Journal of Theoretical and Applied Information Technology
15th April 2022. Vol.100. No 7

ISSN: 1992-8645 www.jatit.org E-ISSN: 1817-3195

ETHEREUM-INSPIRED ACCESS MANAGEMENT ACCOUNT CONTROL FOR A SECURED DECENTRALIZED CLOUD STORAGE

***HALA A. ALBAROODI[1], MOHAMMED ANBAR[2]**

[1]Dr. Senior Lecturer at Gifted guardianship committee, in Ministry of Education, Baghdad/Iraq
[2] National Advanced IPv6 Centre (NAv6), Universiti Sains Malaysia (USM), 11800 Gelugor, Penang, Malaysia

E-mail: [1]hala.albaroodi5@gmail.com, hala.albaroodi@iraqiggc.edu.iq, [2]anbar@usm.my

ABSTRACT

The exponential increase of data storage in cloud computing has raise issues in term of data security and privacy due to data are usually stored on the cloud server ciphertext from. When a user requests access to the encrypted data, an access key distributed by a third party is needed. However, if the third party is not trusted, the security of the system will be threatened easily. Therefore, this paper proposes an approach of blockchain based access management account control to enhance cloud storage security. The proposed approach is a combination of Ethereum Blockchain Contract Accounts (CA) and ciphertext encryption policy and it consists of two main stages. First, access control as the creation and invocation of each contract can be stored in the Blockchain, thus, the function of the trace is achieved. Second, data storage; where the data owner can store ciphertext policy of data in a Blockchain network. In the Blockchain technique data owner can set valid access periods for data usage so that the ciphertext can only be decrypted during valid access periods. Finally, the analysis of the security and experiment shows that proposed approach is feasible in securing the cloud storage.

Keywords: *Security; Blockchain; Cloud Computing, Software as a Services (SaaS), Data Storage, keystor, Accounts Manager, Cloud Storage Security.*

1. INTRODUCTION

In recent years, there has been a rapid advancement in technological innovation and related research on collaborative approaches for sharing users' data among enterprises (Shrestha and Vassileva, 2016) [1]. Research backed data sharing practices are much needed to strike a balance between user privacy, enhanced user experience and profit for businesses (Tenopir et al., 2011) [2]. The questions of when and what data should be shared to whom (Meadows, 2014) [3], and how the data owner should get credit or incentive to share their data are increasingly a matter of intense debate and research. User data is collected by different parties, for example, companies offering apps, social networking sites, and others, whose primary motive is to have enhanced business model while giving optimal services to their customers. However, the collection of user data is associated with serious privacy issues. Some data are contributed voluntarily by the user; others are obtained by the system from observation of user activities, or inferred through advanced analysis of volunteered or observed data (Poslad, 2009) [4]. The currently dominant model of ownership over user data, usually encoded in the service license agreements, presumes that ownership is transferred from the user to the enterprise collecting it and, if shared to the entire network of businesses.

There are privacy and security problems associated with storing personal data. Even the most prominent online services have experienced security breaches and data theft. When trust resides within a centralized service provider for all the storage of data, it could be affected with centrality issues such as intentionally deleting the user data or not delivering the user data due to a technical failure [5].

Sharing user data across applications and enterprises helps to improve personalization of functionality, interface and options and thus creates a better user experience. However, there are problems associated with the data security,

privacy and user control. Security of sharing has been addressed by standard security techniques as well as experimental scenario, for example, carrying out all the communication without trusting anybody and possibly replacing the centralized controlling authority [6]. Various advanced technologies have been deployed as computational backbones to collect and share user data including cloud computing services, Blockchain technology, Radio-Frequency Identification (RFID), as well as various security technologies to protect user data from the hackers (Shrestha, 2014) [7]. Federated learning (McMahan and Ramage, 2017) allows to mine data scattered in distributed locations [8]. However, the data security and privacy in CC are still challenging problems which need to be addressed. Therefore, this paper aims to tackle this problem by combining Blockchain-inspired access management account control for decentralized cloud storage. Proposing such approach will ensure scalable storage services without compromising the security. In particular, the proposed approach will strength the security for the completed eco-system.

The rest of the paper is organized as follows: provides an overview of cloud computing as well as the blockchain technology, and Ethereum. A brief analysis of the proposed approach for user data storage and sharing with their limitations is provided in section 3. Section 3.1 and 3.2 presents the proposed platform and its implementation. Evaluation for decentralized data sharing in a travel domain while ensuring users' privacy is presented in section 4. In section 5, we conclude our work.

2. BACKGROUND

Both cloud computing and Blockchain are playing a vital role in changing enterprises' work environments and the way traditional computing works. Their emergence has not only gained momentum in the existing business infrastructure but has also changed the way the world of application development, storage, online transaction, and other services functions. Although cloud is a well-oiled model that can accelerate Blockchain projects, this merger of Blockchain cloud services are still in infancy. Let us now dig deeper into cloud computing and Blockchain [9].

2.1 Cloud Computing

Cloud computing is the delivery of computing services that includes software, storage, servers, database, networking, analytics, intelligence, etc. over the internet. All of these computing services provide flexible utilization of resources, rapid innovation, and economies of scale [10].

A cloud computing is something that we can gain access to through the internet. It is cyberspace where we can access the data online. Meanwhile, cloud computing are provides services in three principal formats such as Software as a Service (SaaS), Platform as a Service (PaaS), Infrastructure as a Service (IaaS), as illustrated below in Figure. 1.

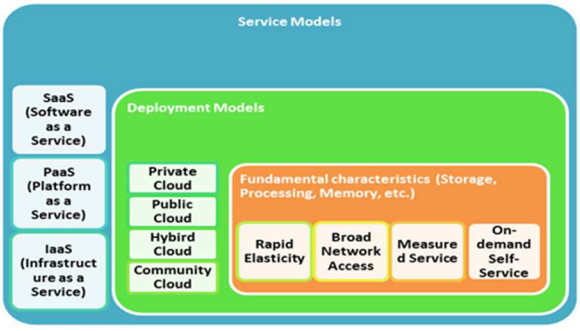

Figure 1: Security elements in (SaaS, PaaS, IaaS) [10].

2.1.1 Security issues in Cloud Computing

Cloud computing can push the execution of Blockchain technology-based projects with centralized control (as all the data remains stored in a company's centralized set of data centers) structure of data fetching [11].

