SUPPORTING CONFIDENTIALITY IN CLOUD DATABASES AND ROLE BASED ACCESS

[1]Anupama P L, [2]Anver S R

[1]M.Tech Scholar, Dept. of CSE, LBSITW, Poojappura, Thiruvananthapuram, India
[2]Associate professor, Dept. of CSE, LBSITW, Poojappura, Thiruvananthapuram, India

Abstract : Cloud computing provides data storage mechanism to access the data anywhere and anytime. Storing the personal data in the cloud causes the problem of confidentiality. The user does not get any idea about where the data resides. In most cases the vital and secret information are in the hands of untrusted third parties i.e., the cloud providers. Encryption is one of the methods available for providing confidentiality. But if anyone gets that key he can decrypt all the data. Here we propose a novel scheme that integrates the encryption of the critical data with user access control mechanisms with multiple decryption keys. The scheme guarantees the confidentiality of the data in the public cloud and the possibility of performing independent and concurrent access of encrypted cloud database. Here both encrypted data and the encrypted metadata are kept in the cloud. The latter can be decrypted only by authorized clients. Encrypted data is then accessed and the plain text is obtained through queries. This scheme also reduces the risk of internal data leakage.

IndexTerms - Cloud, Security, Confidentiality, Secure File Access mechanism.

I. INTRODUCTION

Cloud computing [1] [2] is an emerging technology that change the way we use our computer and Internet. Instead of running programs and data on an individual system, everythingis hosted in the cloud. The cloud system has several security vulnerabilities. Among them confidentiality violation has vital importance. User stores their critical data in cloud to access from anywhere and anytime. The information that is placed in the cloud denoted as tenant data. Since these data is located away from the user system, security of the data is completely depending on the cloud service provider (CSP). CSP is not always a trusted entity. Unauthorized access of user data may cause a serious issue.
Cloud provider offers DBaaS [3] for storing and managinguser data. Placing users' confidential data in a remote machinemay cause confidentiality violation. Cloud provider may itself act as an attacker. So it is an important concern that only the authorised person can access user data. From the existing studies [4] we can conclude that whole user data are encryptedby a master key and stored it in the cloud. Encrypted metadata are placed in metadata storage table and which is also stored inencrypted database. Decrypting the data is a time-consuming procedure. The response time is affected by cryptographicoverhead. Moreover use of master key may cause internalinformation leakage. Adopting the advantages from the current scenario, we enhance the confidentiality of vital data inthe cloud database. For reducing cryptographic overhead, here encrypt the essential details only. Furthermore, we use multiplekeys instead of a master key with different levels of access control mechanism. Thus we can reduce the internal information leakage.

Our aim is to support confidentiality of data located in the untrusted cloud. For this encrypt the data before outsourced. Encrypting with a master key may cause some confidentiality issues. That is, if the key is lost or any client machine compromised the entire data in the cloud database is at risk. Here we introduce multiple keys concept and each client has its own limit for accessing the database depending on their rolethey played. Client can also perform independent and concurrent access of data in encrypted cloud databases.
This paper is organized as follows. Section 2 includes theliterature survey which is a summary of works related to this topic that were referred for the completion of this work. Section 3 describes the architecture of the proposed system and design details. Section 4 provides the operation and section 4 reports some test results.

2. LITERATURE REVIEW

This scheme provides several existing properties that differentiate it from previous studies in the field of confidentiality of data reside in remote database.

The following requirements are satisfied in this proposed work.

- R1 - Any confidential data outside the tenant domain mustbe encrypted.
- R2 - Users must be able to access only authorized dataaccording to SFA.
- R3 - User can only download the data after he/she gets thekeys.
- R4 - Encrypted tenant data stored in the cloud allow theexecution of SQL operations.

For guaranteeing confidentiality the data outsourced must be encrypted. Encryption is the process of encoding the data in such a way that only authorized users can read it. This conceptis defined in R1. If any user key is leaked or compromised, a malicious subject cannot get information about the data that is inaccessible to that user, which is described in R2. To avoid information leakage multiple keys are deployed. Each metafilehas different keys for decrypting. No master key concept is used here. This idea is discussed in R3. User can retrieve their data though SQL quires and performs independent and concurrent access on encrypted data which is detailed in R4.

