Available Online at www.ijcsmc.com

International Journal of Computer Science and Mobile Computing

A Monthly Journal of Computer Science and Information Technology

ISSN 2320–088X

IJCSMC, Vol. 4, Issue. 6, June 2015, pg.106 – 113

RESEARCH ARTICLE

CLOUD SECURITY USING SAP- SHARED AUTHENTICATION PROTOCOL

Dr. Mohammed Abdul Waheed, Anupama Hanamgond
Associate Professor, Department of Computer Science and Engineering, Visvesvaraya Technological University, India
Student (M.Tech) CSE, Department of Computer Science and Engineering, Visvesvaraya Technological University, India
dr.mawaheed@gmail.com; anupamahanamgond3@gmail.com

Abstract—In the cloud computing users (information owners host their information on cloud servers and other users (data consumers) will access the information from cloud servers. During such data accessing, the data sharing between different users is significant to have business or commercial productive benefits. In providing such sharing existing security solutions mainly focus on authentication so as prevent unauthorized access to users' private data, but concerns nothing about the privacy of user when user challenges the cloud server for other user's data. The challenged request itself may reveal user identity irrespective of whether or not user gets the data access permission. To address this privacy issue we propose shared authentication (SAP) protocol. Using SAP 1) shared access authority is achieved with authentication, data anonymity, user privacy and forward security. 2) Attribute based access control is provided. 3) Proxy re-encryption is applied by cloud server to provide data sharing among different users.

Keywords— Cloud computing, authentication protocol, privacy preservation, shared authority, data confidentiality.

I. INTRODUCTION

Cloud computing provides a novel computing paradigm for both individuals and enterprises to store programs and data in the cloud in a transparent manner with advantages such as ubiquitous network access, resource pooling with no infrastructure limitations etc. In cloud computing typical architecture is anything as a service (Xaas) in which software and computing resources are provided with ubiquitous interconnections. Cloud computing is the use of internet for the tasks performed on local machines with hardware and software demands maintained at a users' local machine, which poses new privacy and security challenges. The conventional security approaches focus mainly on strong authentication where a user can access its own authorized data fields and users' private data are not unauthorized accessed by any one.

Along with diversity of cloud services users may want to access and share each others' private authorized data fields to achieve some productive benefits which bring new privacy and security challenges for cloud storage.

To this end we propose a protocol to support shared authority based access while preserving privacy.
Any security protocol in cloud should achieve following needs

1) Authentication: Any legal user can only access authorized partial or entire data fields but not any forged or tampered data fields.

2) Data anonymity: Data is not readable or identifiable by the adversary or by the attacker even if the exchanged messages are captured by the attacker.

3) User privacy: Any adversary cannot guess the user's access desire which can breach the privacy of the user with interest in another user's data fields, if and only if both users want to mutually share each others data only then the cloud server will inform the two users the permission to access each other data.

4) Forward security: When an adversary captures messages between two users he cannot be able to discover the prior interrogations from the communication sessions.

Many researches on cloud application focus mainly on strong authentication and ignores the case of users want to share and access each others' data fields. When a user challenges the cloud server to request other users for data sharing, the challenged request itself may reveal the user's privacy no matter whether the user gets the data access permission.

In this paper we address users' sensitive access desire and design a security scheme to achieve data access control, access authority sharing and privacy preservation.

We address the privacy issue to propose a shared authentication protocol (SAP) to realize authentication and authorization while preserving privacy. The main contributions of this work are

1) Address a privacy issue whenever a user requests the cloud server for data sharing. The challenged request itself should not reveal the user's privacy irrespective of whether or not the user gets the data access permissions.
2) Enhance the user's access request privacy with an authentication protocol and shared access authority is achieved by anonymous request matching mechanism.
3) Attribute based access control is provided so the user can only access its authorized data fields. Proxy re-encryption is applied by cloud server to provide confidential data sharing among different users over the cloud.

II. RELATED WORK

Security architectures proposed by K. Hwang et al. [1] and J. Chen et al. [2] attempt to give security for trusted cloud computing. Data coloring and software water marking techniques protect shared data objects and massively distributed software modules. The techniques are limited by only the focus on multi-way authentication and tighten access control for confidential data in cloud.

