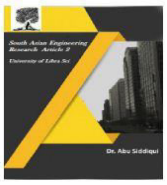
International Journal For Recent Developments in Science & Technology
2581-4575
Crossref
A Peer Reviewed Research Journal



A LIGHTWEIGHT SECURE DATA SHARING SCHEME FOR MOBILE CLOUD COMPUTING

[1]KOTHALANKA DINESH,[2]S.K.ALISHA

[1]MCA Student,B V Raju College, Bhimavaram,Andhra Pradesh,India

[2]Assistant Professor,Department Of MCA,B V Raju College,Bhimavaram,Andhra Pradesh,Indi

ABSTRACT

With the growing popularity of cloud computing, mobile devices have gained the capability to store and retrieve personal data from virtually anywhere at any time. However, this convenience introduces serious concerns regarding data security, particularly within mobile cloud environments. As a result, security issues have increasingly hindered the advancement of mobile cloud computing. Although numerous studies have focused on enhancing cloud security, most existing solutions are unsuitable for mobile devices due to their limited computing resources and battery power. Therefore, there is a pressing need for solutions that impose minimal computational overhead, specifically tailored for mobile cloud applications. To address this challenge, this paper proposes a Lightweight Data Sharing Scheme (LDSS) designed for mobile cloud computing. The proposed LDSS leverages Ciphertext-Policy Attribute-Based Encryption (CP-ABE), a widely used access control mechanism in traditional cloud settings. However, to adapt it for mobile environments, LDSS modifies the structure of the access control tree, making it more compatible with the resource constraints of mobile devices. A key innovation in LDSS is offloading the computationally intensive parts of CP-ABE, such as access control tree transformation, from mobile devices to external proxy servers. Additionally, LDSS tackles the complex issue of user revocation by introducing attribute description fields that enable a lazy-revocation mechanism. This approach significantly reduces the cost and complexity typically associated with revoking user access in traditional CP-ABE systems. Experimental results demonstrate that LDSS effectively minimizes the computational burden on mobile devices while maintaining secure and efficient data sharing in mobile cloud environments.

Keywords: Data Security, Lightweight Data Sharing Scheme (LDSS), Access Control,Secure Data Sharing, Computational Overhead, Mobile Data Privacy

I.INTRODUCTION

With the rapid development of cloud computing and the widespread adoption of smart mobile devices, people are increasingly embracing a new data-sharing model where personal data is stored in the cloud and accessed via mobile devices. While mobile devices offer portability and convenience, they typically have limited storage capacity and computational power. In contrast, cloud service providers (CSPs) offer abundant resources, making cloud storage an appealing solution for managing and sharing data. Leveraging these resources, mobile users can conveniently upload and retrieve files—such as photos, videos, and documents—on the cloud, and share them with others. In such applications, users (referred to as data owners) expect CSPs to provide robust data management functionalities, including flexible sharing

controls. Given the sensitive nature of personal data, data owners often wish to restrict access to only selected individuals (data users). However, the privacy and security of this data is a growing concern. The access control mechanisms currently offered by CSPs are often insufficient or inconvenient. For instance, when users upload data to the cloud, they relinquish direct control over it. This raises concerns about CSPs potentially misusing or accessing the data for commercial or other interests. Additionally, sharing encrypted files with specific users often involves manually distributing passwords, which can be cumbersome and inefficient. One way to simplify this process is by categorizing data users into groups and sharing group passwords. However, this approach still lacks fine-grained access control and suffers from challenges in password management. Encrypting sensitive data before uploading it to the cloud is a common security measure, but it introduces new issues—most notably, how to efficiently enforce access control on the encrypted data so that only authorized users can decrypt and view the content. Moreover, data owners need mechanisms that allow them to easily manage user privileges, including granting or revoking access as needed. Many studies have addressed access control over encrypted data in the cloud. These studies typically assume that the CSP is "honest-but-curious" (i.e., follows protocols but may attempt to access data), and that all sensitive data is encrypted before being uploaded. Access is generally managed by distributing encryption/decryption keys. Based on their techniques, these solutions can be grouped into four categories: simple ciphertext access control, hierarchical access control, fully homomorphic encryption-based control, and attribute-based encryption (ABE).

