ISSN: 2321-2152

IJMECE

International Journal of modern electronics and communication engineering

E-Mail
editor.ijmece@gmail.com
editor@ijmece.com

www.ijmece.com

ISSN2321-2152

www.ijmece .com

Vol 12, Issue 2, 2024

SECURE DEDUPLICATED CLOUD STORAGE WITH ENCRYPTED TWO-PARTY INTERACTIONS IN CYBER-PHYSICAL SYSTEM

[1]Ishrath Nousheen, [2]Vanguru Dharani Reddy, [3]P Kiranmai, [4]K Nikitha

[1]Assistant professor in Department of Information Technology Bhoj Reddy Engineering College forWomen

[2,3,4] UG Scholars in Department of Information Technology Bhoj Reddy Engineering College for Women

Abstract

Cloud envisioned Cyber-Physical Systems (CCPS) is a practical technology that relies on the interaction among cyber elements like mobile users to transfer data in cloud computing. In CCPS, cloud storage applies data deduplication techniques aiming to save data storage and bandwidth for real-time services. In this infrastructure, data deduplication eliminates duplicate data to increase the performance of the CCPS application. However, it incurs security threats and privacy risks. For example, the encryption from independent users with different keys is not compatible with data deduplication. In this area, several types of research have been done. Nevertheless, they are suffering from a lack of security, high performance, and applicability. Motivated by this, we propose a message Lock Encryption with neVer-decrypt homomorphic EncRyption (LEVER) protocol between the uploading CCPS user and cloud storage to reconcile the encryption and data deduplication. Interestingly, LEVER is the first brute-force resilient encrypted deduplication with only cryptographic two-party interactions. We perform several numerical analysis of LEVER and confirm that it provides high performance and practicality compared to the literature.

I INTRODUCTION

The amount of data to be stored by cloud storage systems increases extremely fast. It is thus of utmost importance for Cloud Storage Providers (CSPs) to dramatically reduce the cost to store all the created data. A promising approach to achieve this objective is through data deduplication.

Put simply, data deduplication keeps a single copy of repeated data. When a client wishes to store some piece of data, and if a copy of this data has already been saved in the storage system, then solely a reference to this existing copy is stored at the storage server. No duplicate is created.

There are diverse forms of data deduplication. It can be done by a client solely on the data he/she has previously stored in the system, a technique commonly called intra-user deduplication, or it can be achieved by considering the data previously stored by all the clients.

In this case it is designated as inter-user deduplication. Data deduplication also improves users experience by saving network bandwidth and reducing backup time when the clients perform the deduplication before uploading data to the storage server. This form of deduplication is termed as client-side deduplication, and when it is handled by the storage server it is called server-side deduplication. Due to its straightforward economic advantages, data deduplication is gaining popularity in both commercial and research storage systems.

However, several works have recently revealed important security issues leading to information leakage to malicious clients. These security concerns arise especially in systems performing an inter-user and client-side deduplication which is unfortunately the kind of deduplication that provides the best savings in terms of network bandwidth and storage space

Deduplication and confidentiality: the issues

Hereafter, we summarize in three categories the most common types of attacks described in the literature. Manipulation of data identifiers: A common technique to deduplicate data is by hashing the content of the data and using this unique hashed value as the identifier of the data.

Then the client sends this identifier to the storage server to determine whether such an identifier already exists in the system. An attacker can easily exploit this technique to perform various types of attacks. An example of attack based on data identifier manipulation is the CDN attack that turns a cloud storage system into a Content Delivery Network (CDN). Suppose that Bob wants to share a file F with Alice. Bob uploads F to a cloud storage system and sends F identifier to Alice. When Alice receives it, she tries to upload F to the same cloud storage system by sending F identifier.

The storage system will detect that this identifier already exists. Consequently, solely a reference meaning that Alice also owns file F will be stored in the system which is actually wrong. At this point, when Alice wants to download F, she just needs to request it from the cloud storage system. Another example of attacks based on data identifier manipulation is the one called targeted collision in which, a malicious client uploads a piece of data (e.g, a file) that does not correspond to the claimed identifier. Suppose that Bob wants to cheat Alice. If no control is made by the cloud storage system, Bob can upload a file F1 with the identifier of a file F2. Then, if Alice wishes to upload F2 with its real identifier, the system will detect the existence of F2 identifier in the system and will not store F2. Rather, the system will store a reference meaning that Alice also owns file F1 which is the file corresponding to the identifier of F2 in the system. Later, when Alice will request the file corresponding to the identifier of F2, the system will send F1 to Alice.

