LEDGER
ledgerjournal.org

ISSN 2379-5980 (online)
DOI 10.5195/LEDGER.2021.199

RESEARCH ARTICLE

Enhancing Electronic Voting with a Dual-Blockchain Architecture

Kees Leune,[*] Jai Punjwani[†]

Abstract. Voting is one of the most fundamental aspects of democracy. Over the past few decades, voting methods around the world have expanded from traditional paper ballot systems to electronic voting (e-voting), in which votes are written directly to computer memory. Like any computer system, voting machines are susceptible to technical vulnerabilities that open up opportunities for hackers to tamper with votes, causing the use of electronic voting technology to raise concerns about ballot security. We describe how electronic voting can be supported by blockchain technology to ensure voter secrecy, vote correctness, and equal voting rights. In this paper, we present a system using two separate blockchains, each with separate transactions and consensus algorithms. We describe a prototype implementation that validates our ideas by executing several proof-of-concept simulations of a range of voting scenarios.

1. Introduction

Voting is one of the most fundamental aspects of democracy.[1] Over the past few decades, voting methods around the world have expanded from traditional paper ballot systems to electronic voting (e-voting), in which votes are written directly to computer memory.

Several approaches to computer-supported voting have emerged over time, ranging from systems in which the vote is cast on a paper ballot, which is scanned and processed electronically, to systems that support a fully electronic vote across networks.

E-voting is any form of voting that is supported by electronic machines and takes place in specially designated physical locations, commonly known as polling stations.[2] The polling station is a location to which a voter must travel in order to cast their vote. E-voting is separate and distinct from internet voting (i-voting), in which the complete election process takes place online.[3] Internet voting takes place via networks that are beyond the control of a voting authority. Park, Specter, Narula, and Rivest also point out that any online vote-casting system is vulnerable to catastrophic failures from vastly larger-scale, harder-to-detect, and easier-to-execute attacks than would be possible against paper-ballot-based alternatives.[4] While most ideas presented in this research likely apply to i-voting as much as they do to e-voting, we have not performed any testing in an i-voting scenario. Consequently, i-voting is purposefully kept out-of-scope in this paper.

[*] K. Leune (leune@adelphi.edu) is a faculty member in the Department of Mathematics and Computer Science at Adelphi University.
[†] J. Punjwani (jaipunjwani@mail.adelphi.edu) is an alumnus, who graduated from Adelphi University with a B.S. in Computer Science in 2018.

The use of Direct Recording Electronic (DRE) machines has given rise to concerns regarding their safety and security,[5,6] yet e-voting adoption continues to increase. Recent elections have shown us how much voting systems have to improve before they support democratic elections that are genuinely secure and trustworthy.[7]

One of the key issues with using electronic machines for recording and tallying votes is the possibility for abuse. For good reasons, many voters distrust technology out of fear that it might be tampered with. To address such concerns, and to ensure confidence in election outcomes, it is important to design and build systems that leave a voter-verifiable audit trail.[8] Several approaches have been proposed.

One approach equips DRE machines with a printer to print a voter-verified paper ballot. Voters inspect the printout, which is displayed behind a transparent screen, to make sure that it agrees with the selections that were made electronically, and presses a button if the paper ballot is correct. It is then dropped automatically into a ballot box.[9] Another approach proposes optical-scan machines, in which forms are filled out by hand, and then scanned and counted by machine. In both approaches, the goal is to retain a paper-based audit trail, which can be inspected when the correct functioning of technology is contested.

In line with the ideas presented by Park *et al.*, we strongly support maintaining a paper-based audit-trail, but we propose a mechanism that will add a transparent, immutable electronic audit trail as well.

In 2017, Rivest and Stark wrote that any given voting system should not only produce the correct election outcome, but also produce evidence sufficient to convince losing candidates and their supporters that they lost the election fair and square.[10] This evidence must convince the broader public as well, else we risk fostering mistrust both in the machinery of our democracy and in election outcomes. Such mistrust would engender apathy toward elections—or worse, a belief that changes in power should be effected by other means. Current events in the United States revolving around the 2020 presidential election clearly illustrate how important it is to have a voting system that is both auditable and audited.

Any voting process must adhere to strict requirements, and computer-based processes are no exception to that. Requirements, such as the ones proposed by the European Council, include authentication, equal voting rights, integrity, voter secrecy and anonymity, and election fairness.[11]

The authentication requirement imposes the need to uniquely identify voters, and to exclude any voter who cannot be identified, or whose eligibility to vote cannot be determined. The equal voting rights requirement ensures that voters can only cast the appropriate number of votes, and that all votes that are cast are included in the election result. The voting system should be designed so that the voter's intention is not affected by the voting system or by any undue influence. Furthermore, the system must warn a voter when they are about to cast an invalid vote. The voter must be able to verify that undue influence on votes can be detected throughout the voting process, and that the vote remains confidential at all times.

In the United States, no universal standard for designing and building voting machines exists, leading to variations between vendors and manufacturers. This results in vastly different design choices concerning underlying technology, and fundamentally different auditing processes from state to state. The lack of availability of a reference architecture leads to several issues.

Like in any computer-based system, electronic voting systems are susceptible to any number of technical vulnerabilities that open up opportunities for tampering with votes without sufficient detection. One concern related to the introduction of computers to the voting process is the ability to commit electronic voting fraud. As such, it is not surprising that the use of electronic voting technology raises concerns about ballot security.[12]

The goal of this research is to determine how electronic voting can support the integrity and transparency of the voting process without sacrificing security of the ballot. We ask the following research question: in what way can an electronic voting process provide election integrity while maintaining voter secrecy?

