A Framework for Scrutiny of Secure Data Storage in Cloud Services

S.Venkataramana[1], Dr P.V.G.D. Prasad Reddy[2], Dr. S. Krishna Rao[3]

[1]Research Scholar, Dept. of CS & SE, Andhra University, Visakhapatnam, Andhra Pradesh, India.
[2]Dept. of CS&SE, Andhra University, Visakhapatnam, Andhra Pradesh, India.
[3]Dept. of I.T, Sir CRR Engineering College, Eluru, Andhra Pradesh, India.

Abstract— *In cloud computing, data owners host their data on cloud servers and users (data consumers) can access the data from cloud servers. Due to the data outsourcing, however, this new paradigm of data hosting service also introduces new security challenges, which requires an independent auditing service to check the data integrity in the cloud. Some existing remote integrity checking methods can only serve for static archive data and thus cannot be applied to the auditing service since the data in the cloud can be dynamically updated. Thus, an efficient and secure dynamic auditing protocol is desired 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 and propose an efficient and privacy-preserving auditing protocol. Then, we extend our auditing protocol to support the data dynamic operations, which is efficient and provably secure in the random oracle model. We further extend our auditing protocol to support batch auditing for both multiple owners and file verification. The analysis and simulation results show that our proposed auditing protocols are secure and efficient, especially it reduce the computation cost of the auditor.*

Keywords— Poly-alphabetic Cryptographic algorithm, Auditing, attacks, data privacy problem, multi cloud Batch Auditing, Hash algorithm 256.

I. INTRODUCTION

Cloud storage is an important service of cloud computing [1], which allows data owners (owners) to move data from their local computing systems to the cloud. More and more owners start to store the data in the cloud [2]. However, this new paradigm of data hosting service also introduces new security challenges [3]. Owners would worry that the data could be lost in the cloud. This is because data loss could happen in any Infrastructure, no matter what high degree of reliable measures cloud service providers would take [4]–[8].

Sometimes, cloud service providers might be dishonest. They could discard the data which has not been accessed or rarely accessed to save the storage space and claim that the data are still correctly stored in the cloud. Therefore, owners need to be convinced that the data are correctly stored in the cloud. Traditionally, owners can check the data integrity based on two-party storage auditing protocols [9]–[12]. In cloud storage system, however, it is inappropriate to let either side of cloud service providers or owners conduct such auditing, because none of them could be guaranteed to provide unbiased auditing result. In this situation, *third party auditing* is a natural choice for the storage auditing in cloud computing. A third party auditor (auditor) that has expertise and capabilities can do a more efficient work and convince both cloud service providers and owners. For the third party auditing in cloud storage systems, there are several important requirements which have been proposed in some previous works [13], [14]. The auditing protocol should have the following properties: 1) *Confidentiality*. The auditing protocol should keep owner's data confidential against the auditor. 2) *Dynamic Auditing*. The auditing protocol should support the dynamic updates of the data in the cloud.3) *Batch Auditing*. The auditing protocol should also be able to support the batch auditing for multiple owners and multiple clouds. Recently, several remote integrity checking protocols were proposed to allow the auditor to check the data integrity on the remote server.

In this paper, we propose an efficient and secure dynamic auditing protocol, which can meet the above listed requirements. To solve the data privacy problem, our method is to generate an encrypted proof with the challenge stamp by using the Poly alphabetic cryptographic algorithm, such that the auditor cannot decrypt it but can verify the Correctness of the proof. Without using the mask technique, our method does not require any trusted organizer during the batch auditing for multiple clouds. On the other hand, in our method, we let the server compute the proof as an intermediate value of the verification, such that the auditor can directly use this intermediate value to verify the correctness of the proof. Therefore, our method can greatly reduce the computing loads of the auditor by moving it to the cloud server. Our original contributions can be summarized as follows.

