*Abstract*— In this paper we provide a security analysis of the blockchain-based voting system proposed by Kumar Aakarshan [1]. For this, we take the role of administrators of an election and implement a decentralized voting application. We then explain the implementation details and expose a number of the possible attacks that make the system not optimal for running national elections. Lastly, we compare the safety risks of this technique to those of other online voting systems

# INTRODUCTION

Electronic voting systems require a particular amount of certainty that's not even expected in other high sensitive cybersecurity applications. this is often mainly thanks to the architectural and philosophical challenges round the concept of voting. In most secure systems, there are only a few users who actually operate it with credentials, during a centralized location (think major server for an organization , crime record database for the FBI, etc). Voting systems necessarily need a high level of distribution, such there are many voting machines operated by poll workers without the technical background to accurately assess the safety vulnerabilities within the system. Additionally , with voting comes the power to influence policy and social outcomes for millions, if not billions of individuals . The stakes during a major election during a country like Brazil, India, or the us are extremely high, and therefore the economic benefit to a successful hack might literally get on the order of trillions of dollars (for example, if one candidate is more favorable to your corporation than another and is looking to offer corporate tax incentive). this mix of high stakes and distributed control makes voting systems particularly difficult to secure electronically. the subsequent are considered essential requirements that any electoral system must satisfy:

1. Only eligible voters should be ready to vote, and that they can vote at the most once. There must be, in other words, some sort of authentication available for the system.
2. Each vote must be secret - i.e. it's impossible to work out how a voter voted so as to stop bribing, extortion, or blackmail on the a part of hostile actors.
3. The system must work under the idea that adversaries with tons of power exist and may attempt to attack the system.
4. The results of an election must be verifiable by auditing ballots or using another statistical techniques that don't depend upon the correctness of a software.
5. The system must haven't any a part of the operation that's considered "trusted" - i.e. no actor or piece of software are often in possession of a secret key, or special administrative power that can't be checked.

Paper-based voting systems are the foremost common method used world-wide for running national elections. Voters usually got to attend a voting center, present a government ID to be authenticated then proceed to cast their vote on a a paper ballot that's completely anonymous. the most advantage of this technique is that it truly provides secret ballots: once a voter submits their vote it's basically impossible to differentiate his or her vote from the remainder .

However, paper ballots don't necessarily satisfy all requirements listed above. especially , a corrupt party with access to the ballots could insert multiple paper-ballots with fake votes and while it's possible to seek out out that extra votes were included, the system doesn't allow to verify which ballots are the invalid ones. Or within the opposite side, an adversary could destroy paper ballots before they're counted and therefore the votes couldn't be retrieved.

Many people have tried to develop new voting systems that solve the disadvantages that paper ballots have; however, the most motivation for designing new systems seems to be solving the inefficiencies of the system just like the manual work required or the quantity of your time needed to understand the results, as against solving its security concerns. Consequently, the incorporation of latest technologies to automate and speed up the system is taken into account by many the right solution.

1. Internet Voting

Internet Voting refers to election systems during which ballots are electronic and votes are sew a machine (computers, cellphones, etc). These ballots are sent to and from the voting device via web or email. Multiple sorts of Internet voting are designed, and while it's a big amount of supporters, security experts believe that these systems shouldn't replace paper-based ones since the previous systems introduce multiple security risks.

Supporters of online voting believe some, if not all of the subsequent advantages of incorporating this technology within the election process:

• Election costs might be reduced: less people would wish to be hired and there would be no got to print ballots or buy envelops.

• Simplifies the counting process and provides accurate results: an easy piece of code can quickly compute the entire number of votes per candidate as against expecting people to manually count votes.

• Makes the system more accessible: Citizens could easily vote while abroad since the web is ubiquitous.

However, security experts believe that implementing internet voting for important elections is extremely irresponsible since the system would be exposed to many threats that come inherently with the utilization of the web . the main concerns are

• The entity responsible of the system may need enough power to corrupt the system: whoever has control over the software running the system could potentially change a vote without leaving any evidence. this is able to violate the wants 1), 3), 4) and perhaps 5) of voting systems.

