ISSN 2348-1196 (print)
International Journal of Computer Science and Information Technology Research ISSN 2348-120X (online)
Vol. 3, Issue 2, pp: (948-951), Month: April - June 2015, Available at: www.researchpublish.com

A Secure Data Sharing In Cloud Storage System Using Advanced Encryption Standard

[1] Maheswari .D, [2]Anitha.J, [3]Preethi.C, [4]Suganya Devi.B

[1,2,3] PG Scholar, [4] Assistant Professor, Department of Computer Science and Engineering,
Ranganathan Engineering College, Coimbatore, India

***Abstract*: Considering data privacy, a traditional way to ensure it is to rely on the server to enforce the access control after authentication, which means any unexpected privilege escalation will expose all data. In a shared-tenancy cloud computing environment, things become even worse. Data sharing is an important functionality in cloud storage. In this article, we show how to securely, efficiently, and flexibly share data with others in cloud storage. We describe new public-key cryptosystems which produce constant-size cipher texts such that efficient delegation of decryption rights for any set of cipher texts are possible. The novelty is that one can aggregate any set of secret keys and make them as compact as a single key, but encompassing the power of all the keys being aggregated. In other words, the secret key holder can release a constant-size aggregate key for flexible choices of cipher text set in cloud storage, but the other encrypted files outside the set remain confidential. This compact aggregate key can be conveniently sent to others or be stored in a smart card with very limited secure storage. We provide formal security analysis of our schemes in the standard model. We also describe other application of our schemes. In particular, our schemes give the first public-key patient-controlled encryption for flexible hierarchy, which was yet to be known.**

***Keywords:* Cloud Storage, Data sharing, KAC.**

I. INTRODUCTION

Cloud storage is gaining popularity recently. In enterprise settings, we see the rise in demand for data outsourcing, which assists in the strategic management of corporate data. It is also used as a core technology behind many online services for personal applications. Nowadays, it is easy to apply for free accounts for email, photo album, and file sharing and/or remote access, with storage size more than 25GB. Together with the current wireless technology, users can access almost all of their files and emails by a mobile phone in any corner of the world.

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.

Cloud computing is typically defined as a type of computing that relies on sharing computing resources rather than having local servers or personal devices to handle applications.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 partialdata in cloud storage is not trivial.

II. RELATED WORK

Using cloud storage, users can remotely store their data and enjoy the on-demand high-quality applications and services from a shared pool of configurable computing resources, without the burden of local data storage and maintenance. However, the fact that users no longer have physical possession of the outsourced data makes the data integrity protection in cloud computing a formidable task, especially for users with constrained computing resources. Moreover, users should be able to just use the cloud storage as if it is local, without worrying about the need to verify its integrity. Thus, enabling

public audit ability for cloud storage is of critical importance so that users can resort to a third-party auditor (TPA) to check the integrity of outsourced data and be worry free. To securely introduce an effective TPA, the auditing process should bring in no new vulnerabilities toward user data privacy, and introduce no additional online burden to user. In this paper, we propose a secure cloud storage system supporting privacy-preserving public auditing. We further extend our result to enable the TPA to perform audits for multiple users simultaneously and efficiently. Extensive security and performance analysis show the proposed schemes are provably secure and highly efficient. Our preliminary experiment conducted on Amazon EC2 instance further demonstrates the fast performance of the design.

Wireless Sensors have seen a lot of applications in our daily lives in the recent years. The market has been flooded with high end consumer electronics using wireless sensor technology. However, most of the current technologies require the sensor to be in the vicinity of the end-user application. There has been some study in the techniques for sensor provisioning and sharing for the large number of existing Wireless Sensor Networks. Virtualization of Wireless Sensor Networks (WSNs) is a step forward in exposing these WSNs to large user base from remote locations. However, there is still a huge gap in bringing together information available from heterogeneous, distributed resources of Wireless Sensor Networks to a non-localized user. In this work, we utilize IaaS paradigm of Cloud Computing in virtualization of sensor networks which gives the flexibility of handling heterogeneous systems. The system also enables a smart device user to access information generated by Wireless Sensors through the cloud via SaaS based design. This allows the system to take common computational tasks to be hosted as a service through the cloud. It frees the smart device user from running heavy applications for data processing and storing. Thus, system provides the smart device user a Cloud Enabled Wireless Sensor Network infrastructure. The system architecture provides the necessary features for it to be scalable and flexible, ensuring reliable sensor data transfer and processing through cloud infrastructure. We also present a small test bed implementation of the system.

