ISSN:0975-9646

Public Auditing on Shared Data in the Cloud Using Ring Signature Mechanism

MS. Gayatri D Patwardhan **Prof.B. W. Balkhande**

Abstract: Cloud computing is the long dreamed vision of computing as a utility, where users can remotely store their data into the cloud so as to enjoy the on-demand high quality applications and services from a shared pool of configurable computing resources. By data outsourcing, users can be relieved from the burden of local data storage and maintenance. Thus, enabling public auditability for cloud data storage security is of critical importance so that users can resort to an external audit party to check the integrity of outsourced data when needed. 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. Specifically, our contribution in this work can be summarized as the following three aspects: 1) We motivate the public auditing system of data storage security in Cloud Computing and provide a homomorphic authenticable ring signature mechanism for Public Auditing on shared data in the Cloud, i.e., our scheme supports an external auditor to audit user's outsourced data in the cloud without learning knowledge on the data content.
2) To the best of our knowledge, our scheme is the first to support scalable and efficient public auditing in the Cloud Computing. In particular, our scheme achieves batch auditing where multiple delegated auditing tasks from different users can be performed simultaneously by the TPA.

INTRODUCTION:

Cloud service providers manage an enterprise-class infrastructure that offers a scalable, secure and reliable environment for users, at a much lower marginal cost due to the sharing nature of resources. It is routine for users to use cloud storage services to share data with others in a team, as data sharing becomes a standard feature in most cloud storage offerings, including Drop box and Google Docs.

The integrity of data in cloud storage, however, is subject to skepticism and scrutiny, as data stored in an untrusted cloud can easily be lost or corrupted, due to hardware failures and human errors. To protect the integrity of cloud data, it is best to perform public auditing by introducing a third party auditor (TPA), who offers its auditing service with more powerful computation and communication abilities than regular users.

The first provable data possession (PDP) mechanism to perform public auditing is designed to check the correctness of data stored in an untrusted server, without retrieving the entire data. Moving a step forward, Wang et al. (referred to as WWRL) is designed to construct a public auditing mechanism for cloud data, so that during public auditing, the content of private data belonging to a personal user is not disclosed to the third party auditor.

We believe that sharing data among multiple users is perhaps one of the most engaging features that motivates cloud storage. A unique problem introduced during the process of public auditing for shared data in the cloud is how to preserve identity privacy from the TPA, because the identities of signers on shared data may indicate that a particular user in the group or a special block in shared data is a higher valuable target than others. For example, Alice and Bob work together as a group and share a file in the cloud. The shared file is divided into a number of small blocks, which are independently signed by users. Once a block in this shared file is modified by a user, this user needs to sign the new block using her public/private key pair. The TPA needs to know the identity of the signer on each block in this shared file, so that it is able to audit the integrity of the whole file based on requests from Alice or Bob.

We propose the system, a new privacy preserving public auditing mechanism for shared data in an untrusted cloud. In this system, we utilize ring signatures to construct homomorphic authenticators, so that the third party auditor is able to verify the integrity of shared data for a group of users without retrieving the entire data — while the identity of the signer on each block in shared data is kept private from the TPA. In addition, we further extend our mechanism to support batch auditing, which can audit multiple shared data simultaneously in a single auditing task. Meanwhile, it continues to use random masking to support data privacy during public auditing, and leverage index hash tables to support fully dynamic operations on shared data. A dynamic operation indicates an insert, delete or update operation on a single block in shared data. A high-level comparison between this system and existing mechanisms in the literature is shown. To our best knowledge, this represents the first attempt towards designing an effective privacy preserving public auditing mechanism for shared data in the cloud.

1. EXISTING SYSTEM

Many mechanisms have been proposed to allow not only a data owner itself but also a public verifier to efficiently perform integrity checking without downloading the entire data from the cloud, which is referred to as public auditing . In these mechanisms, data is divided into many small blocks, where each block is independently signed by the owner; and a random combination of all the blocks instead

of the whole data is retrieved during integrity checking. A public verifier could be a data user (e.g., researcher) who would like to utilize the owner's data via the cloud or a third-party auditor (TPA) who can provide expert integrity checking services.

