A QUICK AND SAFE WAY TO SHARE INFORMATION IN THE CLOUD FOR MOBILE DEVICES

Mr. Suman B[1], Dr.V.Anantha Krishna Pranathi Sv[2], P. Sruthi Reddy[3], Maliha Shaik[4]

Article History: Received: 11.02.2023 **Revised:** 26.03.2023 **Accepted:** 11.05.2023

Abstract

The proliferation of cloud computing has made it feasible for users to access and update their personal data from any location using a mobile device. This escalates the already serious data security issue in mobile cloud, which stymies its progress. Much research has been done to improve cloud safety. Nevertheless, due to the restricted computational resources and power of mobile devices, they are not suited for clouds on mobile devices. Cloud applications on mobile devices have a tremendous demand for solutions with reduced computational overhead. For this reason, we propose a Lightweight Data-Sharing-SchemeLDSS for mobile cloud computing. It takes use of CPABE, a standard cloud-based access control method, but modifies the tree structure of that method so that it may be utilized in mobile cloud settings. With CPABE, the access control tree transformation is a computationally costly operation, however with LDSS, this task is offloaded to external proxy servers.

Keywords: Lightweight Data Sharing Scheme, Ciphertext Policy Attribute Based Encryption

[1]Assistant Professor, Computer Science and Engineering, Sridevi Women's Engineering College, B.Tech IVYear Hyderabad, India.
[2,3,4]Computer Science and Engineering, Sridevi Women's Engineering College, B.TechIVYear Hyderabad, India

Email: [1]swecsuman@gmail.com, [2]krishnaananthav@gmail.com

DOI: 10.31838/ecb/2023.12.s3.299

1. Introduction

People are gradually adjusting to a new age of data sharing as a result of the increase in the usage of cloud computing and the ubiquity of mobile devices. The paradigm in which information is kept in the cloud and accessed through these devices. There is often not a lot of room for data or processing power on mobile devices. In fact, the cloud contains an abundance of resources. In this case, using the Cloud-Service-Providers (CSP) facilities for data storage and sharing is crucial for achieving the desired performance. Many cloud-based mobile apps are now commonly utilized. Those who own the data may use these programs to store and share their files in the cloud with anybody else who has an interest in seeing or using that data. Cloud-Service-Providers (CSPs) also provide data management features to their clientele. Owners of the datacan make their files accessible to the public or can restrict access to just authorized individuals, given the sensitive nature of personal data. It's obvious that protecting the privacy of the sensitive personal data is a top priority for many owners of data. The cutting-edge tools for managing privileges and restricting access that the CSP offers are insufficientand inconvenient. They are not able to fulfil all of the requirements of the owners of data. To start, users risk having their data spied on by their CSP since it is stored in a location over which they have no control over information related to the users for its own business purposes and / or other uses. Secondly, it is exceedingly inconvenient to have to give a password for each user if the data owner only wants to share their encrypted data to certain users. The data owner may simplify permission control by assigning data users to certain groups and then sending passwords to those groups. But, granular access control is necessary for this method. Managing passwords is a major problem in both contexts. To avoid the aforementioned issues, it seems prudent to encrypt sensitive personal data before storing it on the cloud. Data encryption, however, introduces other challenges. It's difficult to figure out how to implement an effective access-control mechanism on ciphertext-decryption, ensuring that onlyusers that are authorized may see the plaintext data. The system should also provide efficient user privilege management functionality, allowing data owners to quickly and effectively control which users have access to which data. Data access control over ciphertext is an area that has seen much study. The following are some of the assumptions used in these studies. As a first tenet, the CSP is trustworthy and inquisitive. Second, security is ensured by encrypting all data before to storage in the Cloud. Key distribution for encrypted and decrypted data is the third method of user permission. These methods may be broken down into four broad classes: attribute-based access-control, hierarchical access-control, completely homomorphic encryption,and access-control based on ciphertext encryption (ABE). All such ideas are geared at The cloud does not facilitate mobility. They have a high caloric amount of memory and processing power, both of which are Currently unavailable on mobile platforms. As stated in the using data from experimentation, we find that the Mobile device use has surpassed that of desktop and PCs that sit on a desk. It would take at least twenty seven times as long to carry out on a mobile phone than on a desktop (PC). This signifies that a single step is required for an encrypted operation for a couple of minutes on a computer will need halfanhour on a handheld electronic gadget. In addition, the problems that are being addressed by the issue of a user's permissions being changed successfully. Like this operation may result in very expensive cancellation fees. It's this invalid also for use on mobile devices. Without a doubt, there is currently no adequate means of solving the security Mobile cloud data sharing issues. When the cellphone Once cloud computing gains traction, it will be able to method for mobile cloud data exchange that is both efficient and safe needs immediately. In this study, we present a solution to this problem. The Mobile Lightweight Data Sharing Protocol (LDSS) On the cloud, you may do your work.Some of LDSS's most notable results are as follows:

