Journal of Theoretical and Applied Information Technology
31st October 2020. Vol.98. No 20

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

EFFICIENT MUTUAL USER AUTHENTICATION PROTOCOL TO SHARE FILES USING ID IN CLOUD STORAGE

K. SUJATHA[1] , V.CERONMANI SHARMILA[2]

[1] Research Scholar – CSE, Hindustan Institute of Technology and Science, Chennai, India.

[2] Associate Professor-IT, Hindustan Institute of Technology and Science, Chennai, India.

[1]sujeekevi@gmail.com, [2]csharmila@hindustanuniv.ac.in

ABSTRACT

Transpiring trends inq the world nowadays is cloud computing as a predominant interactive archetype for data to ensure user's data in an online cloud server which is stored remotely. A cloud service provides tremendous conveniences to the users to relish the on-demand cloud applications without any limitations considered in the local infrastructure. In the process of data accessing, different users may be in a collaborative relationship, and thus data sharing becomes significant to achieve productive benefits. In recent days, with the adaptation of cloud services, many users are involved in it. But they are afraid about cloud environment due to security in cloud. In this paper, we proposed a Mutual User Authentication Protocol (MUAP) in which the users can safely share their data in a suspicious cloud servers. This MUAP provides assurance on data over user security and focussed mainly on data sharing among the users by an encrypted way. The data is encrypted and shared on public or private cloud servers due to that the manipulation of information is prohibited to the server. Users can easily spot the particular files from the cloud storage but the server cannot realize the identity of the user which leads that users can connect themselves to the cloud without endanger their privacy. Identity Issuer will keep user's privacy safely. The protocol encrypts all information to get rid of MITM attack. Mutual Authentication is achieved by two level encryption using ID. This protocol deals with individual activities and communications when the legacy protocols deals with group activities. When compared to other protocols the proposed system has higher security with minimal computational and communicational overheads.

Keywords: *Cloud Computing, Sharing Files, Cloud Services, Mutual Authentication Protocol, Insecure Cloud Servers.*

1. INTRODUCTION

Over the last few years, cloud computing is a promising information technology architecture for both individuals and business franchises. Cloud computing provides a stunning storage space for data and standards with palpable assets, including on-demand self services, pervasive network access and location independent resource pooling.

Generally the term cloud is referred to as a set of software, hardware, storage and network which can be delivered as a service. Cloud computing otherwise On Demand Computing is an online usage of resource on demand.

1.1 Types Of Cloud

In Cloud Computing the four types of cloud are Public cloud, Private Cloud, Hybrid Cloud and Community Cloud which are involving for various purposes.

1.1.1 Public cloud

The Cloud Infrastructure is available to public and the processes are performed by large organizations, Governments or both. It reduces customer risks and costs by providing flexible infrastructure. These clouds have more scalability and low capital costs.

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

1.1.2 Private cloud

The Cloud Infrastructure is owned or used by one organization. It may be deployed in an enterprise data centre or at a collocation facility. It provides limited access to its resources for their consumers. It has highest level of security and control level.

1.1.3 Hybrid cloud

It is a computing environment which combines both public and private clouds. It can be used to interact with customers by public cloud and secure the data of the customers by private cloud. It can be used to manage planned workload spikes. It introduces the complexity of determining the distribution of applications among the public and private clouds.

1.1.4 Community cloud

It is an effort of collaborative way in which a specific community's several organisations are sharing the cloud infrastructure. It is costly and has higher level of security. It is managed by 3rd party or internal sources. It is hosted by internally or externally.

The typical service architecture in cloud computing are IaaS, SaaS and PaaS. Continuing that, recently the cloud computing is promoted to evolve the internet of services. With the increasing popularity of Cloud Services, the major key concerns are Security, Privacy and User Authentication.

Existing approaches focuses on the Strong Remote User Authentication. Along with the diversity of the application requirements, sometimes users may want to migrate on other's authorized data fields. It leads to the need of more security and privacy in Cloud Storage.

Mutual Authentication is the security feature which allows both entities of communication link authenticate to each other by their identity. A well designed Mutual authentication helps to prevent various online attacks such as, Trojan Horse, Data Confidentiality, Replay attack and Man In The Middle attack.

1.2 Benefits Of Cloud Storage

Why the cloud storage is such a big deal to large companies. While storing their files in cloud storage there are obvious benefits rather than risks involved in cloud storage. The general benefits of Cloud Storage are Scalability, Reliability and Low Costs.

1.2.1 Scalability

It is the capability of a network, process, software to grow and manage increased demands. By this, we can scale up the data storage capacity or scale it down to meet the demands. Scalability measured by two characteristics of cloud that are, broad network access and resource pooling. Broad network access allows the customers to access the services via heterogeneous client tools. Through multitenant model, multiple customers are served by pooling computing resources.

1.2.2 Reliability

Cloud Storage is much more reliable when used in tandem with another storage system. The biggest concern in cloud storage is about the lost data. It is eliminated if the cloud is used more as a sharing instead of a storage platform. The unlimited capabilities are available at any time and can be purchased countless is called rapid elasticity which is the most important characteristics of cloud computing.

1.2.3 Low Costs

Cloud storage services can offer lower storage rates because they use the server space efficiently and gets reassigned to users almost instantly on demand basis. On-Demand self service and measured services are the cloud characteristics related to costs. The resource usage by provider and customers will be metered, monitored, controlled and reported.

Let us know the risks of storing data in the clouds. First of all, even though reliability is considered as benefit, which is also the major key in risks. When the cloud storage service does not have sufficient infrastructure ot does not contains multiple backups then the data will be at risk. If some vendor discontinues due to legal or financial problems then the data will lost for customers.

Secondly, in terms of security, still some high tech breachers having possibility to break the system and perform some manipulation on sensitive

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

data. The data on the cloud can be accessed by anyone from any location. In this situation, sensitive data cannot be differentiated from common data by the cloud. Due to flexibility and cost effective some of the cloud vendors' attempts to lease a server from other service providers will lead to data theft in the cloud.

Thirdly, some simple mistakes of human beings are also victims that cause risks over the user data. It includes the operations such as exposing data to unauthorized users or permanently deletes data which does not need to do.

Finally, accessing the data over an internet is also a problem when the connection is not available. Access to clouds restricted due to latency in data access and slow connections.

In early stages, Network Attached Storage and Network File System (NFS) are the extra storage devices. By the way, storage devices can be accessed by user via network connection.

Normally, the security protocol should achieve some requirements like Authentication, Data Anonymity, User Privacy and Forward Security in the Cloud Environments.

Authentication: Data fields in the storage can be accessed only by a legal user. Any other tampered and forged data fields cannot swindle the authorized users.

Data Anonymity: Data Anonymity involves in the process of removing or encrypting identifiable personal information from the data fields.

User Privacy: User's information like access desire cannot be guessed by any other irrelevant entity. Sometimes both the users can access each other's data fields those are authorized when they have mutual interest. Then the Cloud server will notify the access permissions to both the users.

Forward Security: To obtain the prior probing related to the currently tampered messages, any irrelevant compromise the two communications sessions.

Let us have a look on a company having a private cloud infrastructure with enormous benefits to their employees. The employees are allowed to share and store their data fields in the cloud. Most of their files will conceive their secret data of the company.

From the user's point of view, it is not safe and trustworthy in cloud storage. To obtain more security in the cloud storage, the files are encrypted and uploaded on the servers. But designing a trustworthy scheme is not a easy task because of the reasons discussed below.

First of all, Privacy Preserving is major barrier to expand cloud environments. Users are not interested to migrate in cloud environments, since the intruders can easily outsource their identity and privacy. On the other hand, the intruders may share the false information if we do not preserve original user's data. So we have to ensure that, user's identity and privacy for the safety of cloud.

Next, some of the existing schemes [2],[3] that allows just one user to manipulate the files for simplicity. But it is not considered as a effective logic for simplicity. The system should allow all the users to access their files freely. It is referred as multi tenant cloud.

Recently, the researchers are concentrating on implementation of groups. The group head will be the responsible person to identify the intruders and to manage the group operations. To concentrate on this process, we should consider for a dynamic groups.

In dynamic group, the users can be either sacked user or a new user. These users often change their group state because of that construction of the dynamic group with trustworthy is difficult. The main challenge of the system is related to access of legacy files by a new user. But it is a barrier for the new user to access the files which is uploaded earlier before they joined the group.

The next challenge of the system depends on the access permission of sacked user. After excising the user's access permission, there is a need to change the keys and distribute among the legal users of the group. By changing keys, the entries to the old groups are prohibited for the sacked user. To make the group efficient we must do something without changing keys.

