ISSN:0975-9646

A Smart Crypto Scheme for Multi Owner Data Authentication over Cloud Service

V.Kiruthika,
Research Scholar ,
School Of IT & Science,
Dr. G.R.D College Of Science ,
Coimbatore,
Tamilnadu.

B.R.Laxmi Sree,
Assistant Professor,
School Of IT & Science,
Dr. G.R.D College Of Science ,
Coimbatore,
Tamilnadu.

***Abstract-* The proposed system implements a prevention technique to protect trespassers, the trespassers is someone who intrudes on the privacy or property of another without permission. This also helps to avoid brute force and dictionary attacks. Text password is the most popular form of user authentication on websites due to its convenience and simplicity. Firstly, users often select weak passwords and reuse the same passwords across different websites. Routinely reusing or using passwords on different unknown machine causes a domino effect; when an adversary compromises one password, she will exploit it to gain access to more websites. Second, typing passwords into un trusted computers suffers password thief threat. An adversary can launch several password stealing attacks to snatch passwords, such as phishing, key loggers and malware .In this paper, we design a user authentication protocol that only requires each participating website possesses a unique phone Number, and involves a telecommunication service provider in registration and recovery phases. Through SMS the encrypted password will be send to the user. The user should forward the same message with their security answer to the server. The server will verify the details and allow them to access their personal page. Here also problem occurs if the user have joint account, here the SMS send to both the person, if one is using net banking then the message is send to both the user from the server side, and then ask for permission whether the other one can use net banking, if the other one says yes to this queries, the other one can start his/her transactions in a secure manner.**

***Index Terms*—Cloud storage, data sharing, key-aggregate encryption, patient-controlled encryption, MOCK Scheme.**

*1.*INTRODUCTION

The term “cloud”, as used in this white paper, appears to have its origins in network diagrams that represented the internet, or various parts of it, as schematic clouds. “Cloud computing” was coined for what happens when applications and services are moved into the internet “cloud.” Cloud computing is not something that suddenly appeared overnight; in some form it may trace back to a time when computer systems remotely time-shared computing resources and applications. More currently though, cloud computing refers to the many different types of services and applications being delivered in the internet cloud, and the fact that, in many cases, the devices used to access these services and applications do not require any special applications.

Many companies are delivering services from the cloud. Some notable examples as of 2010 include the following:

- Google — Has A Private Cloud That It Uses For Delivering Many Different Services To Its Users, Including Email Access, Document Applications, Text Translations, Maps, Web Analytics, And Much More.
- Microsoft — Has Microsoft Sharepoint Online Service That Allows For Content And Business Intelligence Tools To Be Moved Into The Cloud And Microsoft Currently Makes Its Office Applications Available In A Cloud.
- Salesforce.Com — Runs Its Application Set For Its Customers In A Cloud, And It’s Force.Com And Vmforce.Com Products Provide Developers With Platforms To Build Customized Cloud Services.

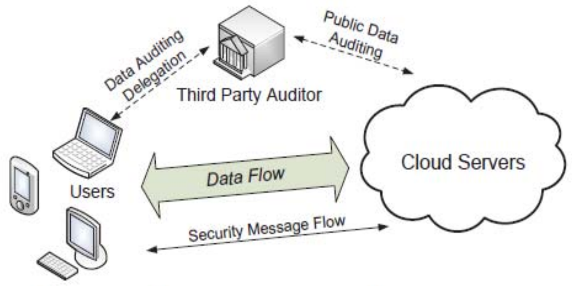

Fig. 1: The architecture of cloud data storage service

*2.*PROBLEM DEFINITION

Data sharing is an important functionality in cloud and network storage. This is very important in distributed environment. When considering the flexibility and scalability in data sharing, there is a huge issue regarding the security. Efficient data encryption and key sharing schemes have been proposed recently, even those schemes were not completely secure in the multi owner data sharing environment. For both security and efficiency, a group key which is shared only by a group of users has been employed for access control. A message for the group is encrypted by the group key which is provided by the group manager and transmitted only once. Then the transmitted message can be decrypted by only group members having the group key. However, the group key is updated whenever the group membership changes for forward and backward secrecy, which can cause a serious problem with rekeying overhead. And key should be created for every file separately. So the communication and key overhead was high. The challenging problem is how to effectively share encrypted data with effective authentication and minimum key generation overhead. And several applications suffer from in efficient key authentication and key management problems.

Transferring the secret keys inherently requires a protected way, and storing these keys requires rather expensive secure storage. And the keys should be unique for every owner for a single file. The costs and complexities involved generally increase with the number of the decryption keys to be shared. In short, it is very heavy and costly to do that.