Moving a step forward, Wang et al. designed an advanced auditing mechanism .so that during public auditing on cloud data, the content of private data belonging to a personal user is not disclosed to any public verifiers. Unfortunately, current public auditing solutions mentioned above only focus on personal data in the cloud .We believe that sharing data among multiple users is perhaps one of the most engaging features that motivates cloud storage. Therefore, it is also necessary to ensure the integrity of shared data in the cloud is correct.

Existing public auditing mechanisms can actually be extended to verify shared data integrity. However, a new significant privacy issue introduced in the case of shared data with the use of existing mechanisms is the leakage of identity privacy to public verifiers. However, a new significant privacy issue introduced in the case of shared data with the use of existing mechanisms is the leakage of identity privacy to public verifiers.

Disadvantages of existing system:

1. Failing to preserve identity privacy on shared data during public auditing will reveal significant confidential information to public verifiers.
2. Protect these confidential information is essential and critical to preserve identity privacy from public verifiers during public auditing.

2. PROPOSED SYSTEM:

To solve the above privacy issue on shared data, we propose a novel privacy-preserving public auditing mechanism. More specifically, we utilize ring signatures to construct homomorphic authenticators in this system, so that a public verifier is able to verify the integrity of shared data without retrieving the entire data while the identity of the signer on each block in shared data is kept private from the public verifier.

In addition, we further extend our mechanism to support batch auditing, which can perform multiple auditing tasks simultaneously and improve the efficiency of verification for multiple auditing tasks. Meanwhile, this system is compatible with random masking, which has been utilized in WWRL and can preserve data privacy from public verifiers. Moreover, we also leverage index hash tables from a previous public auditing solution to support dynamic data. A high-level comparison among this system and existing mechanisms is presented.

Advantages of proposed system:

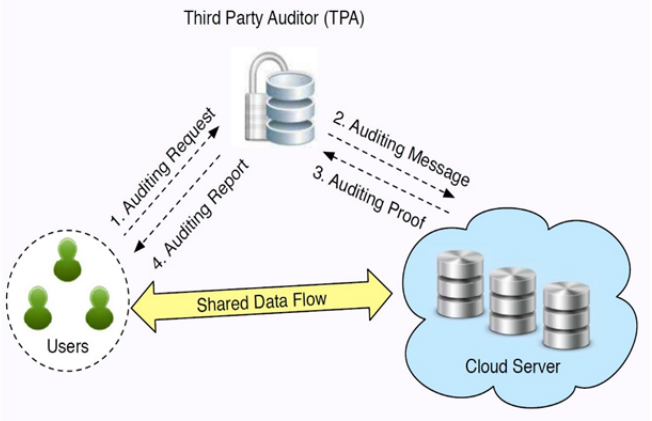
1. A public verifier is able to correctly verify shared data integrity.
2. A public verifier cannot distinguish the identity of the signer on each block in shared data during the process of auditing.
3. The ring signatures generated for not only able to preserve identity privacy but also able to support block less verifiability.

3. PROBLEM STATEMENT:

3.1 SYSTEM MODEL:

This application involves three parties: the cloud server, the third party auditor (TPA) and users. There are two types of users in a group: the original user and a number of group users.

The original user and group users are both members of the group. Group members are allowed to access and modify shared data created by the original user based on access control polices. Shared data and its verification information (i.e. signatures) are both stored in the cloud server. The third party auditor is able to verify the integrity of shared data in the cloud server on behalf of group members. Our system model includes the cloud server, the third party auditor and users. The user is responsible for deciding who is able to share her data before outsourcing data to the cloud. When a user wishes to check the integrity of shared data, she first sends an auditing request to the TPA. After receiving the auditing request, the TPA generates an auditing message to the cloud server, and retrieves an auditing proof of shared data from the cloud server. Then the TPA verifies the correctness of the auditing proof. Finally, the TPA sends an auditing report to the user based on the result of the verification.



3.1 system model

3.2 THREAT MODEL

3.2.1 Integrity Threats

Two kinds of threats related to the integrity of shared data are possible. First, an adversary may try to corrupt the integrity of shared data and prevent users from using data correctly. Second, the cloud service provider may inadvertently corrupt (or even remove) data in its storage due to hardware failures and human errors. Making matters worse, in order to avoid this integrity threat the cloud server provider may be reluctant to inform users about such corruption of data.