Centralized systems have a core authority that dictates the truth to the other participants in the network. Only privileged users or institutions can access the history of transections or confirm new transections. Data and its existence in a cloud can be either public or private, or hybrid which means it can be either visible or kept hidden from other users. Cloud computing mostly runs on a traditional database structure where the stored data resides in the machines involving participants. However, cloud computing presents an added level of risk because essential services are often outsourced to a third party, which makes it harder to maintain data security and privacy, support data and service availability, and demonstrate compliance [12]. As indicated in Figure 2.

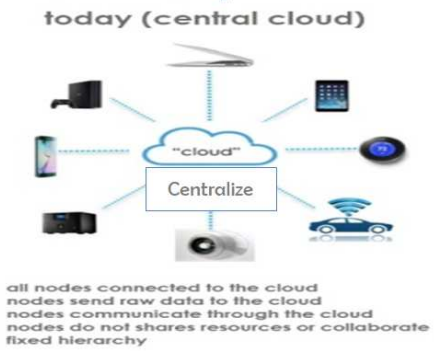

Figure 2: Central database on Cloud.

Some of Companies like Amazon Web Services (AWS), Alibaba Cloud, Google, IBM, and Microsoft provide cloud computing services. The application of cryptographic methods and the sharing of encryption data keys can help maintain confidentiality of sensitive user data against untrusted servers. Moreover, the cryptographic key management need to secure the cryptographic key management system in the cloud paradigm [13].

According to Bennani, key management for databases in cloud computing environments enables subsets of users to use encrypted/decrypted data. The Bennani framework suggests a method by which an encrypted database can be shared among different users in cloud computing environments (Bennani, N. et al., 2010) [14].

2.2 Blockchian

Blockchain is Distributed Ledger Technology (DLT) as it creates shared ledger databases. This technology also keeps a record of the history of the transacted digital assets that are unchangeable and transparent by decentralization and cryptographic hashing. Here, multiple parties agree on transaction specifications while guaranteeing accuracy and preventing tampering of data/records without the support of any trusted centralized authority. Blockchain has gained exponential popularity due to its promising and revolutionary technology. It reduces the risk of any technical transaction, casts out fraud, and delivers transparency in a scalable way for multiple uses. There are three essential concepts in Blockchain: Blocks, Nodes, and Miners [15].

All the data resides within the block. The system randomly generates a 32-bit whole number known as nonce as and when the block generates. A block header hash is then generated. Hash is a 256-bit cipher merged with the nonce. Blockchain classified four main types based on access limit: Public Blockchain, Privet Blockchain, Consortium Blockchain, and Blocks have three essential elements' all the data resides within the block [16].

The system randomly generates a 32-bit whole number known as nonce as and when the block generates. A block header hash is then generated. Hash is a 256-bit cipher merged with the nonce. Data in the form of records are immutable in Blockchain, whereas data residing in the cloud are mutable. Blockchain does not provide any service as it is a magnificent technological advancement that is a decentralized, distributed ledger that keeps a record of the provenance of a digital asset. Blockchain networks can have different implications regarding the privileges for participation and data access. Therefore, you can find two different types of labeling for Blockchain networks. Depending on the privileges for participation in the Blockchain networks [17]. On the other hand, the methods for participants to gain access to the network determine whether the Blockchain network is permissioned or permissionless as will discussed in the next secition.

2.2.1 Security issues in Blockchain

There are three categories of the blockchain, each with a slightly different set of protocols and consensus mechanisms. The consensus is to achieve agreement across validators (or miners) in a network on every new ledger of transactions. The blockchain is usually equipped with consensus protocols to tolerate unreliable involved parties or malicious nodes. The first category of blockchain is public in which anyone can participate in the chain and contribute to the consensus process. The read permission or the right to see the public blockchain is always open to anyone with access to the internet. The second category of blockchain is a consortium in which pre-selected nodes control the consensus process.

The right to see the consortium blockchain remains either public or restricted to the participants. The third type is private blockchain in which the transactions are contained within a closed community and are of interest only to the members of the community present in the chain [18].

The private blockchain adopts the core idea of blockchain as a distributed ledger technology (DLT) but assigns the private validator, which is a member of a consortium or separate legal entities

of the same organization. Blockchain guarantees the prevention of the tempering of data without relying on any third-party trusted centralized authority, whereas the cloud does not assure complete integrity and tamper-free data. Blockchain has a core principle of decentralization, which means it does not store any of its information in one space. Some of projects like Ethereum, Bitcoin, Hyperledger Fabric, and Quorum use Blockchain technology [21,22]. In addition, decentralized systems have a no core authority that dictates the truth to the other participants in the network. Every participant's in the network can access the history of transections or confirm new transection as illustrated in Figure 3.

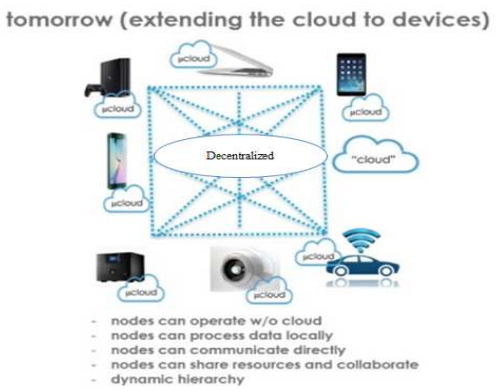

Figure 3: Decentralizing Database On Blockchain[19].

a) Security Issues in Public Blockchain

A public Blockchain is a Blockchain that anyone in the world can read, anyone in the world can send transactions to and expect to see them included if they are valid, and anyone in the world can participate in the consensus process for determining what blocks get added to the chain and what the current state is. As a substitute for centralized or quasi-centralized trust, public Blockchain are secured by crypto economics the combination of economic incentives and cryptographic verification using mechanisms such as proof of work or proof of stake, following a general principle that the degree to which someone can have an influence in the consensus process is proportional to the quantity of economic resources that they can bring to bear. These Blockchain are generally considered to be "fully decentralized". The advantages of public Blockchain generally fall into two major categories:

Public Blockchains provide a way to protect the users of an application from the developers, establishing that there are certain things that even the developers of an application have no authority to do. From a naive standpoint, it may be hard to understand why an application developer would want to voluntarily give up power and hamstring themselves. However, more advanced economic analysis provides two reasons why, in Thomas Schelling's words, weakness can be strength. First, if you explicitly make it harder or impossible for yourself to do certain things, then others will be more likely to trust you and engage in interactions with you, as they are confident that those things are less likely to happen to them. Second, if you personally are being coerced or pressured by another entity, then saying "I have no power to do this even if I wanted to" is an important bargaining chip, as it discourages that entity from trying to compel you to do it. A major category of pressure or coercion that application developers are at risk of is that by governments, so "censorship resistance" ties strongly into this kind of argument [20].

