Deduplication on Encrypted Data in Cloud

T.SESHU CHAKRAVARTHY[1], S.RUPAMAHESWARI[2], A.L.V PRIYANKA[3], P.ASRITHA[4], V.DIVYAKEERTHI[5]
[1, 2,3,4,5] *Department of CSE, VasireddyVenkatadri Institute of Technology*

Abstract -- Cloud computing plays an important role in supporting data storage, processing, and management in the Internet of Things (IoT). To preserve cloud data confidentiality and user privacy, cloud data are often stored in an encrypted form. However, duplicated data that are encrypted under different encryption schemes could be stored in the cloud, which greatly decreases the utilization rate of storage resources, especially for big data. Several data deduplication schemes have recently been proposed. However, most of them suffer from security weakness and lack of flexibility to support secure data access control. Therefore, few can be deployed in practice. This article proposes a scheme based on attribute-based encryption (ABE) to deduplicate encrypted data stored in the cloud while also supporting secure data access control. The authors evaluate the scheme's performance based on analysis and implementation. Results show the efficiency, effectiveness, and scalability of the scheme for potential practical deployment.

I. INTRODUCTION

CSPs provide desirable service properties, such as scalability, elasticity, fault tolerance, and pay per use. Thus, cloud computing has become a promising service paradigm to support IoT applications and IoT system deployment. To ensure data privacy, existing research proposes to outsource only encrypted data to CSPs. However, the same or different users could save duplicated data under different encryption schemes at the cloud. Although cloud storage space is huge, this kind of duplication wastes networking resources, consumes excess power, and complicates data management. Thus, saving storage is becoming a crucial task for CSPs. Deduplication can achieve high space and cost savings, reducing up to 90 to 95 percent of storage needs for backup applications (http://opendedup.org) and up to 68 percent in standard file systems.1 Obviously, the savings, which can be passed back directly or indirectly to cloud users, are significant to the economics of cloud business. At the same time, data owners want CSPs to protect their personal data from unauthorized access. CSPs should therefore perform access control based on the data owner's expectations. In addition, data owners want to control not only data access but also its storage and usage. From a flexibility viewpoint, data deduplication should cooperate with data access control mechanisms. That is, the same data, although in an encrypted form, is only saved once at the cloud but can be accessed by different users based on the data owners' policies. However, current industrial deduplication solutions can't handle encrypted data. Existing solutions for deduplication are vulnerable to brute-force attacks2 and can't flexibly support data access control and revocation (see the ―Related Work in Data Deduplication‖ sidebar for a discussion of some other work in this area).3 Few existing schemes for cloud data access control support data deduplication simultaneously, 4 and few can ensure flexibility and security with sound performance for cloud data deduplication that data owners control directly.5–7 We propose a scheme based on attribute-based encryption (ABE) to deduplicate encrypted data stored in the cloud and support secure data access ontrol at the same time. Analysis and implementation demonstrate that our scheme is secure, effective, and efficient.

II. EXISTING SYSTEM

To ensure data privacy, existing research proposes to outsource only encrypted data to CSPs. However, the same or different users could save duplicated data under different encryption schemes at the cloud. Existing solutions for deduplication are vulnerable to brute-force attacks2 and can't flexibly support data access control and revocation (see the ―Related Work in Data Deduplication‖ sidebar for a discussion of some other work in this area). Existing industrial solutions fail in encrypted data deduplication.

DISADVANTAGES

Deduplication technology has become quite the staple in many data storage environments. But what makes it a good fit in one data center, may not be the case in another. This E-Guide from SearchStorage.com is designed to help you determine what you're trying to solve with deduplication technology. It then outlines: The advantages and disadvantages of dedupe backup Dedupe misconceptions how deduce and compression on primary storage can reduce your data footprint.

III. PROPOSED SYSTEM

In this application we propose to outsource only encrypted data to CSPs. However, the same or different users could save duplicated data under different encryption schemes at the cloud. Although cloud storage space is huge, this kind of duplication wastes networking resources, consumes excess power, and complicates data management. Intra-user deduplication and interdeduplication.6 In their scheme, the ciphertext C of convergent encryption is further encrypted with a user key and transferred to the servers. However, it doesn't deal with data sharing after deduplication among different users.

