TITLE GOES HERE!!!!!!!!!!

'SUBTITLE GOES EHRE!!!!'

UNIVERSITY OF BIRMINGHAM

Matthew Flint

1247903 - mxf203@bham.ac.uk

This work was conducted as part of the requirements of the Module 06-26587 'Computer Science Project' of the Computer Science department at the University of Birmingham, UK, and is submitted in accordance with the regulations of the University's code of conduct.

4th April 2017

Contents

1 Abstract

In abstract, you should give an overall view of the paper including the importance and necessity of the paper topic (the first line of the abstract), the previous works and difficulties on this topic(second and third lines), the big picture of your novelty in this paper (fourth and fifth lines), and verifying your results (the final part of the abstract).

1. It should not exceed 250 words
2. It should be written in one paragraph.
3. It should be written in the past tense as it refers to work done.
4. Long words should be followed by its abbreviation which would be used through out the abstract and paper.
5. It should not cite any references (except in rare cases)
6. It should never give any information or conclusion that is not stated in the paper
7. Must be accurate with respect to figures quoted in the main text.

2 Introduction

Existing electronic voting systems all suffer from a serious design flaw: They are centralized by design, meaning there is a single supplier that controls the code base, the database and the system outputs while also supplying the monitoring tools to verify the result [1]. The lack of an independently verifiable system means that, once voters mark their ballot choice, they must place their trust in the organization, that their vote is recorded and counted as intended. The lack of an independently verifiable output, makes it difficult for these centralized systems to acquire the trustworthiness required by voters, which potentially limits voter participation, or cast doubt upon the published output of an election.

Despite the digitalisation of many aspects of modern life, elections are still being largely conducted offline, on paper [2], although the use of Electronic Voting Machines has been steadily growing over recent years. Paper ballots are the traditional tool for voting and are typically marked by a human (voter) and then tallied by a machine. While costing less than most electronic systems to run they rely on physical security and trust in polling stations to not manipulate and to properly handle them [3]. Postal votes also utilize paper ballots and are used to allow voters to not have to physically attend a location in order to vote. These also suffer from the same flaws as traditional paper ballots while increasing the opportunities for attack during their traversal through the postage system.

Voting systems comprise of five main components:

1. A registration service for verifying & registering legitimate voters
2. Voter authentication stations with the task of determining a voters authorization to vote based on the completed work of the registration service
3. Voting stations where the voter makes choices on a ballot;
4. A device called the ballot box where the ballot is collected
5. A tallying service that counts the votes and announces the results

Of the five main components listed above, only `(2)`, `(3)` and `(4)` are used during an election, with `(1)` being being required for an election to take place and `(5)` happening post election [4].

Current Electronic Voting (E-Voting) takes two forms; using a machine in a polling station, rather than a ballot paper and pencil, or casting a vote over the internet. The former tends to refer to a Direct Recording Electronic system which typically displays ballot options on a screen that can be activated by the voter and then records that voting data in memory components to be processed later. However, as with many electronics there is an inherent problem of the ability to modify software and potentially insert malicious code [3]. This has been an issue raised over several recent elections and a study from 2015 concluded that 43 American states would be using Electronic Voting Machines that are at least 10 years old during the last presidential election [5]. The latter, while having to deal with issues such as privacy, fraud, voting under duress, and corruption, does nothing to improve a voter's trust. The election as the voter must assume that once they have cast their vote it will be recorded and counted honestly.

In order for our proposed E-Voting system to be a tangible challenger to more traditional voting methods it must be able to provide the current systems services, at least at the same level but preferably with improvements, while also providing substantial benefits to justify adoption. There are several standard requirements that a voting system should adhere to, each holding

equal weight: security, functionality, privacy, usability, and accessibility [6]. A "secure" voting system means one that cannot be tampered with or manipulated in any way, ensuring that votes are accurately recorded as cast. It also ensures that additional votes cannot be cast after the polls have closed or tampered with at any stage of the process. System functionality can be broad but should include; The correct registering and recording of all votes cast, permitting a voter to vote for any candidate they have the right to vote for, allowing only eligible registered voters to vote & only allowing each voter to vote once. Voters have the right to a secret ballot and to cast their vote in private [6]. This is essential to protect voters from being coerced or bribed into voting a certain way, this means that our system should not provide a receipt or any way for another person to determine the contents of a voter's ballot. On top of this the system should be easy for voters to use, meaning it's as intuitive as possible, and maintain universal vote access. It should avoid introducing bias by selecting platforms that are more available to some groups than to others as the choice of the platform, language, ballot format, or devices may seem innocuous, but it may actually prevent small factions of the voters to cast their vote [7].

