Decentralized KYC System

Prince Sinha[1], Ayush Kaul[2]

[1,2]*B.E Computer Engineering University of Mumbai, Mumbai, India*

Abstract - *KYC (Know Your Customer) is used for identity verification in banks, Hospitals, Aadhar, Passport verification, etc. A third party independent KYC system that can be used at multiple places to validate the identity of the individual is the need of the hour. The proposed KYC system is a blockchain based decentralized system. The proposed system can be used to establish proof of identity for an individual. The immutability feature of the distributed ledger is the key concept to ensure that the data stored in the system is tamper proof. The proposed system is similar in functionality to the legacy KYC system. Data is stored in a distributed database to ensure data replication, data backup and no single point of failure. The fully decentralized architecture of the proposed system ensures that there is no dependency on centralized client-server architecture. There is no third party involvement to establish trust between the stakeholders. The proposed system is cost efficient in terms of gas used to write a transaction on the blockchain. The data stored on the decentralized database is encrypted to provide an additional layer of security. Even if there is a data stored in the decentralized database is compromised there is still no harm as the data is encrypted.*

Key Words: Blockchain, Cryptographic hash, KYC system, Ethereum, IPFS, Centralized system.

1. INTRODUCTION

Traditional client-server architecture [1] has been in use from around 1960's. First applications of the client-server architecture were OS/360, ARPANET, Xerox PARC, etc [2]. The client-server architecture is based on HTTP request and response between client and server. The client sends an HTTP request to the server (service provider) for its services. The client-server architecture [3] is a centralized architecture due to a central service provider known as server. The Internet is supported by the same architecture. Most of the web applications are centralized in nature. Some drawbacks of centralized architecture have been identified. The major downside of the centralized architecture is the single point of failure [4] of the server. Even with backup servers the system still faces a considerable downtime in case of failure. Centralized architecture is under the control of the administrator having access to confidential data.

Centralized architecture generally uses some third party to establish trust between the stakeholders. Hence the dependence on the third party is established to create an environment of trust.

With research and innovation in the field of technology decentralized architecture has gained momentum and acceptance. Blockchain [5] is a decentralized technology which is the backbone behind cryptocurrencies like Bitcoin [6], Ether, Litecoin, etc. The blockchain is based upon peer-to-peer architecture. Each peer node is responsible for maintaining the distributed ledger. The immutable distributed ledger of blockchain establishes trust between peer nodes in a blockchain based system. The proposed architecture is based on Ethereum blockchain [7]. An ethereum blockchain uses Proof of Work [8] consensus algorithm for committing a transaction to the blockchain. Mining is required in order to validate a block of transactions. Miner nodes are not predefined in the Proof of Work algorithm. All other nodes validate the mining work by computing the hash of the block. Once the majority of nodes validate the block it is committed on the distributed ledger. The distributed ledger is synchronized at every peer node in the chain and have a separate address. network. Ethereum platform supports smart contract [9] for carrying out transactions on the network. The smart contract is used for asset transfer in the Ethereum network. smart contracts are deployed on the chain with a separate address. It defines a set of rules according to which a transaction proceeds.

The proposed system is Ethereum based decentralized solution. The proposed system is cost-efficient due to off-chain data storage. This reduces the amount of gas used for deployment of the smart contract and transaction cost is also reduced in terms of gas used. Only hash and username is stored as on-chain data on the blockchain. In most of the legacy KYC [10] systems, there is an option to upload documents such as ID proof, Passport, Address proof, etc. The proposed system has all the functionalities of a legacy KYC system including image data is stored in the decentralized database.