Many group communications require a security infrastructure that ensures multiple levels-of access privilege fix group members. Access control in hierarchy is prevalent in multimedia applications, which consist of users that subscribe to different quality levels or different sets of data streams. In .this paper, we present a multi-group key management scheme that achieves such a hierarchical access control by employing an integrated key graph and by managing group kegs for all wars with various access privileges Compared with applying existing tree-based group keg management schemes directly to the hierarchical access control problem, the proposed scheme significantly reduces the communication, computation and storage overhead associated with key management and achieves better scalability when the number of access levels increases. In addition, the proposed key graph is suitable for both centralized and contributory environments.

III. MODULE DESCRIPTION

We propose several concrete KAC schemes with different security levels and extensions in our proposed scheme. All constructions can be proven secure in the standard model. A key-aggregate encryption scheme consists of five polynomial-time algorithms as follows.

- Setup
- KeyGen
- Encrypt
- Extract
- Decrypt

The data owner establishes the public system parameter via **Setup** and generates a public/master-secret3 key pair via **Key Gen**. Messages can be encrypted via **Encrypt** by anyone who also decides what cipher text 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 cipher text classes via **Extract**. The generated keys can be passed to delegates securely. Finally, any user with an aggregate key can decrypt any cipher text provided that the cipher text's class is contained in the aggregate key via **Decrypt**.

Authentication:

Authentication is the act of confirming the truth of an attribute of a datum or entity. This might involve confirming the identity of a person or software program, tracing the origins of an artifact, or ensuring that a product is what it's packaging and labeling claims to be. Authentication often involves verifying the validity of at least one form of

identification. We authenticate our system by using username and password. The username-password authentication flow can be used to authenticate when the consumer already has the user's credentials.

File Uploading:

In File uploading module each and every user upload their information's in to our system by using random key generation.

Encryption:

In this module, the information and data's shared by the user in the cloud is encrypted by using AES (Advanced Encryption Standard) algorithm. All of the information shared by every user is encrypted and stored in the cloud.

Key Sharing:

Key Sharing is an important module for sharing our information's to others by using secret key. The key is always hidden. When the user needs a key for a particular file, at a time they will retrieve the key from our system.

Decryption:

In this module, the information and data's shared by the user in the cloud is encrypted by using AES (Advanced Encryption Standard) algorithm. When the user need to download any file, that time the file will be decrypted after that only the file will download.

File Downloading:

When a user download an others files in our system, first the key validation is required. First the key validation is verified after that the file will be decrypted than the file downloaded.

The key aggregation property is especially useful when we expect the delegation to be efficient and flexible.The scheme enable a content provider to share her/his data in a confidential and selective way, with a fixed and small cipher text expansion, by distributing to each authorized user a single and small aggregate Key.In this system data sharing is more secure.

IV. ADVANCED ENCRYPTION STANDARD

As a cipher, AES has proven reliable. The only successful attacks against it have been side-channel attacks on weaknesses found in the implementation or key management of certain AES-based encryption products. (Side-channel attacks don't use brute force or theoretical weaknesses to break a cipher, but rather exploit flaws in the way it has been implemented.) The BEAST browser exploit against the TLS v1.0 protocol is a good example; TLS can use AES to encrypt data, but due to the information that TLS exposes, attackers managed to predict the initialization vector block used at the start of the encryption process.

Various researchers have published attacks against reduced-round versions of the Advanced Encryption Standard, and a research paper published in 2011 demonstrated that using a technique called a biclique attack could recover AES keys faster than a brute-force attack by a factor of between three and five, depending on the cipher version. Even this attack, though, does not threaten the practical use of AES due to its high computational complexity. AES comprises three block ciphers, AES-128, AES-192 and AES-256. Each cipher encrypts and decrypts data in blocks of 128 bits using cryptographic keys of 128-, 192- and 256-bits, respectively. Rijndael was designed to handle additional block sizes and key lengths, but the functionality was not adopted in AES. Symmetric or secret-key ciphers use the same key for encrypting and decrypting, so both the sender and the receiver must know and use the same secret key. All key lengths are deemed sufficient to protect classified information up to the "Secret" level with "Top Secret" information requiring either 192- or 256-bit key lengths. There are 10 rounds for 128-bit keys, 12 rounds for 192-bit keys, and 14 rounds for 256-bit keys -- a round consists of several processing steps that include substitution, transposition and mixing of the input plaintext and transform it into the final output of cipher text.

