# Decentralized Application for Protect Message using Block chain

[1]V.Sureshkrishnan, D.Raghu Raman[2]
[1, 2] *Department of computer science and engineering,*
*IFET college of engineering,*
*Villupuram, Tamilnadu, India.*

**ABSTRACT**

*Block chain technology stores data blocks that are captured together using hashes. Block chain is a way of storing digital data. This app can protect messages from internal and external cyber attack. Reduce cyber attack so peer-to-peer networks use this app. Flash framework can be used in this application.*

***Keywords:-****Block chain, peer-to-peer, Flash framework, Decentralized, Networks.*

## 1. INTRODUCTION

Nowadays, most hackers can be hack third party data based on money. Today, cyber bat is on the rise worldwide, and communication and messaging between two people in the form of a call or message is not secure. They use secure block chain technology through messaging. The app designed may fall under the block chain they use from the peer-to-peer network they protect from messaging. They can network through communication, reduce cyber attacks on messages. Using SHA-512 cryptography and decrypt technology.

## 2. BACKROUND AND DEFINITIONS

### A. Secure Messaging

The decentralized app can enter the message they dynamically generate IP address the can be paired they through network used message can share between each other they enter message they can use SHA 512 cryptography and decrypt technique.

### B. Flask Python web app framework

Flask may be a net application framework written in Python. it's developed by Arminius Ronacher, who crystal rectifier a world cluster of Python enthusiasts referred to as Pamino. supported Flask Workzig WSGI Toolkit and Ginza a pair of example engine.

### C. Block chain

Block chain may be a thanks to store digital knowledge. knowledge may be virtually something. For bit coin, this can be a dealing (transfer of bitcoins from one account to another), however it may also be files; knowledge is hold on in blocks, however they're joined along (or captured) mistreatment cryptological hashes - thence the name "block chain".

There is all the magic of storing this data and adding it to the blockchain. Block chain is essentially a linked list, which contains ordered data, such as some barriers:

- Once added, the block cannot be modified; In other words, it is only a supplement
- It has specific rules for adding data
- Architect whose structure is distributed

### D. Werkzeug

It is a WSGI toolkit that implements requests, response objects, and other utility functions. This allows you to build a web framework on it. The Flask framework uses Werkejug as one of its bases.

### E. Jinja2

Ginz 2 is a popular template engine for Python. Web turbulence in the form of micro-frameworks. It aims to keep the core of the application still extensible. Flask does not have a built-in abstract layer template system, which combines a template with a static data source to provide dynamic web pages.

Flask is often reefed for database management or it does not have validation support. Instead, it supports flask extensions to add such functionality to applications

**F. CSS Frameworks**

If you are a developer who, like me, wants to create good web pages, but doesn't have the time or interest to learn low-level mechanisms to make it more efficient by writing raw HTML and CSS, then the only practical solution is to use the CSS framework to simplify the task. By taking this route you may lose some creative freedom, but on the other hand, your web pages will look good in all browsers with no effort. CSS Framework provides a collection of high-level CSS classes with pre-built styles for common types of user interface elements. Most of these settings provide JavaScript add-ons for things that can't be done with HTML and CSS

**G. Introducing Bootstrap**

Twitter is one of the most popular CSS frameworks created by Bootstrap. You can design with these outlines if you can see with these pages.

The most direct way to use Bootstrap is to import only the Bootstrap. min.sis file into your base template. You can download a copy of this file and add it to your project or import it directly from the CDN.

## 3. EXISTING SYSTEM

Ethereum is an open-source block chain platform with distributed computing that enables developers to execute smart contracts. These are code collections in the Ethereum block chain and run without censorship, fraud, third party interference or useless time. Data integrity is ensured by using the Merkel Patricia tree, which guarantees cryptographic authentication

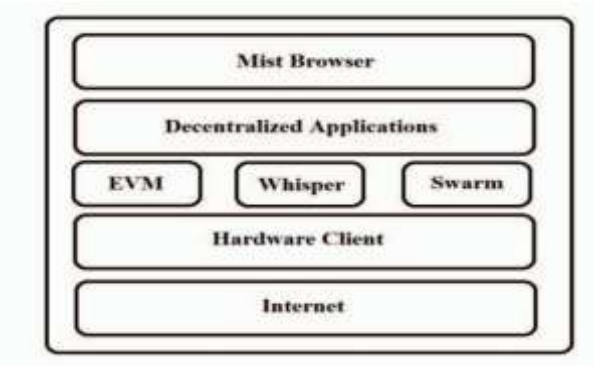


Fig. 1. Ethereum technology stack

*1)* Fog Browser: Interface for using different applications.