3.2.2 Privacy Threats

The identity of the signer on each block in shared data is private and confidential to the group. During the process of auditing, a semi-trusted TPA, who is only responsible for auditing the integrity of shared data, may try to reveal the identity of the signer on each block in shared data based on verification information. Once the TPA reveals the identity of the signer on each block, it can easily distinguish a high-value target (a particular user in the group or a special block in shared data).

3.3 DESIGN OBJECTIVES

To enable the TPA efficiently and securely verify shared data for a group of users, it should be designed to achieve following properties:

(1)	**Public Auditing**	The third party auditor is able to publicly verify the integrity of shared data for a group of users without retrieving the entire data.
(2)	**Correctness**	The third party auditor is able to correctly detect whether there is any corrupted block in shared data.
(3)	**Unforgeability**	Only a user in the group can generate valid verification information on shared data.
(4)	**Identity Privacy**	During auditing, the TPA cannot distinguish the identity of the signer on each block in shared data.

4. METHODOLOGY:

4.1 PRIVACY PRESERVING PUBLIC AUDITING MODULE:

The details of our public auditing mechanism includes: Key-Gen, Sig-Gen, Modify, Proof-Gen and Proof Verify. In Key-Gen, users generate their own public/private key pairs. In Sig-Gen, a user is able to compute ring signatures on blocks in shared data. Each user is able to perform an insert, delete or update operation on a block, and compute the new ring signature on this new block in Modify. Proof Gen is operated by the TPA and the cloud server together to generate a proof of possession of shared data. In Proof-Verify, the TPA verifies the proof and sends an auditing report to the user.

The proposed scheme is as follows:

(1) Setup Phase
(2) Audit Phase

4.1.1 Setup Phase

The user initializes the public and secret parameters of the system by executing

KeyGen, and pre-processes the data file F by using SigGen to generate the verification of metadata. The user then stores the data file F and the verification metadata at the cloud server. The user may alter the data file F by performing updates on the stored data in cloud.

4.1.2 Audit Phase

TPA issues an audit message to the cloud server to make sure that the cloud server has retained the data file F properly at the time of the audit. The cloud server will create a response message by executing Genproof using F and its verification metadata as inputs. The TPA then verifies the response by cloud server via Verify Proof.

A owner is a person who can access resources from the cloud. The owner would first register to the interface to get the services with the valid username and password. In order to correctly audit the integrity of the entire data, a public verifier needs to choose the appropriate public key for each block. Then they can request for the file to the cloud service admin. There will be a third party auditor who performs the integrity checking of the data before providing it to the owner or the users. This is done by 1st splitting the data into blocks and then performing integrity check. The owner has the option of downloading the verified file and also uploads new files.

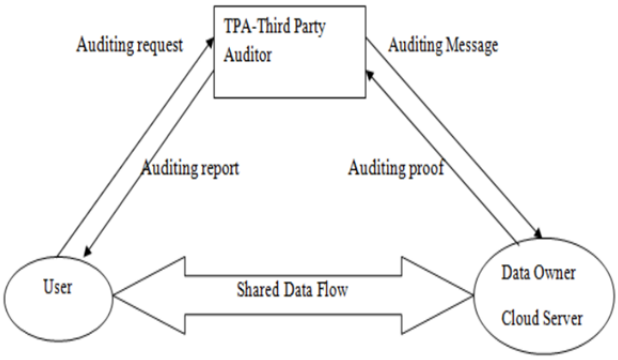


Fig 4.1 system architecture

4.2 BATCH AUDITING MODULE:

With the establishment of privacy-preserving public auditing in Cloud Computing, TPA may concurrently handle multiple auditing delegations upon different users' requests. The individual auditing of these tasks for TPA can be tedious and very inefficient. Batch auditing not only allows TPA to perform the multiple auditing tasks simultaneously, but also greatly reduces the computation cost on the TPA side. Given K auditing delegations on K distinct data files from K different users, it is more advantageous for TPA to batch these multiple tasks together and audit at one time.

The TPA registers to the application with a valid username and password. TPA logins to the application and verifies the integrity of data. TPA views all the list of files uploaded by the owner without the key. Has the privilege of encrypting the data and save it on cloud.TPA also view data which is uploaded by various owner.

4.3 DATA DYNAMICS MODULE:

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.

