ISSN No:-2456-2165

Fortifying E-Voting Privacy: A Triad of Blockchain, Convolutional Neural Networks, and Quantum Key Distribution for Unparalleled Security

[1]Bhaumik Tyagi
Jr. Research Scientist
Computer Science Engineering
Delhi, India

[2]Pratham Taneja
ADGITM
Electronics & Communication Engineering
Delhi, India

[3]Avi Thakur
IIIT K
Computer Science Engineering
Jharkhand, India

Abstract:- The advancement of blockchain has facilitated scholars to remodel e-voting systems for future generations. Server-side attacks like SQL injection attacks and DOS attacks are the most common attacks nowadays where malicious codes are injected into the system through user input fields by illicit users which leads to data leakage in the worst scenarios. Besides, Quantum attacks are also there which manipulate the transactional data between blocks in the blockchain. Stale Block and Blockchain forks are also one of the main issues in existing blockchain-led e-voting systems. In order to deal with all the above-mentioned attacks, integration of Blockchain, CNN & Quantum Key Distribution is done in this very research. In, e-voting systems the employment of blockchain technology is not a novel concept. But privacy and security issues are still there in public and private blockchains. To solve this, Blockchain, CNN & QKD-based proposed model is introduced in this research. This research proposed cryptographic signatures and blockchain smart contract algorithms to validate the origin and integrity of the votes. Also, a comparison between previous smart contract algorithms and proposed smart contract algorithms is done in terms of time complexity.

Keywords:- *Hybrid Blockchain, Secure e-Voting System, Convolutional Neural Networks, Quantum Key Distribution.*

I. INTRODUCTION

E-Voting or Electronic voting refers to voting via the internet using any digital device. The vital challenge for e-voting is the major security risks it might cause as an electronic voting system essentially relies on the internet. Several countries are willing to begin electronic voting for a variety of purposes such as accessibility and decreased costs. Amid the increasing use of the internet, SQL injection attacks, DOS attacks, and Quantum Attacks are the common techniques to outbreak e-voting systems. In, SQL injection attacks, attackers inject some SQL codes into the original code in order to obtain or abolish delicate information. Timing attacks, Blind injection, union queries, illegal queries, and tautologies are some categories of SQL injection attacks. Blind SQL injection attacks are used by hackers in which the hacker requests true or false questions from the user. Using Tautology statements, attackers inject codes into the authentication phase, which says 1=1 is always true so that the injected query becomes valid even if the invalid password and username are entered. In order to join the original query with an injected query, the union operator is used while injecting codes. In recent times, the major apprehension about the e-voting process is its security fallouts. To counter these security issues and attacks, numerous studies have been done but not collectively and completely. Encrypting and decrypting plain-text messages while sending electronic data is the most popular application of cryptography. This technique encrypts data using a symmetric ("secret key") approach. The recipient is then provided with both the secret key and the encrypted message to be decrypted. In the event that a third party obtains the message, they have all the necessary tools to read and decode it.

To solve this issue, "public key systems "(asymmetric) are formulated by cryptologists as a solution. In an asymmetric system, each user has two keys: one private & one public. The communication is encrypted and sent after the sender requests the public key. At the time of message arrival, only the recipient's private key will decode it. To achieve a precise conclusion, implementation, and optimization are required in existing e-voting research. The proposed solutions do not guarantee the security or privacy of voters from various aforementioned attacks. The integration of blockchain technology with convolutional neural networks (CNNs) & quantum key distribution (QKD) could provide secured e-voting solutions. The convolutional neural network is a regularized version of the multilayer perceptron in the system, it is employed to analyse visual imagery upon registration.

ISSN No:-2456-2165

The goal of this research is to modernize cutting-edge blockchain-based electronic voting systems. Quantum attacks can be implemented on blockchain-based systems using algorithms like Grover's technique. Despite of being a secure technology, security of blockchain systems can be compromised by imposing quantum-led attacks. Using such algorithms, the blocks can be easily substituted. Also, the ample record of chains can be refabricated once again by speeding nonce's creation and quickly alters the hashes, jeopardizing the Blockchain's integrity. This research provides solution for privacy and security from attacks like Quantum attacks, SQL Injection attacks, DoS attacks, computational over-head, Stale block, and Blockchain forks.

- *The Most Significant Standards in Order to Create a Secure Voting System are as follows:*

- Unauthorized voters should not participate.
- Authorized voters should not be allowed for casting two votes.
- Voting is confidential.
- Voter's choice should not be public.
- Verifiability: Voters must be able to verify that their votes are correctly calculated [1].
- Privacy of voters should be at first priority while creating any voting system. Identity of voters should not be disclosed for whom they voted for.
- Once the Vote is cast and verified then it must not be directed or controlled in any means.

Traditional voting systems are not preferred over e-voting systems in today's era. Reason behind this is execution time in traditional voting systems is slow as compared to e-voting systems. E-voting systems are immutable whereas traditional voting system are mutable. Manual workflow is required in traditional voting system which in turn makes it expensive. E-voting systems are cheaper option because of automated workflow. Traditional voting systems are prone to error whereas e-voting systems are error free. Traditional voting system's major drawback is they are less secure and purely centralized whereas e-voting systems are secure and decentralized.

Blockchain (BL) is an open distributed ledger. Open; Transactional information is public and available to all and a local copy of every transaction is available to every block. Blockchain is decentralized and fault-tolerant technology. No Central authority or third party is there to control the entire BL process. BL is efficient in terms of security, speed, and scalability. Verification of valid information is done before moving to further transactions. Every information that is stored once in the blockchain after validation is immutable or permanent. Hackers cannot hack the whole blockchain as in order to hack a single block, they should be able to hack another block as well.

