Add-on utility to make Google Docs more Secure

Mrs. Shivani S. Kale[1] Prof. G.A.Patil[2]

[1] *(Department of Computer Science/ D.Y.Patil college of Engg. / Shivaji University, India.)*

[2] *(Department of Computer Science/ D.Y.Patil college of Engg. / Shivaji University, India.)*

ABSTRACT*:* With the advent of Web 2.0, end users generate and share more and more content. One such service in this context is the collaborative edition of online documents. This service is commonly provided through Cloud Computing as Software as a Service. However, the Cloud paradigm still requires users to place their trust in Cloud providers with regard to privacy. This is the case of Google Docs; a very popular service without privacy support for the documents stored on its servers so here we discuss the issues and proposes the add-on utility to guarantee privacy of shared documents in Google Docs.

Keywords*-Authentication, cloud, collaboration, GDocs, Time key.*

I.INTRODUCTION

Cloud solutions are scalable and ubiquitous, and follow a pay per use approach at all levels. One of the main barriers to the adoption of Cloud Computing is security. User data are stored on provider servers and there is no guarantee that this information will not be accessible to a third party. This can contravene legal requirements when the stored data are sensitive, as occurs in health care or banking environments. GDocs resource sharing service gives no guarantee that the documents will be safe and secure at the server side. Here we presented a solution for secure online document sharing and secure resource sharing in group.

II.LITERATURE SURVEY

A number of different methods have been proposed to support secure sharing of GDocs. The recent scheme [3] provides solution to share and edit documents in the Cloud thus improving Google Docs. This offers the possibility of working with personal or shared documents using a public Cloud service, preventing access to third parties. But the scheme used to do the encryption or decryption of the documents while storing or sharing needs the password to be shared. Again in the next scheme [3] the writer is also allowed to do changes in the document. The available schemes explained above do not provide any support to reliable group communication as the GDocs is popularly used as collaboration tool. These schemes also bother to keep 'n' number of password to share 'n' number of documents with 'n' users. As Data is stored or shared with users, data is exposed to third party. So some of the researchers have used the following strategies to enhance the real time services

Lilian Adkinson-Orellana et al. [4] have stated that in the new shared index, document must contain data related to the encryption of the shared document. This data will be created as a hidden file. When an owner shares a document, the new shared index will contain the information associated with the encryption of the document copied from the general index. The content of this shared index is also ciphered using AES with a 128- bit key. The password to encrypt the index will be the shared key, which will preferably be different from the master key. Accordingly, when an owner shares a document he will simply have to give the shared key to the rest of the users. So that the master key will remain private and safe as shown in Fig1.

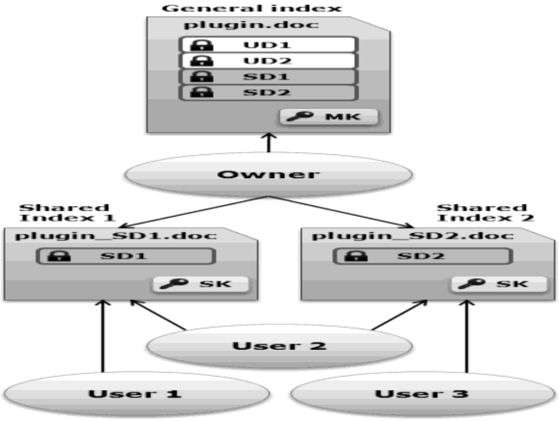


Figure 1: Private and shared indices of the owner of the documents.

But the scheme can further be improved by removing the necessity of keeping both general index and shared index synchronized.

Gabriele D'Angelo et al. [5] have proposed that in Content cloaking (CoClo) content is protected from unauthorized accesses, service providers and third parties. The transmission of clear text content is avoided when the web application does not support secure transmission protocols .With CoClo server has access to encrypted data only.

This scheme supports AJAX based browser, some global add-on required to support the content cloaking.

Daniel A. Rodríguez-Silva1 et al. [3] have presented a new security mechanism for SaaS applications of Google Docs service to have an additional privacy layer to protect their documents. This scheme needs the user's password to encrypt the documents. So if the user forgets the password the information cannot be recovered. This application is currently being improved with the possibility to share encrypted documents with other users. The only condition that all users have installed the Firefox add-on and know the shared password.

