. Bitcoin: A peer-to-peer electronic cash system. (2008).

[17] Arvind Narayanan, Joseph Bonneau, Edward Felten, Andrew Miller, and Steven Goldfeder. 2016. *Bitcoin and Cryptocurrency Technologies: A Comprehensive Introduction*. Princeton University Press.

[18] Steve Omohundro. 2014. Cryptocurrencies, smart contracts, and artificial intelligence. *AI matters* 1, 2 (2014), 19–21.

[19] Marc Pilkington. 2015. Blockchain technology: principles and applications. *Browser Download This Paper* (2015).

[20] Daniel Sandler, Kyle Derr, and Dan S Wallach. 2008. VoteBox: A Tamper-evident, Verifiable Electronic Voting System.. In *USENIX Security Symposium*, Vol. 4. 87.

[21] Melanie Swan. 2015. *Blockchain: Blueprint for a new economy.* " O'Reilly Media, Inc.".

[22] Don Tapscott and Alex Tapscott. 2016. *Blockchain Revolution: How the technology behind Bitcoin is changing money, business, and the world.* Penguin.

[23] Pavel Tarasov and Hitesh Tewari. [n. d.]. Internet Voting Using Zcash. ([n. d.]).

[24] Follow My Vote. 2017. Follow My Vote: The Online Voting Platform of The Future. (2017). https://followmyvote.com/

[25] VoteWatcher. 2017. VoteWatcher - The World's Most Transparent Voting Machine. (2017). http://votewatcher.com/

[26] Gavin Wood. 2014. Ethereum: A secure decentralised generalised transaction ledger. *Ethereum Project Yellow Paper* 151 (2014).

[27] Zibin Zheng, Shaoan Xie, Hong-Ning Dai, and Huaimin Wang. 2016. Blockchain Challenges and Opportunities: A Survey. *Work Pap* (2016).