Convolutional layers are found in the hidden layers of a convolutional neural network. Typically, this entails adding a layer that performs a dot product of the layer's input matrix and the convolution kernel. As the convolution kernel moves across the input matrix for the layer, adding to the input of the following layer, a feature map is created. Following this are additional layers such as normalization, pooling, and fully connected layers.

Nowadays, QKD (Quantum Key Distribution) is the emergent technology in the field of quantum cryptography. Quantum Kye Distribution is a well-known and sophisticated method to come out of quantum cryptography. Using the QKD protocol, a random bit stream can be created between two parties. Then, using an OTP (One-time pad) technique, a secret message is encrypted using this random message. QKD allows us to generate a secure key so that secret information can be sent securely from one to another. The main aim of using QKD is security as it guarantees security. This protocol is used to generate random bitstreams for two points of communication.

II. LITERATURE REVIEW

Integration of Blockchain, CNN, and Quantum Key Distribution to build an e-voting system is a wholly novel concept. Although, there are existing studies that discourses about using blockchain for e-voting or using Blockchain and CNN for e-voting but augmenting QKD, using hybrid blockchain, and using CNN in the registration phase are entirely novel. The main aim of existing studies is to perform e-voting in a step-by-step process but the security of the system and the privacy of the users were not a major concern. In this research, hybrid blockchain is used as a solution for privacy and security due to the failure of public and private blockchains in terms of security and privacy from Quantum Attacks, SQL Injection Attacks, DOS Attacks etc. Existing Facial recognition-led e-voting systems are not at all secure from various attacks and facial recognition using CNN and private blockchain are proposed in some studies but cannot counter attacks and hence voter privacy and system security are still vulnerable. Traditional voting systems are prone to error whereas e-voting systems are error free. Traditional voting system's major drawback is they are less secure and purely centralized whereas e-voting systems are secure and decentralized. This research provides solution for privacy and security from attacks like Quantum attacks, SQL Injection attacks, DoS attacks, computational over-head, Stale block, and Blockchain forks.

ISSN No:-2456-2165

Table 1 Functionality-based Comparison between Existing Voting Approaches

Approach	Smart contract language	Implementation platform	Smart contract-based implementation	Secure voting	CNN Augmentation	Decentralization	Voters Privacy
C.D González et al. (2022) [1]	Golang	Hyperledger Fabric	Yes	NM	NM	Yes	No
Mustafa et al. (2021) [2]	NM	Hyperledger Fabric	Yes	NM	NM	Yes	No
K. Isirova et al. (2020) [4]	NM	NM	No	Yes	NM	Yes	No
Kirillov et al. (2019) [3]	NM	Hyperledger Fabric	Yes	Yes	NM	Partially	No
Chaieb et al. (2018) [6]	Solidity	Ethereum	Yes	Yes	NM	Yes	No
Hardwick et al. (2018) [5]	Solidity	Ethereum	Yes	NM	NM	Partially	No
Messer (2017) [7]	Solidity	Ethereum	Yes	NM	NM	Private	No

Here, NM refers to Not Mentioned, Y refers to Yes, N refers to No, H Fab refers to Hyperledger Fabric, Eth refers to Ethereum, SOL refers to Solidity.

Table1 represents the Existing Approaches of Gonzalez et.al (2022) [1], and Mustafa et.al (2021) [2] use the implementation platform Hyperledger fabric which is purely private and leads to some sought of centralization which directly hampers voters' privacy and secure voting. There is no CNN Augmentation and No QKD implementation. Whereas the approach of Isirova et.al (2020) provides secure voting and decentralization but there is no voter privacy.[4] Approaches proposed by Messer et.al (2017) [7], Hardwick et.al (2018) [5], and Chaise et.al (2018) [6] used Solidity as a smart contract language and Ethereum as an Implementation platform but all the above-mentioned approaches are partially decentralized not purely which directly hampers Voter privacy and secure voting.

Table 2 Comparison of the Proposed E-Voting System based on a Blockchain and Previous Related Work

Properties	[16,17]	[13]	[21]	[20]	[14]	[18]	[19]	[15]	Our Proposed Model
Security	No	No	Yes	Yes	Yes	No	Yes	No	Yes
Consensus	PoW	PoW	PoS	PoS	PoS	PoS	PoW	PoW	Hybrid
Scalability	Yes	No	Yes	Yes	No	No	No	Yes	Yes
Transparency	No	Yes	Yes	No	Yes	Yes	Yes	No	Yes
Verifiability	Yes	No	No	No	Yes	Yes	Yes	No	Yes
Privacy	No	Yes	Yes	No	Yes	No	No	No	Yes

Here, PoW, PoS refers to Proof of Work & Proof of Stake respectively.

Table 2 contrasts our research with earlier studies. Former efforts to design procedures for blockchain-based e-voting systems developed schemes for cryptocurrencies. Cruz and Kaji [16] suggested a procedure for blockchain-based e-voting and explored some of the aspects of security for electronic voting in response to recent developments [16][17]. Votes are tallied by adding up each candidate's tokens in the Bitcoin blockchain under the End-to-End (E2E) voting system that Bistarelli and others [17] proposed. Using Bitcoin, Zhao and Chan [13] created a framework that is comparable to those previously discussed [16][17]. However, due to the time- and resource-intensive nature of the Bitcoin consensus mechanism, all three of the aforementioned protocols have limitations in terms of the scalability of the blockchain-based e-voting system [17]. A voting mechanism for electronic ballots based on the Ethereum blockchain was introduced by other researchers [14][18][20][21]. All of these studies centered on exploiting the security characteristics provided by the Ethereum blockchain contract to increase the security of the electronic voting system. These studies, however, made no mention of the blockchain's scalability or performance. The efficiency and scalability issues for blockchain-based electronic voting systems were thoroughly discussed [19,15].