There is no existing proposals are available for satisfying all these requirements. Here outline some related works. The encryption scheme [5] described some encryption schemes on SQL operations and satisfies R1 and R4, but it neither enforcesthe database access control mechanisms that described in R2 nor the multiple key concepts in R3. Architecture [6] shows distributed concurrent and independent access on encrypted database without any intermediate proxies, thus satisfying R1 and R4. But it uses a master key for encrypting the data. Different encryption methods used in [7] satisfying R1 and R2but not included the execution on encrypted databases. The idea in [8] integrate the execution of SQL query on encrypted data in the cloud with the access control policies satisfying R1,R2 and R4, but not discussed about R3

At first all the data are stored in cloud as plaintext form. But it faces the problems and challenges of trusted cloud, because a malicious user can access the entire data without knowing the actual user and put duplicate data. Ryan K L Ko et.al proposeda Trust Cloud Framework [9], which put some controls in cloud environment for preventing unauthorized access.

In 2005 Hacigumus H., Iyer B., C.Mehrotra S, introduce a proxy between the client and the cloud [3]. Proxy mediates all the interactions between them. The proxy is managed locally and which is considered as a trusted entity. Cloud stored the encrypted tenant data. Here both metadata and the client data are in the hands of proxy. Furthermore the encryption engine isalso executed by the proxy. In short all the interactions are based on the proxy, so there is a problem of bottleneck and single point of failure may occur here.

Damiani E., De Capitani di Vimercati in 2007 proposed a new architecture [6] which does not contain an external proxy.Client manages the local copy of metadata and each client has a proxy like architecture. This proposal is considered as a sub case of proxy based architecture because different proxies in each client. But metadata consistency and synchronization algorithms are needed here for concurrent and independent access of cloud data.

In 2012 L. Ferretti, M. Colajanni, and M. Marchetti introduce their work [10] which does not contain any proxy like architecture. The main focus is here is to put metadata incloud. For security the metadata is also encrypted. But the issue is occurred for guaranteeing consistency among encrypted data and metadata.

SecureDBaas is introduced by Luca Ferretti, Michele Colajanni, and Mirco Marchetti in 2014 [4]. Here metadata stored in client and encrypted metadata in cloud. Thus user canperform concurrent and independent access of cloud data. For cryptographic security a master key is used here. Only clients who know the master key can decrypt the data. But the problem is that if the master key loses, then the entire data in cloud might be at risk.

3. ARCHITECTURE

Figure1 shows the overall architecture of the proposed system.

Consider the typical scenario in which a tenant organization requires a database service from a public cloud DBaaS provider. In the tenant, there is an administrator role and multiple database users. The admin is a trusted subject. He hascomplete access on all database information, and is in charge of enforcing the Secure File Access (SFA) of the tenant users. Each tenant user has a different level of trust and a consequentauthorization to access a specified metafile of the database information. His database view is limited by the SFA that are implemented through authorization mechanisms by the admin.The cloud provider may not be a trusted entity. This security model is known as honest-but-curious. Here assume that the cloud employees execute the database protocols and mechanisms correctly and do not modify the data stored and result of the query. But they could try to access tenant information stored in the cloud database.

The confidentiality is achieved by encrypting the data stored inthe cloud. Admin uploads the encrypted metafiles. The authorized tenant users get the decryption key from admin through email and can download the metafiles. Using thatmetafile they can write query and access the database. Here weencrypt the confidential data only. This scheme has further benefit of allowing clients to access each metadata independently and concurrently.

Assume that tenant users are trusted, the network is untrusted and the CSP is honest-but-curious. CSP executes the operation correctly but the confidentiality of the tenant data is at risk. Encryption is one of the best solutions for preventing unauthorized access. Before exiting from the client side the tenant data and metadata must be encrypted.

In this scheme the secure database includes plaintext data, encrypted data, metadata and encrypted metadata. Theinformation that a tenant wants to store in cloud and executeremotely is called plaintext data. Storing the tenant data inplaintext form is a risky factor because CSP may violate theconfidentiality of user data. So the tenant data must beencrypted by the client machine. Here only encrypt theessential details.

Consider the tenant data are stored in a relational database. To preserve confidentiality encrypt the database structure because even the name of tables and their column name yield the idea about the data saved. For decrease the cryptographic overhead encrypt the necessary fields only. The plaintext table is converted into secure table by encrypting the name of the table. The encryption algorithm and the key are same as that ofthe saved encrypted data.

