AN EFFICIENT AND CONSISTENT DATA RE-ENCRYPTION SCHEME IN UN-RELIABLE CLOUDS

Nelapati Yashoda Rani [1], A Rama Swamy Reddy [2]

[1]Pursuing M.tech (CSE), [2]Assistant Professor,

Nalanda Institute of Technology (NIT) Siddhartha Nagar, Kantepudi(v), Sattenapalli Guntur-522438.

ABSTRACT

This secure cloud key approach mechanism is for the cloud owners to provide security to their files. The data should be encrypting while storing in to the cloud to protect from unreliable things. The secured data will decrypt by the only authorized users, this authentication is also provided by the cloud users to whom give permissions to decrypt the files, here the cloud service providers are involved to provide the services to the users. Whenever the user revocation is happen, the data owner needs to re encrypt the data by using new key to protect from the revocation user, means he needs to send new generated key to the existing users to continue. Sometime this command may not execute properly, since cloud computing environment contains number of servers with unreliable networks. We are resolving this problem by proposing a new technique called time-based re encryption, the server will automatically re-encrypt the data based on internal clock. Our propose is built up with Attribute based encryption, to best access control for the data, and here the clock synchronization is not required for correctness.

Keywords*: *Cloud Computing, Re Encryption, Revocation, Attribute Based Encryption

I INTRODUCTION

The use of cloud computing became popular due to store the data to the cloud service provider very less amount of cost, and we are using a technique data owners encrypt the data to protect from untrusted CSPs. Comfortable encryption technique such as attribute based encryption also used to provide delicious access control.

ABE uses attribute structure to encrypt the data, which contains different attributes. In the place of using particular key for a specific file, users use an attributes as keys. The attributes which are used by the users must be satisfied by the access structure to decrypt the file. If we assume a file uses attribute structure like $\{(\alpha 1 \wedge \alpha 2) \vee \alpha 3\}$ it means a user's has attributes α1 or α2 or α3 to decrypt the particular file.

The major issue of storing the encrypted data in the cloud with revoking user access, after revoking also a user having the access permission to decrypt the file by using old key in initial days. To overcome this problem introduce a naval technique, let a data user can do re encrypt the file after performing revocation, means a new generated key will send to the remaining users. By this a revoked user will not decrypt the file using his old key. Even though it is advantage, it has some disadvantage, when user revocation performing frequently it leads to performance headache.

The proxy re-encryption mechanism is an alternative solution for the above issue. This technique gets the advantage from the releasing resources in the cloud to re-encrypt the data. This technique also called as command driven re encryption technique, where the cloud performs re-encryption by getting the commands from data owners.

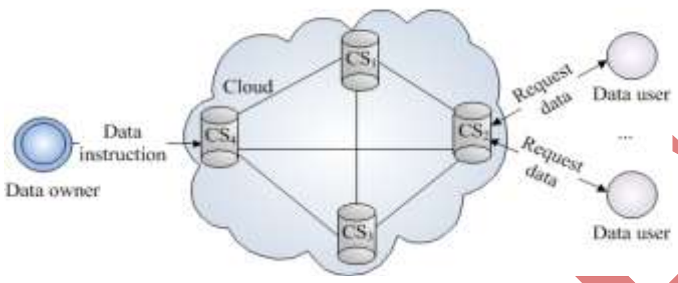

Fig. A: Typical Cloud Environment.

However the command driven re-encryption technique will not consider in the cloud architecture. Because the cloud servers are distributed, the data owners need to prevent the replication of data from multiple servers to quick access. As a distributed environment the cloud known the common failures due to server crashes and network problems. The re-encryption commands send by the data owner not spread to all the servers, it may leads to some risks.

By considering the cloud environment which is showed in Fig-1, the data owner may store their data in to CS1, CS2, CS3, CS4, the re-encryption command sent by the data owner will store in the CS4, to provide security to the data it needs to reach remaining CSPSs, due to network outage, if re-encryption command not reach to CS2. CSPs, did not re-encrypt the data, If the revoked user sends request to CS2, he will get the old cipher text, he can decrypt the data by using his old key.

The best solution is each server needs to independently re-encrypt the data without getting any command from the data owner. In this paper we are proposing a technique is reliable re-encryption in unreliable cloud (The R3 schema). R3 is a time based re-encryption mechanism, it allows every cloud automatically re-encrypt the data based on its internal clock. The common idea behind the R3 technique is link the data with access control and access time. Every user will provide a key which is linked with attribute and attribute effected time. Data can be decrypted by the user using by the key with attribute satisfied the access control. Contrasting from the re-encryption command technique, the data owner and CSP share the key, with this every cloud server will re encrypt the data by the updating the access time conferring its internal time.