ADVANTAGES:

The scheme can easily realize data access control by introducing control policies into AP when calling EncryptKey (DEKu, AP, PKIDu) by updating AP to support both deduplication and access control based on practical demands. Our scheme can also supports digital rights management based on the data owner's expectations. Second, the scheme saves CSP storage space since it only stores one copy of the same data. Storage-based data deduplication reduces the amount of storage needed for a given set of files. It is most effective in applications where many copies of very similar or even identical data are stored on a single disk—a surprisingly common scenario. In the case of data backups, which routinely are performed to protect against data loss, most data in a given backup remain unchanged from the previous backup.

IV. MODULE DESCRIPTION

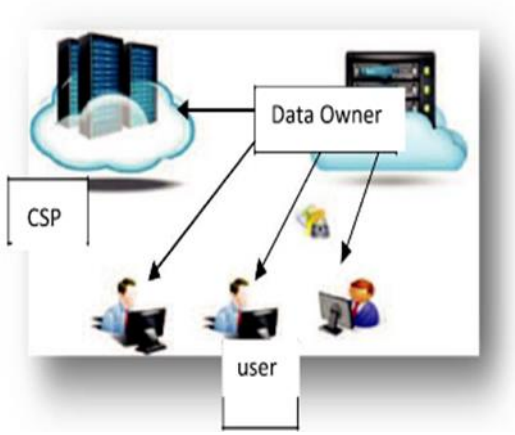


Figure 1: Three entities in Proposed System

Data Owner

In this module, the data owner uploads their data in the cloud server. For the security purpose the data owner encrypts the file and the index name and then store in the cloud. The data encryptor can have capable deleting of a specific file. And also he can view the transactions based on the files he uploaded to cloud.

Data User

In this module, user logs in by using his/her user name and password. After Login user requests search control to cloud and will Search for files based on the index keyword with the Score of the searched file and downloads the file. User can view the search ratio of the files and also the top k documents.

Cloud Server

The cloud server manages a cloud to provide data storage service. Data owners encrypt their data files and store them in the cloud for sharing with Remote User. To access the shared data files, data consumers download encrypted data files of their interest from the cloud and then decrypt them. The cloud server authorizes the data owner and the data user and provides the search requests sent from the users. Also in this module it shows personalized search model and

the interest search model. Can view all the file attackers.

V. ALGORITHMS

Our scheme consists of several fundamental algorithms. We can adopt either cypher text policy ABE (CP-ABE) or key policy (KP-ABE) to implement related algorithms.

Initiate Node

The Initiate Node algorithm takes as input the node identifier u. It outputs several user credentials including (PKu, SKu) and (put, ski). This process is conducted at user u.

CreateIDPK

The CreateIDPK algorithm is executed by u whenever u wants to control its data storage and access in the cloud. The algorithm checks the ID-related policies. If the result is positive, the algorithm outputs a public attribute key about the ID for user u, denoted PKIDu; otherwise, it outputs NULL.

IssueIDSK.

The IssueIDSK algorithm checks wheth- eru′ with public key PKu′is eligible to hold the data. If it is, IssueIDSK outputs a secret attribute key SKID(u,u′)for user u′. Otherwise, it outputs NULL. This process is executed by u based on identity verification by checking that Cert(PKu′) is a valid certificate and the owner of PKu′is an eligible data holder. Data owner u sends SKID(u,u′)to u′ through a secure channel or using the PKC.

Encryptke

The encrypt key algorithm takes as input symmentric key DEKu, acess policy AP and public key PKIDu corresponding to the identity attribute occurring in AP.The algorithm encrypts DEKu with the policy AP and cipher key CKu. This process is conducted at you to support deeduplication of data storage at CSPs.

Decryptkey

The decrypt key algorithm takes as input cipher key CKu produced by encrypt eyalgorithm,the access policy AP uder which cipher key CKuwasv encrypted SKu' and SKID(u,u') it decrypts CKu and outputs corresponding plain keyDEKu if the attributes atre sufficient to satisfy AP otherwise it outputs NULL.it is executed at u'.ifdeplicated storage occur it first checks the access policy AP, then conducts decryption get DEKu.

