Volume: 11 Issue: V Month of publication: May 2023
DOI: https://doi.org/10.22214/ijraset.2023.51881

International Journal for Research in Applied Science & Engineering Technology (IJRASET)
ISSN: 2321-9653; IC Value: 45.98; SJ Impact Factor: 7.538
Volume 11 Issue V May 2023- Available at www.ijraset.com

Blockchain-Based Know Your Customer (KYC) System using Advanced Encryption Standard Algorithm

N Deepak[1], Joy Patel[2], Likith M J[3], Mihika Srivastava[4], Dr. Hema Jagadish[5]
[1, 2, 34]*Department of Information Science and Engineering, Bangalore Institute of Technology, VV Puram, Bangalore*
[5] *Associate Professor, Department of Information Science and Engineering, Bangalore Institute of Technology, VV Puram, Bangalore*

Abstract: Know Your Customer (KYC) is a crucial regulatory obligation for banks and financial institutions to record customer information before providing financial services. However, traditional KYC processes are slow, costly, and prone to errors. To address these challenges, a proposed solution aims to use blockchain technology, smart contracts, Ganache, and Advanced Encryption Standard (AES) encryption in the KYC process. This solution leverages the immutability and transparency of the blockchain to create a secure and efficient KYC process. Smart contracts approve and secure the KYC process, eliminating intermediaries and reducing time and cost. Ganache tests and deploys smart contracts, while AES encryption ensures the confidentiality and integrity of customer data. The proposed architecture involves a proof-of-concept system that can be used as a basis for further development and implementation in the financial industry and other regulated sectors. By leveraging these technologies, the proposed solution can revolutionize KYC processes, providing a secure, decentralized, and efficient process while ensuring the privacy of customers.
Keywords: Blockchain, Know-Your-Customer, Decentralized, AES, Cryptography, KYC Smart Contract

I. INTRODUCTION

Know Your Customer (KYC) is a procedural endeavor undertaken by financial institutions, wherein they acquire discerning particulars pertaining to the personal identity and residential whereabouts of prospective acquirers and borrowers. It is a regulatory-driven undertaking entailing diligent scrutiny to authenticate the identities of clients. This process aids in preventing the misuse of banking services. The banks bear the responsibility of executing the KYC procedure when establishing accounts, and they are also obligated to regularly update their customers' KYC information. KYC procedures can be manual, time-consuming, and duplicative across different institutions. Every company must ascertain its identity through suitable means, and this holds particular significance for financial institutions. From this ‘Know Your Customer,’ or KYC protocols emerge to aid corporations in ascertaining the identity of their business partners. It proffers a comprehensive investigation and engenders a feeling of assurance for financial institutions and the banking domain. Typically, this involves a protracted and meticulous process wherein specific documentation is provided, and various forms of scrutiny and validation take place.

Primarily KYC helps financial institutions to prevent identity thefts, money laundering, terrorist financing, and profiling and eliminating runaway creditors. Within the conventional KYC system, each bank independently carries out its own identity verification process, wherein every user undergoes individual scrutiny by a distinct organization or government entity. Every time one wishes to open a new account in a bank, one will have to undergo the whole KYC process from scratch. Hence, there is a waste of time checking each identity from the start. KYCs also need to be regularly updated, like a change in a phone number or updating of address. This involves a lot of redundancy and manual effort.

In a nutshell, the following are the problems with current KYC:

1) *Money and Time Wasted on False Positives:* A false positive arises when a legitimate customer is identified for heightened due diligence owing to a name resemblance with an entry on the Political Exposed Persons (PEPs) or Sanctions list.
2) *Undetected Risks Due to Poor Data:* On the other hand, a false negative arises when a client who is linked to a sanctioned entity is not identified for enhanced due diligence. The failure to detect a sanctioned entity can have serious consequences, including financial penalties, regulatory enforcement actions, and reputational harm.

3) *Lack of Detail of Alerts leads to Inefficiencies:* When a corporate customer appoints a new director or undergoes a change in ownership, it frequently triggers an alert. However, based on our experience, a significant issue arises as the alerts lack sufficient information to enable a compliance officer to make an informed, real-time, risk-based decision.
4) *Poor Record Keeping:* The existence of disparate systems and, in certain instances, reliance on manual record-keeping has emerged as a significant hurdle when regulators initiate investigations or audits.
5) *Lack of Configurability:* Numerous ongoing monitoring solutions present limited flexibility in terms of reconfigurability, resulting in the flagging of issues that may not be relevant to your organization's specific concerns or priorities.
6) *Holistic Approach To Compliance Is Better Than A Data-Driven Model:* As data volumes persistently expand, certain businesses have opted to align themselves with a "data provider" for their compliance solutions

A Distributed Ledger Technology (DLT) known as Blockchain manifests as a decentralized register, wherein the digital sequence simulates a virtual ledger. Each novel record signifies an appended block, interconnected with the chain. All participants or entities possess a replica of the blockchain to authenticate and expose any illicit or dubious transactions. The Blockchain framework and its distributed ledger technology empower the aggregation of data from diverse service providers into an impervious and unalterable database, obviating the requirement for third-party authentication of information. The autonomous Blockchain itself employs the SHA-1 algorithm to fortify each block or transaction within the chain cryptographically. This encompasses the encryption and decryption of sensitive data. The confidential data of a user can solely be accessed through their exclusive private key, strictly held in their possession. Consequently, a system materializes wherein a user would only need to undergo the KYC process once to corroborate their identity. By eliminating intermediaries and averting repetitive KYC procedures across multiple banks, the blockchain methodology emerges as an enduring and proficient alternative. The integration of KYC information onto the Blockchain empowers financial institutions to deliver enhanced compliance outcomes, augment efficiency, and elevate the customer experience. With thorough background checks and verifications in place, this allows banks to flag or mark suspicious customers, informing all banks or entities in the network. This gives banks a chance to either further probe into the customer or not give out credit or loans to the flagged entity. This ensures the security of not just one bank, but all the financial institutions which are a part of the blockchain network.

II. RELATED WORK

Researchers have proposed various machine learning (M) models to automate the process of detecting money laundering activities. Mohannad Alkhalili Mahmoud H. Qutqut et al. [1] developed a model using Support Vector Machines (SVM) that achieved higher accuracies in predicting transaction decisions. Ashwini Kumar et al. [2] utilized big data analytics techniques and the Naive Bayes classification method, achieving an accuracy score of 0.8125 in detecting money laundering transactions. David Macedo et al. [3] focused on document segmentation in online customer identification and proposed the use of the U-Net model for text detection. They also suggested model optimization using Octave Convolutions to improve computational efficiency.
Moreover, Jose-de-Jesus Rocha-Salazar et al. [4] emphasized the effectiveness of data visualization techniques, particularly link analysis, in identifying suspicious activities and detecting money laundering. Kishore Singh et al. [5] introduced a new method for AML and terrorism financing detection based on typologies described in Financial Action Task Force reports, which reduced false positives and improved accuracy compared to previous rule-based methods.
Ricardo Azevedo Araujo [8] postulated an incentive-driven tactic to combat money laundering wherein financial institutions play a crucial role in reporting suspicious activities. However, the problem of confidential information and hidden selection presents a challenge in the effectiveness of the regulation.
Ismail Alarab et al. [10] proposed the use of graph neural networks to predict illicit behavior in the Bitcoin blockchain. By combining Graph Convolutional Networks (GCN) with linear layers, they achieved better performance in detecting illicit transactions.
Amr Ehab Muhammed Shokry et al. [11] focused on identifying covert patterns, syndicates, and transactions related to money laundering for combating terrorism financing. Their unsupervised machine learning methodology showed promising results in identifying similarities and hidden patterns across transactions and suspicious accounts.
Rasmus Ingemann Tuffveson Jenson et al. [12] conducted a comprehensive literature review on anti-money laundering (AML) in banks. They proposed a standardized terminology and identified the shortage of public datasets as a major challenge in the scientific literature on statistical and machine learning methods for AML.

