Journal of Theoretical and Applied Information Technology
30th November 2021. Vol.99. No 22

ISSN: 1992-8645 www.jatit.org E-ISSN: 1817-3195

A LATENCY-IMPROVED BLOCKCHAIN IMPLEMENTATION MODEL FOR NATION-WIDE ELECTRONIC VOTING SYSTEM

APEH JONATHAN APEH[1], CHARLES K. AYO[2] and AYODELE ADEBIYI[3]

[1] Covenant University, Ota, Ogun State, Nigeria
[2] Trinity University, Yaba, Lagos State, Nigeria
[3] Covenant University, Ota, Ogun State, Nigeria
Email: [1]apeh.jonathan@gmail.com

ABSTRACT

The application of blockchain technology has been considered a breakthrough for the electronic voting system research domain, given that the technology has the potentials to fix the issues of ballot confidentiality, single point of failure, and compromise of election results integrity that have been regarded as the bane of electronic voting systems. However, blockchain technology itself is not without some concerns. Chiefly amongst these concerns are latency and scalability. This paper aims to showcase existing efforts to improve the latency of a blockchain network and to present a Blockchain Implementation Model that improves the latency concern in electronic voting systems, using the Nigeria's Independent National Electoral Commission as a case study. The study evolved a latency-improved blockchain model that showcases the latency performance of the proposed blockchain-based e-voting system. The result showed a 99.36 percent improvement on the existing blockchain-based e-voting system.

Keywords: *Blockchain, Election, Electronic Voting System, Latency, Suffrage*

1. INTRODUCTION

Blockchain has been described as a Distributed Ledger Technology(DLT), and a peer-to-peer software network[1]. It uses distributed computing and cryptography to securely host applications, store data, and easily transfer valuable digital instruments that represent real-world money [2]. Blockchain technology is an emerging technology with so many potentials. But there are a lot of concerns that are inimical to its adoption. These include scalability, privacy leakage, selfish mining, transaction malleability, high electricity cost, absence of standardization, limited interoperability among blockchain networks, and of course latency. Latency has to do with the processing time i.e. the time it takes each transaction on the blockchain to be executed. This is a very important parameter because it suggests how fast a blockchain network is. Bitcoin, for instance, takes an average of ten minutes to complete a transaction whereas, VISA takes a few seconds to complete a transaction[3]. Latency concern is therefore a major issue in considering blockchain technology adoption. Electronic voting system is one of such important use cases for blockchain adoption.

This article focuses mainly on advancing a latency-improved blockchain implementation model which is applied to electronic voting system.

The rest of this paper is organized thus: section 2 discusses the review of proposed cyberthreats solutions to electronic voting system(a device or system for counting ballots and recording votes[4, 5]), Blockchain-based Electronic Voting Systems and improving latency in blockchain networks. Section 3 discusses the methods adopted in the work and the proposed latency improvement model; section 4 presents

the results and discussion and section 5 is the conclusion.

2. LITERATURE REVIEW

2.1. Electronic Voting System and Proposed Solutions to Cyber-threats (Non-blockchain-based solutions)

Leyou et al. [6] introduced an identity-based blind signature scheme for an electronic voting system. The signature scheme is constructed in the standard model to achieve full security (a strong security model in identity-based cryptography) as opposed to the less-secure oracle model. In this work, they used the natural hardness assumption-Computational Diffie-Hellman Problem (CDH) to ensure the security of their proposed identity-based blind signature scheme. They adopted the identity-based blind signature scheme with four main algorithms, namely: Setup, Extract, Blind Signature, Verification. The resulting scheme was then applied to the e-voting system. In the e-voting system, votes are encrypted and voters are issued secret keys. At the end of the voting period, the encrypted votes were published. Voters have to share their keys to the administrator who decrypts and publishes the real votes that ultimately determine how the voting was done or even the winners. The gap with this work was that an administrator is used as an encryption authenticating authority. The implication of this is that if for any reason the administrator is compromised, the e-voting security is in jeopardy.

A related work was done by [7] to propose an e-voting system for Mexican election to eliminates human error and reduce the costs of unused material through the deployment of a client-server, RSA-based system. In their work, all polling units are connected to the public network. Those in remote places connect via cell networks. Votes are transmitted from authorized polling booths via this public network to a central server. The proposed system security is built on the public key system. It is made of four stages: an initial stage, which is where the electoral authorities appoint the representatives in electoral booths and set up public-key system; an authentication stage which handles the verification of eligible voters on the election day; a voting stage in which the voter cast a ballot which is sent to the central server; and a counting stage which handles the counting and publication of the voting outcome. The limitation with this work however, was that the client-server-based deployment model makes the Mexican EV system prone to single point of failure; also, the work focuses on correctness, authenticity and verifiability of voters without considering integrity variable like immutability of votes.

