NEW YORK UNIVERSITY

Votebook

A proposal for a blockchain-based electronic voting system

Written by
Kevin Kirby
Anthony Masi
Fernando Maymi

Submitted
September 29, 2016

The rise of the Internet has ushered in a new set of challenges to the democratic process. Perhaps chief among them is the specter of cyber attacks against electronic voting systems, by which one or few actors could catastrophically disrupt the expression of the will of the people. Cybersecurity was barely on anyone's mind when the first electronic voting machines hit the market in the early 1990s—programmers had little incentive to spend their time securing their software, and election officials were far more concerned with the worthy goal of making voting cheaper, easier, and more accessible to the electorate. A quarter-century later, the machines that we vote with are woefully inept at keeping out today's hackers.

"Voting systems need to convince rational losers that they lost fairly," writes Nicholas Weaver. "In order to do that, it is critical to both limit fraud and have the result be easily explained."[1] The new cybersecurity landscape in which these electronic voting systems operate has thrown Weaver's imperative into the wind: when votes are stored in bits, how hard will it be for an adversary to flip them? Or delete them entirely? Or create them whole cloth? Further still, how can these systems ensure a secret ballot, by which each voter's choice is confidential? Every democratic election needs a system for recording votes that will be 1) as close to tamper-proof as possible, 2) auditable, in the event that tampering is suspected, and 3) consistent with voter's expectations of privacy.

Enter blockchain technology, the apparatus that supports Bitcoin. A blockchain is a distributed, irreversible, incontrovertible public ledger—basically double-entry accounting for the digital age. The technology works through four features. First, the ledger exists in many different places, meaning that if one copy is destroyed the impact to the ledger is negligible: there is no single point of failure in the maintenance of the database. Second, there is control over who can append new transactions to the ledger.[2] Third, any proposed addition (new "block") to the ledger must reference the previous version of the ledger, linking blocks together like a chain and preventing any tampering with the integrity of previous entries. Finally, a proposed new block of entries must achieve consensus from a majority of the

[1] Nicholas Weaver, *Secure the Vote Today*, LAWFAREBLOG (Aug. 8, 2016, 12:42 PM), https://www.lawfareblog.com/secure-vote-today.

[2] Bitcoin, for example, uses a "proof-of-work" mechanism: only the holder of the solution to a cryptographic problem will receive recognition from the rest of the network. The solution to the problem is computationally difficult to find, but trivial to test. Thus when one player finds the solution, the others can quickly verify that the solution is correct.

rest of the network before it can be added to the ledger. The other players on the network will validate the aforementioned conditions as well as the validity of the proposed transactions before they consent. These features operate with the clever use of advanced cryptography, providing properly implemented blockchain systems with a level of security as theoretically solid as any database in the history of data. We therefore believe blockchain technology is an ideal tool for the modern democratic process.

The following proposal offers the possibility of implementing a blockchain-based system that would help secure electronic voting. We recommend a "permissioned blockchain," which allows for ex ante control over who can participate in the database. Part 1 reviews the design considerations at play; Part 2 is an exposition of details of our proposal, which we call Votebook; and Part 3 examines residual risk and policy ramifications. Finally, readers may find a proof of concept in the Appendix. We believe Votebook is a realistic, practical, and scalable solution for electronic voting that satisfies the requirements of democracy.

1. Design Considerations

Although elegant and (thus far) invincible, the original blockchain implementation employed by the Bitcoin network does not immediately lend itself to use in a democratic election. An election involves its own uniquely difficult set of design considerations. First, the system should ensure that an individual can check to see if her own vote was counted, but that individual must not be able to discern what her neighbor's vote was. Second, the system should not enable coerced voting. Third, depending on the rules of the election, the system should either produce or obscure interim results as desired. Fourth, the system must not force undecided voters to vote—those voters must be able to abstain, and their abstentions must not count for a candidate in the election. Fifth, should the election results be contested, the system must be amenable to audit. Our team saw fit to acknowledge three additional considerations: that not every voter has a personal computer with access to the Internet; that elections are not global affairs, i.e. only citizens can vote; and finally, that the most practical system would be the one that demands the least amount of behavioral change from the voter.