*2)* Decentralized application

*3)* Ethereum Virtual Machine (EVM): EVM is an abstraction layer that sits on a hardware client and performs internal calculations and status. All full nodes perform the same code in EVM. EVM is essentially a computer that can run code and have enough availability and fault tolerance data until it is sufficient.

*4)* Whisper: This is Ethereum's P2P communication protocol for decentralized applications. P2P communication between nodes in the Whisper network uses the pVp2p wire protocol. A DAP instance can create an identity on a node associated with Whisper. This identity is required to send or receive messages. Once the message is sent, it must be redirected by each whisper node, in principle. This requires implementing the PoW algorithm to prevent denial-of-service (DoS) attacks. If their POD is found to exceed the predefined limit, the messages will be processed and redirected further. Additionally, Whisper DAP allows developers to configure the security and security of anonymity.

*5)* All messages in Whisper are initially encrypted and sent via a local wire-protocol called PVP2P that supports various sub-protocols such as Whisper. The message is encrypted further by the PVP2P protocol. Currently, the asymmetric form using the Ellipse Curve Integrated Encryption Scheme (ECIS) can be randomly encrypted with all Whisper messages using the Accent-256K1 public key or Symmetrically Advanced Encoding Standard GALIOS / Counter Mode (AES-GCM). This should be encrypted. 96-bit non. Multiple asymmetric and symmetric keys can be owned by a single node. If the message is successfully

decrypted, it is sent to the relevant DAP. Using the Web3-shh package, you can communicate with the Whisper protocol. The following methods are used in the Web 3-Shah package:

- web3.shh.newKeyPair (): This method creates a new private and public key pair that is used for message encryption and encryption.
- web3.shh.newSymKey (): This method randomly generates a symmetric key and stores the key under ID. This key is shared between the communication parties and is used for encryption and decryption.
- web3.shh.getPublicKey (kId): This method returns an associated public key for a given key pair.
- web3.shh.getSymKey (id): This method returns a symmetric key for a given ID.
- web3.shh.post (Object [, callback]): This method is called when a whisper message needs to be posted to the network..

## 4. DEVELOPMENT

### 4.1 Store transactions in blocks

- I store data in JSON which is a widely used format
- The general term "data" is often used in the interchange with the word "transaction".
- Transaction in blocks are packaged in the app
- A block may contain one or several transition.
- Transaction blocks are often generated and added to the To block chain.
- Each block has its own unique ID, as there may be multiple blocks.

### 4.2 Make blocks invarient

- I would like to find out if there is any tampering in the data stored inside the Inside block
- In block chain technology, this is achieved using the hash function
- It is a function that takes data of any size and generates a certain amount of data from it, which is usually used to determine the input.
- Python standard library hash lib library with SHA-256 and SHA-512 hashing functions

### 4.3 Chain of blocks

- Blocks are now set up.
- Block chain is a collection of blocks and I have to implement it accordingly
- I can store all the blocks in Python in a list, but it will not work
- It is not enough
- The index can deliberately change a block of the previous index in an index / list
- Current (incomplete) implementation, creating a new block with changed transactions, counting the hash and replacing it with the old block functions
- I must maintain the absurdity and order of the hard blocks
- Block I need a way to ensure that any change in the previous blocks is not worth the entire chain
- One way to do this is to create sequential blocks with a hash
- I want to add the hash of the previous block in the current block by chaining it
- If any of the previous blocks change, the hash of the block will change, causing a mismatch with the previous hash field in the next block
- If each block is linked to the previous block by the previous hash field, I have to generate the first block myself
- The first block is called the source block and in most cases it is generated manually or by some special logic