An example is mentioned below to know the main motivation about security protection and privacy preservation. In supply management process, there are some groups like Provider,

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

Transporter and Vendor in the system. These groups have its own users who are allowed to access the authorized data and relatively independent access by different users.

Sometimes a provider wants to access a transporter's data, but cannot assure about the transporter state. So this access depends on Transporter decision and it is nothing to reveal the Provider's private data without any considerations on privacy. Some of the possible situations are discussed below under privacy issues.

Situation 1: The Transporter is interested to access the provider's data fields and vice versa. Depends upon the situation if they have mutual interest on other's data then the cloud server is responsible to share the access authority.

Situation 2: The Transporter does not want to access provider's data field's even though the request is made from provider. At the same time the transporter does not want others to have access on their own data. Due to that its authorized data is protected and provider's access request is concealed.

Situation 3: The provider makes the request to access the Transporter's data fields. But the transporter is interested on the data fields of vendor. This leads the request from provider will be rejected and the server will response to the provider by protecting privacy. Depends on the vendor's decision, the server will response to the transporter. The request from the Transporter and the data fields are protected if the access is denied from the vendor. Meanwhile the vendor's data fields are also not revealed.

Figure 1 depicts some of the situations of privacy issues.

While considering attacks on cloud, Man in the Middle (MITM) attack is the common attack. In this attack, the intruder will disconnect the connection between cloud users and cloud server. Then, he will connect himself with the user and the server. While making request the intruder will act as intermediate between cloud user and the server. But both the parties are not aware of the intruder. The intruder will pretend as an authorized party and eavesdrops the information easily.

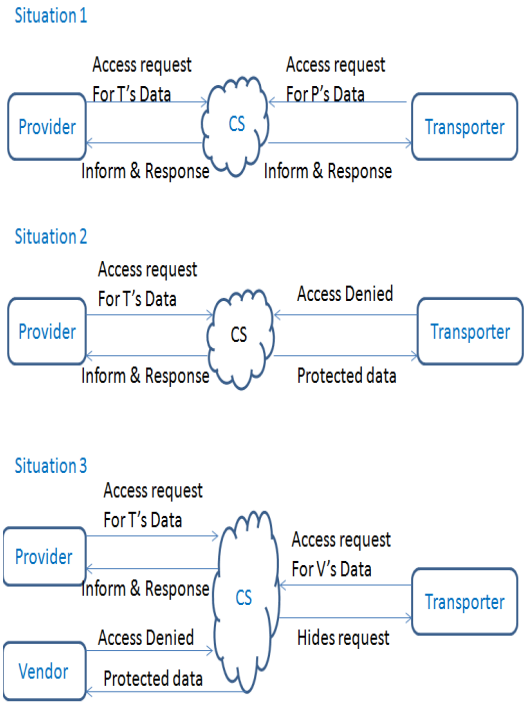


Figure 1: Possible Situations Of Privacy Issues

To strengthen the security protection in cloud applications, the researchers have been worked on security architectures [3][4], data public auditing protocols [5][6][7], data possession protocols [8][9], secure data storage and data sharing protocols [10]-[15], privacy preserving protocols [24]-[27], access control mechanisms [28]-[30], and key management [31]-[34].

2. RELATED WORKS

In this section, some of the legacy works has been discussed related to the proposed authentication protocol for file sharing.

Dunning et al. [10] proposed a Data Sharing Algorithm by using Anonymous ID Assignment (AIDA) to enable multiparty oriented cloud. It works also on Distributed Computing Systems. In this AIDA, for data mining operations they implemented an integer data sharing algorithm. Especially, they implemented Newton's identities and Sturm's theorem to enhance the scalability of algorithm over finite fields.

Liu et al. [11] proposed a data sharing scheme for multiple owners (Mona). It focuses on dynamic groups in cloud application. In this scheme, data is

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

shared securely among the users from any user via the suspicious cloud servers. This scheme supports dynamic group interactions efficiently. Mona has reduced storage overhead and computation complexity with less communication costs. Though it has several advantages still lacks in cost of computation when there are many groups. It is focussed on group activities rather than individual sharing.

Grzonkowski et al. [12] proposed an authentication scheme based on the theory Zero Knowledge Proof (ZKP). The general procedure of ZKP is if a user wants to communicate with other user means then the user has to prove their identity to the other user through some secret information. But, there is no need to reveal this secret to the other user. Figure 2 represents the basic function of Zero Knowledge Proof (ZKP) Authentication Scheme.

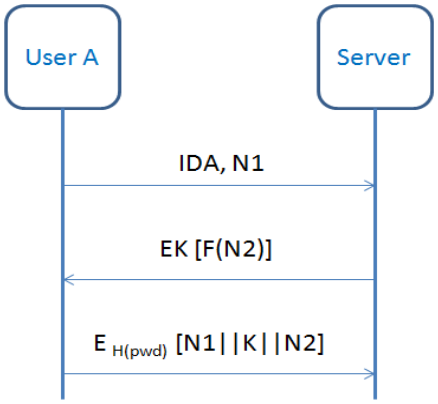


Figure 2: Basic Function Of ZKP Scheme

Notations:

A - User A

S - Server

K - Secret Key

N1, N2 - Nonce

F - Transformation Function

EK - Encryption using K

H[pwd] - Hash function of Password

This scheme allows authentication without the use of password. It avoids stolen verifier attack because the password in the server is in encrypted format. It avoids replay attacks by using nonce values. The security of the protocol is depends on the strength of an encryption algorithm. Though it has several advantages, still suffered in some attacks. When an attacker targets a specific server and user name, they could precompute the values of public key for a list of common passwords. It occurs because the server does not use the salts.

Nabeel et al. [13] proposed a Broadcast Group Key Management (BGKM) which implements dynamical derivation of symmetric keys without the need of public key cryptography during decryption. In this scheme, attribute based access control mechanism and fine grained algorithm are designed. The attribute access control mechanism allows the user to decrypt the contents if and only if the policies of content provider are satisfied by its identity attributes. Fine grained algorithm involves access control vector which benefits users, based on their identity attributes secrets are assigned to them.

It has an evident benefit on manipulation of annulled users or adding new users and amending the policies of access control. It has only one drawback which is, focussed mainly on groups but the advantages are more such as trustworthy, key hiding, key independence, forward security, backward security, collusion resistance, reduced computational costs, minimal storage requirements and ease of maintenance.

Wang et al. [14] proposed an auditing mechanism which enhances trustworthy and reliable storage services. In this mechanism, distributed erasure coded data and homomorphic token are introduced to maintain the storage services in cloud computing. In this scheme the cloud storage is audited by the users. This auditing assures lightweight communication overloads and less computation costs. The result of the auditing assures correctness of strong cloud storage and error localization of fast data. It supports outsourced data operations dynamically. This scheme is resilient against some attacks such as byzantime failure, server colluding attack and data modification attack.

Sundareswaran et al. [15] established a decentralized information accountability framework. It is designed to monitor the usage of user's actual data. This framework enables the enclosure of logging mechanisms with the use of

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

user's data and policies. To achieve authentication with user's data access, a dynamic and mobile object is created by Java ARchieves (JAR). User's data control and the approach is efficiently nourished by distributed auditing mechanisms.

Kallahalla et al. [16] proposed cryptographic storage system named as 'Plutus'. The users can upload their files on suspicious servers securely. The general procedure of the plutus is, some "file block keys" are used to encrypt the grouped files. Only the legal users can get the "lock box keys" from the owners of the files. The "File block keys" is encrypted by the "lock box keys".

Plutus is considered as a comprehensive, secure and efficient file system. It has superior scalability with stronger security. Though the less key maintenance is achieved by grouping files, it is a great overhead for the system. If any one of the key is lost then many files would be stolen.

Another main overhead is after every revocation of the user, the "file block keys" should be updated and distributed among the legal users. On the security side, it does not overcome the Man in the Middle attack. Another problem is that the owner should be in online forever because they need to distribute the keys.

Ateniese et al. [17] proposed a Proxy Re-encryption Scheme to construct a safe distributed cloud.Primary encryption is done by symmetric key and encrypted again with the master public key for every block of data. It focuses only on re-encryption of files and not interested in revealing contents. The drawback of the system is caused by revoked user. Sometimes to obtain the decryption key they can collude with untrusted server.

Figure 3 represents the procedure of file transfer between two parties.

