Crypt-Dac Cryptographically Enforced Dynamic Access Control In The Cloud

Yadlapalli Balatejaswi, Miss G.Keerthana, Sri.V.Bhaskara Murthy
MCA Student, Assistant Professor, Associate Professor
Dept Of MCA
B.V.Raju College, Bhimavaram

ABSTRACT

Enabling cryptographically enforced access controls for data hosted in un trusted cloud is attractive for many users and organizations. However, designing efficient cryptographically enforced dynamic access control system in the cloud is still challenging. In this paper, we propose Crypt-DAC, a system that provides practical cryptographic enforcement of dynamic access control. Crypt-DAC revokes access permissions by delegating the cloud to update encrypted data. In Crypt-DAC, a file is encrypted by a symmetric key list which records a file key and a sequence of revocation keys. In each revocation, a dedicated administrator uploads a new revocation key to the cloud and requests it to encrypt the file with a new layer of encryption and update the encrypted key list accordingly. Crypt-DAC proposes three key techniques to constrain the size of key list and encryption layers. As a result, Crypt-DAC enforces dynamic access control that provides efficiency, as it does not require expensive decryption/re encryption and uploading/re-uploading of large data at the administrator side, and security, as it immediately revokes access permissions. We use formalization framework and system implementation to demonstrate the security and efficiency of our construction.

I.INTRODUCTION

With the considerable advancements in cloud computing, users and organizations re finding it increasingly appealing to store and share data through cloud services. Cloud service providers (such as Amazon, Microsoft, Apple, etc.) provide abundant cloud based services, ranging from small-scale personal services to large-scale industrial services. However, recent data breaches, such as releases of private photos [10], have raised concerns regarding the privacy of cloud-managed data. Actually, a cloud service provider is usually not secure due to design drawbacks of software and system vulnerability [2], [3]. As such, a critical issue is how to enforce data access control on the potentially untrusted cloud.

In response to these security issues, numerous works [1], [4]–[9] have been proposed to support access control on untrusted cloud services by leveraging cryptographic primitives. Advanced cryptographic primitives are applied for enforcing many access control paradigms. For example, attributebased encryption (ABE) [5] is a cryptographic counterpart of attribute-based access control (ABAC) model [11]. However, previous works mainly consider static scenarios in which access control policies rarely change. The previous works incur high overhead when access control policies need to be changed in practice. At a first glance, the revocation of a user's permission can be done by revoking his access to the keys with which the files are encrypted. This solution, however, is not secure as the user can keep a local copy of the keys before the revocation. To prevent such a problem, files have to be re-encrypted with new keys. This requires the file owner to download the file, re-encrypt the file, and upload it back for the cloud to update the previous encrypted file, incurring prohibitive communication overhead at the file owner side. Currently, only a few works investigated the problem of dynamic data access control. Garrison et al. [12] proposed two revocation schemes. The first scheme requires an administrator to re-encrypt file with new keys as discussed above. This scheme incurs a considerable communication overhead. Instead, the second scheme delegates users to re-encrypt the file when they need to modify the file, relieving the administrator from re-encrypting

file data by itself. This scheme, however, comes with a security penalty as the revocation operation is delayed to the next user's modification to the file. As a result, a newly revoked user can still access the file before the next writing operation. Wang et al. [23] proposed another revocation scheme, in which the symmetric homomorphic encryption scheme [24] is used to encrypt the file. Such a design enables the cloud to directly re-encrypt file without decryption. However, this scheme incurs expensive file read/write overhead as the encryption/decryption operation involves comparable overhead with the public key encryption schemes.

