DeCent Vote

Manasa Hari
Software Engineering
San Jose State University

Sung-Yin Yang
Software Engineering
San Jose State University

Nihanjali Mallavarapu
Software Engineering
San Jose State University

Parvathy KannanKumarath Madom Krishnan
Software Engineering
San Jose State University

***Abstract*— This application is to provide a web-based decentralized voting application where each and every voter has a fair role to play. It uses blockchain mechanism and prevents double-voting and fraudulent voting. In this application, each voter registers with valid details and then votes to the corresponding party. Each registered voter has a unique voter ID. A set of authentication keys is generated against each voter. When a user votes, it is checked whether it is the first time, or it is already done before and it is ensured that there can be utmost only one vote casted from each voter. After passing through this verification, the vote is then recorded as done. All votes that are accepted require consensus across the network. No single node controls it and every node is an owner and each vote is peer-peer verified. Given that every recorded vote on the blockchain needs consensus on the network and the fact that it is merely impossible to manipulate too many systems at the time, the chances of fraudulent votes are very low. The voting system is 100% transparent, no central authority owns it and the voter's identification remains confidential. Also, people don't have to leave their sofa to cast their votes, but they can do it online at their own convenience. Since this application is de-centralized, it ensures high availability and data security.**

It provides a. Anonymity of voters ensures that whoever is casting a vote, they are authorized to do so.
b. Only one vote per person: No one would be able to vote more than once in the same election.
c. Data integrity: Ensures that once voted, it cannot be manipulated.

Keywords—Blockchain Platform, Hyperledger Fabric, JavaScript

INTRODUCTION

With the rise of blockchain technology, the core concept of decentralization has gradually drawn attention. In this context, the main objective of this research is to realize more convenient and secure applications through the use of blockchain technology. Currently, the service industry, such as the financial and banking industry, transmits private information through a trusted third party. However, they are facing many problems and complicated procedures. Since the blockchain technology and smart contract have the characteristics of decentralization, the researchers analyzed the architecture of the existing e-voting systems and found the integration of blockchain and smart contract into the application, which could enhance data verifiability and lower the cost while maintaining the openness and transparency of the voting.

The anonymities of voters, the security of ballot transmission and the verifiability of votes during the billing phase are the most fundamental requirements for voting. The anonymity and security can be achieved by the secret sharing scheme with Paillier's public-key cryptosystem while the verifiability of votes can be realized by taking advantage of the transparency and non-repudiation of blockchain. Voters can calculate the ballots and verify the election results on their own without a trusted third party

Application working

An application is built using the ReactJS and NodeJS. The overall front end is written in ReactJS. The user is asked to Sign up or Log in into the application using his trusted credentials, like driver's license or passport. Upon login, the user is presented to enter the details of his voter ID and to pick his candidate.

Proposed Working on Hyperledger and SmartContracts

Hyperledger Fabric is a permissioned blockchain infrastructure, originally contributed by IBM and Digital Asset, providing a modular architecture with a delineation of roles between the nodes in the infrastructure, execution of Smart Contracts (called "chaincode" in Fabric) and configurable consensus and membership services. A Fabric Network comprises "Peer nodes", which execute chaincode, access ledger data, endorse transactions and interface with applications. "Orderer nodes" which ensure the consistency of the blockchain and deliver the endorsed transactions to the peers of the network, and Membership Service Providers (MSPs), generally implemented as a Certificate Authority, managing X.509 certificates which are used to authenticate member identity and roles.[13]

Fabric is primarily aimed at integration projects, in which a Distributed Ledger Technology (DLT) is required, offering no user facing services other than an SDK for Node.js, Java and Go. Fabric supports chaincode in Go and JavaScript (via Hyperledger Composer, or natively since v1.1) out-of-the-box, and other languages such as Java by installing appropriate modules. It is therefore potentially more flexible than competitors that only support a closed Smart Contract language.

A smart contract also can be regarded as a secured stored procedure as its execution and codified effects like the transfer of some value between parties are strictly enforced and cannot be manipulated, after a transaction with specific contract details is stored into a blockchain or distributed ledger. That's because the actual execution of contracts is controlled and audited by the platform, not by any arbitrary server-side programs connecting to the platform.

In 2018, a US Senate report said: "While smart contracts might sound new, the concept is rooted in basic contract law. Usually, the judicial system adjudicates contractual disputes and enforces terms, but it is also common to have another arbitration method, especially for international transactions. With smart contracts, a program enforces the contract built into the code." By implementing the Decree on Development of Digital Economy, Belarus has become the first-ever country to legalize smart contracts. Belarusian lawyer Denis Aleinikov is considered to be the author of a smart contract legal concept introduced by the decree

Seecurity Concerns

A smart contract is "a computerized transaction protocol that executes the terms of a contract". A blockchain-based smart contract is visible to all users of said blockchain. However, this leads to a situation where bugs, including security holes, are visible to all yet may not be quickly fixed.