2. Enhancing the Electronic Voting Process

To answer our research question, we make several assumptions, as shown in Table 1. The first is that we presume that all voters can be unambiguously identified, and that their eligibility to vote can be determined. The mechanism by which this is done is not relevant, as long as an authoritative voter roll can be produced, and as long as voters can be identified immediately prior to casting their votes.

Table 1. Key assumptions

Assumption 1	Voters can be unambiguously identified
Assumption 2	Voter eligibility can be unambiguously determined
Assumption 3	Voting machines ensure only correct votes are cast
Assumption 4	Voting machines are tamper-proof
Assumption 5	A reliable and secure message-broadcast system is in place

Another assumption is that a reliable and secure message broadcast mechanism exists and is available for use.

Furthermore, we limit our research to proposing an architecture for back-end and infrastructure components of a distributed electronic voting system. We assume that the individual machines that are used to interact with this infrastructure have tamper-proof interfaces that are designed to prevent a voter from casting an invalid ballot. Hjálmarsson *et. al.* postulate that voters will have to vote in a supervised environment to satisfy the privacy and security requirements for e-voting, and to ensure that the election system should not enable coerced voting.[13] We follow their guidance in this matter.

Based on these assumptions, several additional architecture goals were formulated, as identified in Table 2. First, we set out to design a system that ensures transparency and accountability by maintaining an immutable audit trail of all transactions that take place. The audit trail is subject to inspection by election authorities at any time, and can be used to identify anomalies. Election authorities may also consider making the audit trail available to the public once the election has been completed.

Second, our system must enforce that no voter will cast more votes than they are eligible to (*i.e.,* equal voting rights), while respecting the need for a secret ballot (*i.e.*, it cannot be determined how any identifiable individual voter cast their ballot).

Third, the system must maintain the integrity of the vote. Once recorded, it must not be possible to alter the vote without detection, and any altered votes must not be counted towards the final tally.

Table 2. Architecture Objectives

Objectives 1	Voting process must be transparent
Objectives 2	Voting process must maintain an immutable audit trail
Objectives 3	Voting process must ensure equal voting rights
Objectives 4	Voter secrecy must be maintained
Objectives 5	Integrity of the vote must be assured

Blockchain technology is a natural candidate for enhancing electronic voting because of the transparency that its immutable audit trail brings. At its core, blockchain technology provides the ability to create a secure digital log of transactions.[14]

Kshetri and Voas describe several early experiments of this nature.[15] Agbesi and Asante describe a blockchain-based voting system to support elections in Africa.[16] Vijayalakshmi and Vimal explore how to maintain voter secrecy in keeping blockchain data encrypted.[17]

It is necessary to closely monitor the behavior of the voting processes in order to detect signs of tampering or malfunction. Robust audit trails that record when devices are removed, inserted, or accessed in any way must be in place. In addition to capturing useful logs, it is imperative to ensure that the log files are not modified or deleted, to assure reliable auditing through data provenance. Some machines have been documented to lack this exact property as they tally votes in a system in which log files can easily be replaced without detection.[18]

While the option of e-voting adds overhead to maintaining an election's integrity, a more fundamental issue stems from the voting process itself. After voters cast their votes, election authorities collect and seal the ballots before transporting them to a central location for tallying.

Several examples of potential vectors for voter fraud exist. For example, without proper safeguards, ballots may be manipulated between the time of submission and collection. Further, voting officials may have the ability to manipulate ballots after they have been cast, or commit fraud during tallying votes.

These scenarios provide windows of opportunity for attackers to tamper with the election process, or even for system errors to have an impact on the election's outcomes. Voters must place a significant amount of trust in authorities and the voting process. Our goal is to eliminate the need for this trust by increasing visibility and accountability during an election. Ultimately this will minimize the chance of vote tampering and increase voter confidence.

Our research follows a number of proposals and implementations of blockchain-based voting, many of which aim to increase transparency in the process while maintaining voter secrecy.

One example of this is Voatz, a mobile platform that allows remote users such as military personnel and people with disabilities to participate in elections.[19] While it adds an audit trail to ballots by recording them in a blockchain, it introduces risks such as vote coercion by allowing remote voting, a form of i-voting. Moreover, it may benefit from tracking voters in a blockchain rather than a centralized source (voter registration system).

Other proposals consider bringing the voting servers (rather than clients) onto the internet by making use of the Bitcoin Blockchain to store votes.[20,21] While Bitcoin may have proven itself a reliable network, carrying out validation for vote transactions on a public network could invite bad actors.

The main contributions of our work are, first, the introduction of separate transaction types which are recorded on two separate blockchains in order to ensure voter secrecy, and second, the introduction of a new consensus algorithm that is specific to electronic voting.

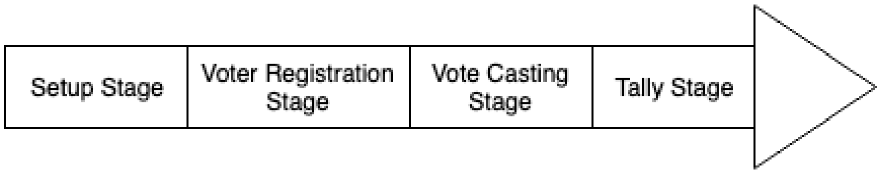

Fig. 1. Stages in the voting process

As previously identified, our research revolves around enhancing electronic voting processes in order to improve election integrity while maintaining voter secrecy. To do so, we made several design choices and also made some key assumptions.

