**Attribute-Based Storage Supporting Secure DE-duplication of Encrypted Data in Cloud**

Harshal Ranshur 1, Swapnil Potdar 2 , Rohan Salve3, Shubham Sapkal4, Prof. D.P.Jain5

U.G. Student, Department of Computer Engineering, Adsul’s Technical Campus, Chas, Ahmednagar, India1

Associate Professor, Department of Computer Engineering, Adsul’s Technical Campus, Chas, Ahmednagar, India5

**Abstract**- In SaaS, tenants usually customize multiple duplications and put them onto different data nodes of service provider to ensure the data reliability in cloud. However un-trust service providers may tamper, delete or forge tenants’ data. On the other hands, since all the duplication look the same, the un-trusted service provider could store only one data copy rather than the required number to cheat tenants. So the tenants need to ensure that the service provider process their data duplication honestly which is not being tampered or partially deleted. This paper presents a tenant du- plication integrity protection mechanisms TDIC (Tenant-oriented Duplication Integrity Checking Scheme). TDIC is tuples sample-based challenge-response model and constructs new tenant duplication authentication structure (TDAS) based on tenant physical tuples. Combing TDAS with homomorphism label,

locally, tenant should be equipped with certain security means so that they can verify the correctness of the remote data even without the existence of local copies. For secure data storage, existing researches major in the scenario that users own the independent data storage mode rather than shared one. But in SaaS, most multitenant applications take the single instance multi-tenancy strategy to make full use of the resource, which leads to the situation that multiple tenants’ data is stored into one data table such as universal table. For this scenario, traditional approach that needed to get file partitions could not work well on the shared physical storage mode in SaaS. Because there may be several tenants’ data in one data block partition, and this partition not only breaks the data isolation requirement of different tenants but also increases the complexity of integrity verification.

**A. Motivation**

Data deduplication is a process that eliminates excessive copies of data and significantly decreases storage capacity requirements. Deduplication can be run as an inline process as the data is being written into the storage system and/or as a background process to eliminate duplicates after the data is written to disk.

***Keywords*** *– Data Deduplication, Encryption Technique, Decryption Technique, Cloud Storage.*

**I. INTRODUCTION**

In SaaS, based on the single instance multi-tenancy mode, tenants’ data is stored and processed at the remote service providers. Meanwhile, for data reliability in cloud tenants may customize multiple duplications and pay for use. However, ser- vice providers maybe un-trust and they may tamper, delete or forge tenants’ data. On the other hands, plain-text data duplication is vulnerable to conspired attacks of the service providers, since all the replicas look the same, the un-trusted service provider may store only one data copy rather than the customized number to cheat tenants. For the upper problems, tenants need to ensure that the service provider process their data duplication honestly which is not being tampered or partially deleted. At the same time, because tenants no longer possess their data

**B. Problem Definition**

The problem is to determine how to design secure deduplication systems with higher reliability in cloud computing. Hence it is been proposed in the distributed cloud storage servers into deduplication systems to provide better fault tolerance. To protect data confidentiality, the secret sharing

technique is utilized, which is also compatible with the distributed storage systems.

**III. SYSTEM DESIGN**

Paper 2: DupLESS: Server-Aided Encryption for Deduplicated Storage

Author: Mihir Bellare,Sriram Keelveedhi,Thomas Ristenpart

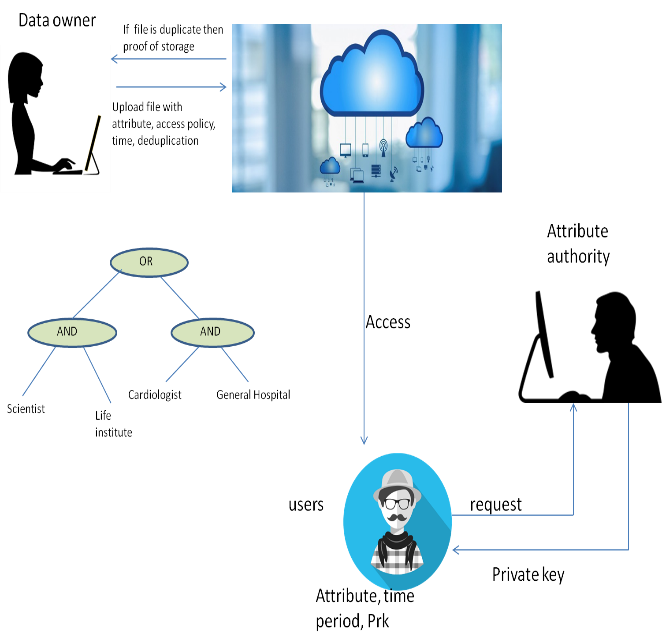
Abstract: Cloud storage service providers such as Dropbox, Mozy, and others perform deduplication to save space by only storing one copy of each file uploaded. Should clients conventionally encrypt their files, however, savings are lost. Message-locked encryption (the most prominent manifestation of which is convergent encryption) resolves this tension. However, it is inherently subject to brute-force attacks that can recover files falling into a known set. We propose an architecture that provides secure deduplicated storage resisting brute-force attacks, and realize it in a system called DupLESS. In DupLESS, clients encrypt under message-based keys obtained from a key-server via an oblivious PRF protocol. It enables clients to store encrypted data with an existing service, have the service perform deduplication on their behalf, and yet achieves strong confidentiality guarantees. We show that encryption for deduplicated storage can achieve performance and space savings close to that of using the storage service with plaintext data.

