International Research Journal in Global Engineering and Sciences. (IRJGES)
ISSN : 2456-172X | Vol. 2, No. 2, June-August, 2017 | Pages 67-71

DECENTRALIZED VOTING SYSTEM USING ETHEREUM BLOCKCHAIN

Dr. A N NANDAKUMAR[1], SAKTHISREE.T[2], Ms.S.P.VIDHYA PRIYA[3]
[1]Professor, [2]Assistant Professor, [3]Assistant Professor
nandakumar57@hotmail.com ,sakthisree@gmail.com spvidhya.priya@gmail.com
Department of CSE, Kathir College of Engineering, Coimbatore, Tamil Nadu, India.

ABSTRACT:

Electronic voting systems are proprietary, that is centralised by design. This means that, there's a single supplier that controls the code base, database and the system outputs and supplies the monitoring tools at the same time. It is difficult for centralised systems to gain trust from voters and election organisers. Open source, independently verifiable systems solves this issue. Voters can vote freely without even visiting the poll, also they can verify their own votes. All votes are recorded publicly on the Ethereum Blockchain and can be viewed and audited by anyone. This kind of decentralised voting systems are significantly advanced and are secure enough to prevent attacks like Man in the Middle, DDoS, eavesdropping etc. and no vote can be manipulated by intruders. Since the system is decentralised, there is no single point of failure. And, this also prevents the web pages from disappearing leaving us with 404 not found error. Though this is more advanced than the traditional system of voting, this does not completely replace the voting polls but can provide a much needed complementary voting method.

I INTRODUCTION

Bitcoin demonstrated that through the power of the default consensus mechanisms and voluntary respect of the social contract it's possible to enhance our preferred network to make the decentralised application of value-transfer system. Democracy providing an immutable, verifiable and secure online voting system to leverage the availability of blockchain as a secure transaction database. From this public ledger, voters will be able to independently audit the inclusion of their vote and the outcome of the election as a whole, while being sure that the results cannot be changed due to immutability of the blockchain.

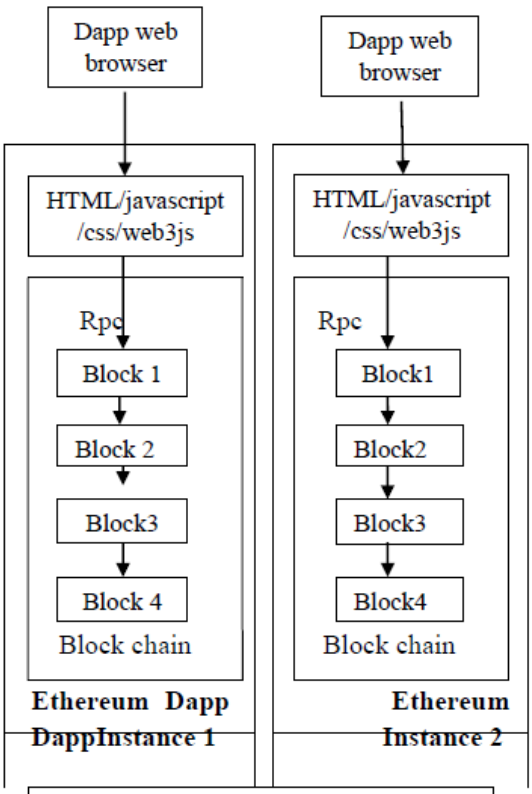

Fig 1: Working of Ethereum Dapp

This can eliminate the need for electronic voting systems which are proprietary and centralised by design, meaning that the single supplier that controls the code, database and the system outputs and supplies the monitoring tools at the same time can be eliminated. The decentralised voting system can gain the trust from voters and election organisers, that the traditional voting system cannot, as it is open source and independently verifiable.

II PROPOSED SYSTEM

Electronic voting systems are proprietary, that is centralised by design. This means that, there's a single supplier that controls the code base, database and the system outputs and supplies the monitoring tools at the same time. It is difficult for centralised systems to gain trust from voters and election organisers. Open source, independently verifiable systems solves this issue. Voters can vote freely without even visiting the poll, also they can verify their own votes. All votes will be recorded publicly on the Ethereum Blockchain and all can be viewed and audited by anyone. These kind of decentralised voting systems are significantly advanced and are secure enough to prevent attacks like Man in the Middle, DDoS etc.Since the system is decentralised, and there is no single point of failure. Though this is more advanced than the traditional system of voting, this does not completely replace the voting polls but can provide a much needed complementary voting method

Advantages:

- Votes are secure and private
- Immutable, Verifiable and the system can be accessed from anywhere at any time.

III MODULE DESCRIPTION

This application initializes a set of contestants, let anyone vote for the candidates and displays the number of votes received by each candidate. In order to do this, we can make use of the smart contract (scripting) facility provided by Ethereum. Smart contracts allow the performance of credible transactions without third parties. They help us in exchanging money, property or anything on process of value in transparent way, conflict-free way where it helps in avoiding the services of a middleman.