Public Blockchains are open, and therefore are likely to be used by very many entities and gain some network effects. To give a particular example, consider the case of domain name escrow. Currently, if A wants to sell a domain to B, there is the standard counterparty risk problem that needs to be resolved: if A sends first, B may not send the money, and if B sends first then A might not send the domain. To solve this problem, we have centralized escrow intermediaries, but these charge fees of three to six percent. However, if we have a domain name system on a Blockchain, and a currency on the same Blockchain, then we can cut costs to near-zero with a smart contract: A can send the domain to a program which immediately sends it to the first person to send the program money, and the program is trusted because it runs on a public Blockchain. Note that in order for this to work efficiently, two completely heterogeneous asset classes from completely different industries must be on the same database not a situation which can easily happen with private ledgers. Another similar example in this category is land registries and title insurance, although it is important to note that another route to interoperability is to have a private chain that the public chain can verify, BTC relay-style, and perform transactions cross-chain [21,22].

In some cases, these advantages are unneeded, but in others they are quite powerful enough to be worth 3x longer confirmation times and paying $0.03 for a transaction (or, once scalability technology comes into play, $0.0003 for a transaction). Note that by creating privately administered smart contracts on public Blockchain, or cross-chain exchange layers between public and private Blockchain, one can achieve many kinds of hybrid combinations of these properties. The solution that is optimal for a particular industry depends very heavily on what your exact industry is. In some cases, public is clearly better; in others, some degree of private control is simply necessary. As is often the case in the real world, it depends.

Even public Blockchain have some disadvantages like the network can be slow, and companies can't restrict access or use. If hackers gain 51% or more of the computing power of a public Blockchain network, they can unilaterally alter it, Godefroy said. Public Blockchain s also don't scale well. The network slows down as more nodes join the network.

b) Security Issues in Private Blockchain

Private Blockchain is a Blockchain where write permissions are kept centralized to one organization. Read permissions may be public or restricted to an arbitrary extent. Likely applications include database management, auditing, etc internal to a single company, and so public readability may not be necessary in many cases at all, though in other cases public auditability is desired. Advantages of private Blockchain like the consortium or company running a private Blockchain can easily, if desired, change the rules of a Blockchain, revert transactions, modify balances, etc. In some cases, eg. national land registries, this functionality is necessary; there is no way a system would be allowed to exist where Dread Pirate Roberts can have legal ownership rights over a plainly visible piece of land, and so an attempt to create a government-uncontrollable land registry would in practice quickly devolve into one that is not recognized by the government itself. Of course, one can argue that one can do this on a public Blockchain by giving the government a backdoor key to a contract; the counter-argument to that is that such an approach is essentially a Rube Goldbergian alternative to the more efficient route of having a private Blockchain, although there is in turn a partial counter-argument to that that I will describe later [20].

The validators are known, so any risk of a 51% attack arising from some miner collusion in China does not apply. Transactions are cheaper, since they only need to be verified by a few nodes that can be trusted to have very high processing power, and do not need to be verified by ten thousand laptops. This is a hugely important concern right now, as public Blockchain tend to have transaction fees exceeding $0.01 per tx, but it is important to note that it may change in the long term with scalable Blockchain technology that promises to bring public-Blockchain costs down to within one or two orders of magnitude of an optimally efficient private Blockchain system.

Nodes can be trusted to be very well connected, and faults can quickly be fixed by manual intervention, allowing the use of consensus algorithms which offer finality after much shorter block times. Improvements in public Blockchain technology, such as Ethereum 1.0's uncle concept and later proof of stake, can bring public Blockchains much closer to the "instant confirmation" ideal (eg. offering total finality after 15 seconds, rather than 99.9999% finality after two hours as does Bitcoin), but even still private Blockchains will always be faster and the latency difference will never disappear as unfortunately the speed of light does not increase by 2x every two years by Moore's law. If read permissions are restricted, private Blockchains can provide a greater level of, well, privacy as well as MultiChain [21,22].

Given all of this, it may seem like private Blockchain are unquestionably a better choice for institutions. However, even in an institutional context, public Blockchains still have a lot of value, and in fact this value lies to a substantial degree in the philosophical virtues that advocates of public Blockchain s have been promoting all along, among the chief of which are freedom, neutrality and openness. In addition, private Blockchain have disadvantages such as include the controversial claim that they aren't true Blockchains, since the core philosophy of Blockchain is decentralization. It's also more difficult to fully achieve trust in the information, since centralized nodes determine what is valid. The small number of nodes can also mean less security. If a few nodes go rogue, the consensus method can be compromised. Additionally, the

source code from private Blockchain is often proprietary and closed. Users can't independently audit or confirm it, which can lead to less security. There is no anonymity on a private Blockchain, either [21,22].

c) Security Issues in Consortium Blockchain

Consortium Blockchain, also known as a federated Blockchain, or permissioned Blockchain, is similar to a hybrid Blockchain in that it has private and public Blockchain features. But it's different in that multiple organizational members collaborate on a decentralized network. Essentially, a consortium Blockchain is a private Blockchain with limited access to a particular group, eliminating the risks that come with just one entity controlling the network on a private Blockchain.

In a consortium Blockchain, the consensus procedures are controlled by preset nodes [21,22]. It has a validator node that initiates, receives and validates transactions. Member nodes can receive or initiate transactions. Advantages of consortium Blockchain tends to be more secure, scalable and efficient than a public Blockchain network. Like private and hybrid Blockchain, it also offers access controls. Consortium Blockchain have some of disadvantages like less transparent than public Blockchain. It can still be compromised if a member node is breached; the Blockchain's own regulations can impair the network's functionality [20].

d) Security Issues in Hybrid Blockchain

Hybrid Blockchain, a type of Blockchain technology that combines elements of both private and public Blockchain. It lets organizations set up a private, permission-based system alongside a public permissionless system, allowing them to control who can access specific data stored in the Blockchain, and what data will be opened up publicly. Typically, transactions and records in a hybrid Blockchain are not made public but can be verified when needed, such as by allowing access through a smart contract. Confidential information is kept inside the network but is still verifiable. Even though a private entity may own the hybrid Blockchain, it cannot alter transactions. When a user joins a hybrid Blockchain, they have full access to the network. The user's identity is protected from other users, unless they engage in a transaction. Then, their identity is revealed to the other party [20].