Further work reviewed was the Estonian Internet Voting System and the India's Electronic Voting Machine. [8] analyzed the Estonian Internet Voting System and discussed the vulnerability points associated with it. [8] through observations and interaction with the I-voting system, its developers, review of published artifacts from the election were able to point out the loopholes in the Estonian I-voting System. These artifacts include source codes, written procedures, watching and review of close to 20 hours of official videos that took records of the I-voting configuration, management and counting processes. [9] on the other hand, highlights some serious ways the e-voting system can get compromised and become vulnerable. This according to [9] includes tampering with software before manufacturing central processing unit (CPU), substituting look-alike CPUs and other units including circuit boards, and tempering with machine state. These vulnerabilities were illustrated using the India's EVM by implementing two demonstration attacks to show experimentally, security vulnerability areas observed in it. They called the attacks "dishonest display attack and clip-on memory manipulator attack". They built these attacks without any access to the machine source code and very limited access to the EVM only at the design and testing phases. The EVM displays are used to show the number of votes received by each candidate running for an office. In a

dishonest display attack, votes are manipulated by hidden microcontrollers attached to the dishonest display screens. The controllers intercept votes during transmission and manipulate the figures in the process. The clip-on memory manipulator was the second demonstration attack. This attack illustrated how malicious hardware is used to alter the internal state of a machine like the EVM, thereby undermining ballot confidentiality. Both attacks demonstrated how criminals can manipulate the outcome of an election easily by employing these methods of attack. The limitation with these works was that in both cases they only highlight the security vulnerability points in the Estonian I-voting system and the India's EVM without suggesting or implementing specific solutions to the risks and cyberattacks-prone areas.

Other writers like Cucurull and Lee [10] have suggested the use of QR stenography for improving the security of the e-voting system. In this work, they proposed the use of steganography to secure QR code in e-voting and by extension, enhancing the security of the e-voting system. The gap with this system is that though steganography provides extra security to encrypted data, the work allows for a single point of failure. i.e. if cybercriminals can get access to the system, the integrity, and privacy of election data & ballot respectively, can easily be compromised.

To improve the Pakistan e-voting system, Ansif & Mohsin [11], proposed an Electronic Voting Machines System (EVMs) that provides transparency, accuracy, security, verifiability, and authenticity of voting for a less expensive and faster method. The gap discovered with this system is that the EVM described uses bio-hash algorithm to encrypt voters fingerprint. Verification of the fingerprint is done by a single-node system. Considering how critical e-voting systems are, a peer-to-peer-based (decentralized) verification system would provide the robust security & trust required.

From a security, large scale, and cost-effective standpoint, Kausal, Srivatsan & Jayprakash [12] proposed an Automated Teller Machine (ATM) terminals and Micro ATMs powered Large-Scale e-voting system for conducting Government Elections. This approach implements a two-tier authentication using One Time Password (OTP) and a Random Security Question (RSQ) hence, it does not allow double voting. This work proposed taking advantage of the existing ubiquitous banking payment infrastructure to make the voting experience better. It proposed the use of the debit card PAN as a voter's unique identification number and the inclusion of extra data field into the existing banks' database which would be shared with the election management body in India. While the proposition has lots of advantages including fewer expenses for setting up polling units, ease of use among others, the system still deploys client-server architecture which allows for a single point of failure for a robust system responsible for the critical processes as national elections.

From a privacy point of view, [13] did a work that "provides the first machine-checked proof of privacy-related properties (including ballot privacy) for an electronic voting protocol in the computational model." The challenge with this work this that while it supports ballot privacy and auditability of the e-voting system, the immutability of cast votes was not considered.

2.2. Blockchain-based Electronic Voting Systems

There are quite several scholarly articles proposing the different ways that blockchain technology could be used to administer an e-voting system. While most of the design propositions are theoretical, a few others have implemented their designs. Below is the summary of some of the blockchain-based e-voting system protocols reviewed:

Nir and Jeffrey [14] posit that e-voting is one of the key public sectors to be disrupted by blockchain technology. This work introduced the application of blockchain technology to the e-voting system by drawing an analogy with its

implementation in cryptocurrency. This work highlights existing implementations of Blockchain-Enabled E-Voting Systems for community and city elections, emphasizing their scale of adoption. The gaps discovered with this work, therefore, include: the proposed blockchain-enabled Electronic Voting System is limited to only the conduct of elections at community and province levels. The system could not cater for a nationwide election. For instance, the report suggests existing blockchain-based e-voting in Mosco could not cater for the twelve million population; the one deployed in March 2018 for Sierra Leone's general elections could only be used for verification in the election and did not power the whole election.

Ahmed [15] also proposed a design for a new electronic voting system based on blockchain technology. The proposed system, rather than decentralizing every stage of the electoral process, centralizes the e-verification by having a separate voters registration database outside the blockchain network.

Dogo et al [16] reviews scholarly articles on the application of blockchain technology for secure electronic voting (e-voting); the feasibility of using blockchain technology to replace the existing manual or semi-digitized voting system in third world countries. The gap discovered here are – just like Ahmed [15], the Blockchain E-Voting System centralizes its voters e-verification instead of decentralizing it. Secondly, the proposed Blockchain-Enabled Electronic Voting (BEEV) System uses voting devices (as interfaces) that are not on the blockchain network thereby making the system prone to cyberattacks.