To overcome these problems, we present Crypt-DAC, a cryptographically enforced dynamic access control system on untrusted cloud. Crypt-DAC delegates the cloud to update encrypted files in permission revocations. In Crypt-DAC, a file is encrypted by a symmetric key list which records a file key and a sequence of revocation keys. In a revocation, the administrator uploads a new revocation key to the cloud, which encrypts the file with a new layer of encryption and updates the encrypted key list accordingly. Same as previous works [12], [23], we assume a honest-but-curious cloud, i.e., the cloud is honest to perform the required commends (such as re-encryption of files and properly update previous encrypted files) but is curious to passively gathering sensitive information. Although the basic idea of layered encryption is simple, it entails tremendous technical challenges. For instance, the size of key list and encryption layers would increase as the number of revocation operations, which incurs additional decryption overhead for users to access files. To overcome such a problem, Crypt-DAC proposes three key techniques as follows.

First, Crypt-DAC proposes delegation-aware encryption strategy to delegate the cloud to update policy data. For a file, the administrator appends a new revocation key at the end of its key list and requests the cloud to update this key list in the policy data. The size of the key list however increases with the revocation operations, and a user has to download and decrypt a large key list in each file access. To overcome this problem, we adopt the key rotation technique [15] to compactly encrypt the key list in the policy data. As a result, the size of the key list remains constant regardless of revocation operations.

Second, Crypt-DAC proposes adjustable onion encryption strategy to delegate the cloud to update file data. For a file, the administrator requests the cloud to encrypt the file with a new layer of encryption. Similarly, the size of the encryption layers increases with the revocation operations, and a user has to decrypt multiple times in each file access. To overcome this problem, we enable the administrator to define a tolerable bound for the file. Once the size of encryption layers reaches the bound, it can be made to not increase anymore by delegating encryption operations to the cloud. As a result, the administrator can flexibly adjust a tolerable bound for each file (according to file type, access pattern, etc.) to achieve a balance between efficiency and security.

During the life cycle of a file, its encryption layers continuously increase until a pre-defined bound is reached. Crypt- DAC proposes delayed de-onion encryption strategy to periodically refresh the symmetric key list of the file and remove the bounded encryption layers over it through writing operations. In specific, the next user to write to the file encrypts the writing content by a new symmetric key list only containing a new file key, and updates the key list in the policy data. With this strategy, Crypt-DAC periodically removes the bounded encryption layers of files while amortizing the burden to a large number of writing users.

Altogether, Crypt-DAC achieves efficient revocation, efficient file access and immediate revocation simultaneously. For revocation efficiency, Crypt-DAC incurs lightweight communication overhead at the administrator side as it does not need to download and re-upload file data. For immediate revocation, the permissions of users are immediately revoked as the files are re-encrypted. For file access efficiency, the files are still encrypted by symmetric keys. We have implemented Crypt-DAC as well as several recent works [12], [23] on Alicloud. Real experiments suggest that Crypt-DAC is three orders of magnitude

more efficient in communication in access revocation compared with the first scheme in [12], and is nearly two orders of magnitude more efficient in computation in file access compared with the scheme in [23]. Finally, Crypt-DAC is able to immediately revoke access permissions compared with the second scheme in [12].

II.EXISTING SYSTEM

- Gudes et al. [27] explore cryptography to enforce hierarchy access control without considering dynamic policy scenarios. Akl et al. [28] propose a key assignment scheme to simplify key management in hierarchical access control policy. Also, this work does not consider policy update issues. Later, Atallah et al. [29] propose a method that allows policy updates, but in the case of revocation, all descendants of the affected node in the access hierarchy must be updated, which involves high computation and communication overhead.
- Ibraimi et al. [30] cryptographically support role based access control structure using mediated public encryption. However, their revocation operation relies on additional trusted infrastructure and an active entity to re-encrypt all affected files under the new policy. Similarly, Nali et al. [31] enforce role based access control structure using public-key cryptography, but requires a series of active security mediators. Ferrara et al. [32] define a secure model to formally prove the security of a cryptographically enforced RBAC system. They further show that an ABE-based construction is secure under such model. However, their work focuses on theoretical analysis.
- Pirretti et al. [33] propose an optimized ABE-based access control for distributed file systems and social networks, but their construction does not explicitly address the dynamic revocation. Sieve [23] is a attribute based access control system that allows users to selectively expose their private data to third web services. Sieve uses ABE to enforce attribute based access policies and homomorphic symmetric encryption [24] to encrypt data. With homomorphic symmetric encryption, a data owner can delegate revocation tasks to the cloud assured that the privacy of the data is preserved. This work however incurs prohibitive computation overhead since it adopts the homomorphic symmetric encryption to encrypt files.