Joana Lorenz et al. [13] addressed the challenge of limited labels for detecting money laundering through cryptocurrencies. They introduced an active learning approach that can perform as well as a fully supervised method with just 5% of the labels, mimicking real-world scenarios where limited labels are available.

In addition to machine learning, the use of blockchain technology shows promise in addressing AML challenges. Yue Shi et al. [14] examined the potential of blockchain technology in cybersecurity, highlighting benefits such as enhanced security and data integrity. They proposed an enhanced access control strategy using attribute-based encryption. Joe Abou Jaoude et al. [15] conducted a literature review on blockchain technology's applications and emphasized its decentralized nature, which eliminates the requisite for trust amidst stakeholders and enables trustless transactions.

Regarding KYC, Prakash Chandra Mondal et al. [9] presented a method for secure and seamless financial access through dynamic KYC-based transaction authorization.

This approach reduces the risk of key theft and the need for additional hardware, making it cost-effective. KYC blockchain offers significant advantages by leveraging blockchain's security, anonymity, and data integrity features. By storing customer identification information securely on the blockchain, it ensures trustless and tamper-resistant KYC processes, reducing the risk of identity fraud and enhancing compliance with AML regulations.

In conclusion, machine learning approaches, such as SVM, Naive Bayes, and graph neural networks, demonstrate promise in automating the detection of money laundering activities.

These models have shown higher accuracies in predicting transaction decisions and identifying suspicious patterns. Nevertheless, there are certain limitations to consider.

One drawback is the reliance on historical labeled data in supervised learning approaches, limiting their effectiveness against sophisticated money launderers who constantly adapt their techniques. This highlights the need for unsupervised machine learning techniques, as highlighted by Prof. Dr. Nevine Makram Labib et al. [7].

Unsupervised learning can discover new patterns and detect all accounts and groups involved in money laundering, reducing the risk of false positives. Future research should focus on contrasting and suggesting unsupervised machine learning techniques explicitly for combating terrorism financing.

Another challenge is the computational intensity of some models, such as the U-Net model proposed by David Macedo et al. [3], which may not be practical for deployment on mobile devices characterized by constrained computational capabilities. Optimizing models using techniques like Octave Convolutions can help address this issue.

Furthermore, the effectiveness of AML regulations and activities heavily relies on the willingness and ability of financial institutions to report suspicious activities, as highlighted by Ricardo Azevedo Araujo [8]. The problem of confidential information and hidden selection can hinder the regulation's effectiveness and create a barrier to combating money laundering.

In the realm of blockchain technology, KYC blockchain presents an opportunity to enhance AML efforts. By securely storing customer identification information on the blockchain, KYC processes become trustless and tamper-resistant. This reduces the risk of identity fraud and improves compliance with AML regulations. However, further research is needed to explore the practical implementation and scalability of KYC blockchain solutions.

In summary, machine learning algorithms have exhibited promise in automating AML processes and detecting money laundering activities. Unsupervised learning approaches and optimization techniques can enhance their effectiveness. Additionally, leveraging blockchain technology, particularly KYC blockchain, can significantly improve the security and efficiency of AML efforts. Continuous research and innovation in these areas are crucial to stay ahead of evolving money laundering tactics and safeguard the integrity of the global financial system.

III. PROBLEM DEFINITION

1) To design a system for the KYC process to assist banks as well as the public in registering their details securely and efficiently.
2) The traditional KYC process requires users to get their KYC performed several times for differentapplications as the process differs for each.
3) KYC is executed repeatedly for different applications as the process differs for each. Banks primarily use KYC to prevent money laundering, and identity theft and prevent terrorist financing and runawayauditors. For this purpose, decentralizing the process to increase security and reusability simplifies the process for both the people who are registering as well as the regulators who are asking for the details

IV. OBJECTIVES

1) The system's objective is to enhance security, and transparency, and address trust concerns.
2) The goal is to surmount the drawbacks of the conventional KYC system by decentralizing the process through blockchain and decentralized ledger technology, thereby boosting efficiency and usability.
3) The challenge at hand is ensuring highly secure KYC processes that involve personal and bank details, as any breach could result in severe losses for both users and banks. The implementation seeks to simplify access to user details while maintaining their reusability.
4) The proposed system leverages Blockchain technology to distribute the database of executed or shared records among participating parties.
5) Every transaction is verified by the majority of system participants, and it contains a comprehensive record of each transaction. If any irregularity is detected, that specific transaction or block is flagged.

V. SYSTEM DESIGN

A KYC utility system based on blockchain technology will enable the financial and banking sectors to emancipate the process of identity verification. Currently, the data is collected and stored in a centralized system, such as a repository. With the introduction of blockchain solutions to handle the KYC process, data will be available on a decentralized network and can, therefore, be accessed by third parties directly after permission has been given. The blockchain-based KYC system will also offer better data security by ensuring that data access is only made after a confirmation or permission is received from the relevant authority.

This will eradicate the possibility of unauthorized entry, thus bestowing individuals with heightened authority over their own data. The ledger will furnish an archival log of all disseminated documents and compliance endeavors undertaken for each clientele. Moreover, Blockchain technology proves advantageous in discerning entities endeavoring to fabricate deceptive histories. Within the parameters of data protection regulations, the data enshrined within the blockchain remains immutable and can be scrutinized to uncover anomalies, directly targeting illicit conduct.

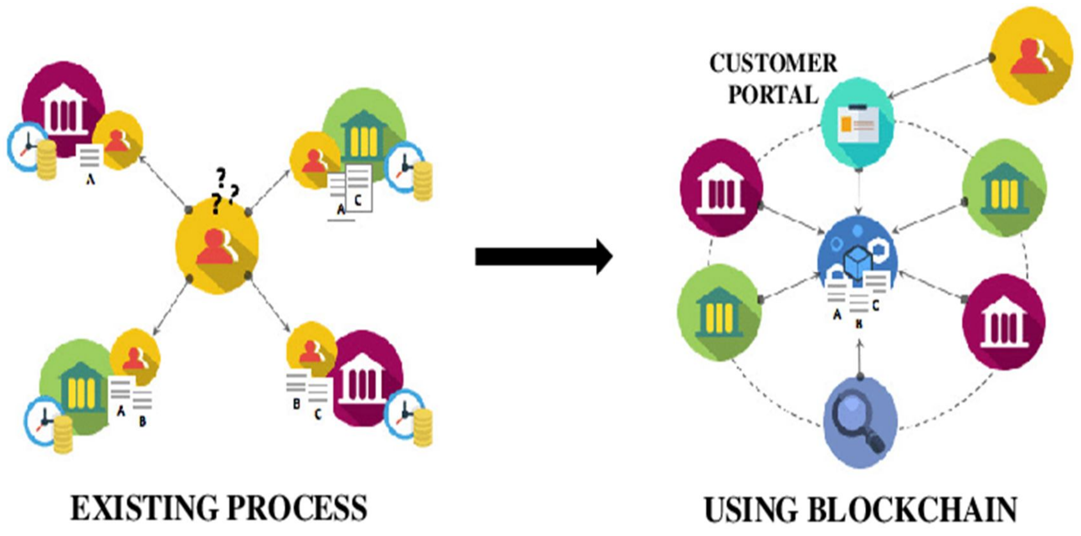


Figure 1: Traditional KYC vs Blockchain KYC

The KYC process is essential in establishing a customer's identity and ensuring compliance with regulatory requirements. As you can see in the figure 2, **(1)** The procedure commences with the initiation of customer providing their personal information to the KYC service provider, which can be a third-party provider or the financial institution itself. (**2, 3, 4)** The information provided by the customer is then verified for its authenticity and accuracy, which involves checking the validity of the documents and cross-checking the information against various databases. The verification process is critical in preventing fraud and financial crimes such as money laundering and terrorist financing. Once the customer's information is verified, it is stored on the blockchain, which is a secure and immutable distributed ledger technology.