One of the big advantages of hybrid Blockchain is that, because it works within a closed ecosystem, outside hackers can't mount a 51% attack on the network. It also protects privacy but allows for communication with third parties. Transactions are cheap and fast, and it offers better scalability than a public Blockchain network. Hybrid Blockchain have disadvantages of isn't completely transparent because information can be shielded. Upgrading can also be a challenge, and there is no incentive for users to participate or contribute to the network.

As summery the right to see the private Blockchain remains restricted to the participants. The Blockchain can also be referred to as permissioned or permissionless, each with slightly different properties. Permissioned Blockchain is faster, usually; a trusted network offering managed upkeep and private membership such that members can contribute to the consensus process only after meeting some criteria [21, 22]. On the other hand, permissionless Blockchain is slower, trust-free, open, transparent, and a public membership network such that any members can contribute to the consensus process without any restriction (Wood, 2016) [27]. Therefore, depending upon the consensus mechanism, different Blockchains may be suitable for distinct types of business use cases as shown in table 1.

Table 1 :Security Issue in Blockchain Categories

	Public	Private	Hybrid	Consortium
Permissioned	Not-Permission	Yes-Permission	Yes-Permission	Yes-Permission
speed	slow	fast	fast	fast
Read	Open to any one	Restricted to an authorize set of participant	Restricted to an authorize set of participant	All of sub set of authorize participant
Write	Anyone	Authorize participants	Authorize participants	Authorize participants
Trust	Trustless	Trusted	Trusted	Trusted
Decentralized	Fully	Not-decentralized	Partially centralized	Partially centralized
Identity	Not known	known	known	known
Consensuses	Proof-of-work	Proof-of-stake/ pre-approved participations	pre-approved participations	pre-approved participations

2.3 Management Account Control (Ethereum)

Accounts play a central role in Ethereum. There are two types of accounts: Externally Owned Accounts (EOAs) and Contract Accounts (CA). Here we focus on externally owned accounts, which will be referred to simply as accounts. Contract accounts will be referred to as contracts. This generic notion of account subsuming both

externally owned accounts and contracts is justified in that these entities are so called state objects [23-25].

These entities have a state: accounts have balance and contracts have both balance and contract storage. The state of all accounts is the state of the Ethereum network which is updated with every block and which the network really needs to reach a consensus about. Accounts are essential for users to interact with the Ethereum BlockChain via transactions. If we restrict Ethereum to only externally owned accounts and allow only transactions between them, we arrive at an "altcoin" system that is less powerful than bitcoin itself and can only be used to transfer ether.

Accounts represent identities of external agents (e.g., human personas, mining nodes or automated agents). Accounts use public key cryptography to sign transaction so that the EVM can securely validate the identity of a transaction sender.

2.3.1 Keyfiles

Every account is defined by a pair of keys, a private key and public key. Accounts are indexed by their address which is derived from the public key by taking the last 20 bytes. Every private key/address pair is encoded in a keyfile. Keyfiles are JSON text files which you can open and view in any text editor. The critical component of the keyfile, your account's private key, is always encrypted, and it is encrypted with the password you enter when you create the account. Keyfiles are found in the keystore subdirectory of your Ethereum node's data directory. It's important to back up our keyfiles regularly. Creating an account must create a key is tantamount [26].

- No need to tell anybody else you're doing it
- No need to synchronize with the BlockChain
- No need to run a client
- No need to be connected to the internet

Of course your new account will not contain any Ether. But it'll be yours and you can be certain that without your key and your password, nobody else can ever access it. It is safe to transfer the entire directory or any individual keyfile between Ethereum nodes. Once you have the geth client installed, creating an account is merely a case of executing the geth account new command in a terminal. Note that you do not have to run the geth client or sync up with the BlockChain to use the geth account command.

- $ geth account new

Your new account is locked with a password. Please give a password. Do not forget this password.

- Passphrase:
- Repeat Passphrase:
- Address: {168bc315a2ee09042d83d7c5811b533620531f67}

For non-interactive use you supply a plaintext password file as argument to the password flag. The data in the file consists of the raw bytes of the password optionally followed by a single newline.

- $ geth password /path/to/password account new

2.3.2 Encrypted Keystores

Although handling accounts locally to an application does provide certain security guarantees, access keys to Ethereum accounts should never lay around in cleartext form. As such, provide an encrypted keystore that provides the proper security guarantees for you without requiring a thorough understanding from your part of the associated cryptographic primitives. The important thing to know when using the encrypted keystore is that the cryptographic primitives used within can operate either in standard or light mode [28]. The former provides a higher level of security at the cost of increased computational burden and resource consumption:

- standard needs 256MB memory and 1 second processing on a modern CPU to access a key
- light needs 4MB memory and 100 millisecond processing on a modern CPU to access a key

As such, standard is more suitable for native applications, but you should be aware of the tradeoffs nonetheless in case you you're targeting more resource constrained environments.

2.3.3 Keystores from Go

The encrypted keystore is implemented by the accounts manager structure from (the github.com/ethereum/go-ethereum/accounts package) [28], which also contains the

configuration constants for the standard or light security modes. Note (to do client side account management from Go, you'll need to import only the accounts package into your code):

- import "github.com/ethereum/go-ethereum/accounts"
- import"github.com/ethereum/go-thereum/accounts/keystore"
- import "github.com/ethereum/go-ethereum/common"

Afterwards you can create a new encrypted account manager via:

- ks:=keystore.NewKeyStore("/path/to/keystore", keystore.StandardScryptN, keystore.StandardScryptP)
- am:=accounts.NewManager(&accounts.Config{InsecureUnlockAllowed: false}, ks)

The path to the keystore folder needs to be a location that is writable by the local user but non-readable for other system users (for security reasons obviously), so we'd recommend placing it either inside your user's home directory or even more locked down for backend applications.

2.3.4 Accounts from Go

An Ethereum account is implemented by the accounts.

- Account struct from the github.com/ethereum/go-ethereum/accounts package.

Assuming has an instance of accounts. Manager called am from the previous section, we can easily execute all of the described lifecycle operations with a handful of function calls (error handling omitted).

If accounts. Account can be used to access various information about specific Ethereum accounts, they do not contain any sensitive data (such as passphrases or private keys) [28], rather act solely as identifiers for client code and the keystore. Susceptibility of Authentication Data (Choudhury, A. J. et al., 2011) [29].