Khan [19] specifically experimented with permissionless and permissioned blockchain settings grounded on a variety of circumstances, such as the quantity of production rate, block, block size, transaction speed, voters, and. Zhang and others [15] asserted have demonstrated the shortcomings of blockchain-based electronic voting schemes. They do not, though, explicitly mention important security characteristics like uniqueness and ballot reception in their document.

Table 3 Contrast of Several Present e-Voting Approaches using Blockchain

Article	Key Design/ Choice of Algorithm	Features of the Existing System	Limitations
Liu, Yi et al. (IACR, 2017) [10]	Blind Signature	A blind signature process is applied to permit the inspector and organizer to mark the vote hash exclusive of disclosing authentic vote. The voting block consists of the 'vote message', 'sender's public key', the 'receiver's public key'.	However, this authentication process augments supplementary safety to the system, it commences greater delay and latency while scaling huge e-voting set-ups.
Ben Ayed. (IJNSA, May 2017) [12]	Candidate-specific blockchains	Proposed a basic e-voting system focusing more on candidate.	Due to diverse blockchains for every candidate, processing overhead & greater storage occurs. The usage of a single blockchain can advance implementation & execution.
Ganji et al. (Dell EMC, 2018) [8]	Framework based on Multichain	In order to restrict respective voter to a sole transaction, this system uses multi-chain network. Trusted Third Party verify the validity of the voter using a secret message provided to the TTP by the voter.	Verification of each voter with a secret message that generates a ref. number leads to greater delay to some extent.
Yu et al. (ISC, 2018) [9]	Hyperledger Fabric with Practical Byzantine Fault Tolerance	Utilizes Hyperledger Fabric as the blockchain framework, consensus using practical byzantine fault tolerance.	The anticipated scheme can be further enhanced by employing 'Hyperledger Sawtooth', in order to carry out the corresponding implementation of transactions.
Yi. (EURASIP, 2019) [11]	Elliptical Curve Cryptography [E.C.C]	Elliptical C.C. is used in which voter generates the signature of their vote block using a private key, with the signature verified using the voter's public key.	Even though the system recommends an intricate process for vote blocks authentication, the PKI database used is still vulnerable, which if unprotected, can nullify the whole procedure.

Algorithms used for creating smart contracts in existing approaches are not optimized in terms of time complexity as time complexity of existing approaches lies somewhere between worst O (N^2) to average O (N log N). Table 3 compares existing e-voting approaches. Liu, Yi et al. (IACR, 2017) used Blind Signature approach as key design and limitation of this system is that large scale e-voting systems cannot be established using this approach as it commences delay and high latency. This happens due to presence of additional security system [10]. Ben Ayed. et al. (IJNSA, May 2017) [12] used Candidate-specific blockchains approach as key design and limitation of this system is processing overhead and greater storage due to diverse blockchains for every candidate. The usage of single blockchain for each candidate can gradually enhance performance [12]. 'Ganji et al. (Dell EMC, 2018)' used multi-chain framework-based system and limitation of this system is that every secret message provided by the voter has to be verified first which leads to greater delay in each transaction [8]. 'Yu et al. (ISC, 2018)' used Practical Byzantine Fault Tolerance with Hyperledger Fabric and this scheme can be more improvised by employing 'Hyperledger Sawtooth' that provisions transactions execution parallelly [9]. Yi. (EURASIP, 2019) [11] used Elliptical Curve Cryptography [E.C.C] approach and limitation is that still scheme suggests an intricate technique of vote blocks confirmation, also the PKI database used is still defenceless, which if exposed, can nullify the whole procedure.

III. METHODOLOGY

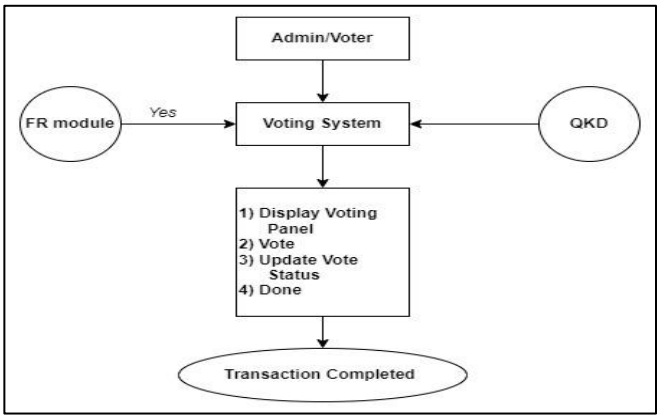

Fig 1 Proposed Model

The proposed work secures the system by reducing the chances of attacks like DOS, SQL Injection, Stale block, Quantum attacks, and Blockchain forks. A blockchain fork occurs when a single chain of blocks gets fragmented into two or more chains that are valid. There are mainly three types of forks: Soft fork, Hard fork, and Temporary fork. Blocks that are no longer part of the current best blockchain because they are overridden by a longer chain are Stale blocks. Quantum attacks can manipulate the transactional information between the blocks which will lead further leads to fraudulent activities.

- Setup: After generating the public and private (asymmetric pair of keys), the input parameters are decrypted and encrypted.
- Registration: Use of IDs in order to generate public or private key as output.
- Vote: Vote parameter is created by the electors and then signature and consequent text is computed.
- Validation: The server chooses the input as a vote and then performs further validation to verify the validity of the vote.
- Append: Cryptographic text is updated.
- Publish: Voting results are published widely.

After the registration process using a facial recognition module, each voter is coupled with a "private and a public key" (Asymmetric Key Pair). Senders send private keys to blockchain Peer-to-Peer network in order to sign messages. In this state, for a point of time these messages are stored and accumulated in a block. Receivers acquire this message recorded on the blockchain and distributed over the network.

