International Journal on Recent and Innovation Trends in Computing and Communication ISSN: 2321-8169
Volume: 2 Issue: 6 1423 – 1428

Intranet Cloud Security

Lekshmy.D.Kumar
M.Tech Student, Dept of MACS
National Institute of Technology
Surathkal,Mangalore, India
lechuvinan@hotmail.com

B.R.Shankar
Associate Professor, Dept of MACS
National Institute of Technology
Surathkal,Mangalore, India
brs@nitk.ac.in

***Abstract*--**Cloud computing has been envisioned as the next-generation architecture of IT enterprise. In contrast to traditional solutions, where the IT services are under proper physical, logical and personnel controls, cloud computing moves the application software and databases to the large data centers, where the management of the data and services may not be fully trustworthy. This unique attribute, however, poses many new security challenges which have not been well understood. Data Security is one of the most critical aspects in a cloud computing environment due to the sensitivity and importance of information stored in the cloud. A basic method to secure data in cloud is to store encrypted data in the cloud by the data owner, and issue decryption keys to the authorized users. The data owner will issue reencryption commands to the cloud to reencrypt the data when a data user is revoked, to prevent the revoked user from decrypting the data and to generate new decryption keys to valid users, so that they can continue to access the data. Since the cloud computing environment consists of many cloud servers, such commands may not be received and executed by all the cloud servers due to network crash problems. This problem can be solved by using time based re-encryption schemes, which enables the cloud server to automatically re-encrypt data based on the internal clocks. This time based re-encryption is based on attribute based encryption scheme to allow fine grained access control and does not require prefect clock synchronization for correctness.

Keywords: *Cloud Computing, Attribute based encryption, Time-based re-encryption*

*****

I. INTRODUCTION

Cloud Computing is a model for delivering information technology services in which resources are retrieved from the Internet through web based tools and applications, rather than direct connection to the server and resources are provided as long as connected to the web. It has become a viable business and technological proposition because of the significant reduction in both infrastructure and operational costs that it offers when compared to traditional IT services. As cloud computing is increasing its popularity, concerns are being raised about the security issues associated with the adoption of the new model. Data Security is one of the most critical aspects in a cloud computing environment due to the sensitivity and importance of information stored in the cloud. Data resides in resources not in the purview of data owner or one which could be accessed by cloud administrators depending on implementation. The data is encrypted and stored in the cloud by the data owner. Attribute Based Encryption [9] scheme is one of the reliable techniques used to encrypt the data for fine grained access control.

A. CLOUD COMPUTING

According to NIST(National Institute of Standards and Technology), Cloud Computing is a model for enabling ubiquitous, convenient, on-demand network access to a shared pool of configurable computing resources(eg: networks, servers, storage, applications and services) that can be rapidly provisioned and released with minimal management effort or service provider interaction.

B. ATTRIBUTE BASED ENCRYPTION

Attribute based encryption is based on Elliptic curve cryptography (ECC).Elliptic curve cryptography (ECC) is an approach to public-key cryptography based on the algebraic structure of elliptic curves over finite fields. The primary benefit promised by ECC is a smaller key size, reducing storage and transmission requirements, i.e. that an elliptic curve group could provide the same level of security offered by an RSA-based system with a large modulus and correspondingly larger key – e.g., a 256-bit ECC public key should provide comparable security to a 3072-bit RSA public key . For current cryptographic purposes, an elliptic curve is a plane curve which consists of the points satisfying the equation $y2=x3+ax+b$ along with a distinguished point at infinity, denoted α.

II. RELATED WORK

The use of cloud computing is increasing popular due to the potential cost savings from outsourcing data to the cloud service provider (CSP). Cloud computing has challenges in addition to the benefits. The biggest challenge is to protect the data that is stored on cloud servers. For securing the data stored in the cloud the data owner can encrypt the data and can issue decryption keys to the authorized users. The key problem of storing the encrypted data in the cloud is that in revoking access rights from users. A user whose permission is revoked will retain the keys which are issued earlier and can open the data using those decryption keys.