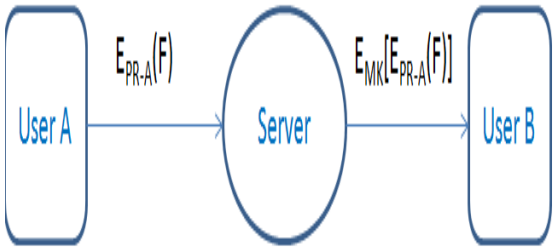


Figure 3: File Transfer Between Two Parties

Notations:

E_{PR}-A - Encryption using user A's private key

E_{MK} - Encryption using Server's Master key

F - File that is to be transferred

Hong liu et al. [18] proposed a Shared Authority Based Privacy Preserving Authentication Protocol in Cloud Computing. It focuses on the access control and access authority sharing of data rather than the particular file based cloud data transmission and management.

In the process of cloud data interactions the system covers the feasible security related threats and vulnerabilities. Still it suffers from computational cost overhead. If both parties are interested on other's data fields then the request has to be made from the other's side considering upon the mutual desire.

Figure 4 represents one sample request of data fields between two parties in two different situations.

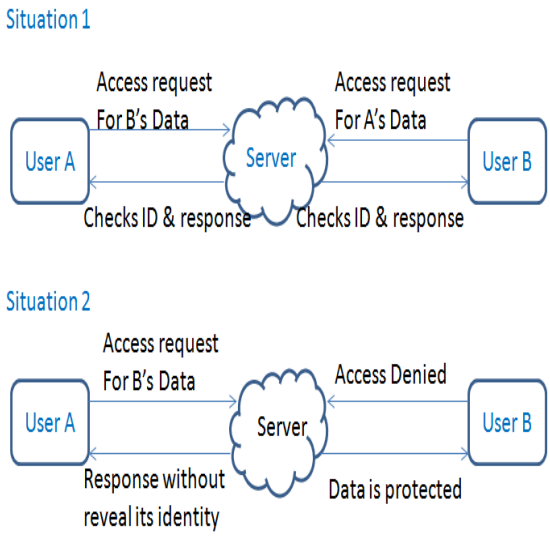


Figure 4: Data Request On Mutual Desire And Denial Of Access.

Yang et al. [19] proposed a Public Audit System to share files. There are five entities implemented such as Group Manager, Group Member, Cloud, Area Member and TPA. Group manager is responsible to manage the members ID and help them to make changes on the files to be

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

shared. Area Manager is intended for group operations and handling user's private keys. The two units of group keys are distributed among the group manager and the user respectively.

This scheme mainly focuses on uploading files on cloud. It ensures the integrity while in the process on cloud. When the scheme focuses on uploading files but not focused on to download files.

The prime weakness of group based scheme is the files have to be shared among all the group members by the users. So the user cannot share their file with the specific member in the group. An attacker can obtain secret key of the user if the system does not use the safe key distribution channel. After owns the secret key of the user an attacker can use it to destroy the files.

Zhu et al. [20] proposed Data sharing using Secure Anti-Collision for dynamic groups in the cloud. It implements three entities such as Group manager, Cloud and Group member. Even though it is similar to Mona scheme, this data sharing is more fulfilled comparatively.

The main advantage of the system is that the user can directly contact the cloud instead of manager to access the file. It achieves secure key distribution and fine grained access control. Secure user revocation is achieved with the help of polynomial function. It is protected from collusion attack and supports dynamic groups efficiently.

Samanthula et al. [21] proposed a secure data sharing query processing framework for sharing files. This scheme is based on attributes and used Proxy Re-encryption with additive homomorphic features [21,22]. There is only one key for each user and each file which leads the cloud to re-encrypt the files. On the user's side, the private key is used for decryption. In the older version of this scheme used only one service provider. So that, it is vulnerable to the threat of collusion with provider. Later, in the enhanced version in 2016, two service providers are implemented so there is no chance of collusions by both the providers.

This scheme has the drawback of the need of many keys. If one user wants to share the files with 'n' users then the system need to generate 'n' keys. The implementation cost of two service provider is also high and communication overhead arises. The implementation of two service providers is susceptible to the system by user revocation attack.

Legacy systems are mainly focussed on group activities, computational and communication overhead, security issues and has own flaws in their way. So the proposed protocol for the authentication scheme is implemented to satisfy the flaws mentioned in the legacy systems.

3. PROPOSED METHODOLOGY

Many problems are listed and discussed in legacy schemes such as group activities, computational and communication overhead, user revocation, communication over unsecure servers, dynamic groups, and collusion and security issues. In our proposed protocol file transfer between parties by using IDI is efficiently implemented in order to overcome those mentioned shortcomings.

To remove the collusion two layer of security is used by some cryptographic technique. Communication and computational overhead is also reduced by the method of effective user revocation. Updating the keys only involved with requested user instead of all the users. This protocol allows the user to communicate with whoever they want to. Due to that individual activities are focussed rather than group activities.

3.1 General Framework

The proposed authentication protocol utilizes the concept of hybrid cloud. So the important files are stored in private cloud and common files are stored in public cloud. The proposed protocol involves three categories of members. They are server, user and Identity Issuer(IDI).

Server: It is responsible for user request on file transfer. It will check the identity of the user and granted the access immediately if the file is in public cloud otherwise the request is proceed to IDI. If the identity of the user is not matched then the access is denied. Less significant files are stored in the server. The keys of server are,

PU_S - Public key of Server

PR_S - Private key of Server.

User: Users are classified as normal users and owners. The users who are only accessing the files are known as normal users. The users who are

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

uploading and share their files are known as owners. Sometimes the normal users can also change into owners. Owners can use both private and public clouds. It has two keys.

PU_U - Public key of User

PR_U - Private Key of User.

IDI: It stores User's Identity, User Name and Access permission. IDI is responsible for collusion management. If any unauthorized user colludes with the server will be prevented by IDI. It has two keys.

PU_I - Public key of IDI

PR_I - Private key of IDI.

3.2 System Operations

Our system has high bold features to implement authentication protocol for file transfer using ID efficiently. It has some necessary steps to implement the protocol. The steps are given below and explained neatly.

3.2.1 User Enrolment
3.2.2 ACL Generation
3.2.3 Exporting Files and ACL
3.2.4 Files Amendment
3.2.5 ACL Amendment
3.2.6 Access of file
3.2.7 User Annulment
3.2.8 File storage
3.2.9 Access request

3.2.1. User enrolment

Users have to enrol themselves by sending user name to IDI if they want to access the cloud. IDI generates ID for that user and responds with a report message. Server received the list from the IDI which contains hashed user name and ID. The user's identity cannot be revealed to the server by getting only hashed user name. Figure 5 represents the process between user and IDI regarding user enrolment.

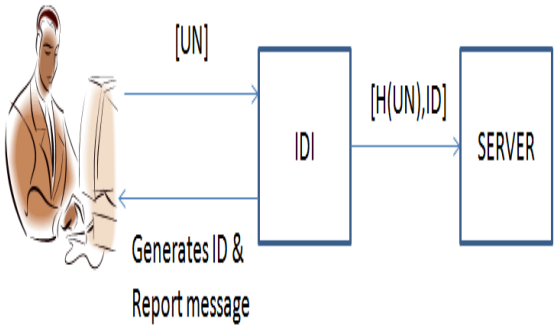


Figure 5: User Enrolment

Notations:

UN - User Name

H(UN) - hashes of User name

ID - ID created by IDI.

Table 1 contains user name and ID. It is generated by the IDI for its own verification.

Table 1: IDI List

User Name	ID
AAA	1
BBB	2
CCC	3

Table 2 contains Hashed user name and ID. It is generated by the IDI and sent to the server to verify the users with ID while in the process of access request from the user for file access.

Table 2: Server List Generated By IDI

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

User Name	ID
ec0a6f4527890vb4d334567dc41xy76a	1
ac2h2t324867gh2c821779bttu34ajk21	2
c4bcdb38f4220f8616319ea0ccd9bsusa	3

3.2.2. ACL generation

After enrolling themselves with IDI in the system they have to generate Access Control List (ACL). Owner of the file must create ACL for the file to be uploaded with the content such as users of the file, owner of the file, time and some indexed words for searching.

Owner is allowed to store their files either in public cloud or private cloud. ACL is generated and uploaded along with the signature of the user. Table 3 represents the example format of the ACL for the File A.

Table 3: ACL For The File A.

FILE A			
Users	Owner	Time	Indexed Words
ec0a6f4527890vb4d334567dc41xy76a	BBB		Cloud
			Mutual

3.2.3. Exporting Files and ACL

The preliminary step to upload ACL and files the owner must sign the ACL. Signing process is processed by RSA algorithm. It computes m^k(mod N) where 'm' is the ACL of the file and 'k' is the private key of the file owner.

If the owner chooses private cloud to store then the owner must sign the ACL and encrypt them. The signed ACL sent to the server and IDI. If the owner chooses public cloud to store the files then the owner has the responsibility to mail the signed ACL to the server.