Field confidentiality defines which columns of the secure tableshare the same encryption key (if any). DBC describes that the entire column in the database share same encryption key. If each column has separate encryption key then its field confidentiality is COL. On the other hand multiple columns share same encryption key, it is MCOL field confidentiality. Inthis scheme we use DBC as field confidentiality.

Admin create metafile for database and each table. This metafile consisting of information required to access the data from the cloud. Metafiles are encrypted and stored in the clouddatabase. Authorized clients can retrieve the required metafile from the cloud through SQL statements. So multiple instances of the clients can access file from the untrusted cloud independently with the guarantee of same availability elasticityand scalability of the typical cloud service.

In this scheme two types of metafiles are used - Database metafiles and Table metafiles.

Database metafiles are related to whole database which includes plaintext name and encrypted name of database and all tables in that database. There is only one instance of this metafile is exists.

Table metafiles are associated with one secure table. Each table metafile consist of all necessary information about that table which includes all column names, if the column is encrypted then that encrypted name and the data types of each column.

Figure 2 shows the table metadata file structure. It contains thename of secure table and the plaintext table name. Furthermore it includes the column name of the related secure table. If the column is encrypted the encrypted name is also specified

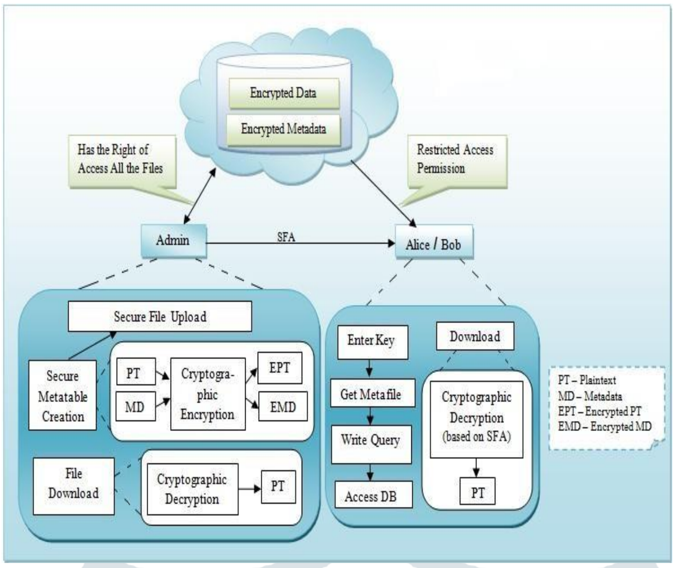

Figure 1: Overall Architecture

Besides these details, the file contains the database plain nameand encrypted name too.

This design choice is required to identify which metafile is needed for executing the SQL query so that the client needs to fetch only the essential metadata file related to the securetable. This choice has an advantage that it reduces the amount of metadata that each client has to fetch from the cloud which decreases the bandwidth consumption and processing time. It allows geographically distributed clients to access the tenant data concurrently and independently with the help of these metadata files. But the metadata do not represent any tenant information.

The tenant users can access all and only authorized data that resides in cloud database. The authorization is defined by SFA mechanism. Admin give permission to each user for accessingthe data. Here we propose an encryption scheme that binds theaccess control mechanism by encrypting the tenant data with multiple keys.

We can explore SFA in the following way. Suppose *U* is the set of users who have the right to access the data. *F* denotes setof metafiles and *A* is access matrix. Each row in the access matrix represents a user and each column is a metafile. The matrix entry is 0 or 1 represents the access right of the user to access the metafile. For example if access to *f* by *u* is denied then the matrix entry is represented as ($a_{u,f} = 0$) and ($a_{u,f} = 1$) defines the access of *f* by *u* is allowed. Constraint list *con* is the set of files to which user *u* has the constraint access.Sanction list *san* is another list which holds the files that the user *u* has the access permission.

P represents the set of plaintext data. *C* represents the set of encrypted cipher text and *K* is the set of keys used for cryptography. Then we can define, Encryption $E = K.P \rightarrow C$. For each encrypted data/file $c \in C$ there exists a key $k \in K$ then we can calculate, $p = D(k,c)$ where $p \in P$. In access matrix if any field has a value 1 shows that the corresponding user can decrypt the corresponding metafile with the help of some keys. For example, consider Table 1.