To enable each user in the group to easily modify data in the cloud and shared the latest version of the data with rest of the group, it should also support dynamic operations on shared data. A dynamic operation includes an insert, delete or update operation on a single block. However, since the computation of a ring signature includes an identifier of a block, traditional methods, which only use the index of a block as its identifier, are not suitable for supporting dynamic operations on shared data. The reason is that, when user modifies a single block in shared data by performing an insert or delete operation, the indices of blocks that after the modified block are all changed and the changes of these indices requires users to recomputed the signatures of these blocks, even though the content of these

blocks are not modified. The details of our public auditing mechanism in it includes: Key-Gen, Sig-Gen, Modify, Proof-Gen and Proof Verify. In Key-Gen, users generate their own public/private key pairs. In Sig-Gen, a user is able to compute ring signatures on blocks in shared data. Each user is able to perform an insert, delete or update operation on a block, and compute the new ring signature on this new block in Modify. Proof Gen is operated by the TPA and the cloud server together to generate a proof of possession of shared data. In Proof-Verify, the TPA verifies the proof and sends an auditing report to the user.

- Modify: A user in the group modifies the block in the shared data by performing one of the following three operations:
- Insert: The user inserts the new block say mj into shared data. Total number of blocks in shared data is n. He/She computes the new identifier of the inserted block mj as idj={vj,rj}where idj=identifier of jth block, vj=Virtual index. For the rest of blocks, identifiers of these blocks are not changed. This user outputs the new ring signature of the inserted block with SignGen, and uploads to the cloud server. Total number of blocks in shared data increases to n+1.
- Delete: The user deletes the block mj , its identifier idj and ring signature from the cloud server. The identifiers of other blocks in shared data are remains the same. The total number of blocks in shared data in cloud decreases to n-1.
- Update: The user updates the jth block in shared data with a new block mj. The virtual index of this block remain the same and new ring signature is computed. The user computes the new identifier of the updated block. The identifiers of other blocks in shred data are not changed. The user outputs the new ring signature of new block with SignGen, and uploads to the cloud server. The total number of blocks in shared data is still n.

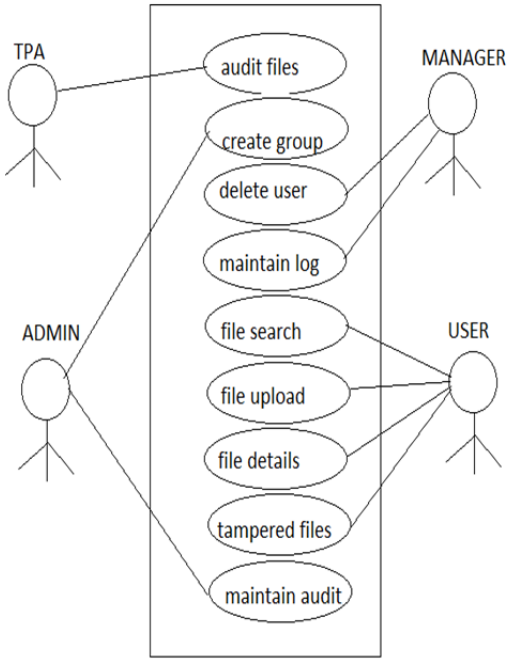


Fig: Use case diagram

Dynamic operation for group users:

The dynamicity can be achieved for number of users in the group.

The group member can be added dynamically as well as any user can be revoked from the group

Algorithm:

The RSA algorithm is used for key generation. RSA is one of the first practical public-key cryptosystems and is widely used for secure data transmission. In such a cryptosystem, the encryption key is public and differs from the decryption key which is kept secret. In RSA, this asymmetry is based on the practical difficulty of factoring the product of two large prime numbers, the factoring problem. RSA is made of the initial letters of the surnames of Ron Rivest, Adi Shamir, and Leonard Adleman, who first publicly described the algorithm in 1977.