1) We design an auditing framework for cloud storage systems and propose a privacy-preserving and efficient storage auditing protocol. Our auditing protocol ensures the data privacy by using cryptography method and Poly alphabetic cryptographic algorithm. Our auditing protocol incurs less communication cost between the auditor and the server. It also reduces the computing loads of the auditor by moving it to the server.
2) We extend our auditing protocol to support the data dynamic operations, which is efficient and provably secure in the random oracle model.
3) We further extend our auditing protocol to support batch auditing for not only multiple clouds but also multiple owners. Our multi-cloud batch auditing does not require any additional trusted organizer. The multi-owner batch auditing can greatly improve the auditing performance, especially in large scale cloud storage systems.

II. RELATED WORK

SECURITY SERVICES

For the third party auditing in cloud storage systems [5], there are several important requirements which have been proposed in some previous works. The auditing protocol should have the following properties: 1) *Confidentiality*. The auditing protocol should keep owner's data confidential against the auditor. 2) *Dynamic Auditing*. The auditing protocol should support the dynamic updates of the data in the cloud. 3) *Batch Auditing*. The auditing protocol should also be able to support the batch auditing for multiple owners and multiple clouds.

Problem Statement: In the auditing process there may be chance to leak the received data. There may be chance to following attacks: Replay attack, Forge attack and Replace attack.

1) *Replace Attack*. The server may choose another valid and uncorrupted pair of data block and data tag (m_k, t_k) to replace the challenged pair of data block and data tag (m_i, t_i), when it already discarded m_i or t_i.

2) *Forge Attack*. The server may forge the data tag of data block and deceive the auditor; if the owner's secret tag keys are reused for the different versions of data.

3) *Replay Attack*. The server may generate the proof from the previous proof or other information, without retrieving the actual owner's data.

The main challenge in the design of data storage auditing protocol is the *data privacy problem* (i.e., the auditing protocol should protect the data privacy against the auditor.). This is because: 1) for public data, the auditor may obtain the data information by recovering the data blocks from the data proof. 2) For encrypted data, the Auditor may obtain content keys somehow through any special channels and could be able to decrypt the data. To solve the data privacy problem, our method is to generate an encrypted proof with the challenge stamp by using the Bi-linearity property of the bilinear pairing, such that the auditor cannot decrypt it. But the auditor can verify the correctness of the proof without decrypting it.

In existing system [6] many of the algorithms encrypting the plain text to cipher text. But the algorithms applying same encryption process to entire plain text. So if the same type of characters repeated in plain text, that all characters converting into the same type of cipher text. The cryptanalysis for this type of cipher texts is becoming easy process. For example if the plain text is "BANANA". In this plain text, A is repeated 3 times and N is repeated 2 times. In the present existed algorithms 3As and 2Ns will be encrypted in to same characters. In decryption 3 characters is enough to get this plain text. For those texts cryptanalysis will become easy for these type plain texts.

III. PROPOSED WORK

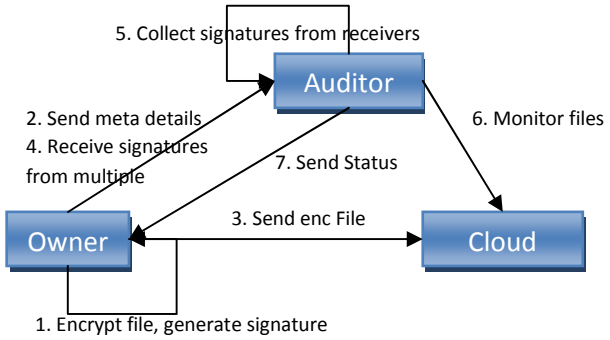
First data owner encrypts the selected file using our poly alphabetic symmetric key cryptographic algorithm. Proposed work consist three modules.

1. Encryption
2. Signature Generation
3. Auditing

1. Encryption: Encryption can be done the initial stage of the auditor after selecting the file that data owner encrypts the selected file using our poly alphabetic symmetric key cryptographic algorithm.

2. Signature Generation: After completion of the encryption the Data Owner generate the signature of the uploaded file in the CSP by using SHA 256.

3. Auditing: Then auditor verifies the files of multiple owners. Auditor gathers file information and decrypt the file and generate signature using the secret key. So that resultant digital signatures are compared with the received digital signature from the data owner. If the two digital Signature algorithms are same File is not corrupted, otherwise file is corrupted. The status of the auditing is sent to data owner. The proposed system architecture is shown below.