III. LIMITATIONS

From the above survey we can identify limitations in GDocs are

1. Password is shared.
2. Other than Owner writer is also allowed to modify the Document
3. Operations are not performed on client side.
4. Group communication is unreliable.
5. N number of passwords is required for sharing n documents.
6. Service provider is able to see the data.

IV.RESEARCH WORK

In the proposed work, the user in the access control lists (acl) is allowed to share the resources via an external storage service, with a desired group of other users. The proposal guarantees that only users in the specified group will be able to access the resources. The resources will remain confidential to all the other parties, including the service itself. The key used to protect a resource can be derived from a secret held by each user .A variation of Diffie-Hellman key agreement method and public tokens are used. The service offered is realized by users desiring to exchange confidential resources compared to existing applications. This approach offers stronger guarantee in terms of protection of resource confidentiality. The approach is fully compatible with the design of cloud storage applications as shown in figure(2).

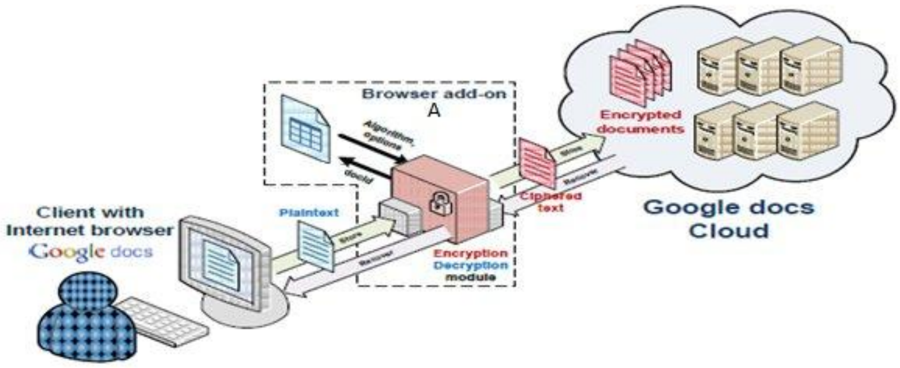


Figure 2: Secure sharing of documents in Google Docs using Add-on Utility

The proposed system (A) will be designed in the following way:

1. Key Agreement Mechanism
   In the key agreement method Slight Variation of Diffie-Hellman (DH) key Exchange agreement method will be used. In this the two involved parties do not directly interact for computing the common secret but they interact with the external service.

Variation of the DH method works as follows.

Let (G, □) be a public algebraic cyclic Group of prime order q =| G | and □ be the internal operation of the group with multiplicative notation.

We assume that G is generated by an element g Z$_p$, where $p = 2q + 1$ and p, q two prime integers such a way that q=| G | and G= *{* g^e mod *p:* $0 \leq e \leq q - 1$*}*.Each user u *U* chooses a secret integer parameter e_u [0, q −1], computes the value g^{eu} G, and inserts g^{eu} in a public catalog managed by the external service.

2. Key agreement function

Once we have generated the key between users from above scheme and whenever user *u* needs to share a common secret with user u_i, user *u* can efficiently compute such a secret by querying the public catalog to retrieve

the public parameters g^{eui} and q, and by applying key agreement function.

Derivation of keys using Tokens

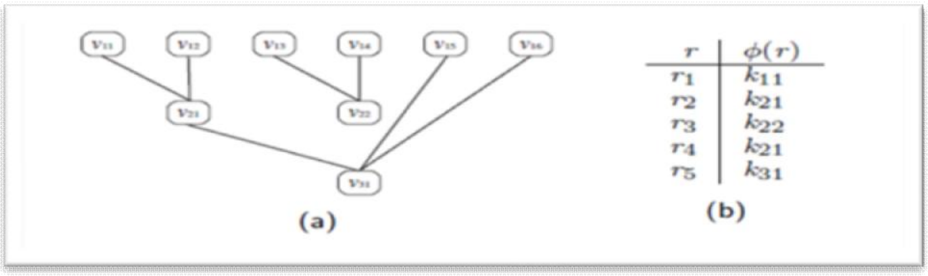
From module 1 key for users to share is agreed, using those keys and tokens we will derive another keys for secure sharing.