Many researchers have proposed encryption of data in cloud to protect from CSP [1][2]. Using this approach a third party re-encrypts data such that the previous keys are no longer valid to decrypt the data. Another solution is to let data owner immediately re-encrypt the data, so that the revoked users cannot decrypt the data using their old keys, while distributing the keys to the remaining authorized users. But this solution will lead to a performance bottleneck, especially when there are frequent user revocations. The solution by [3]-[5] is to let the data owner to issue re-encryption key to an untrusted server to re-encrypt the data. This solution uses the Proxy Re-encryption Technique (PRE)[4], such that the server can re-

encrypt the ciphertext to a different ciphertext that can only be decrypted using another decryption key. The main advantage is that during this process the server does not learn the contents of the ciphertext or the decryption keys.

Attribute Based Encryption (ABE) is relatively a new public key cryptographic technique in which the secret key of user and ciphertext are dependent on the attributes. It supports fine-grained access control. The combination of ABE and PRE was introduced by [6] and extended by [7] and[8]. In [7], a Hierarchical Attribute Based Encryption (HABE) scheme is proposed to achieve high performance and full delegation. Work by [12] proposed a clock synchronization scheme for cloud environments, which uses an authoritative time source shared by all participants in a transaction to achieve clock synchronization between virtual cloud policy enforcement points.

Our scheme mainly relies on time of the cloud server to re-encrypt the data. The main difference between prior work and ours is that it does not depend on the cloud underlying structure and we do not require the underlying cloud structure to be reliable in order to ensure correctness. Moreover for this work proxy re-encryption is not used.

III. BACKGROUND

A. ATTRIBUTE BASED ENCRYPTION

Attribute based encryption is a type of public key encryption in which the secret key of a user and the ciphertext are dependent upon the attributes (eg: the country he lives in or the kind of subscription he has). In such a system decryption is possible only if the set of attributes of the user key matches the attributes of the ciphertext. A crucial security feature of attribute based encryption is collision-resistance. An adversary that holds multiple keys should only be able to access data if at least one individual key grants access. There two major advantages of attribute based encryption are

1. It has the capacity to address complex access control policies.
2. The exact list of users need not be known in prior. Knowledge of the access policy is sufficient.

Another important property that attribute based encryption schemes must satisfy is that of collision resistance. Collision Resistance means that if 2 or more users possessing different keys combine to decrypt the ciphertext, they will be successful if and only if any one of the users could have decrypted it individually. Even if multiple parties collide, they should be able to decrypt the ciphertext unless one of then was able to decrypt it com[lately by himself. These properties ensure that only users possessing the right keys have access to the information. Moreover, as the encryption is based on the access-structure it implicitly assures anonymous access control.

There exists two complimentary forms of attribute based encryption- ciphertext-policy attribute-based encryption (cp-abe)[9] and key policy attribute based encryption(kp-abe). In key policy attribute based encryption, each private key is associated with an access structure that specifies which type of ciphertexts the key can decrypt. The access structure is specified in the private key, while the ciphertexts are simply labeled with a set of descriptive attributes. In ciphertext policy attribute based encryption a user's private key will be associated with an arbitrary number of attributes and message is encrypted using a access structure over attributes. A user will only be able to decrypt the message if that user's attributes pass through the ciphertext's access structure. For our work Cipher-text Policy Attribute Based Encryption is used.

B. CIPHERTEXT ATTRIBUTE BASED ENCRYPTION

CP-ABE ensures confidentiality even if the storage server is untrusted. Whenever a user encrypts sensitive data , the user establish a specific access control policy on who can decrypt the message. For example, we can encrypt a ciphertext such that in a company it can only be decrypted by someone with attributes "Senior" and "Human Resources" or has the attribute "Executive". For instance, the access structure for accessing this information can be specified as ("Senior" AND "Human Resources") OR ("EXECUTIVE"). By this it means that the encrypted data can be decrypted by users whose designation is Senior and works in Human Resources Department or user with designation Executive.

Traditionally, this type of expressive access control is enforced by employing a trusted server to store data locally. The server is entrusted as a reference monitor that checks a user presents proper certification before allowing him to access data. It is difficult to guarantee the security of data using traditional methods, when data is stored at several locations.

Most existing public key encryption schemes allow a data owner to encrypt data to a particular user, but are unable to efficiently handle more expressive types of encrypted access control. CP_ABE addresses this problem.

An ciphertext-policy attribute based encryption scheme consists of four fundamental algorithms: Setup, Encrypt, KeyGen and Decrypt.