We quickly ruled out a model that would implement remote voting. Even if every voter were issued a computer on which to vote, there are too many threats: authentication of the voter is a daunting challenge; the voter's personal computer could be compromised; it is not clear auditing would be available; there could be a coercive agent inside the voter's home; more alarming, even a minimally resourced adversary could launch a denial of service attack against an entire neighborhood.[3] In short, we believe deference is due to the chorus of computer scientists who emphasize the need for an audit trail in the physical world.[4] We therefore determined that the ideal system would incorporate a familiar in-person voting scheme with paper results that back a blockchain-based digital ledger.

2. Proposal

Votebook borrows the most important ideas from blockchain technology but in a slightly altered format, called a "permissioned blockchain." Like Bitcoin: we employ a distributed database, there is no retroactive editing, and there must be consensus for changes. Unlike Bitcoin: there is no proof of work mechanism. Proof of work channels brute computing power to limit permission within a *trustless* system; however, elections are not trustless--they are restricted to their electorate. If there must be trust, we determined that the best leverage of trust would be to allow a centralized authority to oversee the distribution of encryption keys to the nodes in the election network—hence the term "permissioned blockchain." Nodes must have prior permission from the central authority to make changes to the ledger.

[3] *See* Nelson Hastings et al., Nat'l Inst. of Standards and Tech., *Security Considerations for Remote Electronic UOCAVA Voting (2011)*, *available at* https://www.nist.gov/sites/default/files/documents/itl/vote/NISTIR-7700-feb2011.pdf.

[4] *See* Lawrence Norden and Christopher Famighetti, *America's Voting Machines at Risk*, The Brennan Center for Justice, *available at* https://www.brennancenter.org/sites/default/files/publications/Americas_Voting_Machines_At_Risk.pdf (Quoting one of the nation's leading cryptologists, Ron Rivest: "The biggest concern I have about Internet voting is that we don't know how to do it securely...It sounds wonderful but it's an oxymoron. We don't have the Internet experts who know how to secure big pieces of the Internet from attack."); Weaver, *supra* n.1; Ben Wofford, *How to Hack an Election in 7 Minutes*, POLITICO (Aug. 5, 2016), http://www.politico.com/magazine/story/2016/08/2016-elections-russia-hack-how-to-hack-an-election-in-seven-min; Dan Wallach, *Election Security as a National Security Issue*, FREEDOM TO TINKER (Aug. 3, 2016), https://freedom-to-tinker.com/2016/08/03/election-security-as-a-national-security-issue/; Eric Geller, *Online Voting is a Cybersecurity Nightmare*, THE DAILY DOT (Jun. 6, 2016, 6:32 PM), http://www.dailydot.com/layer8/online-voting-cybersecurity-election-fraud-hacking/, utes-214144.

A. The Ledger of Votes

The network nodes are voting machines. Prior to the election, state-identified voting machines will generate a private and public key pair.[5] The machine will securely store its private key, and then send its public key to the server of a central authority such as a state election commission. That centralized server will compile the public keys of the approved voting machines into a table and redistribute the table to each of the voting machines.

Once polls open, voting machines will collect individual votes and periodically organize them into a "block" for proposal to the network. To eliminate collisions, nodes take turns offering their blocks according to a time-based protocol.[6] The proposed block will consist of the node's unique identifier, a timestamp, and three validation segments: first, a set of rows, each row representing one voter and his vote; second, the block will contain a hash of the previous block in the database; finally, the block will contain a digital signature, which means the node will use its private key to encrypt a hash digest of the rest of the block.[7] The proposing node will broadcast the block to the network, and every other node in the network will check the validity of the block's components. First, the other nodes will use the public key that corresponds to the proposing node's unique identifier to decrypt the hash of the proposed block and verify a match. The receiving node will then verify that the hash of the previous block in the database is correct. If these conditions are satisfied, the receiving node will append the existing database with the new block. These steps will be performed nearly simultaneously by each node on the network.

