Netvote: A Blockchain Voting Protocol

Technical White Paper

Jonathan Alexander
jalexander@netvote.io

Steven Landers
steven@netvote.io

Ben Howerton
ben@netvote.io

June 22, 2018
Version 1.12

Abstract

This paper lays out the technical vision for Netvote, a decentralized blockchain voting protocol. Using decentralized applications (dApps) built on the open source Netvote protocol, Election Administrators will be able to set election policies, create ballots, manage Voter registration policies, and open and close voting. Voters will be able to remotely register, access ballots, cast votes, and check their own vote records on the blockchain. Observers will be able to securely store election observations, and all participants will be able to tally election results directly via the blockchain. Netvote will provide reference dApps and developer libraries for the blockchain voting protocol to enable organizations and third parties to develop voting solutions and integrate blockchain voting into existing systems.

The first version of the open source Netvote protocol will be available on the Ethereum public blockchain. The Netvote protocol will be extended to other blockchain platforms in the future so that election bodies can choose the platform that best meets their needs. The Netvote protocol will also be integrated to multiple identity providers (IdPs) and IPFS storage systems. Elections and dApps developed using the Netvote protocol will be portable across all the integrated blockchains, IdPs and IPFS systems.

The Netvote protocol will also include an option for elections to deliver token rewards to Voters when they vote. Voter rewards can be used to incentivize participation and this capability will provide a way for elections to involve community or commercial sponsors to creatively cover election costs.

This paper covers:

- Election roles
- Election smart contracts
- Ballots
- Voter registration
- Vote transactions
- Tallying results
- Election observation, review and certification
- Future enhancements

Utilizing the Blockchain for Elections

The ability of a blockchain to secure online transactions into a decentralized and distributed ledger can be applied to elections by reducing key steps of the voting process, and most importantly votes themselves, to blockchain transactions enabled through a voting protocol. Every vote, once cast, is secured in the ledger and once committed is irreversible. The decentralized architecture of a blockchain and the mechanisms for transactions can also help ensure that votes can be gathered across wide geographies at scale with security across the network, and that multiple parties including Voters can audit the results.

The blockchain can also be used to generate and track tradeable cryptographic tokens. A blockchain-based voting protocol could be used to distribute tokens as a reward to incentivize Voters.

Netvote has chosen to initially deploy its voting protocol on the Ethereum blockchain application platform, utilizing the public Ethereum blockchain ledger. Using a public ledger will provide further confidence and trust for elections, since all election results can be reviewed and audited by voters and public observers, and voters can have high confidence that the blockchain itself is not compromised. The Ethereum core development team has committed to a roadmap of scalability and performance that can support the future needs of the network including cost efficiencies of a worldwide voting system. Netvote intends to support other blockchain application platforms, both private and public, in the near future so that election bodies can choose the blockchain that best satisfies their needs.

Open Source Applications

As described in more detail throughout this paper, the Netvote project will provide three open source reference applications along with developer libraries built on the Netvote protocol. The four applications discussed in this paper are an Administration dApp (Admin dApp) for Election Administrators, a Voter dApp for individual Voters, an Observer dApp for election Observers, and a Tally application for tallying election results.

An Admin dApp will allow validated Election Administrators to utilize the protocol election administration capabilities of the Netvote protocol to set election policies, create ballots, establish registration rules, and open and close voting. The Admin dApp will also allow the Election Administrator to deploy an election for testing before it goes live (using a test network).

A Voter dApp will utilize the protocol capabilities for individual Voters to register and vote and check results. In the future a Voter dApp may be integrated with devices such as biometric readers for Voter identification. A Voter dApp may also provide a built-in wallet capability as one option to receive and retain Voter rewards. A Voter dApp may run via a browser, a smartphone, or a hardened terminal.

An Observer dApp will utilize the protocol capabilities for Observers to log election observations with entries on the blockchain ledger to ensure that observations will be maintained, accessible, and tamper-proof. Election observations might include digital photographs, video and audio files, and notes all of which can be captured and stored through the protocol into a decentralized file system with references on the blockchain.

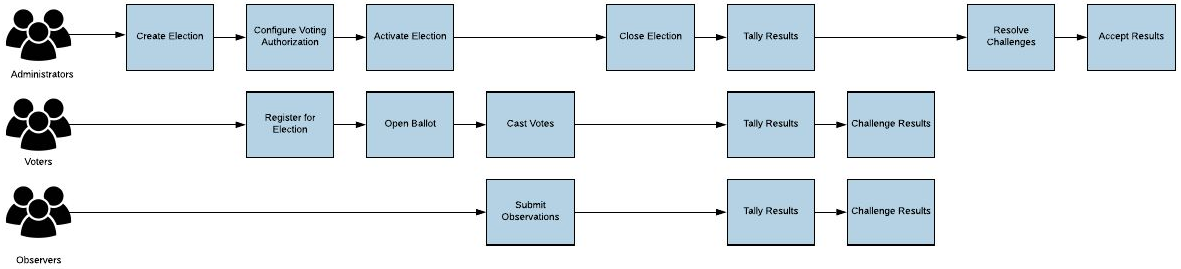
The Netvote protocol will be designed so that applications can hide complexity from users. Election Administrators and Voters and Observers will not be required to have knowledge of blockchain accounts

or to operate any pre-existing account except in specialized situations as described later. Individual Administrators and Voters and Observers will not be required to maintain wallets or pay blockchain transaction fees.

Organizations will be able to customize or embed some or all of the protocol administration or voting or observation functions in their own applications through direct interaction with the Netvote protocol. In order to facilitate development of dApps and other applications on the protocol, Netvote will also provide open source libraries that organizations will be able to utilize to embed Netvote functions and interact with the Netvote smart contract APIs.

The Netvote protocol will be open source licensed under the GNU General Public License v3.0 (GPL3).

Election Roles



Elections run using the Netvote protocol will enable participation of individuals in the following roles. In some cases a single person may assume multiple roles during the course of a single election (for example an Election Administrator might also be a Voter):

Election Administrators
Election Administrators will be the individuals who manage the lifecycle of an election including specification of election type, ballot configuration, configuration of Voter registration rules, distribution of Voter registration codes (when required), opening and closing voting, managing the election review process, and certifying the election. Election Administrators will have the responsibility to pay protocol and platform fees. Election Administrators' will interact through an Admin dApp.