Even with R3 pattern depends on time, it doesn't need exact clock synchronization from all cloud servers. Traditional clock synchronization mechanisms that confirms loose clock synchronization in the cloud enough.

II RELATED WORK

Lot of researchers have proposed so many techniques for storing the encrypted data to secure from CSP. With the previous approach the user revocation was done with third party to re-encrypt the data, with this the revoked user no longer decrypt the data with old key. This solution is an example, the users come with an issue, and the re-encryption key is to untrusted servers to re – encrypt the data. This solution uses PRE, by using this server convert stored cipher text another cipher, which will decrypt with different key. In this process the server don't have chance to learn about the content of cipher text and decryption keys.

ABE is a novel cryptographic technique for effective well access control. In the Hierarchical Attribute Based Encryption (HABE) technique is introduced to reach high performance and delegation. The difference between previous work and our new work is, here no need to depend on clouds infrastructure to be consistent in order to confirm correctness.

Our pattern trusts on re-encrypt the data on time. Though, the internal clock of every server is different from one server to another server. There are lot of solutions for this problem, for a while, introduces a probabilistic synchronization clock pattern. Which delegate a message to get the remote servers exact clock. K Romal used the delay of the message to know the maximum deference between communicating nodes to synchronized clock. L. Autonlpus introduced a clock synchronized pattern for cloud environment. In this authoritative time shared by the participants to access the file. By using this technique achieves the loose synchronization in cloud environment and to regulate the maximum time deference between data owner and every cloud server, our R3 pattern always reached correct access control in untrustworthy clouds.

III PRELIMINARY

3.1 Problem Formulation

We study the cloud computing environment contains data owners, cloud service providers and multiple data users.

TABLE 1

Alias keys

Key	Description
SK^1_{a1}	Keys for attributes a_1 for TS_1
…..	….
SK^1_{am}	Keys for attributes a_m for TS_1
….	….
SK^n_{a1}	Keys for attributes a_m for TS_n
….	….
SK^n_{am}	Keys for attributes a_m for TS_n

The data owners stores his data in the form set of files

F1, F2….. Fn, to the cloud servers. Before uploading the data in to CSP, data owner needs to encrypt that, the data user who ever want to download first he needs to get associated key from the data owner, then he will decrypt the data file. The data owner can have a permission to update the data after uploading into CSP.

Every file encrypted with two parameters one time slicing second one is attribute. Here we divide the time into time slice, each time slice length is equal. We represent a time slice with TS, with subscript of where $TS_i = [t_i, t_{i+1}]$. Attributes are arranged in access structure regulates, which control the access control with a file. A file uses attribute structure like $\{(\alpha 1 ^{\wedge} \alpha 2)_{v} \alpha 3\}$ it means a user's has attributes α1 or α2 or α3 to decrypt the particular file. If a key satisfied time slice and access structure then only a file will be decrypt.

A data user authenticated by the data user, then only he will get the key, which will associated attribute key and length of the time,

The security needs of R3 pattern is as follow

1) **Access control correctness.** This needs that a data user cannot decrypt the file with illegal keys.
2) **Data consistency.** This needs that all data users who wish file *F*, should gain the same information in the same time slice.
3) **Data confidentiality.** Here the CSP is not treated as valid data user, the data user to know content of the file he required valid key.
4) **Efficiency.** The cloud service provider will not re encrypt the file un necessarily, If a file not requested by any data user, that file will not re encrypt by the CSP.

3.2 Adversary model

Our system considering two types of adversaries, one is the CSP and another one is malicious data users. The CSPs are very truthful but they have some curiosity to know the additional information about the file. The malicious data user will try to access the data even though he is unauthorized access, this adversary pose the invalid key, the malicious data user can send request to any server in the cloud, the curiosity CSP and malicious data users are exist to threat the data.

TABLE II

SUMMARY OF NOTATIONS

Notation	Description
PK	System public key
UA	Universal attributes
PK^i_a	Attribute a's public key at TS_i
Sk^i_a	Attribute a's private key at TS_i
MK	Mater key
S	Shared secrete key
A	Alice's attributes
T	Effective time of Alice's attributes
A	Access control
PK_u	User public key
SK_u	User identity secret key
$SK^i_{u,a}$	User attribute secret key
	With attribute a and time TS_i

IV BASIC R3

In this Basic R3 we deliberate some conditions like the data user and cloud server will share synchronized clock, here no transmission delay while they are performing read and write operations.

4.1 Intuition

The owner of the data will first generate sharable secrete key with CSP, after he will encrypt every file using suitable attribute structure and time slice, that data uploaded to CSP, the CSP will copy to all the cloud servers, every cloud server will have a copy of sharable secrete key.