In the first part, we develop an algorithm (LDSS-CPABE) based on the ABE technique of "Attribute-Based-Encryption"efficient ciphertext access control. Two, we encrypt data via proxy servers and operations for decrypting. We take a computational Proxies are used for much of the work in ABE. hosts, which drastically cut down on computational more work for mobile clients. Nonetheless, in In accordance with the LDSS-CPABE, the authorization framework was modified to include a version element.The decryption key'sformat is altered so that it may be sent to safe and sound use of proxy servers. Thirdly, we provide sluggish reencryption and description field of qualities to lessen the burden of revocation addressing the issue of removing a user's access. Fourth, we prototype a system for exchanging information. structure LDSS-based. Experiments have shown that The LDSS protocol may drastically cut down on client-side processing time, This imposes just a little further expense on the On the back end, or server. This method is useful for setting up a feasible mobile data sharing security protocols.

2 Related Work

2.1. How to Make Mobile Devices Use Deniable Storage Encryption

Encryption is a reliable method for keeping private information secret. Users might be forced to provide their decryption keys, making this insufficient in

certain scenarios. Thus, secrecy is required so that the information's mere existence may be denied. To circumvent detection, steganographic methods and obfuscated encryption protocols were developed. The potential and effectiveness of deniable storage encryption for mobile devices are investigated according to current growth of smart phones and other devices. In this paper, we assess the risks posed to mobile PDE by both well-established and emerging threats. Mobiflage is a technique developed to circumvent these problems by concealing encrypted volumes amid seemingly-random data that is stored on the external storage of a mobile device. Issues with deniable encryption on desktops are studied in order to inform the development of new defenses against attacks to mobile systems. There is little effect on throughput from Mobiflage's deniable file systems, no data growth from efficient storage utilization, and a restriction/prevention of recognized causes of leakage and disclosure. To evaluate the viability and efficiency of Mobiflage, we give a proofof the concept for the implementation of the Android operating system. To prevent deniability from being compromised by other leaks and collusion, a set of recommended practices for users to follow has been created. Images of protests and riots are increasingly being captured and shared through mobile devices. Deniable storage encryption might provide a practical technological answer to the problem of authorities gaining access to these concealed documents. Several PDE-enabled storage devices are available for popular desktop/laptop OSes. Perhaps more beneficial to everyday users and human rights advocates, Mobi- flage investigates the design and implementation problems of PDE for all devices. Mobiflage's architecture is informed in part by research into the vulnerabilities and assaults that have been made against desktop PDE implementations. Mobile-specific issues (such a compromised Internet Service Provider or cellphone carrier) are also taken into account. The user must meet specific criteria in order to overcome these obstacles. Deniability-weakening information leakage is avoided by compiling a set of rules the user must adhere to. The design of the Mobiflage cannot be guaranteed to be entirely secure against any leaks, even if users follow all these instructions. We don't want to lull anybody to sleep with a false feeling of security. To inspire further research on mobile systems with PDE support, we provide Mobiflage. On request, we may provide the underlying source code for our prototype implementation.