Setup: The setup algorithm takes no input other than the implicit security parameter. It outputs the public parameters PK and a master key MK.

Encrypt (PK, M, A): The encryption algorithm takes as input the public parameter PK, a message M and an access structure A. The algorithm will encrypt M and produce a ciphertext CT such that only the users who possesses a set of attributes that satisfies the access structure will able to decrypt the message.

KeyGen (MK, S): The key generation algorithm takes as input the master key MK and a set of attributes S that describes the key. It outputs a private secret key SK.

Decrypt(PK,CT,SK): The decryption algorithm takes as input the public parameter PK, encrypted message(ciphertext) CT, which contains the access policy, and secret key SK(private

 ISSN: 2321-8169
1423 – 1428

key for a set S of attributes). If the set S satisfies the access structure A then the algorithm will decrypt the ciphertext and return a message M.

How CPABE works

Consider

- List of users U-u_1,u_2,....u_n and
- List of attributes A-a_1,a_2,....a_n.

Each of the user is assigned a subset of attributes D-d_1,d_2,....d_n where D ∈ A. Each encrypted file will be assigned an access tree T in which

- Leaf nodes are attributes in A
- Each non leaf node is a gate node with assigned threshold.
- The threshold k_x ,$0<k_x \leq num_x$ is the number of children for node x
- If the node is an AND $k_x = num_x$.
- If the node is an OR k_x =1.

Example

- Attribute Doctor, Nurse, A, B,C.
- Users
  - User1: Doctor,A
  - User2: Doctor,C
  - User3: Nurse,B
- Access Tree

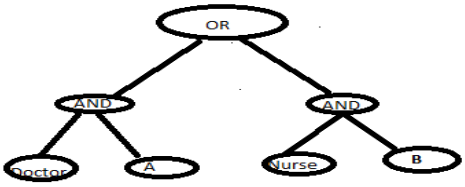

Figure 1: Access Tree Structure

If a particular file is encrypted using an access policy, only those users who has valid keys can decrypt the file. The users will be granted a set of keys via which they can decrypt the file. So if the file is encrypted using the access policy Doctor, A OR Nurse, B only the users with keys corresponding to Doctor, A OR Nurse, B will be able to decrypt the encrypted file

For our work, cpabe tool kit was used. The tool kit is implemented as described in [9]. Cpabe tool kit provides a set of programs implementing cipher-text policy attribute based encryption schemes. It use the PBC (Pairing Based Cryptography) Library for algebraic operations.

In cpabe tool kit private keys will be identified with a set S of descriptive attributes. Data owner who wishes to encrypt the message will specify through an access tree structure a policy that private keys must satisfy in order to decrypt.

Each interior node of the tree is a threshold gate and the leaves are associated with attributes. A user will be able to decrypt a ciphertext with a given key if and only if there is an assignment of attributes from the private key to nodes of the tree such that the tree is satisfied.

IV. CLOUD CONSTRUCTION

We have used Ubuntu OpenStack(FOLSOM)[10] to set up the cloud environment with one Controller node and three compute nodes. With its flexible architecture it is easy create a cloud environment. Ubuntu is the most popular operating system on the leading clouds and the fastest growing scale-out platform in the world. Ubuntu offers all software infrastructure, tools and services you need whether you are building your own cloud or you want to use a public cloud. Ubuntu offers

- Everything you need to efficiently build and integrate an OpenStack cloud.
- Certified Ubuntu Server images for use on the leading public clouds.
- Sophisticated tools to help you provision , build, manage and support your cloud at scale.

OpenStack is a global collaboration of developers and cloud computing technologists producing the ubiquitous open source cloud computing platform for public and private clouds. The project aims to deliver solutions for all types of cloud by being simple to implement, massively scalable and feature rich. The technology consists of a series of interrelated projects delivering various components for a cloud infrastructure solution.