### 4.4 Implementing a proof of work algorithm

- Selective support versus proof of work
- Business compliance for the private (private) block chain is not achieved through mining, but through a process known as selective support
- Network members control the transactions as they are verified and done today
- A problem arises: If I change the previous block, I can easily calculate the hash of all the following blocks and create a separate valid block chain
- To prevent this, I have to compute the hash strictly and randomly
- Instead of accepting any hash for the block, I'll interrupt something
- Hash Consistently let's start with a significant number of leading zeroes
- I know the hash doesn't change until I change the contents of the block
- I will start a new field in the block
- A non-number is a variable that can be changed as long as there is a hash that satisfies the constraint
- Leading defaults to the number 2 of leading zeros, which determines the difficulty of the POD algorithm
- This PoW algorithm is difficult to compute, but it is easy to verify if not found once
- Id validation involves replaying the hash function

### 4.5 Adding and mining blocks to the chain

- To add blocks in the In series
- Provided PoW is correct
- The previous hash field of the block is added to the hash of the latest block Chain
- At this point, I need to implement a mechanism for mining blocks
- Transactions are initially stored in a pool of unverified transactions. Block transaction mining is the process of placing unverified transactions into a block and calculating the PW.
- After a complete non-blocker is found, I can say that a block has been dug
- At that point, the block was placed on the block chain
- In most crypto currencies, miners can be hailed as some crypto currencies for calculating their computing power.

### 4.6 Creating interfaces for Flask web application

- I need to create interface for node to communicate with other peers and applications
- To build the REST-API for interacting with the node I will build it with the Flask Web Framework
- The app needs an end point for me to submit a new transaction
- It is used by the app to add new data (posts) to the block chain
- I also need an endpoint to return a copy of the node in the chain
- To Use this to query all posts to display to the user
- I need an endpoint to request a node for an indirect transaction
- It can be used to issue commands to Khan from within the app
- I will also add an end point for querying raw transactions
- At this point, I have a working block chain where I can create new transactions (posts) and do my job to add them to the block chain
- However, this time the code base is meant to run on the computer
- To manage the block chain I need to add functionality to multiple nodes

### 4.7 Establishing consensus and decentralization

- Although I associate a block with a hash, I cannot trust a single entity
- I need multiple nodes to manage the block chain
- I need to create an end point to mention the other peer nodes in the network
- I also need to create an endpoint to add new subscribers to the Peers network

- There is a problem with multiple nodes
- Due to deliberate manipulation or unintentional manipulation, some nodes may have different copied chains
- In that case, there should be an agreement on some version of the series
- The general consensus algorithm agrees on the longest valid series when looking at the range deviations of the different participants in the network
- The rationale behind this approach is that the long chain is a good predictor of work done at the highest volume
- I need to develop a way to declare that each node has dumped a network so that everyone can update their block chain and forward other transactions
- This is to create another endpoint to add user-mined blocks to successive nodes.
- After each node is mined by the node, it must be declared so that colleagues can add it to their chains
- This is called consensus, which must be obtained to protect the integrity of the entire system
- other nodes can only verify the proof of work and add it to their array

### 4.8 Application Structure

- At this point, the back-end is all setup
- I create the interface for the application, which is a view in the code base
- Using The Flask, I use Ginza 2 template to provide some CSS for web page and style
- The application must connect to the node in the block chain network to receive data and submit new data
- May also have multiple nodes
- An application has an HTML form for the user to take input, and then POST requests the connected node to add the transaction to the unconfirmed transaction pool
- The transaction is then mined by the network, and then the web site is refreshed.

## 5. Peer to peer network architecture

In peer-to-peer networks, data is still transmitted through the physical layer, but it is at the application layer, where colleagues can communicate directly with each other.