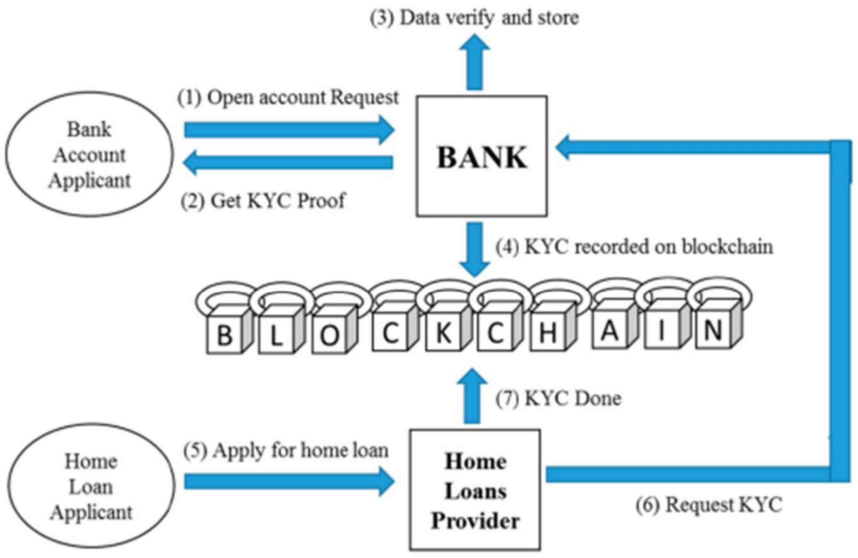


Figure 2: KYC Blockchain Framework

In the next step of the KYC process, **(5)** the customer can request access to a financial service, such as opening a bank account or applying for a loan. **(6)** The financial institution then requests the customer's KYC information from the blockchain. **(7)** Upon receiving the information, the financial institution verifies it again to ensure its accuracy and completeness. The use of blockchain technology in the KYC process provides enhanced security, transparency, and data integrity.
The following figures present the sequential flow of the KYC system in terms of adding, viewing, and modifying customer data. The measure is taken to prevent the fraudulent transaction from happening. There are three entities: Customer, Bank, and KYC Blockchain network, showing the different scenarios much better. The arrows indicate the direction of the application flow.

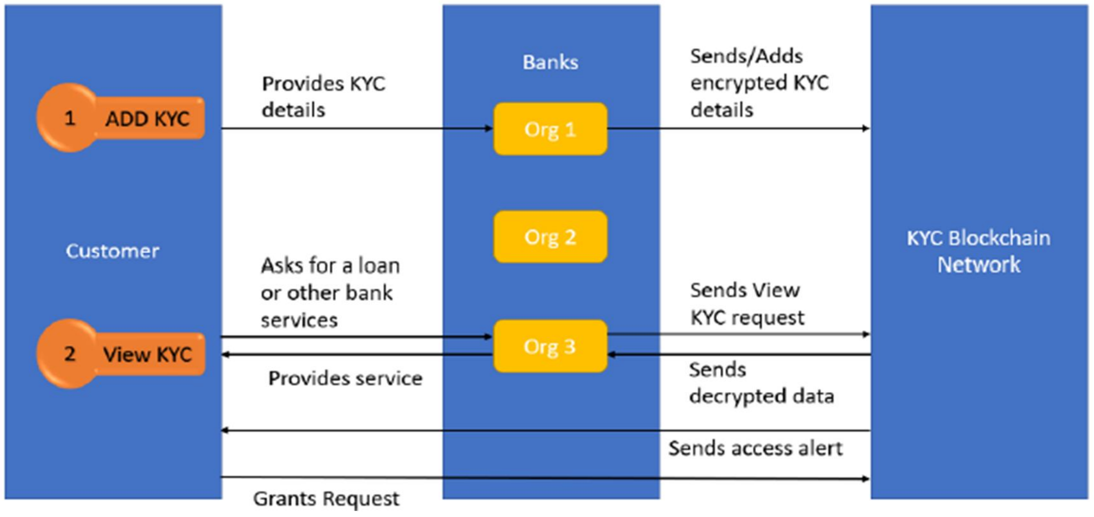


Figure 3: Adding and Viewing KYC details

When a customer adds their personal information to the blockchain through Bank Org 1, the information is encrypted and securely stored on the blockchain. This ensures that the data is tamper-proof and immutable, and can only be accessed by authorized parties with the proper decryption key.When another bank or entity needs to view a customer's KYC information, they must first request access from the customer's bank. The customer is notified of the access request and must provide approval before the data is decrypted and shared. This adds an extra layer of security to the system, ensuring that customer data is not shared without their explicit consent. The use of blockchain technology and cryptography techniques such as AES encryption ensures that the KYC information is stored and shared in a transparent, secure, and privacy-preserving manner.

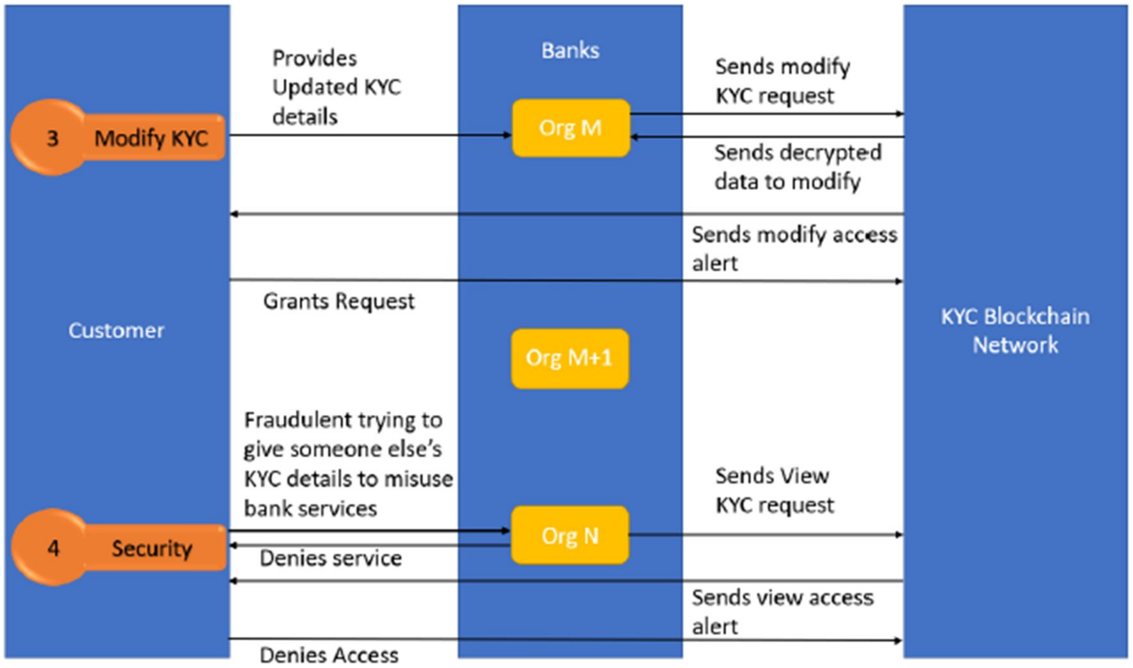


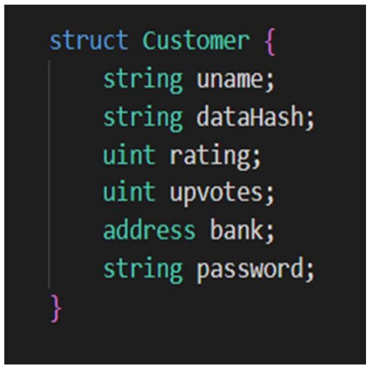
Figure 4: Modifying KYC Details and Security assurance

When a bank needs to modify a customer's KYC information, it sends a request to the blockchain network. The request is then sent to the customer for approval, and only after the customer approves, the decrypted data is sent back to update the KYC information. This guarantees that the customer's information remains unaltered unless explicitly authorized by them, thereby upholding the integrity of the data residing within the blockchain. In the event that a bank harbors suspicions regarding the validity or legitimacy of a transaction, it dispatches a request to the KYC blockchain network, subsequently triggering a request forwarded to the customer for transaction verification. If the customer denies access to the transaction, it is flagged and stopped, preventing any unauthorized modifications to the KYC information or any fraudulent transactions from taking place. This ensures the security and trustworthiness of the KYC blockchain system.