➢ *System Execution in Phases:*

- *The Registration Phase*
- *The Removal Phase*
- *The Voting Phase*
- *The Vote Counting Phase*

➢ *Blockchain Workflow: -*

To perform core implementation, hybrid blockchain is being used. In Hybrid BC, a selected portion of the data in a blockchain can be made public while keeping the rest private. Usually verified by private network, but a transaction can also be distributed in the public network for verification. Hybrid Blockchain provides best security among all other types of blockchain. Hybrid BC works according to situation, it carries out advantages from all the types of Blockchains.

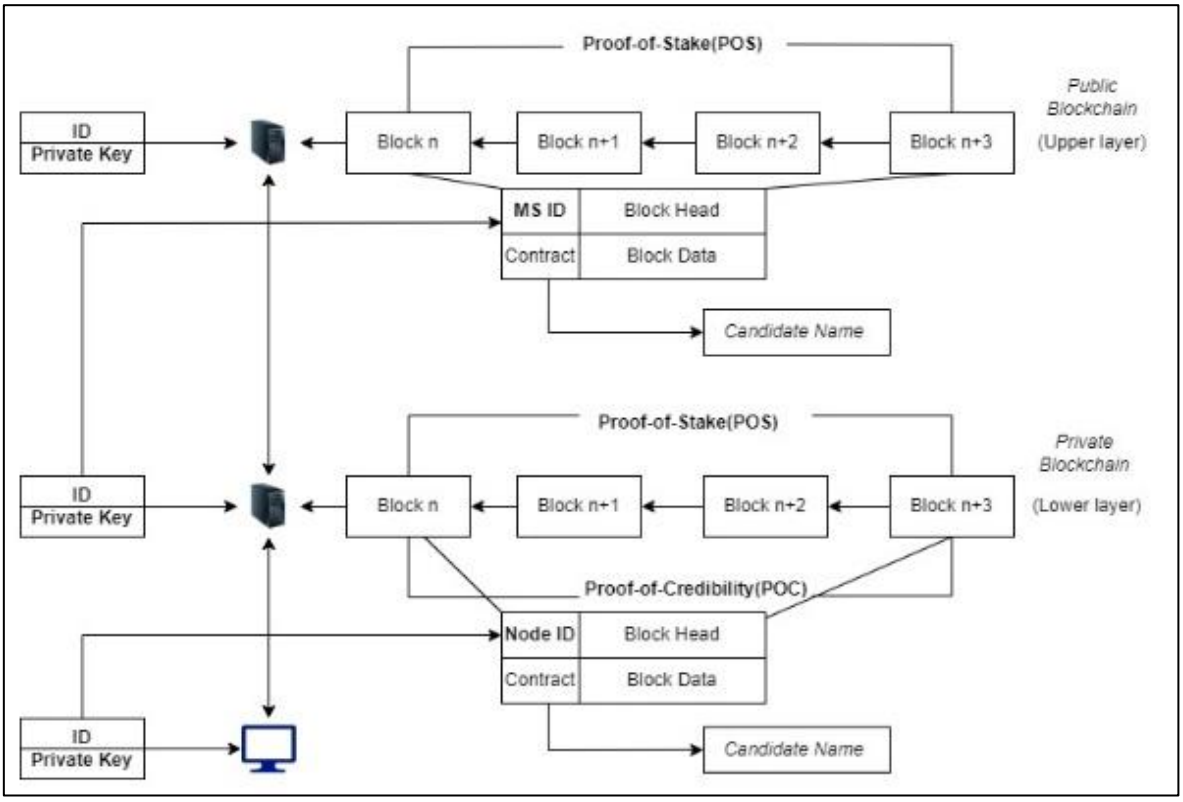

Fig 2 Hybrid Blockchain Consensus Model for e-voting

- *This Model Combines Proof of Credibility (Poc) and Proof of Stake (Pos).*

Proof of Work (PoW) has a negative influence on transaction throughput and latency, which has a direct impact on scalability and performance. Proof of Stake (PoS) has been employed as a result to get near the problems with PoW. In the PoS consensus method, a group of validators take turns proposing and voting on the next block, and the weight(w) of the vote is determined by the value of stake of tokens. PoW is not a computationally difficult system, though. In PoW, multiple nodes & multiple miners compete with each other in order to achieve targeted hash value or can say Complex Mathematical Puzzle. After achieving targeted value, other nodes verify that hash value & post verification that hash is assigned to a block which is (Block Hash). This Block hash represents the work done by the miner which is known as PoW(Proof of Work). Drawback of PoW is that it is expensive.

PoS, a replacement for PoW that is more effective and scalable than PoW, is now available. In PoS, a group of nodes are set up at a stake in order to be candidates, validators, and transaction verifiers. In return, they receive a stake amount (S). Utilizing PoW or PoS can lessen the

possibility of a plot or attack. The hybrid consensus model has drawn a lot of interest as a potential solution to the problems with the PoS & PoW consensus approach.

➢ *QKD Workflow: -*

In this workflow, permission is granted to every user in order to read the information in the blockchain, but only specific entities are permissible in order to become miners.

