Scientific article

'Ballots and Bitcoins'

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.

29th December 2016

1 Abstract

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. 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, thus potentially limiting voter participation, or cast doubt upon the published output of an election.

To provide an immutable, verifiable and secure online voting system I intend to leverage the availability of the Bitcoin Blockchain as a secure transaction database. From this public ledger, voters will be able to independently audit the inclusion of their vote, and the outcome of the election as a whole, while being sure that the results cannot be changed due to the immutability of the Blockchain.

---

2 The three "B's" of Bitcoin

When discussing Bitcoin, it is important to distinguish exactly which technology we are referring to. When broken down to its most primitive level, Bitcoin can be viewed as three separate innovations; The Big "B", the "Blockchain" and the Little "b" [1][2].

2.1 The Big "B"

The Big "B" Bitcoin, that is the singular capitalized term, is referring to the protocol, software, and community of the decentralized computer network [3]. This was originally outlined in a white paper [4] authored under the pseudonym Satoshi Nakamoto in November of 2008 and was quickly followed by an open source release of the Bitcoin proof-of-concept source code in January 2009 [5]. This proof-of-concept software client has been periodically updated (about 75% of the code has been rewritten in the six years) since it was initially released, remains open source and serves as the reference software for individuals seeking to develop Bitcoin-related software (now commonly known as "Bitcoin Core") [6]. This reference client incorporates all improvements or adjustments made to the underlying Bitcoin protocol so any developer creating software adaptations can know that if their software is compatible with systems running the reference client, it will be compatible with the Bitcoin Network as a whole [1].

Prior to bitcoin, there were a number of digital currencies with varying levels of traction (Liberty Reserve [7] and E-Gold [8] being the most prominent) which offered secure payments without revealing account numbers or identities. However, in reality these were centralized services with non-public transaction histories that relied on the central issuer to verify transactions. The Bitcoin protocol describes a decentralized system with no hierarchical structure which is progressively built [9] by a community of contributors. This network exerts a tremendous amount of computing power toward the singular purpose of validating and clearing transactions on the Bitcoin Network.

It is this decentralised, public ledger which is arguably the most powerful innovation to come out of the Bitcoin specification and it is the Bitcoin protocol which sets out the rules for maintaining it. The Bitcoin protocol is, therefore, a solution to the 'Byzantine General's problem [10] which outlines the difficulties in a system generating a consensus and prior to Bitcoin was thought to be unsolvable. We now have a technology which provides with an immutable record of transactions without the need for a central gatekeeper controlling how transactions are recorded [11].

This decentralization has tangible benefits besides immutability as the fact that the protocol is operated by a group of peers means there is no central point of failure which negates a number of potential problems. The first of these being internal abuse, that is, this decentralized system removes the need to trust any central authority to 'do the right thing'. This is best demonstrated when considering the cryptocurrency side of bitcoin. Many fiat currencies have experienced hyperinflation due to a government printing money, for example Zimbabwe in 2008 [12] where inflation reached over 3.5Million%. In contrast, the decentralized Bitcoin network forms a consensus on the difficulty of the mathematical challenge used to confirm the next set of transactions every 2,016 blocks [13]. This is akin to setting the rate at which new bitcoins are created and therefore reduces the risk of internal abuse causing hyper inflation as a majority consensus is needed to alter the rate.

Another problem reduced by decentralization is that of attack. As a majority consensus is needed for software updates to be propagated [12][14] it becomes prohibitively difficult for an attacker to compromise the underlying protocol. Nodes 'vote' on whether to accept protocol changes and an attacker would need a significant percentage of computational power to have a chance of dictating what gets implemented.

A similar case can be made for censorship, A majority vote is required for changes to be applied, potential changes which are not in the interest of the many will likely be rejected. This results in a more open system and stops a single controlling entity dictating what can and cannot be done (such as PayPal restricting its use in gambling-related ventures due to government intervention [15]).

2.2 The Blockchain

The second "B", the Blockchain, represents the second greatest innovation from Nakamoto. This is the distributed ledger which underpins the entirety of the Bitcoin

system. A distributed ledger is a consensus of replicated, shared, and synchronized digital data geographically spread across multiple sites, countries, and/or institutions [16]. This ledger is stored locally on every node in the network which is running the full version of the bitcoin software [6] and records every transaction sent and confirmed on the Bitcoin network (the current size of the Blockchain is around 92GB, December 2016 [17]). This complete history, coupled with the fact that it is an open network means that anyone can see what is happening in the network, not just now but during all periods in the past. This is extremely powerful as it allows an individual to fully audit the entire Bitcoin Blockchain without relying on external parties. This process is, in fact, what happens when you first download the full version of Bitcoin Core [13].