3.2.4. Files amendment

The owner does not depend on the server for the updating of file when it is stored in private cloud. The owner has the rights to amend his files if it is in private cloud. If there is any need to modify indexed words, then the owner has to regenerate ACL and mail the updated ACL again to IDI and the server. Where as in public cloud, if the owner want to amend the files then he need to regenerate ACL again and send it to the server along with the signature.

3.2.5. ACL amendment

When the situation arises like user annulment the owner is the responsible to amend the ACL. The owner has to regenerate the ACL for the particular file with the corrections made. After generating corrected ACL, signing process begins by the owner using RSA algorithm. The signed ACL is sent to the server and IDI depends on the cloud they used.

3.2.6. Access of Files

The process of accessing the files which are stored in the cloud is divided into three steps.

3.2.6.1 User Request
3.2.6.2 Request to IDI
3.2.6.3 File Transfer

3.2.6.1 User Request

Server has its own user list which is sent by the IDI. It contains only hashed user name and their ID. First of all, the user sent the request to the server which contains hashed user name H(UN) and requested file name (FN). Server matches the request with the list for checking their ID. If the ID matches then it checks the freshness of the request with the timestamp specified in the ACL of that requested file.

If everything matches then the server will recognize it as a legal user otherwise the server denies the request. For the legal user, the server responds with the requested file if it is stored in public cloud. Suppose the file is in private cloud the request is forwarded to IDI encrypted with public key of IDI.

Figure 6 represents the flow of interactions between the user and the server.

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

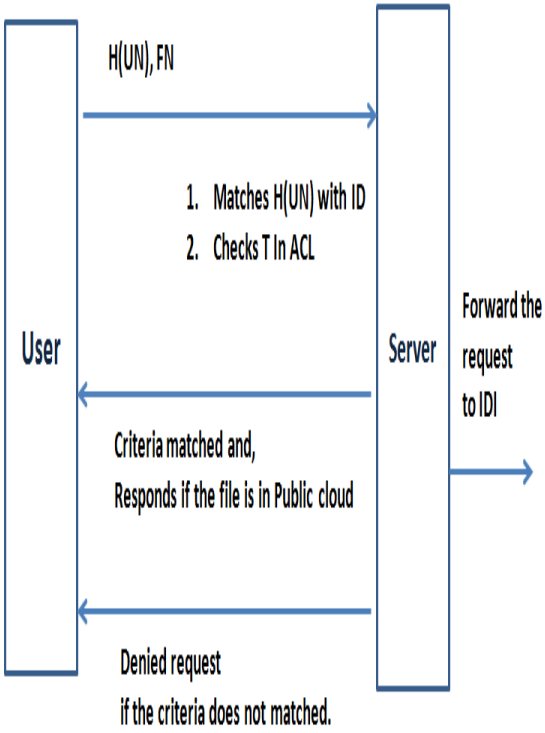


Figure : User Request

Algorithm for User Request:

Step 1: User sends request with H(UN), FN.
Step 2: if H(UN) matches,
Then Checks TS.
Else Abort Request.
Step 3: if Entry found,
Then checks for file status.
Step 4: if FN is found in public cloud
Then reply with the file.
Else Forward request to IDI.

3.2.6.2 Request to IDI

Server forwarded the request to IDI encrypted with the public key of IDI. It decrypts the request with its private key to obtain the user ID. It matches user ID with the list which is generated at the enrolment process.

If it matches then checks the freshness of the request with the timestamp which is presented in the ACL list of the requested file. When the user is confirmed as a legal user from the checking process by IDI, it ensures the evident for the no occurrence of collusion with the server.

If it succeeds then the request is forwarded to the owner and that is encrypted with the public key of the owner. IDI terminates the request if any of the criteria in the request does not match with the list.

Sometimes the user may attempt again and again to get the access of the file when IDI rejects their request. At that time IDI insists the server to include the specified user in the black list.

Figure 7 depicts the sequence of the process between the server and IDI.

Algorithm for Request to IDI:

Step 1: Encrypt the request with public key of IDI.
$E\text{-}PU_I[H(UN),FN]$
Step 2: forward the encrypted request to IDI.
Step 3: IDI decrypts the Request with its private key. $D\text{-}PR_I[H(UN),FN]$
Step 4: if H(UN) matches,
Then Checks TS.
Else Abort Request.
Step 5 : If TS matches,
Then encrypt the request with public key of file owner. $E\text{-}PU_O[H(UN),FN]$

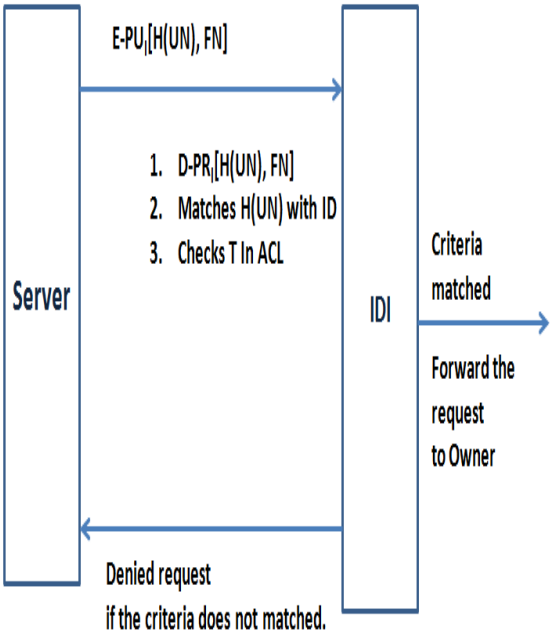


Figure 7: Request To IDI

3.2.6.3 File Transfer

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

IDI transfers the request which is encrypted with the public key of owner. Now the owner will decrypt the request with its private key to obtain the file information. The owner confirms the request is requested by the legal user due to the involvement of IDI.

The owner will encrypt the file with the new key or the old key because of the revoked user. In earlier schemes when there is any annulment we have to send the new key to all the n number of users in the cloud.

In our proposed scheme the owner will send the new or old key only to the requested user. By means of that communication overhead is mainly reduced.

The owner has to send the new key to the requested user to decrypt the file. In this stage, mutual authentication is achieved by encrypting the new key with its private key. Again it is encrypted with the public key of requested user and sends it to the user. Figure 8 represents the process of file transfer from owner to the user.

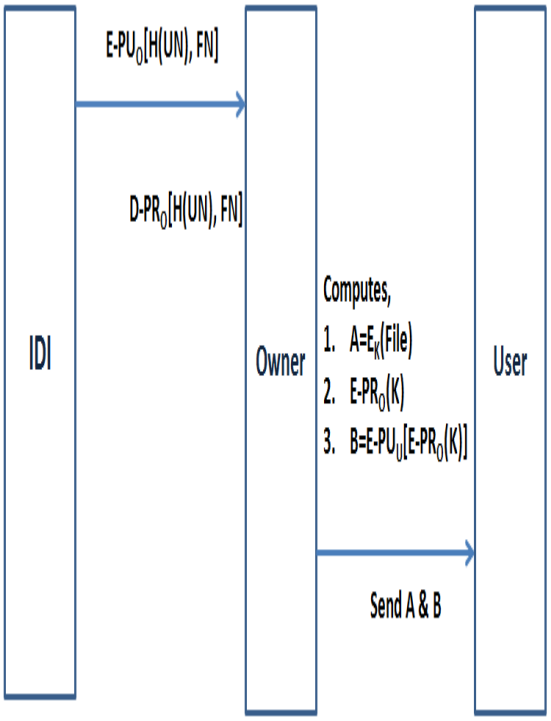


Figure 8: File Transfer

Algorithm for File Transfer:

Step 1: IDI encrypts the request with the public key of data owner. E-PU$_O$[H(UN),FN]
Step 2: encrypted request sent to the owner.
Step 3. Data Owner decrypts the request with its Private key. D-PR$_O$[H(UN),FN]
Step 4: Owner computes the encryption of file with Old or new key. A=E$_K$(File)
Step 5: Owner encrypts that old or new key with its Private key. E-PR$_O$(K)
Step 6 : Encrypted key is again encrypted with Requested user's public key. B=E-PU$_U$(E-PR$_O$(K))
Step 7: data owner sends A and B to the Data Requester.
Step 8: Data requester decrypts with its private key And public key of IDI respectively to get the new or old key.
Step 9: Using that key, data requester decrypts the encrypted file.

Let us take glimpse of notations used in file transfer phase.