2.3.5 Signing from Go

Assuming we already has an instance of an account. Manager called am from the previous sections, then, can create a new account to sign transactions with via it's already demonstrated a new account method; and to avoid going into transaction creation for now, we can hardcode a random common (#Hash to sign instead).

Approach that maintains the integrity of data by using a consensus mechanism. A third-party intermediary is not required and trust is developed through a public ledger stored in a decentralized manner in order to ensure secure distributed transactions in a trustless environment [30]. The blockchain protocol is a potential candidate that may bring some evolutionary changes to traditional IT security [29].

3. THE PROPOSED APPROACH

The proposed approach decentralized cloud Blockchain storage, there is no trusted third party in the proposed approach. The proposed approach has two main stages. First, as the access control, data owner can set valid access periods for data usage so that the ciphertext can only be decrypted during valid access periods; as well as the creation and invocation of each contract can be stored in the Blockchain, thus, the function of the trace is achieved. Second, the cloud Blockchain technology is used; the data owner can store ciphertext of data in a Blockchain network. The analysis of the security and experiment shows that proposed approach is feasible.

This section thoroughly explains the proposed blockchain based access management account control approach to enhance cloud storage security. The proposed approach consists of two main stages namely (i) access control management and (ii) Blockchain cloud storage as shown in Figure 4, 5, and 6.

3.1 Access Control Management

A better access control model in cloud computing provides scalability, security, and the best access control mechanism for outsourced data in cloud computing by combining three different advanced cryptographic and re-encryption techniques (Yu, S. et al., 2010) [31].

Cloud computing servers maintain an Attribute History List (AHL) that includes the evolutionary history of each attribute and the PRE keys; different data files can contain common subsets of attributes. Each attribute is associated with a

version number for attribute update; one dummy attribute (AttD) is utilised for key management. This dummy attribute must be contained in every data file's attribute set and never updated; generally. Security documents suggests the use of eXtensible Access Control Markup Language (XACML) to control access to cloud computing resources. As clarified in Figure 4.

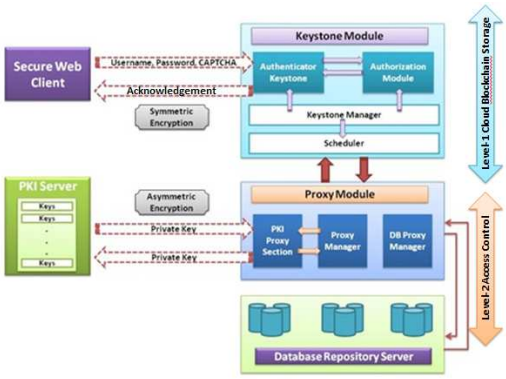

Figure 4: Access Control Management Main Steps

Access control services should be made sufficiently flexible when enforcing the principle of least benefit and they should integrate privacy protection requirements via the use of complex rules. However, (Fakhar, F. and Shibli, M. A., 2013) have suggested an effective, robust, and incorporable security protocol for symmetric cryptographic key management in the cloud computing environment [40]. Their technique is established using secret splitting and files computation mechanisms.

Furthermore, users need to access control and authentication must be designed to prevent accidental users from engaging in malicious practices and to prevent spammers and bots from registering on any site (Choudhury, A. J. et al., 2011) [29]. A user may consider registering for a Completely Automated Public Turing Test to Tell Computers and Humans Apart (CAPTCHA) as CAPTCHA is easy to set up and is well designed, free, and accessible. In addition, it does not require the installation of third-party libraries.

Moreover, several methods have been applied to provide a secure access to the cloud Blokchain resources. However, in many situations this could not be sufficient. For instance, in DDoS attacks, which are mainly based on robots, a mechanism to differentiate between real users and robots, need to be employed. CAPTACH could be one of the best selections to address this issue. In a study conducted by Huang, an implementation to the CAPTCHA in a cloud Blokchain environment has been reviewed [32].

In particular, the solution proposed in this work inherent from the idea of CAPTCHA along with other complicated mathematical-based mechanism which might be inappropriate for the implementation due to its complexity especially for entry-level end user devices such as portable devices.

Further, as evidence that cloud Blokchain suffers from lack such as the default implementation of CAPTCHA, this implementation has been further customized and represented to serve handheld devices with touch sensitivity in [33]. The proposed touch-based CAPTCHA could be differing in terms of user interaction from the typical CAPTCHA. However, this difference is beyond the concern of this paper except that the study shows that such implementation to differentiate between human and robots is important in a cloud computing environments.

3.1.1 Authentication And Authorization

Authorization is determines what level of access an authenticated user should have to the secure resources controlled by the model. A successful authentication creates a token that is used to authorize service requests. The password and username are granted as inputs to the API interface. When authentication is successful, the resulting feedback includes an authentication token and service catalogue [35-37]. Tokens remain valid for 12 hours. Issued tokens become invalid in two situations:

- If the token is expired.
- If the token has been cancelled.

The authentication must be executed over a safe channel, such as TLS. Otherwise, an attacker can obtain a user token by executing a MITM attack and can remove the user who received the token from the authentication system. However, Rostyslav Slipetskyy subjected the algorithms that are imported for token generation to a more detailed examination [36]. The algorithm imitates the approach used to generate a universally unique identifier (UUID) and utilizes a strong source of randomness. Thus, this algorithm is considered secure (Slipetskyy, R., 2011) [36]. Moreover, the

method for protecting sensitive data is the hash function; hash functions are primarily used to generate fixed-length output data that serve as a shortened reference to the original data. In different standards and applications, the most commonly used hash function are MD5 [38-40].

3.1.2 Signing Authorization

As mentioned above, account objects do not hold the sensitive private keys of the associated Ethereum accounts, but are merely placeholders to identify the cryptographic keys with all operations that require authorization (e.g. transaction signing) are performed by the account manager after granting it access to the private keys.

There are a few different ways one can authorize the account manager to execute signing operations, each having its advantages and drawbacks. Since the different methods have wildly different security guarantees, it is essential to be clear on how each works:

Single authorization: The simplest way to sign a transaction via the account manager is to provide the passphrase of the account every time something needs to be signed, which will ephemerally decrypt the private key, execute the signing operation and immediately throw away the decrypted key. The drawbacks are that the passphrase needs to be queried from the user every time, which can become annoying if done frequently; or the application needs to keep the passphrase in memory, which can have security consequences if not done properly; and depending on the keystore's configured strength, constantly decrypting keys can result in non-negligible resource requirements.

