© 2015 IJSRSET | Volume 1 | Issue 4 | Print ISSN: 2395-1990 | Online ISSN : 2394-4099
Themed Section: Engineering and Technology

Aggregate key generation on cloud data storage using PKC Encryption & Decryption

K. Kishor Kumar[1], Gavini Aravind[2]
[1]Assistant Professor, Kakatiya University, Warangal, TS, India
[2]PG Scholor, KUCET, Kakatiya Univerisity, Warangal, TS, India

ABSTRACT

The novelty is that one can aggregate any set of secret keys and make them as compact as a single key, but encompassing the puissance of all the keys being aggregated. In other words, the secret key holder can relinquish a constant-size aggregate key for flexible culls of cipher text set in cloud storage, but the other encrypted files outside the set remain confidential. Data sharing is a paramount functionality in cloud storage. In this article, we show how to securely, efficiently, and flexibly share data with others in cloud storage. We describe incipient public-key cryptosystems which engender constant-size (single key) cipher texts such that efficient delegation of decryption rights for any set of cipher texts are possible. This compact aggregate key can be conveniently sent to security channels with very inhibited secure storage. We provide formal security analysis of our schemes in the standard model. We withal describe other application of our schemes.

Keywords : Cloud storage, Key Gen, Encrypt, Decrypt, Public-key cryptography (PKC).

I. INTRODUCTION

Cloud storage is nowadays very popular storage system. Cloud storage is storing of data off-site to the physical storage which is maintained by third party. Cloud storage is preserving of digital data in logical pool and physical storage spans multiple servers which are manage by third party. Third party is responsible for keeping data available and accessible and physical environment should be forfended and running at all time. In lieu of storing data to the hard drive or any other local storage, we preserve data to remote storage which is accessible from anywhere and anytime. It reduces efforts of carrying physical storage to everywhere. By utilizing cloud storage we can access information from any computer through internet which omitted circumscription of accessing information from same computer where it is stored. While considering data privacy, we cannot rely on traditional technique of authentication, because unexpected privilege escalation will expose all data. Solution is to encrypt data afore uploading to the server with user's own key. Data sharing is again consequential functionality of cloud storage, because utilizer can apportion data from anywhere and anytime to anyone. For example, organization may grant sanction to access part of sensitive data to their employees. But challenging task is that how to apportion encrypted data. Traditional way is utilizer can download the encrypted data from storage, decrypt that data and send it to apportion with others, but it loses the consequentiality of cloud storage. Cryptography technique can be applied in a two major ways- one is symmetric key encryption and other is asymmetric key encryption. In symmetric key encryption, same keys are utilized for encryption and decryption. By contrast, in asymmetric key encryption different keys are utilized, public key for encryption and private key for decryption. Utilizing asymmetric key encryption is more flexible for our approach. This can be illustrated by following example. Suppose Alice put all data on Box.com and she does not optate to expose her data to everyone. Due to data leakage possibilities she does not trust on privacy mechanism provided by Box.com, so she encrypt all data afore uploading to the server. If Bob ask her to apportion some data then Alice use share function of Box.com. But quandary now is that how to apportion encrypted data. There are two rigorous ways: 1. Alice encrypt data with single secret

key and apportion that secret key directly with the Bob. 2. Alice can encrypt data with distinct keys and send Bob corresponding keys to Bob via secure channel. In first approach, unwanted data withal get expose to the Bob, which is inadequate. In second approach, no. of keys is as many as no. of shared files, which may be hundred or thousand as well as transferring these keys require secure channel and storage space which can be expensive.

II. RELATED WORK

A. Existing Systems

Assume that **Alice** puts all her private files on Drop box (cloud), and she does not want to expose her files to everyone. Due to various data leakage possibility Alice cannot feel relieved by just relying on the privacy protection mechanisms provided by Drop box, so she encrypts all the files using her own keys before uploading. One day, **Alice's** friend, **Bob**, asks her to share the files taken over all these years which Bob appeared in. Alice can then use the share function of Drop box, but the problem now is how to delegate the decryption rights for these files to Bob. A possible option Alice can choose is to securely send Bob the secret keys involved. Naturally, there are two extreme ways for her under the traditional encryption paradigm Alice encrypts all files with a single encryption key and gives Bob the corresponding secret key directly.