While the Bitcoin Blockchain is not the only form of distributed ledger in existence, it is the most publicly proven method to achieve distributed consensus and does this via the 'proof-of-work mining' process [16]. This is how new information gets added to the blockchain, by nodes in the network running a special 'mining' variant of the Bitcoin software which uses considerable computing resources to win the right to add another block to the Blockchain which is accompanied by a reward for the winning user. The concept of 'proof-of-work' is a method of ensuring that the information being added to the Blockchain was difficult (in terms of cost and time) to be made, though is easy for others to validate that the requirements were met [18]. This means that the expenditure of computing power serves to secure the integrity of the Blockchain, while the miners themselves verify through public-private key cryptography the validity of each transaction they include in a block.

Blocks are chained together making it impossible to modify transactions included in any one block without modifying all following blocks; as a result, the cost to modify a particular block increases with every new block added to the block chain, magnifying the effect of the proof of work [13][19]. This is why, although a bitcoin transaction is deemed clear upon its inclusion in a block on the Blockchain, best practices dictate that a user considers a transaction confirmed after its inclusion in a block and the addition of five subsequent blocks to the Blockchain [20].

The difficulty of the proof-of-work mining needs to be controlled, so that an average mining time of 10 minutes per block is maintained. This time is somewhat arbitrary but is an attempt to find a balance between accepting transactions quickly and minimizing instability and waste in the network, as, while a new block is being distributed other miners will be working on an obsolete problem. As more miners join the network the block creation rate will increase due to the greater collective computational power. Therefore, every 2,016 blocks the difficulty of the mathematical challenge is recalculated so that the average mining time returns to normal [13][18].

Despite the media often suggesting that bitcoin is an anonymous payment system, the Blockchain is in fact a transparent record of all user transactions on the network. Bitcoin is in fact pseudonymous, and your transactions in the network are like writing under a pseudonym. If an author's identity is ever linked to their pseudonym then everything written under that pseudonym will be revealed [21]. This is particularly poignant when considering the Blockchain as every transaction is stored forever, therefore a compromised address could lead to all transactions being linked to a person. There are however ways to reduce the amount of statistical analysis which can be done on transactions that a person is a part of which help to achieve reasonable anonymity.

2.3 The little "b"

The third "b", bitcoin, refers to the Bitcoin Network's payment system where the lowercase version of bitcoin refers to the core unit of value on the Bitcoin network [1]. New bitcoins enter the network as a reward for miners spending resources to confirm transaction blocks. In total 21 million bitcoins will be created as mining rewards, over 70% of all bitcoins have been mined to date and approximately 90% will have been mined by 2026 [1]. A bitcoin can be subdivided to eight decimal places, with the smallest unit – a satoshi – having a value of 1/100,000,000th of a bitcoin.

The question of 'what is the value of a bitcoin' is somewhat a controversial topic. Bitcoin resembles a currency in many ways; there will only ever be 21 million bitcoins that will exist in the bitcoin network and every bitcoin can be divided to eight decimal places, so there are many partial bitcoins that can exist [22]. Historically, currencies have been backed by a tangible asset, for example gold, but it is the Bitcoin network that gives bitcoins value. Therefore bitcoin should not be considered a currency, but an open ledger within which there is value due to the networks ability to verify the authenticity and ownership of a bitcoin and the ability to transfer possession nearly instantaneously for little to no cost without the reliance on a trusted third party [1].

In general, you must spend some bitcoins to broadcast a transaction into the network and for it to be included in the Blockchain.

Transactions typically include a payment of nominal amount of bitcoins as a "miners fee", usually around 0.0001 bitcoins, but this is optional. However, inclusion of a miners fee will likely prioritize a transaction for inclusion in the next solved block as miners will want to maximise their reward [1]. As there are a finite number of bitcoins in the network, these fees will play a progressively larger incentive in ensuring there are miners available to continue confirming transactions on the network.