2. BACKGROUND

Know your customer (KYC) is used for customer management and identity verification. This document is submitted by the customer to an organization in order to create trust between the two parties. Initially, there was no way to verify the identity of the customers as a result KYC [11] was proposed in the United States in 1990. At that time the purpose of KYC was to stop terrorist financing and money laundering through banks. The main stakeholder of KYC is bank. Banks ask their customers to fill KYC document [12] so that they can verify their identity. Bank crosschecks the information submitted by clients to stop money laundering, terrorist financing, and financial frauds. So, at present banks don't allow any account holder without KYC documentation. KYC document contains customer information, ID proof, address proof, and photograph. Initially, the pen-paper approach was used for submitting KYC document but the problem with maintenance of records was prominent. The tasks became hectic for the bank to check the identity every time through paper filled by the customers. The chances of the document being misplaced were more in such case. So, digital KYC system was proposed, which is called eKYC. In that approach, the customer fills the KYC document through the web application of the organization. Data submitted were stored in centralized databases [13]. At any point in time, the organization can access the customer information through customer id. This system was paperless so overall cost got reduced but since, data is stored in the centralized database so, loopholes of the centralized system like the single point of failure, data redundancy and third-party involvement in verification still exists. Also, data stored in the centralized server can be compromised/attacked by the hackers so, chances of the leak of customer confidential data is more in the existing centralized system architecture.

3. PROPOSED ARCHITECTURE

The system proposed in this paper is a blockchain based decentralized KYC system which uses a decentralized database (IPFS) [14] to store user information as well as verification documents. Ethereum is a blockchain platform which uses a smart contract for processing each and every transaction. The proposed system is capable to perform all the functionality of standard legacy KYC system. At the time of filling the KYC, document user has to provide his/her details including ID proofs and photograph as illustrated in step 1 of fig. 1. The user has to also provide a username so that using that username user can check his/her submitted document in future or can update the document. The system will use the username to generate a public-private key pair along with the wallet address. A new user has to download the key file and need to safely store that file. The key will be used during signing the transaction and it will be unique for every user. The user-supplied data is then converted into JSON object which is then stringified so that it can be stored in IPFS as illustrated in step 2 of fig.1. The form of JSON object will be:-