Alice encrypts files with distinct keys and sends Bob the corresponding secret keys Obviously, the **first method** is inadequate since all un-chosen data may be also leaked to Bob. For the **second method**; there are practical concerns on efficiency. The number of such keys is as many as the number of the shared files, say, a thousand. Transferring these secret keys inherently requires a secure channel, and storing these keys requires rather expensive secure storage. The costs and complexities involved generally increase with the number of the decryption keys to be shared. In short, it is very heavy and costly to do that.

B. Proposed Systems

We solve this problem by introducing a special type of public-key encryption which we call key-aggregate cryptosystem (KAC). In KAC, users encrypt a message not only under a public-key, but also under an identifier of cipher text called class. That means the cipher texts are further categorized into different classes. The key owner holds a master-secret called master-secret key, which can be used to extract secret keys for different classes. More importantly, the extracted key have can be an aggregate key which is as compact as a secret key for a single class, but aggregates the power of many such keys, i.e., the decryption power for any subset of cipher text classes With our solution, Alice can simply send Bob a single aggregate key via a secure e-mail. Bob can download the encrypted files from Alice's Drop box space and then use this aggregate key to decrypt these encrypted files.

III. IMPLEMENTATION

Key Gen: 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
Algorithm:
Public-key cryptography, additionally kenned as asymmetric cryptography, is a class of cryptographic algorithms which requires two separate keys, one of which is secret (or private) and one of which is public. Albeit different, the two components of this key pair are mathematically linked. The public key is utilized to encrypt plaintext or to verify a digital signature; whereas the private key is utilized to decrypt ciphertext or to engender a digital signature. The term "asymmetric" stems from the utilization of different keys to perform these antithesis functions, each the inverse of the other – as contrasted with conventional ("symmetric") cryptography which relies on the same key to perform both.

Public-key algorithms are predicated on mathematical quandaries which currently admit no efficient solution that are innate in certain integer factorization, discrete logarithm, and elliptic curve relationships. It is

computationally facile for a utilizer to engender their own public and private key-pair and to utilize them for encryption and decryption. The vigor lies in the fact that it is "infeasible" (computationally infeasible) for an opportunely engendered private key to be tenacious from its corresponding public key. Thus the public key may be published without compromising security, whereas the private key must not be revealed to anyone not sanctioned to read messages or perform digital signatures. Public key algorithms, unlike symmetric key algorithms, do not require a secure initial exchange of one (or more) secret keys between the parties.

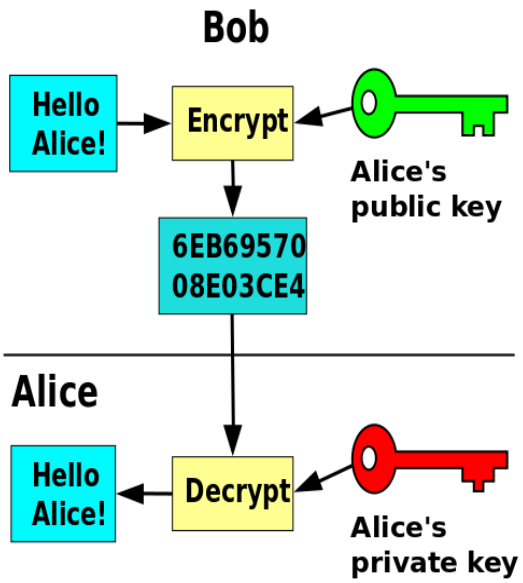

Figure 1. Block Diagram of Public Key cryptography Algorithm

Message authentication involves processing a message with a private key to engender a digital signature. Thereafter anyone can verify this signature by processing the signature value with the signer's corresponding public key and comparing that result with the message. Prosperity attests the message is unmodified since it was signed, and – surmising the signer's private key has remained secret to the signer – that the signer, and no one else, intentionally performed the signature operation. In practice, typically only a hash or digest of the message, and not the message itself, is encrypted as the signature.