Notations:

H(UN) - Hashed User Name

FN - File Name

E-PU$_I$ - Encryption using Public key of IDI

D-PU$_I$ - Decryption using Private key of IDI

E-PU$_O$ - Encryption using Public key of Owner

D-PU$_O$ - Decryption using Private key of Owner

EK - Encryption using Old or New key

E-PU$_U$ - Encryption using Public key of User

T - Timestamp in ACL

ACL - Access Control List

3.2.7. User annulment

The owner is the responsible person to update Access control list of the file to be accessed whenever the user is revoked. He will update about the revoked user to the server and IDI.

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

3.2.8. File storage

Public cloud has high capacity, enormous high speed, robust servers and less security. If the owner relies on public clouds then he need to generate ACL for the particular file and sign with RSA algorithm. Finally he sends this signed ACL to the server.

Private cloud has low capacity, low speed when compared to public clouds and more secure than public clouds. When the owner relies on private cloud, he needs to generate the ACL for the particular file and sign with RSA algorithm. Then he sends the signed ACL to both the server and IDI.

3.2.9. Access request

In some situations the owner would be in the crisis to make some inexperienced faults in generating ACL such as without including the users and information about the legal users. There might be sometimes the user may look upon the file which is stored in different clouds.

At that time the user can make the request to the server. Next, the server will redirect the request to the file's owner. When the owner enters online he may accept or reject the request. Even though the owner rejects the request the user can pursue its request. If this situation happens continuously then the owner will blacklist the user and insists the server to follow.

Regarding this continuous request attack by the user it depicts the situation of Denial of service attack. This makes pressure to the server which affects the quality of the services provided by the owner and makes the system down. To avoid such an attack the owner is responsible to block the user who may cause problems.

4. EVALUATION OF PROPOSED SCHEME

The proposed system is implemented by front end by angular, mongodb for storage Node JS in the server side and uses the AWS cloud computing environment.

The proposed system involves the AWS cloud products for security, storage, and database such as Amazon redshift, Amazon workdocs, Amazon simple storage service, Amazon Inspector, Amazon Identity and access management, Amazon cognito and Amazon Guardduty.

Amazon Redshift is a fast and scalable data warehouse. It delivers high faster performance by using machine learning. It is simple and cost effective when compared to other data warehouses.

Amazon workdocs is a rich API which offers easy collaboration with others and easily shares content. It is used to create, edit and share contents on the cloud. It is a fully managed, storage, secure content creation and collaboration service.

Amazon simple storage service is an object storage service which offers mainly scalability, data availability, performance and security. It provides easy to use management features. Amazon Cognito involves in authentication process such as user sign in and sign up. The users can sign in to the cloud by using their social mediums.

Amazon Guardduty is used as an intelligent and cost effective for threat detection in the cloud. It does not require any software and hardware to deploy. Though it is time consuming the process is simplified to follow.

Amazon identity and Access management involves in creation and management of individual users and group of users. It is used for access control to the resources. Amazon Inspector is an automated security assessment. This product is used to improve the security of the applications.

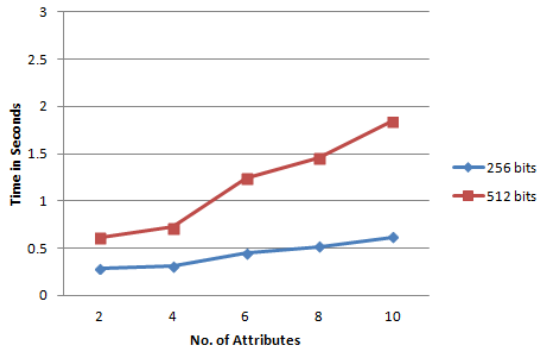
To evaluate the performance of our algorithm, we have taken a desktop machine having an Intel Core i5 processor with 3.80 GHz and 6 MB cache in Windows 7 Enterprise. To evaluate the cost computation, upload and download time, the system is used as remote machine and virtual machine on Amazon EC2. For our experiment, the cryptography part is based on java pairing based Cryptography is used over 256 or 512 bits. The results taken over here are average values of 1000 runs.

4.1 Performance Of Data Requester

During the file request stage, the data requester has to encrypt the request with salt using his private key. The computation time to encrypt the request is calculated for different field size and random number. The computation cost to encrypt the request is being calculated over number of attributes which are accessed by the data requester. The following Graph.1 shows the computation time ranges from 0.257 seconds to 0.605 seconds for the

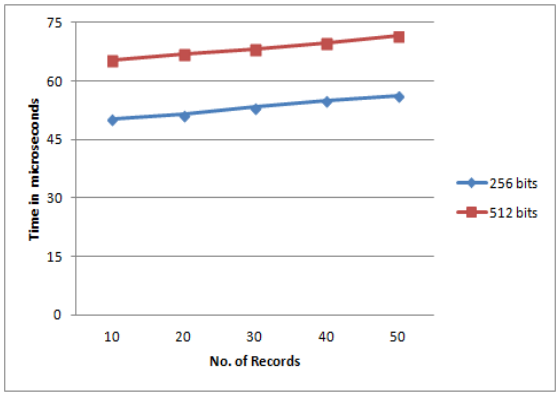
ISSN: 1992-8645 www.jatit.org E-ISSN: 1817-3195

field value 256 bits. Respectively the value ranges from 0.635 seconds to 1.856 seconds for the field value 512 bits. According to the size of the field the computation time is also doubled.



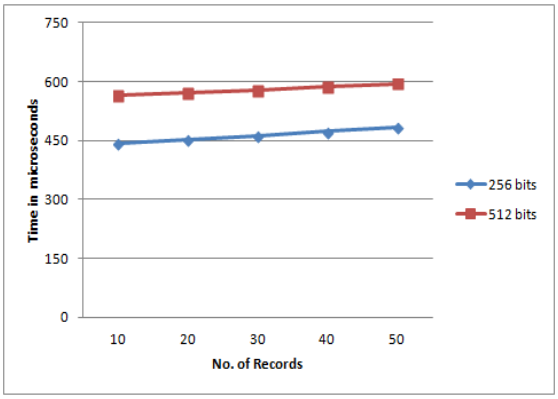
Graph 1: Computation Cost To Encrypt The Request

At the final stage of file request, the data requester has to download the file from either an IDI or the cloud. For that, the system is deployed under two different settings. The data requester's download time on the remote machine varies between 50.45 microseconds to 55.21 microseconds for 10 to 50 records having 256 field sizes. For 512 field sizes, it varies between 63.26 microseconds to 68.91 microseconds for 10 to 50 records. When the field size is doubled, there may be a little hike on starting range but, maintains the consistency over the period values. The following Graph.2 shows the graphical representation of the download time of the records.



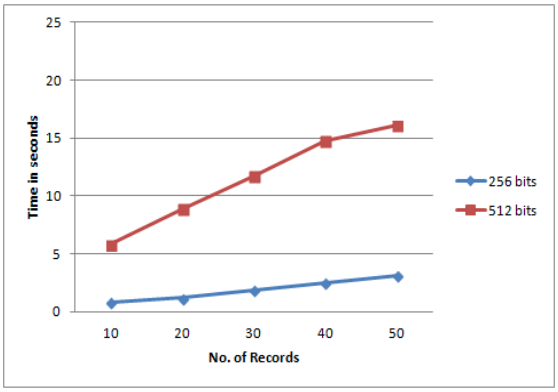
Graph 2: Data Requester's Download Time On A Remote Machine.

Under another system setting as cloud on Amazon EC2, The data requester's download time with the cloud setting varies between 442.43 microseconds to 482.86 microseconds for 10 to 50 records having 256 field sizes. For 512 field sizes, it varies between 565.08 microseconds to 595.10 microseconds for 10 to 50 records. May be, other loads on the cloud leads a little hike on the download time when compared to the remote machine. The following Graph.3 shows the graphical representation of the download time of the records.



Graph 3: Data Requester's Download Time On The Cloud

At the final stage of file request, the decryption time has been calculated for the files which are sent from the cloud to access. The decryption time of the data requester is linearly growing with the increase of records. For the field size 256 bits, it ranges from 0.8 seconds to 3.1 seconds. Respectively for the field size 512 bits the decryption time increases gradually from 5.8 seconds to 16.1 seconds. Graph.4 shows the graphical representation of the decryption time.