[5] Public key cryptography uses a pair of keys that can decrypt cipher text produced by the other. Generally one key is kept secret (the private key), while the other can be shared freely (the public key). If Alice wants to send Bob a secret message *M*, she encrypts *M* with Bob's public key to create cipher text *C* and can send *C* over public channels. Bob then uses his private key to decrypt *C* and retrieve *M*. If Alice wants to prove to Bob that she sent *C* and the *C* has not been tampered with, she can sign it by hashing *C* and then encrypting *C* with her private key to get a signature *S*. If Bob can decrypt *S* to the hash of *C*, he knows that whoever sent the message must have Alice's private key.

[6] Nodes that miss their time slots will be skipped; repeatedly absent nodes will be investigated for corruption.

[7] A cryptographic hash function is a one-way encryption of data. The output of a hash function--a hash or digest--is a bit string of fixed length that is practically irreversible, which allows quick validation of the original data's integrity. A hash of this paper would look completely different if even one letter in the document was changed.

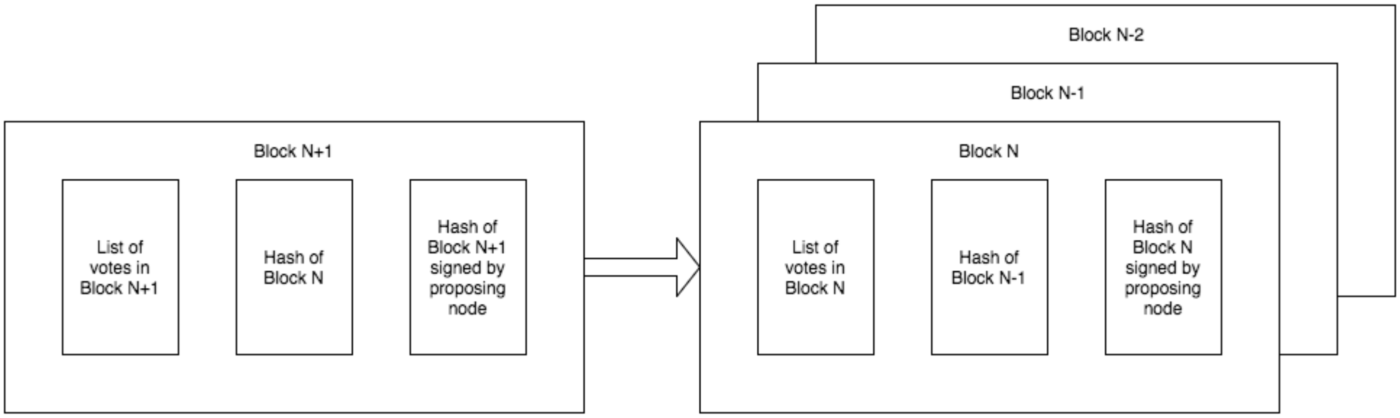
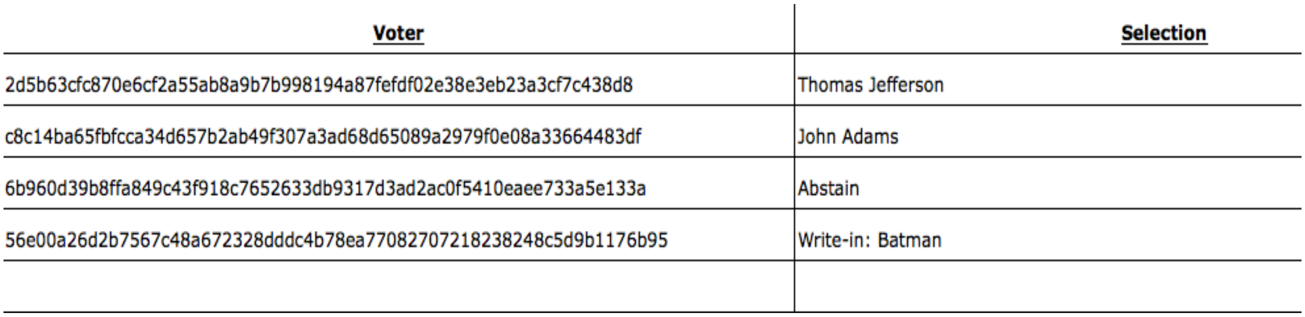

Block N-2
Block N-1
Block N+1
List of votes in Block N+1
Hash of Block N
Hash of Block N+1 signed by proposing node
Block N
List of votes in Block N
Hash of Block N-1
Hash of Block N signed by proposing node

Fig. 1: A proposed block includes the votes proposed, a hash of the previous entry on the chain, and a hash of itself, signed with its own private key.