Encrypt

The algorithm encrypts data M with DEKu and outputs ciphertextCTu. This process is conducted at user u to protect its data with DEKu.

Decrypt

This algorithm is conducted at user u′ or u to decrypt CTu with DEKu and output plain data M.

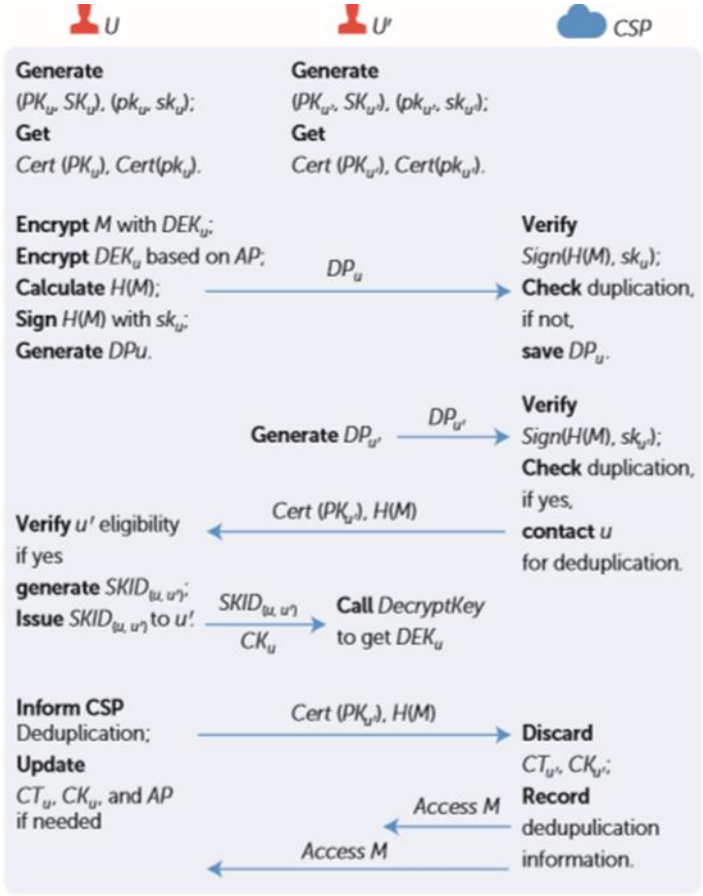


Figure 2 :Data Deduplication process

It illustrates the data deduplication process at a CSP with the data owner's instruction and control. We assume that user u is a data owner who saves data M at the CSP, using DEKuto protect the data, while user u′ is a data holder who tries to save the same data at that CSP.

First, each CSP user generates personal credentials and two key pairs (PKu, SKu) and (pku, sku) and gets the certificates of its public keys Cert(PKu) and Cert(pku). User u saves data M at the CSP. The user generates CTu = Encrypt(DEKu, M) and encrypts DEKu with PKIDu by calling EncryptKey(DEKu, AP, PKIDu) to get CKu. It generates PKIDu according to u's data storage and access policy. User u calculates H(M) and signs it with sku as Sign(H(M), sku). Next, u sends the data package, DPu = {CTu, CKu, H(M), Sign(H(M), sku), Cert(PKu), Cert(pku)}, to the CSP. After receiving DPu, the CSP verifies Cert(PKu) and Cert(pku). If the verification is positive, the CSP uses verify Sign(H(M), sku) to check if duplicate data is saved by finding whether the same H(M) is in its storage. If the check is negative, the CSP saves DPu.

If the check is positive and the prestored data is from the same user, the CSP notifies that user. If a different user is storing the same data, the CSP performs deduplication. If data holder u′ uploads the same data to the CSP, the CSP contacts data owner u by sending H(M) and Cert(PKu′) for deduplication. User u verifies the eligibility of u′. If verification is positive,user u calls IssueIDSK(ID, SKu, PKu′) to generate SKID(u,u′) and issues SKID(u,u′) to u′ to allow it to access M. Next, user u reports the successful data deduplication to the CSP. After receiving this notification, the CSP discards CTu′ and CKu′ and recodes the corresponding deduplication information in the system. In this step, the data owner can also update DEKu, upload new CTu and CKu to the CSP, and send the newly encrypted DEKu (with ABE) to eligible data holders through the CSP or directly. At this moment, both u and u′ can access data.M stored at the CSP. User u uses DEKu directly, whereas u′ gets DEKu by calling DecryptKey (CKu, AP, SKu′ , SKID(u,u′)).