Public-key algorithms are fundamental security ingredients in cryptosystems, applications and protocols. They underpin sundry Internet standards, such as Convey Layer Security (TLS), S/MIME, PGP, and GPG. Some public key algorithms provide key distribution and secrecy (e.g., Diffie–Hellman key exchange), some provide digital signatures (e.g., Digital Signature Algorithm), and some provide both (e.g., RSA).

Public-key cryptography finds application in, amongst others, the IT security discipline information security. Information security (IS) is concerned with all aspects of bulwarking electronic information assets against security threats.[1] Public-key cryptography is utilized as a method of assuring the confidentiality, authenticity and non-repudiability of electronic communications and data storage.

IV. EXPERIMENTAL RESULT

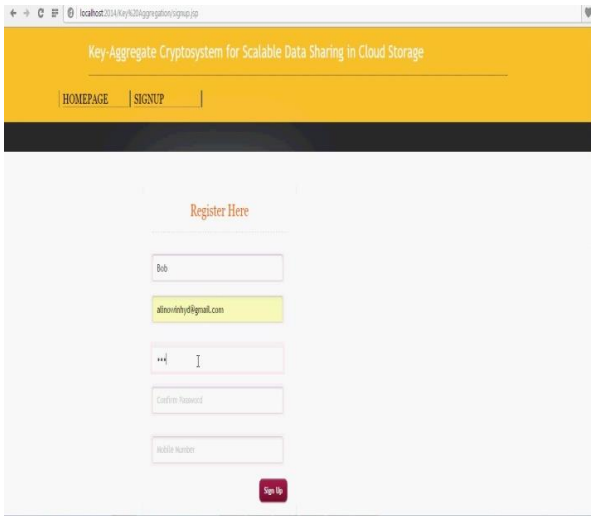

Figure 2 : Users Registration

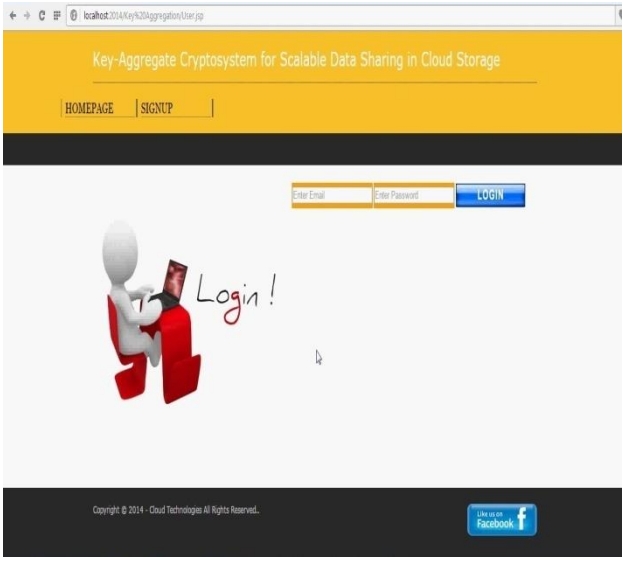

Figure 3 : Users Login

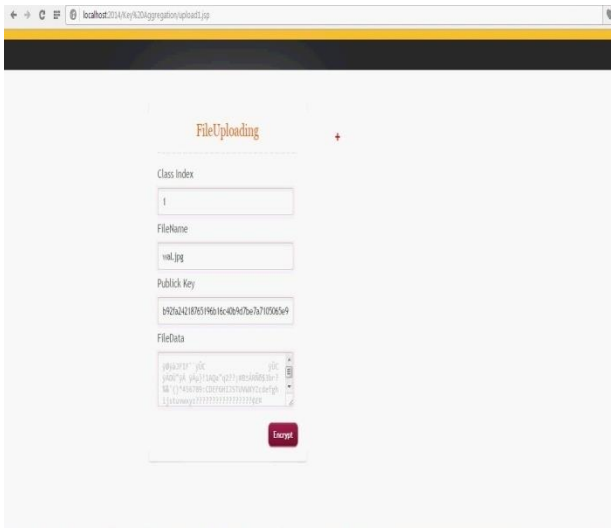

Figure 4 : File Upload

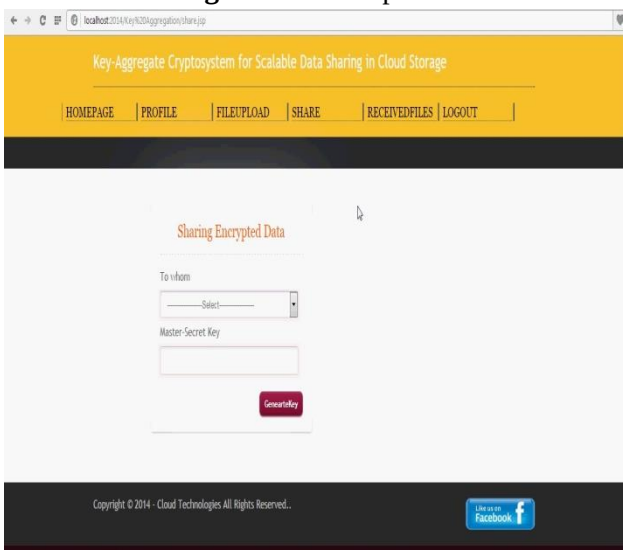

Figure 5 : Key Sharing Page

V. CONCLUSION

User's data privacy is a central question of cloud storage. Compress secret keys in public-key cryptosystems which fortification delegation of secret keys for different cipher text classes in cloud storage. No matter which one among the potency set of classes, the delegatee can always get an aggregate key of constant size. In cloud storage, the number of cipher texts customarily grows rapidly without any restrictions. So we have to reserve enough cipher text classes for the future extension. Otherwise, we require to expand the public-key. Albeit the parameter can be downloaded with cipher texts, it would be better if its size is independent of the maximum number of cipher text classes.

VI. REFERENCES