Data public auditing protocols proposed by:

1) K. Yang et al. [3] supports batch auditing for each multi-owner and multiple clouds.
2) Q. Wang et al. [4] proposed an auditing protocol that ensures integrity of data storage in cloud computing and supports both public data auditing through the involvement of trusted third party and dynamic data operations.

Secure data sharing data storage protocols proposed by:

1) Dunning et al. [5] uses AIDA algorithm to assign each user a unique ID in a multi-user cloud over a distributed computing environment. These IDs can be used as a part procedure for sharing of data storage, communication and any computing resources over a cloud environment.

2) Liu et al. [6] proposed a multi-owner data sharing secure scheme (MONA) for the cloud which supports dynamic groups. In this users can share data securely with un-trusted cloud and any user can decrypt owner files directly without owner's permission. Access control is applied to keep track of any malicious use of cloud resource by any user.

3) Nabeel et al. [7] proposed a broadcast group management (BGKM) which overcomes the drawbacks of symmetric key cryptosystem in public clouds, BGKM also realizes that a user need not utilize public key cryptography and can dynamically derive the symmetric keys during decryption. The algorithm uses access control vector (ACV) for assigning secrets to users based on the identity attributes so that users are allowed to derive actual symmetric keys based on their own secrets and public information

4) Sundareswaran et al. [8] proposed a decentralized answerability framework to trace the users' overall actual data usage in cloud. The framework uses auto-logging and auditing of sharing and accessing performed by any user at any cloud service provider and at any point in time. This also uses JAR to aid an additional concept of security to the platform.

Privacy preserving schemes proposed by Xiao et al. [9] aims at improving privacy preserving authentication, information integrity and public-subscribe confidentiality using ZNP and homo-morphic cryptography.

In all the works mentioned above many security issues are resolved. But a user's access desire related privacy concern caused by the sharing and accessing of data is not addressed. Here we propose a protocol which not only focus on authentication but also provide authorizing to share access authority among many users while preserving privacy. Both attribute based encryption and proxy re-encryption is used to achieve authorization and authentication. The protocol uses anonymity to conceal user identity in the data access request.

III. SYSTEM MODEL

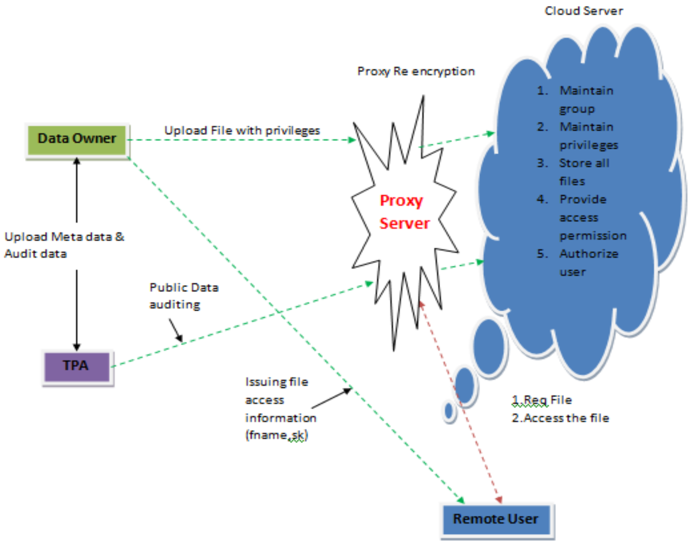

Fig. 1 System model

Fig. 1 illustrates the system model for storage, access and sharing of cloud data with the main entities user (U_x), cloud server (S), Third party authenticator (TPA).

User: user is any individual person or entity or any group of persons who owns data that are stored in the cloud storage for computing services. Different users are provided with different authorities and privileges on cloud data.

Cloud server: it is the main server of computing resources and data storage. It is maintained by a cloud service provider. The cloud server is an entity with unlimited cloud storage and computing resources. The cloud server's main services include maintain group, maintain privileges, store all files, provide access permissions and authorize users.

Third party authenticator: an entity which performs public auditing of data on behalf of users through the proxy server.

Cloud storage is a great upcoming platform for user to store data remotely and to operate it in the internet of services including both distributed and parallel mode of services.

Each user is provided with independent access authority on its own data and users' data are never unauthorized accessed by any other users in cloud.