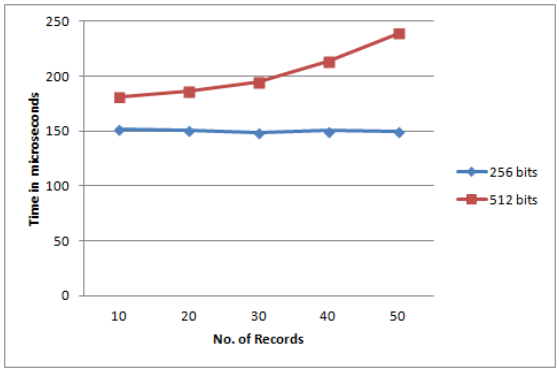


Graph 4: Data Requester's Decryption Time Over Number Of Records

4.2 Performance Of Data Owner

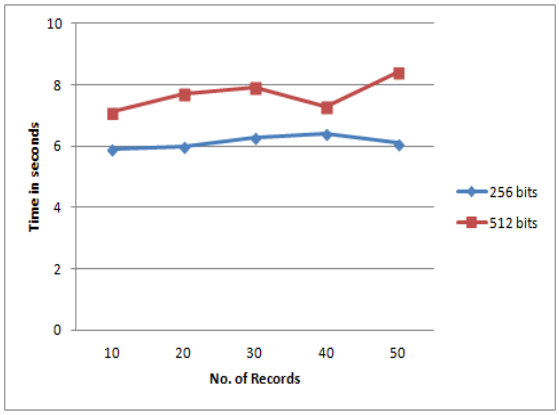
ISSN: 1992-8645 www.jatit.org E-ISSN: 1817-3195

On the other side, the data owner must upload their files either on private cloud or public cloud. While uploading their files, upload time has been calculated using their respective random number on each count. On a remote machine setting, this can be considered as IDI, For 256 bits field size, the values are ranges from 151.7 microseconds to 149.8 microseconds. When the field size is doubled as 512 bits, the values are ranges from 181.3 microseconds to 239.2 microseconds. The following Graph.5 shows the status of data upload time on a remote machine setting.



Graph 5: Data Owner's Upload Time For Varying Records And Field Sizes On A Remote Machine.

When the files are uploaded in the cloud setting environment, upload time has been calculated using their respective random number on each count. On a cloud environment setting, for 256 bits field size, the values are ranges from 5.9 seconds to 6.1 seconds. When the field size is doubled as 512 bits, the values are ranges from 7.1 seconds to 8.4 seconds. The following Graph.6 shows the status of data upload time on the cloud.

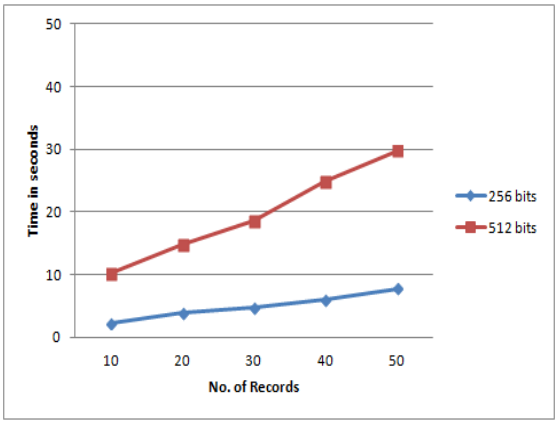


Graph 6. Data Owner's Upload Time For Varying Records And Field Sizes On The Cloud.

4.3 Cloud Performance

As we discussed before, when the system acts as remote machine, the computation time for the records ranges from 10 to 50 for different field sizes has been calculated. For the field size 256 bits, the values are ranges from 2 seconds to 7.8 seconds.

Respectively for the field size 512 bits, the values are ranges from 10.2 seconds to 29.8 seconds. When the field size is doubled, the computation time of the cloud is increased by the factor 5 almost. The Graph.7 depicts the ranges of value of a cloud performance on a remote machine for varying field sizes of records and number of records.

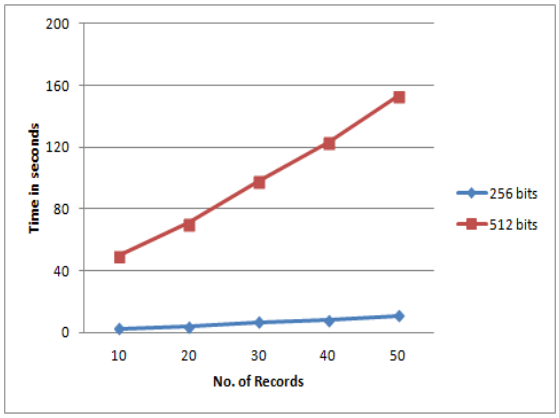


Graph 7: Cloud Performance For Varying Records And Field Sizes On A Remote Machine

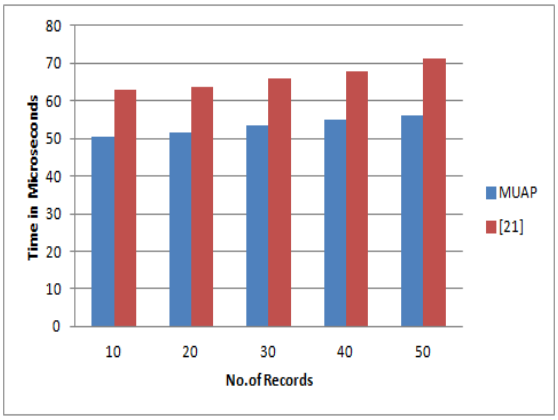
On the second hand, when the system acts as a virtual machine in the cloud, the computation time for the records ranges from 10 to 50 for different field sizes has been calculated. For the field size 256 bits, the values are ranges from 2.7 seconds to 10.9 seconds. Respectively for the field size 512 bits, the values are ranges from 49.8 seconds to 153.2 seconds.

When the field size is doubled, the computation time of the cloud is increased by the factor 15 almost. This increase in computation time may be from other loads on EC2 server. The Graph.8 depicts the ranges of value of a cloud performance on Amazon EC2 for varying field sizes of records and number of records.

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



Graph 8: Cloud Performance For Varying Records And Field Sizes On Amazon EC2



Graph 9: Data Requester's Computation Time To Download Files For Field Size 256 Bits.

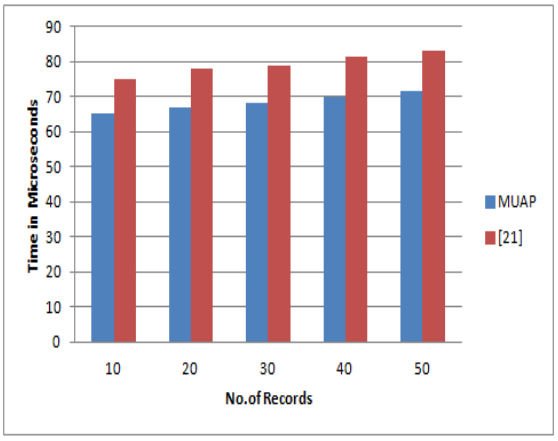
4.4 Comparative Performance

Consider the settings of remote machine, comparative analysis has been worked out between the secure data sharing and query processing system [21] and our proposed Mutual User Authentication Protocol (MUAP) system. The comparative analysis performed under two criteria, which are the computation cost of uploading records on Data owner side and computation cost of downloading records on Data requester side.

The data requester's download time on the remote machine varies between 50.45 microseconds to 55.21 microseconds for 10 to 50 records having 256 field sizes. For 512 field sizes, it varies between 63.26 microseconds to 68.91 microseconds for 10 to 50 records. When the field size is doubled, there may be a little hike on starting range but, maintains the consistency over the period values.

On the other side, the values are ranges from 62.73 microseconds to 71.23 microseconds for varying 10 to 50 records of 256 bits field size. Obviously there is a hike in computation cost values when the field size is doubled. It ranges from 75.23 microseconds to 83.26 microseconds.

Graph 9 and Graph 10 depict the graphical representation of computation time on the data requester side while downloading the files for different field size 256 and 512 bits respectively.



Graph 10: Data Requester's Computation Time To Download Files For Field Size 512 Bits.

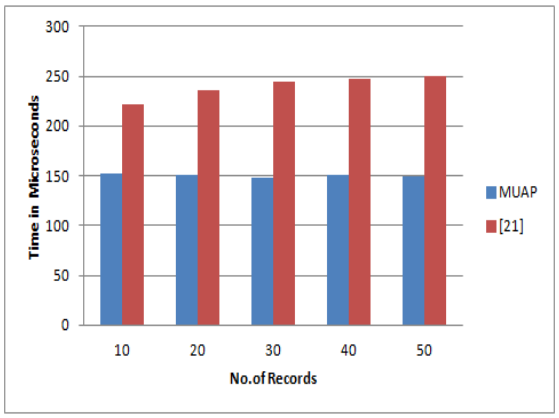
From the point of Data owner side, the data owner must upload their files either on private cloud or public cloud. While uploading their files, upload time has been calculated using their respective random number on each count. On a remote machine setting, this can be considered as IDI, For 256 bits field size, the values are ranges from 151.7 microseconds to 149.8 microseconds.

