IJIRCCE

e-ISSN: 2320-9801 | p-ISSN: 2320-9798

INTERNATIONAL JOURNAL OF INNOVATIVE RESEARCH

IN COMPUTER & COMMUNICATION ENGINEERING

Volume 8, Issue 8, August 2020

INTERNATIONAL STANDARD SERIAL NUMBER INDIA

Impact Factor: 7.488

9940 572 462 | 6381 907 438 | ijircce@gmail.com | www.ijircce.com

International Journal of Innovative Research in Computer and Communication Engineering

| e-ISSN: 2320-9801, p-ISSN: 2320-9798| www.ijircce.com | Impact Factor: 7.488 |

|| Volume 8, Issue 8, August 2020 ||

Privacy-Preserving Mechanism that Supports Public Auditing on Shared Data Stored in Block Chain

Mr.M.Shanmuganathan[1], Praveen M[2], Magesh K[3], Praveen Kumar S[4]

Assistant Professor, Department of CSE, Panimalar Engineering College, Chennai, India[1]

Student, Department of CSE, Panimalar Engineering College, Chennai, India[2,3,4]

ABSTRACT: There is a need to develop an effective public auditing protocol which overcomes the limitation of the existing auditing scheme. The proposed system is developed to verify the correctness of cloud data by TPA, periodically or on demand without retrieving the entire data or without introducing additional online burden to the cloud users and cloud servers. It assure that no data content is leaked to TPA during the auditing process. It maintains storage correctness of data, integrity and confidentiality of stored data. The proposed scheme consists of three basic entities; they are data owner,cloud server storage and TPA. The data owner or the user is responsible for splitting the file into blocks, encrypting those using AES algorithm, generating a MD-5 hash value for each,concatenating the hashes and generates a AES signature on it.The cloud server is used to store the encrypted blocks of files.When the client or data owner request for data auditing to the TPA, it immediately request for the encrypted data from the cloud server. After receiving the data, it generated the hash value for each block of encrypted files. It uses the same MD-5 algorithm which was used by client. It later concatenate those hash values and generates a AES signature for that file. In the Verification process, the signature generated by TPA and the one stored in the TPA which is provided by the data user are compared by the TPA. If they matches with each other it means that the data is intact and data is not been tampered by any outsider or attacker. If it does not matches then it indicates that the data integrity has been affected or tampered. The result for the data integrity check is provided to the data owner.

KEYWORDS: Public Auditing, privacy preserving, blockchain, security, ring signatures, Design, Architecture, client and server codings

I.INTRODUCTION

In this paper, we propose a novel privacy-preserving mechanism that supports public auditing on shared data stored in the cloud. In particular, we exploit ring signatures to compute verification metadata needed to audit the correctness of shared data. With our mechanism, the identity of the signer on each block in shared data is kept private from public verifiers, who are able to efficiently verify shared data integrity without retrieving the entire file. In addition, our mechanism is able to perform multiple auditing tasks simultaneously instead of verifying them one by one.

The propose system, a privacy-preserving public auditing mechanism for shared data in the cloud. We utilize ring signatures to construct homomorphism authenticators, so that a publicverifier is able to audit shared data integrity without retrieving the entire data, yet it cannot distinguish who is the signer on each block. To improve the efficiency of verifying multiple auditing tasks, we further extend our mechanism to support batch auditing.

e-ISSN: 2320-9801, p-ISSN: 2320-9798| www.ijircce.com | Impact Factor: 7.488 |

|| Volume 8, Issue 8, August 2020 ||

II.ARCHITECTURE OF THEPROPOSED SYSTEM

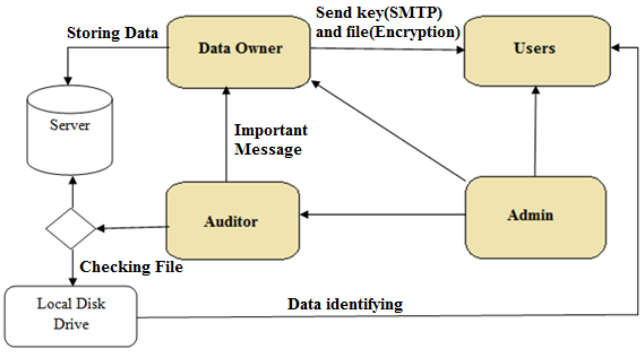

Figure 1: Architecture of System

Architecture Overview:

OVERVIEW OF THE SOFTWARE DEVELOPMENT PROCESS:

Because the Java VM isavailable on many different operating systems,the same class files are capable of running on Microsoft Windows,the SolarisTM Operating System(SolarisOS),Linux,orMacOS. Some virtual machines, such as the Java Hot Spot virtual machine perform additional steps at runtimeto give your applications performance boost. This include various tasks such as finding performance bottlenecks and recompiling (to native code) frequently used sections of code. Through the JavaVM, the same application iscapable of running on multiple platforms.

III.SYSTEM ANALYSIS

Existing System:

• The existing mechanism a new significant privacy issue introduced in the case of shared data with the use of the leakage of identity privacy to public verifiers.

•The traditional approach for checking data correctness is to retrieve the entire data from the cloud, and then verify data integrity by checking the correctness of signatures.
• To securely introduce an effective third party auditor(TPA), the following two fundamental requirements have to be met:

•1)TPA should be able to efficiently audit the cloud data storage without demanding the local copy of data, and introduce no additional on-line burden to the cloud user

• 2)The third party auditing process should bring in no new vulnerabilities towards user data privacy

IV.PROPOSED SYSTEM

[1] The proposed system, a privacy-preserving public auditing mechanism for shared data in the cloud.

[2] We utilize ring signatures to construct homomorphism authenticators, so that a public verifier is able to audit shared data integrity without retrieving the entire data, yet it cannot distinguish who is the signer on each block.

[3] To improve the efficiency of verifying multiple auditing tasks, we further extend our mechanism to support batch auditing.

[4] We can add traceability to the system, which means the ability for the group manager to reveal the identity of the signer based on verification metadata in some special situations.

V.REQUIREMENT ANALYSIS AND SPECIFICATION:

Input Requirement:

Data from user such as Name, DOB, Address, Phone number, E-mail id.

Output Requirement:

All user details are verified by auditor without making any changes to data or security breaches.

Functional Requirement

Software Used:
Language - Java(JDK 1.7)
OS - Windows 7 32bit
My SQL Server
Net Beans IDE 7.1.2

Hardware Used:
1 GB RAM
80 GB Hard Disk
Above 2GHz Processor
Data Card

Technology Used:

Programming Language : Java
Application : Servlet ,MVC
Web Server : Glassfish Server
Front end Language : JSP, Java Script, CSS
Back End: My SQL Server, SQL Yog

VI.MODULE DESIGN SPECIFICATION

Module:

1. user registration.
2. public auditing.
3. sharing data.
4. integrity checking.

1. User Registration:

For the registration of user with identity ID the group manager randomly selects a number. Then the group manager adds into the group user list which will be used in the traceability phase. After the registration, user obtains a private key which will be used for group signature generation and file decryption.

2. Public Auditing:

Homomorphic authenticators are unforgeable verification metadata generated from individual data blocks, which can be securely aggregated in such a way to assure an auditor that a linear combination of data blocks is correctly computed by verifying only the aggregated authenticator. Overview to achieve privacy-preserving public auditing, we propose to uniquely integrate the Homomorphic authenticator with random mask technique. In our protocol, the linear combination of

e-ISSN: 2320-9801, p-ISSN: 2320-9798| www.ijircce.com | Impact Factor: 7.488 |

|| Volume 8, Issue 8, August 2020 ||

sampled blocks in the server's response is masked with randomness generated by a pseudo random function (PRF). The proposed scheme is as follows:

> Setup Phase
> Audit Phase

3. Sharing Data:

The canonical application is data sharing. The public auditing property is especially useful when we expect the delegation to be efficient and flexible. The schemes enable a content provider to share her data in a confidential and selective way, with a fixed and small ciphertext expansion, by distributing to each authorized user a single and small aggregate key.