Multiple authorizations: A more complex way of signing transactions via the account manager is to unlock the account via its passphrase once, and allow the account manager to cache the decrypted private key, enabling all subsequent signing requests to complete without the passphrase. The lifetime of the cached private key may be managed manually (by explicitly locking the account back up) or automatically (by providing a timeout during unlock). This mechanism is useful for scenarios where the user may need to sign many transactions or the application would need to do so without requiring user input. The crucial aspect to remember is that anyone with access to the account manager can sign transactions while a particular account is unlocked (e.g. application running untrusted code).

3.1.3 Account Lifecycle

Having created an encrypted keystore for your Ethereum accounts, you can use this account manager for the entire account lifecycle requirements of your native application. This includes the basic functionality of creating new accounts and deleting existing ones; as well as the more advanced functionality of updating access credentials, exporting existing accounts, and importing them on another device.

Although the keystore defines the encryption strength it uses to store your accounts, there is no global master password that can grant access to all of them. Rather each account is maintained individually and stored on disk in its encrypted format individually, ensuring a much cleaner and stricter separation of credentials. This individuality however means that any operation requiring access to an account will need to provide the necessary authentication credentials for that particular account in the form of a passphrase:

Either creating a new account, the caller must supply a passphrase to encrypt the account with. This passphrase will be required for any subsequent access, the lack of which will forever forfeit using the newly created account. Or deleting an existing account, the caller must supply a passphrase to verify ownership of the account. This isn't cryptographically necessary, rather a protective measure against accidental loss of accounts.

If updating an existing account, the caller must supply both current and new passphrases. After completing the operation, the account will not be accessible via the old passphrase any more.

Exporting an existing account, the caller must supply both the current passphrase to decrypt the account, as well as an export passphrase to re-encrypt it with before returning the key-file to the user. This is required to allow moving accounts between machines and applications without sharing original credentials. While importing a new account, the caller must supply both the encryption passphrase of the key-file being imported, as well as a new passphrase with which to store the account. This is required to allow storing account with different credentials than used for moving them around.

The proposed approach as indicated in Figure. 4 deals with encryption and decryption flow of.. The design of flows and sequences are translated to the algorithms and pseudo-code. Then, it is implemented by software developer using programming language such as Java that is appropriate for the requirement and design. After the implementation and Internal testing, the output of the design is then analyzed and verified as to confirm whether the end result is equivalent to the projected design output. Windows/Linux is utilized as the cross platform. The database management system will be My Sequel (mySQL), which is free and open source and can handle well more than thousand users simultaneously. As verified in Figure 5.

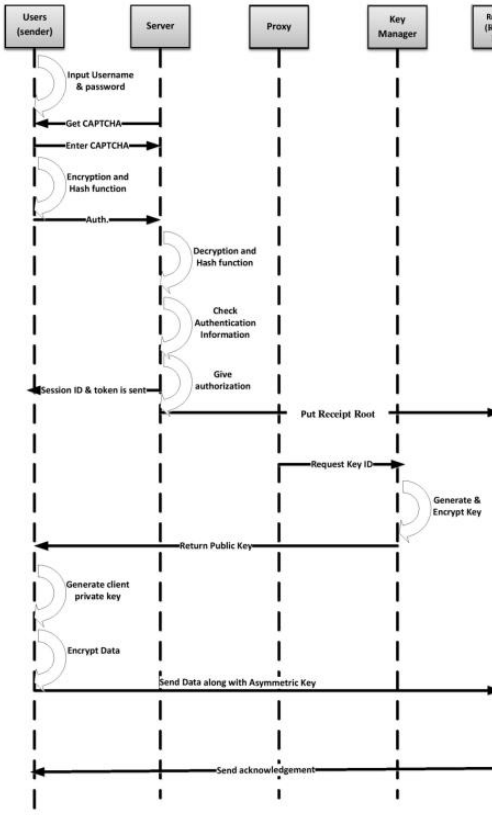

Figure 5: Sequence Diagram Of Proposed Approach

3.2 BLOCKCHAIN CLOUD STORAGE

Access control to your data in the cloud Blockchain storage server is challenging and difficult to implement when sensitive data are located outside on un-trusted domain. The cloud Blokchain is multi user based system where every user would encrypt her/his files; each one will be using different levels of cryptographic security (Jansen, W. and Grance, T., 2011). It is important to reduce the key distribution complexity in such diverse system with variety of security-based settings.

It is worth mentioning that the proposed approach will be using the cryptographic algorithms. In addition to multi user access control model for databases that use Blockchain technology to provide stable, distributed data processing. The model allows the data owner to upload the data via a web portal. So, the user who has the secret key to the particular data that has been uploaded to cloud in encrypted form can only access the folder. Eventually, the system promotes data privacy by maintaining the immutability of the Blockchain by processing it in the cloud. We have proposed a secure; Blockchain based data storage and access control system to increase the security of cloud storage. as clarified in Figure 6.

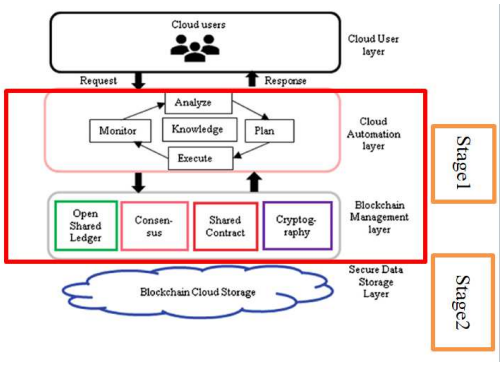

Figure 6: Cloud Blokchain Storage.

A logical isolation mechanism for multi tenancy cloud storage is proposed in the proposed approach. The proposed approach is suggested to avoid unauthorized data access within the same tenant as well as across different tenants. Authorisation is crucial for ensuring that only authorised entities can interact with the data in cloud Blokchain. The difference between the authentication and the authorization of the cloud Blokchain is the backend repository into which the user data is stored normal user or admin(Choudhury, A. J. et al., 2011) [29].

The proposed approach has a strong user authentication for the proposed approach in which a legitimate user proves his/her authenticity prior gaining access to the proposed approach that cloud

Blokchain paradigm. The proposed approach employs a two-step verification procedure involving passwords and CAPTCHA.