Within the rows of votes in each block, one column will be a hash digest that uniquely but anonymously identifies the voter (more below). In the right column will be the voter's choice. Voters will be able to abstain or submit a write-in vote through an on-screen keyboard.[8] Although voters will not know the difference from inside the polling booth, the network will be running one blockchain for each election (President, Governor, Dogcatcher).



Voter	**Selection**
2d5b63cfc870e6cf2a55ab8a9b7b998194a87fefdf02e38e3eb23a3cf7c438d8	Thomas Jefferson
c8c14ba65fbfcca34d657b2ab49f307a3ad68d65089a2979f0e08a33664483df	John Adams
6b960d39b8ffa849c43f918c7652633db9317d3ad2ac0f5410eaee733a5e133a	Abstain
56e00a26d2b7567c48a672328dddc4b78ea77082707218238248c5d9b1176b95	Write-in: Batman

Fig. 2: A proposed block will consist of a set of unique voter hashes paired with the voter's choice. The values in the "Voter" column are comprised of a hash of the voter ID concatenated with the ballot ID. E.g., the first row represent a SHA-256 hash of a voter with ID of JDOE12345 and a Ballot ID of 0123456789.

[8] The keyboard must not contain any characters that enable an injection attack.

B. The Ledger of Voters

In order to detect ballot stuffing, Votebook implements another blockchain running in parallel to the ledger(s) of votes. The blocks on this chain will consist of a list of hashed voter IDs processed by the machine, a hash of the previous block on the ledger, and a hash of the proposed block that is encrypted with the voting machine's private key.

There is a risk of an adversary lining up voter blocks with vote blocks to allow statistical inferences about a given voter's choice. Thus in addition to scrambling the order of voters in the block, the voter ledger must also be kept private and never released. To preserve privacy, the nodes encrypt the voter IDs with a symmetric key from the election authority. Further mitigating the risk of correlation, the blocks on the voter ledger must be of smaller size than the vote ledger's blocks.[9] Voting machines will then block votes that rely on an ID whose hash is already on the voter ledger. Additionally, voting stations will maintain a local list of the voters who are registered to vote at that location. Any unexpected voter IDs can be flagged for further attention or barred from voting, based on the rules of the jurisdiction.

C. The Voter's Experience

The process of voting will be familiar to most voters. During voter registration, which takes place outside of Votebook, the voter will receive a voter ID. This ID can be a driver's license or some other pre-existing form of identification—it does not need to be a secret, and does not need to depart from what most jurisdictions use today. At this point the voter must also specify the location at which she intends to vote, as on Election Day the on-site administrators will check whether a voter with a specific ID is supposed to be at that site.

To cast a vote, the voter will interact with a graphical user interface (GUI) that employs a touch screen for candidate selection and an onscreen keyboard for voter ID and write-in vote entries. There will also be a prompt to ask the voter whether she is voting under duress. After the voter has made her choices,

[9] For example, if the node submits blocks to the vote ledger every five minutes, a good voter ledger interval would be every three minutes. A faster updating rate for the voter rolls than the votes will also improve the speed of detection in the event a voter reuses an ID elsewhere.