4. Integrity Checking:

Hence, supporting data dynamics for privacy-preserving public risk auditing is also of paramount importance. Now we show how our main scheme can be adapted to build upon the existing work to support data dynamics, including block level operations of modification, deletion and insertion. We can adopt this technique in our design to achieve privacy-preserving public risk auditing with support of data dynamics. The user download the particular file not download entire file.

VII.PROGRAM DESIGN LANGUAGE

ALGORITHM

Advanced Encryption Standard (AES):

The Advanced Encryption Standard (AES) is an encryption algorithm for securing sensitive but unclassified material by U.S. Government agencies and, as a likely consequence, may eventually become the de facto encryption standard for commercial transactions in the private sector. (Encryption for the US military and other classified communications is handled by separate, secret algorithms.)In January of 1997, a process was initiated by the National Institute of Standards and Technology (NIST), a unit of the U.S. Commerce Department, to find a more robust replacement for the Data Encryption Standard (DES) and to a lesser degree Triple DES. The specification called for a symmetric algorithm (same key for encryption and decryption) using block encryption (see block cipher) of 128 bits in size, supporting key sizes of 128, 192 and 256 bits, as a minimum. The algorithm was required to be royalty-free for use worldwide and offer security of a sufficient level to protect data for the next 20 to 30 years. It was to be easy to implement in hardware and software, as well as in restricted environments (for example, in a smart card) and offer good defenses against various attack techniques.The entire selection process was fully open to public scrutiny and comment, it being decided that full visibility would ensure the best possible analysis of the designs. In 1998, the NIST selected 15 candidates for the AES, which were then subject to preliminary analysis by the world cryptographic community, including the National Security Agency. On the basis of this, in August 1999, NIST selected five algorithms for more extensive analysis.

These were:

1. MARS, submitted by a large team from IBM Research
2. RC6, submitted by RSA Security
3. Rijndael, submitted by two Belgian cryptographers, Joan Daemen and Vincent Rijmen
4. Serpent, submitted by Ross Andersen, Eli Biham and Lars Knudsen
5. Twofish, submitted by a large team of researchers including Counterpane's respected cryptographer, Bruce Schneier

Implementations of all of the above were tested extensively in ANSI C and Java languages for speed and reliability in such measures as encryption and decryption speeds, key and algorithm set-up time and resistance to various attacks, both in hardware- and software-centric systems. Once again, detailed analysis was provided by the global cryptographic community (including some teams trying to break their own submissions). The end result was that on October 2, 2000, NIST announced that Rijndael had been selected as the proposed standard. On December 6, 2001, the Secretary of Commerce officially approved Federal Information Processing Standard (FIPS) 197, which specifies that all sensitive, unclassified documents will use Rijndael as the Advanced Encryption Standard.Also see cryptography, data recovery agent (DRA)RELATED GLOSSARY TERMS: RSA algorithm (Rivest-Shamir-Adleman), data key, greynet (or graynet), spam cocktail (or anti-spam cocktail), fingerscanning (fingerprint scanning),munging, insider threat, authentication server, defense in depth, non repudiation

HOW THEY WORK

AES is based on a design principle known as a Substitution permutation network. It is fast in both software and hardware. Unlike its predecessor, DES, AES does not use a Feistel network.AES has a fixed block size of 128 bits and a key size of 128, 192, or 256 bits, whereas Rijndael can be specified with block and key sizes in any multiple of 32 bits, with a minimum of 128 bits. The blocksize has a maximum of 256 bits, but the key size has no theoretical maximum.AES operates on a 4×4 column-major order matrix of bytes, termed the *state* (versions of Rijndael with a larger block size have additional columns in the state). Most AES calculations are done in a special field. The AES cipher is specified as a number of repetitions of transformation rounds that convert the input plaintext into the final output of ciphertext. Each round consists of several processing steps, including one that depends on the encryption key. A set of reverse rounds are applied to transform ciphertext back into the original plaintext using the same encryption key.