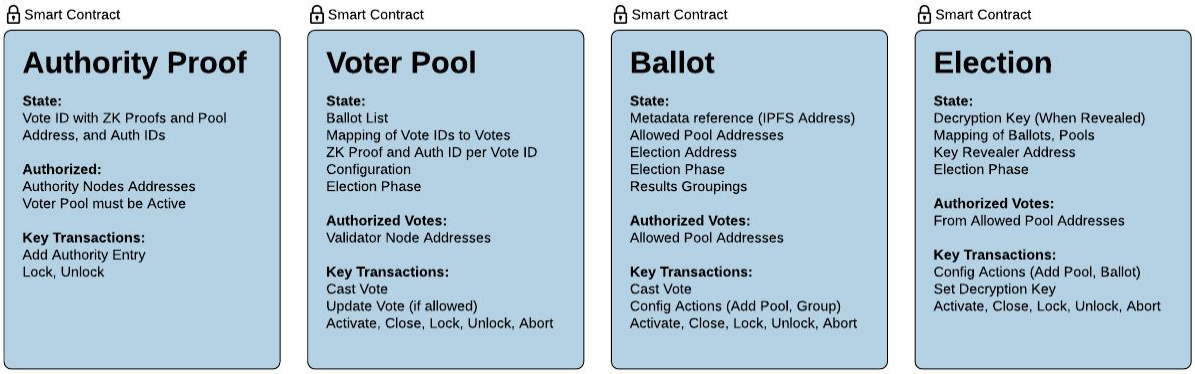
Voters
For elections to which they have access, Voters will be able to register, load election ballots, cast votes, and review their own votes on the ledger. Voters will be able to see the final tally of election results when the results become available, and may also participate in the post-voting election review process. The Voters' primary interface will be through a Voter dApp (either supplied by Netvote or developed by a third party). Voters may be rewarded for voting with tokens delivered through Netvote when they cast a vote in an election (once per election).

Observers
Observers will be able to watch elections and submit observations through an Observation dApp (either supplied by Netvote or developed by a third party). Election observations may include digital photographs, video, audio, or notes. Observers will also be able to review election results by running a

Tally application. As discussed later an Election Administrator will be able to choose to encrypt ballot choices or votes in which case the ability of the Public Observer will be limited to review after the voting closes. Public Observers will also be able to audit the blockchain transactions directly, and participate in the post-voting election review process.

Election Smart Contracts

Each election will be represented by a set of smart contracts which will be instantiated on the blockchain by an Election Administrator. Multiple smart contracts will be involved in an election: the election smart contract, one or more ballot smart contracts, and one or more voter pool smart contracts, all of which will combine to control an election, ballots, Voter registration, and voting. Further details of the election smart contracts are provided in the diagram below.



The Netvote protocol will initially support three types of elections, and each type of election will have its own set of smart contracts. The three initial election types will be:

- **Open Elections**: anyone may vote
- **Private Elections**: only authenticated and authorized Voters may vote
- **Token-Holder Elections**: a specialized type of Private Election where only Voters who operate accounts that have a balance of a designated compliant token may vote

The planned Voter registration process, which covers the necessary steps for authentication and authorization, is described in more detail in a later section.

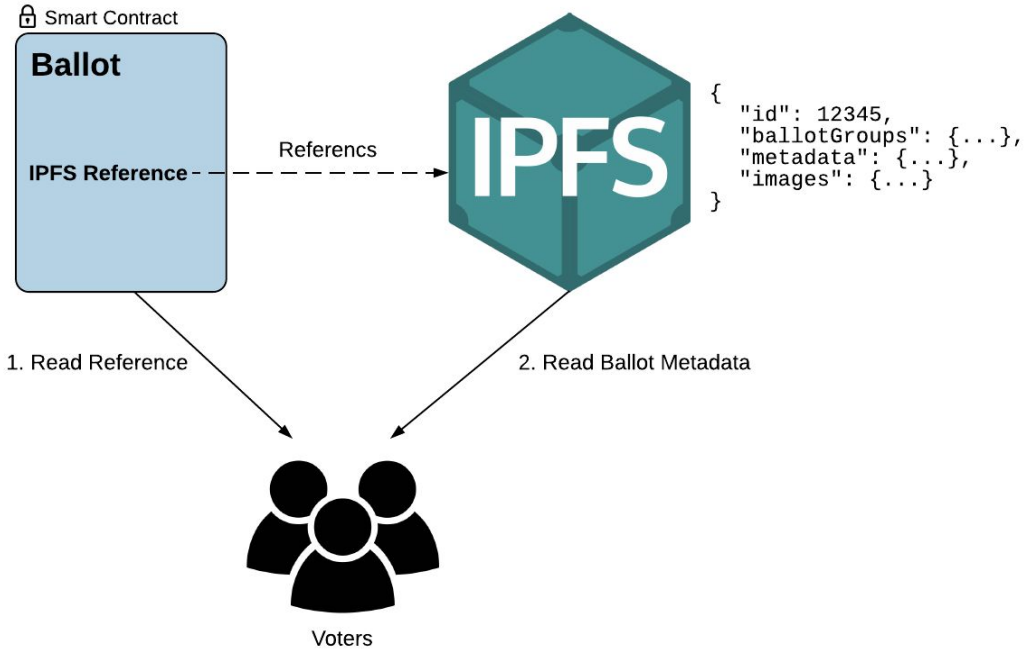
Token-Holder Elections will have the special property where votes for each Voter are weighted by the amount of a designated token they own. The designated token will be a native token on the blockchain platform which the Netvote protocol is utilizing for the specific election. For example, if the election is running on the Ethereum blockchain then any ERC20 token may be used as the designated token. If Voter A owns 3.5 of the designated election token and Voter B owns 1.75 of the token, then Voter A's vote will be weighted as two times as much as Voter B's vote.

In the future other types of elections and other voting options may also be supported.

Ballots

Election Administrators will be able to create election ballots utilizing protocol APIs accessed through the Admin dApp. Each ballot will have an associated smart contract and contain a set of choices with a set of candidates for each choice, each of which may be accompanied by text or multimedia. Multilingual ballots will be supported, with text and multimedia for all choices and candidates grouped by language codes.

As depicted in the diagram below, the text and multimedia contents of each ballot will be stored in a decentralized file storage system that provides immutability such as the Inter-Planetary File System (IPFS), Filecoin, or Swarm. By default the Netvote protocol will provide each election with access to an available file storage system which is fully integrated with the Netvote protocol and the associated dApps, and for which all necessary payments will be fully covered and managed within the protocol. Election Administrators will have the option to utilize an alternative storage network by updating storage provider information on the ballot smart contract which will be configurable through the protocol and the Admin dApp. By choosing a separate storage network the Election Administrator may be required to separately fund the storage system payments.