It is important to maintain privacy of user in data access desire. There are cases such as electronic health record (EHR) system used is by hospitals where there are many users with different attributes and authorities. Any user may want to access other user's private data. This challenged access request must be kept private so as hide the user's identity.

Towards the trust model, this model considers no trust relationships between user (U_x) and cloud server (S)

- S is semi-honest and curious: in this the server S is maintained by a cloud service provider but may want to know about users' private information. In passive but honest model S cannot be supposed to forge the users' data fields.
- U_x is rational but sensitive: by being rational user U_x want to obtain other user's private data with negative or selfish interest. By being sensitive U_x want to keep its sensitive data confidential but want to purposely obtain others users' privacy.

Towards the threat model, this considers all security related at attacks during the data storage and access.

There exist two types of attacks

- Internal attacks: refer to entities S and U_x. U_x want to obtain other users' data with malicious interests.
- External attacks: with these attacks attackers can breach the confidentiality, integrity and availability and can modify (e.g., insert, or delete) user data fields.

IV. METHODOLOGY

Protocol steps:

System Initialization:

The cloud storage system includes a cloud server S, and users $\{U_x\}$ ($x = \{1,\ldots,m\}, m \in N^*$). Thereinto , U_a and U_b are two users, which have independent access authorities on their own data fields. It means that a user has an access permission for particular data fields stored by S, and the user cannot exceed its authority access to obtain other user's data fields. Here we consider S and $\{U_a, U_b\}$ to present the protocol phases for data access control and sharing of authority access with privacy concern.

Let $BG = (q, g, h, G, G', e, H)$ be a pairing group, in which q is a large prime, $\{G, G'\}$ are of prime order $q, G = <g> = <h>$, and H is a collision-resistant hash function. The bilinear map $e : G \times G \rightarrow G'$ satisfies the bilinear non-degenerate properties: i.e., for all $g, h \in G$ and $a, b \in Z_{q,}^*$ it turns out that $e(g^{a,}, h^b) = e(g,h)^{ab}$, and $e(g,h) \neq 1$. Meanwhile, $e(g,h)$ can be efficiently obtained for all $g, h \in G$, and it is a generator of G'.

Let S and U_x respectively own the pairwise keys {p$k_{s,}$ sk_s} and {p$k_{U_x.}$ sk_{U_x}}. Besides, S is assigned with all users' public keys {p$k_{U_{1\ldots\ldots}}$ pk_{U_m}}, and U_x is assigned with pk_s. Here the public key $pk_T = g^{sk_T}$ (mod q) $(T \in \{S, U_x\})$ and the corresponding privacy key $sk_T \in Z_q^*$ are defined according to the generator g.

Let $F(R_{U_x}^{U_y}(R_{U_y}^{U_x})^T) = Cont \in Z_q$ describe the algebraic relation of $\{(R_{U_x}^{U_y}, R_{U_y}^{U_x}$), which are mutually inverse access requests challenged by $\{U_{x,} U_y\}$, and $Cont$ is a constant. Here $F(.)$ collision-resistant function, for any randomized polynomial time algorithm A, there is a negligible function $p(k)$ for a sufficiently large value k.

The Proposed Protocol Descriptions

Fig. 1. Shows the interactions among the users U_x, S and TPA.

$\{U_a, U_b\}$'s Access Challenges and S's Responses:

$\{U_a, U_b\}$ respectively generate the session identifiers $\{sid_{U_a}, sid_{U_b}\}$, extract the token identifiers $\{T_{U_a}, T_{U_b}\}$, and transmits $\{sid_{U_a} \parallel T_{U_a}, sid_{U_b} \parallel T_{U_b}\}$ to S as an access query to initiate a new session. Accordingly we take the interactions of U_a and S as an example to introduce the following authentication phase. Upon receiving U_a's challenge, S first generates a session identifier $sid_{S_{a.}}$ and establishes the master public key $mpk = (g_{i,} h, h_i, BG, e(g,h), H)$ and master privacy key $msk = (\alpha, g)$. Thereinto S randomly chooses $\alpha \in Z_q$, and computes $g_i = g^{\alpha^i}$ and h_i=$h^{\alpha^{i-1}}$ ($i = \{1, \ldots\ldots, n\} \in Z^*$). S randomly chooses $\sigma \in \{0,1\}^*$, and extracts U_a's access authority policy $P_{U_a} = [p_{ij}]n \times m(p_{ij} \in \{0,1\})$, and U_a are assigned with the access authority on its own data fields D_{U_a} within P_{U_a}'s permission. S further defines a polynomial $F_{S_a}(x, P_{U_a})$ according to P_{U_a} and T_{U_a}.