HIGH-LEVEL DESCRIPTION OF THE ALGORITHM

1. KeyExpansion—round keys are derived from the cipher key using Rijndael's key schedule
2. Initial Round
    1. AddRoundKey—each byte of the state is combined with the round key using bitwise xor
3. Rounds
    1. SubBytes—a non-linear substitution step where each byte is replaced with another according to a lookup table.
    2. ShiftRows—a transposition step where each row of the state is shifted cyclically a certain number of steps.
    3. MixColumns—a mixing operation which operates on the columns of the state, combining the four bytes in each column.
    4. AddRoundKey
4. Final Round (no MixColumns)
    1. SubBytes
    2. ShiftRows
    3. AddRoundKey

MD5:

MD5 is an algorithm that is used to verify data integrity through the creation of a 128-bit message digest from data input (which may be a message of any length) that is claimed to be as unique to that specific data as a fingerprint is to the specific individual. MD5, which was developed by Professor Ronald L. Rivest of MIT, is intended for use with digital signature applications, which require that large files must be compressed by a secure method before being encrypted with a secret key, under a public key cryptosystem. MD5 is currently a standard, Internet Engineering Task Force (IETF) Request for Comments (RFC) 1321. According to the standard, it is "computationally infeasible" that any two messages that have been input to the MD5 algorithm could have as the output the same message digest, or that a false message could be created throughapprehension of the message digest. MD5 is the third message digest algorithm created by Rivest. All three (the others are MD2 and MD4) have similar structures, but MD2 was optimized for 8-bit machines, in comparison with the two later formulas, which are optimized for 32-bit machines. The MD5 algorithm is an extension of MD4, which the critical review found to be fast, but possibly not absolutely secure. In comparison, MD5 is not quite as fast as the MD4 algorithm, but offers much more assurance of data security.

VIII.CONCLUSION

We propose a novel privacy-preserving mechanism that supports public auditing on shared data stored in the cloud.

IX.FUTURE WORK

In Our future work will be how to avoid this type of re-computation introduced by dynamic groups while still preserving identity privacy from the public verifier during the process of public auditing on shared data.

e-ISSN: 2320-9801, p-ISSN: 2320-9798| www.ijircce.com | Impact Factor: 7.488 |

|| Volume 8, Issue 8, August 2020 ||

REFERENCES

[1] 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," Communications of the ACM, vol. 53, no. 4, pp. 50–58, Apirl 2010.

[2] G. Ateniese, R. Burns, R. Curtmola, J. Herring, L. Kissner, Z. Peterson, and D. Song, "Provable Data Possession at Untrusted Stores," in Proc. ACM Conference on Computer and Communications Security (CCS), 2007, pp. 598–610.

[3] C. Wang, Q. Wang, K. Ren, and W. Lou, "Privacy-Preserving Public Auditing for Data Storage Security in Cloud Computing," in Proc. IEEE International Conference on Computer Communications (INFOCOM), 2010, pp. 525–533.

[4] R. L. Rivest, A. Shamir, and Y. Tauman, "How to Leak a Secret," in Proc. International Conference on the Theory and Application of Cryptology and Information Security (ASIACRYPT). SpringerVerlag, 2001, pp. 552–565.

[5] D. Boneh, C. Gentry, B. Lynn, and H. Shacham, "Aggregate and Verifiably Encrypted Signatures from Bilinear Maps," in Proc. International Conference on the Theory and Applications of Cryptographic Techniques (EUROCRYPT). Springer-Verlag, 2003, pp. 416–432.

[6] H. Shacham and B. Waters, "Compact Proofs of Retrievability," in Proc. International Conference on the Theory and Application of Cryptology and Information Security (ASIACRYPT). SpringerVerlag, 2008, pp. 90–107.

[7] Y. Zhu, H. Wang, Z. Hu, G.-J. Ahn, H. Hu, and S. S.Yau, "Dynamic Audit Services for Integrity Verification of Outsourced Storage in Clouds," in Proc. ACM Symposium on Applied Computing (SAC), 2011, pp. 1550–1557

INNO SPACE

SJIF Scientific Journal Impact Factor

Impact Factor:
7.488

ISSN INTERNATIONAL STANDARD SERIAL NUMBER INDIA

निस्केयर NISCAIR

INTERNATIONAL JOURNAL OF INNOVATIVE RESEARCH

IN COMPUTER & COMMUNICATION ENGINEERING

9940 572 462 6381 907 438 ijircce@gmail.com

www.ijircce.com

Scan to save the contact details