**II. RELATED WORK**

Paper 1: The Knowledge Complexity of Interactive Proof-Systems

Authors: Shafi Goldwasse,Silvio Micali,Charles Rackoff

**Abstract:** In the first part of the paper, we introduce a new theorem-proving procedure, that is a new efficient method of communicating a proof: Any such method implies, directly or indirectly, a definition of proof. Our “proofs” arc probabilistic in nature. On input an II-bits long statement, we may erroneously be convinced of its correctness with very small probability, say, we propose to classify languages according to the amount of additional knowledge that must be released for proving membership in them. Of particular interest is the case where this additional knowledge is essentially 0 and we show that is possible to interactively prove that a number is quadratic non residue mod m releasing 0 additional knowledge. This is surprising as no efficient algorithm for deciding quadratic mod m is known when m’s factorization is not given. Moreover, all known NP proofs for this problem exhibit the prime factorization of tn. This indicates that adding interaction to the proving process, may decrease the amount of knowledge that must be communicated in order to prove a theorem.



**IV. Module**

* Admin
* In this module, the admin has to log in by using valid user name and pass- word. After login successful he can do some operations, such as View All Users and Authorize, View All E-Commerce Website and Authorize, View All Products and Reviews, View All Products Early Reviews, View All Keyword Search Details, View All Products Search Ratio, View All Keyword Search Results, View All Product Review Rank Results.
* View and Authorize Users
* In this module, the admin can view the list of users who all registered.

In this, the admin can view the user’s details such as, user name, email, address and admin authorize the users.

* View Charts Results
* View All Products Search Ratio, View All Keyword Search Results, View All Product Review Rank Results.
* Ecommerce User
* In this module, there are n numbers of users are present. User should register before doing any operations. Once user registers, their details will be stored to the database. After registration successful, he has to login by using authorized user name and password Once Login is

successful user will do some operations like Add Products, View All Products with reviews, View All Early Product’s reviews, View All Purchased Transactions.

* End User
* In this module, there are n numbers of users are present. User should register before doing any operations. Once user registers, their details will best or to the database. After registration successful, he has to login by using authorized user name and password. Once Login is successful user will do some operations like Manage Account, Search Products by keyword and Purchase, View Your Search Transactions, View.

I

**VI. References**

1. Bitcasa, inifinite storage. http://www.bitcasa.com/.

2. Ciphertite data backup. http://www.ciphertite. com/.

3. Dropbox, a file-storage and sharing service. http://www. dropbox.com/.

4. Dupless source code. http://cseweb.ucsd.edu/ users/skeelvee/dupless.

5. The Flud backup system. http://flud.org/wiki/ Architecture.

6. GNUnet, a framework for secure peer-to-peer networking. https://gnunet.org/.

7. Google Drive. http://drive.google.com.

8. ADYA, A., BOLOSKY, W., CASTRO, M., CERMAK, G., CHAIKEN, R., DOUCEUR, J., HOWELL, J., LORCH, J., THEIMER, M., AND WATTENHOFER, R. Farsite: Federated, available, and reliable storage for an incompletely trusted environment. ACM SIGOPS Operating Sys- tems Review 36, SI (2002), 1–14.

9. AMAZON. Amazon Elastic Block Store (EBS). http:// aws.amazon.com/ebs.

10. AMAZON. Amazon Elastic Compute Cloud (EC2). http:// aws.amazon.com/ec2.

11. AMAZON. Amazon Simple Storage Service (Amazon S3). http://aws.amazon.com/s3.

12. ANDERSON, P., AND ZHANG, L. Fast and secure laptop backups with encrypted de-duplication. In Proc. of USENIX LISA (2010).

**V. Conclusion**

We studied the problem of providing secure outsourced storage that both sup- ports deduplication and resists brute-force attacks. We design a system, DupLESS, that combines a CE-type base MLE scheme with the ability to obtain message-derived keys with the help of a key server (KS) shared amongst a group of clients. The clients interact with the KS by a protocol for oblivious PRFs, ensuring that the KS can cryptographically mix in secret material to the per-message keys while learning nothing about files stored by clients. These mechanisms ensure that DupLESS provides strong security against external attacks which compromise the SS and communication channels (nothing is leaked beyond file lengths, equality, and access patterns), and that the security of DupLESS gracefully degrades in the face of comprised systems. Should a client be compromised, learning the plaintext underlying another client’s ciphertext requires mounting an online brute force attacks (which can be slowed by a rate-limited KS). Should the KS be compromised, the attacker must still attempt an offline brute-force attack, matching the guarantees of traditional MLE schemes. The substantial increase in security comes at a modest price in terms of performance, and a small increase in storage requirements relative to the base system. The low performance overhead results in part from optimizing the client-to-KS OPRF protocol, and also from ensuring DupLESS uses a low number of interactions with the SS. We show that DupLESS is easy to deploy: it can work transparently on top of any SS implementing a simple storage interface, as shown by our prototype for Dropbox and Google Drive.