2.1 LITERATURE REVIEW

Data sharing is an important functionality in cloud storage [1]. For example, bloggers can let their friends view a subset of their private pictures; an enterprise may grant her employees access to a portion of sensitive data. The challenging problem is how to effectively share encrypted data. Of course users can download the encrypted data from the storage, decrypt them, then send them to others for sharing, but it loses the value of cloud storage. Users should be able to delegate the access rights of the sharing data to others so that they can access these data from the server directly.

Kallahalla et al. proposed [2] a cryptographic storage system that enables secure file sharing on un-trusted servers, named Plutus. By dividing files into file groups and encrypting each file group with a unique file-block key, the data owner can share the file groups with others through delivering the corresponding lockbox key, where the lockbox key is used to encrypt the file-block keys. However, it brings about a heavy key distribution overhead for large-scale file sharing. Additionally, the file-block key needs to be updated and distributed again for a user revocation.

In files stored on the un-trusted server include two parts: file metadata and file data. The file metadata implies the access control information including a series of encrypted key blocks, each of which is Encrypted under the public key of authorized users. Thus, the size of the file metadata is proportional to the number of authorized users. The user revocation in the scheme is an intractable issue especially for large-scale sharing, since the file metadata needs to be updated. In their extension version, the NNL construction [3] is used for efficient key revocation. However, when a new user joins the group, the private key of each user in an NNL system needs to be recomputed, which may limit the application for dynamic data in the entire data file shared by the company.

The changes of membership make secure data sharing extremely difficult. On one hand, the anonymous system challenges new granted users to learn the content of data files stored before their participation, because it is impossible for new granted users to contact with anonymous data owners, and obtain the corresponding decryption keys. On the other hand, an efficient membership revocation mechanism without updating the secret keys of the remaining users is also desired to minimize the complexity of key management. Several security schemes for data sharing on untrusted servers have been proposed. In these approaches, data owners store the encrypted data files in untrusted storage and distribute the corresponding decryption keys only to authorized users. Thus, unauthorized users as well as storage servers cannot learn the content of the data files because they have no knowledge of the decryption keys.

However, the complexities of user participation and revocation in these schemes are linearly increasing with the number of data owners and the number of revoked users, respectively. By setting a group with a single attribute, Lu et al proposed a secure provenance scheme based on the cipher text-policy attribute-based encryption technique , which allows any member in a group to share data with others. However, the issue of user revocation is not addressed in their scheme.

In the literature study we have seen many methods for secure data sharing in cloud computing, however most methods failed to achieve the efficient as well as secure method for data sharing for groups. To provide the best solutions for the problems imposed by existing methods, recently the new method was presented called MONA [4]. This approach presents the design of secure data sharing scheme, Mona, for dynamic groups in an untrusted cloud. In Mona, a user is able to share data with others in the group without revealing identity privacy to the cloud. Additionally, Mona supports efficient user revocation and new user joining. More specially, efficient user revocation can be achieved through a public revocation list without updating the private keys of the remaining users, and new users can directly decrypt files stored in the cloud before their participation. Moreover, the storage overhead and the encryption computation cost are constant. Therefore practically in all cases MONA outperforms the existing methods.

However as per reliability and scalability concern this method needs to be workout further as if the group manager stop working due to large number of requests coming from different groups of owners, then entire security system of MONA failed down. In revocation list the time given for each user is fixed after time expire user cannot access the data until group manager update the revocation list and give it to the cloud.

S. Kamara and K. Lauter [5] in this paper consider the problem of building a secure cloud storage service on top of a public cloud infrastructure where the service provider is not completely trusted by the customer. They describe, at a high level, several architectures that combine recent and non-standard cryptographic primitives in order to achieve our goal. Survey the benefits such architecture would provide to both customers and service providers and give an overview of recent advances in cryptography motivated specifically by cloud storage.

A. Fiat and M. Naor [6] they introduce new theoretical measures for the qualitative and quantitative assessment of encryption schemes designed for broadcast transmissions. The goal is to allow a central broadcast site to broadcast secure transmissions to an arbitrary set of recipients while minimizing key management related transmissions. They present several schemes that allow centers to broadcast a secret to any subset of privileged users out of a universe of size so that coalitions of users not in the privileged set cannot learn the secret.