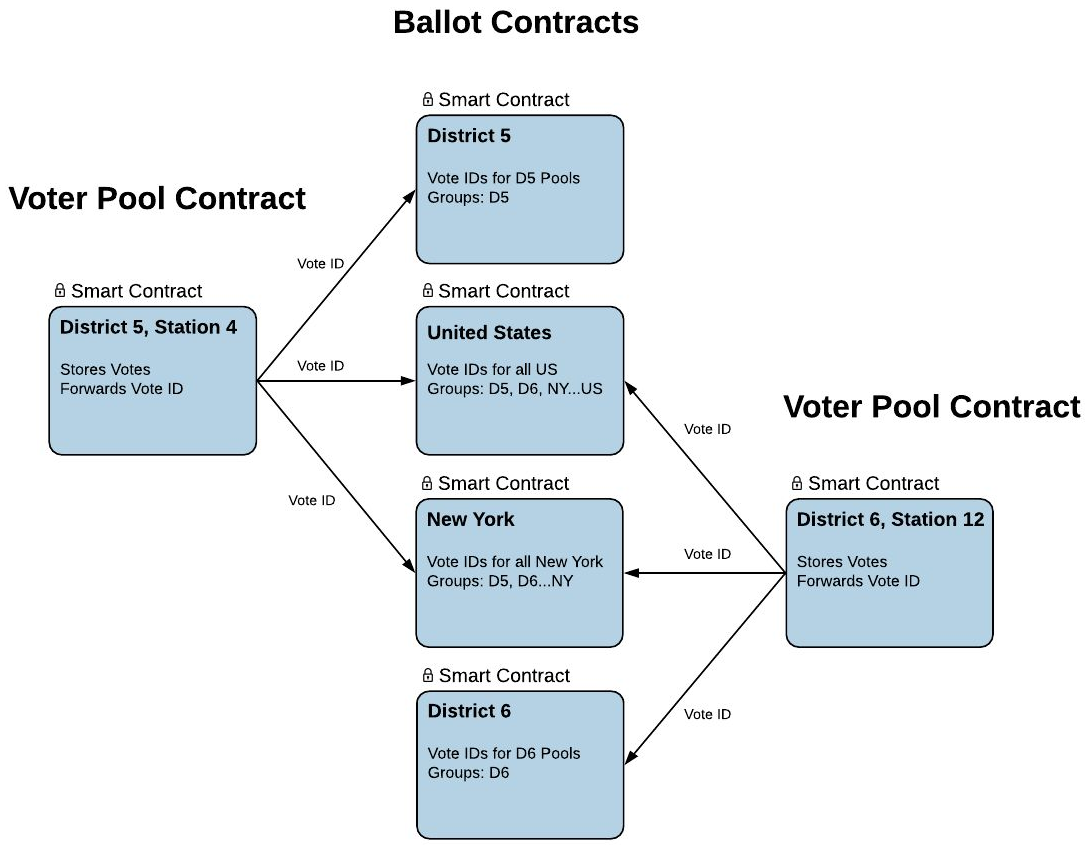
Each time the ballot details are edited for an election through the protocol using the Admin dApp, the updates will be sent to the storage system which will then generate a new reference address for the updated ballot details. The Admin dApp will update the ballot reference address with a transaction on the ballot smart contract. Once voting begins, the ballot smart contract will not allow any further changes to the ballot reference address, thereby making the ballot details permanent.

A Voter dApp will be able to access the ballot for an election directly from the storage system using the system provider information as well as the ballot reference address read from the ballot smart contract. When using an alternative storage system provided by the Election Administrator it may be necessary for the Voter or a Voter dApp to present additional credentials to access the ballot details. Such capabilities may be supported natively in a Voter dApp in the future; until then customizations may be required to integrate with alternative storage systems.

Multi-Tiered Ballots

Each ballot smart contract will reference one ballot. Multiple ballots may be listed together in a specific order through a voter pool smart contract which is discussed in more detail here and in later sections. The voter pool smart contract will be the contract through which Voters register and vote. The voter pool smart contract will have a reference to one election smart contract and to one or more ballot smart contracts. If more than one ballot is referenced then the specific order of listing will identify the order ballots will be presented to Voters.

In a public election, for example, each polling place may be represented on the blockchain by a single voter pool smart contract, and the active ballots such as Federal, State, and Local, may each be represented by a ballot smart contract each of which will be listed in the voter pool contract. A Voter who registers at the polling place would then interact with the voter pool smart contract through a Voter dApp and have access to all the ballots (Federal, State, Local). The ballots will be presented to the Voter in the exact order as listed on the voter pool smart contract.



The configuration of the election and voter pool smart contracts will be done through protocol and the Admin dApp. It will be necessary to record on each election smart contract which ballot and voter pool smart contracts are included in the election, and the election and ballots will also need to be referenced within each voter pool contract. The Admin dApp will provide capabilities to simplify the configuration process (for example the ability to refer to election and ballots and voter pools by text names as opposed to account addresses) and will also provide the ability to view ballots and test voting before going live.

Voter Registration

As described above, the Netvote protocol will support both Open and Private Elections, and a third type for Token-Holder Elections. Voters will register and interact with a voter pool smart contract via a network of Authority and Validator nodes using a Voter dApp. An Authority node will ensure authentication and authorization and in combination with a Validator node will provide additional capabilities to protect Voter anonymity and strengthen coercion resistance. The voter pool contract will be associated to the election smart contract and to one or more ballot smart contracts.

Authority and Validator nodes will be off-chain elements. In the original implementation of the Netvote protocol these will be standalone open source components, but the Netvote project roadmap will contain plans to implement these as decentralized side-chains to the election blockchain which will further ensure security and scalability. There is also the potential in the future that other entities or organizations will implement their own Voter Authority systems or networks in which case Netvote may integrate with those and will not require the use of the Netvote Authority nodes.

The Netvote development team anticipates making blockchain-based voting accessible to both current and future governments and organizations and Voters. As such, it will be necessary to support current (timeframe 1-3 years) practical means of authenticating and authorizing participants, as well as to work towards future (timeframe 3 years+) means. Solutions described in this section are those that the Netvote protocol will support in year 1, with other improvements expected in the future as new identification technologies and further blockchain-based capabilities emerge and gain acceptance and accessibility. In all Voter authentication and authorization solutions, the protocol will seek to ensure these principles:

- Election Administrators can manage the approved Voter lists (ideally transparent to Voters)
- Voters' personal information and identifying credentials will never be stored anywhere
- Voters may repeat the voting process without penalty (to ensure that Voters can cast their votes)
- Votes on the blockchain will be anonymous without any traceability to Voter identity
- Votes on the blockchain cannot be traced to pre-existing blockchain accounts (maintaining anonymity and the privacy of Voter personal accounts)
- Voters will have additional protections for coercion resistance
- Voters will have the option to check and verify their own votes on the blockchain
- Election Administrators and Voters will not be required to understand or directly operate the blockchain mechanisms involved, including accounts and private keys

This section describes the initial plans for achieving these principles within the context of the supported election types.

Zero Knowledge Proofs and Anonymous Votes

The Netvote protocol will utilize homomorphic encryption and Zero Knowledge Proofs (ZKP) to prove that Voters have supplied appropriate credentials and successfully passed the checks of the election Authority before casting a vote. The use of ZKP will provide a tamper-proof audit trail of Voter registration without exposing personal Voter information. An Authority node will generate the ZKP for each Voter and a Validator node will ensure that a valid ZKP exists for a Voter before processing a vote. The ZKP and associated information will be stored on the blockchain ledger for audit purposes.

An Authority node will dynamically generate its own Prover Key (PK) and Validator Key (VK) for the ZKP, the PK will be kept private and destroyed at the end of the election, the VK will be published on the election smart contract. For each Voter, the Authority will check credentials and will then generate a Vote ID combining some element of the Voter credentials with a secret hash key generated for the election and held in an election vault, that hash key is also destroyed at the end of the election. The Authority node then generates the ZKP for the Vote ID which is subsequently submitted by the Voter to a Validator node when casting a vote.