• The system might be attacked: Online systems are susceptible to cyber attacks which will either affect the functionality of the system, leak tip or modify the behavior of the system in unexpected ways in which are often difficult to spot . this is able to violate the wants 3) and 4) of voting systems.

• These systems might ease traceability of votes to voters: The electoral system itself could contain information that reveals information about the voter like number , passwords or maybe the vote itself. this is able to violate the wants 2) and 3) of voting systems.

• Machines used for voting are often hacked: this is often perhaps the most important concern since there's not much administrators of an election could do to stop devices from being victims of hackers. If a machine used for voting is infected with malware the votes are often tampered or trace independently of how secure the electoral system is. this is able to violate the wants 2) and 3) of voting systems.

• Machines could fail to authenticate identities: If for instance authentication relies on passwords or other secret information, adversaries could steal that information from voters and use it to vote with their identity. this is able to violate the wants 1), 2) and 3) of voting systems.

The first governmental election that used Internet voting within the us was the presidential election of the Reform Party in 1996 [7]. Since then, internet voting began to become a more popular idea but with its popularity more security risks also emerged. In 2000, the Democratic Party hired a personal company to run its presidential primary in Arizona with an online electoral system [6]. As a result, multiple security threats including denial of service attacks, voter identity thefts and hacking of voting machines were ubiquitous. the risks of such an election were strongly exposed and a couple of years later the standards of voting systems were strengthened by the Federal committee of the U.S.

Currently, several states allow online voting through web portals or email. However, the bulk of the citizens still don't approve this voting methods since votes cast during this form are exposed to security attacks from the administrators of an election, other citizens or maybe foreign adversaries.

1. BLOCKCHAIN-BASED electoral system

[BB]

The blockchain techonolgy keeps an inventory of records that are very difficult to switch . Specifically, the blockchain arrangement may be a public distributed ledger formed by appending blocks that are linked as a chain: each block contains a hash value that's a function of the previous block. This design assures immutability of the ledger, since modifying the info in one among the blocks would immediately create inconsistencies within the hashes of the descendent blocks.

In a blockchain implementation, writes to the ledger are only allowed if the nodes of the network reach concensus and approve write. Read accesses to the ledger are denoted same as calls and write accesses like transactions. Transactions are then monitored and therefore the consensus protocol wont to approve or reject transactions varies with each implementation. These characteristics make the blockchain technology a distributed and decentralized system that doesn't grant complete power to any party individually but instead gives equal partial power to multiple parties.

The fact that data deployed onto the blockchain can never be modified which there's no central entity on top of things of the ledger has naturally make many of us believe that this technology might be the answer to the safety problems present in online voting systems. And while it is sensible that appending votes onto the blockchain can protect votes from being modified, there's tons of labor that must be done to fill the gaps: how would authentication be successfully achieved? How would cyberattacks be prevented? Who should create and have control over the blockchain network? How can we guarantee that votes are secret?

In this section we'll describe and analyze the election design that was proposed by HjÃa˛lmarsson and Hreiðarsson to implement a blockchain technology as a part of a electoral system [1]. we've taken the role of administrators of an election and that we have implemented a voting web application following their design so as to raised understand their proposal also on explore possible security attacks of the system. This design consists of smart contracts that are deployed onto a personal blockchain and permit to represent votes as irreversible transactions.

1. Blockchain Network

Blockchains have several sorts of access control. For this implementation we'd like a personal blockchain network during which the nodes of the network represent a partition of the voting population. While the planning by HjÃa˛lmarsson and Hreiðarsson specifically focuses on liquid democracies and considers one node per voting district, we here generalize this approach by simply considering any partition of the voters, which might be more granular or more general than a district-based partition. additionally , during this design the nodes are assumed to be trusted parties from each partition set with high computing power.

