KEY UPDATES AND AUDIT SERVICES FOR OUTSOURCED DATA STORAGES

Jefia Wilson R
PG student, Department of CSE
Ponjesly College of Engineering, Nagercoil.

Mrs Manju C Thayammal M.Tech,(Ph.D)
Associate professor of CSE
Ponjesly college of engineering,Nagercoil.

Abstract- Key-exposure resistance is an important problem in cyber defense. In the cloud storage auditing the key exposure problem was well analyzed and studied. In the existing solution the client will be updating the secret keys in every period of time so it is very difficult for the client particularly those who are in limited computation resources. In the paper, we focus on how key updates are transparent to the client and propose a new concept called cloud storage auditing with verifiable outsourcing of key updates. Key updates can be safely outsourced to some authorized party and thus the key update work burden to the client will be reduced. In this we leverage the third party auditor in many existing public auditing designs, The role of authorized party and make it in charge of both the storage auditing and the secure key updates for key-exposure resistance. In this design, TPA only needs to hold an encrypted version of the client's secret key while doing all these burdensome tasks on behalf of the client. The client only needs to download the encrypted secret key from the TPA when uploading new files to cloud. Besides, our design also equips the client with capability to further verify the validity of the encrypted secret keys provided by the TPA. All these salient features are carefully designed to make the whole auditing procedure with key exposure resistance as transparent as possible for the client. The security proof and the performance simulation show that this detailed design instantiations are secure and efficient.

Keywords- Decryption, Encryption, key update, Third party Auditor, Secret key

I. INTRODUCTION

Cloud computing technology is becoming more and more popular now a days. It can provide users with unlimited computing resource. people can outsource time consuming computation workloads to cloud without spending the extra capital on maintaining hardware and software. In recent years, outsourcing computation has attracted much attention and been researched widely.[6] It has been considered in many applications including scientific computations linear algebraic computations linear programming computations and modular exponentiation computations etc. Besides, cloud computing can also provide users with seemingly unlimited storage resource.[7] Cloud storage is universally viewed as one of the most important services of cloud computing. Although cloud storage provides great benefit to users, it brings new security challenging problems. One important security problem is how to efficiently check the integrity of the data stored in cloud.[2] In recent years, many auditing protocols for cloud storage have been proposed to deal with this problem. These protocols focus on different aspects of cloud storage auditing such as the high efficiency the privacy protection of data the privacy protection of identities dynamic data operations the data sharing etc. The key exposure problem, as another important problem in cloud storage auditing, has been considered recently. The problem itself is non-trivial by nature. Once the client's secret key for storage auditing.

The authorized party should only hold an encrypted version of the user's secret key for cloud storage auditing because the authorized party performing outsourcing computation only knows the encrypted secret keys, key updates should be completed under the encrypted state. In other words, this authorized party should be able to update secret keys for cloud storage auditing from the encrypted version he holds. it should be very efficient for the client to recover the real secret key from the encrypted version that is retrieved from the authorized party. Lastly, the client should be able to verify the validity of the encrypted secret key after the client retrieves it from the authorized party. The goal of this paper is to design a cloud storage auditing protocol that can satisfy above requirements to achieve the outsourcing of key updates.

II. RELATED WORK

Outsourcing Computation: Time consuming computations has become a hot topic in the research of the theoretical computer science in the recent two decades. Outsourcing computation has been considered in many application domains [4].A new paradigm called cloud storage auditing system is proposed. In this new technique the key client operation is not performed by the client.

The key update operation is performed by the authorized party. The authorized party holds the encrypted secret key of the client for client for cloud storage auditing.[8] The client downloads the encrypted secret key from the authorized party and decrypt it only when the client need to upload any new files to cloud. The client needs to check the validity of the encrypted secret key. The secret keys for cloud storage auditing are updated periodically.[4] As a result, any dishonest behaviors, such as deleting or modifying the client's data previously stored in cloud, can all be detected, even if the cloud gets the client's current secret key for cloud storage auditing. However, the client needs to update his secret key in each time period.

Existing solutions all require the client to update the secret keys in every time period which may inevitably bring in new local burdens to the client especially those with limited computation resources. The client is the owner of the files that are uploaded to cloud. The total size of these files is not fixed that is the client can upload the growing files to cloud in different time points. The cloud stores the client's files and provides download service for the client. Important security problem is how to efficiently check the integrity of the data stored in storage area. With limited computation resources the users might not like doing such extra computations by themselves in each time period.

III PROPOSED SYSTEM

TPA only needs to hold an encrypted version of the client's secret key while doing all these burdensome tasks on behalf of the client. The client only needs to download the encrypted secret key from the TPA when uploading new files to cloud. our design also equips the client with capability to further verify the validity of the encrypted secret keys provided by the TPA. All these salient features are carefully designed to make the whole auditing procedure with key exposure resistance as transparent as possible for the client. The definition and the security model of this paradigm. The security proof and the performance simulation show that our detailed design instantiations are secure and efficient.

