DYNAMIC AGGREGATE KEY GENERATOR FOR CLOUD COMPUTING

Deepika Patil, Ashish Modak, Swagnik Das, Arshi Akab

Abstract— Cloud storage enables users to remotely store their data and enjoy the on-demand high quality cloud applications without the burden of local hardware and software management. Though the benefits are clear, such a service is also relinquishing users physical possession of their outsourced data, which inevitably poses new security risks towards the correctness of the data in cloud. In order to address this new problem and further achieve a secure and dependable cloud storage service, we propose in this paper a flexible distributed storage integrity auditing mechanism, utilizing the homomorphic token and distributed erasure-coded data. The proposed design allows users to audit the cloud storage with very lightweight communication and computation cost. The auditing result not only ensures strong cloud storage correctness guarantee, but also simultaneously achieves fast data error localization, i.e., the identification of misbehaving server. Considering the cloud data are dynamic in nature, the proposed design further supports secure and efficient dynamic operations on outsourced data, including block modification, deletion, and append. Analysis shows the proposed scheme is highly efficient and resilient against Byzantine failure, malicious data modification attack, and even server colluding attacks.

Index Terms— Cloud storage, data sharing, key-aggregate encryption, patient-controlled encryption

Manuscript received Nov, 2015.
Deepika Patil*, computer engineering, Savatribai Phule University, Pune / K.J.College of engineering/Pune, India,, Mobile No:8482854995*
Ashish Modak*, computer engineering, Savatribai Phule University, Pune / K.J.College of engineering/ Pune, India,, Mobile No:9763294780*
Swagnik Das*, computer engineering, Savatribai Phule University, Pune / K.J.College of engineering/ Pune, India,, Mobile No:8237523483*
Arshi Akab *computer engineering, Savatribai Phule University, Pune / K.J.College of engineering/Pune, India,, Mobile No:9764997867*

I. INTRODUCTION

A. Several trends are opening up the era of Cloud Computing, which is an Internet-based development and use of computer technology. The ever cheaper and more powerful processors, together with the software as a service (SaaS) computing architecture, are transforming data centers into pools of computing service on a huge scale. The increasing network bandwidth and reliable yet flexible network connections make it even possible that users can now subscribe high quality services from data and software that reside solely on remote data centers. Moving data into the cloud offers great convenience to users since they don't have to care about the complexities of direct hardware management. The pioneer of Cloud Computing vendors, Amazon Simple Storage Service (S3) and Amazon Elastic Compute Cloud (EC2) are both well known examples. While these internet-based online services do provide huge amounts of storage space and customizable computing resources, this computing platform shift, however, is eliminating the responsibility of local machines for data maintenance at the same time. As a result, users are at the mercy of their cloud service providers for the availability and integrity of their data . On the one hand, although the cloud infrastructures are much more powerful and reliable than personal computing devices, broad range of both internal and external threats for data integrity still exist. Examples of outages and data loss incidents of noteworthy cloud storage services appear from time to time . On the other hand, since users may not retain a local copy of outsourced data, there exist various incentives for cloud service providers (CSP) to behave unfaithfully towards the cloud users regarding the status of their outsourced data. For example, to increase the profit margin by reducing cost, it is possible for CSP to discard rarely accessed data without being detected in a timely fashion . Similarly, CSP may

ISSN: 2278 – 1323

even attempt to hide data loss incidents so as to maintain a reputation . Therefore, although outsourcing data into the cloud is economically attractive for the cost and complexity of long-term large-scale data storage, its lacking of offering strong assurance of data integrity and availability may impede its wide adoption by both enterprise and individual cloud users. In order to achieve the assurances of cloud data integrity and availability and enforce the quality of cloud storage service, efficient methods that enable on-demand data correctness verification on behalf of cloud users have to be designed. However, the fact that users no longer have physical possession of data in the cloud prohibits the direct adoption of traditional cryptographic primitives for the purpose of data integrity protection

Literature Survey:-

Definition of cloud: Cloud computing is a model to access information and services using existing technology and Internet infrastructures that allows establishing communication between clients and the server .We can imagine cloud computing as the ability of sharing computational resources among different users . In fact it is a platform or infrastructure that runs codes in a managerial and extensible model. Customers do not have the actual physical infrastructure and they just pay a subscription fee to the cloud provider and gain access to resources and infrastructure clouds with minimal effort or interaction with the service provider.