Transactions will propagate across the Bitcoin network and be visible (in an unconfirmed state) in a matter of seconds. However, transactions may not be included in the next available block in a number of situations. The transaction may not have been received by the miner that solves the block, the block may have already been filled with other, higher priority unconfirmed transactions or

the miner may have elected not to include the unconfirmed transaction [1].

---

3 Bitcoin, a deeper dive

In order to construct a voting system on top of Bitcoin there are several areas which we need to understand in detail. The below sections describe these areas in greater depth.

3.1 Transactions

A transaction is a transfer of Bitcoin value that is broadcast to the network and collected into blocks. This transaction typically references previous transaction outputs as new transaction inputs and dedicates all input Bitcoin values to new outputs [23], however, these transactions do not simply consist of a from and to address. Transactions that can be recorded in the bitcoin ledger have a lot of expressive power built into them in the form of a scripting language which allows for complex conditions to be included allowing for contracts to be easily built on top of Bitcoin [24].

This script is essentially a list of instructions recorded with each transaction that describe how the next person wanting to spend the Bitcoins being transferred can gain access to them. Scripting provides the flexibility to change the requirements of what's needed to spend a bitcoin, for example, you could require two private keys to unlock a transaction [25]. You can invisage every bitcoin transaction as a small program which describes under which conditions the transaction is valid, a transaction is considered valid if the combination of the input and output scripts return a boolean, True. The scripting language is stack-based, processed from left to right and is purposefully not turing complete, with no loops (ensuring these scripts will halt). Each transaction consists of two parts, an Input and Output section.

An input contains a reference to an output from a previous transaction (often multiple transactions are listed so that the input values add to the required output value) along with a scriptSig which is the first half of the transactions script. This scriptSig contains two components, a signature and public key, where the public key matches the hash given in the script of the redeemed output and the signature (combined with the public key) proves the transaction was created by the real owner [23].

An output contains the required instructions to send bitcoins. This contains a Value (in Satoshis) which will be the transactions worth when claimed and must be fully spent (as each output transaction can only ever be referenced once to redeem). This means it's common to send 'change' back to an address you own as part of this transaction, though any bitcoins not redeemed are considered as a transaction fee which can be claimed by whoever validates the block in the Blockchain. The output also contains a scriptPubKey, which is the second half of the transactions script.

The scripts included in the input and output sections combine to form the validation script and are evaluated in the order of scriptSig, scriptPubKey, with scriptPubKey using the values left on the stack from scriptSig. Virtually any script can be included in a transaction but miners will only accept standard scripts for security reasons. Bitcoin currently uses five (as of Bitcoin Core 0.10) different scriptSig/scriptPubKey pairs, these are the Pay-to-PubKeyHash (P2PKH), Pay-to-ScriptHash (P2SH), Pay-to-Public-Key (P2PK), Multisignature and `OP_RETURN` transaction types, of which the first one is the most commonly used [26][27].

A typical P2PKH Bitcoin address looks as follows, 15Cytz9sHqeqtKCw2vnpEyNQ8teKtrTPjp, and by having this address we can write an output script which will send coins to it. Conversely by having the private key for this address we can write an input script that is able to spend these coins. A P2PKH input script (the scriptSig) contains only the signature and public key (no other OPCODES) which will then be pushed onto the stack. An example P2PKH scriptSig looks as follows: `<signature> <pubKey>`. A P2PKH output script (the scriptPubKey) contains a pubKeyHash (the hash of the bitcoin address you wish to pay to) along with a set of OPCODES to verify the transaction. An example P2PKH scriptPubKey looks as follows: `OP_DUP OP_HASH160 <pubKeyHash> OP_EQUALVERIFY OP_CHECKSIG` [27][23].

When combined in a transaction they form the validation script which must evaluate to true for the transaction to be valid. The checking process for the above scriptSig and scriptPubKey examples can be seen in 'Table 1' [26][23]:

Table 1: The script checking process

Stack	Script	Description
empty	<sig> <pubKey> OP_DUP OP_HASH160 <pubKeyHash> OP_EQUALVERIFY OP_CHECKSIG	Validation script is the combined scriptSig and scriptPubKey
<pubKey> <sig>	OP_DUP OP_HASH160 <pubKeyHash> OP_EQUALVERIFY OP_CHECKSIG	First two values are pushed onto the stack
<pubKey> <pubKey> <sig>	OP_HASH160 <pubKeyHash> OP_EQUALVERIFY OP_CHECKSIG	Top of the stack is duplicated
<pubKeyHashAnswer> <pubKey> <sig>	<pubKeyHash> OP_EQUALVERIFY OP_CHECKSIG	Top of the stack is hashed, result pushed onto stack
<pubKeyHash> <pubKeyHashAnswer> <pubKey> <sig>	OP_EQUALVERIFY OP_CHECKSIG	Public key value is pushed onto the stack
<pubKey> <sig>	OP_CHECKSIG	Checks equality between the top two stack items.
True		Signature is checked between the top two stack items