At the same point of view, in secure data processing and query processing framework [21] the upload time is calculated and compared with our MUAP system. The values are ranges from 221.6 microseconds to 250.2 microseconds for varying records from 10 to 50. It shows the notable increase in the values which depicts the efficiency of our system.

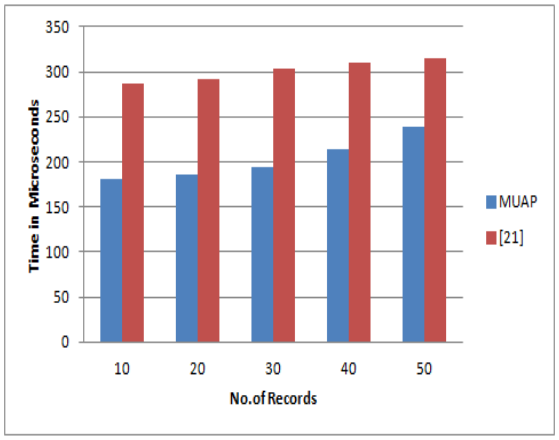
When the field size is doubled as 512 bits, the values are ranges from 181.3 microseconds to

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

239.2 microseconds for varying records from 10 to 50. Respectively in secure data processing and query processing framework [21], the values are increased and observed in the Graph 11 and Graph 12.



Graph 11: Data Owner's Computation Time To Upload Files For Field Size 256 Bits.



Graph 12: Data Owner's Computation Time To Upload Files For Field Size 512 Bits.

The following parameters are considered important criteria to evaluate the proposed scheme:

4.5 Efficiency

This proposed mutual authentication protocol is evaluated under the criteria, efficiency which involves the following key points.

4.5.1 User Annulment
4.5.2 Indexed Searching
4.5.3 Reduced Communication Overhead

4.5.1 User annulment

The owner is the responsible person to update the ACL in the situation of user revocation. He conveys the user name of the annulled user which is hashed to the IDI and the server.

4.5.2 Indexed searching

The owner creates the ACL for every file to be uploaded to the cloud with the list of indexed words. Any user in the cloud those who wants to access the file can search their particular file with the specific words. By this way the server cannot have the access to the whole content of the file.

If the owner needs to add or modify some words in ACL, then it is easily updatable and need to send the updated ACL to the server and IDI.

4.5.3 Reduced communication overhead

In early schemes, whenever the revocation occurred the owner has to generate the new key and distributed among all the users in the cloud. If there are 'n' users, then the owner has to send the key to all the 'n'users. In this situation there are 'n' communications taking place. It increases the communication overhead in the cloud.

In our proposed scheme, even though there are 'n' users in the cloud whenever the revocation occurred the owner has to generate the new key. But the owner has to send the new key only to the user who hss requested the file for access. So, it mainly reduces the communication overhead in the cloud.

Table 4 describes comparison over the criteria efficiency of our proposed scheme to the related earlier schemes.

Table 4: Comparison Over Efficiency

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

Scheme / Criteria	[16]	[18]	[19]	[20]	[22]	MUAP
User Annulment	No	Yes	No	Yes	Yes	yes
Indexed Search	No	No	No	No	No	Yes
Reduced Communication Overhead	No	Yes	No	Yes	No	Yes

4.6 Security

This proposed mutual authentication protocol is evaluated under the criteria security which involves the following key points that addresses threats related to the system.

4.6.1 Mutual Authentication
4.6.2 MITM Attack prevention
4.6.3 Trace out the user
4.6.4 Preserving privacy
4.6.5 Replay Attacks
4.6.6 Malicious User Attacks
4.6.7 Collusion Prevention

4.6.1 Mutual authentication

When the owner encounters the request from any user, then he can assure that the request is from the legal user. Because the request is analyzed and checked by the IDI. To achieve the Mutual Authentication the new key is encrypted by owner's private key and transfers to the user who is requested. On the other side, the user will decrypt the message with the owner's public key.

4.6.2 MITM attack prevention

If the intruder interrupts the connection and tries to catch the message, he would get nothing. When communication takes place all the entities use the asymmetric key for the encryption. Even though the intruder gets the encrypted message while attacking he cannot get the original message due to asymmetric key.

4.6.3 Trace out the user

Sometimes, an attacker will fake the user enrollment process in the cloud and exports some calamitous file. If any user wants to access that file, then he will be infected by that calamitous file.

To avoid this problem, there is one step in creating and uploading ACL of the file. The owner needs to sign up the ACL with the help of RSA algorithm while uploading the file in the cloud. Due to that sign up procedure, the server can trace out the user who caused that terrible situation.

4.6.4 Preserving privacy

In the ACL, the owner uploads only the hashed content of the user names. While sending a request to the server, it sends only hashed name of the user. This preserves the identity of the claiming person from the server.

4.6.5 Replay attack

When there is any situation regarding the attacks when catch up the request in the middle of communication and resend the request to the server. By that time the IDI or the server will check the freshness of the request that is related to the ACL of the file. If it is not matched, then the request is denied from the server or IDI.

4.6.6 Malicious user attack

Consider the situation, in which the attacker continuously sends the request to the server for any particular file access, the server will forward the request to the owner. If the attacker pressures the owner then the owner will push the user in the blocked lists.

4.6.7 Collusion prevention

In the Access of Files stage, IDI will check the freshness of the request from the server. If the timestamp is fresh then it will find the match from the ACL. It leads that there was no collusion and the request is forwarded to the owner. If the IDI does not find the match then, there will be collusion. Finally the IDI will terminate the request from the server. So, IDI will take care of the collusion prevention.

Table 5 depicts the comparison of the proposed scheme to the earlier schemes which is related to security issues.

Table 5: Comparison Over Security

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

Scheme / Criteria	[16]	[18]	[19]	[20]	[22]	MUAP
Mutual Authentication	No	No	No	No	No	Yes
MITM Attack Prevention	No	Yes	Yes	No	No	Yes
Secure Distribution	No	Yes	Yes	No	No	Yes
Collusion Prevention	No	No	No	No	Yes	Yes

5. CONCLUSION

The proposed scheme that is MUAP mechanism using ID in cloud computing, which allows the sharing of files securely over untrusted and unsecured servers. By using hybrid cloud in the proposed scheme, the user can export less significant files in the public cloud and most significant files in a private cloud. The IDI is responsible for the security of the most important files which is saved in the private cloud. User annulment and key distribution are performed efficiently to achieve higher security. The proposed system makes the key distribution simple so that it reduces communication overhead. It also makes the generation of the keys very less when compared to related systems the computation overhead is enormously reduced. The proposed system focuses on the individual activities rather than the group activities. It achieves mutual authentication while transfer the files to the user which overcomes the breaches of main security threats. Finally, the proposed scheme is efficient, secure, reliable and easy approach to the users with reduced computational and communication overhead.

REFERENCES

[1] P.Mell and T. Grance, "Draft NIST Working Definition of Cloud Computing", national Institute of Standards and Technology, USA,2009.

[2] A. Mishra, R.Jain and A.Durresi, " Cloud Computing: Networking and Communication Challenges", IEEE communications Magazine, Volume 50, no. 9, pp 24-25, 2012.

[3] K.Hwang and D.Li, " Trusted Cloud Computing with secure resources and Data Coloring", IEEE Internet Computing, Volume 14, no. 5, pp 14-22, 2010.

[4] J. Chen, Y. Wang, and X. Wang, "On-Demand Security Architecture for Cloud Computing," Computer, vol. 45, no. 7, pp. 73-78, 2012.

[5] K. Yang and X. Jia, "An Efficient and Secure Dynamic Auditing Protocol for Data Storage in Cloud Computing," IEEE Transactions on Parallel and Distributed Systems, [online] ieeexplore.ieee.org/stamp/stamp.jsp?tp=&arnumber=6311398, 2012.

[6] Q. Wang, C. Wang, K. Ren, W. Lou, and J. Li, "Enabling Public Auditability and Data Dynamics for Storage Security in Cloud Computing," IEEE Transactions on Parallel and Distributed Systems, vol. 22, no. 5, pp. 847-859, 2011.

[7] C. Wang, K. Ren, W. Lou, J, Lou,"Toward Publicly Auditable Secure Cloud Data Storage Services," IEEE Network, vol. 24, no. 4, pp. 19-24, 2010.

[8] Y. Zhu, H. Hu, G. Ahn, and M. Yu, "Cooperative Provable Data Possession for Integrity Verification in Multi-cloud Storage," IEEE Transactions on Parallel and Distributed Systems, vol. 23, no, 12, pp. 2231-2244, 2012.