ISSN2321-2152

www.ijmece .com

Vol 12, Issue 2, 2024

Network traffic observation: Monitoring the network traffic on the client side gives an attacker the capability to determine whether an inter-user deduplication has been applied on a given piece of data. Such an attacker can exploit this knowledge to identify data items stored in a cloud storage system by other users or even learn the content of these data items.

For instance, to determine whether a file F is stored in a system, the attacker just tries to upload F and observes the network traffic from his/her device toward the storage server. If F is uploaded, it means that F does not exist in the storage system. The attacker is not even obliged to upload the complete file. He/she can prematurely abort the upload, so that the attack can be repeated later. On the other hand, if the file is not uploaded, the attacker can conclude that the file already exists in the system. This fact makes the storage system vulnerable to brute force attacks on file contents.

For instance, to determine which copies of a movie exist on a specific cloud storage system, an attacker can upload each format or version of such a movie, and show that those that are not uploaded already exist in the storage system. Backup time observation: Detecting an inter-user deduplication by observing the backup duration gives an attacker the ability to perform the same attacks as the ones done with network traffic observations. However, observing the duration of a backup operation is less accurate than a network traffic monitoring as it depends on the size of the file and the state of the network. For small files, observation may not be accurate, while for larger ones, it gains in accuracy. There have been lots of efforts devoted to deal with these issues and provide a secure and efficient data deduplication. However, we are not aware of any single solution that is capable of addressing, at the same time, the three types of attacks that malicious users can attempt on the deduplication system. This is the objective of this paper.

II LITERATURE SURVEY

"Understanding data deduplication ratios"

Data deduplication and other methods of reducing storage consumption play a vital role in affordably managing today's explosive growth of data. Optimizing the use of storage is part of a broader strategy to provide an efficient information infrastructure that is responsive to dynamic business requirements. This paper will explore the significance of deduplication ratios related to specific capacity optimization techniques within the context of information lifecycle management.

"A study of practical deduplication"

We found that whole-file deduplication achieves about three quarters of the space savings of the most aggres- sive block-level deduplication for

storage of live file systems, and 87% of the savings for backup images. We also studied file fragmentation finding that it is not prevalent, and updated prior file system metadata studies, finding that the distribution of file sizes continues to skew toward very large unstructured files.

"Proofs of ownership in remote storage systems"

In this work we identify attacks that exploit client-side deduplication, allowing an attacker to gain access to arbitrary-size files of other users based on a very small hash signatures of these files. More specifically, an attacker who knows the hash signature of a file can convince the storage service that it owns that file, hence the server lets the attacker download the entire file. (In parallel to our work, a subset of these attacks were recently introduced in the wild with respect to the Dropbox file synchronization service.)

"Side channels in cloud services: Deduplication in cloud storage,"

Cloud storage refers to scalable and elastic storage capabilities delivered as a service using Internet technologies with elastic provisioning and usebased pricing that doesn't penalize users for changing their storage consumption without notice.

"Dark clouds on the horizon: using cloud storage as attack vector and online slack space,"

Within this paper we give an overview of existing file storage services and examine Dropbox, an advanced file storage solution, in depth. We analyze the Dropbox client software as well as its transmission protocol, show weaknesses and outline possible attack vectors against users. Based on our results we show that Dropbox is used to store copyright-protected files from a popular file sharing network. Furthermore, Dropbox can be exploited to hide files in the cloud with unlimited stor-age capacity.

III EXISTING SYSTEM

Data deduplication keeps a single copy of repeated data. When a client wishes to store some piece of data, and if a copy of this data has already been saved in the storage system, then solely a reference to this existing copy is stored at the storage server. No duplicate is created. There are diverse forms of data deduplication. It can be done by a client solely on the data he/she has previously stored in the system, a technique commonly called intra-user deduplication, or it can be achieved by taking into account the data previously stored by all the clients. In this case it is designated as inter-user deduplication. Data deduplication also improves users experience by saving network bandwidth and reducing backup

time when the clients perform the deduplication before uploading data to the storage server. This form of deduplication is termed as client-side deduplication, and when it is handled by the storage server it is called server-side deduplication. Due to its straightforward economical advantages, data deduplication is gaining popularity in both commercial and research storage systems. several works have recently revealed important security issues leading to information leakage to malicious clients. These security concerns arise especially in systems performing an inter-user and client-side deduplication which is unfortunately the kind of deduplication that provides the best savings in terms of network bandwidth and storage space. In this part, we explain the solutions applied to encrypted data deduplication. Convergent Encryption (CE) and its generalization, MLE are the easiest methods to address the privacy issues without compromising the deduplication effectiveness. Specifically, the user calculates and uploads Eh(f)(f), where Ek(·) stands for the symmetric encryption with key k. Since the users with f are all can derive the same h(f) and Eh(f)(f), the deduplication still takes place on Eh(f)(f). The follow-up studies further validate the MLE protocol to preserve message correlation and parameter dependency. This method is weak against the bruteforce attack, particularly in the case of predictable data. The brute-force mainly incurs when we face low min-entropy characteristics, and the keyspace in CE is identical to the plaintext space.