Using key derivation method we will be computing key starting from the value of another key and a publicly available piece of information, called token. Given a set K of keys and

k$_i$, k$_j$ □ K, a token t$_{i,j}$ between them is defined as t$_{i,j}$ = E$_{ki}$ (k$_j$) as shown in fig 3(a) Once the keys are derived, it

will be used to encrypt resources using key assignment function from fig 3(b) that determines the keys used for encrypting resources.

Key Assignment Function: Given a set R of resources and a set K of keys, the key assignment function □ : R □□ K associates with each resource r R the (single) key with which the resource is encrypted.

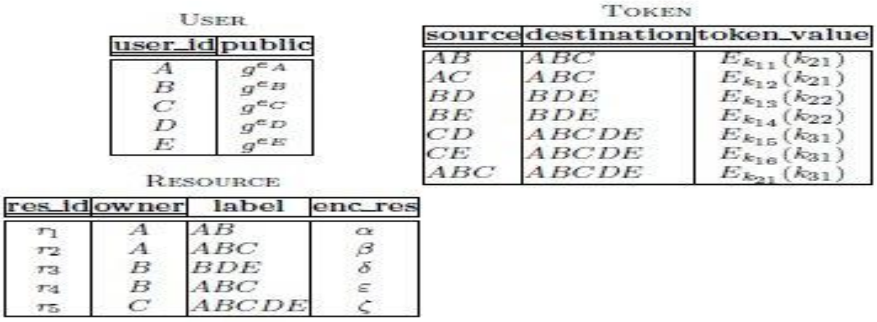


r	$\phi(r)$
r_1	k_{11}
r_2	k_{21}
r_3	k_{22}
r_4	k_{21}
r_5	k_{31}

Figure 3: (a) An example of key and token graph (b) and key assignment function

After encrypting the resources using key assignment function, the encryption policy will be used.

Encryption policy: An encryption policy using Fig (4) regulates which resources are encrypted with which keys and which keys can be directly or indirectly computed by which users.



USER

user_id	public
A	g^{e_A}
B	g^{e_B}
C	g^{e_C}
D	g^{e_D}
E	g^{e_E}

TOKEN

source	destination	token_value
AB	ABC	$E_{k_{11}}(k_{21})$
AC	ABC	$E_{k_{12}}(k_{21})$
BD	BDE	$E_{k_{13}}(k_{22})$
BE	BDE	$E_{k_{14}}(k_{22})$
CD	$ABCDE$	$E_{k_{15}}(k_{31})$
CE	$ABCDE$	$E_{k_{16}}(k_{31})$
ABC	$ABCDE$	$E_{k_{21}}(k_{31})$

RESOURCE

res_id	owner	label	enc_res
r_1	A	AB	α
r_2	A	ABC	β
r_3	B	BDE	δ
r_4	B	ABC	ε
r_5	C	$ABCDE$	ζ

Figure 4: An encryption policy catalog

V.Implementation of Authentication and Authorization algorithm

Once the keys are generated and the keys for resource encryption are decided using the above module we will design the Authentication functionality that allows data owners to compute the digest, sign (DSA), and Correctly encrypt their resources and to deliver the encrypted resources to the service for their management. Each user is only able to first create and then use token chains whose starting points are the root vertices corresponding to keys that user can compute through modified version of Diffie-Hellman computations. Therefore, whenever user needs to share a resource r with other users in the system, the user must first encrypt r with a new key and then must add the appropriate tokens that the other users in acl(r) can use to derive the new key.

The authorization functionality will be implemented to allow users to retrieve the resources that they are authorized to access and to verify their signature. In particular, every time an authorized user *u* needs to access a resource *r*, the service has to deliver the encrypted resource to *u* along with a token chain ending to the vertex representing *acl*(*r*), which the user follows to derive the decryption key. User *u* can then decrypt the resource and use the public Diffie-Hellman parameter of owner(r) to verify the signature of *r*.