V. OUR CONSTRUCTION

Consider the cloud environment where in which data owner's data is stored on cloud servers CS1, CS2, CS3 and CS4. Assume that the data owner issues a re-encryption command to CS2 which should be propagated to all other three cloud servers also. Due to some network failure CS3 did not receive the re-encryption command. At this particular time, if the revoked user queries cloud server CS3, he can access the data using old decryption key. Even though many solutions has been proposed, a better solution is that

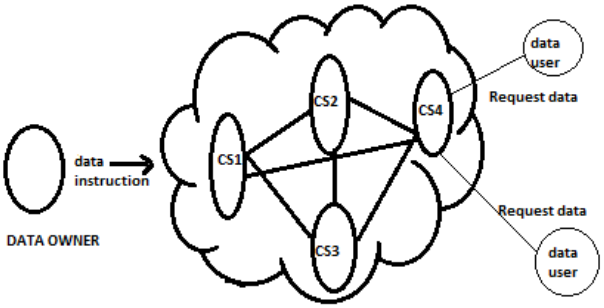

Figure 2: A typical Cloud Environment

each cloud server to independently re-encrypt the data without receiving any command from the data owner. In this thesis we propose a time based encryption scheme, which enables the cloud servers to automatically re-encrypt the data based on

their internal clock. This is built upon Attribute based Encryption to allow fine grain access control and does not require perfect clock synchronization for correctness. The data is associated with access control and access time. Each of the user is issued keys associated with attributes. The data can be decrypted only by the users using the keys with attributes satisfying access control and effective time satisfying the access time.

Unlike the other schemes the data owner shares a secret key with the cloud server so that the cloud server can re-encrypt the data by updating the data access time according to its internal clock. Even though the scheme depends on the time, it does not require a perfect clock synchronization among cloud servers. The main contribution in this paper are:

- An automatic time based re-encryption suitable for the cloud environments with unpredictable server crashes and network outages.
- Attribute based Encryption schemes is extended to incorporate time.
- Does not require clock synchronization among cloud servers.

Consider a cloud environment consisting of a data owner, a cloud service provider and multiple data owners. The data owner outsources his data to cloud service providers in form of a set of files F_1 .... F_n. Before uploading the data to the cloud service provider the data owner encrypts the data. Data users who want to access a particular file must first obtain necessary keys from the data owner to decrypt the file. The data owner can also update the contents of the file after uploading it to service provider which is termed as write command.

Each file, F , is encrypted using two parameters attributes and time. The time is divided into time slices of equal length. Attributes are organized into access structure which controls the access to the file. For example if a file with attributes α_1,α_2,α_3 and access structure A=($\alpha_1 \vee \alpha_2$) $\wedge \alpha_3$ data user requires either α_1 and α_2 or α_3 to satisfy the access structure. A file F can only be decrypted with keys that satisfy both the access structure and time slice.

The data users are granted a set of keys, associated with attributes and time slices. For example if Alice is authorized with a set of attributes a_1, a_2,.....a_n and is authorized to access the file from time slice T S_1 to T S_n then the data owner will grant a set of keys. The key set will contain key for each of the attribute and key for each of the time slice.

The security requirements of our scheme are as follows:

- Access Control correctness: A data user with invalid keys cannot decrypt the file
- Data Consistency: All data users who request a particular file at a particular time slice must obtain same file content.
- Data Confidentiality: Cloud Service Provider is not a valid user. The file content is visible to data users with valid keys.
- Efficiency: Cloud server should not re-encrypt the file unnecessarily. The file is re-encrypted only when a data user requests for that file.

Our system considers about two security issues. The first one is the Cloud Service Provider who is considered to be honest but curious. This means that the CSP will always correctly execute a protocol, but will try to gain some additional information about the stored data. The second issue is malicious users. This type of users will try to gain the contents of the files that he/she is not authorized to access. Malicious users possess invalid keys, that is , incorrect attributes or time.

A. IMPLEMENTATION

The data owner will generate a shared secret key and sends it to the cloud service provider. Before uploading the file to the service provider the data owner will encrypt the file using ABE (with appropriate access structure and time slice). As the result of encryption two files will be created for the original file. One encrypted with the access structure and other with the current time. The CSP will replicate the encrypted files to various cloud servers. Each cloud server will have a copy of the shared secret key.

Assume that a file is encrypted with access structure A and T S_i . When a user queries a particular cloud server , it first uses its internal clock to determine the current time slice. If the current time slice is T S_{i+k} the cloud server will automatically re-encrypt the encrypted file with T S_{i+k} (the file which was encrypted by the time) without receiving any command from the data owner. During the re-encryption process the cloud server cannot gain the contents of the ciphertext and the new decryption keys.