RSA involves a *public key* and a *private key*. The public key can be known by everyone and is used for encrypting messages. Messages encrypted with the public key can only be decrypted in a reasonable amount of time using the private key. The keys for the RSA algorithm are generated the following way:

1. Choose two distinct prime numbers p and q.
   - For security purposes, the integers p and q should be chosen at random, and should be of similar bit-length. Prime integers can be efficiently found using a primality test.
2. Compute $n = pq$.
   - n is used as the modulus for both the public and private keys. Its length, usually expressed in bits, is the key length.
3. Compute $\varphi(n) = \varphi(p)\varphi(q) = (p - 1)(q - 1) = n - (p + q - 1)$, where φ is Euler's totient function. This value is kept private.
4. Choose an integer e such that $1 < e < \varphi(n)$ and $\gcd(e, \varphi(n)) = 1$; i.e., e and $\varphi(n)$ are coprime.
   - e is released as the public key exponent.
   - e having a short bit-length and small Hamming weight results in more efficient encryption – most commonly $2^{16} + 1 = 65{,}537$. However, much smaller values of e (such as 3) have been shown to be less secure in some settings.[8]
5. Determine d as $d \equiv e^{-1} \pmod{\varphi(n)}$; i.e., d is the modular multiplicative inverse of e (modulo $\varphi(n)$).
   - This is more clearly stated as: solve for d given $d \cdot e \equiv 1 \pmod{\varphi(n)}$
   - This is often computed using the extended Euclidean algorithm. Using the pseudocode in the *Modular integers* section, inputs a and n correspond to e and $\varphi(n)$, respectively.
   - d is kept as the private key exponent.

   The *public key* consists of the modulus n and the public (or encryption) exponent e. The *private key* consists of the modulus n and the private (or decryption) exponent d, which must be kept secret. p, q, and $\varphi(n)$ must also be kept secret because they can be used to calculate d.
   - An alternative, used by PKCS#1, is to choose d matching $de \equiv 1 \pmod{\lambda}$ with $\lambda =$

$\text{lcm}(p-1, q-1)$, where lcm is the least common multiple. Using λ instead of $\varphi(n)$ allows more choices for d. λ can also be defined using the Carmichael function, $\lambda(n)$.

Since any common factors of (p-1) and (q-1) are present in the factorisation of p*q-1,[9] it is recommended that (p-1) and (q-1) have only very small common factors, if any besides the necessary 2.[10]

Encryption

Alice transmits her public key (n, e) to Bob and keeps the private key d secret. Bob then wishes to send message M to Alice.

He first turns M into an integer m, such that $0 \leq m < n$ and $\gcd(m, n) = 1$ by using an agreed-upon reversible protocol known as a padding scheme. He then computes the ciphertext c corresponding to

$$c \equiv m^e \pmod{n}$$

This can be done efficiently, even for 500-bit numbers, using Modular exponentiation. Bob then transmits c to Alice.

Note that at least nine values of m will yield a ciphertext c equal to m,[note 1]

Decryption

Alice can recover m from c by using her private key exponent d via computing

$$m \equiv c^d \pmod{n}$$

The Advanced Encryption Standard (AES), also referenced as Rijndael (its original name), is a specification for the encryption of electronic data established by the U.S. National Institute of Standards and Technology (NIST) in 2001.The key size used for an AES cipher specifies the number of repetitions of transformation rounds that convert the input, called the plaintext, into the final output, called the cipher text. The number of cycles of repetition is as follows:

10 cycles of repetition for 128-bit keys. 12 cycles of repetition for 192-bit keys.

14 cycles of repetition for 256-bit keys.

Each round consists of several processing steps, each containing four similar but different stages, including one that depends on the encryption key itself. A set of reverse rounds are applied to transform cipher text back into the original plaintext using the same encryption key.

High-level description of the algorithm

1. KeyExpansions—round keys are derived from the cipher key using Rijndael's key schedule. AES requires a separate 128-bit round key block for each round plus one more.

2. Initial Round

1. AddRoundKeyeach byte of the state is combined with a block of the round key using bitwise xor.

3. Rounds

1. Sub Bytes—a non-linear substitution step where each byte is replaced with another according to a lookup table.

2. Shift Rows—a transposition step where the last three rows of the state are shifted cyclically a certain number of steps.