As will be described further below, the Netvote protocol will utilize the Vote ID through Validator nodes to maintain anonymity for Voters even in elections that seek to enforce "one person one vote" rules. This will be accomplished by using Validators to record votes on the blockchain ledger using non-personal accounts for blockchain transactions and by associating the votes to the anonymous Vote ID generated by an Authority and validated by its associated ZKP. This identity masking solution that will also enforce "one person one vote" is discussed in more detail in the Private Elections section below.

Open Elections

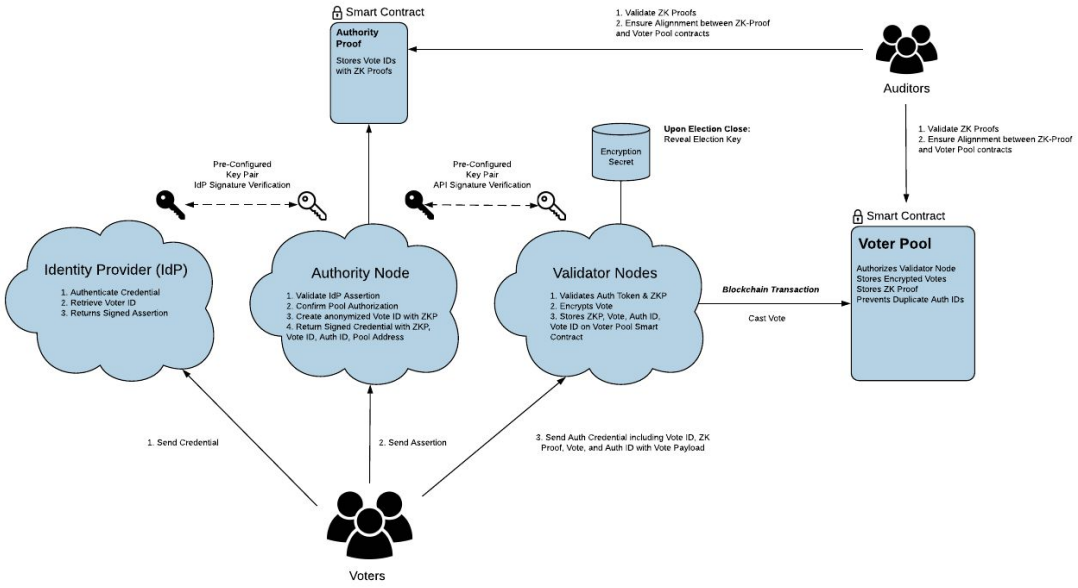
In Open Elections, anyone will be able to vote without authentication or authorization. In this case, the Voter will obtain an election access code (may be a QR code with the voter pool smart contract address) from the Election Administrator. Using the access code, a Voter will be able to access an election and ballot through a Voter dApp, and submit a vote via a Validator node without any additional identification. In the case of an Open Election, there will be no additional step for Voter authentication or authorization, so all valid vote payloads sent to a Validator node will be accepted and recorded on the blockchain.

A Voter dApp may provide some protections against repeated votes according to the election configuration. For example a dApp may locally record an indicator that might be used to block the Voter from voting again on the same election. But it should be recognized that in Open Elections the protocol itself has no absolute protection against individuals voting multiple times.

Authentication and Authorization for Private Elections

For a Private Election, the following components may be utilized to ensure that the election are limited to the Voters that have been allowed by Election Administrators to vote in the election:

- An identification system integrated with a Voter dApp to gather Voter credentials, such as:
    - a code input or QR code reader
    - a TPM card reader
    - a biometric reader
- An Identity Provider (IdP) connected to a database containing Voter IDs and credentials for Voters allowed to vote in the election (IdP may be provided by Election Administrator)
- An Authority node configured to check IdP assertions and Voter credentials to generate anonymous Vote IDs for each Voter along with associated ZKP for each Vote ID
- A Validator node to validate the Vote ID using the associated ZKP before processing a vote



The Netvote protocol will support the following steps for registration in a Private Election, although in the future the Authentication and Authorization steps may be covered or combined in solutions from other providers and integrated with the Netvote protocol:

1. Authentication
    - The Voter will enter credentials in a Voter dApp or via another application or device
    - Voter credentials will be transmitted securely over TLS to an IdP, which will check the credentials against a database of approved Voter credentials
    - The IdP will cryptographically sign and return an authentication token with claims (for example claims might stipulate expiration time) to the Voter dApp
2. Authorization, Vote ID and ZKP Generation
    - A Voter dApp will securely transmit the cryptographically signed IdP authentication token and the public ID for the Voter along with any other required credentials to prove the Voter's right to vote in an election to an Authority Node
    - The Authority node will cryptographically verify the IdP signature
    - The Authority node may verify additional Voter credentials or supplied proofs
    - The Authority node will lookup the voter pool mapping for the Voter using some part of the Voter credentials or supplied proofs
    - The Authority node will retrieve the Vote ID secret key from a Vault (the secret key is specific to the election and destroyed when voting closes as described in later section)
    - The Authority node will generate an HMAC of the public Voter ID with the secret key to generate an anonymous Vote ID
    - The Authority node will produce a ZKP for the Vote ID
    - The Authority node will produce a unique Auth ID that identifies this specific authorization
    - The Authority node will store the ZKP, Vote ID and Auth ID on an authority proof smart contract for audit purposes
    - The Authority node will return the ZKP and the Vote ID and the voter pool
3. Vote ID Validation
    - A Voter dApp will securely transmit the Vote ID and the associated ZK Proof to a Validator node
    - The Validator node will validate the Vote ID using the ZKP and the Validator Key which the Authority node previously published on the election smart contract
    - The Validator node will store the Vote ID and ZKP on the voter pool smart contract for audit purposes (voter pool records can be compared to authority proof records)

The protocol's authentication and authorization process will be deterministic, so that a Voter may repeat the process multiple times and it will result in the same Vote ID. This ensures that a Voter can repeat the process if necessary (if for example the system failed to complete the process or if the election permits a Voter to change their vote which when permitted can provide an extra element of coercion resistance). However the Vote ID will be anonymous to all blockchain observers.

The protocol APIs and the Admin dApp will provide facilities to allow Election Administrators to configure the elements of the authentication and authorization process.

Proof of Token Ownership for Token-Holder Elections

The Netvote protocol will include another option for Voter registration in what is referred to as a Token-Holder Election. In this type of election, the right to vote will be based on the Voter's possession of an identified token native to the blockchain platform prior to the start of voting (for example on Ethereum it must be an ERC20 token). An Election Administrator may create and distribute the token to Voters or will be able to specify any already distributed token so that only owners of that token may vote. Token-holders will register to vote by presenting proof of token ownership which will be done by signing the vote payload using a valid blockchain account that holds the required token. In a Token-Holder Election the value of each vote will be weighted by the amount of tokens in the Voter's account at a specified time (typically will be either when the election starts or ends).