Also, Freya, Apostolos, Raja, & Konstantinos [17] proposed an E-Voting Protocol with Decentralisation and Voter Privacy. The protocol was designed to implement the fundamental properties of an e-voting system which include fairness, eligibility, privacy, verifiability, and coercion-resistance as well as offering a degree of decentralization and allowing for the voter to change/update their vote (provided it's within the permissible voting period). The paper also highlights the merits and demerits of using blockchain for such a proposal from the standpoint of both development/deployment and usage contexts. The limitation of this work is that the protocol has a central authority (CA) that manages the identity of the voters. Such implementation is not only prone to a single point of failure (maybe from DDOS) but undermines the integrity of the voters' information and by extension the voting result.

Rifa and Budi [18] proposed the recording of votes using blockchain algorithm from every place of election. This work modeled the Bitcoin system but unlike it which uses the Proof of Work consensus algorithm, the work proposed a method based on a predetermined turn on the system for each node in the proposed blockchain network. The work suggests a design flow based on verification, get a turn, update the database, create a new block, and broadcast to get votes securely registered and eventually broadcasted. While the work considers large-scale elections, it is not properly adaptable as the number of polling units increase (i.e. as more nodes are added to the network). This is because the proposed system utilizes a take-turn scheme for its nodes. This means the more the nodes increase with polling units, the longer time it would take for each node to add their votes for eventual broadcast.

Friðrik, Gunnlaugur, Mohammad, and Gísli [19] evaluate the potential of distributed ledger technologies for national election through the description of the election process and the implementation of a blockchain-based application, which betters the security and decreases the cost of hosting a nationwide election. In this work, they proposed a blockchain architecture built on Go-Ethereum permissioned Proof-of-Authority (PoA) to achieve faster transactions through a consensus mechanism that is based on identity as stake.

The proposed work identifies two sets of nodes, the district, and the boot nodes. The district nodes represent each voting district. They autonomously interact with the "boot node" and manages the life cycle of the smart contract on them. They are also responsible for verifying vote casts before they are appended to the blockchain [20, 21, 22] On the other hand, the boot nodes are hosted by institutions with permissioned access to the network. They help the district nodes to discover each other and communicate quickly. As a result, they do not keep any state of the blockchain and are on static IPs. This work centralizes the identity management of the voters which undermines the

integrity of the voting result. This is because the voting outcome is as good as the voters' data.

2.3. Improving Blockchain Latency

To the best of our knowledge, compared to scalability of blockchain, only a few works have been done on how to improve the latency of a blockchain network. Some of the works include:

i. Modification of the consensus algorithm: A consensus algorithm is a strategy that determines how a blockchain network works to achieve a consensus on a transaction. It is what determines the response time, number of transactions per second, etc of a blockchain network. A change in the consensus algorithm among transaction validators to practical Byzantine fault tolerant (PBFT) types can help improve the latency of a chain [23]. An example of a project that implements a bitcoin-like protocol but rather than use Proof of Work (PoW), used PBFT is the ByzCoin. ByzCoin is a Byzantine consensus protocol (built on CoSi) that takes advantage of scalable collective signing to commit Bitcoin transactions irreversibly within seconds. ByzCoin has capability to achieves Byzantine consensus while preserving Bitcoin's open membership by dynamically forming hash power-proportionate consensus groups that represent recently successful block miners [24].

ByzCoin makes use of communication trees for the optimization of transaction commitment and verification under normal operation while guaranteeing safety and liveness under Byzantine faults, up to a near-optimal tolerance of f faulty group members among 3f + 2 total. It lessens double spending and selfish mining attacks by producing collectively signed transaction blocks within one minute of transaction submission. Its tree-structured communication further reduces this latency to less than 30 seconds. With these optimizations, ByzCoin achieves a throughput higher than Paypal presently handles, with a confirmation latency of 15-20 seconds.

ii. Scale Nakamoto consensus by increasing either the block frequency or block size [25]. The challenge with the resulting systems, however, is that they suffer from the same security shortcomings as Bitcoin.

iii. Blinkchain: this is a technique that decreases consensus latency while maintaining security in a Byzantine fault-tolerant scenario. Blinkchain is a latency-aware blockchain that provides bounds/limits on consensus latency. Like other latency-improving protocols, it retains sharding as a technique to achieve horizontal scalability. The difference however is in the way validators are appointed into a shard.

To achieve low-latency consensus among validators, the blinkchain uses a technique called Crux (a general framework to build locality-preserving distributed systems) [26] which enhances scalable distributed protocols with low latency. At the same time, we maintain the security guarantee that an adversary cannot take over a particular shard by focusing its efforts (e.g., proof of work) in the vicinity of the shard.