The purpose of this proposed approach is to deploy mutual authentication, session key establishment, identity management, user privacy, and security against many popular forms of hacking. The server is assumed to never grant concessions to network adversaries in cases in which attacks occur after a connection is established. According to the NIST, the enterprise must use two authentication models: an internal model and an external model, which is considered to be a temporary solution; both models are custom implementations in cloud Blokchain.

4. EVALUATION PEER PARTICIPANTS SCENARIO

Let's begin by examining simple symmetric key encryption. The same key is used for encryption and decryption, so it is called symmetric key. Look at this scenario, Ceasar encryption is the simplest one with alphabets of a message are shifted by a fixed number, and this number is called the Key. In this example F is a function defined by shift by three in alphabet. Consider, "Meet me at the cinema." You shift by three the S key value of every letter to encrypt it, and your receiver decrypts it using the same three as the key. Shift the other way every character to view the original message. Three is the key in this trivial example. Since the same key is used for encryption and decryption, it is a symmetric key. Note that the key and the encryption and decryption functions are typically much more complex in a real application.

Note: that symmetric key encryption has issues. Number one, it is easy to derive the secret key from the encrypted data. And number two, the key distribution. These issues are further exasperated in a Blockchain decentralized network where participants are unknown to each other.

Let's now examine how Public-key cryptography addresses these issues. Instead of a single secret key, it employs two different keys that take care of both the issues of symmetric key encryption. Look at this scenario, lowercase b uppercase B be the private public-key pair for a participant in Baghdad/ Iraq/ Iraq. Let lowercase k and uppercase K be the pair of keys for the participant and Karbalaa/ Iraq. Public-key is published; private key is kept safe and locked. Typically using a passphrase and the pair works as follows; encrypting function holds two properties with a key pair. The public-key private key pair has the unique quality that even though a data is encrypted with the private key, it can be decrypted with the corresponding public-key and vice versa. As clarified in Figure 7.

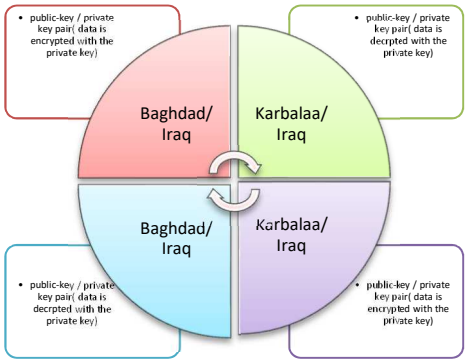

Figure 7: Examine Public-Key Cryptography Addresses Scenario.

Now let's from different perspective on the previous scenarios, Authenticate the sender and the receiver. We'll examine just one common use of a symmetric key encryption. Let's say a participant in Baghdad/ Iraq wants to transact with the participant in Karbalaa/ Iraq. Instead of sending just a simple message, a participant in Baghdad/ Iraq will send a transaction data encrypted by Baghdad/ Iraq's private key, and then encrypted by Karbalaa/ Iraq's public key. Karbalaa/ Iraq will first decrypt the data using its own private key, then use Baghdad/ Iraq's public key to decrypt assigned transaction data. This ensures that only Karbalaa/ Iraq can decrypt and receive the data and that only Baghdad/ Iraq could have sent the data.

5. CONCLUSION

A decentralized Blockchain is distributed and an open ledger that records transactions. It is a peer to peer network. All the peers share the responsibility of running the network; a server manages complete access to an application. A Person with full access to the server can control the application fully. The decentralized application (dApp) is distributed across the Blockchain. The data is distributed and cannot be changed on Blockchain.

In the proposed approach, we developed a decentralized application where the personal data access by any organization is recorded, and this data is stored on Blockchain where it cannot be altered. The decentralized proposed approach application also gives full control to the user to share the personal information with the organization they like. The personal data is encrypted using symmetric algorithm and the keys are further encrypted to add another layer of security. Finally, as the proposed approach creation and invocation of each contract can be stored in the Blockchain.

The privacy of user information should be ensured when using Blockchain in the cloud computing environment and the user information should be completely encrypted. If the user information is not encrypted and left behind, the user information can be guessed from the remaining information. Therefore, this study discussed the proposed approach of providing security by presenting a secure Blockchain use and encryption technique. The analysis of the security and experiment shows that the proposed approach has achieved better storages scalability while providing higher security outcomes compared to other approaches.

REFERENCES

[1] Shrestha AK, Vassileva J. Towards decentralized data storage in general cloud platform for meta-products. InProceedings of the International Conference on Big Data and Advanced Wireless Technologies 2016 Nov 10 (pp. 1-7).

[2] Tenopir C, Allard S, Douglass K, Aydinoglu AU, Wu L, Read E, Manoff M, Frame M. Data sharing by scientists: practices and perceptions. PloS one. 2011 Jun 29;6(6):e21101.

[3] Ndiyoi M, Rweyemamu M, Meadows K. Strengthening livelihoods through food and nutrition security in vulnerable SADC countries. Midterm review of OSRO/RAF/510-511/SAF, RIACSO, Johannesburg, viewed. 2014 Jan;20.

[4] Poslad S. Using Multi-agent Systems to Specify Safe and Secure Services for Virtual Organisations. InSafety and Security in Multiagent Systems 2009 (pp. 258-273). Springer, Berlin, Heidelberg.

[5] Landau S. Privacy and security A multidimensional problem. Communications of the ACM. 2008 Nov 1;51(11):25-6.

[6] Shrestha AK, Vassileva J. Towards decentralized data storage in general cloud platform for meta-products. InProceedings of the International Conference on Big Data and Advanced Wireless Technologies 2016 Nov 10 (pp. 1-7).

[7] Shrestha P, Maharjan S, de la Rosa GR, Sprague A, Solorio T, Warner G. Using String Information for Malware Family Identification. InIbero-American Conference on Artificial Intelligence 2014 Nov 24 (pp. 686-697). Springer, Cham.

[8] McMahan B, Moore E, Ramage D, Hampson S, y Arcas BA. Communication-efficient learning of deep networks from decentralized data. InArtificial intelligence and statistics 2017 Apr 10 (pp. 1273-1282). PMLR.

[9] Shrestha AK, Vassileva J, Deters R. A blockchain platform for user data sharing ensuring user control and incentives. Frontiers in Blockchain. 2020 Oct 22;3:48.

[10] Albaroodi H, Manickam S, Bawa PS. Critical Review of OpenStack Security: Issues and Weaknesses. J. Comput. Sci.. 2014 Jan 1;10(1):23-33.