the machine will generate a random ballot ID. If the voter indicated duress, the voting machine will provide a ballot ID that secretly codes the vote to remain uncounted: when hashed, the ID will contain a certain character in a certain position of the hash,[10] signaling the machine to flag the vote. The coerced vote will go onto the public ledger, but the machine will locally store the votes that should not be counted and then securely convey them to election authorities for analysis. Public knowledge of this countermeasure should render coercion impotent. A set time after the conclusion of the election, the coercion codes will be released so that third parties can independently and accurately analyze the vote ledger. When a successful ballot ID has been generated that properly matches the voter's duress indication, the machine will print out two ballot receipts. One receipt, containing only the ballot ID, will be for the voter. The other receipt will contain the hash value (voter ID and ballot ID combined) and vote choices, including duress indication, and will remain at the voting site in a secure space for paper auditing. No one will be able to look up the voter's selection unless she has both the voter ID and the ballot ID.[11]

Because interim results may or may not be permitted, depending on the jurisdiction, the ledger can be shared beyond the network of voting machines as desired. Whenever appropriate (e.g. at the conclusion of the election), the ledger and the set of public keys for each voting machine can be released to the public at large, at which point any voter will be able verify that her own vote was counted once and may even examine the integrity of each block, but will not be able to decipher the hashes representing the identities of other voters.[12]

[10] For instance, the code could be the appearance of the second or tenth letter of a hash of the voter ID in the fifth position of the hash of the ballot ID. SHA-256 uses 16 possible characters, so there is approximately a .12 probability of a match. The voting machine must compute an average of 8.25 hashes to develop a ballot ID that contains the code for a coerced voter. A malicious actor would have difficulty identifying the code simply by analyzing the public ledger of votes. Obviously, if duress has not been indicated, the machine must generate a ballot ID that does not relay the secret code.

[11] In a system in which the voting machines aggregate votes sequentially (as they are received), a voter in the database could identify the choice of the voter who voted immediately before or after him on the same machine. To mitigate this threat, the voting machines need to scramble the ordering of their blocks prior to block proposal.

[12] Unfortunately, the only way to check whether a voting machine did not propose a set of votes created out of whole cloth is to compare the volume of votes proposed with the paper ballots the machine was supposed to have processed. We assess that with the existence of paper ballots, the risk of this type of fraud will be minimal.

D. Voting Machine Security Specifications

Access to Votebook will be as limited as possible. First, voter interaction will be restricted to input through a touchscreen or mouse that interacts with the GUI. The computers themselves will be stored in a locked container to prevent physical tampering and placed in public view at the voting station. For transparency and to help detect software tampering, a system image of the voting machines should be released shortly before voting begins and immediately following the conclusion of voting. These images provide a snapshot of the contents of the computer and could be examined in the event that the voting machine's integrity is in question. During the election, the voting machines will be set to run on a special, single permission user mode that can only run the voting software.

Voting machines will never touch the open web. On Election Day, the computers will be connected to a virtual private network (VPN) that connects them to the other voting machines, establishing a virtual tunnel of encryption insulating the network's communications from the open Internet. Firewall white listing will allow only trusted IP addresses to send the voting machines data and only through specified ports.

Blocks will be signed with private keys that are generated by the voting machines before Election Day.[13] The public keys are then collected by a central administrator and sent out to all voting machines. Each machine signs the entire table with its private key and then sends that signature back to the administrator. The administrator appends the signatures to the table and then redistributes the table. When a new block is proposed, each machine will check the signature against the public key on the table and the table is checked against its signatures. Should a node's table be corrupted, that node may query other nodes for their tables.

[13] Notwithstanding budgetary constraints, a trusted platform module would be ideal for segmenting the voting machine's private key from the disk.

3. Assessment

While our proposed blockchain voting system provides the means to create a database with desirable features, it does not solve every threat facing electronic voting. First, individual voting machines can still be tampered with or simply denied service. Through physically securing the machines, the use of a VPN and a firewall, and paper auditing, we believe we have mitigated these threats. Of course there is going to be added pressure on the physical security of the paper ballots, but that is a problem that today's paper voting systems have already evolved to deal with. Second, the need to publish accurate counts is in tension with the desire to obscure uncounted coerced votes. Voting jurisdictions will have to determine when is the appropriate time to release the codes that indicate which votes were not counted due to indicated duress. Finally, there is the threat that voter registration databases will be targeted.[14] Those databases are beyond the scope of this proposal, but registration systems appear to be ripe for their own blockchain implementation.