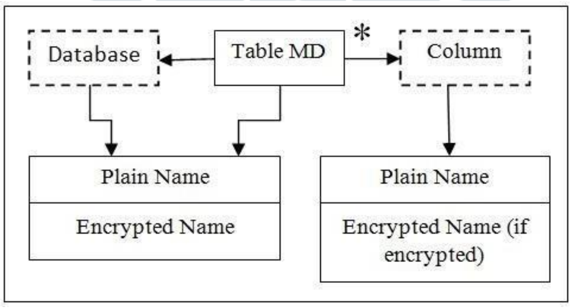

Figure 2: Structure of Table Metadata

Here u_x can decrypt f_a by using k_x & k_a and decrypt f_b by k_x & k_b. k_x is assigned to user u_x by admin at the time of approval. k_a & k_b are allotted when the user u_x gets the access permission ofthat file. k_u is the set of all decryption keys that user *u* owns, represented as $k_u \subseteq K$ and p_u is the set of all and only plaintext files that *u* able to decrypt using the keys in k_u.

We can express it as follows:-

$\forall u \in U, \forall p_u \in P, k_u \in K \Leftrightarrow p \in (con \wedge san)$

Table 1: Access Matrix

	f_a	f_b	f_c
u_x	1	1	0
u_y	1	0	1

In this scheme the metadata stores in metadata storage table
[4] and it is placed in untrusted cloud. For ensuring security the metadata is also in encrypted form. Each metafile is encrypted through different keys. Only the trusted clients who get the key can decrypt the metafile and acquire information that is necessary to access the database information. Eachmetafile can be obtained by the clients through an ID which is the primary key of metadata storage table. If the secure clientsknow the name of the object (database or/and table) the ID is generated for fetching the encrypted metafile. The authorised clients who already know the key can decrypt the file and access the tenant data from the encrypted database. In additionsecure clients can use caching policies to reduce the bandwidthoverhead.
Structure of metadata storage table is shown in figure 3

ID	Encrypted Metadata
ID1	Enc(Db metadata)
ID2	Enc(Tbl1 metadata)
ID3	Enc(Tbl2 metadata)
⋮	⋮

Figure 3: Structure of Metadata Storage Table

4. OPERATION

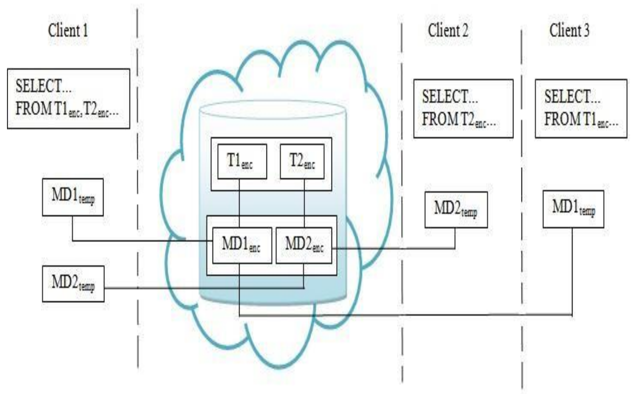

Figure 4: Operation on Cloud Database

At first, the client connect with the cloud database is for authentication and authorization purpose. This is similar to theoriginal DBMS server. After the authentication, secure clients can interact with the cloud.
Supposes the database consist of two tables T1 and T2. They are stored as encrypted in cloud database, represented as T1enc and T2enc. Each table is associated with a metafile MD1 and MD2, which are independent of each other. All metadata files are encrypted and stored in database as MD1encand MD2enc. Client 1 executes queries on tables T1 and T2. Itobtains MD1enc and MD2enc, decrypts them and maintains temporary version of metafiles in local system as MD1temp and MD2temp. Client 2 executes queries on table T2 hence it acquires MD2enc and maintains MD2temp. Client 3 executes queries on table T1 hence it retrieves MD1enc and keepsMD1temp. The clients access the database concurrently and independently from each other since they handle independent metafile. The following figure 4 shows the metadata operation with the cloud database.

The operation identifies which tables are involved and to access their metafile from the cloud database. The metafile is decrypted through the key that is provided by admin and the information is used to write queries for accessing theencrypted database. The operation contains neither plaintext database (tables and column names) nor plaintext tenant data. Nevertheless, they are valid SQL quires that the client can issueto the cloud database. The operation is executed over the encrypted tenant data. The complexity of the process depends on the type of SQL statements. As the metadata do not change a client can read them once and cache them for further use, thus improve the performance.