Whilst maintaining these essential foundations already provided in traditional voting systems, there are several improvements and additions which I intend to explore. The first benefit in using a blockchain to log votes is in its decentralized nature. This means there is less need for trust to be placed in a centralized organization where votes are hidden behind closed doors. It also has the benefit of being significantly harder to tamper with as, once a transaction has been verified, an attacker would need to possess at least 51% of the computing power of the network to attempt to forge transactions. Any attempts to otherwise use a forged block will be noticed by the rest of the network and ignored. This decentralized system also brings in more transparency as anyone can view transactions in the blockchain leading to higher levels of trust in the elections outcome. This is further strengthened by the independent verifiability which could be performed by anyone, therefore removing the need to trust the election organizers declaration of the outcome. Once a transaction (vote) has been confirmed in the blockchain (and has further blocks built upon it) this vote for a candidate becomes immutable, meaning that the entire outcome of an election will be stored indefinitely and is able to be accessed at any time in the future.

Verifiability properties of electronic voting are divided in two categories. "Individual verifiability", which involves auditing the processes of vote creation and vote storage by the voter; and "Universal verifiability", which ensures that only votes from eligible voters are stored in the ballot box and that all stored votes are properly tallied, which can be performed by anyone [8]. Systems providing both types of verifiability are known as "end-to-end verifiable systems".

One of the individually verifiable properties is cast-as-intended verification, which is focused on the audit of the vote creation process. Another property is recorded-as-cast verification, aimed at auditing the correct reception and storage of the vote in a remote voting server [8].

There has been some research conducted in this area already and several protocols for blockchain based voting have been proposed. *Pierre Noizat* proposes a system [1] where each candidate provides a unique public key, KeyC, to each individual voter along with a singular bitcoin address, AddressC, which the final sum of the voting transactions will be sent to. Each voter is also assigned an individual public key, KeyA, by the election organizers and; either they generate a Key Pair themselves (this could be done by the voting software for better usability), KeyB, or be assigned the KeyB by the organization. From these three public keys the voter can generate a `2-of-3` multi signature address which represents the vote of B in favour of C. This address is then funded with a bitcoin micropayment (around the price of a postage stamp), either funded by the voter or organization, and this is the voters confirmation of their vote. After a few hours this

ballot is securely logged on the blockchain and, as the multisignature address was funded, a voter can check that this address is represented in the blockchain and that their vote was registered. It should be noted that there is no way to guess neither the voter (B) nor the candidate (C) from a multisignature address without knowing all three public keys (KeyA, KeyB, and KeyC) and knowing to whom they belong. Once the election is concluded the organization is able to, via the `2-of-3` multi signature address, spend the coins the voter gave to the candidate to fund the address of the candidate, AddressC. This provides an unequivocal link between the vote and the candidate which can be seen and validated by anyone.

While the proposed method does provide a valuable alternative to current, proprietary electronic voting systems and has the benefits of protecting the secrecy of the ballots, allowing free, independent audits of the results and minimizing the trust level required from the organizers [1]. It does come with several drawbacks; the independent validation of votes before the organization funds a candidate's public address, AddressC, can only be done by the voting individual on their ballot choice. The protocols dependence on the Bitcoin blockchain could pose problems with subsequent elections as there is no definitive boundary between one election and the next. The currency units, although in very small denominations, could be transferred out of the election to private addresses for an individual's gain (though even if 100% of the vote currency was lost, the cost of an election would likely be less than if done using current methods).