V. Goyal, O. Pandey, A. Sahai, and B. Waters [7] they develop a new cryptosystem for One-grained sharing of encrypted data that call Key-Policy Attribute-Based Encryption (KP-ABE). In cryptosystem, cipher texts are labelled with sets of attributes and private keys are associated with access structures that control which cipher texts a user is able to decrypt. They demonstrate the applicability of our construction to sharing of audit-log information and broadcast encryption. Our construction supports delegation of private keys which subsumes Hierarchical Identity-Based Encryption (HIBE).

Yu et al. presented [8] a scalable and fine-grained data access control scheme in cloud computing based on the KP-ABE technique. The data owner uses a random key to encrypt a file, where the random key is further encrypted with a set of attributes using KP-ABE. Then, the group manager assigns an access structure and the corresponding secret key to authorized users, such that a user can only decrypt a ciphertext if and only if the data file attributes satisfy the access structure. To achieve user revocation, the manager delegate's tasks of data file reencryption and user secret key update to cloud servers. The data owner uses a random key to encrypt a file, where the random key is further encrypted with a set of attributes using KP-ABE. Then, the group manager assigns an access structure and the corresponding secret key to authorized users, such that a user can only decrypt a cipher text if and only if the data file attributes satisfy the access structure. To achieve user revocation, the manager delegates' tasks of data file re-encryption and user secret key update to cloud servers. However, the single owner manner may hinder the implementation of applications with the scenario, where any member in a group should be allowed to store and share data files with others.

Another concern is that the computation overhead of encryption linearly increases with the sharing scale.

Ateniese et al [9] leveraged proxy re encryptions to secure distributed storage. Specifically, the data owner encrypts blocks of content with unique and symmetric content keys, which are further encrypted under a master public key. For access control, the server uses proxy cryptography to directly reencrypt the appropriate content key(s) from the master public key to a granted user's public key. Unfortunately, a collusion attack between the un-trusted server and any revoked malicious user can be launched, which enables them to learn the decryption keys of all the encrypted blocks.

However, the single-owner manner may hinder the implementation of applications with the scenario, where any member in a group should be allowed to store and share data files with others. Lu et al. proposed a secure provenance scheme, which is built upon group signatures and cipher text-policy attribute based encryption techniques. Particularly, the system in their scheme is set with a single attribute. Each user obtains two keys after the registration: a group signature key and an attribute key. Thus, any user is able to encrypt a data file using attribute-based encryption and others in the group can decrypt the encrypted data using their attribute keys. Meanwhile, the user signs encrypted data with her group signature key for privacy preserving and traceability. However, user revocation is not supported in their scheme.

*3.*PROPOSED WORK

MOCK Scheme:

This introduces a new key aggregation scheme which is named as MOCK technique, which collects the keys from all owners and creates an aggregated key for data accessing. Unlike the previous works, the average size and time of rekeying messages have been avoided. So the communication overhead and time factors are considered here.

The proposal develops a five-step scheme for MOCK implementation.

* The first one is initial key creation for both single owner data and multi owner data group.
* The second step of the mechanism in MOCK is the creation of onetime encrypted key for data accessing.
* The third scheme is the MOR which is abbreviated as Multi Owner Rule specification.
* The fourth scheme is the crypto scheme which facilitates the encryption and decryption for data storage.
* The fifth step of the scheme is the mobile and session based authentication scheme, which helps to gather the encrypted keys from the device and aggregates together for data access.

Step 1: initial key creation
Step 2: **MD5**

Algorithm:-MD5

Description: processes a variable-length message into a fixed-length output of 128 bits.

STEPS:

Step1: The input message is broken up into chunks of 512-bit blocks; the message is padded so that its length is divisible by 512.

Step2: The padding works as follows: first a single bit, 1, is appended to the end of the message.

Step3: This is followed by as many zeros as are required to bring the length of the message up to 64 bits less than a multiple of 512.

Step4: The remaining bits are filled up with a 64-bit integer representing the length of the original message, in bits.

Step5: The MD5 algorithm uses 4 state variables, each of which is a 32 bit integer (an unsigned long on most systems). These variables are sliced and diced and are (eventually) the message digest.

The variables are initialized as follows:
A = 0x67452301
B = 0xEFCDAB89
C = 0x98BADCFE
D = 0x10325476.