Data Owner Management

During data deletion, the CSP also asks the data owner to allow it to decide if the raw data needs to be reencrypted. For security purposes, the data owner could select a new DEKu, reencrypt data M, update CTu and CKu at the CSP, and issue a new CKu to existing eligible users for managing data deletion.

VI. PERFORMANCE ANALYSIS AND EVALUATION

Our scheme ensures data confidentiality through ABE, symmetric key encryption, and PKC. The original user data is encrypted using symmetric encryption with DEK, which is then encrypted using the Encrypt Key algorithm under access policy AP. Assuming that the symmetric key algorithm is secure (for example, using a standard algorithm such as AES), the scheme's data confidentiality merely relies on the security of the Encrypt Key algorithm.

The CSP and other parties have no way to know the original data without DEK. We use ABE to preserve DEK. A security proof has been given under an attribute based selective-set module.8 The CSP and other users with unmatched identities can't get the secret key SKID (u,u′) from the data owner, so they can get nothing about DEK and the original data.

The proposed scheme involves three system entities: data owner, data holder, and CSP. We adopt AES for symmetric encryption, RSA for PKC. And CP-ABE for data deduplication.

The Initiate Node algorithm includes the key generation operation of two key pairs, which are PKC (RSA) key pair generation and ABE key pair generation. The RSA key generation needs one modular inversion. The key generation of ABE includes an exponentiation for PKu and an exponentiation for SKu on a group. The CreateIDPK algorithm contains two exponentiations on a group and the IssueIDSKalgorithm contains only one exponentiation. The computation complexity of encrypting data using AES depends on the data size, which is inevitable in any cryptographic method for protecting data.

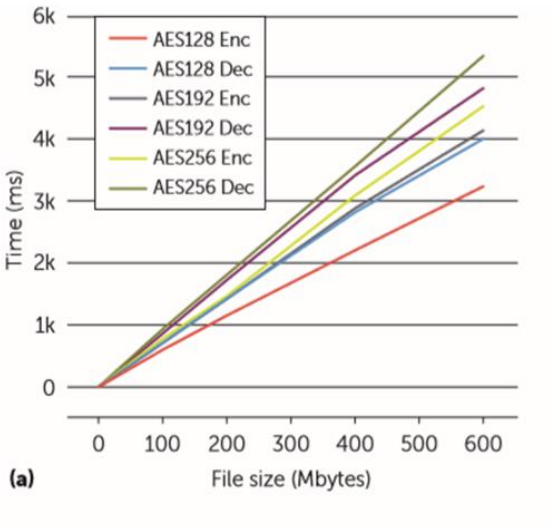


Figure 3.Operation time of (a) file encryption and decryption with AES and (b) the Encrypt Key algorithm.

However, for the Encrypt Key algorithm, the computation contains three exponentiations on a group no matter how many IDs are in access policy AP. The signing operation requires one exponentiation.

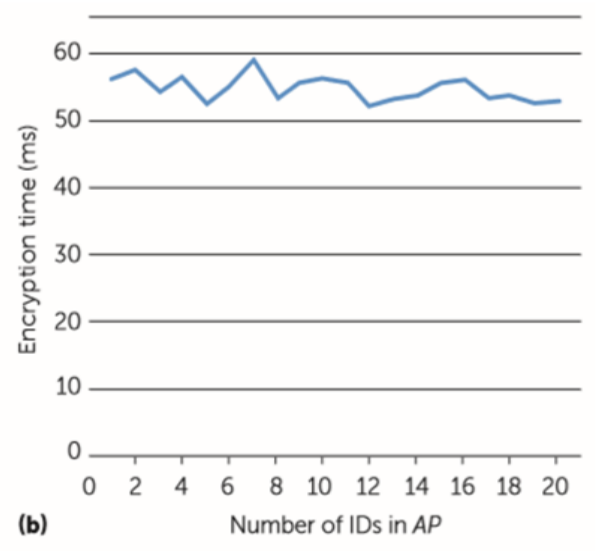
Each user uploads its data to the CSP. The CSP should first check the signature and decide if it's saving the appropriate copy of the data. It needs only one exponentiation for signature verification. Its computation complexity depends on the number of data holders.