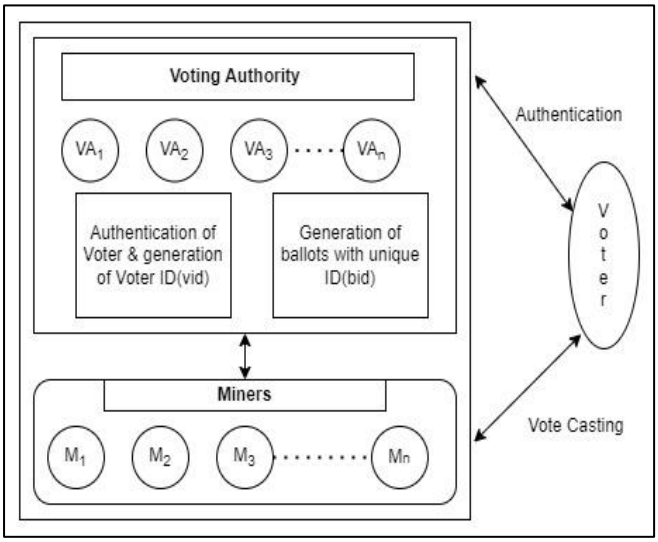

Fig 3 QKD Workflow

Voting Authority (VA): Voting authority (VA) has the inclusive duty to operate the election process. The key responsibility of the VA is to take consideration of the voter's registration by verifying their credentials and create a set of voters (V), who are fit to cast their votes in the elections. Voter set (V) provides a random unique id to every voter, and to begin the voting process this information is recorded by VA for verification. For adding votes into the blockchain VA have the accountability to appoint the authorities to the miners. Likewise For the intent of authentication of the votes, VA will interact with the miners. Lastly, after achieving the inspection of the votes, VA will announce the outcome of the elections.

Voters: Voters are vital to any election process. Before the casting of votes, every voter has to get registered with VA in order to become a part of the eligible set of voters (V). VA will set rules for voters in order to cast vote by broadcasting their votes to the miners.

Miners(m): Miners (m) have the inclusive duty to maintain the blockchain. With the help of the VA, miners will verify the votes which they will collect from the voters. By using "DPoS" protocol vote is lastly appended to the blockchain after verification by miners(m). In this protocol, VA will select 'r ' miners by considering the particular depictions from the voter set(V). All the miners will add the block to the voting blockchain after validating and verifying the votes. If the miner is not available at any point of time, mining power will be shifted to the subsequent accessible miner. Also, when the block metadata is validated by the miners in the majority, only then a block will be taken into consideration as a valid block. Cryptography helps in transmitting secret messages, and "this secret message is secure until the key for encoding that message is secure".

The challenge of secure communication can be ruminated as the difficulty for secure distribution of secret keys. By resolving this challenge, public key cryptography postulated an uprising in this domain, in which the message is encoded and decoded by the public key & private key of the receiver respectively. Acquiring the private key is computationally unattainable for any hacker or external entity. Public key cryptography is thus computationally secure. Though, Shor demonstrated in 1994 that the RSA algorithm can be broken if a quantum computer is used. This sparked renewed interest in developing unconditionally secure key distribution. The keys generated by quantum cryptography are unconditionally secure in comparison to the computational security provided by classical cryptography. This field of unconditionally secure quantum cryptography began in 1984 with Bennett and Brassard's first QKD protocol, now known as the BB84 protocol.

➢ *CNN Workflow: -*

The data collected is pushed to the CNN architecture. This architecture comprises of fully connected layers that are followed by repetitions of a stack of numerous convolutional layers and a pooling layer.

- *User Registration Process using Facial Recognition Module-*

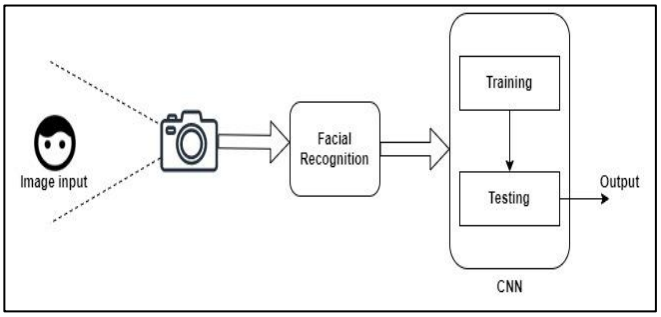

Fig 4 High-Level Overview of the Real-Time Facial Recognition

For face recognition and image classification this algorithm follows the appearance-based approach. Variations in a pool of facial imageries are acquired by using this algorithm. The specific descriptions of individual faces are then encoded using the gathered data. The collection of facial photos is then generally compared to the encoded photographs. The facial photos will be divided up using this process, which is grounded on *information theory*. Then, a small group of distinctive features in images known as "Eigen faces" develop. This is nothing more than the main structural elements of the training set's face photos. The Eigen face method is among the most effective and straightforward techniques for face identification. The distance between pairs of photographs is measured in the Eigen face approach. It is a recognised face if the space is at a lesser extent than a specified threshold value; else, it is an unidentified face. There are two sets of picture blocks in the algorithm: a training set image block and a test set image

block. The trained image, or the Eigen face of the image in the database, is first obtained in the training set image block. The weight W1 is then determined using the training set and the Eigen face. *Img* I is the unknown input Img, that is simply the acquired image, in testing set image block. The input image and then Eigen's face are used to determine the weight We 2. By calculating the average distance between We 1 and We 2, the worth of D is determined. The face is recognised if the D value is smaller than 0. The X and W2 values of the input image are then saved. The face is not recognised if the D value is higher than 0.

IV. IMPLEMENTATION

➢ *Registration Phase*

Implementation is divided into two phases: One is Initial Implementation and the Second is Core Implementation. This is the initial implementation which comprises of Registration Phase, Vote Counting Phase & Removal Phase whereas Core implementation includes voting demonstration using decentralized app and performing various smoke tests.