Object= {‘username’:’###’,’name’:’####’,’image’:’###’}

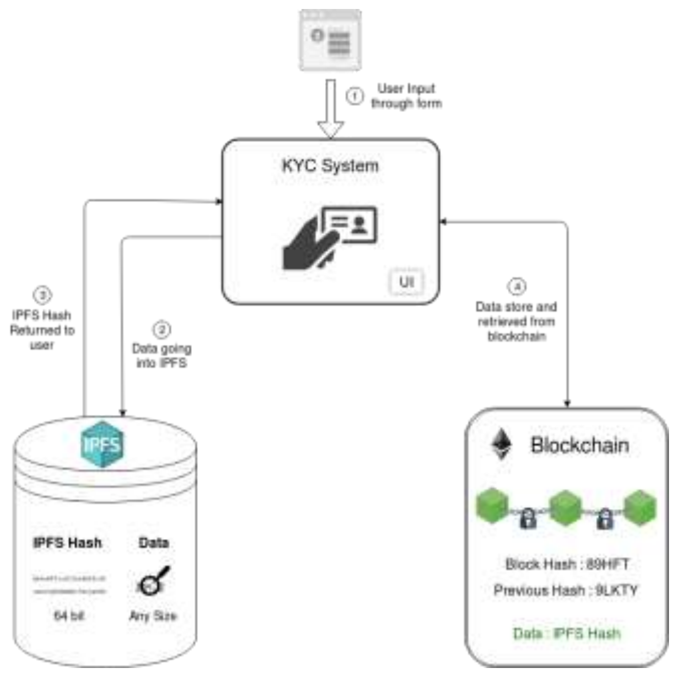

Fig -1: Architecture diagram

Any type of information can be stored in the IPFS. It works like the general database but it has many added feature due to its decentralized nature. For an additional layer of security of user data, the data can be encrypted using an encryption algorithm [15] like DES. After successful storage of data, IPFS returns a hash of 46 characters that can be used for retrieval of data in future as illustrated in step 3 of fig. 1. The IPFS gives a multihash and starts with “Qm”. The letters “Qm” corresponds to the hashing algorithm (SHA-256) used by IPFS. The size of the hash used in IPFS is 32 bytes long. In step 4 of fig. 1, the IPFS hash along with username will be stored in ethereum blockchain by triggering the smart contract which will make data immutable and tamper resistant. The size of the wallet address is 42 character which is difficult to remember. So proposed system will map username with wallet address using mapping function of solidity. This will be used to link username with associated wallet address. At the time of data retrieval, the user has to only provide the username which will be also unique for every user to access his/her detail. From username, the wallet address can be accessed and through that one can get the stored data. The user has to keep his/her private key in any input/output device such as a pen drive for the safety purpose.

3.1 Key Generation Algorithm

In this system, Keythereum [16] an open source javascript tool is used for key generation purpose. Fig. 2 illustrates the steps in key pair and wallet address generation.

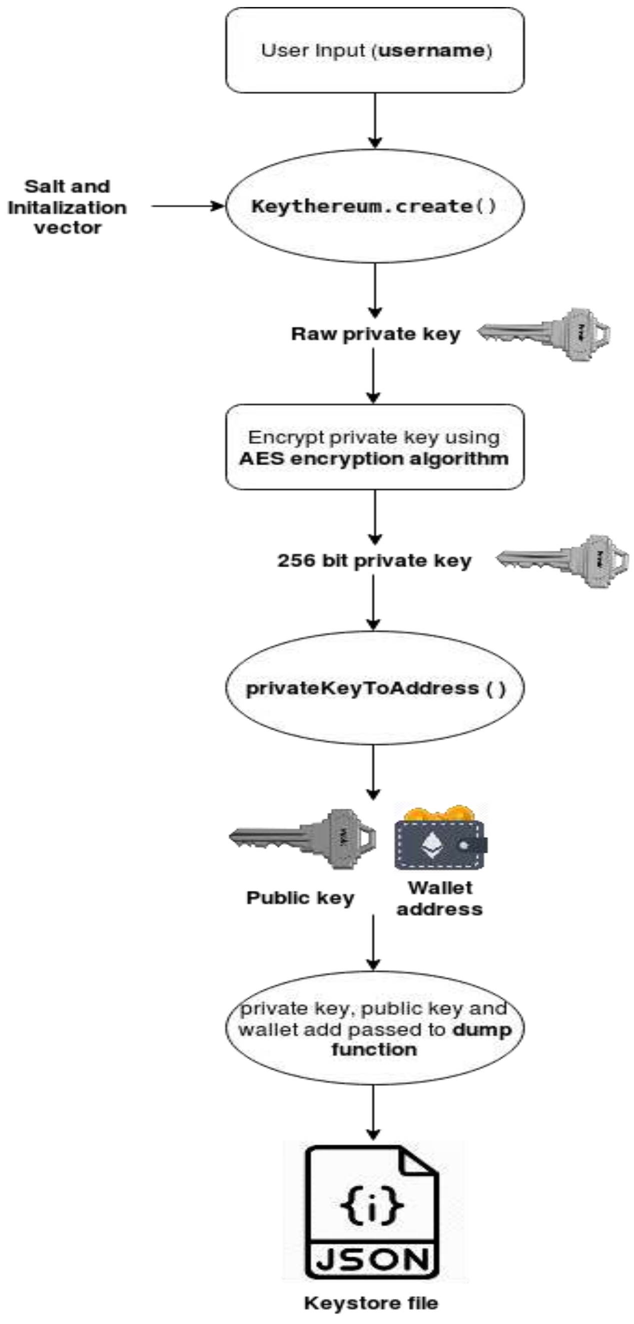

Fig -2: Key generation flowchart

Following are the steps for key generation :-

1. Generating a 256 bit or 32-byte random private key and a salt of length 256 bit or 32 bytes used by the KDF(key derivation function), and the initialization vector of length 128 bit or 16 bytes used to AES-128-CTR encrypt the key. Private key, initialization vector and salt can also be generated by using keythereum.create function.