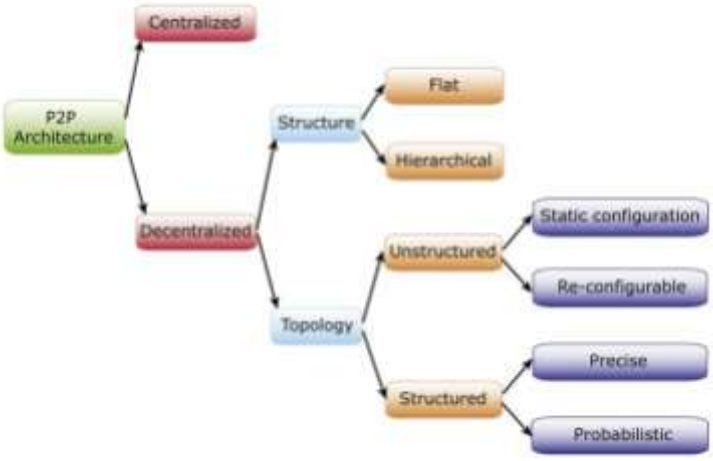


**Fig. 5.1 shows a tree for classifying the different structures of peer-to-peer networks**

### 5.1 Decentralized

In this structure, each fellow has the same rights and responsibilities. Each companion sees only a partial network and can provide only a few resources. Finding colleagues who provide the necessary resources becomes an important issue, which reflects their structure. Decentralized architecture can be further

classified based on network structure and topology.

### 5.2 Topology

In structured systems, each peer is only responsible for its own resources and contains a list of neighbors that can forward queries for additional resources. This makes the search time of O (n) worse because the query has to travel across the entire network to find the data.

On the other hand, data in structured systems with its peers can be mapped to some strategy in the form of distributed hash tables (DHT). For security reasons this is not real data, it can be mapped to peers but only to meta data. This reduces the search time for the additional cost of maintaining DHT

## 6. Advantage

- peer to networks are more fault tolerant
- If a peer fails the network can be used as a peer to peer network
- The number of peers that increase the bandwidth of the P2P network is shared among different peers
- The higher the number of peers, the faster the file can be downloaded because the same file can be downloaded from different peers
- Messages can be secure through the network

## 7. CONCLUSION

In this paper, we propose that the system proposed by Secure Messaging, Flash Framework enhances privacy. This system can be used peer-to-peer systems more efficiently.

They can use the encryption technology SHA-512. Designed applications fall under the block chain, they use peer-to-peer networks that protect the message. They can be networked through communication, reducing the cyber attack on the message.

## 8. REFERENCE

[1] N. Aitzhan and D. Svetinovic, "Security and Privacy in Decentralized Energy Trading Through Multi-Signatures, Blockchain and Anonymous Messaging Streams", *IEEE Transactions on Dependable and Secure Computing*, vol. 15, no. 5, pp. 840-852,2018.

[2] t. Gu, "Onion: Peer-to-Peer Anonymous Message with Promotional Systems", Master, The University of Bright Columbia, 2018.

[3] H. Halpin and M. Pirska, "Introduction to Security and Privacy on the Blockchain", 2017 IEEE European Symposium on Security and Privacy Workshops (Euro and PW), p. 2.2017.

[4] J. Parthala, "Provably Secure Secret Communication on the Blockchain", Cryptography, Vol. 2, no. 3, p. 18.2018.

[5] G. Ziskin, o. Nathan and A. Pentland, "Decentralizing Privacy: Using Blockchain to Protect Personal Data", 2015 IEEE Security and Privacy Workshop, 2015.

[6] "web3.shh - web3.js 1.0.0 Documentation", web3js.readthedocs.io, 2018. [Online]. Available at: https://web3js.readthedocs.io/en/1.0/web3- shh.html #. [Access: 2 Access - August 2011-].

[7] P. Memouncov and D. "Cadelia: A Peer-to-Peer Information System, Based on XOR Metrics" in Meziers, Peer-to-Peer Systems. Springer, 2002, pp.53-65.

[8] Z. Zheng, S. Xie, H. Dai, X. Chen, and H. Wang, "An Overview of Blockchain Technology: Architecture, Consensus, and Future Trends," *2017 IEEE International Congress on Big Data (BigData Congress)*, pp. 558–560,2017.

[9] V. Botterin, "A Next Generation Smart Contract and Decentralized Application Platform," White Paper, 2014.

[10] S. Neighborsages & c. Shelder, "Hashed Patricia Troy: Longest Prefix Matching Efficiency in Peer-to-Peer Systems", Walcom: Algorithms and Computations, pp. 1703–1,2011.