- GORAM [25] allows a data owner to enforce an access matrix for a list of authorized users and provides strong data privacy in two folds. First, user access patterns are hidden from the cloud by using ORAM techniques [26]. Second, policy attributes are hidden from the cloud by using attribute-hiding predicate encryption [21], [22]. The cryptographic algorithms, however, incur additional performance overhead in data communication, encryption and decryption. Also, GORAM does not support dynamic policy update. Over encryption [34], [35] is a cryptographical method to enforce an access matrix on outsourced data. Over-encryption uses double encryption to enforce the whole access matrix. As a result, the administrator has to rely on the cloud to run complex algorithms over the matrix to update access policy, assuming a high level of trust on the cloud.

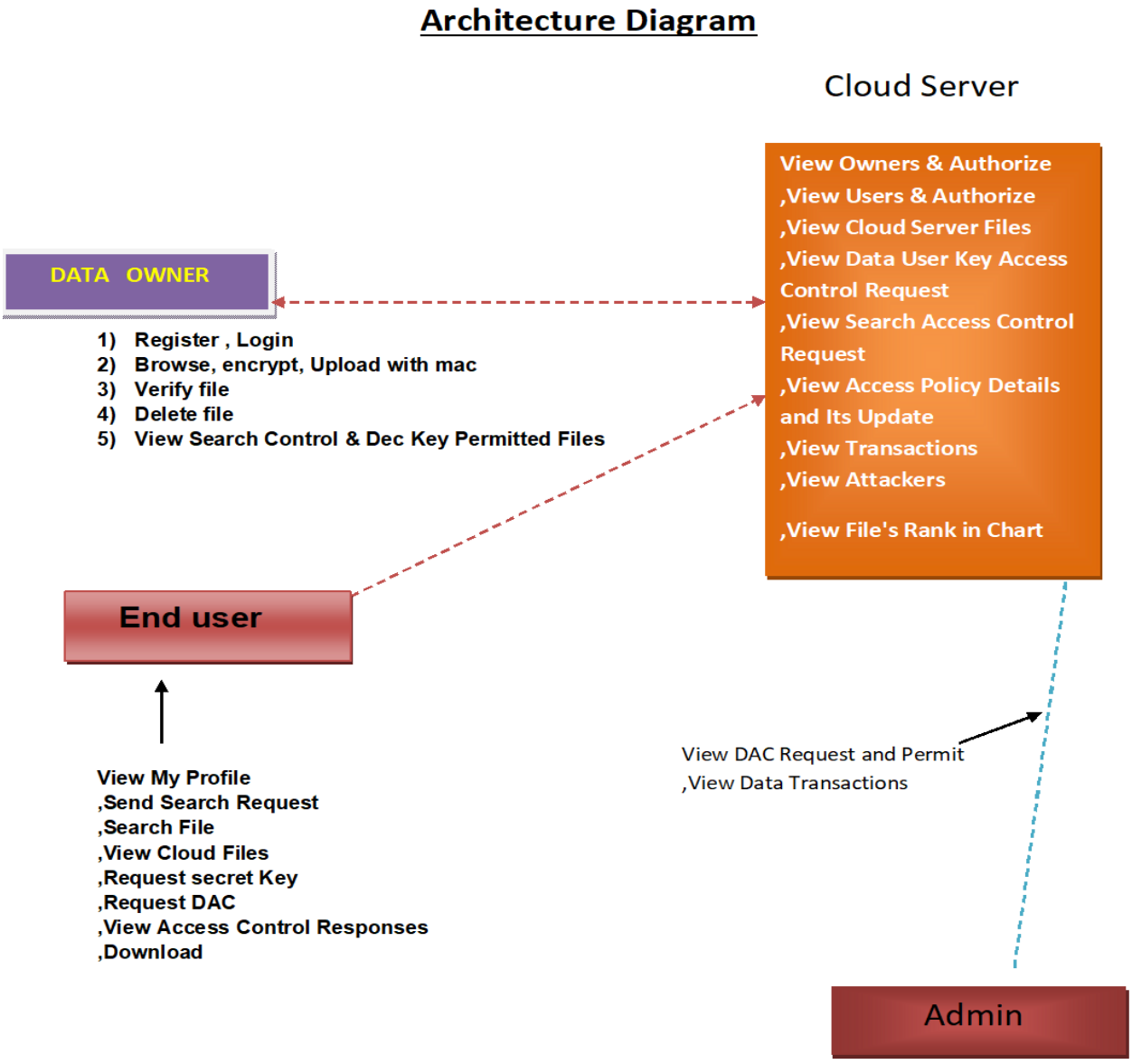
III.PROPOSED SYSTEM

- The proposed system presents Crypt-DAC, a cryptographically enforced dynamic access control system on un trusted cloud. Crypt-DAC delegates the cloud to update encrypted files in permission revocations. In Crypt-DAC, a file is encrypted by a symmetric key list which records a file key and a sequence of revocation keys. In a revocation, the administrator uploads a new revocation key to the cloud, which encrypts the file with a new layer of encryption and updates the encrypted key list accordingly. Same as previous works, we assume a honest-but-curious cloud, i.e., the cloud is

honest to perform the required commends (such as re-encryption of files and properly update previous encrypted **files**) but is curious to passively gathering sensitive information. Although the basic idea of layered encryption is simple, it entails tremendous technical challenges. For instance, the size of key list and encryption layers would increase as the number of revocation operations, which incurs additional decryption overhead for users to access files. To overcome such a problem, Crypt-DAC proposes three key techniques as follows.