Another proposal is that of Universal Cast-as-Intended Verifiability [8] which allows any party (not only the voter) to publicly verify that an encrypted cast vote really matches the selection of a voter. Their proposal allows a voter who's eligible to vote, to register with a registrar who then generates a pair of public-secret values for each voting option in the election. These secret values are sent to the voter, while the public ones are published, linked to the voting options they are related to. During the voting phase, the voter provides her selected voting options and a subset of the secret values she received during registration to the voting device. The voting device then encrypts the voters selections and creates a non interactive zero-knowledge proof, which will be valid only in the case that the voting device encrypted what the voter selected. Thanks to the zero-knowledge property of the proofs, they can be publicly verified while maintaining the voters privacy.

While this may seem like a vastly superior proposal externally, the additional complexity of the underlying system, that is the inclusion of zero-knowledge proofs, should not be underestimated. Furthermore, zero-knowledge protocols, despite being proposed in the late 1970s, are still in their relative infancy when compared to Blockchain technology (e.g. zCash whitepaper [9], 2014) and therefore have not been through the same level of scrutiny nor do they have the same level of development or adoption. The protocol also requires the voter to supply a secret value for each of the voting options they did not choose which, may require considerable effort if the ballot is large enough, and is counter intuitive to what would usually be expected to cast a vote.

In this paper I outline the design of an "end-to-end verifiable system" built on the Ethereum Blockchain. In this system, a voter can register an Ethereum address which is then added to a list of *allowed addresses* inside a smart contract. Upon the ballot commencing, the voter will be able to modify the allocation of their vote inside this contract up to the point of the ballot ending. Due to the strict, programmable nature of Ethereum contracts we can be sure that; no votes can be modified after the ballots end date, only the voter in control of the Ethereum address can change the vote associated with that address and only Ethereum addresses pre-registered to the contract are able to vote. Anyone can verify the number of votes for a candidate by querying the contract, and as they can be assured that only those addresses added to the contract are able to vote, we can be sure of the validity of each of these votes.

3 Design

3.1 Overview

In this paper, I present an "end-to-end verifiable system" built on the Ethereum Blockchain, i.e., a system where a voter can be assured their vote has been fairly counted, only eligible voters are allowed to vote and the tallied results of the election are publicly verifiable.

Although this system has been designed and developed with the idea of a national general election in mind, the protocols and ideas involved could be applied to smaller scale ballots which wish to provide transparency in their audit. Although we wish to minimize trust in a central authority, due to the nature of these type of elections (where there needs to be some degree of voter eligibility verification), we cannot fully decentralize this system as we need to only allow those eligible the rights to vote. Despite needing to verify an individual we still need to ensure that their votes are publicly anonymous, especially given the public transactions underpinning the blockchain concept while providing the ability for an individual to verify that their vote was correctly counted.

The designed schema for this protocol is the following:

1. Ballot creation:
    - The available ballots in the election are designed and decided upon.
    - A smart contract is created and pushed to the blockchain for each ballot containing all of the voting options.
2. Pre-election voter verification:
    - Voter registers with an external voter registrar after providing a valid ID (this could be accomplished using pre-existing government electoral registration protocols).
    - This external registrar generates a *user_id* and *nonce* which can be used by the voter to log in to the system.
    - This *user_id* is then tied to any ballots the voter is eligible for.
3. Voter registration:
    - The voter logs into the system using the received *user_id* and nonce upon which time they are immediately required to change their login credentials.
    - The voter can then register to vote through the online system for each of the ballots they are eligible for.
    - A unique Ethereum address, *voter_address*, is generated and validated (while not being linked to the *user_id*)
    - The *user_address* is added to the ballots smart contract which entitles this address to vote in that ballot.
    - The address is funded with enough Ether for the voter to cast their vote.
4. Voting:
    - When the voter decides to cast their vote they are presented with an interface mirroring the options in the ballot smart contract.

- Upon the voter selecting their options the contract is funded with the voters selected options.
- At this point the voters choice is immutably entered into the blockchain and the tally is verifiable by all.

5. Election result:
    - Once the election is over, due to the nature of the smart contract design, no more votes can be added for any candidate.
    - The tally for each candidate is publicly verifiable by anyone along with all of the funded transactions casting votes.

3.2 Docker

Over the past few years, container technology has become increasingly promising as a means to seamlessly make software available across a wider range of platforms & allow developers to worry less about the eventual runtime environment (as this can be standardized). Docker containers provide a way to "wrap up a piece of software in a complete filesystem that contains everything it needs to run" [51].