In a Token-Holder Election, the Election Administrator will add the address of the contract for the designated token to the election smart contract and distribute a registration access code to Voters (may be a QR code with the voter pool smart contract address). The Voter will open a Voter dApp in a dApp browser which has access to an account where the Voter holds some amount of the required token. In this case the Voter will submit a cryptographically signed request to an Authority node from the token-holding account. The Authority node will verify the account address and signature, and may also validate that the account holds some amount of the required token and that those funds were not received after the voting on the election began. The Authority node will then produce and return a Vote ID for the Voter and the associated ZKP as described above. In this case the Authority node may also determine the amount of tokens held which may be used as the number of votes that the Voter can cast, and that information may also be combined with the Vote ID and the ZKP.

The Voter will then use the Voter dApp to access the election ballot and cast a vote by sending the vote along with the Vote ID and ZKP to a Validator node which will validate and process the vote.

Vote PINs and Decoy PINs for Coercion Resistance

Coercion of Voters is a serious real-world problem. Voters may be pressured in a variety of ways to cast votes by another person or group. The Netvote protocol will be designed to provide a number of features for coercion resistance. This will include the ability to support trustworthy mobile voting so that Voters can choose to avoid polling places and locales where they are physically or otherwise threatened, and the assurance that all votes will be anonymous on the blockchain ledger.

In addition, Netvote will introduce another option for coercion resistance in the form of voting PINs (not available for Open Elections). Netvote will provide a PIN Repository, though it is expected that other Identity Providers or solution providers may provide their own integrated user PIN Repositories in the future. The option to use a PIN Repository, and thereby to allow Voters to configure voting PINs, will be configurable on the election smart contract.

Voters will be able to configure voting PINs securely through a Voter dApp using the identical authorization process that will be used for voting, prior to the start of the election. The voting PINs will then be stored in the PIN Repository so they will be retrievable when actual voting occurs.

A Voter will be able to configure two types of PINs:

- A Vote PIN which when used will indicate the Voter's intent to cast a real vote
- A Decoy PIN which when used will indicate that the Voter is under coercion and that the subsequent vote, or any repetition of that vote, should not be tallied in the election results

The voting PINs will be specific to an election, and all voting PINs must be configured by Voters ahead of the election. The PIN Repository will not allow any changes to voting PINs for an election once the voting in an election begins. This will be so that Voters under distress cannot be coerced into using the system to reveal or change voting PINs during an election.

When casting a vote in an election, if voting PINs are enabled the Voter dApp will prompt the Voter to enter a PIN after the vote selections are made but before the vote is submitted to the network of Validator nodes. If the Voter did not configure a voting PIN then the Voter dApp will not prompt for a PIN.

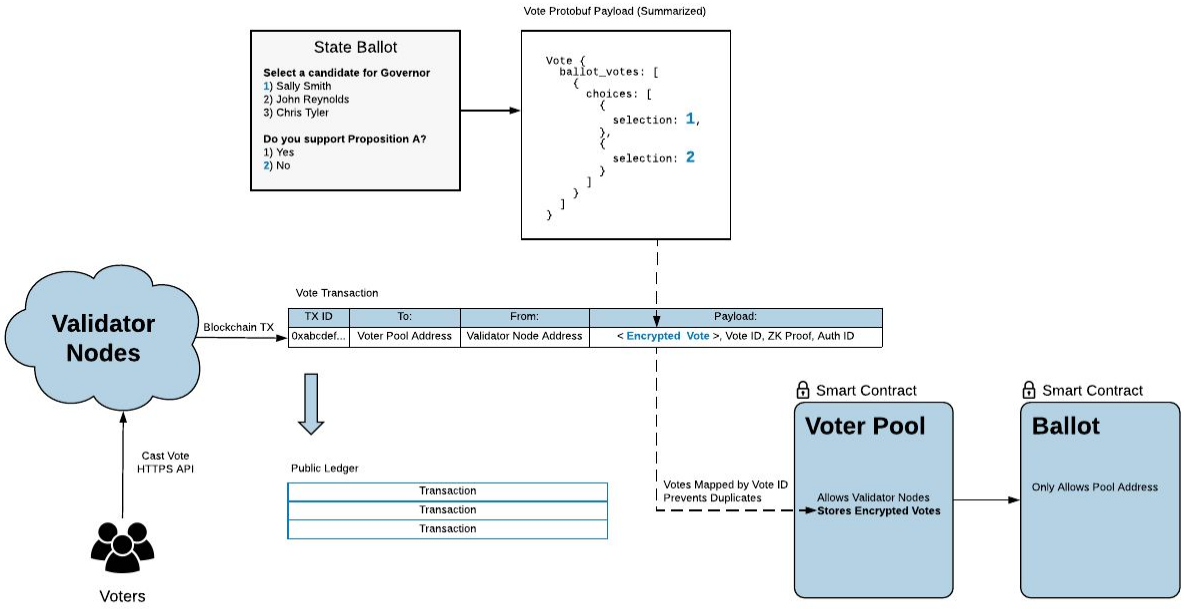
If the Voter is under coercion, the Voter will be able to signal that situation to the Netvote protocol without revealing it to the coercing party by entering the preconfigured Decoy PIN. This will be different than entering an invalid PIN which would result in the Voter dApp notifying the Voter that the PIN entered is invalid.

A Validator node will check the PIN submitted with the vote to determine if it is a Vote PIN or a Decoy PIN. If it is the valid Vote PIN, the vote payload processing will complete as described in sections above. If however the Voter has provided the Decoy PIN, the Validator node will record the vote payload on the blockchain with an encrypted flag that will be revealed during the tally to indicate that the vote should not be counted (see more on delayed reveal in a later section). In that case, any subsequent votes on the blockchain by the same Voter with the same vote payload (same ballot selections) would also be ignored in the tally. This will protect against coercion where a Voter is forced to cast votes using multiple PINs as an attempt to force use of the valid Vote PIN.

Vote Transactions

Voters will access ballots and cast votes in an election through a Voter dApp. Voters will be able to access and vote on the ballots associated with ballot smart contracts that are listed in the voter pool smart contract against which the Voter will vote. As described above, there may be a one-to-many relationship between the voter pool smart contract and ballot smart contracts, and the explicit order of ballot contracts listed on the voter pool contract will indicate the order that ballots will be presented to a Voter. Ballot contents will be read by the Voter dApp from the ballot storage system.