VI. IMPLEMENTATION

Smart contracts are autonomous computer programs that autonomously enforce the provisions of a contract upon the fulfillment of predefined conditions. These intelligent contracts are constructed on the foundation of blockchain technology and employ cryptography to ensure the security and validation of transactions. In our system, the contract is denoted as "kyc.sol," with the ".sol" extension signifying the utilization of the solidity programming language for composing the contract. The contract comprises three primary data structures, namely Customer, Organisation (Bank), and Request, each of which is defined as a struct.

The Customer struct has the following fields:

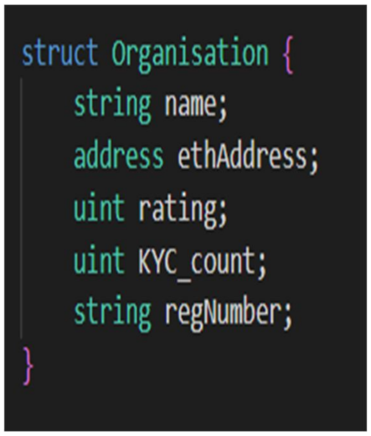


```
struct Customer {
    string uname;
    string dataHash;
    uint rating;
    uint upvotes;
    address bank;
    string password;
}
```

Figure 5: Customer Entity

- uname: a string representing the username of the customer
- dataHash: a string representing the hash of the customer's data
- rating: an integer representing the rating given to the customer based on their regularity
- upvotes: an integer representing the number of upvotes received from banks
- bank: an Ethereum address representing the address of the bank that validated the customer account
- password: a string representing the customer's password

The Organisation struct has the following fields:

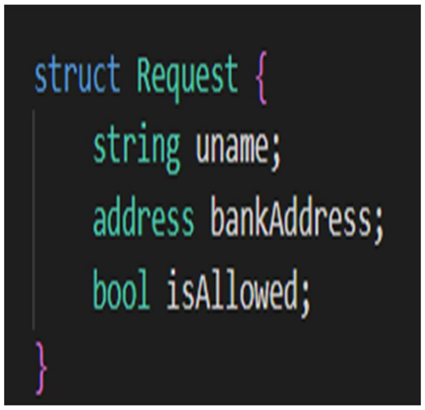


```
struct Organisation {
    string name;
    address ethAddress;
    uint rating;
    uint KYC_count;
    string regNumber;
}
```

Figure 6: Organization Entity

- name: a string representing the name of the bank or organization
- ethAddress: an Ethereum address representing the address of the bank or organization
- rating: an integer representing the rating predicted upon the number of valid/invalid verified accounts
- KYC_count: an integer indicating the number of KYCs verified by the bank or organization
- regNumber: a string representing the registration number of the bank or organization

The Request struct has the following fields:



```
struct Request {
    string uname;
    address bankAddress;
    bool isAllowed;
}
```

Figure 7: Request Entity

- uname: a string representing the username of the customer requesting validation
- bankAddress: an Ethereum address representing the address of the bank that the request was sent to
- isAllowed: a boolean representing whether or not the bank has been allowed to validate the customer

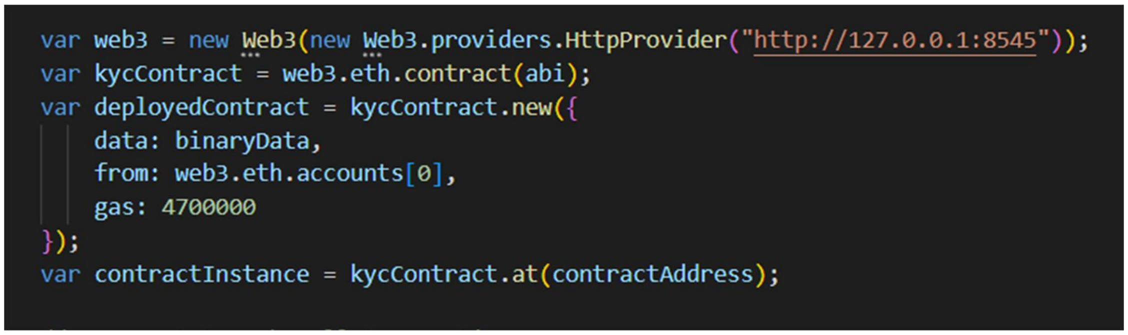
In addition to the three main data structures, there are three arrays that store all of the Customer, Organisation, and Request objects. The functions defined in the contract are used to manipulate these data structures and perform operations such as adding and removing banks, adding requests, and allowing banks to validate customers as given below:

Table 1: KYC Smart Contract Functions

Function Name	Input	Output	Description
addBank	string Uname, string name, string regNumber, uint KYC_count	bool	Adds a new bank to the system
removeBank	string Uname	bool	Removes a bank from the system
modifyBank	string Uname, string newName, string newRegNumber	bool	Modifies the name or registration number of a bank
viewBanks	None	string[]	Returns an array of all the bank names
addCustomer	string Uname, address bank, string password	bool	Adds a new customer to the system
removeCustomer	string Uname	bool	Removes a customer from the system
modifyCustomer	string Uname, string DataHash	bool	Modifies the data hash of a customer
viewCustomer	string Uname	string	Returns the data hash of a customer
updateRatingCustomer	string Uname, bool ifIncrease	uint	Increases or decreases the rating of a customer
updateRating	address bankAddress, bool ifAdded	uint	Increases or decreases the rating of a bank
checkCustomer	string Uname, string password	bool	Checks if the provided password is correct for the given customer
setPassword	string Uname, string password	bool	Sets the password for a customer account
getBankName	address ethAcc	string	Returns the name of a bank based on its Ethereum address
getBankEth	string uname	address	Returns the Ethereum address of a bank based on its name
getBankReg	address ethAcc	string	Returns the registration number of a bank based on its Ethereum address
getBankKYC	address ethAcc	uint	Returns the KYC count of a bank based on its Ethereum address
getBankRating	address ethAcc	uint	Returns the rating of a bank based on its Ethereum address
getCustomerBankName	string Uname	string	Returns the name of the bank associated with a given customer account
getCustomerBankRating	string Uname	uint	Returns the rating of the bank associated with a given customer account
getCustomerRating	string Uname	uint	Returns the rating of a given customer

Ganache is a personal blockchain for Ethereum development. It gives a local blockchain environment that developers can use to evaluate and deploy their smart contracts. Ganache is especially useful for developing KYC blockchain applications because it allows developers to evaluate and experiment with their code in a safe, sandboxed environment before deploying it on the live Ethereum network.