- Voting contract
- Web3.js
- Node.js console
- Geth
- Truffle Suite

3.1 Voting contract

The solidity programming language is used to write the smart contract for voting. The contract will contain a constructor which initializes array of candidates. The constructor is invoked only once when you deploy the respective contract to the respective blockchain.Unlike in the web world, where every deploy of our code will overwrites the old code, deployed code in the blockchain.

The voting contract consists of mapping field which is equal to an associative array or associative hash. The candidate name is nothing but the key of mapping which will be stored as type 32bytes and the retrieved value is an unsigned integer to store each and every vote count. Thus an array of 32bytes is used to store the list of candidates. The constructor which will be called once we deploy the contract to the blockchain will

pass an array of candidates who will be contesting in the election. There are a few methods that the contract contains, one is to return the total votes a candidate has received, and second method is to increment vote count for the candidate and a third to help voters buy tokens so that they will be eligible to vote.

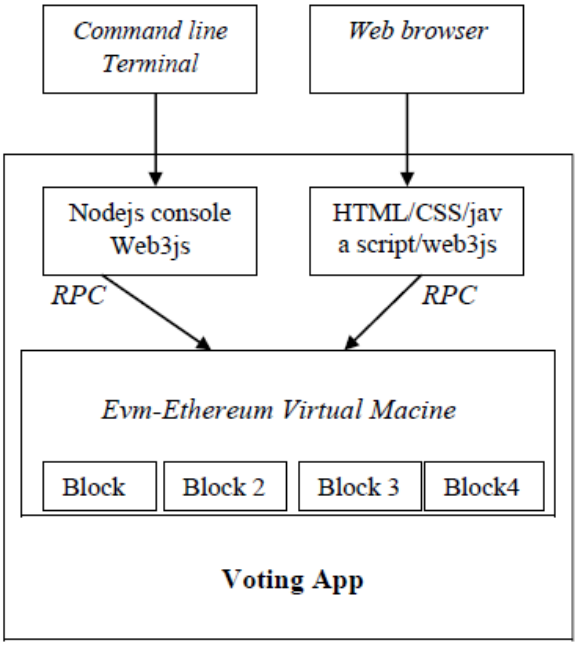

Fig 2: Visualisation of Application

These respective modules are used to compile the code to Ethereum byte code and then deploy that byte code to the blockchain. We will use this library within a node console to compile the contract. So basically, the blockchain will store the data, code and also runs the code in the Ethereum Virtual Machine

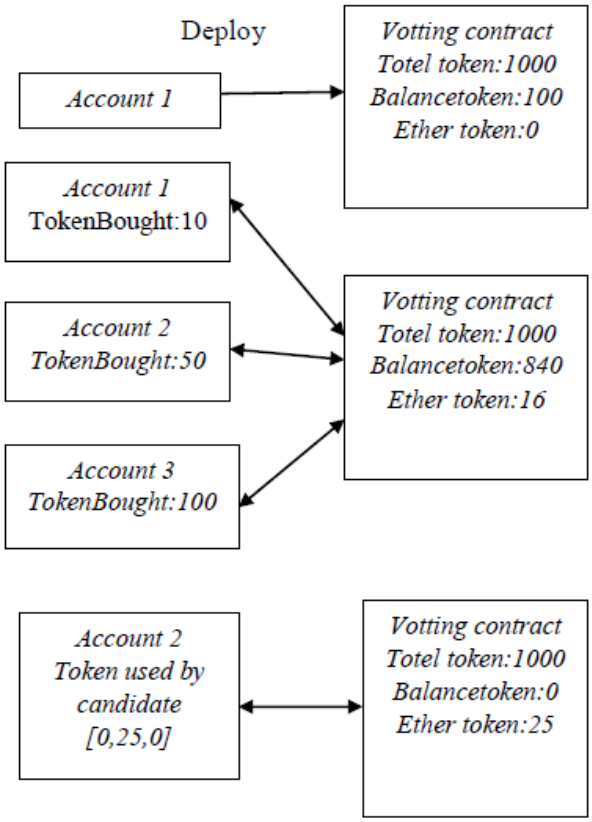

Fig 3: Visualisation of working contract

3.2 Web3js

To make the application work on Ethereum, the web3 object provided by the web3.js library is used. Under the hood it communicates to a local node through Remote Procedure calls. Web3javascript works with any Ethereum node, which exposes an Remote Procedure Call layer. The server.js file includes the voting contract setup and deployment. To compile the contract, load the code from voting.sol file into a string and compile it.

The interface or template of the contract (called ABI) which tells the contract user what methods are available in the contract. This Application binary Interface (ABI) definition is needed whenever we have to interact with the contract in the future.

A contract object is used to deploy and initiate the contract in the blockchain. The first argument is an array of candidates competing in the election. The second argument is the hash which contains,

Data - compiled byte code which we deploy to the blockchain.
From - the blockchain will keep track of who deployed the contract by using the account information. this account is a wallet account which has to been owned and unlocked before transacting.
Gas - the money given to the miners who do all the work to include our code in the blockchain. The ether balance in the 'from' account will be used to buy gas. The price of gas is set by the network.After deploying the contract, whenever it is necessary to interact with it, it's important to have the contract address and Application Binary Interface (ABI)definition.