Now a days large organizations store their data on cloud. Due to which data leakage takes place. Organizations increasingly may be harmed by data being revealed to unauthorized parties. Such *data leaks* can cause harm in a variety of ways.

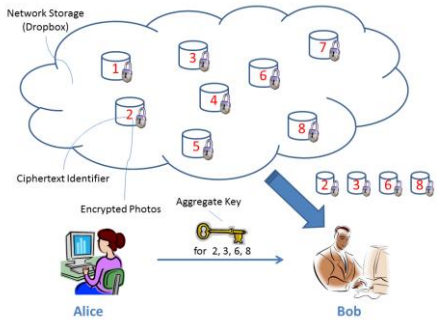
Cloud computing provide a way to store and share data as service over the internet. It enables several data sharing capabilities. There is an increase in the usage of cloud in several organizations to increase their data sharing efforts and to reduce their maintenance costs. Data sharing is an important functionality in cloud storage. So it is necessary that sharing of data between the users should be efficient and secure. It is been accepted that data encryption provides a better solution for this problem. This can be implemented in cloud system by several cryptosystem schemes that use encryption and decryption for data to be shared securely.

Data sharing is an important functionality in cloud storage. For example, bloggers can let their friends view a subset of their private pictures; an enterprise may grant her employees access to a portion of sensitive data. The challenging problem is how to effectively share encrypted data. Of course users can download the encrypted data from the storage, decrypt them, then send them to others for sharing, but it loses the value of cloud storage. Users should be able to delegate the access rights of the sharing data to others so that they can access these data from the server directly. However, finding an efficient and secure way to share partial data in cloud storage is not trivial. Below we will take Dropbox1 as an
example for illustration.

Assume that Alice puts all her private photos on Dropbox, and she does not want to expose her photos to everyone. Due to various data leakage possibility Alice cannot feel relieved by just relying on the privacy protection mechanisms provided by Dropbox, so she encrypts all the photos using her own keys before uploading. One day, Alice's friend, Bob, asks her to share the photos taken over all these years which Bob appeared in. Alice can then use the share function of Dropbox, but the problem now is how to delegate the decryption rights for these photos to Bob.

Therefore, the best solution for the above problem is that Alice encrypts files with distinct public-keys, but only sends Bob a single (constant-size) decryption key. Since the decryption key should be sent via a secure channel and kept secret.
This can be done in following way. Diagram shows it.

ISSN: 2278 – 1323



It also presents a possible architectural framework capable of powering the brokerage based cloud services.

The concepts of cloud bursting and cloud brokerage and discusses the open management and security issues associated with the two models.

Praposed system:

We propose executing server side operations in the encrypted domain, so that both the operands and the results are opaque to the server. The user can deliberately assume the risks of placing information on the Cloud, there exist activities in which law regulates data protection. There are proposals of additional security layers to protect the user from data mishandling by Cloud providers.

They enable customers to realize the full potential that a cloud provider has to offer by creating a trusted, governed and secured cloud management platform between the provider and the consumer of the cloud services. The main benefit of using the broker in this scenario is its ability to integrate more than one provider.

It covers the broad range of topics from service development to VM management and placement. From the point of view of the cloud service providers, extra capabilities need to be exposed to enable the enterprise to fulfil all orchestration requirements.

SLA-based requirements and the cloud broker then picks up the best match in terms of the functions as well as variables like pricing, SLA parameters and other non-functional requirements like compliance and certification capabilities.

Unlike most prior works for ensuring remote data integrity, the new scheme supports secure and efficient dynamic operations on data blocks, including: update, delete and append. Extensive security and performance analysis shows that the proposed scheme is highly efficient and resilient against Byzantine failure, malicious data modification attack, and even server colluding attacks. The straightforward and trivial way to support these operations is for user to download all the data from the cloud servers and re-compute the whole parity blocks as well as verification tokens. The user can always ask servers to send back blocks of the rows specified in the challenge and regenerate the correct blocks by erasure correction. We believe that data storage security in Cloud Computing, an area full of challenges and of paramount importance, is still in its infancy now, and many research problems are yet to be identified.