- First, Crypt-DAC proposes delegation-aware encryption strategy to delegate the cloud to update policy data. For a file, the administrator appends a new revocation key at the end of its key list and requests the cloud to update this key list in the policy data. The size of the key list however increases with the revocation operations, and a user has to download and decrypt a large key list in each file access. To overcome this problem, we adopt the key rotation technique to compactly encrypt the key list in the policy data. As a result, the size of the key list remains constant regardless of revocation operations.

- Second, Crypt-DAC proposes adjustable onion encryption strategy to delegate the cloud to update file data. For a file, the administrator requests the cloud to encrypt the file with a new layer of encryption. Similarly, the size of the encryption layers increases with the revocation operations, and a user has to decrypt multiple times in each file access. To overcome this problem, we enable the administrator to define a tolerable bound for the file. Once the size of encryption layers reaches the bound, it can be made to not increase anymore by delegating encryption operations to the cloud. As a result, the administrator can flexibly adjust a tolerable bound for each file (according to file type, access pattern, etc.) to achieve a balance between efficiency and security.
- Crypt- DAC proposes delayed de-onion encryption strategy to periodically refresh the symmetric key list of the file and remove the bounded encryption layers over it through writing operations. In specific, the next user to write to the file encrypts the writing content by a new symmetric key list only containing a new file key, and updates the key list in the policy data. With this strategy, Crypt-DAC periodically removes the bounded encryption layers of files while amortizing the burden to a large number of writing users.



IV.SYSTEM ARCHITECTURE

V.IMPLEMENTATION

1. **Data Owner**

In this module, the data owner uploads their data with its chunks in the cloud server. For the security purpose the data owner encrypts the data file's chunks and then store in the cloud. The data owner can change the policy over data files by updating the expiration time. The Data owner can have capable of manipulating the encrypted data file. And the data owner can set the access privilege to the encrypted data file.

Dynamic Operation

Upload: is the operation to encrypt and upload the file

Delete: Is the operation to delete a corresponding data owner file in the cloud.