$$F_{S_a}(x, P_{U_a}) = \prod_{i=1, j=1}^{n,m}(x + ijH(T_{U_a}))^{p_{ij}} \quad (mod\ q)$$

S computes a set of values $\{M_{S_a 0}, M_{S_a 1}, \{M_{S_a 2i}\}, M_{S_a 3,} M_{S_a 4}\}$ to establish the ciphertext $C_{S_a} = \{M_{S_a 1}, \{M_{S_a 2i}\}, M_{S_a 3,} M_{S_a 4}\}$, and transmits $sid_{S_a} \parallel C_{S_a}$ to U_a.

$$M_{S_a 0} = H(P_{U_a} \parallel D_{U_a} \parallel T_{U_a} \parallel \sigma),$$

$$M_{S_a 1} = h^{F_{S_a}(\alpha, P_{U_a}) M_{S_a 0}},$$

$$M_{S_a 2i} = (g_i)^{M_{S_a 0}}, \quad (i = 1, \ldots\ldots, n)$$

$$M_{S_a 3,} = H(e(g,h)^{M_{S_a 0}}) \oplus \sigma,$$

$$M_{S_a 4} = H(sid_{U_a} \parallel \sigma) \oplus D_{U_a},$$

Similarly, S performs the corresponding operations for $U_{b,}$ including that S randomly chooses α'∈Z_q and $\sigma' \in \{0.1\}^*$, establishes $\{g_i', h_i'\}$, extracts $\{P_{U_{b.}} D_{U_b}\}$, and defines $F_{S_b}(x, PU_b)$, and computes , $\{M_{b1}, \{M_{S_b 2i}\}, M_{b3,} M_{S_b 4}\}$ to establish a ciphertext C_{S_b} for transaction.

$\{U_a, U_b\}$'s Data Access Control:

U_a first extracts it data attribute access list $A_{U_a} = [a_{ij}](a_{ij} \in \{0,1\}, a_{ij} \leq p_{ij})$ to re-structure an access list $L_{U_a} = [l_{ij}]n \times m$ for l_{ij}=$p_{ij} - a_{ij}$. U_a also defines a polynomial $F_{U_a}(x, L_{U_a})$ according to $L_{U_a}$and T_{U_a}.

$F_{U_a}(x, L_{U_a}) = \prod_{i=1,j=1}^{n,m}(x + ijH(T_{U_a}))^{l_{ij}} \quad (\mathit{mod}$ q)

It turns out that $F_{U_a}(x, L_{U_a})$ satisfies the equation.

$$F_{U_a}(x, L_{U_a}) = \prod_{i=1,j=1}^{n,m}(x + ijH(T_{U_a}))^{p_{ij}-a_{ij}}$$

$$= F_{S_a}(x, P_{U_a}) \big/ F_{S_a}(x, A_{U_a}).$$

Afterwards, U_a randomly chooses $\beta \in Z_q$, and the decryption key $k_{A_{U_a}}$ for A_{U_a} can be obtained.

$k_{A_{U_a}} = (g^{(\beta+1)/F_{S_a}(\alpha, A_{U_a})}, h^{\beta-1})$

U_a further computes a set of values $\{N_{U_a1}, N_{U_a2}, N_{U_a3}\}$.

Here, $f_{s_{ai}}$ is used to represent $x^{i'}$s co-efficient in $F_{S_a}(x, P_{U_a})$, and $f_{U_{ai}}$ is used to represent $x^{i'}$s co-efficient in $F_{U_a}(x, L_{U_a})$.

$$N_{U_a1} = e(M_{S_a21}, \prod_{i=1}^{n}(h_i)^{f_{U_a i}} h^{f_{U_a 0}}),$$

$$N_{U_a2} = e(\prod_{i=1}^{n}(M_{S_h 2i})^{f_{U_a}{}^{i}}, h^{\beta-1}),$$

$N_{U_a3} = e(g^{(\beta+1)/f_{S_a(\alpha, A_{U_a})}}, M_{S_a1}).$

It turns out that $e(g, h)^{M_{S_a0}}$ satisfies the equation.