Rolling out any new voting system is likely to be politically fraught. In the U.S., where the electoral process is governed at the state level, the Election Assistance Commission[15] publishes voluntary guidelines, but states are sensitive to federal encroachments that carry any more weight. In August, Department of Homeland Security Secretary Jeh Johnson proposed federal assistance for voting machine security, but was rebuffed by some state leaders. Georgia's Secretary of State, for instance, speculated whether the federal government could "subvert the Constitution to achieve the goal of federalizing elections under the guise of security."[16] There is one potentially ameliorative quality of our blockchain model, however: Votebook can work at any size. A county or state could implement its own Votebook ledger and keep other states or the federal government from participating by limiting the public and private keys it issues to the nodes in its network. In this respect, the proposed system works quite well

[14] Wallach, *supra* n. 4.

[15] Following *Bush v. Gore*, Congress passed the Help America Vote Act of 2002, creating the Election Assistance Commission (EAC) and creating $4B fund to help states purchase new voting systems. The EAC was tasked with creating and updating a list of "Voluntary Voting System Guidelines." Norden and Famighetti, *supra* n. 4 at 3; Wofford*, supra* n.4.

[16] Eric Geller, *Election Cybersecurity: Federal Help or Power Grab?* POLITICO (Aug. 28, 2016, 7:09 AM), http://www.politico.com/story/2016/08/election-cyber-security-georgia-227475.

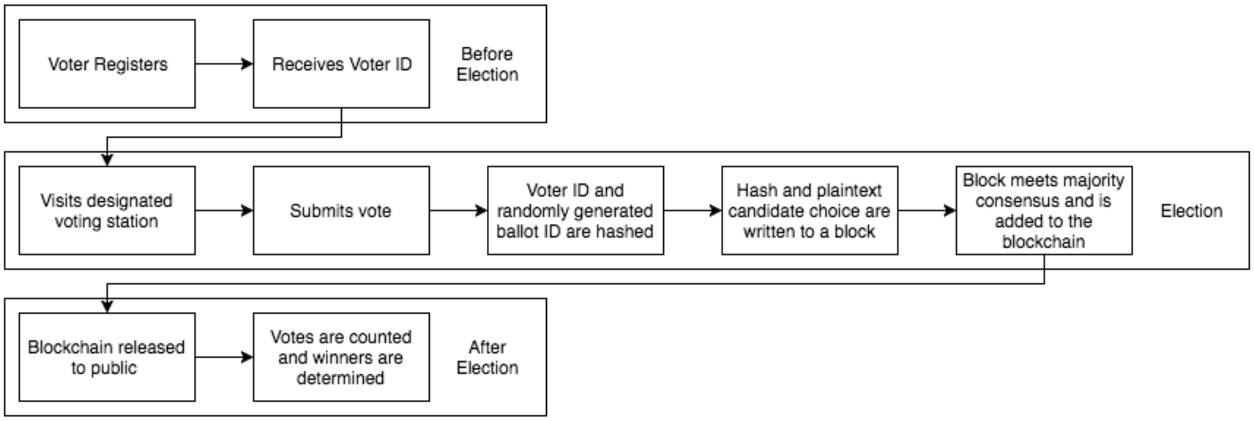
within the electoral apparatus in place today. There is no need for the federal government to issue a mandate that all states join the same blockchain: each state could produce its own count from its own blockchain.

Democratic societies embarking into the 21st century face a great challenge in finding harmony between their values and the technological tools that emerge. As one historian quipped, “Technology is neither good nor bad, nor is it neutral.”[17] The moral task for engineers is to build technological solutions that serve their community and the values of their society. Our blockchain proposal is an earnest attempt to employ strong cryptography to secure the election process. It offers an antidote to the pervasive fear that electronic voting has slipped beyond our control, that somehow our right to self-determination is under threat. Votebook not only satisfies the requirements of an acceptable voting system, it is also realistically feasible immediately, with minimal disruption of voter expectations. We have shown that we should and *can* harness the power of blockchain technology to serve democracy.

[17] Wikipedia, *Melvin Kranzberg*, https://en.wikipedia.org/wiki/Melvin_Kranzberg (accessed Sep. 29, 2016, 2:51 PM).