2.2. Outsourced Data Access That Is both Safe And Effective

Cloud computing relies heavily on the safe and quick access it provides to vast amounts of outsourced data. To address this issue in owners-write-and-users-read applications, we suggest a technique in this work. To provide versatile cryptography-based access control, we suggest to encrypt each data block with a unique key. The use of key derivation techniques reduces the number of secrets the owner must keep hidden. The analysis demonstrates that the computing cost introduced by the hash function-based key generation technique is negligible. To stop the revoked users from obtaining access to current data, we propose using overencryption and/or slow revocation. Mechanisms for handling changes to outsourced data and authorized users are something we create. We look at the suggested method's overhead and security and analyse ways to boost data access efficiency.In owner_write_users_readsituations, we present a technique for achieving safe and efficient access to the data that is outsourced. To keep costs down for both the owner and the service-provider, we assume the outsourced data is massive. To implement scalable cryptography-based access control, we suggest to encrypt each data block with a unique key. By using a key derivation technique, the proprietor has to keep track of fewer secrets. Key derivation using hash functions is shown to have negligible overhead by the analysis. To stop the revoked users from obtaining access to current data, we propose using overencryption and/or slow revocation. Mechanisms for handling changes to outsourced data and authorized users are something we create. We examine the method's computational, storage, and communication costs. We also look at how well the method scales and how secure it is. The following are some ways in which we've improved upon our original strategy. To begin, we want to build a new key management scheme based on this method that can be used in many-write-many-read scenarios. Second, we'd want to create dynamic mapping functions between hierarchy key values and data block index numbers so that we may gradually restructure the data blocks according to their access patterns. As a result, we can cut down on the number of keys the owner gives to the consumer even more. Lastly, we want to establish a novel way to secure Storage-as-a-Service by integrating current techniques to access control, proved data ownership, and key management for outsourced data.

2.3. How to Create a Reliable Database with Insecure Media

For certain new uses, it's necessary for programs to store private data on untrusted servers. In this work, we discuss the design and implementation of a trusteddatabase-system – TDB, that uses a negligible amount of storage to secure a vast quantity of scalable and untrusted storage. Applications that are not trusted are unable to access or edit the database without being detected since it is encrypted and verified against a collision-resistant hash stored in trusted storage. Unlike solutions built on top of a traditional database system, TDB secures data and metadata uniformly by integrating encryption and hashing with a low-level data architecture. Synergies

between hashing and log-structured storage are used in the implementation. To bolster the viability of the TDB architecture, preliminary performance results suggest that TDB performs better than a commercially available embedded database system.To extend tamper-detection and secrecy to a scalable quantity of untrusted storage, we have introduced a trusted database system that makes use of a trusted processing environment and a limited amount of trusted storage. Data and information are universally protected thanks to the architecture's incorporation of encryption and hashing with a low-level data model. Transactions, backups, and indexing are just some of the advanced database operations that may be supported by this approach. We discovered that log-structured storage works well for this purpose. Log-structured systems' central location map also includes a hash tree, simplifying implementation by allowing items to be checked as they are found. Checkpointing is an optimization that delays and centralizes hash value propagation along a tree. As changes are not applied directly to the data, copy-on-write makes it possible to take a "snapshot" of the database at any given time, making incremental backups possible. We used both microbenchmarks and a large-scale workload to evaluate TDB's performance. Writes to the untrusted store and the tamper-resistant store accounted for the bulk of the database's overhead, but this might very well change depending on the devices being utilized. Encryption and hashing accounted for just 6% of the overall overhead. TDB outperformed a system that adds encryption to a commercial embedded database system in terms of performance and security on this workload. This demonstrates the TDB architecture's viability.

2.4. Attribute-Based Digital Rights Management System with Efficient Cancellation in the Cloud

Existing cloud-based digital rights management (DRM) solutions impose a significant computational burden on content providers in order to distribute keys. In this research, we combine the methods of the ciphertext-policy, attribute-based encryption (CPABE) and proxy reencryption to propose an attribute-based Digital Rights Management strategy for use in cloud computing (PRE). The content encrypting key is initially split into the content master key and the content helper key. The content master key is then securely dispersed using attribute-based access controls. The content master key may be recovered by users who meet the access policy's requirements, who can then use the key server's helper key to decrypt the material. By empowering the attribute authority to instruct the key server to withhold the assistant key from the revoked users, we are able to efficiently accomplish both attribute and user revocation. The suggested method has been shown to be secure, efficient, and private via various performance and security assessments.Cloud computing represents a paradigm shift in the IT industry and has the potential to drastically alter the way the sector does business. In this research, we present an attribute-based digital rights management (DRM) approach by fusing ciphertext-policy, attribute-based encryption with proxy reencryption. To accomplish finegrained access control and privacy protecting, we create and implement access policy based on characteristics in the proposed system. Users who meet the policy requirements may access the key server to get the helper key and the content master key to decrypt the data. To further improve the efficiency of user and revocation of an attribute, we enable the attribute authority to transfer most of the revocation operations to the key server without releasing the assistant key. Our technique is both more efficient and more secure than competing DRM approaches on the cloud. In the future, we want to work on improving cloud computing's capabilities for dynamic use management.