2. The proposed system uses the username as a passphrase. Which is passed to PBKDF2-SHA256 (Password-Based Key Derivation Function 2 - Secure Hash Algorithm 256) will be used to derive the AES secret key.

3. Then using iv (initialization vector), salt the system will generate a 256-bit private key. And encrypt it using AES secret key generated in step 2.

4. Using privateKeyToAddress(), public key and wallet address are generated from the 256-bit private key.

5. Then by using the dump function the raw private key and wallet address is converted into a JSON object. Using export To File function the JSON object is converted to a text file with a timestamp in the filename, which can be downloaded for safe keeping.

3.2 Sample Contracts

Compose mail contract with IPFS:-

```
pragma solidity ^0.4.21;

contract Kyc_With_IPFS

{

        bytes32 username;

        string ipfs;

        mapping(bytes32 => address) public get_name;

        mapping(bytes32 => bool) public namesInUse;

        mapping(address => bool) public addInUse;

function submit_detail(bytes32 username,string _ipfs)

        {

        get_name[username] = msg.sender;

        ipfs = _ipfs;

        namesInUse[username] = true;

        addInUse[msg.sender] = true;

        }

}
```

This contract is of submitting KYC with IPFS written in pragma solidity compiler version 0.4.21. The name of contract is Kyc_With_IPFS and it has only one function store_detail which will store ipfs_hash with username in the blockchain.

Compose mail contract without IPFS:-

```
pragma solidity 0.4.21;

contract Kyc_Without_IPFS

{

        bytes32 username;

        string name;

        string father_name;

        string mother_name;

        string addr;

        string city;

        string dob;

        string mobile_no;

        string addhar_no;

        string pan_no;

        string photograph;

        mapping(bytes32 => address) public get_name;

        mapping(bytes32 => bool) public namesInUse;

        mapping(address => bool) public addInUse;


function send_detail(bytes32 _username, string _name, string _father_name, string _mother_name, string _addr, string _city, string _dob, string _mobile_no, string _addhar_no, string _pan_no, string _photograph) public

        {

        username = _username;

        get_name[username] = msg.sender;

        name = _name;

        father_name = _father_name;

        mother_name = _mother_name;

        addr = _addr;

        city = _city;

        dob = _dob;

        mobile_no = _mobile_no;

        addhar_no = _addhar_no;

        pan_no = _pan_no;

        photograph = _photograph;

        namesInUse[username] = true;

        addInUse[msg.sender] = true;

        }

}
```

His contract is also a contract for KYC but, it doesn't use IPFS to store data. All the data are directly stored in blockchain.

4. RESULTS AND DISCUSSIONS

The system that is proposed in the paper is the KYC system which uses blockchain and IPFS a decentralized database to store user data.

There are many benefits of using the proposed system over existing system :-

- The proposed system will make the system fault tolerant, attack resistant and immutable [17].
- The user can get his/her detail using username so he/she doesn't need to remember a long wallet address.
- The proposed system will remove third-party intervention of the existing system.
- The system is cost efficient in terms of gas used for each transaction.
- Issues of the centralized system like the single point of failure got solved in the proposed system.
-

Table -1: Comparison of gas used for creating and executing the contract with IPFS and without IPFS

	Contract with IPFS	Contract without IPFS	Gas saved (gas)
Contract creation cost	339433 gas	569684 gas	230251
Transaction cost	147617 gas	424218 gas	276601

From table 1, the transaction cost incurred for the creation of KYC contract with IPFS comes to 339433 gas while without IPFS it comes to 569684. So, by using IPFS, system efficiency has been increased by 67.83 % for contract creation.

Efficiency percentage for contract creation $=(\frac{569684 - 339433}{339433}) \times 100 = 67.83\ \%$

As contract creation is a the one-time process so, it will not affect too much on system performance but for each transaction means executing store_detail function the transaction cost with IPFS is only 147617 gas while without IPFS it comes to 424218 that is too much. So, by using IPFS, System efficiency is increased by 183.77%.