An election can be divided into distinct and separate stages. Hardwick *et al.* distinguish four such phases (initialization, preparation, voting, and counting).[22]

In our research, we adopt a similar model, as described in Figure 1. The first stage in the process is the *setup stage*, in which the sociotechnical infrastructure is prepared for the remaining activities. The methods to execute this stage are currently out of scope for our research and will not be discussed in depth. Instead, we describe the desired outcomes of the stage. Upon completion of the setup stage, all polling locations are known and voting machines are assigned to the various locations. Cryptographic key pairs for each machine are securely generated by the election authority, have been verified and recorded on a private public-key infrastructure, and have been distributed to electronic voting machines. The election ballots with all candidates are prepared and installed on the machines, and all counters are reset to their initial values. After completion of the setup stage, voter registration begins.

During the *voter registration* stage, eligible voters register to cast their vote(s) in a specific election. Determining the voters' identities and their eligibility to participate in the election takes place by external processes and is out of scope for our research; however, we assume that a mechanism is in place to reliably determine a voter's identity at the registration phase, as well as during the vote casting stage. The voter registration stage has a start time and a predefined end time.

The voter registration stage must fully complete before the *vote casting* stage begins. As voters arrive at the polling location, they are asked to identify themselves, and the voter registry is used to cross-reference their identities. As before, we do not impose a specific method to determine the identity of voters, but we do assume that it is reliable.

A voter is authorized to vote when three conditions are satisfied: they have been authenticated, are on the voter roll, and have not yet used their allotted amount of votes.

Once voters have been authenticated, and the determination has been made that they are indeed authorized to vote, they will be issued a *ballot claim ticket*. Once a voter has collected their claim ticket, the voter registry is updated so that equal voting (*i.e.*, that one person does not vote more times than they are allowed to) is ensured.

The claim ticket is evidence of eligibility to cast a vote and can be used to collect a specific ballot. The claim ticket does not contain any references to the identity of the voter. This ensures voter secrecy.

Before a voter is allowed to cast a vote, they must be issued a valid claim ticket and be presented with a list of choices. A claim ticket is valid if it has been issued by the voter authority, and if it has not been previously used to cast a vote. The list of choices takes the form of simple key-value pairs, where the key identified the polling item for which the vote is cast, and the value represents the candidate supported by the voter.

In the final stage, the *tally stage* is initiated immediately after the vote casting stage ends. During the tally stage, all choices are analyzed and counted, and the winners of the election are determined.

3. A Blockchain-Based Electronic Voting Architecture

Voter secrecy is maintained by the use of two separate blockchains. This is in line with other proposals, such as the architecture proposed by Barnes, Brake, and Perry, who describe the use of dual blockhains in elections,[20] a proposal by Goel, Ruhl, and Zavarsky who apply the concept to personal health information,[23] or by Liang, Lei, Li, Fan, and Cai, who apply the idea to copyright registration.[24] However, our solution differs in several key aspects: we provide additional details about the transactions used to update the blockchains, and we discuss the details of our consensus algorithms. Important to note is that we are not proposing the use of parallel blockchains to improve performance, as proposed by Lightman *et al.* in their Dualchain Network Architecture (DNA) whitepaper.[25]

Liang *et al.* separate their copyright data over two blockchains: one public blockchain that contains public data, and one private, permissioned blockchain that contains sensitive private information. As such, their decision to separate data in two chains is predicated on much the same reason: separation of public data and private data. However, as both chains contain references to the same subjects, synchronization between the chains is not an issue. In a voting scenario, the goal is to ensure that subjects are non-identifiable in order to ensure voter secrecy, and additional steps to keep both chains synchronized must be taken.

Figure 2 shows how voter data is maintained on the Voter Blockchain, while the Ballot Blockchain records the votes cast by voters. The Voter Blockchain is updated by *Authentication Machines* broadcasting `claim`-messages, while the Ballot Blockchain is updated when *Vote Recording Machines* broadcast `cast`-messages.

While it is convenient to think about a blockchain as a tangible entity, each voting machine acts as a peer node in a network that maintains a consensus opinion about the state of the blockchain. Since no single node acts as an authoritative source, developing and implementing a consensus protocol is a critical part of maintaining a blockchain-based infrastructure. All nodes connect to a reliable and secure message bus system that supports directed broadcasts.

The Voter Blockchain maintains the Voter Register and the number of unused allocated ballots for each voter. As voters authenticate during a voting stage, it is trivial for a front-end voting machine to determine if the voter is eligible to cast more ballots. If so, the voter is issued a ballot claim ticket. As the ticket is issued, the voting machine broadcasts a `claim`-transaction to all nodes participating in the Voter Blockchain. Figure 3 illustrates the process.

Similar to Votebook, as proposed by Kirby *et. al*, we assume that trust in the election is derived from the presence of an established voting authority, which maintains voting machines.[26] Each blockchain relies on a public-key infrastructure (PKI) that is maintained by the voting

