Security Check in Cloud Computing through Third Party Auditor

Govinda.K[1] J.Keerthana[2] U.Pavithra[3]
[1,2,3]SCSE, VIT University, Vellore, India

Abstract— In cloud computing, data owners crowd their data on cloud servers and users (data consumers) can access the data from cloud servers. Due to the data outsourcing, however, it requires an independent auditing service to check the data integrity in the cloud. Some existing remote integrity checking method scan only serve for static records data. Thus, cannot be used in the auditing service since the data in the cloud can be animatedly updated. Thus, an efficient and secure dynamic auditing protocol is required to convince data owners that the data are correctly stored in the cloud. In this paper, we first design an auditing framework for cloud storage systems for privacy-preserving auditing protocol. Then, we extend our auditing protocol to support the data dynamic operations, which is efficient to secure the random model.

Key words: Resources, Rack, Region, Data, Cloud

I. INTRODUCTION

Cloud Computing has been envisioned as the next generation architecture of IT enterprise, due to its extraordinary advantages in the history of IT: on-demand self-service, everywhere network access, location independent resource pooling, rapid resource elasticity, usage-based pricing and transference of risk [1]. As a troublemaking technology with profound implications, Cloud Computing is transformed how business peoples use IT. One fundamental aspect of this concept shifting is that data is being centralized to the Cloud. From us viewpoint , including both individuals and IT enterprises, storing data remotely into the cloud in a flexible on-demand manner brings appealing benefits: relief of the burden for storage management, universal data access with independent geographical locations, and avoidance of capital expenditure on hardware, software, and personnel maintenances, etc [2].While Cloud makes these advantages more Applicant than ever, it also brings new and challenging security to users outsourced data. Since cloud service providers (CSP) are separate administrative entities, data outsourcing is actually surrender user's ultimate control over the outcome of their data. The correctness of the data in the cloud is being put at risk due to the following reasons. First of all, although the infrastructures under the cloud are much more powerful than personal computing Devices. Examples: outages and security breach of important cloud services appear from time to time [3]–[5].

Secondly, they exist various motivations for cloud service provider behaves disloyally towards the cloud users regarding the status of their outsourced data. Examples: cloud service providers, for monetary reasons, reclaiming storage by discarding data that has not been loss incidents so as to maintain a reputation [6]–[8]. Although outsourcing data into the cloud is economically attractive and complexity of long-termdata storage, it does not suggest any security on data honesty and availability. This problem, if not properly addressed, may lead to the successful exploitation of the cloud architecture. As users no longer physically own the storage of their data, cryptographic primitives for the purpose of data security protection can't be directly adopted. Thus, how to efficiently verify the correctness of outsourced cloud data without the local copy of data files becomes a big challenge for data storage security in Cloud Computing.

Note that simply downloading the data for its integrity verification and not a practical solution due to the expensiveness in I/O cost and transmitting the file across the network. Besides, it is often insufficient to detect the data corruption when accessing the data. Considering the large amount of the outsourced data and the user's controlled resource capacity, the ability to audit the correctness of the data in a cloud environment can be terrible and expensive for the cloud users [8], [9].Hence, to fully ensure the data security and save the cloud users computation resources, it is of critical importance to enable public auditability for cloud data storage. so that the users may resort to a third party auditor (TPA), who has expertise that the users do not, to audit the outsourced data when needed. Based on the audit result, TPA could release an audit report, which would not only help users to evaluate the risk of their subscribed cloud data services, but also be useful for the cloud service provider to improve their cloud based service platform [7]. By enabling public risk auditing protocols will play an important role on cloud economy to become fully established, where users can assess risk and gain trust in Cloud.

Recently, the notion of public auditability has been proposed in the context of ensuring remotely stored data integrity under different systems and security models [6], [8], [10], [11].Public auditability allows an external party, in addition to the user himself, to verify the correctness of remotely stored data. However, most of these schemes [6], [8], [10] do not support the privacy protection of users' data against external auditors, i.e., they may potentially reveal user data information to the auditors, as will be discussed in Section III-C. This severe drawback greatly affects the security of these protocols in Cloud Computing. From the perspective of protecting data privacy the users, who own the data and rely on TPA just for test orage security of their data, do not want this auditing process introducing new vulnerabilities of unauthorized information leakage towards their data security [12]. Moreover, there are legal regulations, such as the US Health Insurance Portability and Accountability Act (HIPAA) [13], further demanding the outsourced data not to be leaked to external parties [7].Exploiting data encryption before outsourcing [11] is one way to mitigate this privacy concern, but it is only complementary to the privacy-preserving public auditing scheme to be proposed in this paper. Without a properly designed auditing protocol, encryption itself cannot prevent data from "flowing away" towards external parties during the auditing process. Thus, it does not completely solve the problem of protecting data privacy but just reduces it to the one of managing the encryption keys. Unauthorized data leakage still remains a problem due to the potential exposure of encryption keys.