Such an attack, difficult to fix quickly, was successfully executed on The DAO in June 2016, draining US$50 million in Ether while developers attempted to come to a solution that would gain consensus. The DAO program had a time delay in place before the hacker could remove the funds; a hard fork of the Ethereum software was done to claw back the funds from the attacker before the time limit expired.

Issues in Ethereum smart contracts, in particular, include ambiguities and easy-but-insecure constructs in its contract language Solidity, compiler bugs, Ethereum Virtual Machine bugs, attacks on the blockchain network, the immutability of bugs and that there is no central source documenting known vulnerabilities, attacks and problematic constructs.

Distributed Ledger Technology for Decentralized applications

The distributed ledger database is spread across several nodes (devices) on a peer-to-peer network, where each replicate and saves an identical copy of the ledger and updates itself independently. The primary advantage is the lack of central authority. When a ledger update happens, each node constructs the new transaction, and then the nodes vote by consensus algorithm on which copy is correct. Once a consensus has been determined, all the other nodes update themselves with the new, correct copy of the ledger. Security is accomplished through cryptographic keys and signatures.

PROPOSED BLOCKCHAIN-BASED E-VOTING SYSTEM

There are seven roles in the system: (1) Voters(Vi) with qualification for voting; (2) Registration Server (RS) which verifies the voter's identity and provides eligible voters with voting certificate, CertðViÞ; (3) Authentication Server (AS) verifies the certificate derives from RS which issued Vi's CertðViÞ; (4) Voting Website (VWeb), the voting site of the system which is under the control of the electoral authorities; (5) Recording Center (RC) stores CertðViÞ and ballot signatures when Vi is voting; (6) Distributed Data Servers (DDS) store the encrypted coordinates of points of the selected number when Vi is voting; (7) Smart Contract (SC), a dynamic space to replace the functionality of the traditional bulletin board. It can count the ballots to enhance the credibility and reliability of the election. In the system, we assume that there are n1 voters, n2 candidates and 5 distributed data servers. Moreover, all the transmission procedures are executed via https connection.

3.1 Initial Phase Before the protocol, AS and RS have to generate their RSA-based public/private key pair (e; NÞ=ð Þ d; N and (e0 ; N0 Þ= d0 ; N0 ð Þ, respectively, where ð Þ d; N , d0 ; N0 ð Þ are the signing key and (e; NÞ, (e0 ; N0 Þ are the public key used for signature verification. h denotes the hash function (e.g., SHA-256 or SHA-3). On the other hand, RC has its Pallier-based encryption and decryption key pair (pkRC;skRCÞ:

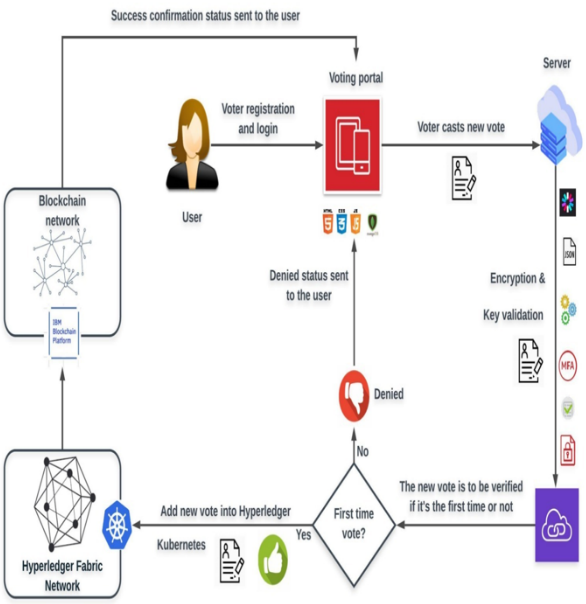
3.2 Registration Phase This phase includes two major procedures and is done off-line before the election: 306 J.-H. Hsiao et al. Step1: User code generation, Vi; 1 i n1. It proceeds by each Vi as follows: • Pick a random number t 2 Z N and generate its unique user code PIDi where PIDi ¼ hðSSNVi ktÞ: Here SSN is the social security number. Finally, send PIDi to RS for verification. Step2: Verification of Vi's identity,1 i n1, RS proceeds as follows: • Accept PIDi and issue voting certificate CertðViÞ ¼ PIDi; Sigd f g 0ð Þ PIDi to Vi if PIDi is correct. The certificate is the signature of PIDi signed by RS. • Publish the PIDi of eligible voters onto the bulletin board. Here the bulletin board is implemented by the smart contract (SC).