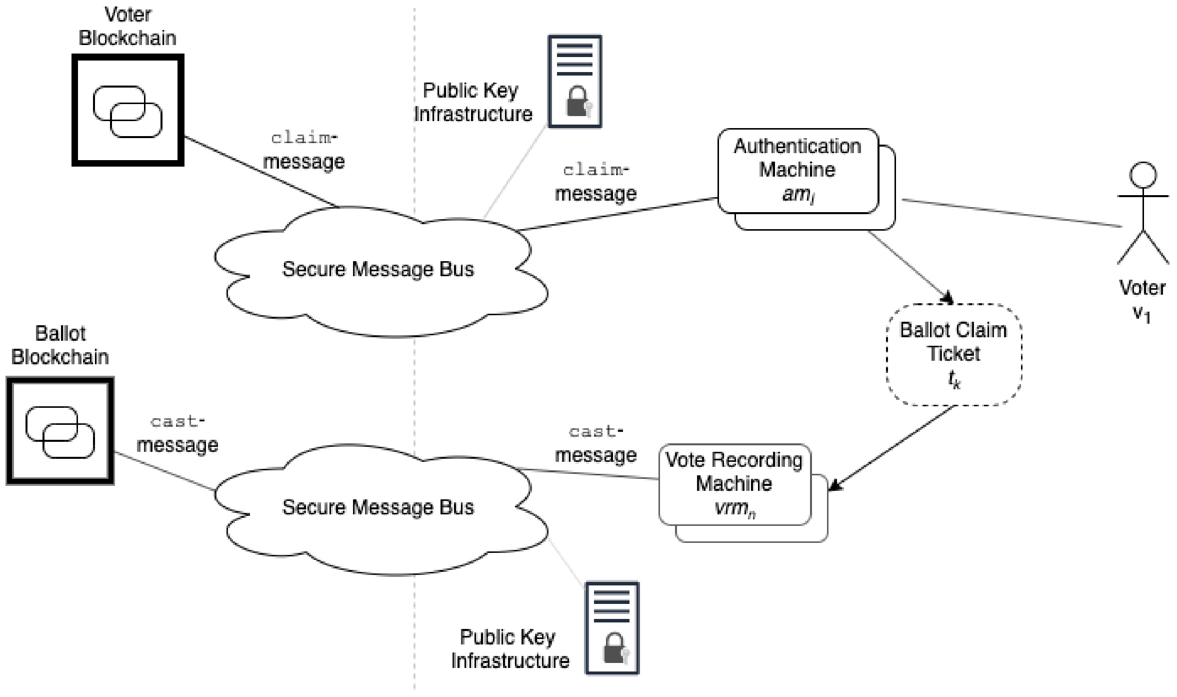
Voter Blockchain
claim-message
Public Key Infrastructure
claim-message
Authentication Machine am_l
Voter v_1
Secure Message Bus
Ballot Claim Ticket t_k
Ballot Blockchain
cast-message
cast-message
Vote Recording Machine vrm_n
Secure Message Bus
Public Key Infrastructure

Fig. 2. Proposed architecture

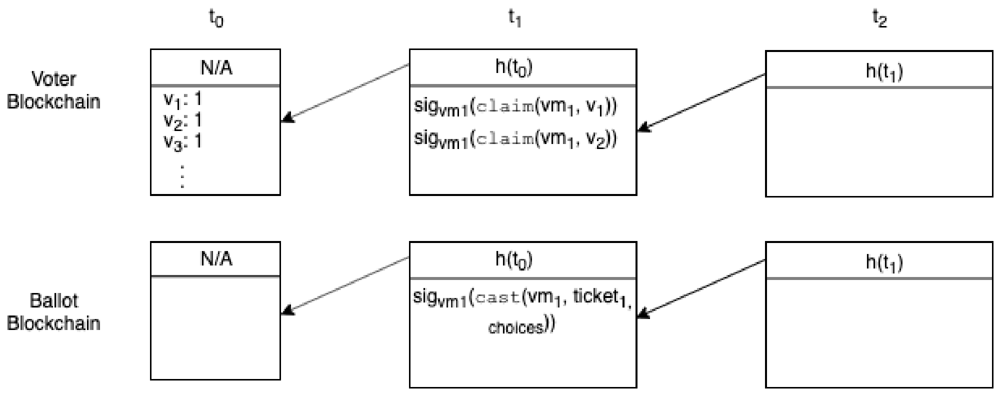

Fig. 3. Transactions on the blockchains

authority. Prior to election, the authority-identified machines will generate an asymmetric cryptographic key-pair and register their public keys with the PKI.

The `claim`-transaction includes an identifier of the voting machine that identified the voter, the voter's identification, and a digital signature created with the voting machine's private key. The transaction reduces the number of eligible votes of the voter who claimed their ticket. Each node records, broadcasts, and synchronizes them into the next block on the blockchain during a consensus round.

The ballot claim ticket consists of three parts: an identifier for the voting machine, a randomly-generated nonce, and a digital signature using the machine's private key. The voting system will maintain a list of previously-used nonces to prevent reusing the same number multiple times. Voting machines have globally-unique identifiers.

The ballot claim ticket unlocks the ability to cast votes to a candidate. The voting machine will create a ballot claim ticket after it identifies a voter, and it will broadcast a transaction representing that fact. Due to the need to maintain voter secrecy, the claim ticket itself is not part of the broadcast.

As voters make their choices, each voting machine will broadcast `cast`-transactions to the nodes participating in the Ballot Blockchain. Each `cast`-transaction includes the identifier of the voting system used to record the vote, the ballot claim ticket, and also the elections made by the voter. The transaction is digitally signed using the private key of the voting system broadcasting it.

The Voter Blockchain maintains an immutable audit trail of voters who have received ballots. The Ballot Blockchain maintains an audit trail of votes. The ballot claim ticket effectively decouples the vote from the voter.

4. Consensus Algorithm

As nodes participating in both the Ballot and Voter Blockchains create transactions, they will have to periodically synchronize these transactions and achieve consensus on the next block. In developing a consensus algorithm, we were quick to rule out Proof-of-Work as an option. Unlike dual-chain e-voting proposals that carry out consensus on Bitcoin, we do not feel that the integrity of a voting system should rely on incentivizing unknown users.[20,27] Furthermore, Bitcoin may have started as a very decentralized network, but it has increasingly become more centralized in the hands of a few large mining groups today.[28]

Our algorithm takes inspiration from Practical Byzantine Fault Tolerance (PBFT), which is able to achieve consensus in the face of approximately 33% Byzantine faults.[29] Byzantine faults are failures in a distributed system in which individual nodes stop communicating, produce errors, or act maliciously. In our model, we assume that our network infrastructure is reliable and that Byzantine faults may only occur on individual nodes, but not in the links between them. PBFT has a lot of overhead in its protocol, and we considered using the Ripple Protocol Consensus Algorithm (RPCA),[30] which reduces the amount of communication significantly by creating overlapping subsets of servers that participate in consensus among themselves. However, by doing so, Ripple lowers its Byzantine Fault Tolerance to approximately 20% and we aspired for a more resilient solution.