II. LITERATURE REVIEW

ABE was proposed by Sahai and Waters . In ABE, a user has a set of attributes in addition to its unique ID. There are two classes of ABEs. In Key-policy ABE or KP-ABE (Goyal*et al.* the sender has an access policy to encrypt data. A writer whose attributes and keys have been revoked cannot write back the old informations. The receiver receives attributes and secret keys from the attribute authority and is able to decrypt information if it has matching attributes. In Cipher text-policy, CP-ABE the receiver has the access policy in the form of a tree, with attributes as leaves and monotonic access structure with AND, OR and other threshold gates.

All the approaches take a centralized concept and allows only one KDC, which is a single point of failure. Trail proposed a multi-authority ABE, in which there are several KDC authorities (coordinated by a trusted authority) which distribute attributes and secret keys to users. Multi-authority ABE protocol was studied in which required no reliable authority which requires every user to have attributes from at KDCs. Recently, Lewko and Waters proposed a fully decentralized ABE where users could have zero or more attributes from each authority and did not require a trusted server. In all these cases, decryption at user's end is computation demanding. So, this technique might be useless when users access their mobile devices. To clear this problem, Green *et* al. planned to outsource the decryption task to a replacement server, so that the user can compete with minimum resources. Example: hand held devices. However, the occurrence of one replacement and one key distribution centre makes it less forceful than decentralized approaches. Both these approaches had no way to authenticate users, anaemically. Yang *et al.* presented a modification of authenticate users, who want to remain secrete while accessing the cloud. To ensure anonymous user authentication Attribute Based Signatures were introduced by Maji*et al.* [23]. This was also a centralized approach. A recent scheme by the same authors [24] takes a decentralized approach and provides authentication without disclosing.

III. PROPOSED METHOD

The third party auditor will use MD5 algorithm to do security check.

This message can be a string of any variable length containing alphabets, digits and special symbols. In general formula:

Message = ((a-z) + (A-Z) + (0-9) + (!-;))*

The message digest should be a string of fixed length containing of all the alphabets and special symbols.

Message Digest= ((a-z) + (A-Z) + (0-9) + (! -;))n,

Where n=a (a:fixed natural number)

While working on MySQL databases and PHP for web applications we came across MD5. We also marked an application called "Cain n Able" which can crack messages from their MD5 hash value.

- When analytic work indicated that MD5's predecessor MD4 was likely to be insecure. MD5 was designed in 1991 to be a secure replacement.
- In 1993 Den Boer gave a result of finding a "pseudo-collision" of the MD5 compression function. That is, two different initialization vectors which produce an identical digest.
- In 1996Dobbertin announced a collision of the compression function of MD5 While this was not an full MD5 hash function, it was close enough for cryptographers to recommend switching to a replacement, such asSHA-1 or RIPEMD-160.
- Recently, a number of projects have created under MD5 rainbow tables which are easily available in online, and can be used to reverse many MD5 hashes into strings that bump with the original input, usually for the purposes of password cracking. If passwords are combined with a salt before the MD5 digest is generated, rainbow tables become much less useful.

So we thought of designing a new algorithm for calculating message digest.

IV. ALGORITHM

Input: A message in plaintext of any length.

Output: A message digest of 128 bit for the input message.

The algorithm works in the following steps:

Step1: Enter the input string.

Step2: Find the equivalent binary string.(Use ASCII conversion) .

Step 3:

i.) Append a bit sequence ("01") to the binary string so that the length of the resulting string is 64 shorter than a multiple of 512.

ii.) Append 64 more bits by scanning the binary string of step3 starting from an arbitrary location.

Step 4:

i.) Divide the output binary string of Step3 in 128 bit blocks.

ii.) Generate a 128 bit binary key using a random number generator.

iii.) Perform a bitwise operation (like OR , AND, XOR, followed by Left Shift, Right Shift, zero fill shifting etc.) among the 128 bit block and 128 bit random key.

iv.) Store the output of Step4(iii) as stepwise message digest.

Step 5:

i.) Perform a bitwise operation among the current stepwise message digest and the previous stepwise message digest.

ii.) Go to Step4 until all the blocks of input message are exhausted.

Step 6: Convert the output of Step5 into corresponding character value and store it as the final message digest.

V. DISCUSSION

The input message can be of any type. It may contain character, digit, special symbols etc.

Message = ((a-z) + (A-Z) + (0-9) + (! - ;))*

Binary equivalent of input message is calculated. This is done by using ASCII code for corresponding character.

For example: A = 01000001

B = 01000010 …….. etc.

The resulting binary string may contain any No. of digits. Then a bit sequence is appended (say "01") to the input binary string so that the length of the resulting string is 64 shorter than a multiple of 512. Mathematically length of binary string is %512=448. The bit sequences can also be 11, 10, 00 etc. Again 64 more bit sequence is appended to it from the input binary string itself starting from a arbitrary position. Here it starts from ([length of string]/3)th position. Now the binary string is of length which is a integer multiple of 512.this message string is then divided into blocks of 128 bits. A key is generated which is also of 128 bit. The key can be generated by using any of the random number.