We believe is a model in which public verifiability is enforced. We can construct a scheme to achieve both public verifiability and storage correctness assurance of dynamic data.

ISSN: 2278 – 1323

II. MATH

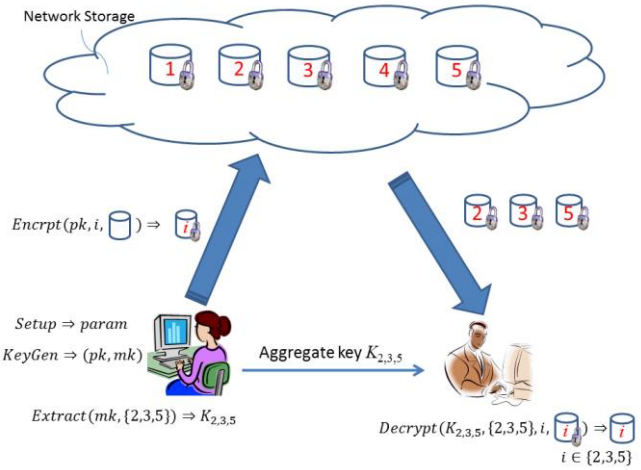
A key-aggregate encryption scheme consists of five polynomial-time algorithms as follows. The data owner establishes the public system parameter via Setup and generates a public/master-secret3 key pair via KeyGen. Messages can be encrypted via Encrypt by anyone who also decides what ciphertext class is associated with the plaintext message to be encrypted. The data owner can use the master-secret to generate an aggregate decryption key for a set of ciphertext classes via Extract. The generated keys can be passed to delegatees securely (via secure e-mails or secure devices) Finally, any user with an aggregate key can decrypt any ciphertext provided that the ciphertext's class is contained in the aggregate key via Decrypt4.

• Setup(1λ, n): executed by the data owner to setup an account on an *untrusted* server. On input a security
level parameter 1λ and the number of ciphertext classes n (i.e., class index should be an integer bounded by 1 and n), it outputs the public system parameter param, which is omitted from the input of the other algorithms for brevity.

• KeyGen: executed by the data owner to randomly generate a public/master-secret key pair (pk, msk).
• Encrypt(pk, i,m): executed by anyone who wants to encrypt data. On input a public-key pk, an index i denoting the ciphertext class, and a message m, it outputs a ciphertext C. • Extract(msk, S): executed by the data owner for delegating the decrypting power for a certain set of ciphertext classes to a delegatee. On input the mastersecret key msk and a set S of indices corresponding to different classes, it outputs the aggregate key for set S denoted by KS.

• Decrypt(KS, S, i, C): executed by a delegatee who received an aggregate key KS generated by Extract. On input KS, the set S, an index i denoting the ciphertext class the ciphertext C belongs to, and C, it outputs the decrypted result m if $i \in S$. There are two functional requirements:
• Correctness For any integers λ and n, any set $S \subseteq\{1, \cdots, n\}$, any index $i \in S$ and any message m, Pr[Decrypt(KS, S, i, C) = m : param ← Setup(1λ, n), (pk, msk) ← KeyGen(), C ← Encrypt(pk, i,m), KS ← Extract(msk, S)] = 1.
• Compactness For any integers λ, n, any set S, any index $i \in S$ and any message m; param ← Setup(1λ, n), (pk, msk) ← KeyGen(), KS ←Extract(msk, S) and C ← Encrypt(pk, i,m); |KS| and |C| only depend on the security parameter λ but independent of the number of classes n.