2.5. Searching for Multi-Keyword Similarity in the Cloud While Protecting User Privacy

Data production in both the private and public sectors is rising exponentially. The new cloud computing paradigm allows for the outsourcing of data and the associated complicated administrative duties, providing more management flexibility and cost savings. Due to the potentially sensitive nature of the data, direct data outsourcing would provide a privacy leakage risk. Prior to data outsourcing, encryption may be utilized to alleviate any worries about the viability of cloud-based processing. We think about a similarity search with many keywords over data stored in the cloud. In particular, the user specifies many keywords for usage solely with the text data. Using the edit distance measure, the cloud provides the files that include more than the specified number of the input keywords or ones with a sufficiently high degree of similarity. We suggest three alternatives, including blind signature for user access privacy and a unique use of the Bloom_filter bit pattern to accelerate the search work on the cloud side. Our final search-achievement architecture is both cloud-side search-time efficient and safe from insider attacks. We employ analysis and assessment of performance to show that our ideas have a chance of success in the real world.

For the first time, we take into account PPMSSPrivacy_Preserving_Multikeyword_Similarity_Search for data stored in the cloud. PPMSS-1, PPMSS-2, and PPMSS-3 are offered as possible solutions to this search issue using the keyword suppressing approach and the Bloom filter. In particular, PPMSS-3 has very low resource requirements in terms of memory, processing power, and network traffic. To further safeguard the privacy of our users, we develop a blind signature-based user authorisation technique. Our analysis shows that the

suggested systems have the potential to be realistically beneficial in providing PPMSS.

2.6. Controlling who may see what data in the cloud in a way that is secure, scalable, and fine-grained

Existing solutions often use cryptographic techniques, providing data decryption keys only to authorized users, to protect sensitive user data from services that are not trusted. When finegrained data access control is desired, however, such solutions unavoidably introduce a heavy computational overhead on the owner regardingthe distribution of keys and management of data and therefore do not scale well. Access control that is both granular and scalable, while still protecting sensitive information, is an open challenge. As users outsource data storage and sharing on cloud servers, this article proposes certain services for data safety and access control. This paper solves this difficult open problem by defining and carrying out policies for access based on the attributes of the data, and by letting the owner consign most of the computationalfunctions involved in finegrained data access-control to unfrosted cloud servers without revealing the rudimentary contents. Our suggested method allows the owner to safely consign the reencryption of data files to cloud servers while protecting sensitive user access privileges. To this end, we use a novel combination of proxy reencryption, slow re-encryption, and attribute-based encryption (ABE). For data access-control in the cloud, our suggested method is fine-grained, scalable, and secures sensitive information while maintaining the privacy of individual users. Comprehensive study demonstrates the superior efficiency and provable security of our suggested strategy in comparison to other security methods.

Cloud computing with granular permissions for accessing data. One difficulty is that existing work does not give a solution that concurrently addresses the need for fine-grainedness, data secrecy, and scalability. In this research, we suggest a strategy to do this by making use of KPABE in conjunction with proxy reencryption and lazy reencryption in a novel way. In addition, our suggested system may allow the data owner to offload most of the heavy lifting of processing to efficient cloud servers. The identities of authorized users and their secret keys may be kept secure. Our suggested approach has been shown to be safe through formal security proofs using industry-standard cryptographic models.

2. Methodology

Here, we lay out the framework for the LDSS system. First, we introduce LDSS and provide several examples. operations of the LDSS-CPABE algorithm and system, which are what the LDSS algorithm relies on. Lastly, we outline what LDSS is. Namely; in finer detail. For the mobile cloud, we propose LDSS as a framework for lightweight datasharing. There are the the six elements below.