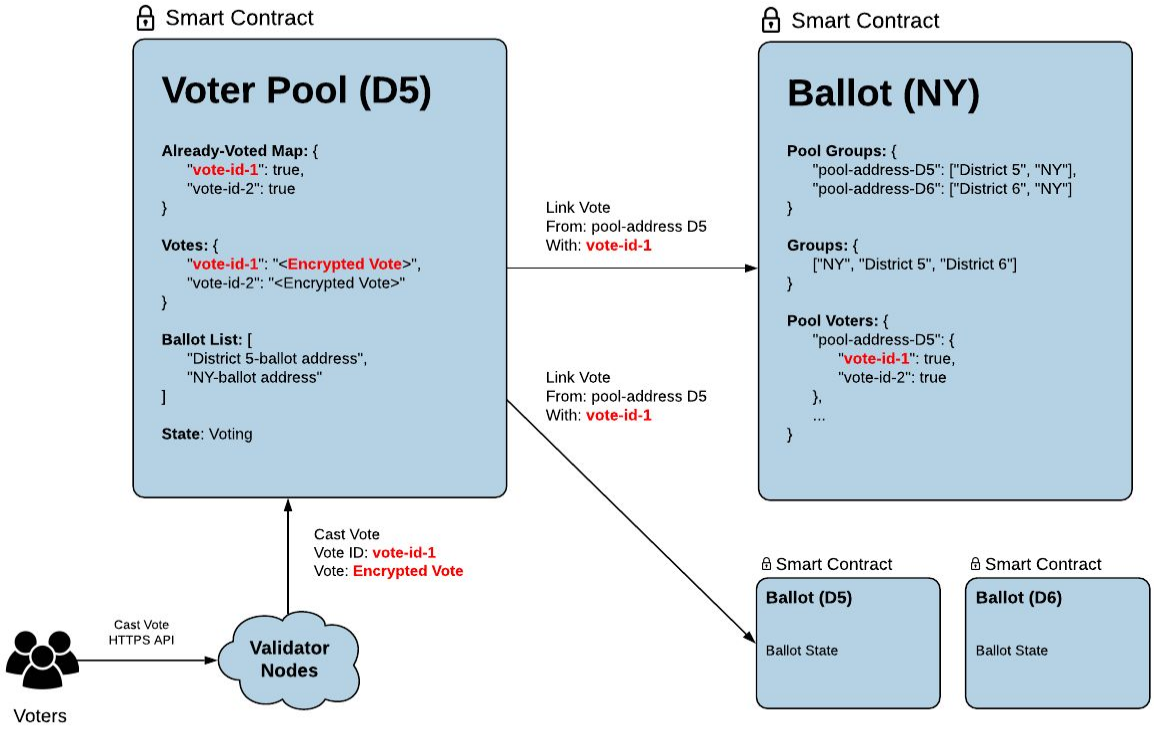
Voters will vote through a Voter dApp which will then securely transmit the vote payload (along with Vote ID and ZKP when necessary as described in the previous section) to a Validator node that is allowed to transact on the voter pool smart contract. For private elections, the Validator node will verify the anonymous Vote ID by checking the ZKP from an Authority node, and will provide further capabilities to protect Voter anonymity and election integrity such as vote shuffling and hiding (encrypting) results until voting is closed.



Having validated the Vote ID using the ZKP, the Validator node will submit the vote payload in a blockchain transaction to the voter pool smart contract with the vote payload mapped to the Vote ID.

Each vote will be stored in a transaction on the blockchain, providing the ability for each Voter to audit their own vote. Each vote will contain the Voter choices on individual ballot items in a Protobuf, all ballots and choices will be contained in the buffer string (ballot choices are fixed once voting commences). Write-in candidates will also be supported. A Voter dApp may provide the Voter access to their Vote ID which will allow them to check their vote on the blockchain at a later time.

The voter pool smart contract will store the vote values and distribute vote reference links to the associated ballot smart contracts. In the case of multiple ballots, each ballot smart contract will receive a vote reference. This will make it easier to tally results for each ballot (starting from the ballot contract). As described in more detail below, the vote string may be encrypted if the election requires that voting results must be hidden until the voting is closed.



Changing Votes

Private Elections and Token-Holder Elections may allow Voters to change votes. If permitted, the Voter will be able to submit multiple votes and each vote submitted will replace the prior vote. The mapping of votes to Voter is via the Vote ID which will be deterministic from the information generated during Voter authentication and authorization as described above. The voter pool smart contract will enforce one-vote per Vote ID or allow change vote according to the configuration of the election smart contract.

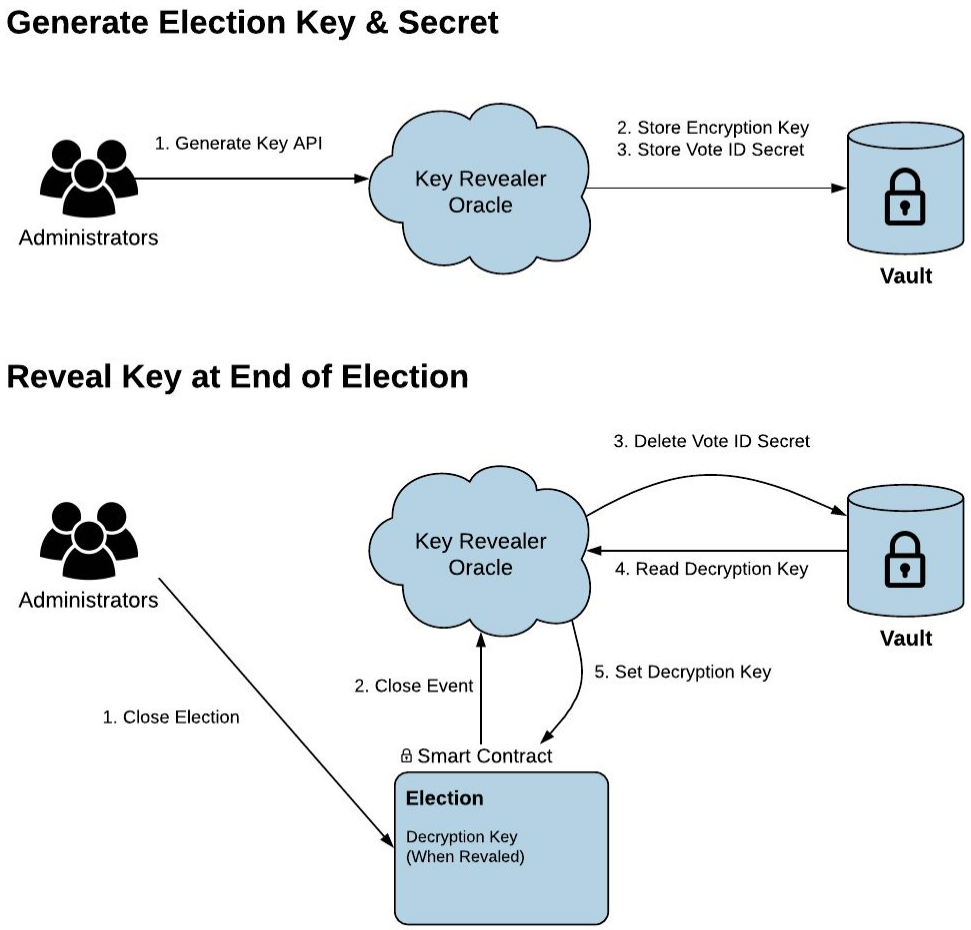
Opening, Pausing and Closing Registration and Voting

Election Administrators will have control to open and close voting for an election which will be done through transactions on the voter pool smart contracts and which may be configured using the Admin dApp. Registration and voting may be opened and closed manually or at specified times. Once voting is closed no other registration or voting transactions will be accepted by the smart contract. Election Administrators also have the ability to pause voting via a transaction on a voter pool or election smart contract (for all voter pools) which will stop all further registration and voting until resumed.