A. *Figures and Tables*



	Decryption key size	**Ciphertext size**	**Encryption type**
Key assignment schemes most likely non-constant constant symmetric or public-key for a predefined hierarchy	most likely non-constant constant symmetric or public-key for a predefined hierarchy ((depends on the hierarchy)	Constant	Symmetric or public key
Symmetric-key encryption with Compact Key	Contant	Constant	Public key
IBE with Compact Key	constant	Non-constant	Symmetric key
Attribute-Based Encryption (e.g., [10])	Non constant	Constant	Symmetric key
KAC	constant	Constant	Symmetric key

sentence punctuation follows the brackets [2]. Multiple references [2], [3] are each numbered with separate brackets [1]–[3]. When citing a section in a book, please give the relevant page numbers [2]. In sentences, refer simply to the reference number, as in [3]. Do not use "Ref. [3]" or "reference [3]" except at the beginning of a sentence: "Reference [3] shows ... ." Number footnotes separately in superscripts (Insert | Footnote).[1] Place the actual footnote at the bottom of the column in which it is cited; do not put footnotes in the reference list (endnotes). Use letters for table footnotes (see Table I).

Please note that the references at the end of this document are in the preferred referencing style. Give all authors' names; do not use "*et al.*" unless there are six authors or more. Use a space after authors' initials. Papers that have not

ISSN: 2278 – 1323

been published should be cited as "unpublished" [4]. Papers that have been submitted for publication should be cited as "submitted for publication" [5]. Papers that have been accepted for publication, but not yet specified for an issue should be cited as "to be published" [6]. Please give affiliations and addresses for private communications [7].

B. Abbreviations and Acronyms

1. KAC-Key Aggregate Cryptosystem
2. IBE-Identity based Encryption
3. ABE-Attribute Based Encryption
4. CS-Cloud Server
5. CSP-Cloud Service Provider

III. CONCLUSION

Overall we produce an aggregate key Cryptosystem which produces effective constant size private key by means of derivations of different cipher text classes.Proposed approach proves more secure and efficient cryptographic scheme in which we have an effective derivation of secret key generation and key management for the outsourced Cloud data.

.

Acknowledgment

It is our foremost duty to express our deep sense of gratitude and respect to the guide **Prof.U.A. Bodke** for her uplifting tendency and inspiring us for taking up this project work successful.

We are also grateful to **Prof. D.C. Mehetre (**Head of Department of Computer Engineering) for providing all necessary facilities to carry out the project work and whose encouraging part has been a perpetual source of information.

We are highly indebted to **Dr. S, J. Wagh**for their guidance and constant supervision as well as for providing necessary information regarding the project & also for their support in completing the project.

REFERENCES

1.S. S. M. CHOW, Y. J. HE, L. C. K. HUI, AND S.-M. YIU, "SPICE - SIMPLE PRIVACY-PRESERVING IDENTITY-MANAGEMENT FOR CLOUD ENVIRONMENT," IN APPLIED CRYPTOGRAPHY AND NETWORK SECURITY – ACNS 2012, SER. LNCS,VOL. 7341. SPRINGER, 2012, PP. 526–543.

2.. HARDESTY, "SECURE COMPUTERS AREN'T SO SECURE," MIT PRESS,2009,HTTP://WWW.PHYSORG.COM/NEWS176107396.HTML.3.C. Wang, S. S. M. Chow, Q. Wang, K. Ren, and W. Lou, "Privacy- Preserving Public Auditing for Secure Cloud Storage," IEEE Trans. Computers, vol. 62, no. 2, pp. 362–375, 2013.

4.B. WANG, S. S. M. CHOW, M. LI, AND H. LI, "STORING SHARED DATA ON THE CLOUD VIA SECURITY-MEDIATOR," IN INTERNATIONAL CONFERENCEON DISTRIBUTED COMPUTING SYSTEMS - ICDCS 2013. IEEE, 2013.

Deepika Patil*, computer engineering, Savatribai Phule University, Pune / K.J.College of engineering/ Pune, India,, Mobile No:8482854995*

Ashish Modak*, computer engineering, Savatribai Phule University, Pune / K.J.College of engineering/ Pune, India,, Mobile No:9763294780*

Swagnik Das*, computer engineering, Savatribai Phule University, Pune / K.J.College of engineering/ Pune, India,, Mobile No:8237523483*

Arshi Akab *computer engineering, Savatribai Phule University, Pune / K.J.College of engineering/ Pune, India,, Mobile No:9764997867*

.

ISSN: 2278 – 1323