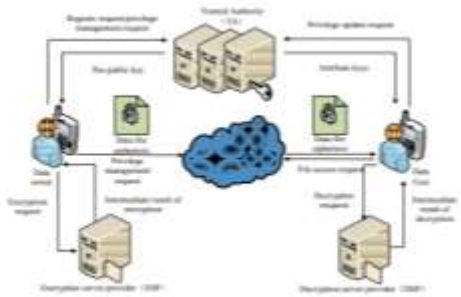
(1) Data-Owner (DO): The owner sends information to the mobile device. cloud and publish it for social sharing. Access is controlled by DO. regulation procedures.

(2) Data-User (DU): A user accesses mobile data by inputting a cloud.

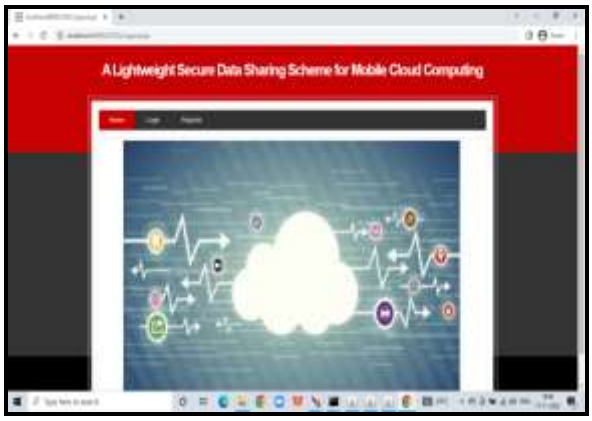
(3) The Trustee_Administrator (TA) is responsible for producing and handing out keys to identifying attributes.

(4) An encryption_service_provider (ESP) offers DO's Encryption Process Encrypting Data.

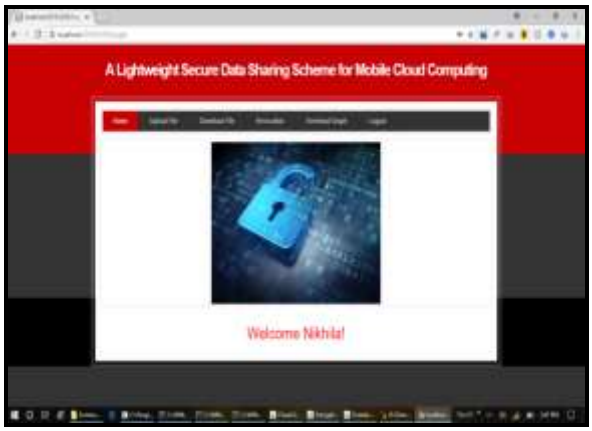
(5) There's the DSP, or decryption_serviceprocessing of decrypted data for DU. Data storage is handled by a (6) Cloud-Service-Provider (CSP) for the owner. It carries out the instructions given by While it is possible for DO to access the information stored in meaning "the cloud" here.A DO uploads information to the internet. Since If you store information in the cloud, you should encrypt it first. Uploading it now. Access policy in the DO is defined by the structure of permissions recursion (for definition in order to designate which features a user should contain if he needs to access a specific file, he must first obtain. What's LDSS all about? Using symmetric encryption, all data files are secured. data encryption mechanism, where the symmetric key is additionally encoded with attribute-based-encryption (ABE). The ciphertext contains the access policy that will be enforced. This is the symmetric key, or equivalent. Any user who has access to the attribute keys in accordance with the rules of the access control policy text and decipher it to get the symmetric key. As the Computing is required for both encryption and decryption. Intensive, they add a lot of extra work for people who use mobile devices. In order to lessen the load on the mobile clients, server-side encryption and decryptionservice-provider carrier (DSP) are employed. Each of the encryption tool the encryption service-provider and the decryption service-provider semitrusted. We put our own spin on the classic CPABE LDSS-CPABE algorithm, and develop it in order to make sure your data is secure when using a third-party computational responsibility for these tasks to ESP and DSP.