5. RESULT

The new scheme has several advantages over the papers reviewed in the literature survey. CSP may not be a trusted entity. Here we introduce more security in confidentiality for data in cloud. All data outsourced from the client side must be encrypted. Instead of a master key here different decryption keys are used for different metafiles in cryptographic algorithms. We store both tenant data and metadata in cloud asencrypted form. Moreover, we encrypt only the essential details. Figure 5 shows the graphical representation ofattributes Vs cryptographic overhead.

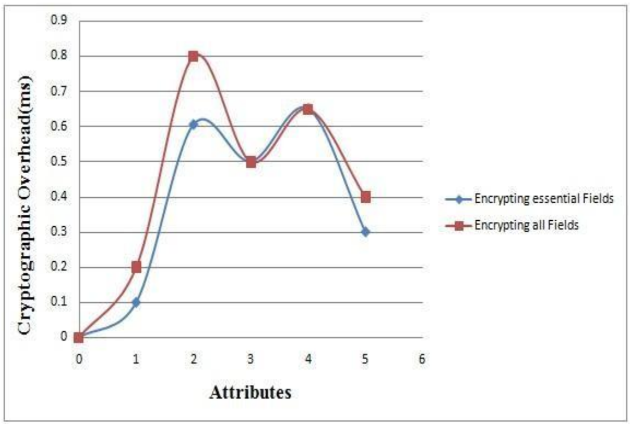

Figure 5: Attributes Vs Cryptographic Overhead

Our main focus is supporting confidentiality for retrieving theencrypted database information. For this decryption is done with the help of multiple keys and retrieval of database is depends on the role based access permission. Tenant user who knows the object name can generate the unique ID and gets theencrypted metafile. Decrypt the metafile using the key shared by admin. Using the metafile user write the query for accessingthe data in cloud database. Even though the essential parts are encrypted the query is valid. The authorized user got the resultas plaintext form. Figure 6 shows the graphical representation of time taken for plain text and cipher text.

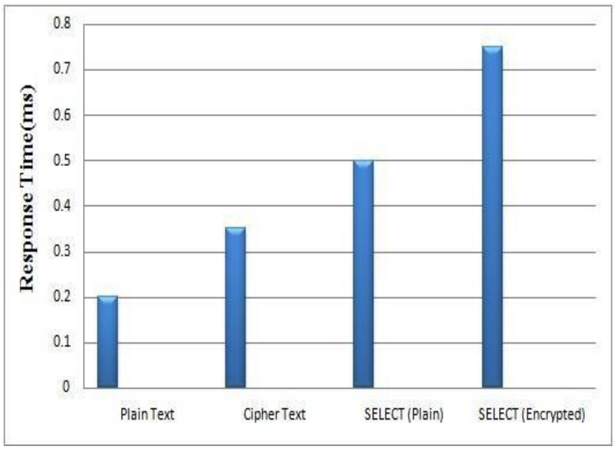

Figure 6: Comparison of Response Time for Plaintext andCipher text

For improving the performance cache the downloaded metafiles. Figure 7 shows the comparison of time taken when downloading from cloud and cache.

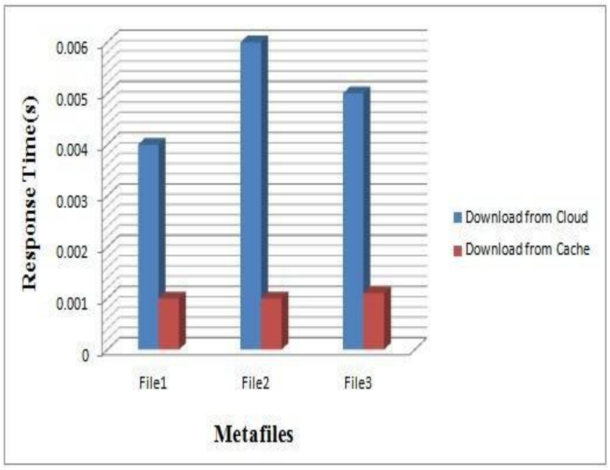

Figure 7: Comparison of Response Time when downloadingfrom cloud and cache

6. CONCLUSION

Cloud computing is an emerging technology to store and sharethe data publically where ensuring the security is of paramount importance. We proposed an innovative architecture that guarantee confidentiality with role based access of encrypted data located in cloud. Moreover our proposal does not rely ona master key that may leads to confidentiality violation. The architecture allows