3.3 Voting Phase In which follows are the procedures for Vi; 1 i n1, to obtain ballot signature and personal key pair from AS, then he/she can use the ballot signature for voting. Step1: Vi provides CertðViÞ to AS and asks for a ballot signature. Step2: AS generates V0 is Paillier-based public/private key pair (pkVi ;skVi Þ and sends it back to Vi if V0 is CertðViÞ is correct. Step3: Assume that Vi wants to vote to the k-th candidate, k 2 f g 1; ...; n2 ; Vi proceeds as follows: • After receiving (pkVi ;skVi Þ, compute hð Þk and generate EpkVi ð Þ hð Þk . • Pick n2 random numbers rj 2 ZN; and generate cj ¼ mj rj; 1 j n2. Here mj means the ballot which is correspond to the j-th candidate. • Send EpkVi ð Þ hð Þk and cj; 1 j n2; to AS. Step4: After receiving EpkVi ð Þ hð Þk and cj from Vi; 1 i n1; 1 j n2; AS proceeds as follows: • First, check each mj from cj, to avoid signing on incorrect or unrelated documents and compute the hash value of each cj, (i.e., hðcjÞ). Second, sign each to generate the RSA signatures, h cj d ; by using key d. Finally, encrypt the signatures and the hash values of kj, (i.e., hðkjÞ) using Vi's public key pkVi to obtain Xj ¼ EpkVi h cj d and EpkVi hðkjÞ • Pick n2 random numbers, kj 2 Z N; 1 j n2 and compute ðEpkVi ð ÞÞ hð Þk kj and ðEpkVi hðkjÞ Þ kj . Notice that according to the Pallier's additive homomorphic property, EpkVi ð Þ hð Þk j ¼ EpkVi ð Þ hð Þk kj and EpkVi h kj kj ¼ EpkVi h kj kj • Compute Mð Þ i;j ¼ EpkVi kj hð Þ k hðkjÞ þ h cj d ; and sent Mð Þ i;j to Vi. Step5: Vi; 1 i n1; proceeds as follows: Decentralized E-Voting Systems Based on the Blockchain Technology 307 • Obtain Mð Þ i;j ; 1 j n2; then use his private key skVi to decrypt Mð Þ i;j , and to get n2 ciphertexts and verify the k-th ciphertext by using AS's public key to get only the kth ballot signatures; h cð Þk d . After voting, the candidate number k which chosen by Vi will be divided into k plaintext coordinates PCð Þ i;k ¼ ð Þ xk; yk ; 1 i n1; 1 k 5 via 3ð Þ ; 5 secret sharing scheme which can be recovered from 3-out-of-5 plaintext coordinates. VWeb will use Vi's public key pkVi to encrypt the data and store it together with PIDi in DDS. After DDS receives the coordinates, it will use RC's public key pkRC to encrypt the coordinates and ultimately announce the coordinates and PIDi; 1 i n1; by the SC for Vi to check whether the correctness of counting.

3.4 Billing Phase After voting, an event will be sent via SC to notify all Vi; 1 i n1, the recording of ballots is ready. The procedures are as follows: Step1: AS proceeds as follows: • Publish Vi's private key skVi ; and the random number rj; 1 j n2; selected by voting phase onto SC. Step2: RC publishes its own private key skRC onto SC. Step3: SC computes and proceeds as follows: • First, decrypt CCð Þ i;k ¼ EpkRC xk; EpkVi ðykÞ ; 1 k 5 and recover the plaintext coordinates PCð Þ i;k ¼ ð Þ xk; yk ; 1 k 5; by using skVi . Second, recover k from plaintext coordinates by 3ð Þ ; 5 secret sharing scheme and verify the correctness of the ballot signatures and check the value of k to see whether it's consistent with the information on the ballot signatures by mj.

3.5 Security Analysis The e-voting systems proposed in this paper meet the following security requirements: Voter qualification, protecting the anonymity of voter's identity, non-repeatable, ballot eligibility and ballot verifiability. Due to the page limitation, we will not discuss this in detail.

The architecture diagram of the application is as follows.



With its distributed ledger, smart contracts, and repudiation capabilities, blockchain is revolutionizing the way organizations do business, and the election industry is no exception. This developer code pattern shows you how to implement a web-based blockchain app using the IBM Blockchain Platform to facilitate voting and help ensure the prevention of double-voting.

Technology Stack

Programming languages & Libraries: JavaScript
Technologies: Blockchain Platform, Hyperledger Fabric,
Web Technologies: HTML5, CSS3, JavaScript, Vue, Node.JS
Miscellaneous: SOA, Agile, Git, Trello.

Acknowledgment

The authors gratefully acknowledge the contribution of Horea Porutiu for providing the guidelines on the e-vote platform. Also, we would like to thank Rakesh Ranjan and the IBM Design Team for their inputs on this topic which also helped us in research.

References

[1] Zyskind, G., et al.: Decentralizing privacy: using blockchain to protect personal data. In: Security and Privacy Workshops (SPW), pp. 180–184. IEEE (2015)

[2] Kosba, A., et al.: Hawk: the blockchain model of cryptography and privacy-preserving smart contracts. In: IEEE Symposium on Security and Privacy (SP), pp. 839–858. IEEE (2016)

[3] Shamir, A.: How to share a secret. Commun. ACM 22(11), 612–613 (1979)

[4] Rabin, M.O: How to exchange secrets by oblivious transfer. Technical report, Aiken Computation Laboratory, Harvard University (1981)

[5] Paillier, P: Public-key cryptosystems based on composite degree residuosity classes residues. In: Advances in Cryptology, Eurocrypt 1999, pp. 223–238 (1999)