VI. Implementation of Combined Group key and Time Key Server using Hierarchical tree of Time Key

In this module the group and time key will be generated by server for secure group communication as we are using GDocs as the collaboration tool .This method is used basically in the concept of **Broker architecture for Publish Subscribe Scenario** [2] and we will be using this for secure group communication for GoogleDocs to share documents securely.
The following two server services will be implemented to generate Group key and Time key. Group key This key will be generated according to the groups as shown in figure (5)

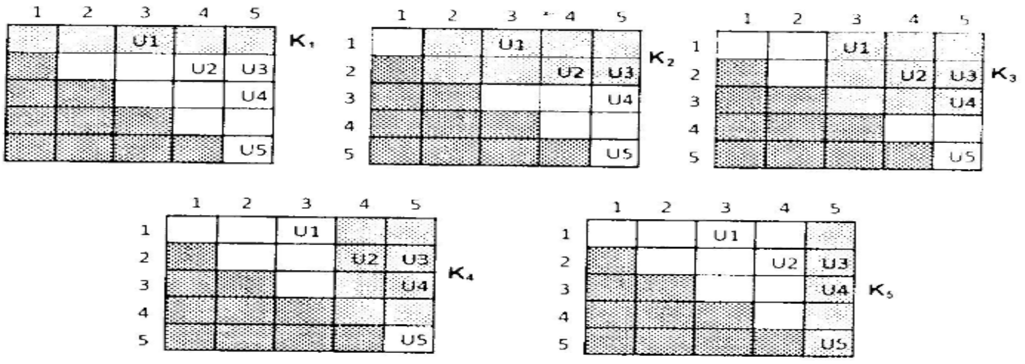


Figure 5: Example of Group Key

2) Time Key:

In this the server will generate the time key for the particular session for the group of user so that for every session avoids the key updating every time a new user joins or leaves the group.

$K_E = K_i$ □ $k^{t,\, t}$

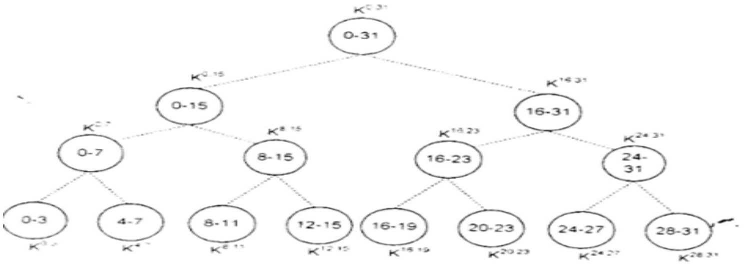


Figure 6: Hierarchical tree of Time Key

VII.CONCLUSION

The proposed feature will enable the secure access over GDocs. It also increases the usage of GoogleDocs by user as security is increased. The time key concept will provide very secure group collaboration for highly secured communication.

REFERENCE

[1] De Capitani di Vimercati, S. DTI,Univ. degli Studi di Milano, Crema, Italy Foresti, S.; Jajodia, S.; Paraboschi, S. Pelosi, G. ; Samarati, P. “Encryption based Policy Enforcement for Cloud Storage” in Distributed Computing Systems Workshops (ICDCSW), 2010 IEEE 30th International Conference on 21-25 June 2010, PP: 42 – 51.

[2] Tien-Dung Nguyen Dept. of Comput. Eng., Internet Comput. & Security Lab., Suwon, South Korea Eui- Nam Huh“An Efficient key MANAGEMENT FOR SECURE MULTICAST IN SENSOR CLOUD “Computers, Networks, Systems and Industrial Engineering (CNSI), 2011 First ACIS/JNU International Conference on 23-25 May 2011 PP: 3 – 9.

[3] Lilian Adkinson-Orellana1, Daniel A. Rodríguez-Silva1, Felipe Gil-Castiñeira2, Juan C. Burguillo- Rial2. “Privacy for Google Docs: Implementing a Transparent Encryption Layer” www-gti.det.uvigo.es/~darguez/pub/2010_CloudViews_GoogleDocs.

[4] Lilian Adkinson-Orellana, Daniel A. Rodriguez-Silva, Francisco J. “Sharing Secure Documents in the Cloud. A Secure Layer For GoogleDocs” Proceedings of 1st International Conference on Cloud Computing and Services ... www-gti.det.uvigo.es/~darguez/publications1.html.

[5] Gabriele D’Angelo Fabio Vitali University Bologna Italy, “content cloaking: preserving privacy with Google Docs and other web applications” proceedings of the 2010 ACM symposium on Applied Computing .PP: 826-830, ACM New York.