ISSN (Online) : 2319 - 8753
ISSN (Print) : 2347 - 6710

International Journal of Innovative Research in Science, Engineering and Technology

An ISO 3297: 2007 Certified Organization Volume 6, Special Issue 2, February 2017

National Conference on Recent Advancements in Engineering & Technology and Management 2017 (NCRAETM 2K17)

18th February 2017

Organized by

Sree Rama Engineering College (SRET), Tirupati- 517 507, India

Anatomy of Conventional Conceal in Cloud Reckon by Source Controlled Strategy

C.Jhansi Lakshmi[1], Chejarla Raja.S[2]

M.Tech, Department of Computer Science and Engineering, Sree Rama Engineering College, Tirupati, A.P., India.[1]

Assistant Professor, Department of Computer Science and Engineering, Sree Rama Engineering College, Tirupati, A.P., India.[2]

ABSTRACT : Storing touchy information on un-depended on servers is a challenging obstacle in cloud computing. To assurance confidentiality and suitable access manipulate of outsourced touchy data, classical encryption systems are used. Nonetheless, such entry manipulate schemes are not feasible in cloud computing on account that of their lack of flexibility, scalability, and pleasant-grained entry manage and computational price at consumer side is excessive. We advise OPoR, a new cloud storage scheme involving a cloud storage server and a cloud audit server, where the latter is assumed to be semi-honest. In exact, we take into account the project of allowing the cloud audit server, on behalf of the cloud users, to pre-process the data earlier than importing to the cloud storage server and later verifying the info integrity. OPoR outsources the heavy computation of the tag new release to the cloud audit server and eliminates the involvement of user in the auditing and within the preprocessing phases. In addition, we improve the Proof of Retrievabiliy (PoR) model to support dynamic data operations, as good as be certain security in opposition to reset attacks launched through the cloud storage server within the upload segment.

KEYWORDS: cloud computing, Retrievability, public verifiability, computational burden, data integrity.

I. INTRODUCTION

Cloud computing is becoming ubiquitous as it presents fast and efficient on-demand services for storage, community, hardware, and application by way of the internet. Cloud computing offers new amenities to enterprises, companies, and most people, and provides lowcost computing infrastructure for IT-based solutions. Cloud Computing has been predicted as the following generation structure of the IT enterprise due to its long list of remarkable advantages: on-demand selfservice, ubiquitous community access, area-independent resource pooling, speedy useful resource elasticity, and usagebased pricing. In designated, the ever more cost effective and extra strong processors, together with the program as a carrier (SaaS) computing structure, are transforming knowledge centers into swimming pools of computing provider on a big scale. Even though having attractive benefits as a promising carrier platform for the web, this new data storage paradigm in Cloud brings many difficult problems which have profound influence on the usability reliability, scalability, security, and efficiency of the overall process. Some of the largest considerations with remote information storage is that of data integrity verification at untrusted servers. For illustration, the storage carrier supplier may just decide to hide such information loss incidents because the Byzantine failure from the customers to maintain a repute. What's extra severe is that for saving cash and cupboard space the provider supplier might deliberately discard hardly ever accessed data records which belong to an normal consumer. Due to the fact the big measurement of the outsourced electronic knowledge and the customers constrained useful resource ability, the core of the main issue can be generalized as how can the consumer in finding an effective option to participate in periodical integrity verification with out the nearby replica of data records.