- *Algorithm 1: Registration Contract for Voter*

```
Input: stub ChaincodeStubInterface, args[] string
Output: Response
1  begin
2  voter:<-InitVoterStruct(args)
3  if IsVoterExist(stub,voter.ID) then
4    return Error("Voter"+voter.ID+"already exists")
5  end
6  PutState(voter.ID,Marshal(voter))
// voter has been registered
7  reglist:<-GetHID_List(stub,GetHL_Key())
8  tcount:<- GetCount(reglist)
9  reglist.Totalcount<-tcount+1
10 Append(reglist.HID List, GetSecHash(voter.ID))
11 PutState(GetHL_Key,Marshal(reglist))
12 return Success("Voter has been registered")
13 end
```

Fig 5 Registration Contract for Voter

Algorithm 1 defines the voter registration contract that takes i/p as an argument and stub and returns a response as an o/p. Steps involved in this contract are:

- The *"Marshal(voter)"* value is stored by using *"PutState"* function. Using *"Marshal()"* , the object named voter is converted to JSON format.
- To return the reg. list *"GetHID_List"* function is used. Then, to return the no. of entries in the list *"GetCount"* function is used. After this, *"tcount"* is updated into the *"reglist"* after getting incremented by one.
- *"GetSecHash()"* hashes the newly reg. voter id. And further appended it into the *"reglist"*.
- *"GetHL_Key"* function generates the key which is then stored with *"reglist"*.
- Lastly, it gives a output with the voter registration message.

- *Algorithm 2: Contract for Candidate's Registration*

```
Input: stub ChaincodeStubInterface, args[] string
Output:Response
1 begin
2 post:<-SetPID(args[2])
3 if ! IsPostExist(stub,post.ID) then
4 return Error("Post"+ post.ID+"is not exist")
5 end
6 candidate:<-InitCandidateStruct(args)
7 if IsCandidateExist(stub,candidate.ID) then
8 return error("Candidate"+candidate.ID+"is already exist")
9 end
10 return Succees("Candidate has been registered")
11 end
```

Fig 6 Registration Contract for Candidate

Algorithm 2 defines the candidate registration contract that takes arguments & stubs as an i/p and returns a response as an o/p. It verifies whether the candidate is listed & gives an error if they are previously listed. Steps intricated in this contract are:

- To store the *"Marshal(post)"* value the *"PutState"* function is used.
- To return the candidate registration list *"GetCandList"* function is used. Then, to return the number of entries in the list the *"GetCount"* function is used. After this, *"tcount"* is updated into the *"creglist"* after getting incremented by one. In *"creglist"*, newly registered candidate is appended.
- *"GetCL_Key"* function generates the key which is then stored with *"creglist"*.
- Lastly, it gives a response with the candidate registration message.

➢ *Removal Phase*

- *Algorithm 3: Withdrawal Cont. for Voter Removal*

```
Input: stub ChainCodeStubInterface, args[] string
Ouput: Response
1 begin
2 vid:<-args[0]
3 voter:<-GetVoter(stub,vid)
4 DelState(voter.ID)
//Voter has been deleted
5 reglist:<-GetHID_List(stub,GetHL_Key())
6 if
7 ! IsElementExist(reglist.HIDList,GetSecHash(voter,ID))
  then
  return Error("No entry found!")
8 end
9 return Success("voter has been removed")
10 end
```

Fig 7 Withdrawal Contract for Removing Voter

Algo. 3 defines the voter withdrawal contract which takes input as arguments and stubs and returns a response as an o/p. Stages intricated in this algorithm are:

- Voter id is initialized with *"[cid]"* in order to get the voter object with an input *"args[0]"*.
- To remove the entry of the voter "DelState" function is used.
- To return the registration list *"GetHID_List"* function is used.
- *"GetHID_List"* is to check the existence of the voter in the *"reglist"* and return an error in case the voter does not exist.
- *"GetHL_Key"* function generates the key which is then stored with *"reglist"*.
- Lastly, it gives a response with the voter withdrawal message.

- *Algorithm 4: Contract for Removing the Candidate*

```
Input: stub ChainCodeStubInterface, args[] string
Ouput: Response
1 begin
2 cid:<-args[0]
3 Candidate:<-GetCandidate(stub,cid)
4 DelState(Candidate.ID)
//Candidate has been deleted
5 reglist:<-GetHID_List(stub,GetHL_Key())
6 if
7 ! IsElementExist(reglist.HIDList,candiate.ID)
  then
  return Error("No entry found!")
8 end
9 return Success("candidate has been removed")
10 end
```

Fig 8 Contract for Removing the Candidate

Algorithm 4 states the contract for removing the candidate in which arguments and stub are taken as input. The steps intricated in this contract are:

- Candidate id is initialized as *"[cid]"* in order to get the candidate object with an input *"args[0]"*.
- To remove the entry of the candidate *"DelState"* function is used.
- To return the registration list *"GetCandList"* function is used.
- To check the existence of the candidate in the *"reglist GetHID_List"* function is used.
- *"GetCL_Key"* function generates the key which is then stored with *"reglist"*.
- Lastly, it gives a response with the voter withdrawal message.

- *Algorithm 5: Contract for Disabling the Voter*

```
Input: stub ChainCodeStubInterface, args[] string
Ouput: Voter, error
1 begin
2 vid:<-args[0]
3 if ! IsVoterExist(stub,vid) then
4 return Error("This voter does not exist"+ vid)
5 end
6 voter:<-GetVoter(stub,vid)
7 if voter.Tokens<=0 then
8 voter.Permit<-false
9 return voter, nil
10 end
11 return voter, errors.New("Removing Tokens:"+voter.Tokens)
12 end
```

Fig 9 Contract for Disabling the Voter

Algorithm 5 states the contract to disable a voter, that takes input as arguments and stub & further gives voter object as a response. Once the voter is disabled, he/she would not be able to vote in the voting process again. The steps intricated in this contract are:

- *"[vid]VoterId"* is initialized with an input *"args[0]"*.
- Corresponding to the vid it gets the voter object.
- If tokens<=0, *"Voter.tokens"* returns the voter object.
- Else, it will return with an error message *"errors.New"*.