the organization to encrypt their essential details and store it in encrypted database, which contains both encrypted data and encrypted metadata. Metadata file containsthe essential details for accessing the database.
Encrypted metafiles are stored in metadata storage table. Legitimate users are supplied with different decryption keys for different metafiles. This scheme has the advantage that it does not allow the users to write and modify the data in the cloud database. It reduces the internal information leakage by not relying on any intermediate server. Concurrent and independent access of the encrypted cloud database is also the highlight of the proposed system.

7. FUTURE WORK

The proposed system will not allow modifying the cloud databases. Concurrent read and write operation can cause data consistency issue which depends on the type of SQL queries that are executed. It also increases the overhead. We can improve and modify the current scheme for an application such as uploading the personal details (including id numbers, certificate details, etc) to cloud storage and give permission to others to access those details for any verification procedure. Not allow others to modify the details. The field confidentiality of the current architecture is DBC. That is samekey is used for all the columns in all tables. The confidentialityis improved by using COL or MCOL field confidentialities. InCOL each column has separate key for cryptographic algorithms. On the other hand in MCOL same column in different table use same key for cryptographic protection.

ACKNOWLEDGEMENT

I feel unequivocal gratification in acknowledging the service and co-operation rendered by Mr. Anver S.R., Associate Professor, Department of Computer Science and Engineering, for his valuable suggestions and guidance throughout the completion of my work. I also acknowledge with grateful thanks the authors of the references and other literature referred to in this paper.

REFERENCES

[1] **P. Mell and T. Grance,“Draft NIST working definition of cloud computing,” Online at http:// csrc.nist.gov/groups/SNS/cloud- computing/ index.html,Referenced on June. 3rd, 2009.**

[2] **M. Armbrust, A. Fox, R. Griffith, A. D. Joseph, R. H. Katz, A. zonwinski, G. Lee, D. A. Patterson, A. Rabkin, I. Stoica, and M. Zaharia, “Above the clouds: A berkeley view of cloud computing,”University of California, Berkeley, Tech. Rep. UCBEECS-2009-28, Feb 2009.**

[3] **Hacigumus, H., Iyer, B., Mehrotra, S., “Providing database as a service,” In: Proceedings of the 18th International Conference on Data Engineering, pp. 2938 (2005).**

[4] **Luca Ferretti, Michele Colajanni, and Mirco Marchetti, ”Distributed, Concurrent, and Independent Access to Encrypted Cloud Databases,”IEEE transanctions on parallel and distributedsystems, Vol. 25, No. 2, pp 437-446, Feb 2014**

[5] **R. A. Popa, C. M. S. Redfield, N. Zeldovich, and H. Balakrishnan, ”Crypt DB: protecting confidentialitywith encrypted query processing”, in Proc. of the23rd ACM Symposium on Operating Systems Principles, October 2011**

[6] **Damiani, E., De Capitani di Vimercati, S., Foresti, S., Jajodia, S., Paraboschi, S.,Samarati, P,”Metadata management in outsourced encrypted databases,” Secure Data Management, pp. 1632, 2007**

[7] **E. Damiani, S. De Capitani di Vimercati, S. Foresti, S. Jajodia, S. Paraboschi, and P. Samarati, ”Keymanagement for multi-user encrypted databases,” InProc. of the ACM workshop on Storage security andsurvivability, November 2005**

[8] **Ferretti, Luca, Michele Colajanni, and Mirco Marchetti, ”Access control enforcement on query- aware encrypted cloud databases”, IEEE 5th International Conference on Cloud Computing Technology and Science (Cloud Com), Vol. 2. 2013**

[9] **Ryan K L Ko, Peter Jagadpramana, Miranda Mowbray, Siani Pearson, Markus Kirchberg,Qianhui Liang , Bu Sung Lee, ”TrustCloud: A Framework for Accountability and Trust in Cloud Computing,” Secure Data Management, pp. 1632,2007**

[10] **L. Ferretti, M. Colajanni, and M. Marchetti,” Supporting Security and consistency for CloudDatabase,” Proc. Fourth Intl Symp. Cyberspace Safety and Security, Dec 2012.**