$e(g, h)^{M_{S_a0}} = (N_{U_a3}/(N_{U_a1}N_{U_a2}))^{1/f_{U_a0}}$

For the right side of (1), we have,

$$N_{U_a1} = e(g^{\alpha^i M_{S_a0}}, \prod_{i=1}^{n}(h_i)^{f_{U_a}{}^{i}} h^{f_{U_a}{}^{0}})$$

$=e(g, h)^{\alpha M_{S_a0}} \sum_{i=1}^{n}(\alpha^{i-1} f_{U_a} i + f_{U_a} 0)$

$= e(g, h)^{M_{S_a0} f_{U_a(\alpha, L_{U_a})}},$

$N_{U_a2} = e(\prod_{i=1}^{n} g^{\alpha^i M_{S_a0} f_{U_a}{}^{i}}, h^{\beta-1})$

$= e(g, h)^{\alpha M_{S_a0}(\sum_{i=1}^{n} \alpha^i f_{U_a} i + f_{U_a} 0 - f_{U_a} 0)(\beta-1)}$

$= e(g, h)^{M_{S_a0}\beta f_{U_a}(\alpha, L_{U_a}) - M_{S_a0} f_{U_a} 0,}$

$N_{U_a3}=e(g, h)^{(\beta+1)/f_{S_a(\alpha, A_{U_a})}}, h^{f_{S_a0} M_{S_a0}} \prod_{i=1}^{n}(h_i)^{f_{S_a i} M_{S_a0}})$

$= e(g, h)^{(\beta+1)/F_{S_a(\alpha, A_{U_a})} F_{S_a(\alpha, P_{U_a})} M_{S_a0}}$

$=e(g, h)^{M_{S_a0}\beta F_{U_a}(\alpha, L_{U_a}) + M_{S_a0} F_{U_a}(\alpha, L_{U_a})},$

$(N_{U_a1 N_{U_a2}}/N_{U_a3})^{-1/F_{U_a0}} = (e(g, h)^{-M_{S_a0} f_{U_a0}})^{-1/f_{U_a0}}$

$= e(g, h)^{M_{S_a0}}.$

U_a locally re-computes $\{\sigma^{l, M^{l}_{S_a0}}\}$, derives its own authorized data fields D_{U_a}, and checks whether the ciphertext C_{S_a} is encrypted or not by $M^{l}_{S_a0}$. If this holds then U_a will be the legal user to decrypt C_{S_a}; otherwise the protocol will terminate.

$\sigma^{l} = M_{S_a3} \oplus H(e(g, h)^{M_{S_a0}}),$

$M^{l}_{S_a0} = H((P_{U_a} \parallel T_{U_a} \parallel \sigma^{l}),$

$D_{U_a} = M_{S_a4} \oplus H(sid_{U_a} \parallel \sigma^{l}).$

U_a further extracts its pseudonym PID_{U_a} a session sensitive access request $R^{U_a}_{U_b}$, and the public key pk_{U_a}. Here, $R^{U_a}_{U_b}$ is introduced to S know its data access desire. It turns out that $R^{U_a}_{U_b}$ makes S to know the facts: 1) U_a wants to access U_b's temp authorized data fields D_{U_b}; 2)R_a will also agree to share its temp authorized data fields D_{U_a} with U_b in the case that U_b grants its request.

Afterwards, U_a randomly chooses $U_a \in Z_q^*$, computes a set of values $\{M_{U_a0}, M_{U_a1}, M_{U_a2}\ , M_{U_a3,}\}$ to establish a ciphertext C_{U_a}, and transmits C_{U_a} to S for further request matching.

$$M_{U_a0} = H(sid_{S_a} \| PID_{U_a}) \oplus R_{U_b}^{U_a},$$

$$M_{U_a1} = g^{pk_{U_a} r_{U_a}},$$

$$M_{U_n2} = e(g,h)^{r_{U_a}},$$

$$M_{U_n3} = h^{r_{U_a}},$$

Similarly, U_b performs the corresponding operations, including that U_b extracts A_{U_b} , and determines $\{L_{U_h}, F_{U_h}(x, L_{U_h}), F_{U_h\,i}\}$. U_b further randomly chooses $\beta' \in Z_q$, and computes the value $\{N_{U_a0}, N_{U_a1}, N_{U_a2}\ , N_{U_a3,}\ \sigma'^{l}, M_{U_b}^{l}\}$ to derive its own data fields D_{U_b}. U_b also extracts its pseudonym PID_{U_b} and an access request $R_{U_b}^{U_a}$ to establish C_{U_b} with the elements $\{M_{U_b0}, M_{b1}, M_{U_b2}\ , M_{U_b3,}\}$.