While effective in traditional cloud environments, these schemes are not suited for mobile cloud computing due to their high computational and storage demands. For example, experiments show that ABE operations on mobile devices can be extremely slow—up to 27 times slower than on a desktop or laptop. An encryption task that takes one minute on a personal computer could take about half an hour on a smartphone. Additionally, most existing methods do not handle user privilege changes efficiently, often resulting in costly revocation processes, which further limits their practicality on mobile platforms. Clearly, there is a lack of suitable solutions for secure data sharing in mobile cloud environments. As mobile cloud computing becomes more prevalent, there is an urgent need for lightweight, efficient, and secure data-sharing mechanisms tailored to mobile devices. To address this challenge, this paper introduces a Lightweight Data Sharing Scheme (LDSS) designed specifically for mobile cloud computing environments. The key contributions of LDSS are as follows:

We design LDSS-CP-ABE, an algorithm based on Ciphertext-Policy Attribute-Based Encryption (CP-ABE), which enables fine-grained and efficient access control over encrypted data. To minimize the computational burden on mobile devices, LDSS offloads resource-intensive encryption and decryption operations to proxy servers. A version attribute is also introduced in the access structure to preserve data privacy, and the decryption key format is modified to ensure secure communication with proxy servers.

We propose a lazy re-encryption strategy using descriptive fields for attributes, significantly reducing the overhead associated with user revocation—a known challenge in CP-ABE systems. A prototype framework based on LDSS is implemented and tested. Experimental results demonstrate that LDSS substantially reduces client-side overhead with only minimal additional cost on the server side. Furthermore, LDSS outperforms existing ABE-based access control solutions in terms of performance and practicality for mobile cloud environments.

The rest of the paper is organized as follows: Section 2 discusses fundamental concepts related to secure mobile cloud data sharing and outlines the security assumptions. Section 3 details the design of LDSS. Sections 4 and 5 present the security analysis and performance evaluation, respectively. Section 6 reviews related work, and Section 7 concludes the paper and outlines future research directions.

II.LITERATURE SURVEY

[1] Gentry and Halevi (2011) implemented the first fully homomorphic encryption (FHE) scheme, which allows computation on encrypted data without decryption. Although groundbreaking, the scheme was computationally intensive, making practical deployment in real-world applications particularly challenging, especially for resource-limited devices such as smartphones.

[2] Brakerski and Vaikuntanathan (2011) proposed an improved FHE scheme based on the Learning With Errors (LWE) problem, which significantly enhanced efficiency. This work brought FHE closer to practicality by simplifying the bootstrapping process and reducing computational overhead.

[3] Wang and Jin (2011) focused on data leakage mitigation in collaborative cloud environments using discretionary access control. They proposed a model to limit data exposure while still supporting collaboration, addressing privacy concerns in enterprise cloud usage.

[4] Skillen and Mannan (2013) explored deniable storage encryption for mobile devices. Their work aimed at enabling users to hide sensitive data effectively, even under coercion, thus contributing a valuable security layer for mobile cloud storage systems.

[5] Wang et al. (2009) discussed secure and efficient access mechanisms to outsourced data in the cloud. They presented methods to preserve confidentiality and data access efficiency, addressing core challenges in cloud data outsourcing.

[6] Maheshwari, Vingralek, and Shapiro (2000) introduced a trusted database system that operates over untrusted storage. This system ensured data integrity and security, even when the underlying storage infrastructure could not be fully trusted.

[7] Yang, Jia, and Ren (2013) proposed an attribute-based encryption scheme with efficient revocation for cloud storage. This model supports fine-grained access control and enables the revocation of access rights without compromising system performance or security.

[8] Crampton, Martin, and Wild (2006) analyzed hierarchical access control

schemes and developed an efficient key assignment approach. Their work is critical in managing large-scale user roles and permissions in organizational settings.

[9] Shi et al. (2007) introduced a scheme for executing multi-dimensional range queries over encrypted data. Their solution preserved query functionality and data privacy, which is vital for encrypted database systems and cloud-based applications.

[10] Wang et al. (2012) tackled the challenge of similarity search over encrypted data. They developed a method that ensures privacy while still enabling practical, user-friendly data retrieval from encrypted cloud storage.

[11] Yu et al. (2010) presented a scalable and fine-grained data access control mechanism tailored for cloud computing. Their framework was based on attribute-based encryption and aimed at balancing scalability, flexibility, and security.

[12] Yang et al. (2013) introduced DAC-MACS, a multi-authority access control scheme that supports secure and effective data sharing in cloud environments. This system enhances security by allowing multiple trusted authorities to manage different attribute sets.

[13] Stehlé and Steinfeld (2010) developed a faster FHE scheme that improved on existing models by optimizing key components and reducing runtime. This progress was significant for the adoption of homomorphic encryption in performance-sensitive applications.

[14] Lai, Deng, and Li (2014) designed a fully secure key-policy attribute-based encryption system that maintains constant-size ciphertexts and supports rapid decryption. Their work greatly enhanced the performance and usability of ABE systems in mobile and cloud settings.