Verify: Verifying the data whether it is safe or not in the cloud.

2. Cloud Server

The cloud service provider manages a cloud to provide data storage service. Data owners encrypt their data files and store them in the cloud for sharing with data consumers. To access the shared data files, data consumers download encrypted data files of their interest from the cloud and then decrypt them. The end user request will be processes based on the queue.

End User

The Cloud User who has a large amount of data to be stored in multiple clouds and have the permissions to access and manipulate stored data. The end user sends the request for corresponding file request and it will be processed in the cloud based on the queue and response to the end user.

VI.CONCLUSION

We presented Crypt-DAC, a system that provides practical cryptographic enforcement of dynamic access control in the potentially untrusted cloud provider. Crypt-DAC meets its goals using three techniques. In particular, we propose to delegate the cloud to update the policy data in a privacy-preserving manner using a delegation-aware encryption strategy. We propose to avoid the expensive re-encryptions of file data at the administrator side using a adjustable onion encryption strategy. In addition, we propose a delayed de-onion encryption strategy to avoid the file reading overhead. The theoretical analysis and the performance evaluation show that Crypt-DAC achieves orders of magnitude higher efficiency in access revocations while ensuring the same security properties under the honestbut curious threat model compared with previous schemes.

REFERENCES

[1] J. Bethencourt, A. Sahai, and B. Waters, Ciphertext-policy attribute based encryption, in IEEE S&P, 2007.
[2] X. Wang, Y. Qi, and Z. Wang, Design and Implementation of SecPod: A Framework for Virtualization-based Security Systems, IEEE Transactions on Dependable and Secure Computing, vol. 16, no. 1, 2019.
[3] J. Ren, Y. Qi, Y. Dai, X. Wang, and Y. Shi, AppSec: A Safe Execution Environment for Security Sensitive Applications, in ACM VEE, 2015.
[4] V. Goyal, A. Jain, O. Pandey, and A. Sahai, Bounded ciphertext policy attribute based encryption, in ICALP, 2008.
[5] V. Goyal, O. Pandey, A. Sahai, and B.Waters, Attribute-based encryption for fine-grained access control of encrypted data, in ACM CCS, 2006.
[6] J. Katz, A. Sahai, and B. Waters, Predicate encryption supporting disjunctions, polynomial equations, and inner products, in EUROCRYPT, 2008.
[7] S. Muller and S. Katzenbeisser, Hiding the policy in cryptographic access control, in STM, 2011.
[8] R. Ostrovsky, A. Sahai, and B. Waters, Attribute-based encryption with non-monotonic access structures, in ACM CCS, 2007.
[9] A. Sahai, and B. Waters, Fuzzy identity-based encryption, in EUROCRYPT, 2005.
[10] T. Ring, Cloud computing hit by celebgate, http://www.scmagazineuk. com/cloud-computing-hit-by-celebgate/article/370815/, 2015.
[11] X. Jin, R. Krishnan, and R. S. Sandhu, A unified attribute-based access control model covering DAC, MAC and RBAC, in DDBSec, 2012.
[12] W. C. Garrison III, A. Shull, S. Myers, and, A. J. Lee, On the Practicality of Cryptographically Enforcing Dynamic Access Control Policies in the Cloud, in IEEE S&P, 2016.
[13] R. S. Sandhu, Rationale for the RBAC96 family of access control models, in proceedings of ACM Workshop on RBAC, 1995.
[14] T. Jiang, X. Chen, Q. Wu, J. Ma, W. Susilo, and W. Lou, Secure and Efficient Cloud Data Deduplication With Randomized Tag, IEEE Trasactions on Information Forensics and Security, vol. 12, no. 3, 2017.
[15] M. Kallahalla, E. Riedel, R. Swaminathan, Q. Wang, K. Fu, Plutus: Scalable Secure File Sharing on Untrusted Storage, in proceedings of USENIX FAST, 2003.