Step6: Now on to the actual meat of the algorithm: the main part of the algorithm uses four functions to

thoroughly goober the above state variables. Those functions are as follows:
F(X,Y,Z) = (X & Y) | (~(X) & Z)
G(X,Y,Z) = (X & Z) | (Y & ~(Z))
H(X,Y,Z) = X ^ Y ^ Z
I(X,Y,Z) = Y ^ (X | ~(Z))
Where &, |, ^, and ~ are the bit-wise AND, OR, XOR, and NOT operators
Step7: These functions, using the state variables and the message as input, are used to transform the state variables from their initial state into what will become the message digest. For each 512 bits of the message, the rounds performed (this is only pseudo-code, don't try to compile it)
After this step, the message digest is stored in the state variables (A, B, C, and D). To get it into the hexadecimal form you are used to seeing, output the hex values of each the state variables, least significant byte first. For example, if after the digest:
A = 0x01234567;
B = 0x89ABCDEF;

4. Results And Discussion

The proposed work is successfully implemented using C#.net. The performance of this proposed work MOCK using aggregation scheme and new crypto scheme is compared with the existing approaches. The figure below shows the configuration of the system requirement
- Visual studio.Net 2010
- C#.net
- Sql server

The results prove the proposed system is outperformed than the existing techniques. This considered the verification delay and aggregation key creation delay for deployed data on the cloud in the process of retrieval. Encryption and key generation and verification delay are specified below.

Encryption Process	Data size (512 bits)	Data size (1024 bits)
Existing System	8.4	14.9
Proposed system	3.6	6.7

Table 1: Comparison of Data Size

The above table [1] compares existing and proposed methods in the form of data size. The encryption delay has been compared. The existing system need more time to perform a 512 bit data. This is very high when comparing with the existing system.

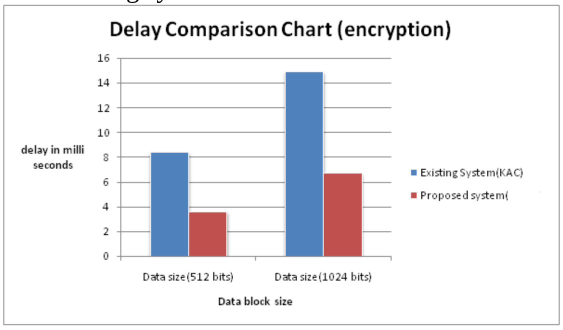

Fig 2. delay comparison chart

The above delay comparison chart indicates the execution time for the algorithm to produce cipher texts and corresponding keys before storing the data.

Key verification Process	owner size (2 users)	owner size (4 users)
Existing System	32.4	78.9
Proposed system	31.6	69.7

Table 2: Key Verification Process

Table [2] compares existing and proposed methods in the form of key verification process. The key verification delay has been compared. The existing system need more time to perform a 2 users data. This is very high when comparing with the existing system.

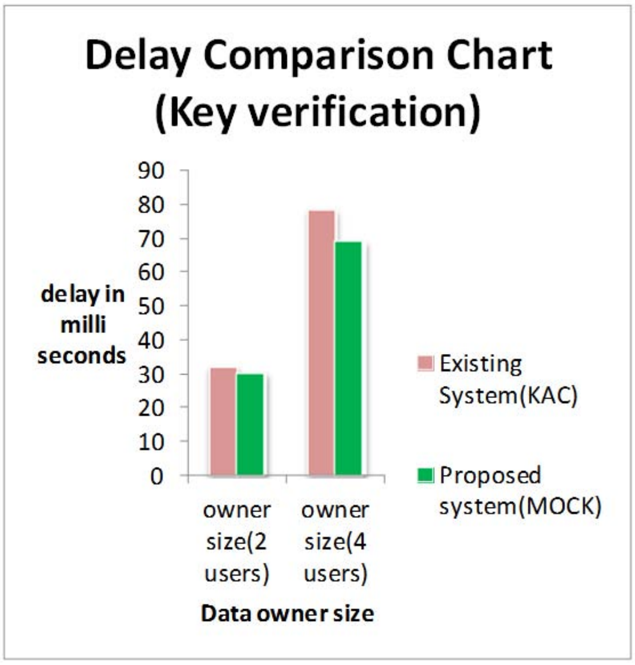

Fig 3: Key verification delay comparison chart

The above delay comparison chart indicates the execution time for the algorithm to produce normal texts from the cipher data and corresponding keys before storing the data.
Aggregation delay is also evaluated by measuring time spent on processing time on aggregating cipher texts and hash in the proposed schemes. The last delay is decryption delay, which is not considered in the server side. Therefore, this delay is negligible and can be ignored. Another criterion is cost evaluation. Cost evaluation involves aggregation and key generation and computation aspects.