For example a cloud server stores an encrypted file F with TS_i. When a user request to cloud server, the cloud server will uses its own clock to know the time slice. If the current time slice is TS_{i+k}, he will re encrypt the file F with TS_{i+k} without get any command from data owner. While performing this process the CSP will not get the information about the cipher text and newly generated decrypt keys, only the users with key satisfying A with TS_{k+1}can decrypt file F.

4.2 Protocol Description

We segregate the description of R3 pattern in to 3 parts: one is owner initialization, data user read operation, data owner write operation. We depend on following functionalities. Table 2 shows the representation used in the description.

```
Algorithm 1 Basic R3 (Synchronized clock with no delays)
While get a write command W (F, seqnum) at particular $TS_i$, violate the write command at the end of $TS_i$
While get the read command R (F) at $TS_i$ perform re encrypt the file with $TS_i$.
```

1) *Data owner initialization*: To initiate the system the data owner setup a function, if the data owner wants to upload a file F to CSP, it first describe the access control structure A to F, and control the time slice Ts_i, finally it perform the encrypt function with A and TS_i to produce the cipher text, when the data owner wants to allocate set of attributes in a time period to the data user, it needs to run *GenKey* function to generate the key and accurate time slice for Alice.
2) Data user read data: When data user Alice wants to download the file F, at that time he needs to send access command to cloud server, whenever the cloud server gets the read command he will perform re encryption to re encrypt the file F, to decrypt the file F Alice perform decrypt function using key satisfied A and Ts_i .
3) Data owner written data: If the data owner wants to write the file F at time slice TSi, he needs to send write command to the cloud server in the form of W (F, seqnum). This seqnum is necessary for ordering when data owner sends multiple write commands at one time slice. With data owner, by the receiving write command the cloud server commit it at the end of time slice TS_i.

4.3 Security Analysis

Access control correctness. The acceptance of access control is most susceptible when change the TS. For example Alice has the keys up to TS_i, The Bob has the keys starting from the TS_{i+1}. The data owner changes his file F to F’, assume the data owner requesting for file f at TSi he will get F, if he requesting for a file at TS_{i+1}he will get F’, otherwise if a Alice having the capable to get the file F’ at TS_i (Attack 1) and Bob has the capable to get the file F at TS_{i+1}(Attack 2).

In the Attack 1, the best time to do attack is before t_{i+1}, and Alice has permission to decrypt the file up to t_{i+1}.After reaching the cloud server will violate the write command as well as his own clock, with the data owner does not have permissions after the change, if he trying to access he will not get anything, this is attack fail.
In the Attack 2 the best time to attack is after t_{i+1}. And Bob has permissions to decrypt after t_{i+1}.in the earlier to t_{i+1} The Bob does not have valid key to access the file.

Data Consistency. This property needs the user to send the query with in same time TS, must get same data. Example Alice and Bob have the valid keys to access the file specific time slice.
We first reminder that any write command after Alice and before Bob does not on the correctness of the R3 pattern due to this command is violated at t_{i+1}. Moreover, we have to guarantee that all the write commands violated at t_i must have already reached before t_i. With all servers clocks are insistent and there is no delay. Whatever the write command committed at ti will only be received by the cloud server at ti. Though the data reached to Alice and Bob is consistent.

Data Confidentiality. In our pattern we only store the encrypted data, the R3 pattern will provide high confidentiality using HABE, and Keeps the same confidentiality keys to the user, the CSP will not get any useful information without key.

Data efficiency. The CSP will not re encrypt file until he gets request from the data user. Using the properties function ReEncrypt. We know that the cloud server contains the re encryption operations until it gets the access request.

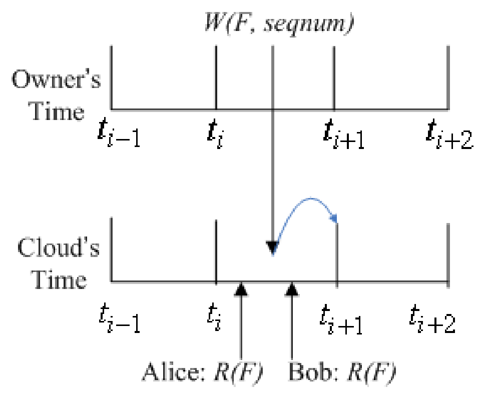

Fig.3. Attacks to cooperation the property of data consistency

Algorithm 2 Extended R3 (a synchronized clock with delay)

```
while Obtain a write command $W(F, ti+1, seqnum)$ do
    if Current time is earlier than $ti+1 + \alpha$ then
        Build Window $i$ for file $F$
        violate the write command in Window $i$ at $ti+1 + \alpha$
    else
        Reject the write command
```

```
      Apprise the data owner to send write command earlier
while Receive a read request $R$ ($F$, $TSi$) do
   if Current time is later than $ti$+1 + $\alpha$ then
      Re-encrypt the file in Window $i$ with $TSi$
   else
      Grip on the read command until $ti$+1 + $\alpha$
```

V EXTENDED R3

In this session, we are considering the situation where there are no synchronized clocks, the message transmission and the queue delay between the write and read operations.