3. Mix Columns—a mixing operation which operates on the columns of the state, combining the four bytes in each column.

4. AddRoundKey

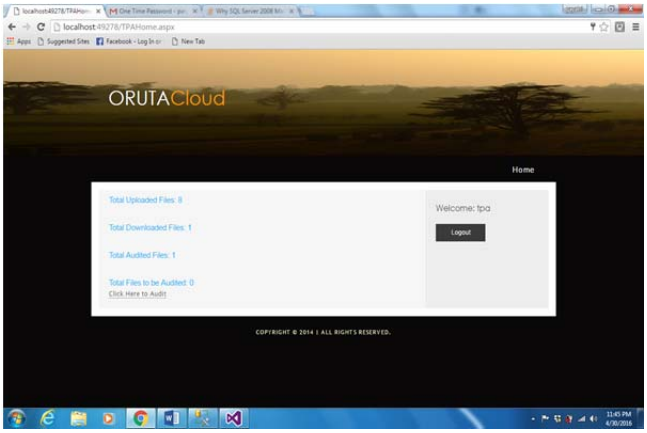
4. Final Round (no Mix Columns)

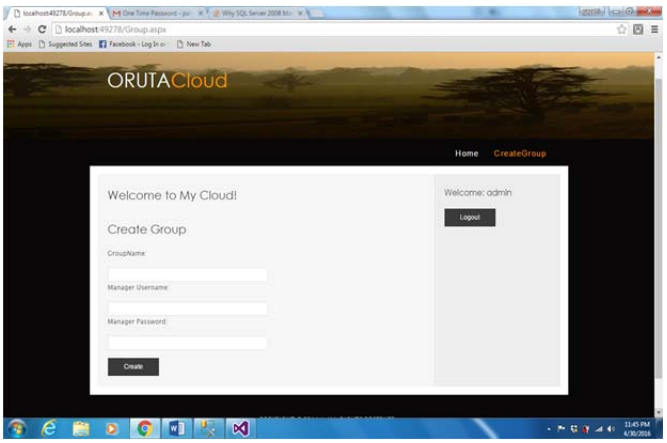
1. Sub Bytes

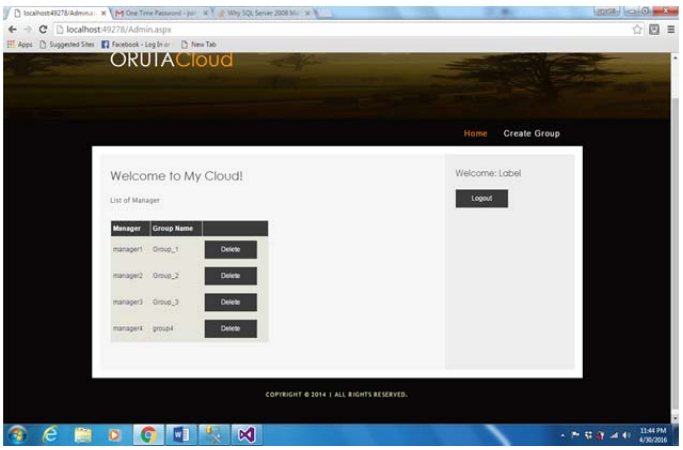
2. Shift Rows

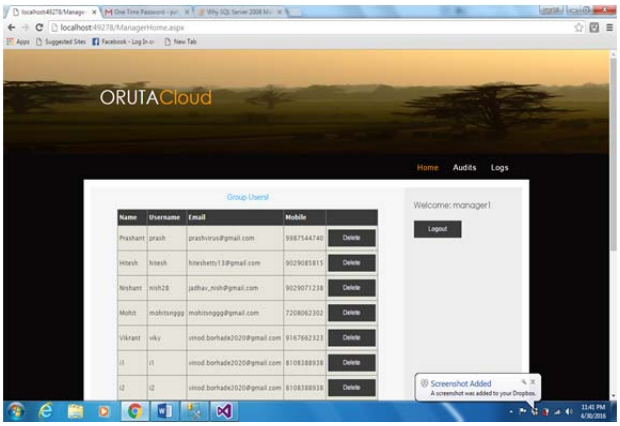
3. AddRoundKey

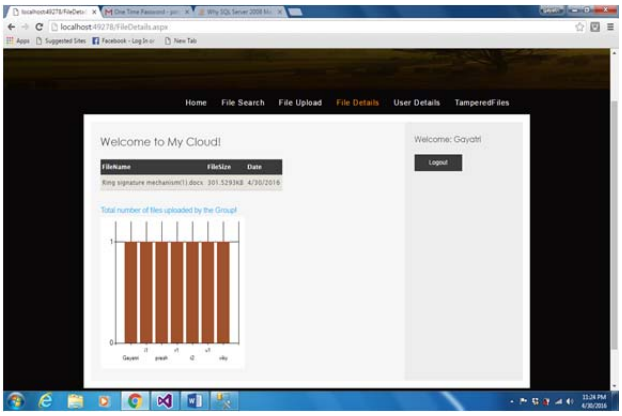
OUTPUT SCREENSHOTS:

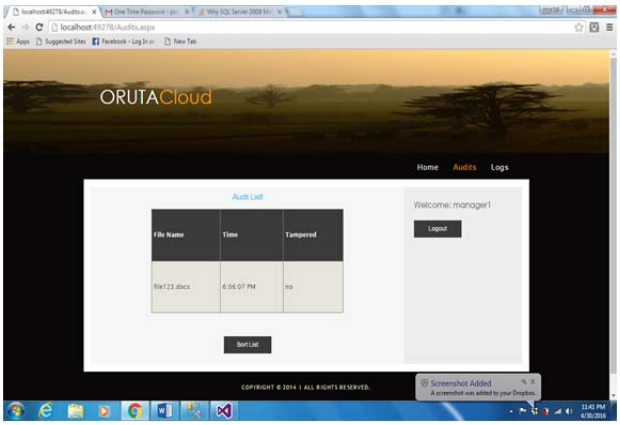


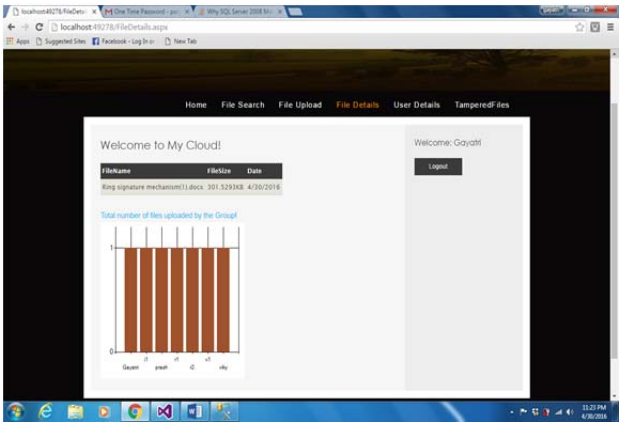


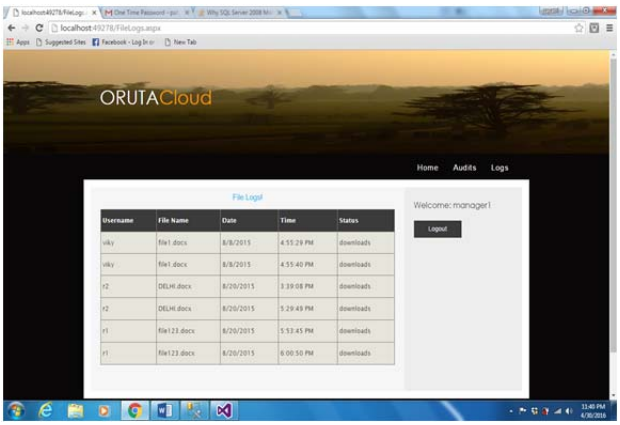


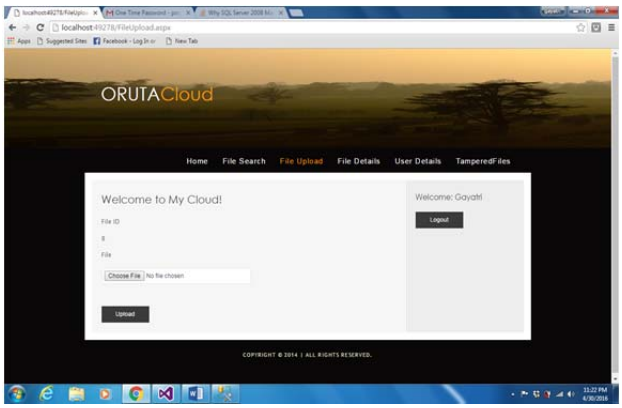

ORUTACloud
Home Audits Logs
Welcome: manager1
Logout

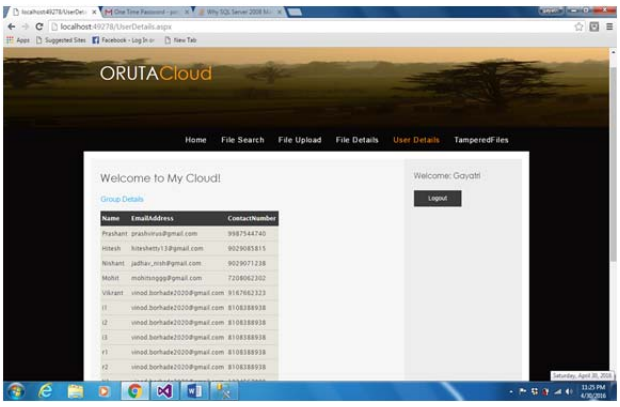

Home File Search File Upload File Details User Details TamperedFiles
Welcome to My Cloud!
Welcome: Gayatri
Logout