Initially during uploading of the file, data owner select particular auditor to audit his file. At that time auditor provide a security key to the data owner.

At the time auditing, auditor selects data owners and verifies the uploaded files using Meta data received from the data owner and security key. For authentication data owner generates signature using security key. That key generation process is explained below.

KeyGen(λ)$\rightarrow$(*pkt* , *skt* , *skh*). The key generation algorithm takes no input other than the implicit security parameter λ. It chooses two random numbers, for selecting random numbers generate two prime numbers from Z*p*. Calculate primitive roots of the two prime numbers and those two primitive roots are sk_t , sk_h *respectively and belongs to* Z*p* as the secret tag key and the secret hash key. It outputs the public tag key as $pk_t = g^{skt}{}_2 \in G_2$, the secret tag key sk_t and the secret hash key sk_h. Hash generated for sk_h is calculated by using simple hash function, which means second random value given input to hash function that explains as follows. For instance, suppose that each input is an integer *z* in the range 0 to $N-1$, and the output must be an integer *h* in the range 0 to $n-1$, where *N* is much larger than *n*. Then the hash function could be $h = z \bmod n$ (the remainder of *z* divided by *n*), or $h = (z \times n) \div N$ (the value *z* scaled down by n/N and truncated to an integer), or many other formulas.

For Signature Generation we adapt Secure Hash algorithm 256. Data owner encrypts the data and store in the server. Then he creates digital signature and send that and the details of file to auditor. That the data owner work is completed.

Then auditor verifies the files of multiple owners. Auditor gathers file information and decrypt the file and generate signature using the secret key. So that resultant digital signatures are compared with the received digital signature from the data owner. If the two digital signature Algorithms are same File is not corrupted, otherwise file is corrupted. The status of the auditing is sent to data owner.

Poly Alphabetic Cryptography Algorithm

Encryption:

1. Add the randomized characters in between the plain text. For every 3 characters add one duplicate character.
2. Get the ASCII codes for the characters in plain text.
3. Convert the ASCII codes into Binary format.
4. Do the complement of the plain text.
5. Select any series of prime numbers and convert into Binary format.
6. Do the first level Exclusive OR (XOR) between characters of plain text and selected series of prime numbers.
7. Select any Randomized number (key). Get the keyth prime number from the prime numbers table.
8. Do the Second level of XOR operation between result of step5 and Randomized prime number.
9. Convert the result of step7 into decimal values. Now you will get the cipher text.

Decryption:

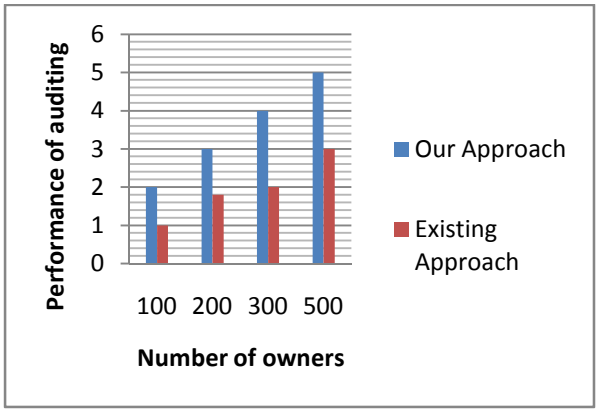
1. Convert the cipher text into Binary format. Get the Keyth prime number from the prime numbers table. And convert it into binary format.
2. Do the first level of Exclusive OR (XOR) operation between cipher text and Keyth primary key.
3. Select the series of prime numbers and convert it into the binary format (the series must be same in both encryption side and decryption side).
4. Do second level of XOR operation between result of step2 and selected series of prime numbers.
5. Get complement of the result of step4.
6. Convert the result from binary to decimal format.
7. Remove the randomized stuffed numbers.
8. Now you can get the plaintext.

By using this encryption algorithm maximum security provided to the file.