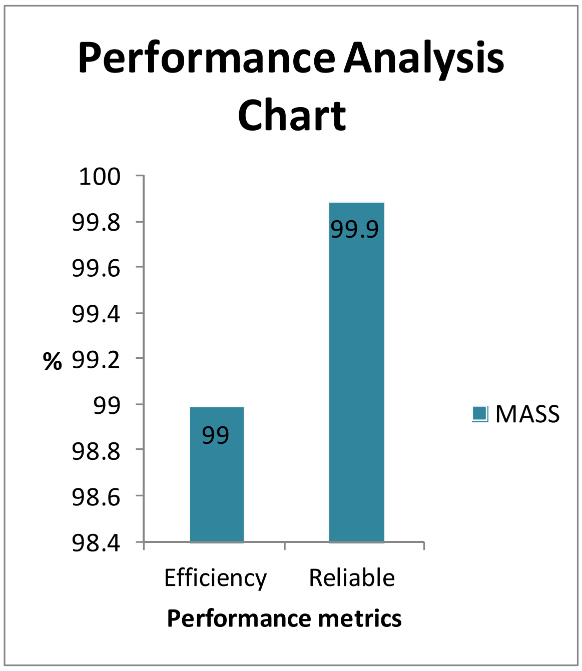

Fig 4: performance analysis chart

The above performance analysis chart indicates the execution time for the MOCK scheme with the performance parameters of efficiency and reliability. The proposed system yields maximum point for trustworthiness.

5. Conclusion And Future Work

The distributed systems are very useful and attractive environment for data sharing in term of providing required services in a very cost effective way to their clients. Still the distributed authentication system affects by the un trusted host access and several network attacks. The user need to send authenticated key via an un trusted hosts every time. To overcome the above issue and enhancing security and privacy practices in distributed systems, the MOCK protocol has been proposed with a new innovative functions and mechanisms.

In this scheme, we proposed a new user authentication protocol named MOCK which leverages cellphones and SMS to stop key stealing and key misuse attacks. The current proposal named as MOCK presenting the new encryption and authentication framework for multi owner data security on the distributed networks.

The system may extend with some other changes in the proposed MOCK in order to reduce the authentication collection risk. In the future, the method further may include the risks like failure of owner authentication. Extensive analyses show that the proposed "MOCK" scheme satisfies the desired security requirements and guarantees efficiency as well. Here we also show that how user gets extra time even after the time out this also one of the advantage of proposed schema.

Referneces

[1] Chu, C., et al. "Key-Aggregate Cryptosystem for Scalable Data Sharing in Cloud Storage." (2014): 1-1.

[2] Kallahalla, Mahesh, et al. "Plutus: Scalable Secure File Sharing on Untrusted Storage." Fast. Vol. 3. 2003.

[3] E. Goh, H. Shacham, N. Modadugu, and D. Boneh, "Sirius: Securing Remote Untrusted Storage," Proc. Network and Distributed Systems Security Symp. (NDSS), pp. 131-145, 2003.

[4] Liu, Xuefeng, et al. "Mona: secure multi-owner data sharing for dynamic groups in the cloud." Parallel and Distributed Systems, IEEE Transactions on 24.6 (2013): 1182-1191.

[5] S. Kamara and K. Lauter, "Cryptographic Cloud Storage," Proc. Int'l Conf. Financial Cryptography and Data Security (FC), pp. 136- 149, Jan. 2010.

[6] A. Fiat and M. Naor, "Broadcast Encryption," Proc. Int'l Cryptology Conf. Advances in Cryptology (CRYPTO), pp. 480-491, 1993.

[7] V. Goyal, O. Pandey, A. Sahai, and B. Waters, "Attribute- Based Encryption for Fine-Grained Access Control of Encrypted Data," Proc. ACM Conf. Computer and Comm. Security (CCS), pp. 89-98, 2006.

[8] Yu, Shucheng, et al. "Defending against key abuse attacks in KP-ABE enabled broadcast systems." Security and Privacy in Communication Networks. Springer Berlin Heidelberg, 2009. 311-329.

[9] Ateniese, Giuseppe, et al. "Improved proxy re-encryption schemes with applications to secure distributed storage." ACM Transactions on Information and System Security (TISSEC) 9.1 (2006): 1-30.

[10] Lu, Rongxing, et al. "Secure provenance: the essential of bread and butter of data forensics in cloud computing." Proceedings of the 5th ACM Symposium on Information, Computer and Communications Security. ACM, 2010.

[11] C. Wang, Q. Wang, K. Ren, and W. Lou, "Privacy-preserving public auditing for data storage security in cloud computing," in *InfoCom2010, IEEE*, March 2010.

[12] C. Wang, S. S.-M. Chow, Q. Wang, K. Ren, and W. Lou, "Privacypreserving public auditing for secure cloud storage." Cryptology ePrint Archive, Report 2009/579, 2009. http://eprint.iacr.org/.