The main advantage of personal blockchains is that they restrict both read and write accesses to specific participants. As a consequence, private blockchains are composed of a limited number of fixed nodes, which provides other benefits like making transactions cheaper and faster as they have to be validated only by a couple of number of nodes. We then consider a network with a group of k nodes

*N* = {*n*1*,...,nk*}

that are in one-to-one correspondence with the weather of a partition

P = {P1,...,Pk}

of the set of eligible voters for the election. the precise value of k for a selected election also because the exact partition sets depend upon the extent of distribution desired for the system also as other considerations just like the physical location of voters.

Bootnodes: Besides the nodes in N, which keep a uniform copy of the ledger and may read the blocks or write of them, our network must also include other sort of nodes called bootnodes, which don't keep the state of the blockchain. The unique purpose of those nodes is to permit the nodes in N (the partition nodes) to get one another and keep an equivalent copy of the ledger.

Concensus Protocol: The Proof of Authority [PoA] is that the consensus protocol implemented for this blockchain network. Unlike the more common Proof of labor protocol used for the bitcoin blockchain, during which the nodes must solve complicated puzzles so as to append transactions to the blockchain and obtain bitcoin rewards, the PoA protocol relies on validator nodes that get payed to verify the validity of a transaction. Specifically, when a transaction is requested, each node that's declared as a validator within the blockchain network proceeds to either approve it or reject it. The transaction is then appended onto the ledger if and as long as the bulk of the validators approve it.

In this implementation, all nodes in N are declared as validators and thus a vote transaction is successfully appended onto the blockchain if an as long as the bulk of the partition nodes approve it. Standard transactions include information about the sender, the receiver and therefore the timestamp of the transaction; however, transactions during this design must only include the transaction ID, the block number, the district from which the vote was cast and therefore the value of the vote. We also clarify that while it's possible that this partition nodes become corrupted and validate fraudulent votes, the reputation of those validators is at stake and a minimum of half them would wish to become corrupt to approve a nasty transaction.

1. Wallets

Users of a blockchain network got to own a minimum of one account within the blockchain. An account consists of a public key that's wont to identify transactions to and from the user, also as a personal key that the users use to prove that an account is indeed theirs. Because public keys tend to be very large, blockchain uses shorter strings called addresses that are representative sorts of the general public key for an account. A wallet may be a common program used for managing users’ accounts and their respective keys.

In order to permit voters within the election to cast their votes as blockchain transactions, the election administrators must create the blockchain network described above also together wallet with one account per eligible voter. If the system were ready to match accounts to their corresponding owners, this election system would violate requirements 2) and 3) of voting systems. However, the blockchain network must somehow verify that a voter is indeed the owner of the account that he or she is using to cast the vote. to bypass this problem, the utilization of Zero Knowledge Interactive Proofs are often wont to generate and authenticate voter’s accounts [1].

Zero Knowledge Proofs [ZKP]: This cryptographical tool allows one party to convince another party that they know a selected piece of knowledge without revealing any information aside from their knowledge of that value. as an example , consider a signature scheme during which the parties involved have a secret key x and a public key gx. Suppose Alice wants to convince Bob that gx is her public key. A round of a Zero Knowledge Proof would go as follows:

• Alice chooses a random number r and sends m = gr mod p to Bob.

• Bob asks Alice either for the worth of x + r mod (p−1) or the worth of gx+r mod (p−1).

• Alice sends Bob a worth y

• Bob verifies that y satisfies either gx · m ≡ gy mod(p−1) or m ≡ gy mod p respectively.

If Alice doesn't know the worth of x she features a

probability of answering correctly Bob’s request. this suggests that after multiple rounds Bob can confirm that Alice knows x if she answers consistently whenever .

Non-Interactive Zero Knowledge Proofs [NIZKP]: This method may be a variant of ZKPs during which interaction between the 2 parties isn't necessary. Instead, a reference string that's common and shared between them is enough for reaching Zero Knowledge proof with none rounds of repetitive interactions.