IV. RESULT ANALYSIS

The batch auditing for multiple owners can greatly reduce the computation cost. Although in our simulation the number of data owners goes to 500, it can illustrate the trend of computation cost of the auditor that is much more efficient than existing scheme in large scale cloud storage systems that may have millions to billions of data owners.Our framework introduces secure data auditing for multiple owners and secure data verification of multiple files. By using our protocol auditing process can be done in less amount of time. It supports more Scalability of users. This contains secure public tags and verification process such as auditing. It reduces work load to server because simple verification process is only done by server all other security issued can done by auditing. Furthermore, our auditing scheme incurs less communication cost and less computation cost of the

auditor by moving the computing loads of auditing from the auditor to the server.



V. CONCLUSION

In this paper we proposed framework that combines with cryptographic properties with secure storage. This contains secure public tags and verification process such as auditing. It reduces work load to server because simple verification process is only done by server all other security issued can done by auditing. Furthermore, our auditing scheme incurs less communication cost and less computation cost of the auditor by moving the computing loads of auditing from the auditor to the server.

REFERENCES

[1] P. Mell and T. Grance, "The NIST definition of cloud computing," National Institute of Standards and Technology, Tech. Rep., 2009.

[2] M. Armbrust, A. Fox, R. Griffith, A. D. Joseph, R. H. Katz, A. Konwinski, G. Lee, D. A. Patterson, A. Rabkin, I. Stoica, and M. Zaharia, "A view of cloud computing," *Commun. ACM*, vol. 53, no. 4, pp. 50–58, 2010.

[3] T. Velte, A. Velte, and R. Elsenpeter, *Cloud Computing: A Practical Approach*, 1st ed. New York, NY, USA: McGraw-Hill, Inc., 2010, ch. 7.

[4] J. Li, M. N. Krohn, D. Mazi`eres, and D. Shasha, "Secure untrusted data repository (sundr)," in *Proceedings of the 6th conference on Symposium on Operating Systems Design & Implementation*, Berkeley, CA, USA, 2004, pp. 121–136.

[5] G. R. Goodson, J. J. Wylie, G. R. Ganger, and M. K. Reiter, "Efficient byzantine-tolerant erasure-coded storage," in *DSN*. IEEE Computer Society, 2004, pp. 135–144.

[6] V. Kher and Y. Kim, "Securing distributed storage: challenges, techniques, and systems," in *StorageSS*, V. Atluri, P. Samarati, W. Yurcik, L. Brumbaugh, and Y. Zhou, Eds. ACM, 2005, pp. 9–25.

[7] L. N. Bairavasundaram, G. R. Goodson, S. Pasupathy, and J. Schindler, "An analysis of latent sector errors in disk drives," in *SIGMETRICS*, L. Golubchik, M. H. Ammar, and M. Harchol-Balter, Eds. ACM, 2007, pp. 289–300.

[8] B. Schroeder and G. A. Gibson, "Disk failures in the real world: What does an mttf of 1, 000, 000 hours mean to you?" in *FAST*. USENIX, 2007, pp. 1–16.

[9] M. Lillibridge, S. Elnikety, A. Birrell, M. Burrows, and M. Isard, "A cooperative internet backup scheme," in *USENIX Annual Technical Conference, General Track*. USENIX, 2003, pp. 29–41.

[10] Y. Deswarte, J. Quisquater, and A. Saidane, "Remote integrity checking," in *The Sixth Working Conference on Integrity and Internal Control in Information Systems (IICIS)*. Springer Netherlands, November 2004.

[11] M. Naor and G. N. Rothblum, "The complexity of online memory checking," *J. ACM*, vol. 56, no. 1, 2009.

[12] A. Juels and B. S. K. Jr., "Pors: proofs of retrievability for large files," in *ACM Conference on Computer and Communications Security*, P. Ning, S. D. C. di Vimercati, and P. F. Syverson, Eds. ACM, 2007, pp. 584–597.

[13] S.Venkataramana. PVGD Prasad Reddy and S.K. Rao, "A Novel Technique For Scalable Efficient Deployment Of Software In Cloud Computing Environments" in International Journal Of Engineering Sciences & Research Technology (IJESRT), 2016, p.489-492.

[14] T. J. E. Schwarz and E. L. Miller, "Store, forget, and check: Using algebraic signatures to check remotely administered storage," in *ICDCS*. IEEE Computer Society, 2006, p. 12.