Ganache also comes with a suite of developer tools, including a user-friendly GUI and a command-line interface, that make it easy to interact with the blockchain and test transactions. Ganache provides a set of pre-funded Ethereum accounts for testing and development purposes. These accounts come with a private key and a balance of 100 Ether by default. The private keys of these accounts are not secure and should not be used in a production environment. In the context of KYC blockchain development, these accounts can be used to simulate various use cases and test different scenarios, such as adding new banks, removing banks, and granting access to customer KYC data.



```
var web3 = new Web3(new Web3.providers.HttpProvider("http://127.0.0.1:8545"));
var kycContract = web3.eth.contract(abi);
var deployedContract = kycContract.new({
    data: binaryData,
    from: web3.eth.accounts[0],
    gas: 4700000
});
var contractInstance = kycContract.at(contractAddress);
```

Figure 8: Ganche Connection to KYC Contract

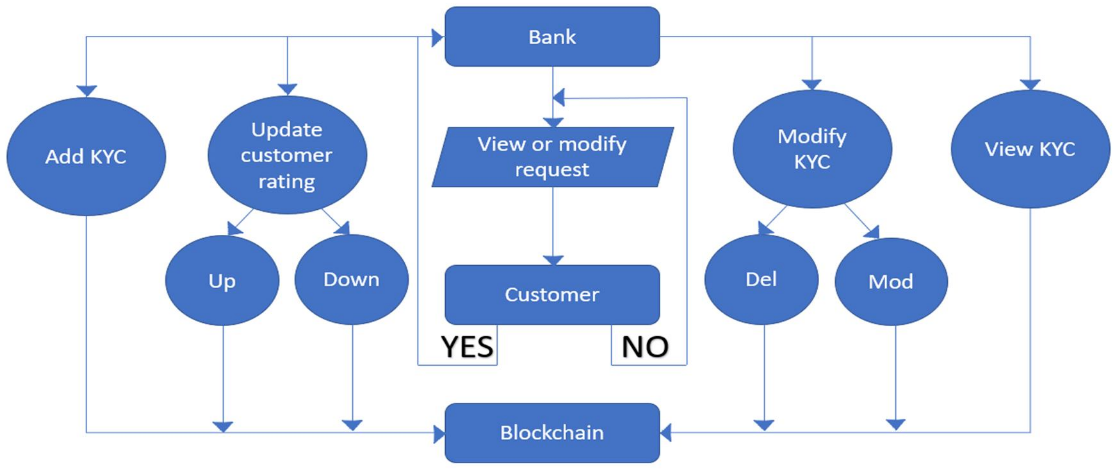


Figure 9: Technical Workflow

- ADD_KYC: Enables users to give their personal information to the blockchain. Once the user has been authenticated and authorized to use the system, they can provide their personal information, such as name, address, date of birth, and other relevant data. This information is then encrypted and added to the blockchain, making it tamper-proof and immutable.
- MODIFY_KYC: Allows users to update their personal information on the blockchain. For example, if a user changes their address or phone number, they can modify the relevant fields in their profile. The modified data is encrypted and added to the blockchain as a new block, ensuring that the original information is preserved and the change is recorded in a transparent and secure manner.
- VIEW_KYC: Allows authorized parties to view and verify the information stored on the blockchain. This can include government agencies, financial institutions, or other entities that need to verify the identity and personal information of customers. The data is securely stored on the blockchain, and authorized parties can access it by using their private keys to decrypt the information

- CUSTOMER_RATING: Can be used to assign a trust score or rating to users based on their history of providing accurate information and complying with KYC regulations. The rating can be based on factors such as the accuracy of the user's information, their compliance with KYC regulations, and their overall level of trustworthiness. The rating can be used by financial institutions or other entities to assess the risk associated with a particular customer and determine the appropriate level of due diligence required.

There is an in-built feature that does not require manual input but rather instead it is automatic called **BANK_RATING**, which is a measure or score assigned to a bank to evaluate its performance and reliability. It provides an indication of the bank's credibility and trustworthiness in the financial industry. In the context of the KYC process stored on the blockchain, the bank rating can be incremented or decremented based on the number of KYC processes conducted and stored. When a bank successfully completes a KYC process for a customer and stores the verified information on the blockchain, it demonstrates its commitment to compliance and risk management. Each completed and verified KYC process can contribute positively to the bank's rating. The more KYC processes the bank successfully completes, the higher its rating may increase. This indicates that the bank has a robust and efficient KYC process in place, assuring adherence to regulatory mandates and mitigating potential hazards linked to fraudulent endeavors. On the other hand, if the bank fails to perform the KYC process accurately or faces issues with data integrity, it may lead to a decrement in its rating. Instances such as incomplete KYC information, fraudulent activities detected, or violations of regulatory guidelines can negatively impact the bank's rating. This signals potential weaknesses in the bank's compliance procedures and raises concerns about its reliability and ability to handle customer data securely.

VII. METHODOLOGY

Advanced Encryption Standard (AES) is a widely used encryption algorithm that is considered to be one of the most secure encryption methods available today. It embodies a symmetric encryption algorithm, thereby signifying that an identical key is employed for both the encryption and decryption of data. AES was selected by the U.S. National Institute of Standards and Technology (NIST) in 2001 as the standard for securing sensitive government information and is now used in a wide range of applications, including in the financial industry. AES is a symmetric encryption algorithm that uses a block cipher to encrypt and decrypt data. In the context of KYC blockchain, AES encryption can be used to secure the personal and sensitive information of customers that is stored on the blockchain. When a customer provides their personal information to the KYC service provider, that information can be encrypted using AES before being stored on the blockchain. Overall, the use of AES encryption in KYC blockchain provides an additional layer of security and ensures that sensitive information is protected at all times. The process of AES encryption involves several steps:

1) Key Generation: The AES algorithm requires a key of 128, 192, or 256 bits to encrypt and decrypt data. The key expansion process converts the key into a set of round keys that will be used in the encryption and decryption rounds.
2) Initial Round: The first step in the encryption process is to XOR the plaintext block with the first-round key Rounds: The number of rounds depends on the key size. For a 128-bit key, there are 10 rounds, for a 192-bit key, there are 12 rounds, and for a 256-bit key, there are 14 rounds. Each round consists of four steps: SubBytes, ShiftRows, MixColumns, and AddRoundKey.
3) SubBytes: In this step, each byte of the block is replaced with a corresponding byte from a lookup table called the S-box.
4) ShiftRows: In this step, the rows of the block are shifted by different amounts depending on the row number.
5) MixColumns: In this step, each column of the block is multiplied with a fixed matrix.
6) AddRoundKey: In this step, the round key is XORed with the block.
7) Final Round: The final round is similar to the other rounds, except that it does not include the MixColumns step.
8) Output: The output of the encryption process is the ciphertext block, which is the result of the final round.

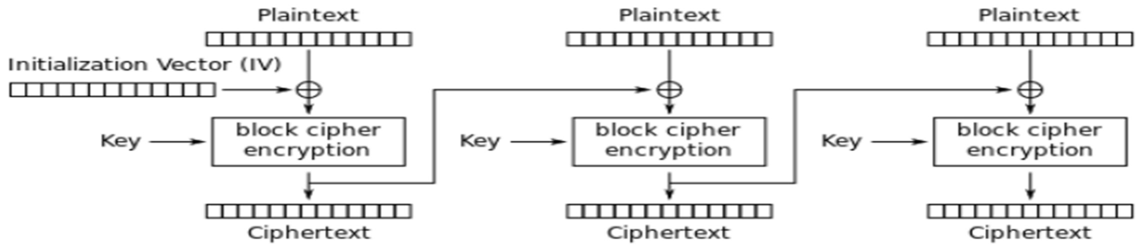


Figure 10: AES Encryption Process

The decryption process is similar to the encryption process, except that the order of the round keys is reversed, and the Mix Columns step is replaced with an inverse Mix Columns step.