Voter Authentication: In their implementation, HjÃa˛lmarsson and Hreiðarsson assume that every voter will got to register for the election by physically presenting a government ID to election administrators. During registration, eligible voters would be assigned an electronic ID and would be prompted to settle on a 6-digit PIN for the corresponding ID employing a secure service provider for biometric identification . This design assumes that NIZKPs are wont to generate the wallets of eligible voters and to prove that a selected wallet belongs to a selected voter without revealing the identity of the voter.

Therefore, when a users want to vote they will simply use their electronic ID and PIN to authenticate themselves and obtain access to their wallet. The wallet must also contain the partition node which will be employed by the user to interact with the ballot smart contract like the set where the voter belongs.

1. Smart Contracts

Smart contracts are pieces of code that self-execute during a decentralized application. The functions contained within the smart contracts must specify the agreements of the contract, which may be deployed onto a blockchain to form those agreements trackable and irreversible. After deployment the code can't be changed and therefore the parties involved are sure to follow the principles as written. an enormous advantage of those contracts is that they're selfverifiable as their code specify requirements that trigger events when those aren't satisfied. additionally , calling a public function within the smart contract corresponds to creating a transaction within the blockchain. In our case, this suggests that a function can only be executed if it satisfies the contract’s rules and therefore the majority of the validator nodes within the blockchain network successfully approve that the account making the transaction has permission to try to to so.

In this implementation, smart contracts are used as ballots to enforce the election agreement. Specifically, we would like the contract to contain a voting function which will only be called once per valid voter which is executed as a transaction within the blockchain. Moreover, so as to facilitate the vote count process, we would like our contract to supply functionality for checking the entire number of votes that a selected candidate has. However, since voters shouldn't have access to partial vote counts before casting their votes, only the administrators of the election should be ready to call these functions.

For the system to be distributed and decentralized, also on optimize for performance, we use a ballot contract per partition set Pi and that we restrict permissions for the ith ballot contract in order that only the corresponding node ni can interact with it. Thus, voters that belong to Pi can execute the vote function within the ith ballot contract by connecting to the network through node ni. To enforce that voters choose the middle like the partition set they belong to, the voter’s wallet contain information on the node they need to interact with to form their vote transaction.

In order to deploy all k ballot contracts directly , we use a factory smart contract that makes as many instances of Ballot contracts as necessary and deploys them onto the blockchain. We modified the factory contract proposed by HjÃa˛lmarsson and Hreiðarsson so as to form the appliance easier to implement. Specifically, we added functionality in order that functions of a specific ballot contract are often called from the factory contract and not necessarily directly from the Ballot. This simplifies our implementations since nodes only directly interact with one contract. These modifications don't affect the aim of the distributed design because nodes can only call these intermediate functions if they need permission to interact with the corresponding ballot contract.

1. Implementation Software and Parameters

To run our election as a sensible contract application we utilize one among the implementations of the Etherum protocol called Geth. Taking the role of administrators of an election, we used this interface to make our private blockchain with three nodes that implemented the Proof of Authority consensus protocol with a transaction rate of 5 seconds. We used just one bootnode for discovery of nodes since our resources were limited and it had been enough for the safety analysis of this technique design.

Moreover, we wrote our smart contracts with the programing language Solidity and used the Truffle development environment to compile and deploy them onto the blockchain. The Truffle framework also allowed us to interact with the contracts after deployment via web3 in client-side JavaScript.

For this application we created alittle election with only 3 candidates, 10 voters and three partition sets. we've overlooked the authentication part in our implementation, but we'll assume the NIZKP distribution of wallets as indicated earlier for the safety analysis. Similarly, our implementation doesn't modify the default data included during a transaction but we'll analyze the system assuming that timestamps and senders’ addresses aren't recorded.

1. Security Analysis

We will now proceed to research both the strengths and weaknesses of the BB system described above in terms of security. While we've used our own implementation to check a number of the arguments that we'll discuss, we've not implement all the attacks discussed during this section.

Security Analysis

• The administrators of the election have complete power over the system and will corrupt it: as administrators of the implemented election we are ready to manipulate the blockchain network in multiple ways. as an example , we will create more wallets that don't correspond to eligible voter and use them to fake votes or to permit non eligible voters to vote. this is often particularly dangerous because as a right nobody should be ready to match the address that committed a transaction with a specific voter, and thus verifying that wallets belong to an eligible voter and to not a ghost voter becomes difficult.