3.1 Data Producer and Retriever

The client is the owner of the files that are uploaded to cloud. The total size of these files is not fixed that is the client can upload the growing files to cloud in different time points. The cloud stores the client's files and provides download service for the client. A cloud storage auditing protocol with secure outsourcing of key updates is composed by seven algorithms (*Sys Setup*, *Ekey Update*, *Ver ESK*, *Dec ESK*, *Auth Gen*, *Proof Gen*, *Proof Verify*), shown below:
1) SysSetup: the system setup algorithm is run by the client. It takes as input a security parameter k and the total number of time periods T , and generates an encrypted initial client's secret key *ESK*0, a decryption key *DK* and a public key *PK*. Finally, the client holds *DK*, and sends *ESK*0 to the TPA.

2) EkeyUpdate: the encrypted key update algorithm is run by the TPA. It takes as input an encrypted client's secret key *ESKj* : the current period *j* and the public key *PK*, and generates a new encrypted secret key *ESKj*+1 for period *j* + 1.
3) *VerESK*: the encrypted key verifying algorithm is run by the client. It takes as input an encrypted client's secret key *ESKj* : the current period *j* and the public key *PK*,
if *ESKj* is a well-formed encrypted client's secret key, returns 1; otherwise, returns 0.
4) *DecESK*: the secret key decryption algorithm is run by the client. It takes as input an encrypted client's secret key *ESKj* , a decryption key *DK*, the current period *j*
and the public key *PK*, returns the real client's secret key *SK j* in this time period.
5) *AuthGen*: the authenticator generation algorithm is run by the client. It takes as input a file *F*, a client's secret key *SK j* , the current period *j* and the public key *PK*, and generates the set of authenticators _ for *F* in time period *j* .
6) *Proof Gen*: the proof generation algorithm is run by the cloud. It takes as input a file *F*, a set of authenticators _, a challenge *Chal*, a time period *j* and the public key *PK*, and generates a proof *P* which proves the cloud stores *F* correctly.
7) *Proof Veri f y*: the proof verifying algorithm is run by the TPA. It takes as input a proof *P*, a challenge *Chal*, a time period *j* , and the public key *PK*, and returns "*True*" if *P* is valid; or "*False*", otherwise.

3.2 Third Party Auditor

The TPA plays two important roles:

- The first is to audit the data files stored in cloud for the client.
- The second is to update the encrypted secret keys of the client in each time period.

The TPA can be considered as a party with powerful computational capability or a service in another independent cloud. TPA does not know the real secret key of the client. Traditional encryption technique is not suitable because it makes the key update difficult to be completed under the encrypted condition. Besides, it will be even more difficult to enable the client with the verification capability to ensure the validity of the encrypted secret keys. To address these challenges, the proposel of explore the blinding technique with homomorphic property to efficiently "encrypt" the secret keys. It allows key updates to be smoothly performed under the blinded version, and further makes verifying the validity of the encrypted secret keys possible. The security analysis later on shows that such blinding technique with homomorphic property can sufficiently prevent adversaries from forging any authenticator of valid messages. Therefore, it helps to ensure design goal that the key updates are as transparent as possible for the client.

3.3 Key Updates

The TPA updates the encrypted client's secret key for cloud storage auditing according to the next time period. But the public key keeps unchanged in the whole time periods. The client sends the key requirement to the TPA only when he wants to upload new files to cloud. And then the TPA sends the encrypted secret key to the client. After that, the client decrypts it to get his real secret key, generates authenticators for files, and uploads these files along with authenticators to cloud. The key update operation is performed by the authorized party. The authorized party holds the encrypted secret key of the client for client for cloud storage auditing. The client downloads the encrypted secret key from the authorized party and decrypt it only when the client need to upload any new files to cloud. The client needs to check the validity of the encrypted secret key. The secret keys for cloud storage auditing are updated periodically. As a result, any dishonest behaviors, such as deleting or modifying the client's data previously stored in cloud, can all be detected, even if the cloud gets the client's current secret key for cloud storage auditing. However, the client needs to update his secret key in each time period.