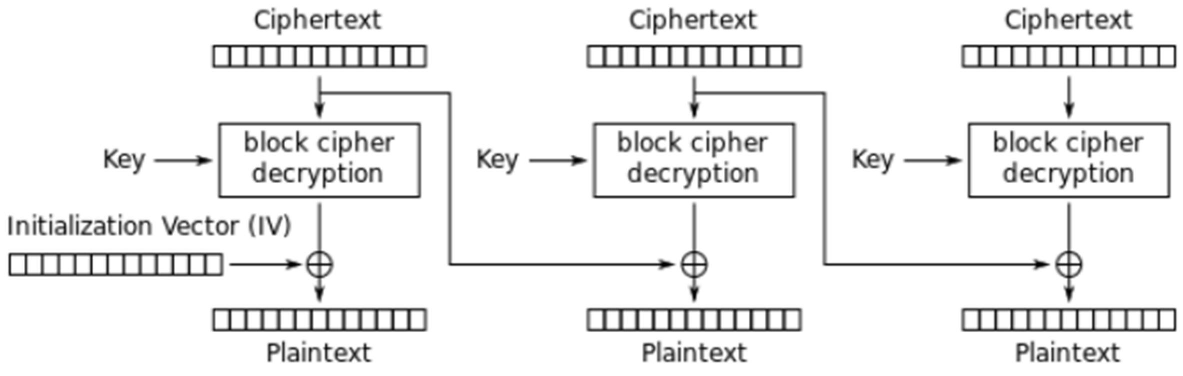


Figure 11: AES Decryption

In the KYC Blockchain System, the cryptographic technique for encryption and decryption of customer data when sending and retrieving from the KYC network is done with the help of the Advanced Encryption Standard (AES). Now the choice of using Symmetric cryptography such as AES instead of Asymmetric cryptography such as Rivest-Shamir-Adleman (RSA) is because of the following technical reasons:

- Efficiency: Symmetric cryptography is generally faster and more efficient than asymmetric cryptography because it involves the use of a single key for both encryption and decryption. This is particularly important for a KYC blockchain system, which may require the encryption and decryption of large amounts of data.
- Key Management: Symmetric cryptography requires the use of only one key, which can simplify the key management process. In contrast, asymmetric cryptography involves the use of two keys (public and private), which can increase the complexity of key management and increase the risk of key compromise.
- Security: While asymmetric cryptography provides better security for key exchange and digital signatures, symmetric cryptography can also provide strong security for data encryption when used with appropriate key management practices.
- Resource Constraints: Asymmetric cryptography requires more processing power and resources than symmetric cryptography. In a resource-constrained environment, such as a blockchain system, it may be more practical to use symmetric cryptography for data encryption.
- Compatibility: Many encryption protocols, such as TLS/SSL, use both symmetric and asymmetric cryptography. However, some systems may be more compatible with symmetric cryptography than asymmetric cryptography, which could make it more practical to use symmetric cryptography in a KYC blockchain system.

Overall, while asymmetric cryptography can provide better security in certain situations, the efficiency, simplicity, and compatibility of symmetric cryptography may make it more appropriate for a KYC blockchain system.

VIII. EVALUATION AND RESULTS

When it comes to secure communication of sensitive data, choosing the right encryption algorithm is crucial. AES, Data Encryption Standard (DES), and Triple-DES are widely used symmetric cryptographic algorithms that provide encryption and decryption of sensitive data. In order to determine which algorithm is most suitable for a particular use case, it is important to compare their performance characteristics.

This can involve analyzing factors such as security, speed, key length, and compatibility, as well as conducting experiments to determine the encryption and decryption time of each algorithm. By comparing AES, DES, and 3DES, one can make an informed decision about which algorithm to use based on the specific security requirements, performance characteristics, and compatibility of their system or use case.

A. Experiment Setup

"Tech Company Fundings" is the dataset used for comparing AES, DES, and Triple-DES in terms of their execution time (both encryption and decryption time). This dataset contains up-to-date information about tech company funding across the globe. The dataset contains information from January 2020 and contains 3575 company funding information. The data attributes include –

1) Company name: The name of the technology company that received funding.
2) Website: The website address of the technology company.
3) Region: The region or city in which the technology company is based.
4) Vertical: The industry or category of the technology company.
5) Funding stage: The type of funding round, such as seed, Series A, Series B, etc.
6) Funding date: The date on which the funding round was announced or completed.
7) Funding amount (USD): The amount of funding raised by the company in US dollars.

From the attributes, it is clear that the dataset contains different datatypes such as string, integer, url, and date. All of these can be used to showcase how AES, DES, and Triple DES would encrypt and decrypt that data. It is also important to note that there is no dataset containing KYC details as it contains sensitive information about customers and thereby needs to be confidential at all cost. This is why we have chosen this dataset to validate the cryptography technique used in the KYC system i.e, AES.

B. Testing Framework

1) Collect or obtain a suitable dataset with a sufficient number of rows. (such as Tech Fundings)
2) Choose a programming language and a cryptographic library that supports all AES, DES and Triple-DES algorithms (e.g., Python and PyCrypto, Node.js and CryptoJS).
3) Write a script to read the dataset row by row and concatenate all values into a single string.
4) Set up an encryption key for each algorithm (i.e., a 16-byte key for AES, a 56-bit key for DES, and a 168-bit key for Triple DES).
5) For each row, measure the time taken to encrypt the concatenated string using each algorithm and their respective keys.
6) Measure the time taken to decrypt the ciphertext generated in the previous step using each algorithm and their respective keys.
7) Write the results (i.e., the encryption and decryption times) to an output file, such as a CSV file.
8) After processing all rows in the dataset, calculate the total execution time for the entire script.
9) Analyze the output file to compare the execution times of the different algorithms.
10) Draw conclusions about the relative efficiency of each algorithm based on the execution time measurements.

C. Script Testing

There are two different types of scripts written in order to analyze the memory usage of AES, DES, and Triple-DES algorithms in terms of their execution time. However, the execution steps remain the same as shown below:

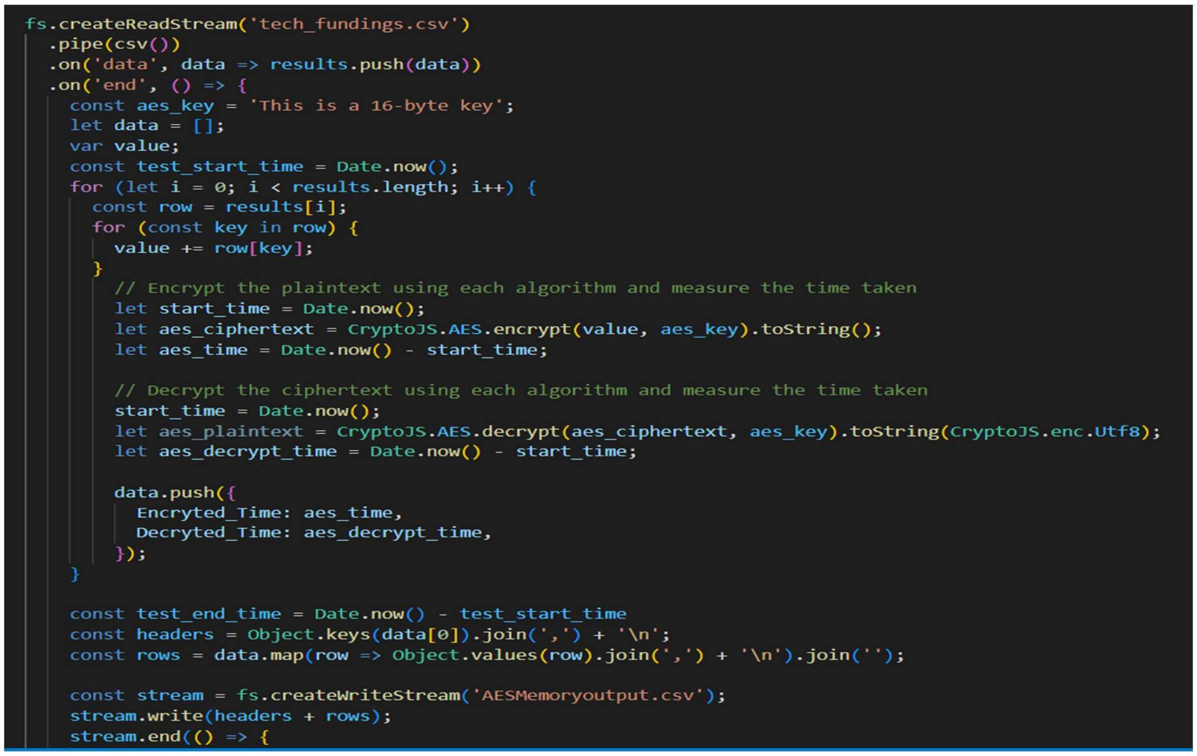
1) Import necessary libraries - The script begins by importing the necessary libraries, including fs, csv-parser, and CryptoJS.
2) Initialize variables - The script initializes several variables, including the AES key used for encryption, the headers for the output CSV file, and a stream object used to write data to the output file.
3) Read input CSV file - The script starts reading data from the input CSV file using the csv() function, which returns a readable stream that emits each row of data as a JavaScript object.
4) Concatenate row values - For each row of data, the script concatenates all of the values into a single string, which will be encrypted using AES.
5) Encrypt data - The concatenated string is encrypted using AES using the CryptoJS.AES.encrypt() function, and the time taken to encrypt the data is measured using the Date.now() function.
6) Decrypt data - The encrypted data is then decrypted using AES using the CryptoJS.AES.decrypt() function, and the time taken to decrypt the data is measured in a similar way.
7) Convert decrypted plaintext - The decrypted plaintext is converted from a WordArray object to a UTF-8 string using the toString() function.
8) Write output to CSV - The script writes the encrypted ciphertext, along with the time taken to encrypt and decrypt the data and the resulting plaintext, to the output CSV file using the stream.write() function.
9) Log output to console - Finally, when all of the data has been processed, the script logs a message to the console indicating that the output data has been saved to the output CSV file and the total time taken for the encryption and decryption process.