[11] Albaroodi, Hala A., Selvakumar Manickam, Mohammed Faiz Aboalmaaly, and Hemananthan Palakarnim. "A Pilot Study on Open Source Cloud Computing (OSCC) Awareness in the Universiti Sains Malaysia Education Sector." 1264-1273.

[12] Wang Z, Zhu Y. A centralized HIDS framework for private cloud. In2017 18th IEEE/ACIS International Conference on Software Engineering, Artificial Intelligence, Networking and Parallel/Distributed Computing (SNPD) 2017 Jun 26 (pp. 115-120). IEEE.

[13] Vilaplana J, Solsona F, Abella F, Filgueira R, Rius J. The cloud paradigm applied to e-Health. BMC medical informatics and decision making. 2013 Dec;13(1):1-0.

[14] Bennani N, Guegan CG, Musicante MA, Solar GV. Sla-guided data integration on cloud environments. In2014 IEEE 7th International Conference on Cloud Computing 2014 Jun 27 (pp. 934-935). IEEE.

[15] Alharby M, van Moorsel A. Blocksim: a simulation framework for blockchain systems. ACM SIGMETRICS Performance Evaluation Review. 2019 Jan 25;46(3):135-8.

[16] Lin IC, Liao TC. A survey of blockchain security issues and challenges. Int. J. Netw. Secur.. 2017 Sep 1;19(5):653-9.

[17] Workie H, Jain K. Distributed ledger technology: Implications of blockchain for the securities industry. Journal of Securities Operations & Custody. 2017 Aug 1;9(4):347-55.

[18] Shetty S, Kamhoua CA, Njilla LL, editors. Blockchain for distributed systems security. John Wiley & Sons; 2019 Apr 16.

[19] Matrix AI Network. (2021, 30-12-2011). New Direction for Public Chains in the Carbon

Neutral Future (Part Two). Available: https://matrixainetwork.medium.com/new-direction-for-public-chains-in-the-carbon-neutral-future-part-two-4b6935a4a862
[20] Hassan MU, Rehmani MH, Chen J. Privacy preservation in blockchain based IoT systems: Integration issues, prospects, challenges, and future research directions. Future Generation Computer Systems. 2019 Aug 1;97:512-29.
[21] Lin IC, Liao TC. A survey of blockchain security issues and challenges. Int. J. Netw. Secur.. 2017 Sep 1;19(5):653-9.
[22] Stephen R, Alex A. A review on blockchain security. InIOP Conference Series: Materials Science and Engineering 2018 Aug 1 (Vol. 396, No. 1, p. 012030). IOP Publishing.
[22] Iyer K, Dannen C. The ethereum development environment. InBuilding games with ethereum smart contracts 2018 (pp. 19-36). Apress, Berkeley, CA.
[23] Aung YN, Tantidham T. Review of Ethereum: Smart home case study. In2017 2nd International Conference on Information Technology (INCIT) 2017 Nov 2 (pp. 1-4). IEEE.
[24] Metcalfe W. Ethereum, Smart Contracts, DApps. InBlockchain and Crypt Currency 2020 (pp. 77-93). Springer, Singapore.
[25] Panescu AT, Manta V. Smart contracts for research data rights management over the ethereum blockchain network. Science & Technology Libraries. 2018 Jul 3;37(3):235-45.
[26] Team UP. Security Analysis of TrueCrypt 7.0 a with an Attack on the Keyfile Algorithm. Technical report (Aug. 14, 2011). https://www. privacy-cd. org/downloads/truecrypt 7.0 aanalysis-en. pdf; 2011 Aug 14.
[27] Wood, G. 2016. Ethereum: A Secure Decentralized Generalised Transaction Ledger. Github. https://github.com/ethereum/yellowpaper.Google Scholar
[28] Tayal S, Gupta N, Gupta P, Goyal D, Goyal M. A review paper on network security and cryptography. Advances in Computational Sciences and Technology. 2017;10(5):763-70.
[29] Kumar P, Choudhury AJ, Sain M, Lee SG, Lee HJ. RUASN: a robust user authentication framework for wireless sensor networks. Sensors. 2011 May;11(5):5020-46.
[30] Lim SY, Fotsing PT, Almasri A, Musa O, Kiah ML, Ang TF, Ismail R. Blockchain technology the identity management and authentication service disruptor: a survey. International Journal on Advanced Science, Engineering and Information Technology. 2018 Sep;8(4-2):1735-45.
[31] Yu S, Wang C, Ren K, Lou W. Achieving secure, scalable, and fine-grained data access control in cloud computing. In2010 Proceedings IEEE INFOCOM 2010 Mar 14 (pp. 1-9). Ieee.
[32] Huang X, Zhou J, editors. Information Security Practice and Experience: 10th International Conference, ISPEC 2014, Fuzhou, China, May 5-8, 2014, Proceedings. Springer; 2014 Apr 28.
[33] Shrestha B, Saxena N, Harrison J. Wave-to-access: Protecting sensitive mobile device services via a hand waving gesture. InInternational Conference on Cryptology and Network Security 2013 Nov 20 (pp. 199-217). Springer, Cham.
[34] R. H. Khan, J. Ylitalo, and A. S. Ahmed, "OpenID authentication as a service in OpenStack?" Information Assurance and Security (IAS), 2011 7th International Conference on, 372-377
[35] K. Jackson, *OpenStack Cloud Computing Cookbook*: Packt Publishing Ltd, 2012.
[36] G. von Laszewski, J. Diaz, F. Wang, and G. C. Fox, "Comparison of multiple cloud frameworks?" Cloud Computing (CLOUD), 2012 IEEE 5th International Conference on, 734-741
[37] Rivest R, Dusse S. The MD5 message-digest algorithm.
[38] Sivathanu G, Wright CP, Zadok E. Ensuring data integrity in storage: Techniques and applications. InProceedings of the 2005 ACM workshop on Storage security and survivability 2005 Nov 11 (pp. 26-36).
[39] Dabas P, Wadhwa D. A recapitulation of data auditing approaches for cloud data. Int. J. Comput. Appl. Technol. Res.(IJCATR). 2014;3(6):329-32.
[40] Fakhar F, Shibli MA. Management of Symmetric Cryptographic Keys in cloud based environment. In2013 15th International Conference on Advanced Communications Technology (ICACT) 2013 Jan 27 (pp. 39-44). IEEE.