[9] H. Wang, "Proxy Provable Data Possession in Public Clouds," IEEE Transactions on Services Computing, [online] ieeexplore.ieee.org/stamp/stamp.jsp?tp=&arnumber=6357181, 2012.

[10] L. A. Dunning and R. Kresman, "Privacy Preserving Data Sharing With Anonymous ID Assignment," IEEE Transactions on Information Forensics and Security, vol. 8, no. 2, pp. 402-413, 2013.

[11] X. Liu, Y. Zhang, B. Wang, and J. Yan, "Mona: Secure MultiOwner Data Sharing for Dynamic Groups in the Cloud," IEEE Transactions on Parallel and Distributed Systems, [online] ieeexplore.ieee.org/stamp/stamp.jsp?tp=&arnumber=6374615, 2012.

[12] S. Grzonkowski and P. M. Corcoran, "Sharing Cloud Services: User Authentication for Social Enhancement of Home Networking," IEEE Transactions on Consumer Electronics, vol. 57, no. 3, pp. 1424-1432, 2011.

[13] M. Nabeel, N. Shang and E. Bertino, "Privacy Preserving Policy Based Content Sharing in Public Clouds," IEEE Transactions on Knowledge and Data Engineering, [online] ieeexplore.ieee.org/stamp/stamp.jsp?tp=&arnumber=6298891, 2012.

[14] C. Wang, Q. Wang, K. Ren, N. Cao, and W. Lou, "Toward Secure and Dependable Storage Services in Cloud Computing," IEEE

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

Transactions on Services Computing, vol. 5, no. 2, pp. 220-232, 2012.

[15] S. Sundareswaran, A. C. Squicciarini, and D. Lin, "Ensuring Distributed Accountability for Data Sharing in the Cloud," IEEE Transactions on Dependable and Secure Computing, vol. 9, no. 4, pp. 556-568, 2012.

[16] M. Kallahalla, E. Riedel, R. Swaminathan, Q. Wang, and K. Fu,"Plutus: Scalable Secure File Sharing on Untrusted Storage," Proc. USENIX Conf. File and Storage Technologies, pp. 29-42, 2003.

[17] G. Ateniese, K. Fu, M. Green, and S. Hohenberger, "Improved Proxy Re-Encryption Schemes with Applications to SecureDistributed Storage," Proc. Network and Distributed Systems Security Symp. (NDSS), pp. 29-43, 2005

[18] Hong Liu, Huansheng Ning, Qingxu Xiong and L. T. Yang, "Shared Authority Based Privacy-Preserving Authentication Protocol in Cloud Computing," in IEEE Transactions on Parallel and Distributed Systems, vol. 26, no. 1, pp. 241-251, Jan. 2015.

[19] Yang, Guangyang, et al. "Enabling public auditing for shared data in cloud storage supporting identity privacy and traceability." Journal of Systems and Software 113 (2016): 130-139.

[20] Zhu, Zhongma, and Rui Jiang. "A Secure Anti-Collusion Data Sharing Scheme for Dynamic Groups in the Cloud.", IEEE Transactions on Parallel and Distributed Systems 27.1 (2016): 40-50.

[21] Samanthula, Bharath K., et al. "A secure data sharing and query processing framework via federation of cloud computing." Information Systems 48 (2015): 196-212.

[22] G. Ateniese, K. Benson, S. Hohenberger, Key-private proxy reencryption, in: Proceedings of the Cryptographers' Track at the RSA Conference (CT-RSA'09), Springer-Verlag, 2009, pp. 279–294.

[23] M. Blaze, G. Bleumer, M. Strauss, Divertible protocols and atomic proxy cryptography, in: EUROCRYPT, Springer-Verlag, 1998, pp. 127–144, 2016.

[24] R. S´anchez, F. Almenares, P. Arias, D. D´ıaz-S´anchez, and A. Mar´ın, "Enhancing Privacy and Dynamic Federation in IdM for Consumer Cloud Computing," IEEE Transactions on Consumer Electronics, vol. 58, no. 1, pp. 95-103, 2012.

[25] H. Zhuo, S. Zhong, and N. Yu, "A Privacy-Preserving Remote Data Integrity Checking Protocol with Data Dynamics and Public Verifiability," IEEE Transactions on Knowledge and Data Engineering, vol. 23, no. 9, pp. 1432-1437, 2011.

[26] Y. Xiao, C. Lin, Y. Jiang, X. Chu, and F. Liu, "An Efficient Privacy-Preserving Publish-Subscribe Service Scheme for Cloud Computing," in Proceedings of Global Telecommunications Conference (GLOBECOM 2010), December 6-10, 2010.

[27] I. T. Lien, Y. H. Lin, J. R. Shieh, and J. L. Wu, "A Novel Privacy Preserving Location-Based Service Protocol with Secret Circular Shift for K-nn Search," IEEE Transactions on Information Forensics and Security, [online] ieeexplore.ieee.org/stamp/stamp.jsp?tp=&arnumber=6476681, 2013.

[28] Y. Tang, P. C. Lee, J. C. S. Lui, and R. Perlman, "Secure Overlay Cloud Storage with Access Control and Assured Deletion," IEEE Transactions on Dependable and Secure Computing, vol. 9, no. 6, pp. 903-916, 2012.

[29] Y. Zhu, H. Hu, G. Ahn, D. Huang, and S. Wang, "Towards Temporal Access Control in Cloud Computing," in Proceedings of the 31st Annual IEEE International Conference on Computer Communications (IEEE INFOCOM 2012), pp. 2576-2580, March 25-30, 2012.

[30] S.Ruj,M.Stojmenovic,andA.Nayak,"DecentralizedAccessControl with Anonymous Authentication for Securing Data in Clouds," IEEE Transactions on Parallel and Distributed Systems, [online] ieeexplore.ieee.org/stamp/stamp.jsp?tp=&arnumber=6463404, 2013.

[31] A. Barsoum and A. Hasan, "Enabling Dynamic Data and Indirect Mutual Trust for Cloud Computing Storage Systems," IEEE Transactions on Parallel and Distributed Systems, [online] ieeexplore.ieee.org/stamp/stamp.jsp?tp=&arnumber=6392165, 2013.

[32] H. Y. Lin and W. G. Tzeng, "A Secure Erasure Code-Based Cloud Storage System with Secure Data Forwarding," IEEE Transactions on Parallel and Distributed Systems, vol. 23, no. 6, pp. 995-1003, 2012.

[33] J. Yu, P. Lu, G. Xue, and M. Li, "Towards Secure MultiKeyword Top-k Retrieval over Encrypted Cloud Data," IEEE Transactions on Dependable and Secure Computing, [online] ieeexplore.ieee.org/stamp/stamp.jsp?tp=&arnumber=6425381, 2013.

[34] K. W. Park, J. Han, J. W. Chung, and K. H. Park, "THEMIS: A Mutually Verifiable Billing

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

System for the Cloud Computing Environment," IEEE Transactions on Services Computing, [online] ieeexplore.ieee.org/stamp/stamp.jsp?tp=&arnumber=6133267, 2012.

[35] F. Ghasemi, R. Parsamehr, "A multi authentication scheme for RFID systems." Cumhuriyet Science Journal, Volume 36, Issue 3, Pages 1400-1406, 2015.

[36] Armen Dzhagaryan, Aleksandar Milenkovic, "A framework for optimizing file transfers between mobile devices and the cloud", IEEE 28 th Annual International symposium on Personal, Indoor, and Mobile Radio Communications(PIMRC), 2017.

[37] Wei-Hao Chen, Chun-I Fan, Yi-Fan Tseng," Efficient Key Aggregate Proxy Re-Encryption for secure data sharing in clouds", IEEE conference on Dependable and Secure(DSC), 2018.

[38] Youcef Imine, Djamel Eddine Kouicem, Abdelmadjid Bouabdallah, "MASFOG: An Efficient Mutual Authentication Scheme for Fog Computing Architecture" 17th IEEE International Conference on Trust, Security and Privacy in Computing and Communications/12th IEEE International Conference on Big Data Science and Engineering (TrustCom/BigDataSE), 2018.

[39] Xiaoying Jia, Debiao He, Neeraj Kumar, Kim Kwang, Raymond Choo, " A Provably Secure and Efficient Identity Based Anonymous Authentication Scheme for Mobile Edge Computing" , IEEE Systems Journal, Issue 99, 2019.

[40] Abdelouahid Derhab, Mohamed Belaoued, Mohamed Guerroumi, Farrukh Aslam Khan, "Two-Factor Mutual Authentication Offloading for Mobile Cloud Computing", IEEE Access,Volume 8, 2020.