In PBFT, clients send a request to a primary node, which is responsible for broadcasting it to all secondary nodes in the network for that round. The client waits for a response from a minimum number of nodes before considering their request to be complete. Meanwhile, nodes must first come to consensus on whether or not they will carry out the request, and which order it will be executed in, relative to other requests. They do this in multiple rounds of communication for each request, which is very expensive. In PBFT, clients are expected to consider transactions as blocking, which means that a new transaction can only be initiated once a previous one fully completes.

Our architecture removes the strain on a single node bearing all this responsibility, since all transactions are distributed over all participating nodes. Requests, or transactions on our voting machines, are validated locally, immediately broadcast to other voting machines as the transactions are entered, and are periodically committed to the blockchain. As a consequence, transaction can be pipelined, where a subsequent transaction can be initiated immediately after the preceding one has been broadcast.

Our algorithm is summarized in Algorithm 1.

Algorithm 1: Generalized Consensus Algorithm

```
1 forall nodes in network do
2     broadcast hash of previous block;
3     exclude nodes with different hash;
4     broadcast candidate set of transactions;
5     validate global candidate set of transactions;
6     broadcast transaction approvals;
7     aggregate transactions with enough approvals into new block;
```

It is worth noting that our algorithm is also similar to Votebook's in that it is time-based.[26] However, instead of a single node proposing its transactions at a given time, all nodes synchronize their transactions during each round. This approach minimizes the duration that transactions remain uncommitted and mitigates transaction loss from the blockchain due to any denial-of-service attacks.

The assumption that nodes are connected using a reliable and secure infrastructure that provides directed broadcasts implies that nodes will be aware when nodes do not receive transaction announcements. We also assume that all nodes have a synchronized time reference, which allows consensus rounds to take place at pre-determined intervals, and that the time has a high resolution.

Consensus-building takes place when each node broadcasts its current block hash, along with the collection of valid transactions that were not recorded on the previous block. This collection is known as the candidate set.

Nodes first look for consensus on the block hash to account for the possibility that nodes are not adequately synchronized. As nodes announce their perception of the state of the blockchain, the most prevalent opinion prevails. Nodes that agree with the majority opinion about the state of the blockchain continue to participate in building consensus. They do so by validating their combined list of transactions.

Nodes that disagree with the state of the blockchain, as well as their transactions, are excluded from consensus-building. The presence of excluded nodes should be exceptional, and any time that nodes are excluded, election authorities must investigate and re-synchronize such nodes with the network.

The validation for all transactions includes several cryptographic checks. The digital signature used to notarize transactions must be made by a machine whose public key was certified by the public-key infrastructure operated by the election authority. Additionally, validating transactions on the Voter Blockchain requires that the voter is on the voter roll and has enough ballot claim tickets left. Similarly, validation for the Ballot Blockchain is determined by the validity of the ballot claim ticket and the ballot as it is cast. If any transaction is in direct conflict with another (*i.e.*, a ballot claim ticket was used twice), then our algorithm chooses the transaction with the earlier timestamp and marks the other for inspection.

Each node will validate all pending transactions and will broadcast their verdict on each transaction in a single message to the other nodes. Each node will aggregate the results and will include those transactions that receive the required amount of approvals in the next block on the blockchain. If a node finds that another node is consistently exhibiting adversarial behavior, it can flag that node as malicious and exclude it from consensus altogether.

Calculating a single hash value over the hash of the previous block, combined with all transactions that exceeded the threshold for acceptance, yield the block hash for the new block. Once the consensus protocol finalizes the current block, it creates a new block and initializes it with the new hash. The process then starts again.

5. Experiment

To achieve full transparency of the architecture and to ensure that the consensus algorithm can be thoroughly reviewed and understood, we built a software prototype of our system using the Python programming language.

The software is capable of running a user-interactive election as well as a simulation with a configurable voter roll and ballot template. We used Python libraries to add digital signatures to transactions as well as logging so that the user can see how the system handles unexpected events, including any malicious behavior. The readability of Python code also served as documentation for our voting process.

To determine if the prototype implementation works correctly under favorable conditions, as well as in the presence of an adversary, we have used it to simulate several scenarios. Associated with each scenario are descriptions, justifications outlining why they are relevant, expected behavior, and actual outcomes.

The simulated scenarios are summarized in Table 3. Scenario 1 acts as a control scenario, which demonstrates that the system works as expected under normal circumstances. Scenarios 2–4 explore situations in which the voting system itself is not compromised, but votes are attempting to subvert the integrity of the election. Lastly, Scenarios 5–7 assume the presence of an active adversary on the network of nodes.

Each of these scenarios has been implemented as a proof-of-concept and can be run as a simulation using the software prototype.

In Scenario 1, 100 registered voters successfully authenticate and cast valid votes. The election is set up with two ballot items and includes two candidates per item. Sixty percent of the votes will be cast to one candidate, while forty percent of voters will cast to the other. We

Table 3. Proof-of-Concept Testing Scenarios

Scenario	Description	Purpose
Scenario 1	Valid voters casting valid votes	Control
Scenario 2	Unknown voter attempting to cast vote	Voter fraud
Scenario 3	Valid voters attempting to cast extra vote	Voter fraud
Scenario 4	Valid voters attempting to cast invalid vote	Voter fraud
Scenario 5	Node broadcasting invalid transaction	Infrastructure attack
Scenario 6	Adversarial node creating invalid claim tickets	Infrastructure attack
Scenario 7	Adversarial node not participating in consensus round	Infrastructure attack

expected to see this division of votes to be accurately reflected in the final tally. Actual outcomes matched expectations.