3. METHODOLOGY

Here we present the processes and methods followed to design and implement the proposed latency-improved, nationwide blockchain-based model for Electronic Voting Systems. In this work, the organization and conduct of elections in Nigeria is considered hence, Nigeria's election management body, the Independent National Electoral Commission (INEC) is used as a case study.

Figure 2 represents the network view of the proposed model. It is made up of two logical blockchains, namely, the state and the federal blockchains. The essence of these logical blockchains is to better handle the latency concern in querying the blockchain. While the State blockchain is made up of all the Polling Units (PU) in each state and the State Miner(node), the federal blockchain constitutes of the 36 state miners (each representing the state in Nigeria) and the Federal Capital Territory (FCT), Abuja.

On the one hand, the state miners are full nodes, on the other hand, the PU nodes are lightweight. A full node has a complete copy of the distributed ledger, validates new vote blocks, and verify transactions (in this case, the cast votes). The distributed ledger contains information about all registered voters and cast votes in the country.

For latency consideration, PU nodes (i.e. lightweight nodes), do not store a full copy of the

blockchain. They have a copy of the block header (i.e. metadata) which they use to stay updated with the main blockchain and to verify transactions.

With the Web3 API, a copy of the blockchain with only eligible voters in every state is made available at the respective state PUs. Communications between state miners and the PUs are digitally signed. Every PU sends a payload that is signed with its public key. With this, voters can be verified during voting before they cast their votes. The cast votes are sent as a payload which is digitally signed with each PU public key. If peradventure, the PU loses connectivity to the blockchain, voting could be done offline, but verification gets done online later once connectivity is reestablished. In a double voting scenario, the system could allow such wrong actors to vote offline but when the PU connects back, such votes would be invalidated because they had voted.

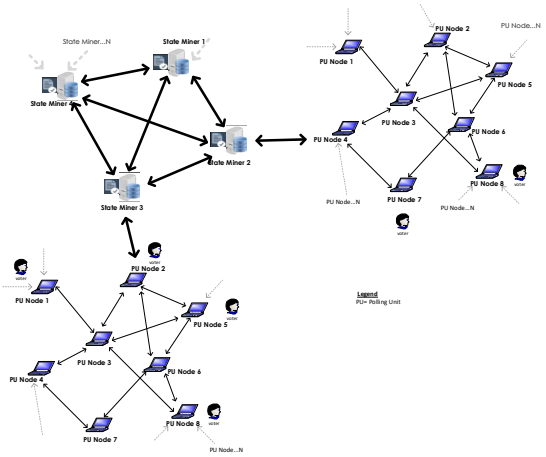

Figure 2: Network view of the Blockchain-enabled Model for Electronic Voting System

3.1. Model Setup

The following steps were taken to set up the proposed latency-improved blockchain-based electronic voting system for national election: First Node set up on the private blockchain, network creation, smart contract deployment, polling units (PU) module deployment.

3.1.1. First Node Setup

A blockchain is a collection of computers with duplicate records of data, all communicating and synchronizing over peer-to-peer communication channels. These computers are referred to as nodes. To set up our private blockchain, we started with the first node. Our proposed model's proof of concept (PoC) setup has three full nodes which are called state miners. Their names are presented below, starting with the first node: eth-node1.codecty.com, eth-node2.codecty.com, eth-node3. codecty.com

These are Ubuntu EC2 nodes set up in the Amazon cloud i.e. they have installed the Ubuntu Operating System. On top of the Linux-based (Ubuntu) webserver is installed the Ethereum "Geth" application. To connect to each of these nodes, the following command is issued using a Git bash tool:

$ ssh -i key-pair.pem user@eth-node1.codecty.com

The ssh stands for secure shell. It is a secure protocol that supports the encrypted transfer of data between two computers. The key-pair.pem file is the identity file with the public key of the eth-node1.codecounty.com node for a secured connection between the source machine and the blockchain node. Ubuntu is the user with which the blockchain nodes are being accessed.

Geth is installed to set up a custom/private Ethereum Node. To install "Geth" the command below is run from the node terminal: # sudo add-apt-repository -y ppa:ethereum/ethereum

The above command adds the Ethereum repository to the operating system source list. The command following command set updates the repositories and installs Ethereum alongside its command line (CLI) environment "Geth".

```
# sudo apt-get update
# sudo apt-get -y install Ethereum
```

Usually, the block holds data and transaction records, each block has a limited size and it's chained to a pre-

existing block hence producing a chain of blocks. In our case, there is no pre-existing block so it has to create what is known as the genesis block.

3.1.2. Creating the Private Network

To create a network, first, a new node has to be created, then a second node to communicate and synchronize with the first node. Another alternative is replicating the above process with the same genesis file and chain ID. Once the node has been created, run *geth --datadir "./db" --networkid 1947 console* to initialize the node and access its console.

3.1.3. Deploying Smart Contract

A smart contract is required to be in place to have a decentralized application. It is a self-executing contract. It is deployed to autonomously enforce an agreement between multiple parties, in this case, an election. Because of the nature of the blockchain, a smart contract like blockchain transactions cannot be modified, manipulated, or revoked once it has been deployed is required.