Efficiency percentage for each Transaction $=(\frac{424218 - 147617}{147617}) \times 100 = 183.377\ \%$

The difference comes because all the data are directly stored in the blockchain in case of a system without IPFS. While with IPFS only IPFS hash along with username is stored on the ledger.

5. CONCLUSION

Blockchain technology is the current boom in the field of research and IT. The Proposed system is an appropriate replacement for legacy KYC system. The proposed system provides all the necessary features of a legacy KYC system. IPFS file system is a cost-effective and decentralized database for storing data which is used throughout the system. The proposed system is cost-efficient as compared to other decentralized KYC architectures. The decentralized architecture also benefits the users in terms of security, usability, and trust. Third party dependence is also eliminated in the decentralized architecture.

REFERENCES

1. Berson, Alex, IEEE-802 Title “Client - server architecture”, McGraw-Hill, 1992.

2. “Client-server Model”, Retrieved from https://en.wikipedia.org/wiki/Client%E2%80%93server_model

3. Haroon Shakirat Oluwatosin, “Client-Server Model”, IOSR JCE p-ISSN: 2278 8727 volume 16, February 2016.

4. Sourabh Goyal,”Centralized vs Decentralized vs Distributed”, July 2015, Reterived from https://medium.com/delta-exchange/centralized-vs-decentralized-vs-distributed-41d92d463868

5. Zibin ZHeng, Shaoan Xie, Hongning Dai, Xiangping Chen, and Huaimin Wang, “An overview of Blockchain Technology : Architecture, Consensus, and Future Trends”, 2017 IEEE 6th International Congress on Big Data, October 2017.

6. Satoshi Nakamoto, “Bitcoin: A Peer-to-Peer Electronic Cash System”, 2008.

7. James Ray, welcome to the Ethereum Wiki, February 2014, Github repository, https://github.com/ethereum/wiki/wiki

8. Georgios Konstantopoulos, Understanding Blockchain Fundamentals, Part 2: Proof of Work & Proof of Stake , Retrived from https://medium.com/loom-network/understanding-blockchain-fundamentals-part-2-proof-of-work-proof-of-stake-b6ae907c7edb

9. Solidity, December 2013, Github repository, https://github.com/ethereum/solidity/blob/v0.4.24/docs/index.rst

10. Njagi Lucy Wawira, Effectiveness of know your customer (KYC) policies adopted by commercial banks in Kenya in reducing money laundering and fraud incidence, October 2009, Reterived from http://erepository.uonbi.ac.ke/bitstream/handle/11295/96611/Njagi%20Lucy%20W_Effectiveness%20of%20Know%20Your%20Customer%20(Kyc)%20Policies%20Adopted%20by%20Commercial%20Banks%20in%20Kenya%20in%20Reducing%20Money%20Laundering%20and%20Fraud%20Incidences.pdf?sequence=1

11. Ketul Shah, “All you wanted to know about KYC”, September 2006, Retrieved from http://www.rediff.com/money/2006/sep/12guest.htm

12. Prof. Venkatesh U. Rajput, “Research on Know Your Customer (KYC)”, ISSN 2250-3153, July 2013.

13. Nihaal Mehta, Sudarshan Shinde, Nishi Tiku, “Centralized Database for Android and Web Application”, IJIRSET ISSN: 2319-8753, November 2015.

14. Juan Benet,, IPFS is the Distributed Web, December 2013 , Github repository, https://github.com/ipfs/ipfs

15. Dr. Prerna Mahajan, Abhishek Sachdeva, “A Study of Encryption Algorithm AES, DES and RSA for Security, Global Journals INC. ISSN: 0975-4172, 2013.

16. Jack Peterson, keythereum, August 2015, Github repository, https://github.com/ethereumjs/keythereum/

17. Vitalik Buterin, “The meaning of decentralization”, February 2017, Reterived from https://medium.com/@VitalikButerin/the-meaning-of-decentralization-a0c92b76a274