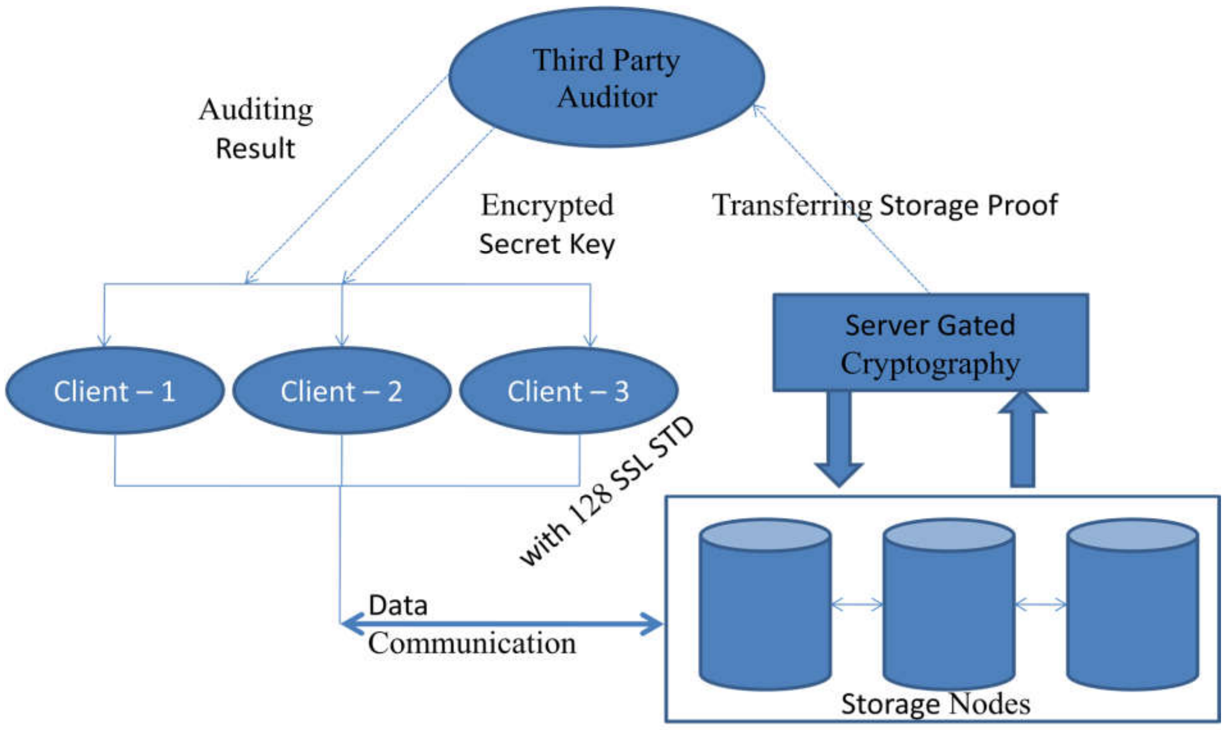

Figure 1: Architecture Diagram

IV CONCLUSION

key updates for cloud storage auditing with key-exposure resilience. The first cloud storage auditing protocol with verifiable outsourcing of key updates. In this protocol, key updates are outsourced to the TPA and are transparent for the client. The TPA only sees the encrypted version of the client's secret key, while the client can further verify the validity of the encrypted secret keys when downloading them from the TPA. Thus the security performance is high.

REFERENCES

1. M. J. Atallah, K. N. Pantazopoulos, J. R. Rice, and E.E.Spafford, "Secure outsourcing of scientific computations," Adv. Comput., vol. 54, pp. 215–272, 2008.
2. G. Ateniese, R. Di Pietro, L. V. Mancini, and G. Tsudik, "Scalable and efficient provable data possession," in Proc. 4th Int. Conf. Secur.Privacy Commun.Netw., 2008, Art. ID 9.
3. G. Ateniese et al., "Provable data possession at untrusted stores," in Proc. 14th ACM Conf of. Comput. Commun.Secur., 2007, pp. 598–600.
4. D. Benjamin and M. J. Atallah, "Private and cheating-free outsourcing algebraic computations," in Proc. 6th Annu. Conf. Privacy, Secur.200
5. X. Chen, J. Li, J. Ma, Q. Tang, and W. Lou, "New algorithms for secure outsourcing of modular exponentiations," in Proc. 17th Eur. Symp.Res.Comput.Secur.,2012,pp.541–556.
6. C. Wang, K. Ren, W. Lou, and J. Li, "Toward publicly auditable secure cloud data storage services," IEEE Netw., vol. 24, no. 4, pp. 19–24, Jul./Aug. 2010 Trust, 2008, pp. 240–245.

7. G. Yang, J. Yu, W. Shen, Q. Su, Z. Fu, and R. Hao, "Enabling public auditing for shared data in cloud storage supporting identity privacy and traceability," J. Syst. Softw., vol. 113, pp. 130–139,Mar. 2016.
8. J. Yu, F. Kong, X. Cheng, R. Hao, and G. Li, "One forward-secure signature scheme using bilinear maps and its applications," Inf. Sci., vol. 279, pp. 60–76, Sep. 2014.
9. J. Yu, R. Hao, H. Zhao, M. Shu, and J. Fan, "IRIBE: Intrusion resilient identity-based encryption," *Inf. Sci.*, vol. 329, pp. 90–104,Feb. 2016.