The user who has valid key for that particular time slice T S_{i+k} will be able to decrypt the original file. If the user has valid key for that time slice the file encrypted with attributes will be send to the data user . Then the data user will decrypt the file using the valid keys for the attributes by which the file was encrypted. In short, only users with keys satisfying T S_{i+k} and access structure A can decrypt the file.

Protocol Description

The functions used are

1)Setup() → (PK,MK,s)-At time T S_0 , the data owner publishes the system public key PK, keeps the master key MK secret and sends the shared secret key s to the cloud.

2)GenKey(PK,MK,s,Attrs,T,PK_X) →SK_X^{Attrs},SK_X

When a data owner wants to grant keys to data user with set of attributes Attrs and time period T, the data owner generates secret key for data user X with using at- tributes Attrs and another set of secret key using the time period. The time period is divided into equal time slices. The keys are generated for data user X using the public key PK, the master key MK , the shared secret key ,the public key of X , the attribute set Attrs and time period T.

3)Encrypt(PK,A,s,TS_t ,F) → C^t_A , C^t_T-At time T S_t , the data owner encrypts the file using the access structure A

and produces the ciphertext C^{t}_{A} using the system public key PK, access structure A , shared secret key s ,time slice T S_t and the plain text file F.

4)Check(PK,C^{t}_{T} ,SK_X^{t}) → C^{t}_{A}-Before decrypting a particular file , check function checks whether the user has key for that particular time.

5)Decrypt(PK,C^{t}_{A} , SK_A^{Attrs}) → F-At T S_t , user X who possesses version t attribute secret key , recovers F using the system public key PK, cipher text C^{t}_{A} and the secret key of the set of attributes SK_A^{Attrs}.

6)Re-encrypt (C^{t}_{T} ,s,T S_{t+k}) → C^{t+k}_{T} - When the cloud server wants to return a data user with the file at T S_{t+k} it up- dates the ciphertext from C^{t}_{T} to C^{t+k}_{T} using the shared secret key.

The description can be divided into three section: data owner initialization, data user read data and data owner write data.

1) Data Owner Initialization: The data owner runs the Setup function to initiate the system. On running the Setup , the function generates public key PK, master secret key MK and the shared secret key s which is used for re-encryption. The shared secret key is send to the service provider. When the data owner wants to upload a file F to the cloud he/she first defines a access structure A and determines the current time slice T S_i . Finally , the encrypt function , encrypts the file F with access structure and current time slice T S_i . So for file F two encrypted files will be generated one encrypted with the access structure (F_1) and other with the current time slice T S_i (F_2). After encryption the data owner uploads these two files to the cloud service provider. The service provider will replicate these files and store them in multiple cloud servers. When the data owner wants to grant access to some data user it runs GenKey function and generates keys for each attribute and stores them in prvkey file and also generates keys for time slice and stores them in timekey file. That is, for each user he will be given two files , one with keys for attributes and other with keys for the time period.

2)Data user read data: When user U wants to access a file F at T S_i , U sends a R(F) command and F_2 (that consists of the keys for the time period) to the cloud server, where F is the file name. On receiving the command the cloud server will run the re- encrypt function to re-encrypt the file with T S_i and checks whether the user U have the key for the current time slice (T S_i) by running the Check function . If it has the key, the encrypted file F_1 is send to the user U . The user U on receiving F_1 runs Decrypt function using the keys satisfying access structure A.

3)Data owner write data: When the data owner wants to write file F at T S_i , it will send a write command to the cloud. On receiving the write command, the cloud server will commit it at the end of T S_i.

Security Analysis

Access Control Correctness- The correctness of access control is vulnerable when a TS changes. Consider the situation in which user A has keys with effective time upto T S_i and user B has effective time starting from T S_{i+1} . Assume that the data owner updates the file F to F' such that the user querying at T S_i should obtain F and user querying at T S_{i+1} should obtain F'. The property of access control correctness fails if user A is able to access F' and user B ia able to access F.

User A's best time to launch attack is just before T S_{i+1} , as user A only has the key to access F. But if the cloud server's clock is consistent with the data owner's clock the cloud server will commit only at t_{i+1} so that user A never reads F' and so the attack fails. User B's best time to launch attack is just after t_i . Querying earlier than t_i does not help user B since he does not have the keys to decrypt the data. Since the cloud server will commit the write command at t_i as long as its clock is consistent with the data owner's clock user B will never access F but only F'. Thereby provides access control correctness.