Here we have used the method as

Key = (key*39) % 967

Keyf = (key)!

Binary equivalent of keyf is calculated and any 128 bit is taken as the key for a step. In each step one block of message string and a key is processed to give a message process. The process may involve operations like OR, AND, XOR, left shift, right shift etc. The stepwise message digests performs a bitwise operation with previous step message digest to form the final message. The resulting binary string is then converted into character string to get the final message digest. In this any input string the output message digest is of constant length. The message digest may contain any characters, digits, special characters etc.

i.e.: Message Digest= ((a-z) + (A-Z) + (0-9) + (! - ;)) n

Where n= a fixed natural number.

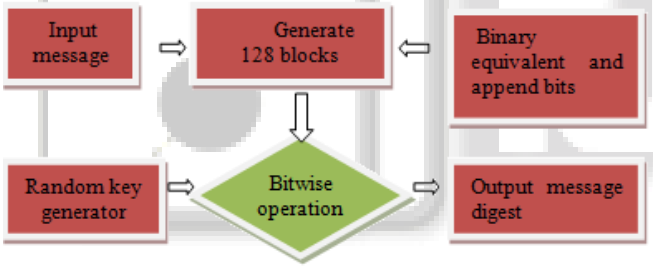

Fig.1: Layout of our proposed algorithm

VI. Conclusion

In this paper we explained different existing paper techniques and their merits and demerits. We discussed their methods of data security and privacy etc. In all those papers some have not described proper data security mechanisms, some were lack in supporting dynamic data operations, some were lack in ensuring data integrity, while some were lacking by high resource and computation cost. Hence this paper gives overall idea of all existing techniques for cloud data security and methods proposed for ensuring data authentication using TPA.

References

[1] Elsenpeter Robert, Anthony T.Velte and Toby J.Velte, Cloud Computing a Practical Approach 2010.

[2] Qian Wang and Cong Wang and Kui Ren, Wenjing Lou, Jin Li "Enabling Public Auditability And Data Dynamics For Storage Security in Cloud Computing" in IEEE transactions on parallel and distributed systems, 2011, vol. 22, no. 5.

[3] Cong Wang and Kui Ren and Wenjing Lou and JinLi,"Toward Publicly Auditable Secure Cloud Data Storage Services" in IEEE, 2010.

[4] M.Ashah and R. swaminathan and m.baker"Privacy-Preserving Audit And Extraction of Digital Contents", 2011.

[5] H. Shacham and B. Waters "Compact Proofs of Retrivability" in proc. of asiascrypt, 2008.

[6] Xiang Tan and Bo Ai "The Issue of Cloud Computing Security in High-Speed Railway" international confer. on electronic and mechanical engi. And information technology, 2011. Beijing p.r china,.

[7] FarzadSabahi, "Cloud Computing Security Threats and Responses" ,IEEE confer. 2011, 978-1-61284-486-2/111

[8] Ravi Kant Sahu and Abhishek Mohta, L.K. Awasthi "Robust Data Integration While Using Third Party Auditor For Cloud Data Storage Services", conf. IJARCSSE, 2012, Volume 2, Issue 2,ISSN: 2277 128X.

[9] Govinda V, and Gurunathaprasad, H Sathshkumar,"Third Party Auditing For Security Data Storage in cloud through digital signature using RSA" IJASATR, 2012, issue 2,vol-4, Issn 2249-9954.

[10] P. Mell and t. Grance "Draft Nist Working Definition of Cloud Computing", 2009.

[11] Elinor Mills, "Cloud Computing Security Forecast: Clear Skies", 2009.

[12] H. Shacham and B. Waters, "Compact Proofs of Retrievability," Proc. 14th Int'l Conf. Theory and Application of Cryptology and Information Security: Advances in Cryptology, J. Pieprzyk, ed.,pp. 90-107, 2008.

[13] C.C. Erway, A. Ku¨ pc¸u¨ , C. Papamanthou, and R. Tamassia, "Dynamic Provable Data Possession," Proc. ACM Conf. Computer and Comm. Security, E. Al-Shaer, S. Jha, and A.D. Keromytis, eds.,pp. 213-222, 2009.

[14] Q. Wang, C. Wang, K. Ren, W. Lou, and J. Li, "Enabling Public Auditability and Data Dynamics for Storage Security in Cloud Computing," IEEE Trans. Parallel Distributed Systems, vol. 22, no. 5, pp. 847-859, May 2011.

[15] C. Wang, Q. Wang, K. Ren, and W. Lou, "Privacy-Preserving Public Auditing for Data Storage Security in Cloud Computing," Proc. IEEE INFOCOM, pp. 525-533, 2010.

[16] Y. Zhu, H. Hu, G. Ahn, and M. Yu, "Cooperative Provable Data Possession for Integrity Verification in Multi-Cloud Storage," IEEE Trans. Parallel and Distributed Systems, vol. 23, no. 12, pp. 2231-2244, Dec. 2012.