To ensure that voters can only cast their vote for listed candidates, a valid voter attempts to cast a ballot for a non-existing candidate. A second test will simulate that a vote will be cast for a valid candidate, but includes an invalid value for the ballot item.

For this to work, the voter must have a method to inject votes in a way that bypasses the user interface of the voting machine. If this is indeed possible, the transaction will broadcast to the network. However, the consensus algorithm will determine that both votes were invalid. Consequently, it will exclude the transactions from entering the next block, and the integrity of the election is assured.

The previous scenarios described situations in which undesirable voter behavior occurs on nodes with valid cryptographic keys, and that are not under the control of an opponent. However, the introduction of computer technology into the voting process must also accommodate for situations in which an adversary has control of at least one voting computer and may use it to influence outcomes of an election. In these adversarial situations, we maintain the assumption that the adversary cannot access the node's private key, and that they are unable to create cryptographic signatures using a key that is recognized by the public-key infrastructure.

Adversaries who have been able to intrude on voting machines are presumably able to broadcast any transaction. However, transactions must be valid to be considered for adoption into the next block. Transactions can be invalid for several reasons. First, a transaction is invalid when it does not contain a valid digital signature. Nodes that receive unsigned broadcasts must always reject them without applying any further processing. When nodes receive a vote-casting transaction without a proper ballot claim ticket associated with it, the consensus protocol flags it as invalid during a consensus round.

Each of the scenarios described in table 3 is implemented in the proof-of-concept, which is publicly accessible online.[31]

Scenario 1 is the control scenario, and it provides the base upon which all other scenarios are built. The file `main.py` provides the starting point of execution, and maps out the different scenarios. Scenarios 2 and 3 do not require any specific coding changes, as they can be executed by simply entering the appropriate parameters into the simulation. The `simulation_map` dictionary defines this behavior.

Scenario 4–7 are implemented in `adversary.py` by overriding specific methods in the base classes. For example, Scenario 4 "breaks" a voting computer by defining an inherited class

`InvalidBallotVotingComputer` which overrides the `get_ballot()`-method. Similarly, Scenario 5 is implemented by overriding the `sign_message()`-method in the `UnrecognizedV⌋ oterAuthenticationBooth` class. By defining adversarial behavior in inherited classes that override correct behavior, we have created an elegant and extensible mechanism to rapidly implement additional adversarial scenarios.

6. Conclusions

The goal of our research is to determine how electronic voting can support the integrity and transparency of the voting process, without sacrificing the security of the ballot. Specifically, we set out to determine if it is possible to design and build a system in which the voting process is transparent, generates an immutable electronic audit trail, voter secrecy is maintained, and the integrity of the vote is assured.

To do so, we asked the question: in what way can an electronic voting process provide election integrity while maintaining voter secrecy? The proposed dual-chain blockchain voting architecture and the accompanying consensus algorithm are an answer to that question, and we demonstrate, through a proof-of-concept, that all design objectives are met.

In our pursuit of reaching this answer, we designed a blockchain-based system that allows voters to use electronic voting machines to cast ballots. We propose the use of two separate blockchains to ensure that voter secrecy is maintained and that voters are anonymous.

The blockchain consensus algorithm proposed in the research will ensure equal voting rights, and that election authorities can tally votes correctly; however, for the system to be secure, several assumptions had to be made. These assumptions include the ability to uniquely identify voters, and to determine their eligibility to participate in the election. The technological implementation of our ideas relies on the availability of tamper-proof voting machines, which contain user interfaces that ensure only correct votes are cast. We also assume the availability of a reliable and secure message broadcast mechanism to which all voting locations (nodes) are connected, the use of a public-key infrastructure that manages trust relationships, and the assumption that private-key materials are unavailable to adversaries. However, subject to these assumptions, the proposed system ensures that votes can only be cast by authorized voters, that authorized voters are only able to cast valid votes (*i.e.*, choose known candidates and only vote as many times are they are allowed to), maintain voter secrecy, establish an immutable audit trail, and tally results without losing any votes. The system is also resistant to several forms of tampering.

The architecture proposed in this paper offers several benefits with regards to elections that are supported by electronic voting machines. The main benefit of the use of blockchain technology is establishing an immutable audit trail that can be used to detect anomalies from the voting stage onward. While the use of blockchain technology is not a solution to all election-related issues, it can address some problems.

Specifically, our contributions are to be found in the overall systems architecture, but specifically in the use of a dual blockchain architecture and in the consensus algorithm. We propose to use separate chains to ensure voter secrecy. Our contribution demonstrates that a dual-chain architecture is not only useful for efficiency of processing, but also for preservation of secrecy using two chains that are indirectly linked. Like other similar algorithms, our consensus algorithm is time-based, but it proposes that all nodes participate. This approach minimizes the duration that transactions remain uncommitted and mitigates transaction loss as the result of technical failure of denial-of-service conditions.

7. Future Work

7.1. Relaxing Assumptions—Our work has focused on incorporating blockchain technology into the electronic voting process to improve transparency and to improve the ability to electronically audit the voting process, while maintaining voter secrecy. In developing our approach, we have made several assumptions, such as outlined in Table 1. In future work, we will explore the feasibility of relaxing some of those assumptions.