[15] Bethencourt, Sahai, and Waters (2007) pioneered the ciphertext-policy attribute-based encryption (CP-ABE) model, where access policies are embedded in the ciphertext. This approach provides a flexible and scalable method for enforcing access control in cloud environments.

III.PROPOSED MECHANISM

In this section, we present the design of LDSS, a lightweight data-sharing scheme tailored for mobile cloud environments. LDSS is built upon a modified CP-ABE (Ciphertext-Policy Attribute-Based Encryption) algorithm, called LDSS-CP-ABE, and consists of several interconnected components: Data Owner (DO), Data User (DU), Trust Authority (TA), Encryption Service Provider (ESP), Decryption Service Provider (DSP), and Cloud Service Provider (CSP). The DO uploads encrypted data to the cloud and defines access control policies, while the DU retrieves data based on their granted privileges. The TA generates attribute keys for users, the ESP handles encryption operations for DOs, and the DSP supports decryption operations for DUs. The cloud stores encrypted data and access policies but is considered semi-trusted, making pre-encryption necessary.

To secure data sharing, the LDSS framework employs symmetric encryption for data files and encrypts the symmetric keys using ABE. The access policy, defined

as an access control tree, is embedded within the ciphertext of the symmetric key. Only DUs whose attribute keys satisfy the access policy can retrieve and decrypt the symmetric key. Since cryptographic operations are computationally intensive, ESP and DSP are introduced to offload processing from mobile devices. The modified LDSS-CP-ABE algorithm ensures privacy is maintained even when encryption and decryption tasks are outsourced.

The LDSS-CP-ABE algorithm is structured using the following key concepts: *attributes*, *access control trees*, and *version attributes*. Attributes are defined by DOs and assigned to DUs, representing access privileges. Access control policies are represented as access control trees, where leaf nodes are attributes and non-leaf nodes are logic gates (AND, OR, threshold). A version attribute, introduced in LDSS, is a security enhancement that ensures data integrity even when attributes change. It forms a new root in the access tree, combining the original tree with the versioning mechanism.

The algorithm includes four main functions: Setup() generates public and master keys; KeyGen() produces attribute keys for DUs; Encryption() encrypts the symmetric key using the access policy tree; and Decryption() decrypts the key if the DU's attributes match the access policy. TA is responsible for running Setup() and key distribution, while ESP and DSP assist in offloading the heavy computational operations from users.

To enhance privacy and enable dynamic privilege management, LDSS introduces an attribute description field—a binary string representing attribute status. It has three variants: one each for DO, DU, and data files. The DO's attribute description field is created by TA during registration and stored securely. The DU's field is maintained by both TA and the cloud and contains up-to-date information on the DU's privileges. For data files, the field indicates which attributes are required for access; revoked attributes are marked with a '#'. These fields help maintain access policy secrecy and facilitate updates without exposing sensitive control strategies.

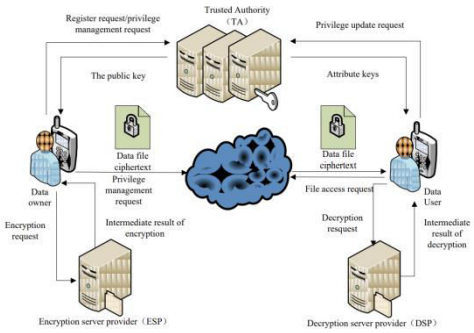
The LDSS framework operates through several key stages. During **system initialization**, TA generates the cryptographic keys and stores attribute information. In the **file sharing** phase, DO encrypts data using symmetric encryption, then encrypts the symmetric key with ESP's help, and uploads everything to the cloud. **User authorization** involves TA verifying and updating DU's attribute keys based on their current status. If discrepancies exist between TA and DU attribute fields, appropriate updates or revocations are applied.

For **file access**, DUs request data from the cloud. If they satisfy the policy, the encrypted file and key are sent. The DU, with DSP's help, decrypts the symmetric key and then accesses the data. **Privilege revocation** allows DOs to revoke specific attributes from a DU by updating records in TA and cloud databases. However, immediate re-encryption is not performed to reduce system overhead.

Finally, LDSS adopts **lazy re-encryption** for efficient file and attribute updates. When a DO updates a file and the file includes revoked attributes (marked as '#'), only then is re-encryption triggered. TA generates new values for the updated attributes,

updates the keys, and provides a new public key to the DO. The DO re-encrypts the file, ensuring revoked DUs cannot access the updated content. This deferred approach balances security and performance, ensuring revoked users cannot access new data while avoiding unnecessary computational costs.