Remix and Ethfiddle are tools required to build and deploy a smart contract. While EthFiddle is a browser-based solidity Integrated Development Environment(IDE) that allows users to write and test their solidity codes without deploying to a blockchain, R*emix*, also a browser-based IDE, generates the bytes codes used to deploy the smart contract and ABI data which used to interact with the smart contract over the RPC protocol.

When the byte code of the contract is generated on *Remix* a transaction is created and broadcasted to the network and in response, a transaction hash is generated, this hash acts as a transaction reference that can be used to get the transaction details. The contract byte data is first read from a file and is used to create a transaction object. The Chain ID is the same as the network's Chain ID. The transaction is then signed using a private key and sent to the blockchain network.

Upon propagation, a view into the transaction using the transaction hash generated above would reveal the new contract's address. This address is copied into the *"app-config.json"* file of the NodeJS application.

3.1.4. Deploying Polling Unit Module

Until this point, all we have done is setting up and deploying the state miners on the blockchain network. This is important as the PU module will need the state miner blockchain network for connectivity, voter's verification, cast votes validation and adding them to the blockchain. As represented in section 3.1, the PU is the module of our proposed model that is deployed at various polling units across the country. This module is installed and configured on all PU computing devices (Hewlett Packard laptops used for our proposed system) that will be used for the voting on the day of the election. As a proof of concept, ten laptops operated by service integrators (SIs) were used. These SIs are at different locations, tech-savvy, and well-educated.

3.1.5. Performance Impact of the Proposed Model Design

The proposed design as represented in section 3.1, is meant to improve the latency performance matrix of the blockchain-based e-voting system. The method for implementing this matrix is presented below.

From a latency standpoint, to improve the response rate (i.e. reduce latency) in querying the blockchain, the PU nodes in the proposed model design are set up with lightweight blockchain nodes. As indicated in the model in section 3.1, in Figure 2, a lightweight blockchain node is used. Unlike the full-weight blockchain node that keeps the full history of all the transactions that have ever been made on the network, the lightweight nodes only store a copy of the block header (i.e. metadata) which allows them to stay updated with the main blockchain and to verify transactions. This means a very short query response time.

Secondly, our design makes use of the Web3 API. With this API, we can determine

voters' records that can be found on any PU. Base on our design, voters are distributed base on the state of registration. This means that a voter that registered in state A can only find his records only in the same state and not in state B. As a result, voters can only vote in PUs within the state in which they registered. Although all registered voters across the federation are available on the federal blockchain consisting tof the full-weight, state miner but programmatically the Web3 API is used to make available on a PU node only the metadata of only the voters who registered in the state within which the PU is assigned. The Web3.js API is programmatically used to make available in a PU, information about only registered voters in a state in which the PU nodes are cited as against having all registered voters in the federation.

Thirdly, our design makes use of MongoDB on the PUs to cache voting transactions. This is necessary for a situation where internet connectivity is not available and to provide a temporary memory for voting data before they are transferred to the main blockchain. This is another feature built into the design to improve the model's latency. The MongoDB transaction cache is represented in Figure 3 below.