For example, if we relax that assumption of strong voter authentication practices, we may introduce situations in which fraudulent voters are able to spoof the identity of legitimate ones. This can potentially lead to multiple ballots being submitted for the same voter. Specifically, we consider four separate scenarios:

(1) One voter deliberately votes at multiple stations during the same time consensus round. While that would be difficult to do based on simple logistics, it could lead to a race condition that causes multiple claim tickets to be issued to the same voter.
(2) One voter colludes with another to cast multiple votes at the same time, in different polling stations. This scenario is somewhat easier to accomplish than the previous one, but would still require significant levels of coordination. However, if successful, it could lead to the same result: multiple claim tickets might be issued on behalf of the same voter.
(3) We also consider a scenario in which an adversary steals the identity of a voter who is not in collusion with them. Both the adversary and the legitimate voter would have to collect their ballots at the same time slot. That is a high burden, but if successful, could lead to multiple claim tickets being issued.
(4) An adversary steals the identity of a voter who is not in collusion with them, and casts their vote(s) in a consensus round before the legitimate voter does. In that case, the legitimate voter will be locked out and lose their vote(s).

Since each transaction is timestamped based on a high-resolution clock, we currently resolve the issue by accepting the oldest transaction during each consensus round, and refusing all later ones. A more subtle solution to the first three scenarios may be found through a combination of steps.

First, a voter would be assigned to a specific voting location. Any votes casts at different locations would be invalid by default. However, by doing so, we impose barriers on the voting process, which is undesirable. To address that issue, we might introduce an additional transaction that would facilitate updating of the polling site. To minimize the risk of similar race conditions, moving the polling site, claiming a ticket, receiving a ticket, and casting a vote would have to take place in distinct consensus rounds.

7.2. Extending the Scope—While our solution addresses some of the most critical elements of election security in an electronic voting delivery model, several unaddressed issues warrant further investigation.

As mentioned in the introduction, we have not tested this model in an i-voting election. While there are no immediate reasons to assume that our approach will break down in a fully online election, we did not test that assumption.

Since the consensus algorithms reject votes for unknown candidates, elections using write-in candidates are currently not possible. Exploring how write-in candidates can be supported is a research question for future work.

We anticipate extending the proof-of-concept with additional audit trails that will allow greater transparency. In particular, it would be interesting to capture an audit trail of the consensus rounds themselves and use it to detect election fraud at an early stage. We anticipate that establishing these additional audit trails would not cause any difficulties, but have not tested it yet. In addition, we consider the idea of publicly releasing audit trails in order to improve election transparency.

In future research, we also wish to explore the possibility to integrate this system with a hybrid system in which ballots are cast on paper and then scanned into the electronic voting machine. This method of voting is commonplace in many U.S. voting districts.

The current prototype depends on two key assumptions: the presence of a secure and reliable directed-broadcast infrastructure, and the ability to create cryptographic keys that cannot be compromised by advanced adversaries. Additional research is needed to determine if these two assumptions can be relaxed.

Acknowledgement

Mateusz Gembarzewski worked closely with Jai Punjwani when they began the initial work on this idea as part of the undergraduate curriculum. Mark Grabowski provided valuable feedback and was kind enough to proofread this paper.

Author Contributions

JP implemented the proof-of-concept implementation. Its source code is posted at `https://github.com/jaipunjwani/blockchain-based-voting`. Both authors worked closely together on conducting the research and on designing the architecture. KL was JP's thesis supervisor and is the main author of the paper.

Notes and References

[1] Khan, K. M., Arshad, J., Khan, M. M. "Investigating Performance Constraints for Blockchain Based Secure E-Voting Systems." *Future Generation Computer Systems* **105** 13–26 (2020) `https://doi.org/10.1016/j.future.2019.11.005`.

[2] Yi, H. "Secure E-Voting Based on Blockchain in P2P Network." *EURASIP Journal on Wireless Communications and Networking* **137** (2019) `https://doi.org/10.1186/s13638-019-1473-6`.

[3] Jefferson, D., Rubin, A., Simons, B., Wagner, D. "Analyzing Internet Voting Security." *Communications of the ACM* **47.10** 59–64 (2004) `https://doi.org/10.1145/1022594.1022624`.

[4] Park, S., Narula, N., Rivest, R. L. "Going from Bad to Worse: From Internet Voting to Blockchain Voting." *MIT* (2020) `https://people.csail.mit.edu/rivest/pubs/PSNR20.pdf`.

[5] Selker, T., Cohen, S. "An Active Approach to Voting Verification." *MIT* (accessed 17 January 2021) `https://dspace.mit.edu/handle/1721.1/96565`.

[6] Ryan, P. Y., Bismark, D., Heather, J., Schneider, S., Xia, Z. "Prêt à Voter: A Voter-Verifiable Voting System." *IEEE Transactions on Information Forensics and Security* **4.4** 662–673 (2009) `https://doi.org/10.1109/TIFS.2009.2033233`.

[7] Moura, T., Gomes, A. "Blockchain Voting and its Effects on Election Transparency and Voter Confidence." In *dg.o '17: Proceedings of the 18th Annual International Conference on Digital Government Research* 574–575 (2017) `https://doi.org/10.1145/3085228.3085263`.

[8] Oostveen, A.-M., Besselaar, P. v. d. "Security as Belief: User's Perceptions on the Security of Electronic Voting systems." In A. Prosser, R. Krimmer (Eds.), *Electronic Voting in Europe - Technology, Law, Politics and Society, Workshop of the ESF TED Programme Together with GI and OCG* Bonn: Gesellschaft für Informatik e.V. 73–82 (2004) `https://dl.gi.de/handle/20.500.12116/29131`.

[9] Appel, A. W. "Effective Audit Policy for Voter-Verified Paper Ballots." *American Political Science Association* (accessed 19 January 2021) `https://www.cs.princeton.edu/~appel/papers/appel-nj-audits.pdf`.

[10] Rivest, R. L., Stark, P. B. "When Is an Election Verifiable?" *IEEE Security and Privacy* **15.3** 48–50 (2017) `https://doi.org/10.1109/msp.2017.78`.