For cryptographers, a cryptographic "break" is anything faster than a brute force—performing one trial decryption for each key (see Cryptanalysis). This includes results that are infeasible with current technology. The largest successful publicly known brute force attack against any block-cipher encryption was against a 64-bit RC5 key by distributed.net in 2006.AES has a fairly simple algebraic description. In 2002, a theoretical attack, termed the "XSL attack", was announced by Nicolas Courtois and Josef Pieprzyk, purporting to show a weakness in the AES algorithm due to its simple

description. Since then, other papers have shown that the attack as originally presented is unworkable; XSL attack on block ciphers. During the AES process, developers of competing algorithms wrote of Rijndael, "...we are concerned about [its] use...in security-critical applications." However, in October 2000 at the end of the AES selection process, Bruce Schneier, a developer of the competing algorithm Twofish, wrote that while he thought successful academic attacks on Rijndael would be developed someday, he does not "believe that anyone will ever discover an attack that will allow someone to read Rijndael traffic."

V. CONCLUSION AND FUTURE WORK

How to protect users' data privacy is a central question of cloud storage. With more mathematical tools, cryptographic schemes are getting more versatile and often involve multiple keys for a single application. In this article, we consider how to "compress" secret keys in public-key cryptosystems which support delegation of secret keys for different cipher text classes in cloud storage. No matter which one among the power set of classes, the delegate can always get an aggregate key of constant size. Our approach is more flexible than hierarchical key assignment which can only save spaces if all key-holders share a similar set of privileges. A limitation in our work is the predefined bound of the number of maximum cipher text classes. In cloud storage, the number of cipher texts usually grows rapidly. So we have to reserve enough cipher text classes for the future extension. Otherwise, we need to expand the public-key. Although 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. On the other hand, when one carries the delegated keys around in a mobile device without using special trusted hardware, the key is prompt to leakage, designing a leakage resilient cryptosystem yet allows efficient and flexible key delegation is also an interesting direction. In feature we can enhance file sharing to multiple users at a time an also we can enhance the cloud services.

REFERENCES

[1] Cheng-Kang Chu, Sherman S.M. Chow, 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, Vol. 25, NO. 2, Feburary 2014.

[2] 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.

[3] L. Hardesty, "Secure computers aren't so secure," MIT press, 2009, http://www.physorg.com/news176107396.html.

[4] 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.

[5] B. Wang, S. S. M. Chow, M. Li, and H. Li, "Storing Shared Data on the Cloud via Security-Mediator," in International Conference on Distributed Computing Systems - ICDCS 2013. IEEE, 2013.

[6] S. S. M. Chow, C.-K. Chu, X. Huang, J. Zhou, and R. H. Deng, "Dynamic Secure Cloud Storage with Provenance,"

[7] D. Boneh, C. Gentry, B. Lynn, and H. Shacham, "Aggregate and Verifiably Encrypted Signatures from Bilinear Maps,"

[8] M. J. Atallah, M. Blanton, N. Fazio, and K. B. Frikken, "Dynamic and Efficient Key Management for Access Hierarchies,"

[9] J. Benaloh, M. Chase, E. Horvitz, and K. Lauter, "Patient Controlled Encryption: Ensuring Privacy of Electronic Medical Records,"

[10] F. Guo, Y. Mu, Z. Chen, and L. Xu, "Multi-Identity Single-Key Decryption without Random Oracles,"

[11] V. Goyal, O. Pandey, A. Sahai, and B. Waters, "Attribute-Based Encryption for Fine-Grained Access Control of Encrypted data,"

[12] S. G. Akl and P. D. Taylor, "Cryptographic Solution to a Problem of Access Control in a Hierarchy,"