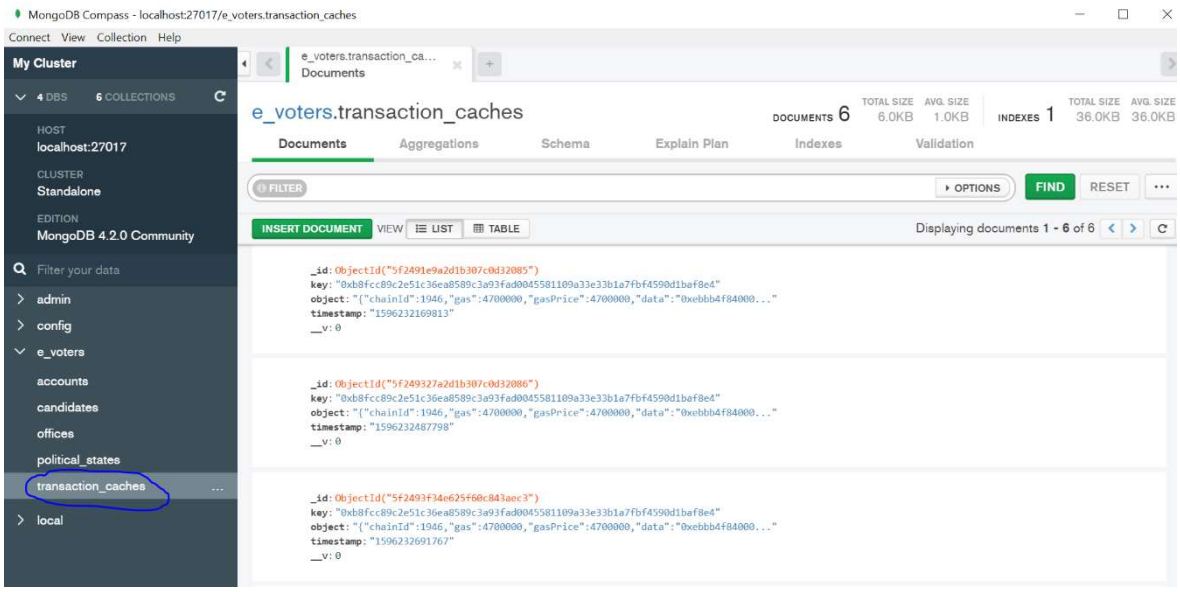

Figure 3: MongoDB transactions cache

To evaluate the latency of the proposed system, we set up two processes/approaches to recording the amount of time required to fetch voting data over a 60-minutes period. The two processes are the Decentralized or Blockchain State Caching and Centralized System approaches.

Decentralized/Blockchain State Caching

This process basically tests the performance of our proposed system from a latency standpoint. This is done by taking records of request and response time of transactions on the proposed system for a period of sixty minutes, calculate the latency and at the end, compare the average latency of the system with a centralized system that is based on HTTP requests. The process includes the following steps:

i. Within the PU module, a function was written to read data from the cached state in MongoDB. It is critical in ensuring that the blockchain states are cached and synced

once a stable internet connection is available.

ii. We recorded the time before the request was made (request time) and the time right after the response is received (response time).

iii. We then deducted the request time from the response time to get the wait period (latency) in milliseconds

iv. We then repeated the process 59 times over a 60-minutes period.

v.

Centralized system

This is the second method that we used to gauge the latency performance of our proposed system. This approach utilizes HTTP requests to a Server over 60 minutes. The method is implemented to replicate the work done by [16]. Though the authors did not take cognizance of latency as a performance matrix, we implemented a centralized system that was used to measure the latency of HTTP requests to the webserver. Afterward, we compared the average latency (in milliseconds), which is a very key latency matrix from this approach as well as that of the blockchain state cache. In this process the following steps were involved:

i. To simulate an HTTP request, we set up ngnix web server on port 80 on node1 (i.e. http://eth-node1.codecounty.com/) and then created an empty JSON file in web server's home directory as shown in figure 4. With an empty response body, the time it takes for the server to create a response is greatly reduced, leaving only the network request/response time.

Figure 4: node1 with empty test.json file

ii. Again, we recorded the time before the request is made (request time) and the time right after the response is received (response time).

iii. we then deducted the request time from the response time to get the wait period (latency) in milliseconds.

iv. we repeated the process 59 times over a 60 minute period

The result of these two approaches are presented and discussed in section 4.

4. RESULTS AND DISCUSSION

Base on the aim of this research work which is to advance a latency-improved blockchain-enabled electronic voting model, we present below performance outcomes from the experimentation conducted and represented in section 3.1.5 with respect to latency enhancement of our model.

4.1. Latency

Two processes were setup to evaluate the latency of the proposed system. These were done to record the amount of time required to fetch voting data over a 60-minute period. The two processes which were Decentralized/Blockchain State Caching approach and Centralized System approach produced the following results. As indicated earlier in section 3.1.5, the centralised system showcases the performance of a client-server deployment model which is represented by the work done by Ahmed [15] and Dogo et al [16]. In their work, the blockchain-based electronic voting had a centralised voters register. This means that system utilised a client-server deployment model in its request-response. Whereas, the proposed

system utilized a combination of distributed computing, cache database(MongoDB) and Web3 API to improve its latency. Table 4 below compares the latency of the system with that of the proposed system.

Table 4: Latency (in ms) for blockchain cached state and HTTP request to an API server

Timestamp (in ms)	Date Time	Blockchain Cached State (Latency in ms)	HTTP Request to an API Server(ceentralised system) (Latency in ms)
1578838920663	12/1/2020 14:22	87	831
1578838982561	12/1/2020 14:23	9	2784
1578839040531	12/1/2020 14:24	5	754
1578839100574	12/1/2020 14:25	5	798
1578839160540	12/1/2020 14:26	6	762
1578839221525	12/1/2020 14:27	5	1747
1578839280542	12/1/2020 14:28	7	764
1578839340507	12/1/2020 14:29	6	729
1578839400554	12/1/2020 14:30	6	776
1578839460527	12/1/2020 14:31	6	749
1578839520568	12/1/2020 14:32	6	790
1578839582402	12/1/2020 14:33	5	2624
1578839640552	12/1/2020 14:34	6	774
1578839700563	12/1/2020 14:35	6	784
1578839760508	12/1/2020 14:36	5	728
1578839822859	12/1/2020 14:37	5	3080
1578839880503	12/1/2020 14:38	5	724
1578839940565	12/1/2020 14:39	6	784
1578840000520	12/1/2020 14:40	5	739
1578840060531	12/1/2020 14:41	5	750
1578840120545	12/1/2020 14:42	6	764
1578840180525	12/1/2020 14:43	6	744
1578840240565	12/1/2020 14:44	5	783
1578840300515	12/1/2020 14:45	7	734
1578840360566	12/1/2020 14:46	6	783
1578840420527	12/1/2020 14:47	10	744
1578840480556	12/1/2020 14:48	5	773
1578840540516	12/1/2020 14:49	5	732