3.2 Validated transactions

Due to the distributed nature of the Bitcoin Network, the prefered method of transmitting new data across the network is peer-to-peer. This has the drawbacks of less efficiency, as data is replicated many times and updates need to be sent to each node in the network, but results in each peer being more independent and thus the network as a whole being more robust (as there is no central server which can be controlled or closed down) [14].

In peer-to-peer models, even if all peers can be 'trusted', there can be the problem of consensus of what is the 'real' state of the data' say if each peer is operating at different speeds. In the Bitcoin network we cannot afford to trust any of the other peers as a malicious peer could seek to gain the system to their own ends. How, therefore, can all nodes agree which transaction blocks are genuine and should be accepted to build future transactions on.

In short, it is the voting system (briefly outlined earlier) that shows the majoritys support and fixes the transaction order in the Blockchain. This is done via the 'proof-of-work' voting system which ensures voting has a cost, computing resources and time, which makes it infeasible for a single anonymous voter to sway the vote. This vote therefore acts as a form of 'proof' to the rest of the world of the majorities will, because the solution could not be generated without the majorities computing resources [16]. The 'proof-of-work' challenge itself comes from the world of cryptography, specifically, the goal is to find a hash input thats smaller than a given target. The smaller the target, the fewer number of outputs that fulfil the equation and the harder it is to find a successful input.

In order for transactions to become validated, new transactions are forwarded around the network and placed into a pool of unconfirmed transactions. These are not considered 'accepted' yet but are available for all to see almost instantaneously. Miners draw from this pool to create a candidate next set of transactions to be officially accepted which will form the next block. The full text of all of these candidate transactions, along with the hash of the previous block and a nonce, are input into the the hash function (SHA256) and miners will try different values for the nonce until the resulting hash is below a certain value. Because its a cryptographic hash, theres no way to find a nonce that satisfies the output hashs condition other than attempting to guess [13]. At this point, all of the miners are in a competition to find the hash first, each with a potentially different set of transactions to confirm. Once a miner succeeds they announce their solution to the rest of the network, their block becomes the next block in the Blockchain, and the transactions therein become confirmed. This strategy means that one miner will choose the next set of confirmed transactions, but the hash function effectively makes the miner a random one. All other mines then validate this new block, and the transactions held within, and can choose to accept it and start work on the next block. It's important to note the properties of the hash function being used which, although requiring many billions of hashes to compute (the network hash rate was around 2,400,000 trillion hashes per second in January 2015 [28]), allowing trivial verification of the hash. As the new block contains the hash of the previous block, this forms a chain of confirmed blocks securing the order of the transactions held within.

Occasionally, two miners may find a solution to the problem at the same time creating two potential next blocks in the chain. When miners produce simultaneous blocks at the end of the block chain, each node individually chooses which block to accept, this is usually the first block they see. Eventually another miner finds the solution to another block which attaches to only one of the competing blocks. This makes that side of the fork stronger and, as the general consensus is to use the strongest chain, other

nodes will switch to this longer Blockchain [13]. While this is statistically unlikely to happen, it is even more unlikely for the subsequent blocks to be solved at the same time, meaning that one fork will grow quicker than the other and the fork will resolve itself quickly. Transactions that were in the fork that wasn't chosen are not lost and are placed back into the unconfirmed transactions pool [29]. The fact that the end of the chain can be forked and rearranged means you shouldn't trust transactions at the end of the chain as much as ones further back. In Bitcoin, a transaction is not considered confirmed until it is part of a block in the longest fork, and at least five blocks follow it. In this case we say that the transaction has "6 confirmations". This gives the network time to come to an agreed-upon the ordering of the blocks [30].