DupLESSis a suitable solution to defend against the brute-force attack in cloud storage. In the DupLESS, they introduce an additional key server (KS) to generate the key. In particular, the user u with the aid of KS calculate a content-dependent key kf for the file f such that no one can drive the key except the user. Later on, deduplication Ekf (f) can be done using kf . Based on the trusted server, many other approaches are also proposed. Encrypt-with-Signature (EwS) is another solution using KS for calculating Ekf (f). Both of are requiring to communicate with the trusted server to calculate deduplication Ekf (f). It increases the privacy issue in the network. Liu et al.

client-as-a-key-server (CaS) framework, where users are potential key servers, to deduplicate encrypted data. The solutions mentioned above suffer from performance degradation, because of heavyweight cryptographic primitives (e.g., oblivious pseudorandom function , homomorphic encryption and PAKE are used. Also, they are not practically useful because the trusted server does not have a reasonable business justification.

One may have the hybrid use of various solutions to simultaneously develop the deduplicated storage resistant to security and privacy issues. Nevertheless, the particular design of certain encrypted deduplication hinders the joint use with

the current brute-force defences. On the other hand, even if the hybrid construction is possible, as each solution for the individual issue has the performance and security weaknesses, the deduplicated storage system based on these building blocks also inherits the performance and security/privacy problems. There are several protocols in the literature targeting the encrypted data deduplication problems in cloud storage. However, the current encrypted data deduplication either suffer from a brute-force attack or rely on computation-intensive operations and independent key servers. The state-of-the-art method [11] claims to eliminate the need for the independent server but uses cloud users as key servers. The method in [11] also uses two heavyweight cryptographic techniques, password authenticated key exchange (PAKE) and homomorphic encryption (HE), leading to the computation inefficiency.

IV PROBLEM STAEMENT

we aim to tackle the brute-force attack in cloud storage, applying in CPS. To fill the above gaps and refer to the paper's contribution, some questions arise: i) Is it possible to design an efficient encrypted data deduplication algorithm in CPS? ii) Can we assure that the proposed algorithm could bring data privacy, high/practical performance compared to the literature? And, iii) How can the encrypted method support two-party interaction between the uploader and the cloud server?

V PROPOSED SYSTEM

Here, we present our message Lock Encryption with neVerdecrypt homomorphic EncRyption (LEVER). We divide the encrypted data deduplication into three phases; the user derives the chunk key for the chunk to be uploaded in the first phase, transforms the chunk to be uploaded into an encrypted form in the second phase, and then runs the ordinary data deduplication protocol in the third phase. The design challenge of encrypted data deduplication is that a high min-entropy key can only encrypt the chunk. However, it is still difficult for different users with the same f to calculate the standard high min-entropy key.

presents the proposed cross-user cloud envision cyber-physical system (CCCPS) architecture. As we can see, user 1 from CPS 1 and user 2 from CPS 2 upload the same file f into the CDD (see the continuous arrows in Fig. 1a). Meanwhile, an adversary aims to gain information about the file f and get access to the encrypted data to change its integrity in CDD. We require to impose our cloud server to preserve the data's security and privacy while satisfying the data deduplication technique in the system.

Our work takes place in the context of an Internet Service Provider (ISP) providing also the storage

system2. That is to say, the ISP which is also the CSP has strong economical reasons to (i) save storage space and network bandwidth as it masters all the network and storage infrastructure, and (ii) provide a secure storage service to its consumers. We propose a deduplication solution to build a storage system that positively answers the following three questions:

1) Can we guarantee that the identifier used to trigger deduplication corresponds to the claimed data item?

2) Can we determine that a client actually owns the data item corresponding to the identifier issued to the storage system?

3) Can we make the inter-user deduplication transparent (i.e. unnoticeable even in the case of backup time or network traffic observation) to clients and still provide network bandwidth savings?

Our deduplication scheme is simple and robust against the aforementioned attacks while remaining efficient in terms of storage space and bandwidth savings for both clients and the CSP. We consider data deduplication at a file level granularity but our solution can be extended to the block level.

Specifically our approach is a two-phase deduplication that leverages and combines both intra- and inter-user deduplication techniques by introducing deduplication proxies (DPs) between the clients and the storage server (SS). Communications from clients go through these DPs to reach the SS which allows splitting the deduplication process.

VI IMPLEMENTATION

Client (C): Any authenticated user accessing the storage system.