$\{U_a, U_b\}'s$ Access Request Matching Data Access Authority Sharing

Upon receiving the ciphertexts $\{C_{U_a}, C_{U_b}\}$ within an allowable time interval, and S extracts $\{PID_{U_a},\ PID_{U_b}\}$ to derive the access requests $\{R_{U_a}^{U_b}, R_{U_b}^{U_a}\}$.

$$R_{U_a}^{U_b} = H(sid_{S_a} \| PID_{U_a}) \oplus M_{U_a0},$$

$$R_{U_b}^{U_a} = H(sid_{S_b} \| PID_{U_b}) \oplus M_{U_b0},$$

S checks whether $\{R_{U_a}^{U_b}, R_{U_b}^{U_a}\}$ satisfy $F(R_{U_a}^{U_b}(R_{U_b}^{U_a})^T) = F(2) = Cont$. If it holds, S will learn that both U_a and U_b have the access desires to access each other's authorized data, and to share its authorized data fields with each other. S extracts the keys $\{sk_S, pkU_a, pkU_b\}$ to establish the aggregated keys $\{k_S, k_{\Sigma_\theta}\}$ by the Diffie-Hellman key agreement and computes the available re-encryption key k_{U_θ} for U_θ $(\theta \in \{a, b\})$.

$$k_S = (pkU_a\, pkU_b)^{sk_s} = g^{(sk_{U_a} + sk_{U_b})sk_S},$$

$$k_{\Sigma_\theta} = (pk_{U_\theta})^{sk_S} = g^{sk_{U_\theta} sk_S},$$

$$k_{U_\theta} = k_{\Sigma_\theta} / pk_{U_\theta}.$$

S performs re-encryption to obtain $M'_{U_\theta 1}$. Towards U_a/U_b , S extracts U_a/U_b's temp authorized data fields D_{U_b}/D_{U_a} to compute M'_{U_h2}/M'_{U_n2}.

$$M'_{U_\theta 1} = \left(M_{U_\theta 1}\right)^{k_{U_\theta}} = g^{k_{\Sigma\theta} r_{U_\theta}},$$

$$M'_{U_a2} = M_{U_a2} E_{k_{\Sigma_b}}(D_{U_a}),$$

$$M'_{U_b2} = M_{U_b2} E_{k_{\Sigma_a}}(D_{U_b}).$$

Thereafter, S establishes the re-structured ciphertext $C'_{U_\theta} = \left(M'_{U_\theta 1}, M'_{U_\theta 2,} M_{U_\theta 3}\right)$, and respectively transmits $\{\ C'_{U_h} \| k_S, C'_{U_n} \| k_S\ \}$ to $\{\ U_a, U_b\ \}$ for access authority sharing. Upon receiving the messages, U_a computes $k_{\Sigma_a} = (pk_S)^{sk_{U_a}}$, and performs verification by comparing the following equation.

$$e(M'_{U_h1}, \mathrm{h}) \overset{?}{=} e(g^{k_S/k_{\Sigma a}}, M_{U_b3})$$

For the left side of (2), we have,

$$e(M'_{U_h1}, \mathrm{h}) = \mathrm{e}(g^{g^{sk_{U_b} sk_S r_{U_b}}}, h)$$

For the right side of (2), we have,

$$e(g^{k_S/k_{\Sigma a}}, M_{U_b3}) = e(g^{(pk_S)^{sk_{U_b}}}, h^{r_{U_b}})$$

$$= e(g,h)^{g^{sk_S sk_{U_b}} r_{U_b}}$$

U_a derives U_b's temp authorized data fields D_{U_b} .

$$D_{U_b} = E^{-1}_{k_{\Sigma_a}}(M'_{U_h 2} e(M'_{U_h 1}, h)^{-k_{\Sigma_a}/k_S})$$

Similarly, U_b performs the corresponding operations, including that U_b contains the keys $\{k_S, k_{\Sigma_a}\}$, checks U'_b 's validity, and derives the temp authorized data field D_{U_a} .