There are several benefits to the use of Docker containers; This could substantially reduce the effort required to create and validate a new software releases, since docker containers create their own dedicated environment, testing on one OS means that the application will run the same on any OS capable of running Docker. In addition, docker containers provide a quick and easy way to install and use a software release, for our application this could mean faster patches if needed as you would simply need to swap out the docker image being used. Other benefits include faster bootup of containers (compared to just virtualization), closer development to production parity, immutable infrastructure and improved scaling (on a per-container basis) [52]. There is also a security argument to be made for using containers, as they offer a degree of isolation for each enclosed application and only expose those services which you decide. The previous point about patching also add to security, as legacy applications often forgo patches in sensitive environments due to the possibility of breakages. When using containers, changes can be fully tested in the container which can then be swapped into the production environment [53].

All of my development for this project was conducted inside of Docker containers. I decided on this because there are very distinct separations between the applications which make up my system (this is described in section LINK TO SECTION), so running each one inside a docker container seemed the logical thing to do. It also meant that I could control the startup of the system as a whole & expose (between containers) only those services necessary to communicate.

3.3 Ethereum

3.3.1 What is Ethereum

Ethereum is a digital currency, designed on a block chain open platform, a decentralized application that can be used by anyone. Ethereum cannot be controlled similar to BITCOIN. Ethereum platform was globally designed by many people. However, Ethereum has a strong protocol unlike BITCOIN, in which the user can create an application or use the tool safely with the introduction of Homestead. It is designed to be flexible and adaptable by a large audience.

In addition, Ethereum is said to be the next generation technology based on block chain, originally designed by Satoshi Nakamoto, a Peer-To-Peer Electronic Cash System, named BITCOIN. The block chain is a computer structured framework in which each node of network is executed and recorded similar transactions that are grouped within blocks. One block is added at a time where each block contains a proof of mathematics that verifies the chain, linked to earlier block. The transactions of the user are protected by a strong cryptography.

3.3.2 Blockchain

3.3.3 Ether

mining

3.3.4 Transaction Costs & Gas

3.3.5 Smart Contracts

solidity

3.3.6 Ethereum Clients

geth

3.3.7

3.4 System Design

References

[1] Pierre Noizat. *Blockchain Electronic Vote*. 2016. URL: http://www.the-blockchain.com/docs/blockchain-electronic-vote.pdf (visited on 17/11/2016).

[2] Adam Ernest. *The Key To Unlocking The Black Box: Why The World Needs A Transparent Voting DAC*. 2014. URL: https://followmyvote.com/wp-content/uploads/2014/08/The-Key-To-Unlocking-The-Black-Box-Follow-My-Vote.pdf (visited on 15/12/2016).

[3] Willem Wyndham, Spencer Chen and Saurav Das. *Proposal For Secure Electronic Voting*. 2016. URL: http://www.economist.com/sites/default/files/maryland_cyber_ctr.pdf (visited on 15/12/2016).

[4] Safevote. *Voting System Requirements*. 2001. URL: http://www.thebell.net/papers/vote-req.pdf (visited on 19/12/2016).

[5] Briony Holmes. *Blockchain voting systems offer transparency and security Brave New Coin*. 2016. URL: http://bravenewcoin.com/news/blockchain-voting-systems-offer-transparency-and-security/ (visited on 16/12/2016).

[6] 2016. URL: http://www.ncsl.org/research/elections-and-campaigns/voting-system-standards-testing-and-certification.aspx (visited on 17/12/2016).

[7] Sabina Petride. *Security Properties for Electronic Voting*. 2016. URL: http://www.cs.cornell.edu/courses/cs513/2002sp/proj.00.StuSolns/sp2580.htm (visited on 18/12/2016).

[8] Alex Escala et al. *Universal Cast-as-Intended Verifiability*. 2015. URL: https://fc16.ifca.ai/voting/papers/EGHM16.pdf (visited on 21/12/2016).

[9] Eli Ben-Sasson et al. *Zerocash: Decentralized Anonymous Payments from Bitcoin*. 2014. URL: http://zerocash-project.org/media/pdf/zerocash-oakland2014.pdf (visited on 21/12/2016).