5.1 Protocol Description

We tenancy the data owner and cloud server approve on maximal waiting time α, though, the cloud server needs to wait until $t_{i+\alpha}$ to require the write commands that must be committed at t_i, and give response to read command to read the data at given TS_i. The data user and data owner having extra information in their read or write command. When data owner want to change his file F at Tsi, he needs to give a command W (F, t_{i+1}, *seqnum*). Here F is file name, t_{i+1} denotes when the updates are happened, *seqnum* is write command order. If the data user wants to read the file F at TS_i, he needs to use a command R (F, TS_i).

Then we need to remind the maximum time difference between cloud server and data owner. We represent time difference as $\triangle$, here $\triangle$ is not larger than one time slice duration. In another way, when the owner of the data at TS_i, the time of the cloud server may be at TS_{i-1} or TS_i or TS_{i+1}. We push the data owner send his write command before $t_{i+1,}$ whenever he want to reflect this update at TS_{i+1}. Algorithm 2 expresses the cloud server actions.

5.2 Security Analysis

Access control correctness. In this section we need to demonstration how algorithm 2 maintain the access control property using same attack 1 and attack 2 like basic R3 analysis.

In attack 1, the best time for Alice to launch attack is just before t_{i+1}. Because he only knows the keys to decrypt the data up to TS_i. But the cloud server will commit his write command at t_{i+1}+α. Thus Alice will never read F', so his attack fails.

In attack 2, the best time for Bob launch his attack is just after t_{i+1}. Requesting before t_{i+1} not help to the Bob, because before he don't have valid key to decrypt the data. Though, the write command committed by the cloud server at t_{i+1}+α.Footing the all read commands until violet all write commands, so doesn't have chance to read F instead of F'.

Data consistency. The cloud server will reject all the read commands until violets the write commands.

The Data confidentiality and Data efficiency same as basic R3 pattern.

VI CONCLUSION

In this paper we are proposing a technique called R3 scheme, to effectively managing the access control based on the cloud server internal clock, our technique is not depends on cloud servers re encryption to all servers do correctness. This technique uses the synchronized clock time to re encrypt the data at server side, here every cloud server uses access control structure and time slice to re encrypt the data.

REFERENCES

[1] S. Kamara and K. Lauter, "Cryptographic cloud storage," *Financial Cryptography and Data Security*, 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*, 2010.

[3] A. Sahai and B. Waters, "Fuzzy identity-based encryption," *Advances in Cryptology–EUROCRYPT*, 2005.

[4] V. Goyal, O. Pandey, A. Sahai, and B. Waters, "Attribute-based encryption for fine-grained access control of encrypted data," in *Proc. of ACM CCS*, 2006.

[5] J. Bethencourt, A. Sahai, and B. Waters, "Ciphertext-policy attributebased encryption," in *Proc. of IEEE Symposium on S&P*, 2007.

[6] M. Blaze, G. Bleumer, and M. Strauss, "Divertible protocols and atomic proxy cryptography," *Advances in Cryptology–EUROCRYPT*, 1998.

[7] A. Boldyreva, V. Goyal, and V. Kumar, "Identity-based encryption with efficient revocation," in *Proc. of ACM CCS*, 2008.

[8] G. Wang, Q. Liu, and J. Wu, "Hierarchical attribute-based encryption for fine-grained access control in cloud storage services," in *Proc. Of ACM CCS (Poster)*, 2010.

AUTHOR PROFILE

Nelapati Yashoda Rani is currently pursuing M.Tech in the Department of Computer Science & Engineering, from Nalanda Institute of Engineering & Technology (NIET), siddharth Nagar, Kantepudi(V), Sattenapalli (M), Guntur (D), Andhra Pradesh , Affiliated to JNTU-KAKINADA.

A. Rama Swamy Reddy working as Assistant Professor at Nalanda Institute of Engineering & Technology (NIET), siddharth Nagar, Kantepudi(V), Sattenapalli (M), Guntur (D), Andhra Pradesh , Affiliated to JNTU-KAKINADA.