[11] Cucurull, J., Rodríguez-Pérez, A., Finogina, T., Puiggalí', J. "Blockchain-Based Internet Voting: Systems' Compliance with International Standards." In *BIS 2018 International Workshops* Springer Nature Switzerland AG 300–312 (2019) `https://doi.org./10.1007/978-3-030-04849-5_27`.

[12] Wagner, A. "Can Blockchain-Enabled Voting Meet Security and Secrecy Standards?" *Chicago Policy Review (online)* (2019) `https://chicagopolicyreview.org/2019/04/09/can-blockchain-enabled-voting-meet-security-and-secrecy-standards/`.

[13] Hjálmarsson, F., Hreidarsson, G., Hamdaqa, M., Hjálmtýsson, G. "Blockchain-Based E-Voting System." In *Proceedings of 2018 IEEE 11th International Conference on Cloud Computing* 983–986 (2018) `https://doi.org/10.1109/CLOUD.2018.00151`.

[14] Racsko, P. "Blockchain and Democracy." *Society and Economy* **41.3** 353–369 (2019) `https://doi.org/10.1556/204.2019.007`.

[15] Kshetri, N., Voas, J. "Blockchain-Enabled E-Voting." *IEEE Software* **35.4** 95–99 (2018) `https://doi.org/10.1109/MS.2018.2801546`.

[16] Agbesi, S., Asante, G. "Electronic Voting Recording System Based on Blockchain Technology." In *2019 12th CMI Conference on Cybersecurity and Privacy (CMI)* 1–8 (2019) `https://doi.org/10.1109/CMI48017.2019.8962142`.

[17] Vijayalakshmi, V., Vimal, S. "A Novel P2P Based System with Blockchain for Secured Voting Scheme." In *2019 Fifth International Conference on Science Technology Engineering and Mathematics (ICONSTEM)* 153–156 (2019) `https://doi.org/10.1109/ICONSTEM.2019.8918895`.

[18] No Author. "UniLect Patriot (and VMR 138)." *verifiedvoting.org* (accessed 7 April 2019) Page no longer available at original address, archived at `https://web.archive.org/web/20200807000332/https://verifiedvoting.org/resources/voting-equipment/unilect/patriot/`.

[19] Moore, L., Sawhney, N. "Under the Hood: The West Virginia Mobile Voting Pilot." (2019) `https://blog.voatz.com/wp-content/uploads/2019/02/West-Virginia-Mobile-Voting-White-Paper-NASS-Submission.pdf`.

[20] Barnes, A., Brake, C., Perry, T. "Digital Voting with the Use of Blockchain Technology." *The Economist* (accessed 1 March 2018) `https://www.economist.com/sites/default/files/plymouth.pdf`.

[21] Lee, K., James, J. I., Ejeta, T. G., Kim, H. J. "Electronic Voting Service Using Block-Chain." *The Journal of Digital Forensics, Security and Law* **11.8** (2016) `https://doi.org/10.15394/jdfsl.2016.1383`.

[22] Hardwick, F. S., Akram, R. N., Markantonakis, K. "E-Voting With Blockchain: An E-Voting Protocol with Decentralisation and Voter Privacy." *2018 IEEE International Conference on Internet of Things (iThings) and IEEE Green Computing and Communications (GreenCom) and IEEE Cyber, Physical and Social Computing (CPSCom) and IEEE Smart Data (SmartData)* 1561–1567 (2018) `https://doi.org/10.1109/Cybermatics_2018.2018.00262`.

[23] Goel, U., Ruhl, R., Zavarsky, P. "Using Healthcare Authority and Patient Blockchains to Develop a Tamper-Proof Record Tracking System." In *2019 IEEE 5th Intl Conference on Big Data Security on Cloud (BigDataSecurity), IEEE Intl Conference on High Performance and Smart Computing, (HPSC) and IEEE Intl Conference on Intelligent Data and Security (IDS)* 25–30 (2019) `https://doi.org/10.1109/BigDataSecurity-HPSC-IDS.2019.00016`.

[24] Liang, W., Lei, X., Li, K.-C., Fan, Y., Cai, J. "A Dual-Chain Digital Copyright Registration and Transaction System Based on Blockchain Technology." *Communications in Computer and Information Science* **1156** (2020) https://doi.org/10.1007/978-981-15-2777-7_57.

[25] Lightman, A., Gu, E., Huang, K., Hao, C. "Dualchain Whitepaper." *Metaverse Foundation* (accessed 2 January 2021) https://mvsdna.com/MVS_Dualchain_White_Paper-v.1.00.pdf.

[26] Kirby, K., Masi, A., Maymi, F. "Votebook, A Proposal for a Blockchain-Based Electronic Voting System." *The Economist* (2016) https://www.economist.com/sites/default/files/nyu.pdf.

[27] Lee, K., James, J. I., Ejeta, T. G., Kim, H. J. "Electronic Voting Service Using Block-Chain." *The Journal of Digital Forensics, Security and Law* **11.8** (2016) https://doi.org/10.15394/jdfsl.2016.1383.

[28] Schwartz, D. "The Inherently Decentralized Nature of XRP Ledger." (2018) https://ripple.com/insights/the-inherently-decentralized-nature-of-xrp-ledger/.

[29] Castro, M., Liskov, B. "Practical Byzantine Fault Tolerance." In *Proceedings of the Third Symposium on Operating Systems Design and Implementation* 173—-186 (1999) https://doi.org/10.5555/296806.296824.

[30] Schwartz, D., Youngs, N., Britto, A. "The Ripple Protocol Consensus Algorithm." *arXiv* (2018) https://arxiv.org/abs/1802.07242.

[31] https://github.com/jaipunjwani/blockchain-based-voting.