When a data holder wants to store data to a CSP, it operates the same way as a data owner. But if it wants to access the data, it needs to conduct a decryption operation. Here, we only consider the Decrypt Key algorithm since the computation complexity of the Decrypt algorithm depends on the data size. The DecryptKey algorithm contains two bilinear pairings, which are cost constant.

VII IMPLEMENTATION AND EVALUATION

We implemented the proposed scheme using CPABE with the pairing-based cryptography (PBC) library and the OpenSSL library. We tested our scheme's performance in a Linux operating system (Linux 2.6.32-71. el6.i686 CentOS 6.0) running an Intel CPU (Intel Core i5-2450 Quad CPU 2.5 GHz, 2 Gbytes SDRAM). We used SHA-1 as the hash function in implementation.

We tested the operation time of data encryption and decryption with AES by applying different AES key sizes (128, 196, and 256 bits) and different data size (from 10 to 600 Mbytes). As Figure 2a shows, even when the data is as large as 600 Mbytes, the encryption/decryption time is less than 6,000 milliseconds (ms) using a 256-bit AES key. Therefore, applying symmetric encryption for data protection is a reasonable and practical choice. Based on our implementation, we found the operation time of InitiateNode, CreateIDPK, IssueIDSK, and DecryptKey to be 69.2, 4.3, 3.9, and 12.7 ms.All of them are constant. For EncryptKey, the operation time is about 54 ms, as Figure3b shows, approximately constant, which doesn't vary much with the number of IDs in AP. This is because AP only contains one type of attribute ID. This result proves that our scheme is efficient, with sound scalability and extensibility. The operation time of fundamental algorithms is stable and doesn't change with the number of data holders.



VIII. CONCLUSION

Managing encrypted data with deduplication is significant in practice for running a secure, dependable, and green cloud storage service, especially for big data processes. Future work includes efficient data ownership verification, scheme

optimization with hardware acceleration at IoT devices for practical deployment, and development of a flexible solution to support deduplication and data access controlled by either the data owner or its representative agent. REFERENCES

[1] D.T. Meyer and W.J. Bolosky, "A Study of Practical Deduplication," ACM Trans. Storage, vol. 7, no. 4, 2012, pp. 1–20.

[2] M. Bellare, S. Keelveedhi, and T. Ristenpart, "Message-Locked Encryption and Secure Deduplication," Advances in Cryptology (EUROCRYPT 13), LNCS 7881, 2013, pp. 296–312.

[3] J. Li et al., "A Hybrid Cloud Approach for Secure Authorized Deduplication," IEEE Trans. Parallel Distributed Systems, vol. 26, no. 5, 2015, pp. 1206–1216.

[4] Z. Wan, J. Liu, and R.H. Deng, "HASBE: A Hierarchical Attribute-Based Solution for Flexible and Scalable Access Control in Cloud Computing," IEEE Trans. Information Forensics and Security, vol. 7, no. 2, 2012, pp. 743–754.

[5] M. Fu et al., "Accelerating Restore and Garbage Collection in Deduplication-Based Backup Systems via Exploiting Historical Information," Proc. Usenix Ann. Technical Conf., 2014, pp. 181–192.

[6] M. Kaczmarczyk et al., "Reducing ImpactData Fragmentation Caused by In-Line Deduplication," Proc. 5th Ann. Int'l Systems and Storage Conf., 2012, pp. 1–12.

[7] M. Lillibridge, K. Eshghi, and D. Bhagwat, "Improving Restore Speed for Backup Systems That Use Inline Chunk-Based Deduplication," Proc. 11th Usenix Conf. File and Storage Technologies, 2013, pp. 183–198.

[8] Z. Yan and M.J. Wang, "Protect Pervasive Social Networking Based on Two Dimensional Trust Levels," IEEE Systems J., Sept. 2014, pp. 1–12; doi: 10.1109/JSYST.2014.2347259.