a) With Memory Usage

Figure 12 shows a script that stores all data in an array and writes it to the output file at the end of the script, which could consume more memory.

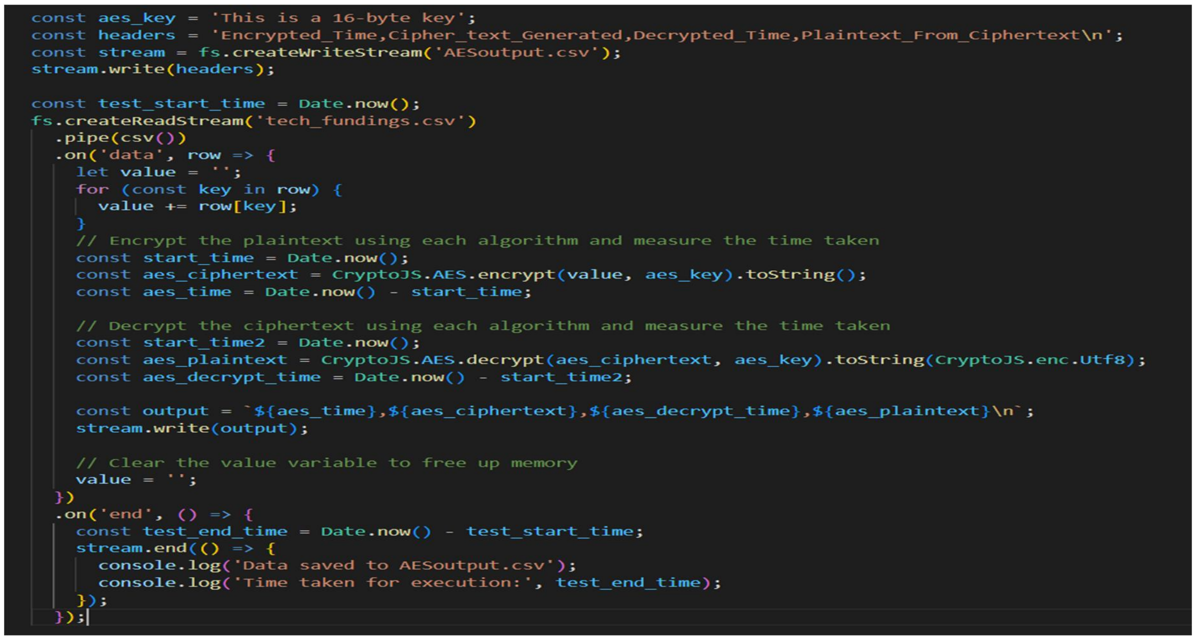
b) Without Memory Usage

Figure 13 shows a script that writes data to the output file as soon as it is encrypted and decrypted, which could consume less memory.



```
fs.createReadStream('tech_fundings.csv')
  .pipe(csv())
  .on('data', data => results.push(data))
  .on('end', () => {
    const aes_key = 'This is a 16-byte key';
    let data = [];
    var value;
    const test_start_time = Date.now();
    for (let i = 0; i < results.length; i++) {
      const row = results[i];
      for (const key in row) {
        value += row[key];
      }
        // Encrypt the plaintext using each algorithm and measure the time taken
        let start_time = Date.now();
        let aes_ciphertext = CryptoJS.AES.encrypt(value, aes_key).toString();
        let aes_time = Date.now() - start_time;

        // Decrypt the ciphertext using each algorithm and measure the time taken
        start_time = Date.now();
        let aes_plaintext = CryptoJS.AES.decrypt(aes_ciphertext, aes_key).toString(CryptoJS.enc.Utf8);
        let aes_decrypt_time = Date.now() - start_time;

        data.push({
          Encryted_Time: aes_time,
          Decryted_Time: aes_decrypt_time,
        });
    }

    const test_end_time = Date.now() - test_start_time
    const headers = Object.keys(data[0]).join(',') + '\n';
    const rows = data.map(row => Object.values(row).join(',') + '\n').join('');

    const stream = fs.createWriteStream('AESMemoryoutput.csv');
    stream.write(headers + rows);
    stream.end(() => {
```

Figure 12: Script consuming more memory



```
const aes_key = 'This is a 16-byte key';
const headers = 'Encrypted_Time,Cipher_text_Generated,Decrypted_Time,Plaintext_From_Ciphertext\n';
const stream = fs.createWriteStream('AESoutput.csv');
stream.write(headers);

const test_start_time = Date.now();
fs.createReadStream('tech_fundings.csv')
  .pipe(csv())
  .on('data', row => {
    let value = '';
    for (const key in row) {
      value += row[key];
    }
    // Encrypt the plaintext using each algorithm and measure the time taken
    const start_time = Date.now();
    const aes_ciphertext = CryptoJS.AES.encrypt(value, aes_key).toString();
    const aes_time = Date.now() - start_time;

    // Decrypt the ciphertext using each algorithm and measure the time taken
    const start_time2 = Date.now();
    const aes_plaintext = CryptoJS.AES.decrypt(aes_ciphertext, aes_key).toString(CryptoJS.enc.Utf8);
    const aes_decrypt_time = Date.now() - start_time2;

    const output = `${aes_time},${aes_ciphertext},${aes_decrypt_time},${aes_plaintext}\n`;
    stream.write(output);

    // Clear the value variable to free up memory
    value = '';
  })
  .on('end', () => {
    const test_end_time = Date.now() - test_start_time;
    stream.end(() => {
      console.log('Data saved to AESoutput.csv');
      console.log('Time taken for execution:', test_end_time);
    });
  });
```

Figure 13: Script consuming less memory

IX. RESULTS AND ANALYSIS

Based on the output CSV files generated from the testing phase, analyzing the encryption, decryption and overall execution time of AES, DES, and Triple-DES algorithms, over the 3575 rows of data is quite easy. The findings are presented in the form of graphs and tables so that it's visually appealing and quickly understood. Note that testing is done from two different perspectives as shown below:

Table 2: Execution Time over the entire dataset when more memory is used.