➢ *Vote Counting Phase*

- *Algorithm 6: Contract for Vote Counting*

```
Input: stub ChaincodeStubInterface, args[] string
Output:Response
1 begin
2 err:<-checkSecretsValidity()
3 if err!=nil then
4 jsonResp<-"{\"Error\":\"Invalid Secrets"+"\"}"
5 return Error(jsonResp)
6 end
7 Vote Arr:<-ExtractMapInfo(secret_map)
//secretVote contains Candidate information that is extracted from the secret_map.
//The function ExtractMapInfo extracts vote information from the secret_map.
//secret_map where each secret contains candidate information
8 result:<-CountVotes(VoteArr)
9 DisplayResult(result)
10 return Succees(nil)
11 end
```

Fig 10 Contract for Vote Counting

Algorithm 6 defines counting contract, which takes input as arguments and stub and proceeds with a conforming response. Steps involved in this contract are:

- In case of misrepresentation identified, function named *"checkSecretsValidity()"* gives and error as response. It proceeds further if there is no error reported.
- In order to accumulate row-wise secrets, it generates a *"secret_map"*.
- By calling Combine function, candidate information is extracted.

- Information like *"VotesReceived"*, *"CID"* is extracted by function named *"ExtractMapsInfo"* from *"secret_map"*.
- *"CountVotes"* function is then called in order to generate the whole voting fallouts.
- Lastly, a successful response is returned.

V. RESULT & DISCUSSION

Table 4 Time Complexities of Smart Contract Algorithms

Algorithm for smart contracts	Voter	Candidate
Registration	O(N)	O(C)
Withdraw	O(N)	O(C)
Remove/Delete	O(N)	O(C)

There are "C" total number of voters and "N" voters.**The operational intricacies of the proposed smart contracts are depicted in Table 4.**

In, afore-mentioned algorithms, k posts, n voters, t candidates, m authorities, involved in an election process. At voter registration, all formerly recorded entries are required to get access for a particular registration operation in order to authenticate the presence of the specific individual. So, for voter registration the time required is O(n). Also, time required for the authority is O(m), time required for the registration of candidate is O(t) & time required for the post is "O(k)". Furthermore, before execution both Remove and Disable contracts need the precise list in order to check the presence of record. Thus, these processes take "O(m)" time in case of authority, "O(t)" time in case of candidate, "O(n)" time in case of voter, and "O(k)" time in case of post.

- *E.A stands for Existing Approaches & P.A stands for Proposed Approach.*

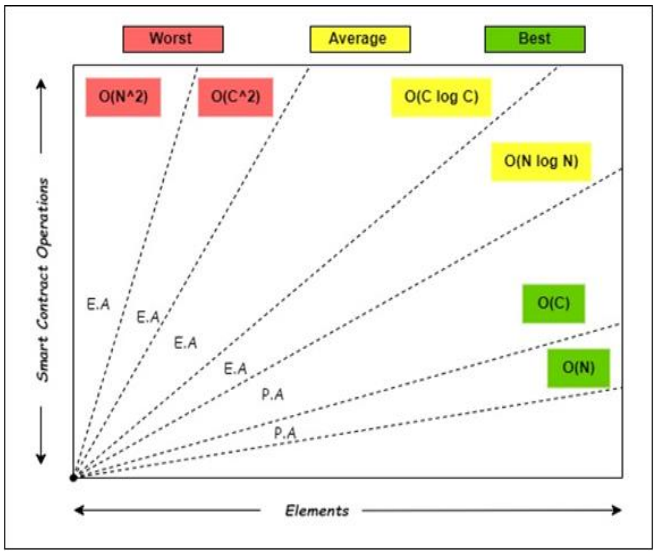

Fig 11 Smart Contracts Complexity Graph

Time complexity for Voter Regis. contract is O(N) and Candidate Registration contract is O(C). Time complexity of Voter Withdraw and Candidate Withdraw are O(N) & O(C) respectively. Time complexity for Voter-Candidate Remove/Delete contract is O(N) & O(C) respectively. This concludes that proposed smart contracts algorithms are best complexities in terms of time and space. However, smart contract algorithms for existing approaches are somewhere lies at O (N^2) and O (N log N) which are horrible and worst complexities in terms of time and space. Fig 12 depicts smart contracts complexity graph by dividing approaches as existing and proposed. Algorithms used for creating smart contracts in previous approaches are not optimized as compare to proposed approach because time complexity of existing approaches lies somewhere between worst O (N^2) to average O (N log N). This shows that our proposed algorithms are optimized and ready to be executed.

VI. CONCLUSION & FUTURE WORK

Regardless of being a reliable technology, blockchain occupies Quantum key distribution order to counter the risk of quantum attacks. The proposed e-voting system is decentralized and does not need to rely on a third party or centralized authority. Authorized and eligible voters can vote using their e-devices connected to the internet. In this system, each voter can validate individual voting actions. This system defends the voting process from fraudulent activities and attacks. Time complexities of smart contracts state that these contracts are best suited in terms of time and space complexities. The proposed system is divided into two implementations: initial implementation and core implementation. The initial implementation is discussed in this paper. Core implementation includes syncing all the technologies and their workflow on a practical level which will be discussed in future work. By performing initial implementation, choices of algorithms and contracts are well-defined to proceed to the next stage of implementation. The study aims to build a "Blockchain-CNN-QKD" grounded elective system which accomplishes all necessities of a voting process by upholding security, privacy, and a scale of decentralization. The objective of this study is to set standards for future secure e-voting systems. Integration of Blockchain, CNN & Quantum Key Distribution leads to a triple-layered security scheme. In blockchain, all blocks are linked cryptographically using quantum key distribution protocol. Sooner, for applied high-security communications, QKD will certainly become a standard.