***Storage Server (SS)*:** A server in charge of storing and serving clients files. The storage server also maintains an index of all the files stored in the storage system and their owners.

Deduplication Proxy (DP): A server associated with a given number of clients. Clients communicate with the SS via their associated deduplication proxy. A deduplication proxy is involved in both the intra-user and the inter-user deduplication.

The contributions of this paper are summarized as follows. • An obstacle in designing encrypted data deduplication is how the cloud can find that two distinct encrypted chunks are from the same content. Thus, our first contribution is developing a message Lock Encryption with neVer-decrypt homomorphic EncRyption (LEVER) by taking advantage of the property of homomorphic encryption without further decryption and resiliency against brute-force by external attackers.

• In LEVER, only the uploader and cloud participate in the uploading process, in contrast to most current solutions in need of the third party's participation in CPS.

• LEVER can work transparently with any current cloud storage linked with the CPS users without the need for cloud storage provider's engineering work at the backend. • We validate LEVER in terms of communication and deduplication costs using two datasets, namely Enron [12] and Oxford [13].

• Lastly, in addition to the analysis and numerical simulations, we have a LEVER prototype to demonstrate the practicality. A comparison among different systems is shown in Table I. The python implementation of LEVER is available in [14]

We have implemented two storage system prototypes to compare the performance overhead of our proposition with respect to a classic storage system with no data encryption. Specifically, we have developed a classic storage system with a client and storage server software modules, and one implementing our proposition with a client, a deduplication proxy and storage server software modules. All these software modules are implemented in python 2.7.6 and access the pycrypto library for the cryptographic operations. We use the SHA256 algorithm as the hash function, RSA1024 for asymmetric encryption operations, and AES for the symmetric encryption operations with keys that are 256 bits long. The storage servers use the MongoDB3 database to store the meta-data of the stored files as well as files owners. The software modules are executed on three different virtual machines (VMs) running on Ubuntu 12.04.4 LTS with 1GB of memory and an AMD Opteron(TM) octa-core Processor 6220@3GHz. The network topology is as follows: the VM executing the DP software is located on the network path between the VM running the different clients modules and the one executing the different SSs modules.

The average duration values of the these operations. We observe that the intra-user deduplication in our scheme consumes more communication resources than an inter-user deduplication in a classic scheme due to the use of a DP (around 18ms of overhead for a file of 64MB length). However most of the overhead comes from the encryption key creation, the encryption key encryption4 and the file encryption which depends on the file size. We also observe that operations involving the interuser deduplication in our scheme have higher durations than the ones involving an intra-user deduplication. This is due to (i) the overhead of communications when applying the interuser deduplication because the file has to be uploaded to the DP and then a reference is uploaded to the the SS and (ii) the added delay by the DP to make unnoticeable the inter-user deduplication to client. Actually in our scheme, a put operation

involving an inter-user deduplication has a similar duration than a file uploads from the client to the SS.

VII CONCLUSION

In conclusion, the successful implementation of LEVER, a secure deduplicated cloud storage system with encrypted two-party interactions in cyber-physical systems, relies heavily on the integration of various libraries and tools. These libraries provide essential functionality across encryption, deduplication, secure computation, access control, user interface development, logging, monitoring, testing, and more. By leveraging these libraries effectively, LEVER can achieve its objectives of ensuring data privacy, security, efficiency, and reliability in the context of cyber-physical systems. As technology advances and new challenges emerge, the continuous improvement and adaptation of these libraries will be essential to maintaining LEVER's effectiveness and relevance in securing sensitive data in increasingly interconnected environments.

REFERENCES

[1] S. Halevi, D. Harnik, B. Pinkas, and A. Shulman-Peleg, "Proofs of ownership in remote storage systems," in Parallel to our work, a subset of these attacks were recently introduced in the wild with respect to the Dropbox file synchronization service, [Online]. Available: [URL]

. [2] D. Harnik, B. Pinkas, and A. Shulman-Peleg, "Side channels in cloud services: Deduplication in cloud storage," Cloud storage refers to scalable and elastic storage capabilities delivered as a service using Internet technologies with elastic provisioning and use-based pricing that doesn't penalize users for changing their storage consumption without notice, [Online]. Available: [URL].

[3] PD. Harnik, B. Pinkas, and A. Shulman-Peleg, "Side channels in cloud services: Deduplication in cloud storage," Comput. Secur., vol. 108, Sep. 2022, Art. no. 102328, [Online]. Available: [URL].

[4]A. Basit, M. Zafar, X. Liu, A. R. Javed, Z. Jalil, and K. Kifayat, ‘‘A comprehensive survey of AI-enabled phishing attacks detection techniques,’’ Telecommun. Syst., vol. 76, no. 1, pp. 139–154, Jan. 2022