ORUTACloud
Home Audits Logs
Audit List
Welcome: manager1
Logout
Sort List

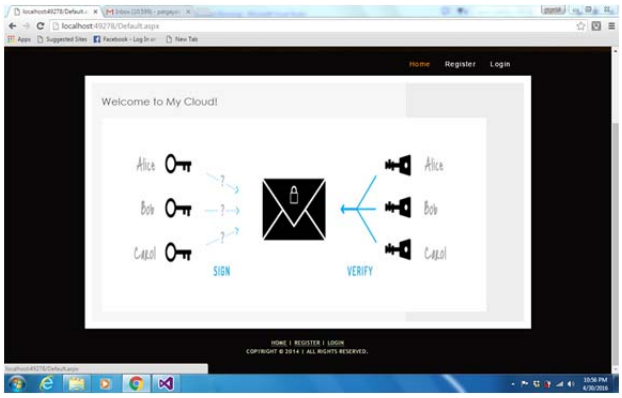

Home File Search File Upload File Details User Details TamperedFiles
Welcome to My Cloud!
Welcome: Gayatri
Logout

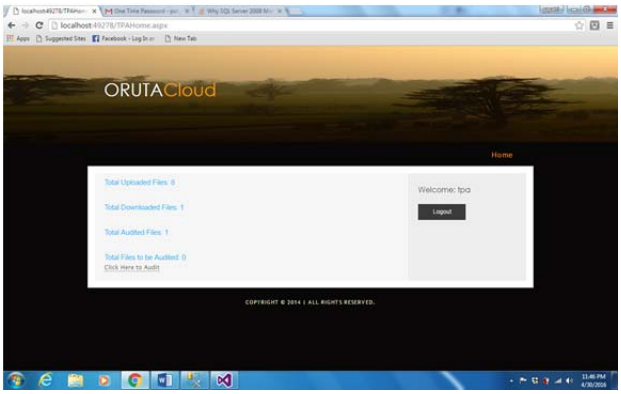

ORUTACloud
Home Audits Logs
File Logs!
Welcome: manager1
Logout


ORUTACloud
Home File Search File Upload File Details User Details TamperedFiles
Welcome to My Cloud!
Welcome: Gayatri
Logout
Upload


ORUTACloud
Home File Search File Upload File Details User Details TamperedFiles
Welcome to My Cloud!
Welcome: Gayatri
Logout


Home Register Login
Welcome to My Cloud!
Hashing algorithm
(MD5, SHA1, ...)
Signed message
Current hash-value
Compare
= or ≠
Decryption (DSA, RSA, ...)
Digital signature
Public key
Original hash-value





CONCLUSION:

We propose the system, the first privacy preserving public auditing mechanism for shared data in the cloud. We utilize ring signatures to construct homomorphic authenticators, so the TPA is able to audit the integrity of shared data, yet cannot distinguish who is the signer on each block, which can achieve identity privacy. To improve the efficiency of verification for multiple auditing tasks, we further extend our mechanism to support batch auditing. An interesting problem in our future work is how to efficiently audit the integrity of shared data with dynamic groups while still preserving the identity of the signer on each block from the third party auditor.

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.

[8] S. Yu, C. Wang, K. Ren, and W. Lou, ―Achieving Secure, Scalable, and Fine-grained Data Access Control in Cloud Computing,‖ in Proc. IEEE International Conference on Computer Communications (INFOCOM), 2010, pp. 534–542.

[9] D. Boneh, B. Lynn, and H. Shacham, ―Short Signature from the Weil Pairing,‖ in Proc. International Conference on the Theory and Application of Cryptology and Information Security (ASIACRYPT). Springer-Verlag, 2001, pp. 514–532.