	AES	DES	Triple-DES
Execution Time (in ms)	182052	798487	4168975
Execution Time (in min)	3.03	13.03	69.48

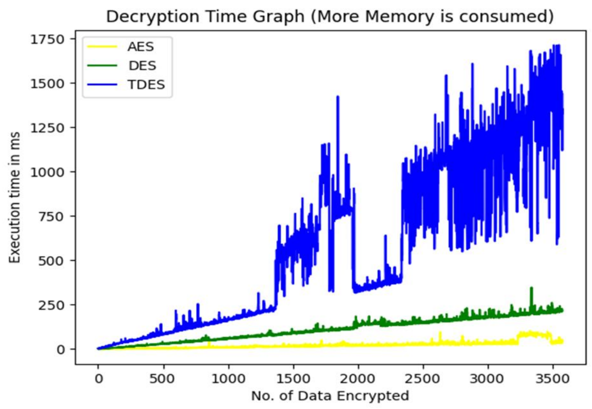


Figure 14: Encryption Graph of AES, DES and Triple-DES during less memory consumption

Table 3: Execution Time over the entire dataset when less memory is used.

	AES	DES	Triple-DES
Execution Time (in ms)	1471	1926	3609
Execution Time (in min)	0.02451	0.0321	0.06015

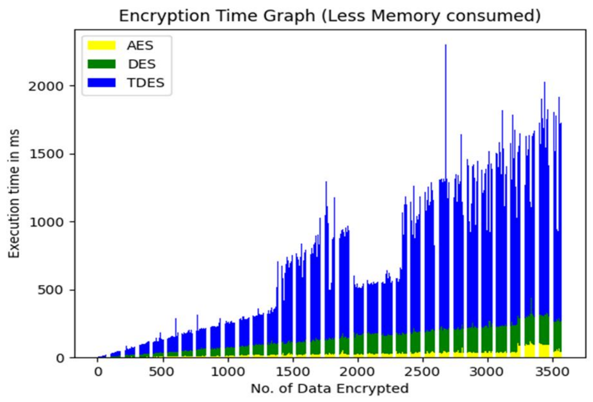


Figure 15: Encryption Graph of AES, DES, and Triple-DES during more memory consumption

A. *More Memory Consumption*

The values present in Table 2 are the total time taken for encrypting and decrypting the 3575 rows of data simultaneously. Based on this, it is clear that AES was executed in the shortest time possible. Figure 14 shows a line graph, where the color yellow indicates AES, green indicates DES, and blue indicates Triple-DES. X-axis represents the number of rows of data encrypted against the time taken to encrypt data in milliseconds (ms) on the Y-axis. From the graph, it is clear that AES takes the least amount of time to execute, then comes DES, and finally Triple-DES. The same result can also be seen in the case of decryption when a similar testing process is done.

B. *Less Memory Consumption*

The values present in Table 3 are the total time taken for encrypting and decrypting the 3575 rows of data simultaneously. Based on this, it is clear that AES was executed in the shortest time possible. Figure 15 shows a stacked bar graph, where the color yellow indicates AES, green indicates DES, and blue indicates Triple-DES. X-axis represents the number of rows of data encrypted against the time taken to encrypt data in milliseconds (ms) on the Y-axis. From the graph, it is clear that AES takes the least amount of time to execute, then comes DES, and finally Triple-DES even in the case of less memory consumption. This case is true even in the case of decryption of data. In conclusion, AES outperforms DES, and Triple-DES in terms of execution speed, security and memory consumption comparatively. It is also noted that DES performs good than Triple-DES but not that well against AES. Choosing AES as cryptography for the KYC Blockchain as a security measure is ideal as per the results obtained.

X. CONCLUSION

The proposed system is an enhanced and dynamic KYC system, built on blockchain technology, which effectively diminishes the expenses associated with conventional KYC procedures. Moreover, it facilitates the proportional distribution of these expenses among stakeholders.

Through the implementation of smart contract, users can securely store their personal information on the blockchain via banks, and grant or revoke access to them and making the system truly decentralized, and that it makes possible a distributed data storage architecture.

The use of blockchain technology with AES cryptography in a KYC system provides a secure and tamper-proof way of storing personal information.

The system allows for the easy addition, modification, and viewing of KYC data by authorized parties. The use of customer ratings and bank ratings also provides an additional layer of trustworthiness and credibility to the system. The BANK_RATING feature is a unique aspect of the system, which incentivizes banks to conduct accurate and reliable KYC processes. It encourages banks to comply with regulatory requirements and mitigate risks associated with fraudulent activities, which in turn improves their credibility and trustworthiness.

Overall, a KYC blockchain system that uses AES as the cryptography technique can provide a secure and efficient way to manage customer information and improve the regulatory compliance and risk management practices of financial institutions.

REFERENCES

[1] “Investigation of Applying Machine Learning for Watchlist-Filtering in Anti-Money Laundering” by Mohannad Alkhalili Mahmoud H. Qutqut and Fadi Almasalha 2021 IEEE
[2] “Anti-Money Laundering Detection using Naïve Bayes Classifier” by Ashwini Kumar, Sanjoy Das, andVishu Tyagi 2020 IEEE
[3] “A Fast Fully Octave Convolutional Neural Network for Document Image Segmentation” by Ricardo Batista das Neves Junior, Luiz Felipe Vercosa, David Macedo , Byron Leite Dantas Bezerra 2020 IEEE
[4] “Money laundering and terrorism financing detection using neural networks and an abnormality indicator” by Jose-de-Jesus Rocha-Salazar, Maria-Jesus Segovia-Vergas and Maria-del-Mar Camacho-Minano 2020 Elsevier
[5] “Anti-Money Laundering: Using data visualization to identify suspicious activity” by Kishore Singh and Peter Best 2019 Elsevier
[6] “Combating money laundering with machine learning – applicability of supervised-learning algorithms at cryptocurrency exchanges” by Eric Pettersson Ruiz and Jannis Angelis, 2019, JMC
[7] “Survey of Machine Learning Approaches of Anti-money Laundering Techniques to Counter Terrorism Finance” by Prof. Dr. Nevine Makram Labib, Prof. Dr. Muhammed Abu Rizka, and Amr Ehab Muhammad Shokry, 2020 IEEE
[8] “Assessing the efficiency of the anti-money laundering regulation: an incentive-based approach” by RicardoAzevedo Araujo, 2008, JMC
[9] “Transaction Authorization from Know Your Customer (KYC) Information in Online Banking” by Prakash Chandra Mondal, Rupam Deb, and Mohammad Nurul Huda, 2016 IEEE Conference
[10] “Competence of Graph Convolutional Networks for Anti-Money Laundering in Bitcoin Blockchain” by Ismail Alarab, Simant Prakoonwit, Mohamed Ikbal Nacer

[11] "Counter Terrorism Finance by Detecting Money Laundering Hidden Networks Using Unsupervised Machine Learning Algorithm" by Amr Ehab Muhammed Shokry, Mohammed Abo Rizka, and Nevine MakramLabib, 2020 ICT
[12] "Fighting Money Laundering with Statistics and Machine Learning" by Rasmus Ingemann Tuffveson Jensen and Alexandros Iosifidis, 2022 IEEE
[13] "Machine Learning Methods to Detect Money Laundering in the Bitcoin Blockchain in the Presence of Label Scarcity" by Joana Lorenz, Maria Ines Silva, David Aparicio, Joao Tiago Ascensao, and Pedro Bizarro, 2020,
[14] "From Bitcoin to Cybersecurity: A Comparative Study of Blockchain Application and Security Issues" byFangfang Dai, Yue Shi, Nan Meng, Liang Wei, and Zhiguo Ye 2017, IEEE
[15] "Blockchain Applications – Usage in Different Domains " by Joe Abou Jaoude And Raafat George Saade2019, IEEE