• This distributed and decentralized system is more immune to attacks: Most attacks would wish to successfully attack multiple nodes within the network so as to affect the system’s functionality. A DDoS attack, for instance, would wish to form unavailable all bootnodes so as to affect the interactions between validator nodes. While this is often still a possible attack, the proposed system is more resistant than other decentralized applications. additionally , other approaches sort of a Sybil attack are executable during this system because the utilization of a personal restricts access to make new nodes. additionally , corruption of one node within the network doesn't leave corruption of the ledger since honesty from the bulk of the nodes avoid corruption of the blockchain.

• Traceability of votes to voters isn't allowed: the utilization of Non Interactive Zero Knowledge Proofs prevents the system from matching wallets to the voters’ identities while still verifying that the voter casting the vote is that the owner of the wallet from which the transaction is formed . additionally , the very fact that transactions don't include the address from which they were sent nor the time at which they were made prevents adversaries from using time data to work out who made a selected transaction, not even when the adversary knows the address of the voters’ wallet.

• The devices from where voters interact with the blockchain are often hacked: by inserting an easy script that records the moves, scrolls and clicks of the mouse during a device used for voting, we were ready to find the candidate that a selected voter chose to vote. Other possible attacks include modification of the interface so on flip the order of the candidates and make voters believe they're voting for a special candidate, or maybe modifying the parameters when calling the vote function to pick a special candidate.

• Failure to authenticate identities: during this system all what a voter must be authenticated by the system and gain access to the corresponding wallet is that the electronic ID and a 6-digit PIN, and thus adversaries could threat voters to steal their credentials and vote with their identities. Including other sorts of identity authentication like body metrics could help alleviate this risk, although it might still be possible to hack the authentication devices to form them fail.

The code for our implementation are often found at https://github.mit.edu/kimvc/6. 857-Voting-System.git and a video of the appliance also as an attack are often found at https://www.youtube.com/watch?time\_ continue=6&v=RkYbPhdXGFI.

1. OTHER ONLINE VOTING SYSTEMS

Currently, there are several online voting systems that are getting used within the us both for little and enormous elections. Here we'll analyze the 2 of these that appear to be the foremost popular: Helios and Voatz. we'll describe their designs and compare both them to the BB System in terms of security guarantees and risks.

1. Helios

Helios is a web electoral system that claims end-to-end verifiability. This reflects within the system that's wont to audit ballots and to trace that a ballot has been submitted or tallied. Note that the manufacturers of Helios themselves don't believe that their software should be used for major federal elections, because they believe voter’s computers aren't secure enough to be utilized in elections that are exposed to powerful attacks.

Helios security guarantees are supported El Gamal reencryption, which is employed to enforce anonymity of the votes. Voting during this system works within the following way:

(a) An administrator creates the election and inputsthe names of the candidates and therefore the exact times at which the election must begin and end. The administrator next creates an inventory of emails to which credentials must be sent. These emails contain the username and password for every one that is an eligible voter and therefore the administrator never has access to such passwords.

(b) A voter uses her credentials and casts her voteusing the web interface. At the top , the voter has the choice to audit her vote, during which case the signature is checked to demonstrate that the ballot is really being cast correctly.

(c) At the top of the election, the tally takes placeand the tracker related to each ballot informs voters that their vote has been tallied. Since the tracker may be a finger print of the encryption of the vote itself, it are often wont to specifically reference the voter.

Security Analysis:

• The administrator has no power to corrupt the system since Helios provides unconditional integrity. this is often an enormous improvement over the BB system, during which despite of the very fact that the system was distributed and descentralized, the administrators had power to make extra wallets, and corruption of multiple nodes could corrupt the whole system.

• A disadvantage of this technique is that Helios is susceptible to attacks like browser corruption. While the BB system didn't specify interface details, this might be an attack that affects that system also . generally , Helios’ major problem is that the server itself has complete control over the election and thus voters’ privacy and election honesty relies on Helios.