3.3 Multisignature transactions

Multisignature (also called multisig) refers to the requirement of more than one key being present to authorise a transaction. Bitcoin has had an alternative to single-key addresses since early 2012 [31], when a new type of address called Pay-to-Script-Hash (P2SH) was defined and standardized. A typical P2SH address looks like 347N1Thc213QqfYCz3PZkjoJpNv5b14kBd. Note that P2SH addresses always begins with a '3', (instead of a '1' as in P2PKH addresses), because P2SH addresses have a version byte prefix of 0x05, which resolves as a '3' after base58check encoding [32]. A P2SH address can support arbitrary sets of N keys, any M of which are required to transact. This is commonly referred to as `'M-of-N'` [33].

There are many advantages to using multisignature transactions. Firstly, we could eliminate single points of failure by ensuring that keys are stored on completely separate devices in an `2-of-2` address. This effectively creates 2 factor authentication for your funds. For example, if one wallet is on your primary computer and the other on your smartphone. The funds cannot be spent without a signature from both devices. Thus, an attacker must gain access to both devices in order to steal your funds which is much more difficult than one device [31]. If a third key was added to the this scenario in a `2-of-3` address then the user could lose either device and still be able to access their funds using the remaining device along with the third offline key.

We can also address the problem of access control, for example, to funds of an organization. Trusting all of the funds to one key (perhaps in control of the CEO) is both dangerous, as anyone who knows the key could potentially steal the money with no way to trace it; and cumbersome, as the individual holding the key would be required to approve each transaction. Alternatively you could give more people access to this key, though this then increases the risk of the first scenario or of one of the parties being compromised and the key stolen. You can imagine an n-of-m address being very useful in this situation, as you could give one key to a department and one to a central policy controller. If a department then wishes to spend funds they must seek the approval of the central controller who can either approve or deny in accordance with company policy.

This also opens the possibilities of new transactions which can be unlocked using multisig technology. For example, we could perform a trustless escrow transaction using a `2-of-3` multi signature address, between two parties (Alice and Bob) and a trusted third party who could adjudicate a dispute (Charlie). Bob creates a P2PH address using the public keys of the three parties which Alice then pays the money to. After confirming payment has been sent, Bob sends the item and, if the transaction goes smoothly, when Alice receives the item Bob can construct an Output transaction which he will then send to Alice to release the funds. Note that there was no need to involve the arbitrator, Charlie, during the entirety of the undisputed transaction. At any point during the transactions process either party can enlist the help of Charlie to resolve a dispute. After evaluating evidence from both sides Charlie can then release the funds as he sees fit using his key along with either party [29][33].

P2PH was designed so that, rather than having senders enter long scripts into their scriptPubKey, they would allow a hash of their spending conditions instead. These spending conditions are known as a 'redeem script' and allows the sender to use a concise 34-character address (the hash of the redeem script) rather than a long unwieldy one containing full details of the spending conditions [32]. The redeem script is only revealed during the spending transaction, when it is checked against the redeem script hash which puts the responsibility for providing the redeem script on the recipient of the funds. This has the added benefit of the sender being able to fund any arbitrary transaction without knowing what the spending conditions are, or indeed, any of the public keys involved [32].

3.4 Security justification

With ever increasing quantities of money being transferred over the Bitcoin network, and the scope of this project aiming to design a secure voting system built on top of it; it's essential to confirm the level of security and to understand the risks involved.

Bitcoin is often called a cryptocurrency and, as the name suggests, cryptography is a central part of the protocol. Bitcoin makes use of two hashing functions, SHA-256 and RIPEMD-160, but it also uses Elliptic Curve DSA on the curve secp256k1 to perform signatures [34].

Signatures in bitcoin are performed exactly the same as in conventional public-key cryptography. We can verify that Alice is the owner of a transaction because she has signed it with her private key, which can then be verified with her public key. Bitcoin uses the secp256k1 elliptic curve for generating the key pairs which appears to have been chosen for possible speed optimizations in the future [34]. Attacks here could involve breaking the underlying elliptic curve cryptography either by solving the discrete logarithm problem (which could be possible

with quantum computers, but there is currently no efficient non-quantum algorithm), or by finding vulnerabilities in the specific curve chosen.

As with most public-key cryptography, Bitcoin does not sign the entirety of the message as this would be too expensive. Instead, it signs a hash which can then be checked against the message to verify its integrity. If we could break the hash function, it could be possible to generate a secondary input transaction to hash to the same value as an original. This could then be used to perform a signature replay attack to forge a message. Note: this would not be possible if a unique address is used for every transaction (as suggested in the Bitcoin specification) as there would be signiture to reply [34].