Data Consistency- The users who query within the same TS should get same data. Assume that both user A and user B have valid keys and we want to show that as long as user A and user B query within the same time slice , they receive same data. Assume that user A and user B pick up time slice T S_i to attack the scheme. The best time for user A to attack is to query just after t_i and best time for user B to attack is to query just before t_{i+1}.

Any write command that occurs after user A and before user B does not affect the correctness of the scheme since this command will be committed at t_{i+1}. Furthermore, we have to ensure that all writes committed at t_i must have already arrived before t_i . Since all partie's clocks are consistent and there are no delays, any write command committed at t_i can only be received by the cloud server before t_i. Thus the data returned to both user A and user B are consistent.

Data Confidentiality- We only store encrypted data in the cloud. Our scheme preserves the data confidentiality operations from ciphertext policy attribute based encryption (CP-ABE) scheme, and retain the same confidentiality properties, the cloud without knowledge of keys cannot learn any useful information about the stored data. Also the cloud server won't be able to learn the contents while it performs re-encryption.

CONCLUSION

The project has a wide scope as it ensures security in cloud environment. This may help the industry to minimize the effort to enforce security. Our project has proposed a method to manage access control based on cloud server's internal clock

without the proxy re-encryption scheme. This technique does not rely on the cloud to reliably propagate re-encryption commands to all servers to ensure access control correctness. This project allows only the data owner to perform data updates. This can be extended to allow data users to perform data updates in addition to the data owners. Two encrypted files are created for a single file when data owner encrypts a file which is one of the disadvantages with the scheme.

REFERENCES

[1] S. Kamara and K. Lauter, ``Cryptographic Cloud Storage", Financial Cryptography and Data Security, Canary Islands, Spain,January 2010.

[2] M. Armbrust, A. Fox, R. Griffith, A.Joseph, R. Katz, A. Konwinski, G.Lee,D. Patterson, A.Rabkin and I.Stoica, ``A view of cloud computing",Communications of the ACM, April 2010.

[3] M. Kallahalla, E. Riedel, R. Swaminathan, Q.Wang and K. Fu, ``Scalable Secure File Sharing on Untrusted Storage",in Proc of USENIX FAST,2003.

[4] G. Ateniese, K. Fu, M. Green and S. Hohenberger, ``Improved proxy re-encryption schemes with applications to secure distributed storage",ACM Transactions on Information and System Security,Vol 9, Issue 4, November 2006.

[5] S. Di Vimercati, S. Foresti,S. Jajodia, S. Paraboschi and P. Samarati, ``Over-encryption: management of access control evolution on outsourced data", in Proc of VLDB, Vienna, Austria, September 2007.

[6] S. Yu, C.Wang, K. Ren and W. Lou, ``Achieving secure, scalable and fine-grained data access control in cloud computing" , in Proc of IEEE INFOCOM, Santiago, March 2010.

[7] G. Wang, Q. Liu and J. Wu, ``Hierarchical attribute based encryption for fine-grained data access control in cloud storage services" , in Proc of ACM CCS, Chicago, October 2010.

[8] S. Yu, C.Wang, K. Ren and W. Lou, ``Attribute based data sharing with attribute revocation" ,in Proc of ACM ASIACCS, Beijing, China, April 2010.

[9] J. Bethencourt, A.Sahai and B. Waters ``Ciphertext-policy attribute based encryption" ,in Proc of IEEE Symposium, New York, October 2007.

[10] http://www.openstack.org

[11] N. Antonopoulos and L. Gillam ``Cloud Computing : Principles, Systems and Applications", Springer, 2010, XVIII.

[12] V. Goyal, O. Pandey, A. Sahai and B. Waters. Attribute based Encryption for Fine-Grained Access Control of Encrypted Data., In ACM Conference on Computer and Communication Security (ACM CCS), Alexandria, USA, October 2006.

[13] Qin Liu,Chiu C.Tan, Jie Wu, and Guojun Wang. "Reliable Re-encryption in Unreliable Clouds," In Proc. of IEEE Globecom, Texas, December 2011.