Appendix: Proof of Concept

Process Diagram


Voter Registers
Receives Voter ID
Before Election
Visits designated voting station
Submits vote
Voter ID and randomly generated ballot ID are hashed
Hash and plaintext candidate choice are written to a block
Block meets majority consensus and is added to the blockchain
Election
Blockchain released to public
Votes are counted and winners are determined
After Election

Detailed Process

Before election

Voting stations initialized after being validated (outside the system)
Test VPN connections
Voting stations generate private/public key pair using TMP if possible
Share public key with central admin

> Central admin validates the voting station is authorized (outside system)
> Compile public keys received
> Send table of public keys to each voting station

Voting station receive table of public keys from central admin
Sign table, send back to central admin

> Central admin validates each voting station's signature
> Appends each signature to the table of public keys
> Sends signed table of keys back to each voting station
> Generates and exchanges symmetric key for all stations to use on voter ledger
> Central admin indicates start and stop times to the election

Voting station receives database of registered voters from outside the system
At start of election, take an image and submit to central admin

During election

(Processing a voter)

```
During designated election period,

For each voter, receive voter’s ID
If voter ID is not registered to vote here,
    Reject voter
If hashed voter ID is on local database of who has already voted here today,
    Reject voter
If hash of voter ID is on distributed voter ledger,
    Reject voter
Receive voter’s selection
If duress indicated, copy voter ID, ballot ID, vote, timestamp to local database
    While hash of random ballot ID + voter ID does not contain duress code,
        Generate random ballot ID
Else, while hash of random ballot ID + voter ID contains duress code,
    Generate random ballot ID
Concatenate ballot ID with voter ID. Hash the value = hashed ID combo
Add hashed ID combo with plaintext voter choice or each election to buffered partial block for each corresponding election blockchain
Add hashed voter ID to buffered partial block for the voter blockchain
Add hashed voter ID to local database of who has already voted here today
Print ballot ID to voter-facing printer
Print hashed ID combo with plaintext voter choice to station admin printer
Repeat for all new voters

(Proposing a new block)
If it is time to submit votes in any of the elections, make block:
    Randomly reorder the rows in the buffer of votes and hashed ID combos
    Provide voting machine’s unique identifier and a timestamp
    List values in the election buffer and hash of previous block on the ledger
    Hash the above = block hash
    Sign the block hash
Propose block

If it is time to submit voters, make block:
    Randomly reorder the rows in the buffer of hashed voter IDs
    Provide voting machine’s unique identifier and a timestamp
    Use symmetric key to encrypt hashed voter IDs
    List encrypted hashed voter IDs and hash of previous block on the ledger
    Hash the above = block hash
    Sign the block hash
Propose block

(Receiving a new block)
If another voting machine proposes a block of votes,
    If public key on public key table decrypts the block hash to the correct value
        If hash of previous block on the ledger matches my hash of same
            Then add the votes on that block to the ledger
        Else reject
    Else reject

If another voting machine proposes a block of voters,
    If public key on public key table decrypts the block hash to the correct value
        If hash of previous block on the ledger matches my hash of same
            Then use symmetric key to decrypt hashed voter IDs
            Add the voters on that block to the voter ledger
        Else reject
    Else reject


(Integrity check)
Periodically,
For each signature attached to public key table,
    Decrypt signature using corresponding public key
    If decrypted signature not equal to the hash of the table,
        Public key table corrupted, check failed
Check passed

If check failed
    Request new copy from the network
        Perform integrity check on each table received
        Discard tables that fail check
        If more than one passing table,
            Table that can validate most signatures on current blockchains

If node requests copy of blockchain,
    If local table passes integrity check,
        Send new table to requesting node
```

After Election

```
Submit buffer of duress votes to central admin
Take an image and submit to central admin
(At this time, a copy of the blockchain will be released to the public, and the machines will be taken offline)

            Central admin releases list of voting machine public keys
            Once reports of coercion have had a chance to be investigated, coercion code will be released to public so blockchain can be counted correctly

End
```