3.3 Node.js console

Node.js console is used to interact with the contract. Every time a vote is contributed to a candidate, a transaction id is provided. This transaction id is the proof that the transaction occurred and it can be referred anytime in the future.

3.4 Geth

Geth is the client software used to download the blockchain and run the Ethereum node on a local machine. It is a Go language implementation, hence meaning Go-Ethereum for Geth. It enables us to sync with the test net or the main net blockchain in order to interact with it. Once Geth is initiated, it will connect to the respective blockchain, the peers and start downloading the blocks. While the sync is happening, the block numbers will be displayed in the output which can be used to verify the blocks.

3.5 Truffle Suite

When it comes to deploying our contract on the real blockchain, we will make use of the truffle suite for compiling and deploying our contract. It can be installed with the help of the node package manager, after which it can be used to setup the project. When truffle is installed, it brings some necessary files to run a full stack dapp. The files in the migrations directory inside truffle setup directory are used to deploy the contracts to the blockchain, while in case of local deployment we would use the **VotingContract.new** in the web3js console. The very first migration **1_initial_migration.js** deploys a contract named Migrations to the blockchain and is used to store the latest contract that has been deployed. Every time the migration is run, truffle queries the blockchain to get the last contract that has been deployed and then deploys any contracts which haven't been deployed yet. It then updates the last completed migration field in the Migrations contract to indicate the latest contract deployed. The second is the array of candidates and the third is a hash where we specify the gas required to deploy the code. The gas amount varies depending on the size of the contract.

When you compile and deploy your Voting contract, truffle stores the Application Binary Interface and deployed address in a JSON file in the build directory. We will use this information to setup a Voting abstraction. We will use this abstraction later to create an instance of the Voting contract. Once it is deployed, it can be accessed from anywhere in the world using the contract address. After

that, when the contract is tried to access we will receive a transaction receipt indicating that the contract is live and functional. The voting page will be visible at localhost: 8000.

V CONCLUSION

Fully functional decentralised voting application deployed on the Ethereum Blockchain which can be used to purchase tokens, vote for candidates using the tokens and lookup voter information by their address. This makes more voters to vote without fear of outsiders. The voter's privacy is protected and the votes are in no way manipulated. Since the system is completely decentralised, attacks like man in the middle, eavesdropping, DDoS etc. Thus preventing attackers/hackers from manipulating our votes and other information. The immutable nature of the blockchain helps in keeping the vote information forever or a long time, enabling verification, proper validation and counting of votes.

VI FURTHER DEVELOPMENT

The proposed voting system can be improved by implementing the following

Only eligible voters-The eligibility of the voters can be verified by using information such as aadhar information to prevent malpractices.

Public Poll-The user can choose if the poll should be public or not. If the is private only people with the link can participate.

Vote Limit-Limiting the number of people that can participate and limiting the number of the votes that can be casted.

Time Limit-A time limit for the polls to accept votes.

Providing Ether-Providing limited amount of ether sufficient for the casting of votes.

REFERENCES

[1] A. Corsaro, "P2P Communication", *Proc. 7th ACM/SPEC Int. Conf. Perform. Eng.*, pp. 261, 2016.

[2] J. Kelly and A. Williams. (2017). *Forty Banks Test Blockcahin-Based Bond Trading System*. [online].

[3] I. kar.(2016). *Estonian Citizens Will soon Have the World's Most hack-Proof Health-Care Records*.[Online].

[4] H. Pigot, B. Lefebvre, J. G. Meunier, B. Kerherv, A. Mayers, and S. Giroux, "The role of Blockcahin implementation," in *5th International Conference on Simulations in Blockchain*, Slovenia, Apr. 2003, pp. 497-506.

[5] P. Bailis, J. Yang, V. J. Reddi, and Y. Zhu, "Research for practice: seb security and mobile web computing," in *Communications of the ACM*, vol. 60, no. 1, pp. 50-53, Jan. 2017.

[6] C. Treude, and M. A. Storey, "*How tagging helps bridge the gap between social and technical aspects in software development*," in *Software Engineering (ICSE) 2009*. IEEE 31st International Conference on, pp. 12-22, IEEE., May. 2009

[7] B. J. Jansen, and A. Spink, "How are we searching the World Wide Web? A comparison of nine search engine transaction logs," *Information Processing & Management*, vol. 42, no. 1, pp. 248-263, Jan. 2006.

[8] W. Jones, A. J. Phuwanartnurak, R. Gill, and H. Bruce, "Don´t take my folders away!: organizing personal information to get things done," in *CHI´05 Extended Abstracts on voting systems*, ACM, Portland, OR, USA, Apr. 02-07, 2005, pp. 1505-1508.

[9] J. Hirschberg, and C.D. Manning, "Advances in natural language processing," in *Science*, vol. 349, no. 6245, pp. 261-266, Jul. 2015.

[10] C. Fellbaum, "WordNet: An Electronic Lexical Database," *in banking Language*, vol. 76, no. 3, pp. 706-708, Sep. 2016.