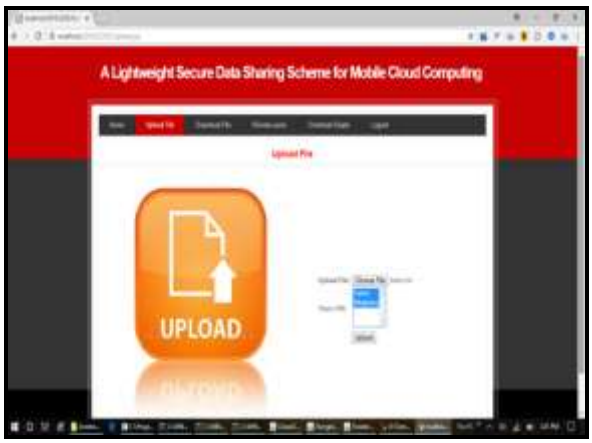
3. Result and Discussion



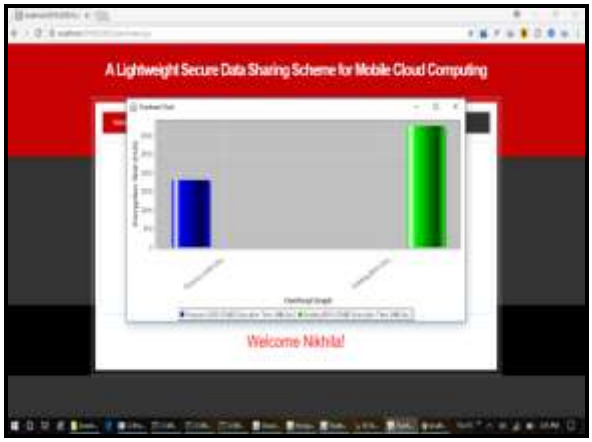
A login/register button is prominently featured on the homepage.The usermust register into the website before being able to get access to the features provided.



Only after a successful login will the welcome page be shown.



The website now only supports the encryption of text files. This means that the user must only upload text files. Upon uploading the material, users may also decide who will have access to it.



On the side of the proposed system, encryption time and overhead are reduced relative to the conventional method, as shown by the overhead graph.

4. Conclusion

Several recent works on cloud access control have relied on attribute-basedencryption algorithms (ABE).Nevertheless, conventional ABE does not work well with mobile cloud. as it requires a lot of processing power and the capabilities of mobile devices are restricted. In this work, we advocate for LDSS as a solution to this problem. It is able to address the issue of safe data sharing in mobile cloud by using an unique LDSS-CPABE method to shift the burden of computing from mobile devices to servers. Data privacy can be protected in the mobile cloud, and user burden may be lessened, as shown by the experiments. New methods for protecting the honesty of data will be developed in our future projects. We will also investigate ciphertext_retrieval via preexisting data sharing mechanisms to fully use the mobile cloud.

5. References

Gentry C, Halevi S. Implementing gentry's fully-homomorphic encryption scheme. in: Advances in Cryptology–EUROCRYPT 2011. Berlin, Heidelberg: Springer press, pp. 129-148, 2011.

Brakerski Z, Vaikuntanathan V. Efficient fully homomorphic encryption from (standard) LWE. in: Proceeding of IEEE Symposium on Foundations of Computer Science. California, USA: IEEE press, pp. 97-106, Oct. 2011.

Qihua Wang, HongxiaJin. "Data leakage mitigation for discertionary access control in collaboration clouds". the 16th ACM Symposium on Access Control Models and Technologies (SACMAT), pp.103-122, Jun. 2011.

Adam Skillen and Mohammad Mannan. On Implementing Deniable Storage Encryption for Mobile Devices. the 20th Annual Network and Distributed System Security Symposium (NDSS), Feb. 2013.

Wang W, Li Z, Owens R, et al. Secure and efficient access to outsourced data. in: Proceedings of the 2009 ACM workshop on Cloud computing security. Chicago, USA: ACM pp. 55-66, 2009.

Google developers: Safe Browsing API. https://developers.google.com/ safe-browsing/, 2012.

Java Programming- SIA Publishers & Distributors Pvt. Ltd.

Software Engineering: A Practitioner's Approach.

Software Testing, 3rd edition, P.C. Jorgensen, Aurbach Publications.