1578840600585	12/1/2020 14:50	8	801
1578840660527	12/1/2020 14:51	5	743
1578840720575	12/1/2020 14:52	5	791
1578840780693	12/1/2020 14:53	5	907
1578840840611	12/1/2020 14:54	5	826
1578840900545	12/1/2020 14:55	6	758
1578840962533	12/1/2020 14:56	6	2746
1578841020566	12/1/2020 14:57	5	779
1578841081550	12/1/2020 14:58	5	1762
1578841142552	12/1/2020 14:59	5	2763
1578841200743	12/1/2020 15:00	5	953
1578841260532	12/1/2020 15:01	5	742
1578841320576	12/1/2020 15:02	5	785
1578841380739	12/1/2020 15:03	5	947
1578841440510	12/1/2020 15:04	4	717
1578841500557	12/1/2020 15:05	5	763
1578841560598	12/1/2020 15:06	6	806
1578841620544	12/1/2020 15:07	5	752
1578841682510	12/1/2020 15:08	5	2717
1578841740510	12/1/2020 15:09	5	717
1578841800509	12/1/2020 15:10	3	715
1578841860519	12/1/2020 15:11	4	725
1578841920520	12/1/2020 15:12	5	726
1578841980541	12/1/2020 15:13	4	745
1578842040538	12/1/2020 15:14	5	743
1578842100532	12/1/2020 15:15	4	737
1578842160528	12/1/2020 15:16	6	731
1578842220585	12/1/2020 15:17	7	788
1578842280573	12/1/2020 15:18	5	776
1578842342457	12/1/2020 15:19	5	2660
1578842400807	12/1/2020 15:20	6	1010
1578842460811	12/1/2020 15:21	5	1013
Average Latency in Milliseconds		**6.833333333**	**1044.533333**
Approximately		**6.83**	**1044.53**

The result from the readings is represented in the graph in Figure 8 below.

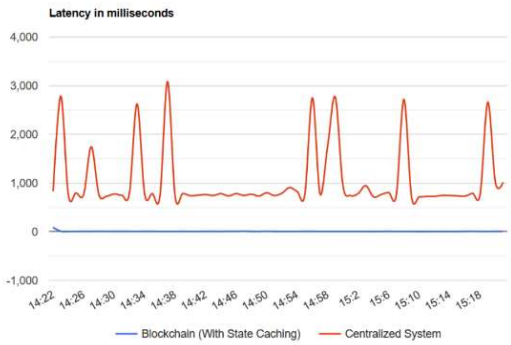

Figure 8. proposed system's latency evaluation

Table 4 shows the average latency of 6.83 milliseconds for the proposed system (i.e. blockchain with state caching) and 1,044.53 milliseconds for the centralized system. This means a 99.35 percent reduction in latency when compared to a centralized system.

5. CONCLUSION

A latency-improved blockchain implementation model for electronic voting system was designed and implemented. Its design adopts the combination of a cache database, web3 API and lightweight blockchain node called PUs to improve the latency of the blockchain-based model and adoptable for large-scale elections like Nigeria's general elections. The model has been demonstrated to be an improvement on the existing one.

Existing security implementations for electronic voting system and the blockchain latency improvement efforts were discussed in detail.

Our model implements the entire electoral processes (registration, transmission, tallying/counting, and result visualization) on the blockchain unlike on existing systems where certain critical process like the voters' registration is deployed off the blockchain where it is susceptible to single point of failure and increased latency.

REFERENCES

[1] Deloittecom. (2017). Deloittecom. Retrieved 20 November, 2017, from https://www2.deloitte.com/content/dam/Deloitte/ie/Documents/Technology/IE_C_BlockchainandCyberPOV_0417.pdf

[2] Dannen, C.(2017). Introducing Ethereum and Solidity: Foundations of Cryptocurrency and Blockchain Programming for Beginners.Brooklyn, New York, USA: Apress

[3] BOJANA, K., ELENA, K. and ANASTAS, M.(2017). Blockchain Implementation Quality Challenges: A Literature Review. Sixth Workshop on Software Quality Analysis, Monitoring, Improvement, and Applications, 1938, pp8-16

[4] Asif, A.A., Rayeesa, J., Abul, F.A.., & Nowroze, A.(2017). Design of a Solar Power Electronic Voting Machine. 2017 International Conference on Networking, Systems and Security (NSysS), 2017, pp. 127-131, doi: 10.1109/NSysS.2017.7885813.

[5] Muharman, L., Mira, K., & Sonny, Z.,(2016). Current State of Personal Data Protection in Electronic Voting: Demand on Legislature's Bill. 2016 International Conference on Informatics and Computing (ICIC), 80-86.

[6] Leyou, Z., Yupu, H., Xu'an, T., Yang, Y. (2010). Novel Identity-based Blind Signature for Electronic Voting System. 2010 Second International Workshop on

Education Technology and Computer Science, 122-125.