Encrypted Votes

In any election, Election Administrators may specify that votes should remain secret on the blockchain until the election is closed. During voting, the votes will remain secret, but once voting is closed all votes will be readable and auditable by all parties. To support this required "temporary secrecy" and "delayed reveal", an off-chain component referred to as the Key Revealer will be utilized to provide the following:

- Dynamically generate a symmetric encryption key used to encrypt votes for the election
- Store the encryption key in a Vault so it will be accessible only to Validator nodes during voting
- Reveal the encryption key to all parties by putting it on the blockchain once voting is closed



The open source Key Revealer will be utilized before voting begins and when voting closes, and will interact with an open source secure key storage Vault. The Key Revealer will have an associated blockchain account address which is allowed to perform transactions on the election smart contract. Before voting begins on an election, the Key Revealer will be used to:

- Generate a single election hash secret which will be used by an Authority node to generate unique and anonymous Vote IDs (as described in previous section) and the associated ZKPs which will be used to map vote payloads in the voter pool smart contract
- Generate a single election vote encryption key which will be used by a Validator node to encrypt vote payloads before storing the vote payloads on the blockchain
- Store both the election Vote ID hash secret and the vote encryption key in the Vault which can only then be read by the Key Revealer itself and by an authorized Authority or Validator node

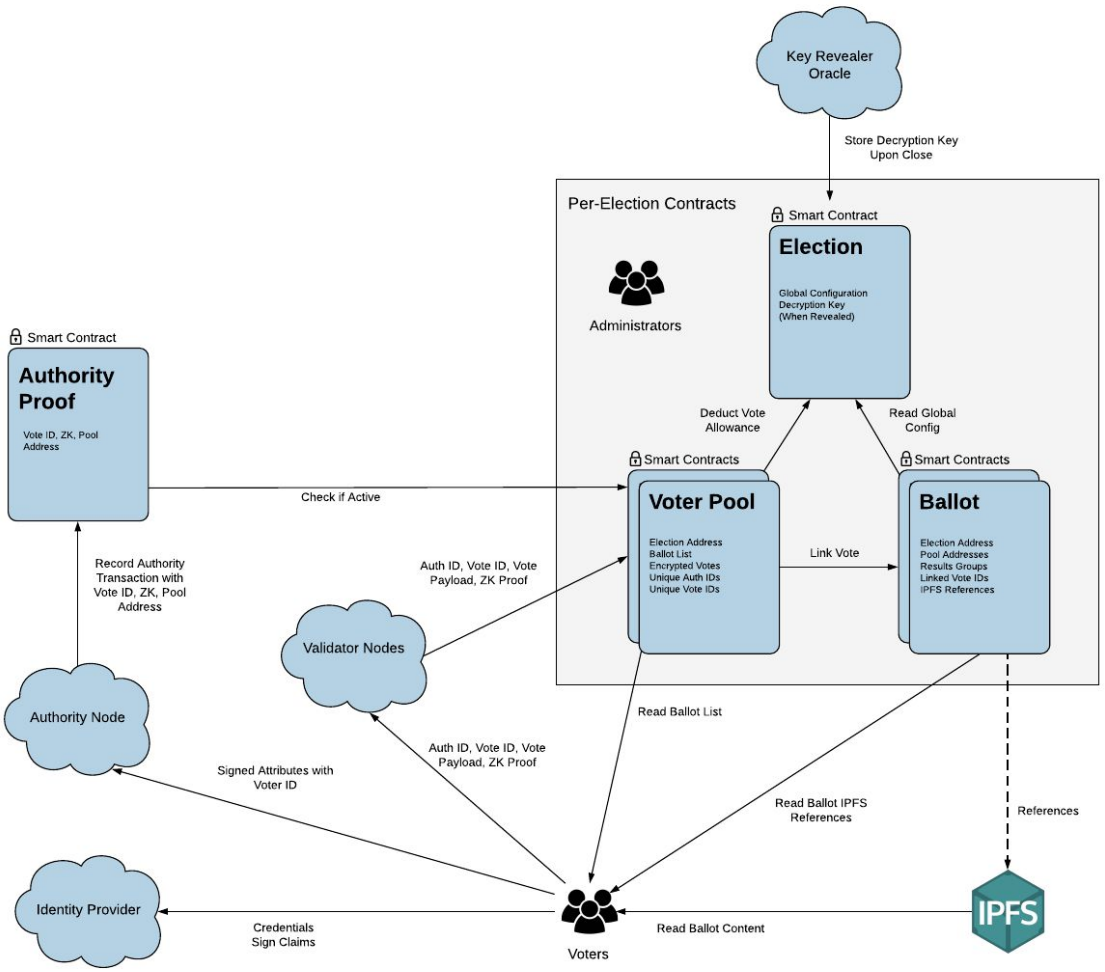
During election voting, an Authority node will utilize the election hash secret from the Vault to generate unique Vote IDs, and a Validator node will utilize the election vote encryption key to encrypt vote payloads. Election Administrators will not have access to the hash secret or the vote encryption key. When voting is closed (as indicated in all voter pool smart contracts listed in the election smart contract), then the Key Revealer will perform two final actions:

- Destroy the election hash secret (so that no one can in the future can recalculate Vote IDs for the election using specific Voter credentials)
- Read the election vote encryption key and make it available to all by writing it on the election smart contract (so that it can be used to verify and tally the votes)

After the election vote encryption key is published on the blockchain in the election smart contract, it can then be used to decrypt the vote strings so that the entire election results can be tallied and individual Voters can check their votes on the blockchain. This can be handled by Tally and Voter dApps.

For a vote transaction to occur, the election must have a sufficient allowance of votes. A vote allowance will be created through assignment or the payment of fees and controlled by the smart contracts. If an election lacks a sufficient vote allowance or exhausts its vote allowance, the smart contracts will not accept new votes. The Admin dApp will provide capabilities to manage and monitor vote allowances.

The entire view of the protocol registration and voting process in a Private Election with encrypted votes is depicted in the diagram below.



Tallying Results

An open source Tally application will be provided by Netvote that will utilize the protocol to tally election results. Given an election smart contract address (which may be encoded in an QR code published by an Election Administrator) the Tally app will read from the blockchain and tally and output the election results which may then be downloaded and saved. The app will tally results for all ballots included in the election, determined by reading the list of ballot smart contracts from the election smart contract. The associated ballot details will be read from the ballot storage system (information of which will be recorded in the ballot smart contracts). The Tally app will also support tallying the results for individual ballots or voter pools.

Since all Netvote protocol election votes are stored on a blockchain, other parties may develop tally applications which may utilize protocol features or read directly from the blockchain.

In the case where election results are hidden (encrypted) by the protocol until the voting closes, a Tally app will not be able to tally and show the results until after voting has closed and the vote encryption key required for decrypting the votes has been published.