[1]. Cheng-Kang Chu ,Chow, S.S.M, Wen-Guey Tzeng, Jianying Zhou, and Robert H. Deng , Key-Aggregate Cryptosystem for Scalable Data Sharing in Cloud Storage, IEEE Transactions on Parallel and Distributed Systems. Volume: 25, Issue: 2. Year :2014.

[2]. J. Benaloh, M. Chase, E. Horvitz, and K. Lauter, Patient Controlled Encryption: Ensuring Privacy of Electronic Medical Records, in Proceedings of ACM Workshop on Cloud Computing Security (CCSW '09). ACM, 2009, pp. 103-114.

[3]. J. Benaloh, Key Compression and Its Application to Digital Fingerprinting, Microsoft Research, Tech. Rep., 2009.

[4]. B. Alomair and R. Poovendran, Information Theoretically Secure Encryption with Almost Free Authentication, J. UCS, vol. 15, no. 15, pp. 2937-2956, 2009.

[5]. D. Boneh and M. K. Franklin, Identity-Based Encryption from the Weil Pairing, in Proceedings of Advances in Cryptology - CRYPTO '01, ser. LNCS, vol. 2139. Springer, 2001, pp. 213-229.

[6]. A. Sahai and B. Waters, Fuzzy Identity-Based Encryption, in Proceedings of Advances in Cryptology - EUROCRYPT '05, ser. LNCS, vol. 3494. Springer, 2005, pp. 457-473.

[7]. S. S. M. Chow, Y. Dodis, Y. Rouselakis, and B. Waters, Practical Leakage-Resilient Identity-Based Encryption from Simple Assumptions, in ACM Conference on Computer and Communications Security, 2010, pp. 152-161.

[8]. F. Guo, Y. Mu, and Z. Chen, Identity-Based Encryption: How to Decrypt Multiple Ciphertexts Using a Single Decryption Key, in Proceedings of Pairing-Based Cryptography (Pairing '07), ser. LNCS, vol. 4575. Springer, 2007, pp. 392-406.

[9]. F. Guo, Y. Mu, Z. Chen, and L. Xu, Multi-Identity Single-Key Decryption without Random Oracles, in Proceedings of Information Security and Cryptology (Inscrypt '07), ser. LNCS, vol. 4990. Springer, 2007, pp. 384-398.

[10]. S. S. M. Chow, Y. Dodis, Y. Rouselakis, and B. Waters, Practical Leakage-Resilient Identity-Based Encryption from Simple Assumptions, in ACM Conference on Computer and Communications Security, 2010, pp. 152-161.