REFERENCES

[1]. Electronic Voting System Using an Enterprise Blockchain Camilo Denis González, Daniel Frias Mena, Alexi Massó Muñoz, Omar Rojas , and Guillermo Sosa-Gómez , Institute of Cryptography, University of Havana, Havana 10400, Cuba.

[2]. Mustafa, M.K., Waheed, S. (2021). An E-Voting Framework with Enterprise Blockchain. In: Tripathy, A., Sarkar, M., Sahoo, J., Li, KC., Chinara, S. (eds) Advances in Distributed Computing and Machine Learning. Lecture Notes in Networks and Systems, vol 127. Springer, Singapore. https://doi.org/10.1007/978-981-15-4218-3_14

[3]. Kirillov, D., Korkhov, V., Petrunin, V., Makarov, M., Khamitov, I.M., Dostov, V. (2019). Implementation of an E-Voting Scheme Using Hyperledger Fabric Permissioned Blockchain. In, *et al.* Computational Science and Its Applications – ICCSA 2019. ICCSA 2019. Lecture Notes in Computer Science(), vol 11620. Springer, Cham. https://doi.org/10.1007/978-3-030-24296-1_40

[4]. Decentralized Electronic Voting System Based on Blockchain Technology Developing Principals Kateryna Isirova , Anastasiia Kiian , Mariia Rodinko and Alexandr Kuznetsov V. N. Karazin Kharkiv National University, 4 Svobody Sq., Kharkiv, 61022, Ukraine.

[5]. E-Voting with Blockchain: An E-Voting Protocol with Decentralisation and Voter Privacy Freya Sheer Hardwick, Apostolos Gioulis, Raja Naeem Akram, and Konstantinos Markantonakis ISG-SCC, Royal Holloway, University of London, Egham, United Kingdom

[6]. Chaieb, M., Yousfi, S., Lafourcade, P., Robbana, R. (2019). Verify-Your-Vote: A Verifiable Blockchain-Based Online Voting Protocol. In: Themistocleous, M., Rupino da Cunha, P. (eds) Information Systems. EMCIS 2018. Lecture Notes in Business Information Processing, vol 341. Springer, Cham. https://doi.org/10.1007/978-3-030-11395-7_2

[7]. Meeser, Fernando Lobato. "Decentralized, transparent, trustless voting on the ethereum blockchain." (2017).

[8]. Ganji, Raghavendra, and B. N. Yatish. ”ELECTRONIC VOTING SYSTEM USING BLOCKCHAIN.” (2018). 10.1109/MysuruCon 55714.2022.9972479

[9]. Yu, Bin, Joseph K. Liu, Amin Sakzad, Surya Nepal, Ron Steinfeld, Paul Rimba, and Man Ho Au.” Platform-independent secure blockchain-based voting system.” In International Conference on Information Security, pp. 369-386. Springer, Cham, 2018. https://doi.org/10.1007/978-3-319-99136-8_20

[10]. Liu, Yi, and Qi Wang. ”An E-voting Protocol Based on Blockchain.” IACR Cryptology ePrint Archive 2017 (2017) https://ia.cr/2017/1043

[11]. Yi, Haibo. ”Securing e-voting based on blockchain in the P2P network.” EURASIP Journal on Wireless Communications and Networking 2019, no. 1 (2019): 137 https://doi.org/10.1186/s13638-019-1473-6

[12]. Ayed, Ahmed Ben. ”A conceptual secure blockchain-based electronic voting system.” International Journal of Network Security & Its Applications 9, no. 3 (2017)

[13]. Z. Zhao and T.-H. Hubert Chan, How to vote privately using bitcoin, in Proc. Int. Conf. Inf. Commun.Security (Beijing, China), 2015, pp. 82–96. https://doi.org/10.1007/978-3-319-29814-6_8

[14]. S. Shukla et al., Online voting application using ethereum blockchain, in Proc. Int. Conf. Advances Comput., Commun. Inform. (Bangalore, India), 2018, pp. 873–880. 10.1109/ICACCI.2018.8554652

[15]. S. Zhang, L. Wang, and H. Xiong, Chaintegrity: Blockchain-enabled large-scale e-voting system with robustness and universal verifiability, Int. J. Inf. Secure. 19 (2019), 1–19. https://doi.org/10.1007/s10207-019-00465-8

[16]. J. P. Cruz and Y. Kaji, E-voting system based on the bitcoin protocol and blind signatures, IPSJ Trans. Math. Model. Appl. 10 (2017), 14–22. http://id.nii.ac.jp/1001/00178502/

[17]. S. Bistarelli et al., An end-to-end voting-system based on bitcoin, in Proc. Symp. Appl. Comput. (Marrakech, Morocco), 2017, pp. 1836–1841. https://doi.org/10.1145/3019612.3019841

[18]. S. Pareek et al., E-voting using ethereum blockchain, IJRTI. 3 (2018), 2456–3315.

[19]. K. M. Khan, J. Arshad, and M. M. Khan, Investigating performance constraints for blockchain based secure e-voting system, Future Gener. Comput. Syst. 105 (2020), 13–26. https://doi.org/10.1016/j.future.2019.11.005

[20]. G. G. Dagher et al., Secure voting system using Ethereum’s blockchain, in Proc. Int. Conf. Inf. Syst, Security Privacy (Madeira, Portugal), 2018, pp. 96–107. http://dx.doi.org/10.5220/0006609700960107

[21]. E. Yavuz et al., Towards secure e-voting using Ethereum blockchain, in Proc. Int. Symp. Digital Forensic Security (Antalya, Turkey), 2018, pp. 1–7. 10.1109/ISDFS.2018.8355340