The likelihood of either of these two scenarios being immediate threats is however, relatively low. Bitcoin uses industry standard technology and a breach in any of the underlying cryptography would undermine a large proportion of the security we enjoy on the internet.

Despite the underlying cryptography being secure, there are still a number of possible attacks which could be utilised against the network. While a dishonest miner cannot make bitcoins (illegitimately), steal bitcoins or make payments on your behalf (pretend to be you), they could delay or refuse the relaying of valid transactions to other nodes, attempt to create blocks which exclude specific transactions of their choosing or attempt to create a longer blockchain that would render previously accepted blocks invalid [14]. However all of the above require the attacker to have sufficient block creation power, effectively a 51% attack on the network. When an individual or group owns more than half of the network they could produce enough "computational work" to convince others that their blockchain is the best choice [34]. The bitcoin network was nearly exposed to such an attack in January of 2014 as the mining group *Gash.io* started to approach 50% of the mining power of the entire network. At one point the group had collectively solved 42% of the blocks in a 24 hour period [35]. The situation was resolved without incident, due to miners leaving *Ghash.io* for smaller pools, as well as the pools own decision to stop accepting new miners [34].

---

4 Electronic Voting Systems

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 [36]. 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 [37] 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 [38]. 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; and `(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 [39].

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 [38]. 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 [40]. 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 [41]. 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 cor-

rect 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 [41]. 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 [42].

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, (as discussed previously) 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.

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 [36] 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 favor 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 [36]. 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 [43] 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 whos 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 noninteractive 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 Bitcoin (e.g. zCash whitepaper [44], 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.

References

[1] KAYE SCHOLER. *An Introduction to Bitcoin and Blockchain Technology*. 2016. URL: http://www.kayescholer.com/docs/IntrotoBitcoinandBlockchainTechnology.pdf (visited on 02/12/2016).

[2] itBit. *Five-Minute Bitcoin Primer: What is Bitcoin and How Does it Work?* 2015. URL: https://cdn2.hubspot.net/hub/424565/file-2382046756-pdf/itBit_Five-Minute_Bitcoin_Primer_-_What_is_Bitcoin_and_How_Does_it_Work.pdf?t=1446765105287 (visited on 02/12/2016).

[3] 2016. URL: https://en.bitcoin.it/wiki/Help:Introduction#Capitalization_.2F_Nomenclature (visited on 02/12/2016).

[4] Satoshi Nakamoto. *Bitcoin: A Peer-to-Peer Electronic Cash System*. 2008. URL: https://bitcoin.org/bitcoin.pdf (visited on 02/12/2016).

[5] Satoshi Nakamoto. *Bitcoin v0.1 released*. 2009. URL: http://www.metzdowd.com/pipermail/cryptography/2009-January/014994.html (visited on 02/12/2016).

[6] 2009. URL: https://github.com/bitcoin/bitcoin (visited on 02/12/2016).

[7] 2014. URL: http://www.investopedia.com/terms/l/liberty-reserve.asp (visited on 03/12/2016).

[8] Douglas Jackson. *e-gold is...* 2007. URL: http://blog.e-gold.com/2007/08/e-gold-is.html (visited on 03/12/2016).

[9] Simon Thorpe. *All you want to know about Blockchain but were afraid to ask*. 2016. URL: https://www.linkedin.com/pulse/all-you-want-know-blockchain-were-afraid-ask-simon-thorpe (visited on 03/12/2016).

[10] Leslie Lamport, Robert Shostak and Marshall Pease. *The Byzantine Generals Problem*. 1982. URL: http://research.microsoft.com/en-us/um/people/lamport/pubs/byz.pdf (visited on 03/12/2016).

[11] Dug Campbell. *The Byzantine Generals' Problem - dugcampbell.com*. 2015. URL: http://www.dugcampbell.com/byzantine-generals-problem/ (visited on 03/12/2016).

[12] 2015. URL: https://www.rt.com/business/267244-zimbabwe-currency-compensation-hyperinflation/ (visited on 03/12/2016).

[13] 2016. URL: https://bitcoin.org/en/developer-guide#proof-of-work (visited on 03/12/2016).

[14] Brave New Coin. *A gentle introduction to blockchain*. 2016. URL: http://www.the-blockchain.com/docs/A-gentle-introduction-to-blockchain-technology-web.pdf (visited on 03/12/2016).