• Similar to the BB system, Helios guarantees that voters’ ballots are completely secret. While we are ready to see who voted, we aren't ready to see what each individual person voted for, nor indeed is it a tall possible to seek out out. The tally happens only at the top , and is conducted by combining the encryptions of the first votes, and only then decrypting.

• Like every other internet electoral system , Helios’ security cannot protect the election against attacks to the voters’ machines.

• Like within the BB system, the authentication method during this system allows for violations of the wants of a electoral system . Using emails for authentication means an adversary could vote with someone else’s identity by hacking the voters’ email or threatening them to point out them their usernames and passwords.

1. Voatz

Voatz may be a mobile elections platform that runs on only latest smartphones and makes use of a descentralized blockchain network. It relies on the safety of smartphone technology and immutability of a blockchain. this suggests that the voters got to have the newest smartphone and therefore the Voatz app so as to be ready to vote. Voters who don't own one among these devices will then not be ready to vote through

Voetz. Voting during this system works within the following way:

(a) Election officials grant Voatz access to a databasecontaining information of all the voters.

(b) Before voters cast a vote, they need to authenticatetheir identity by scanning a legitimate ID and sending a live snapshot also as their their fingerprint. The Voetz app matches the voter’s fingerprint to the device then verifies the validity of the ID and therefore the snapshot.

(c) Each vote is then represented as a transaction thatmust be verified by a selected set of servers.

Security Analysis

• Vaotz prints paper ballots for every vote. These ballots are then audited to verify that every voter voted exactly once which all the votes were counted. This limits the administrator’s ability to switch the votes or add fake ones.

• Similar to Helious, the server itself has complete control over the election and will corrupt it; Voatz gains access to databases of voters and therefore the developers themselves could possibly be collecting all the voting data they claim that's secure and anonimous. albeit Voatz has been largely used across the U.S. many of us are still reluctant to simply accept Vaotz as a politician system for state elections, since it's hard to trust a personal company when there's tons at stake.

• In terms of traceability, Vaotz ensures it on a special way than the BB system. The identity of every voter is protected twice: first by the app within the smartphone, then within the blockchain. It also uses end-to-end encryption. Note that this technique generates accounts for the voters and features a database for them, implying that the system knows which account corresponds to which voters. However, for every vote the name of the voter is anonymous and encrypted. Thus, there's no way of telling the voter’s account from tracing a vote.

• In terms of security of the device used for voting, unlike the BB system, Voatz takes measures to make sure that any device used for voting isn't hacked. so as to realize that, it only allows last generation smartphones to be used. That way it takes advantages of their various security capabilities. for instance , it detects malware or changes on the OS . This greatly lowers the danger of the device being attacked. However, this negatively impacts the acessibility of the system, as many citizens don't own one among these devices.

1. CONCLUSION

The blockchain system analyzed during this paper doesn't satisfy the essential security requirements because despite the cutting-edge technology utilized in the implementation, there are always security concerns inherent to Internet Voting. We believe the foremost important limitation for implementing secure internet electoral system s is that the devices used for voting might be hacked independently of how secure the voting system is.

REFERENCES

1. Friðrik Þ. HjÃa˛lmarsson, Gunnlaugur K. Hreiðarsson. *Blockchain-Based E-Voting System*. School of Computer Science, Reykjavik University, Iceland, (2018).
2. Voatz, https://voatz.com/faq.html
3. Electronic Voting, http://lorrie.cranor.org/pubs/evotingencyclopedia.html.
4. Ben Adida. *Helios: Web-based Open-Audit Voting*, Harvard University, (2008).
5. Ronald L. Rivest. *Internet Voting—Seriously?*, EVN Conference, (2016).
6. Dennis Berman. *We the E-People*, BusinessWeek, (2000).
7. *Arizona Democratic Party Selects Votation.com to Hold World’s First Legally-Binding Public Election Over the Internet*, The Free Library, (2014).