In SAP, S acts as a semi-trusted proxy to realize $\{U_a, U_b\}$'s access authority sharing. During the proxy re-encryption, $\{U_a, U_b\}$ respectively establish ciphertexts $\{M_{U_a 1}, M_{U_b 1}\}$ by their public keys $\{pk_{U_a}, pk_{U_b}\}$, and S generates the corresponding re-encryption keys $\{k_{U_a}, k_{U_b}\}$ for $\{U_a, U_b\}$. Based on the re-encryption keys, the ciphertexts $\{M_{U_a 1}, M_{U_b 1}\}$ are re-encrypted into $\{M'_{U_a 1}, M'_{U_h 1}\}$, and $\{U_a, U_b\}$ can decrypt the re-structured ciphertexts $\{M'_{U_a 1}$, $M'_{U_h 1}\}$ by their own private key $\{sk_{U_a}, sk_{U_b}\}$ without revealing any sensitive information.

V. RESULTS AND DISCUSSION

The expected results are as follows:

- **Data owner**

  In this module, the data owner should register by providing user name, password, email and group, after registering owner has to Login by using valid user name and password. The Data owner browses and uploads their data to the cloud server. For the security purpose the data provider encrypts the data file and then stores in the cloud server via proxy server. The Owner is also responsible for uploading metadata to the Third Party Authenticator (TPA). The Data owner can be capable of manipulating the encrypted data file.

- **Proxy Server**

  The Proxy server is a proxy-based design that interconnects cloud repositories, as shown in this system. The proxy serves as an interface between client applications and the cloud. The attribute based access control and proxy re-encryption mechanisms are jointly applied for authentication and authorization in proxy server. Proxy server is also acts as an intermediate to perform cloud functions so as keep the cloud server from overwhelmed requests.

- **Cloud Server**

The cloud server is responsible for data storage and file authorization for an end user. The data file will be stored in cloud server with their tags such as Owner, file name, secret key, mac and private key, can also view the registered Owners and End-users in the cloud server. The data file will be sending based on the privileges. If the privilege is correct then the data will be sent to the corresponding user and also will check the file name, end user name and secret key. If all are true then it will send to the corresponding user or he will be captured as attacker.

- **Data Consumer(End User)**

  The data consumer is nothing but the end user who will request and gets file contents response from the corresponding cloud servers. If the file name and secret key, access permission (.java, .txt, .log) is correct then the end user is getting the file response from the cloud or else he will be considered as an attacker and also he will be blocked in corresponding cloud. If he wants to access the file after blocking he wants to unblock from the cloud.

- **Attacker**

  Threat model is one who is integrating the cloud file by adding fake key to the file in the cloud. The attacker may be within a cloud or from outside the cloud. If attacker is from inside the cloud then those attackers are called as internal attackers. If the attacker is from outside the cloud then those attackers are called as external attackers.

VI. CONCLUSIONS

A new privacy challenge is identified to achieve privacy-preserving access authority sharing when a user tries to access cloud data. Data confidentiality and data integrity are achieved using authentication. User privacy is achieved by anonymous access request matching mechanism to privately inform cloud server about the user's data access desires. Proxy re-encryption is applied to provide data sharing among many users. Forward security is achieved by session identifiers. This shows that the proposed protocol SAP is efficient for access authority data sharing in the cloud.

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 Magzines*, vol. 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*, vol. 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] 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. .

[6] 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.

[7] S. Sandareswaran, 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.

[8] C. Wang, Q. Wang, K. Ren, N. Cao, and W. Lou, "Towards Secure and Dependable Storage Services in Cloud Computing," *IEEE Transactions on Services Computing*, vol. 5, no. 2, pp.220-232,2012.

[9] 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.

[10] 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.

[11] 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.

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

[13] Y. Zhu, H. Hu, G. Ahn, D. Huang, and S. Wang, "Towards Temporal Access Control in Cloud Computing," in *Proceedings of the 32st Annual IEEE International Conference on Computer Communications (IEEE INFOCOM* 2012), pp. 2576-2580, March 25-30,2012.

[14] 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.

[15] 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.