[15] 2016. URL: https://www.paypal.com/selfhelp/article/FAQ915 (visited on 03/12/2016).

[16] 2016. URL: http://www.blockchaintechnologies.com/blockchain-definition (visited on 03/12/2016).

[17] 2016. URL: https://blockchain.info/charts/blocks-size?timespan=all (visited on 03/12/2016).

[18] 2016. URL: http://www.blockchaintechnologies.com/blockchain-mining (visited on 03/12/2016).

[19] 2016. URL: http://redpinata-development.com/bitcoin-academy/index.php/reader/items/proof-of-work.html (visited on 07/12/2016).

[20] 2016. URL: https://en.bitcoin.it/wiki/Confirmation (visited on 03/12/2016).

[21] 2016. URL: http://bitcoinsimplified.org/learn-more/anonymity/ (visited on 04/12/2016).

[22] itBit. *Bitcoin, Blockchain, and the Future of Financial Transactions*. 2015. URL: http://www.the-blockchain.com/docs/Bitcoin,%20Blockchain,%20and%20the%20Future%20of%20Financial%20Transactions.pdf (visited on 02/12/2016).

[23] 2016. URL: https://en.bitcoin.it/wiki/Transaction (visited on 05/12/2016).

[24] Albert Wenger. *Bitcoin As Protocol — Union Square Ventures*. 2013. URL: https://www.usv.com/blog/bitcoin-as-protocol (visited on 05/12/2016).

[25] 2016. URL: https://en.bitcoin.it/wiki/Script (visited on 05/12/2016).

[26] Davide De Rosa. *Standard scripts*. 2015. URL: http://davidederosa.com/basic-blockchain-programming/standard-scripts/ (visited on 05/12/2016).

[27] Roland Kofler. *Thinking in Transactions*. 2014. URL: https://bitcoinmagazine.com/articles/thinking-transactions-1401650873 (visited on 05/12/2016).

[28] 2016. URL: https://en.bitcoin.it/wiki/Hash_per_second (visited on 06/12/2016).

[29] Scott Driscoll. *Introduction to Bitcoin and Decentralized Technology — Pluralsight*. 2016. URL: https://app.pluralsight.com/library/courses/bitcoin-decentralized-technology/table-of-contents (visited on 30/11/2016).

[30] Michael Nielsen. *How the Bitcoin protocol actually works*. 2013. URL: http://www.michaelnielsen.org/ddi/how-the-bitcoin-protocol-actually-works/ (visited on 06/12/2016).

[31] 2016. URL: https://en.bitcoin.it/wiki/Multisignature (visited on 07/12/2016).

[32] Soroush Pour. *Bitcoin multisig the hard way: Understanding raw P2SH multisig transactions*. 2014. URL: http://www.soroushjp.com/2014/12/20/bitcoin-multisig-the-hard-way-understanding-raw-multisignature-bitcoin-transactions/ (visited on 07/12/2016).

[33] Ben Davenport. *What is Multi-Sig, and What Can It Do? — Coin Center*. 2015. URL: https://coincenter.org/entry/what-is-multi-sig-and-what-can-it-do (visited on 07/12/2016).

[34] Edward Z Yang. *The Cryptography of Bitcoin : Inside 736-131*. 2011. URL: http://blog.ezyang.com/2011/06/the-cryptography-of-bitcoin/ (visited on 07/12/2016).
[35] Alec Liu. *Bitcoin's Fatal Flaw Was Nearly Exposed*. 2014. URL: http://motherboard.vice.com/blog/bitcoins-fatal-flaw-was-nearly-exposed (visited on 07/12/2016).
[36] Pierre Noizat. *Blockchain Electronic Vote*. 2016. URL: http://www.the-blockchain.com/docs/blockchain-electronic-vote.pdf (visited on 17/11/2016).
[37] 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).
[38] 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).
[39] Safevote. *Voting System Requirements*. 2001. URL: http://www.thebell.net/papers/vote-req.pdf (visited on 19/12/2016).
[40] 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).
[41] 2016. URL: http://www.ncsl.org/research/elections-and-campaigns/voting-system-standards-testing-and-certification.aspx (visited on 17/12/2016).
[42] 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).
[43] Alex Escala et al. *Universal Cast-as-Intended Verifiability*. 2015. URL: https://fc16.ifca.ai/voting/papers/EGHM16.pdf (visited on 21/12/2016).
[44] 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).