SYSTEM ARCHITECTURE:



IV.CONCLUSION

In recent years, access control in cloud environments has gained significant attention, with many approaches relying on attribute-based encryption (ABE). However, traditional ABE methods are not ideal for mobile cloud scenarios due to the heavy computational demands they place on devices with limited processing capabilities. To overcome this challenge, we introduced LDSS, a lightweight data-sharing scheme designed specifically for mobile cloud environments. By incorporating the LDSS-CP-ABE algorithm, our framework effectively offloads the computational burden from mobile devices to proxy servers, enabling secure and efficient data sharing. Experimental evaluations demonstrate that LDSS not only preserves data privacy but also significantly reduces the processing overhead for mobile users. As part of our future work, we plan to explore mechanisms for ensuring data integrity within this framework and investigate methods for performing ciphertext retrieval to enhance the functionality and usability of data sharing in mobile cloud systems.

V.REFERENCES

[1] C. Gentry and S. Halevi, “Implementing Gentry’s Fully-Homomorphic Encryption Scheme,” *Advances in Cryptology – EUROCRYPT 2011*, Springer, Berlin, Heidelberg, pp. 129–148, 2011.

[2] Z. Brakerski and V. Vaikuntanathan, “Efficient Fully Homomorphic Encryption from (Standard) LWE,” in *Proceedings of the IEEE Symposium on Foundations of Computer Science*, California, USA, IEEE Press, pp. 97–106, Oct. 2011.

[3] Q. Wang and H. Jin, “Data Leakage Mitigation for Discretionary Access Control in Collaboration Clouds,” in *Proceedings of the 16th ACM Symposium on Access Control Models and Technologies (SACMAT)*, pp. 103–122, Jun. 2011.

[4] A. Skillen and M. Mannan, “On Implementing Deniable Storage Encryption for Mobile Devices,” in *Proceedings of the 20th Annual Network and Distributed System Security Symposium (NDSS)*, Feb. 2013.

[5] W. Wang, Z. Li, R. Owens, et al., “Secure and Efficient Access to Outsourced Data,” in *Proceedings of the 2009 ACM Workshop on Cloud Computing Security*, Chicago, USA, ACM Press, pp. 55–66, 2009.

[6] U. Maheshwari, R. Vingralek, and W. Shapiro, "How to Build a Trusted Database System on Untrusted Storage," in *Proceedings of the 4th Symposium on Operating System Design and Implementation (OSDI)*, USENIX Association, pp. 10–12, 2000.

[7] K. Yang, X. Jia, and K. Ren, "Attribute-Based Fine-Grained Access Control with Efficient Revocation in Cloud Storage Systems," in *Proceedings of ASIACCS 2013*, pp. 523–528, 2013.

[8] J. Crampton, K. Martin, and P. Wild, "On Key Assignment for Hierarchical Access Control," in *Computer Security Foundations Workshop*, IEEE Press, pp. 14–111, 2006.

[9] E. Shi, J. Bethencourt, T. H. H. Chan, et al., "Multi-dimensional Range Query over Encrypted Data," in *Proceedings of the IEEE Symposium on Security and Privacy (SP)*, pp. 350–364, 2007.

[10] C. Wang, K. Ren, S. Yu, and K. M. Raje Urs, "Achieving Usable and Privacy-Assured Similarity Search over Outsourced Cloud Data," in *Proceedings of IEEE INFOCOM 2012*, Orlando, Florida, Mar. 25–30, 2012.

[11] S. Yu, C. Wang, K. Ren, and W. Lou, "Achieving Secure, Scalable, and Fine-Grained Data Access Control in Cloud Computing," in *Proceedings of IEEE INFOCOM 2010*, pp. 534–542, 2010.

[12] K. Yang, X. Jia, K. Ren, B. Zhang, and R. Xie, "DAC-MACS: Effective Data Access Control for Multi-authority Cloud Storage Systems," *IEEE Transactions on Information Forensics and Security*, vol. 8, no. 11, pp. 1790–1801, 2013.

[13] D. Stehlé and R. Steinfeld, "Faster Fully Homomorphic Encryption," in *Proceedings of the 16th International Conference on the Theory and Application of Cryptology and Information Security*, Springer, Singapore, pp. 377–394, 2010.

[14] J. Lai, R. H. Deng, Y. Li, et al., "Fully Secure Key-Policy Attribute-Based Encryption with Constant-Size Ciphertexts and Fast Decryption," in *Proceedings of the 9th ACM Symposium on Information, Computer and Communications Security (ASIACCS)*, pp. 239–248, Jun. 2014.

[15] J. Bethencourt, A. Sahai, and B. Waters, "Ciphertext-Policy Attribute-Based Encryption," in *Proceedings of the 2007 IEEE Symposium on Security and Privacy (SP)*, Washington, USA: IEEE Computer Society, pp. 321–334, 2007.