[7] Figueroa, K., Lopez, E., & Garcia, J. M.(2013). Electronic Voting System in Mexican Elections. 2013 Mexican International Conference on Computer Science, 1-6.

[8] Drew, S., Travis, F., Zakir, D., Jason K., Harri, H., Margaret M., Halderman.J.A. (2017). Security Analysis of the Estonian Internet Voting System. Proceeding of the 21st ACM Conference on Computer and Communications Security CCS '14, Scottsdale, AZ, November 2014. Retrieved 2 November, 2017, from https://jhalderm.com/pub/papers/ivoting-ccs14.pdf

[9] Scott,W., Eric W., Halderman, J.A.,Hari K. P., Arun K., Sai K. S., Vasavya, Y.,(2010). Security Analysis of India's Electronic Voting Machines. Proceeding of 17th ACM Conference on Computer and Communications Security CCS '10, Chicago, October 2010

[10] Cucurull, J. & Lee,.I. (2015). QR Steganography. A Threat to New Generation Electronic Voting Systems.,1-8

[11] Ansif,A.,& Mohsin,R.,(2016). Electronic voting with biometric verification offline and hybrid evms solution. The Sixth International Conference on Innovative Computing Technology(INTECH 2016), 332-337.

[12] Kausal, M., Srivatsan, S & Jayprakash, L.T. (2014). Architecting A Large-Scale Ubiquitous E-voting Solution for Conducting Government Elections. 2014 International Conference on Advances in Electronics, Computers and Communications (ICAECC), 44(12), 1-6.

[13] Cortier, V., Constantin, C.D.,Francois,D., Benedikt, S., Pierre,Y.S., & Bogdan, W.(2017). Machine-Checked Proofs of Privacy for Electronic Voting Protocols. 2017 IEEE Symposium on Security and Privacy, 993-1008.

[14] Nir, K., and Jeffrey, V.(2018). Blockchain-Enabled E-voting.IEEE Software 35(4),95-99. doi.org/10.1109/MS.2018.2801546

[15] Ahmed , B.A. (2017). A CONCEPTUAL SECURE BLOCKCHAIN-BASED ELECTRONIC VOTING SYSTEM. International Journal of Network Security & Its Applications (IJNSA) , 9(3), 1-9.

[16] Dogo, E., Nwulu, N., Olaniyi, O., Aigbavboa, C., and Nkonyana, T. (2018). Blockchain 3.0: Towards a Secure Ballotcoin Democracy through a Digitized Public Ledger in Developing Countries. 2nd International Conference on Information and Communication Technology and Its Applications (ICTA 2018), 477-484.

[17] Freya, S.H., Apostolos, G., Raja, N.A, & Konstantinos, M.(2017). E-Voting with Blockchain: An E-Voting Protocol with Decentralisation and Voter Privacy

[18] Rifa, H., and Budi, R.(2017)."Blockchain based e-voting recording system design". 2017 11th International Conference on Telecommunication Systems Services and Applications (TSSA). DOI: 10.1109/TSSA.2017.8272896

[19] Friðrik, H., Gunnlaugur, H., Mohammad, H., and Gísli, H. (2018). "Blockchain-Based E-Voting System". 2018 IEEE 11th International Conference on Cloud Computing (CLOUD). DOI: 10.1109/CLOUD.2018.00151

[20] Sudhir, K., (2018). What Are Smart Contracts in Relation To Ethereum? Retrieved from:https://coinsutra.com/smart-contracts/

[21] Darra, L.H.(2017). Legally Speaking: Smart Contracts, Archival Bonds, and Linked Data in the Blockchain. IEEE Computer Society, 1-4.

[22] Kim, H., laskowski, M., & Di matteo, T. (2017). A Perspective on Blockchain Smart Contracts:Reducing Uncertainty and Complexity in Value Exchange. IEEE , 1-6.

[23] Zhou , Q., Huang, H., Zheng, Z. and Bian, J., 2020. Solutions to Scalability of Blockchain: A Survey. IEEE Access, 8, pp.16440-16455.

[24] Kogias, E., Jovanovic, P., Gailly,N., Khoffi, I., Gasser, L., and Ford, B. (2016). Enhancing bitcoin security and performance with strong consistency via collective signing. in 25th USENIX Security Symposium USENIX Security 16. pp. 279–296.

[25] Li, X., Jiang, P., Chen, T., Luo, X., and Wen, Q. (2020). "A survey on the security of blockchain systems." Future Generation Computer Systems. Vol 107, pp. 841-853, doi: doi.org/10.1016/j.future.2017.08.020

ISSN: 1992-8645 www.jatit.org E-ISSN: 1817-3195

[26] Basescu, C., Kokoris-Kogias, L., & Ford, B. (2018). Poster: Low-latency blockchain consensus. Https://Core.Ac.Uk/Download/Pdf/14803 1098.Pdf. https://core.ac.uk/