A Voter dApp may also provide Voters the ability to audit their own vote transaction on the blockchain. Various versions of Voter dApps are envisioned in the future which might support Voter audit. When permitted by Election Administrators, the Netvote protocol may notify a Voter dApp of the encrypted Vote ID for their vote which could be used to look up the Voter's vote on the blockchain at a later time. For elections where coercion resistance is paramount this feature may not be included, since any transaction link shared with the Voter might threaten anonymity.

Election Observation, Review and Certification

The Netvote protocol will support the submission of election observations by Observers during an election. Observations may contain text and media such as digital photos, videos, or audio. Observations will be processed by Validator nodes which will store media on decentralized file storage system and then record observation records with links to the media files onto the election blockchain ledger. This will ensure that observations are tamper-proof, accessible, and anonymous when necessary.

After voting has closed and the tally results are available, the Netvote protocol will support a configurable period of public review for each election. The minimum length of time allocated for review will be established before the election starts by the Election Administrator and recorded in the election smart contract. During the review period, any participant or Observer will be able to file a notice of review findings or issues on an election, possibly through a Voter dApp or through another dApp implemented specifically for election review. All issue filings will be recorded on the blockchain with any associated text and multimedia details stored in a decentralized file storage system. Once created, changes to the issue text and multimedia will not be permitted to preserve the public record.

Election Administrators will be able to view and manage submitted election issues through the protocol via the Admin dApp. All issues will have a status and may have notes and comments which may be added or changed by the Election Administrator. All submitted details and actions will be published on the blockchain and publicly viewable using associated dApps.

It will be up to the Election Administrator to close the review period and certify the election, which will be recorded as the final election transaction on the blockchain. Once the protocol receives certification, the election will be closed and the smart contracts will not permit further change.

Voter Rewards

One of the unique options of the Netvote protocol will be the ability to designate a pool of tokens as a reward to Voters when they cast votes. This will be an option for Election Administrators. Any token native to the blockchain platform on which the election will be run may be designated as a reward token (for example on Ethereum the reward token may be any ERC20 token). The reward pool size and the number of tokens to reward each Voter will be controlled through the protocol by an Election Administrator and will be configurable through the Admin dApp.

In the case where the election will designate a Voter reward, the designated tokens will be distributed with each (original) vote cast by a single Voter. The token reward will be associated to the Vote ID generated for the Voter, so any individual Voter will only be rewarded once in an election the first time they cast a vote. Via the logic in the smart contracts, the Netvote protocol may transfer the reward tokens to an account address associated with the Voter or to an account from which they Voter can claim the reward in the future. There will be nothing on the blockchain that ties a vote to a Voter's blockchain account, so the vote transactions will remain anonymous.

A Voter dApp, if run on a Voter's personal device, may also be able to auto-generate a wallet which may be used to hold the Voter reward. In that case the Voter dApp may send the account address to a Validator node when sending the vote transaction, and the Validator node (after confirming the Vote ID and casting the vote to the blockchain as described in prior section) may then initiate the Voter reward transfer back to a Voter account. In the future the Netvote protocol may be integrated with other providers or offer new solutions that allow Voters to receive and then utilize token rewards.

The Netvote protocol will enable the Voter token reward as an optional way to reward and incentivize Voters, and also as a way for Election Administrators to involve sponsors to provide creative funding for elections. A sponsor might provide the reward token, which Voters might trade for discounted goods or services, and in exchange a sponsor might cover some or all of the election fees.

Future Enhancements

The following are enhancements that may be implemented in the Netvote voting network and dApps in the future. It is expected that all change proposals and release plans will be made public at regular intervals with an opportunity for open community feedback on each item and each release.

Enhancements for future consideration include:

- Support for alternative blockchains as requested or required by the community
- Support for alternative ballot file storage systems
- Extending Authority and Validator nodes into a decentralized proof-of-stake consensus network
- Integration with selected identity providers and Voter identification devices
- Support for self-sovereign or other blockchain-based Voter identification capabilities
- Integration with OCR devices for elections that require paper ballots for voter input
- Blockchain-based encryption for Vote ID and vote encryption if it becomes available
- Weighted voting for Private Elections (initially supported for Token-Holder Elections)
- Permissions, roles and separation of duties for Election Administrators on elections
- More specialized dApp functionality for government and corporate elections
- Work with third-parties to provide accessible and secure solutions for Voter rewards
- Allow for multiple reward tokens on a single election

Disclaimer

This document may contain forward-looking statements including, but not limited to, statements as to future plans that involve risks and uncertainties. The use of words such as “expects”, “anticipates”, “believes”, “estimates”, “will”, “plans”, the negative of these terms and similar expressions identify forward-looking statements. Such forward-looking statements involve known and unknown risks, uncertainties and other factors which may cause the actual results, performance or achievements of Netvote Corporation to differ materially from any future results, performance or achievements expressed or implied by those projected in the forward-looking statements for any reason.

This document does not constitute a prospectus or offer document of any sort and is not intended to constitute an offer of securities or a solicitation for investments in securities in any jurisdiction. No person is bound to enter into any contract or binding legal commitment, and no form of payment is to be accepted based on this presentation. Any agreement relating to the potential sale and purchase of cryptographic tokens or any securities issued by Netvote Corporation is to be governed solely by any purchase documents as Netvote Corporation may require and no other document (including this presentation).

No regulatory authority has examined or approved of any of the information set out in this document. No such action has been or will be taken under the laws, regulatory requirements or rules of any jurisdiction. The publication, distribution or dissemination of this document does not imply that any such applicable laws, regulatory requirements or rules have been complied with.

There are material risks and uncertainties associated with Netvote Corporation, its business and operations. Netvote Corporation and its affiliates do not make or purport to make, and hereby disclaim, any representation, warranty or undertaking in any form whatsoever to any entity or person, including any representation, warranty or undertaking as to the accuracy and completeness of any of the information set out in this document.

References

1. Votebook Proposal for a Blockchain-Based Electronic Voting System (NYU)
   http://www.economist.com/sites/default/files/nyu.pdf

2. A Smart Contract for Boardroom Voting (Newcastle University)
   http://fc17.ifca.ai/preproceedings/paper_80.pdf

3. Digital Voting with Blockchain (Plymouth University)
   https://www.economist.com/sites/default/files/plymouth.pdf

4. Blockchain-Based Digital Voting System (Univ. of Arkansas at Little Rock)
   http://www.economist.com/sites/default/files/ualr.pdf

5. Civitas: Toward a Secure Voting System (Clarkson/Chong/Myers, Cornell Univ.)
   http://www.cs.cornell.edu/projects/civitas/papers/clarkson_civitas.pdf

6. Non-interactive Zero Knowledge from Homomorphic Encryption
   (Damgard, Fazio, Nicolosi)
   https://www.cs.stevens.edu/~nicolosi/papers/tcc06.pdf

7. Bitcoin: A Peer-to-Peer Electronic Cash System (Satoshi Nakamoto)
   https://bitcoin.org/bitcoin.pdf