ISSN (Online) : 2319 - 8753
ISSN (Print) : 2347 - 6710

International Journal of Innovative Research in Science, Engineering and Technology

An ISO 3297: 2007 Certified Organization Volume 6, Special Issue 2, February 2017

National Conference on Recent Advancements in Engineering & Technology and Management 2017 (NCRAETM 2K17)

18th February 2017

Organized by

Sree Rama Engineering College (SRET), Tirupati- 517 507, India

II. SYSTEM MODEL

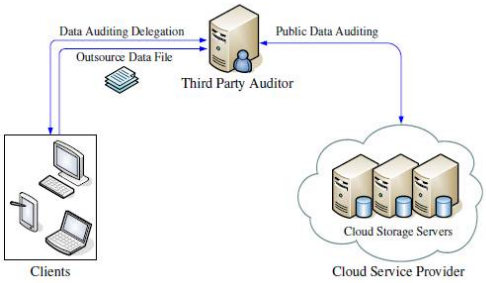

Fig. 1: Cloud data storage architecture

Three unique network entities can also be identified as follows:

- customer: an entity that has massive knowledge documents to be stored within the cloud and relies on the cloud for data upkeep and computation, can be either person purchasers or firms.
- Cloud Storage Server (CSS): an entity, which is managed by way of Cloud provider provider (CSP), has enormous storage space and computation useful resource to maintain patrons knowledge. The CSS is required to furnish integrity proof to the purchasers or cloud audit server during the integrity checking phase.
- Cloud Audit Server (CAS): a TPA, which has skills and capabilities that purchasers would not have, is relied on to examine and expose hazard of cloud storage offerings on behalf of the purchasers upon request. In this system, the cloud audit server additionally generates all the tags of the documents for the customers before uploading to the cloud storage server.

In the cloud paradigm, through placing the enormous information records on the remote servers, the clients will also be relieved of the burden of storage and computation. As clients not possess their information in the neighborhood, it is of central significance for the consumers to ensure that their information are being properly stored and maintained. That's, clients will have to be prepared with particular protection way so that they can periodically affirm the correctness of the far off data even without the existence of local copies. In case that clients don't necessarily have the time, feasibility or resources to observe their information, they can delegate the monitoring venture to a trusted cloud audit server of their respective picks.

III. CONSTRUCTION OF POR SCHEMES WITH PUBLIC VERIFIABILITY AND DYNAMIC DATA OPERATION SUPPORT

We begin with some notations and definitions of our scheme, adopted by means of the construction details and discussion of dynamic data operation support.

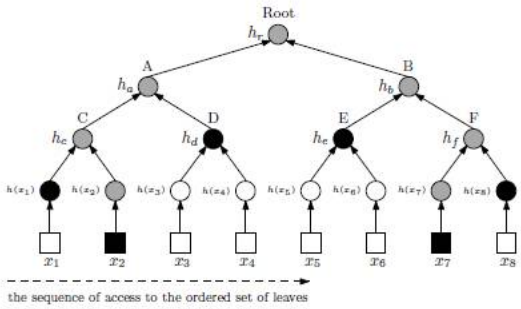

Fig. 2: Merkle hash tree authentication of data elements. The access sequence of leaf nodes h(x1), . . . , h(xn) is defined as the search order from left to right with depth first priority.

Bilinear Map:

A bilinear map is a map e **:** G×G → GT ,

where G is a hole Diffie-Hellman (GDH) group and GT is one more multiplicative cyclic workforce of prime order p with the following homes: (i) Computable: there exists an efficiently computable algorithm for computing e; (ii) Bilinear: for all h1, h2 G and a, b Zp , e(ha 1 , hb2) = e(h1,h2)ab; (iii) Non-degenerate: e(g, g)≠ 1, where g is a generator of G.

ISSN (Online) : 2319 - 8753
ISSN (Print) : 2347 - 6710

International Journal of Innovative Research in Science, Engineering and Technology

An ISO 3297: 2007 Certified Organization Volume 6, Special Issue 2, February 2017

National Conference on Recent Advancements in Engineering & Technology and Management 2017 (NCRAETM 2K17)

18th February 2017

Organized by

Sree Rama Engineering College (SRET), Tirupati- 517 507, India

Merkle Hash Tree:

A Merkle Hash Tree (MHT) is a good-studied authentication constitution , which is meant to efficaciously and securely show that a collection of elements are undamaged and unaltered. It's developed as a binary tree the place the leaves within the MHT are the hashes of respectable knowledge values. Figure 2 depicts an instance of authentication. The verifier with the professional hr requests for {x2, x7 and requires the authentication of the obtained blocks. The prover presents the verifier with the auxiliary authentication understanding (AAI)Ω2 =< h(x1), hd > and Ω7 =< h(x8),
he >. The verifier can then affirm x2 and x7 by first computing h(x2), h(x7),
hc =h(h(x1)||h(x2)), hf = h(h(x7)||h(x8)),
ha = h(hc||hd), hb = h(he||hf) and hr = h(ha||hb), and then checking if the calculated hr is the same as the authentic one. MHT is frequently used to authenticate the values of knowledge blocks. Nevertheless, in this paper we additional employ MHT to authenticate both the values and the positions of data blocks. By the way of computing the basis in MHT, the leaf node positions can be targeted decided by way of the left-to-right and depth-first sequence.

IV. THE CORE CONSTRUCTION

Now we start to present the major suggestion at the back of our scheme. As within the prior PoR techniques, we assume the patron encodes the raw information file e F into F making use of some rate-ρ error correcting codes, e.G. Reed- Solomon codes. To additional diminish the computation load of the consumer, we can require that e F is pre-processed by way of the cloud audit server. The encoded file F is divided into n blocks M1, . . . ,Mn, and every block has s sectors, i.e.
Mi = (Mi1,Mi2, · · · ,Mis), where Mij ∈ Zp fori = 1, . . . , n, j = 1, . . . , s and p is a big top.Let e : G × G → GT be a bilinear map, with three cryptographic hash capabilities
H, h : {0, 1}∗ → G and f : {0, 1}∗ → Zp, viewed as random oracles. Let g be the generator of G. The method of our protocol execution is as follows:
Setup: Setup: The cloud audit server chooses a random α ← Zp, u1, u2, · · · , us ← G, and computes v ← g^{α}. The secret key is sk = (α) and the general public key is pk = (v, {uj}1≤j≤s).
Upload (Phase 1: Client→ Cloud Audit Server):

The patron uploads F = (M1, . . . ,Mn) to the cloud audit server. Given the file F, the cloud audit server generates a root R situated on the development of Merkle Hash Tree (MHT), where the go away nodes of the tree are an ordered set of hashes of file blocks H(Mi) (i = 1, . . . , n). Next, he signs the foundation R underneath his exclusive key α as h(R)_ ← sigsk(R). The file tag t = sigsk(R) is shipped back to the purchaser as a receipt.
(Phase 2: Cloud Audit Server→ Cloud Storage Server):
The homomorphic authenticators along side metadata are produced as follows: for every block Mi = (Mi1,Mi2, · · · ,Mis), the cloud audit server computes a signature σi as

$$\sigma_i \leftarrow \left(H(M_i) \cdot \prod_{j=1}^{s} u_j^{M_{ij}} \right)^{\alpha} . \quad (1)$$

Denote the set of signatures by means of
Φ = {σi}1≤i≤n. The cloud audit server sends F∗ = {F,Φ} to the cloud storage server. to the cloud storage server. Then, the cloud audit server keeps the receipt t and deletes F∗ from its local storage.

Integrity Verification:
Both the consumer or the cloud audit server can confirm the integrity of the outsourced information with the aid of challenging the cloud storage server. To generate the assignment query, the cloud audit server (verifier) picks

ISSN (Online) : 2319 - 8753
ISSN (Print) : 2347 - 6710

International Journal of Innovative Research in Science, Engineering and Technology

An ISO 3297: 2007 Certified Organization Volume 6, Special Issue 2, February 2017

National Conference on Recent Advancements in Engineering & Technology and Management 2017 (NCRAETM 2K17)

18th February 2017

Organized by

Sree Rama Engineering College (SRET), Tirupati- 517 507, India

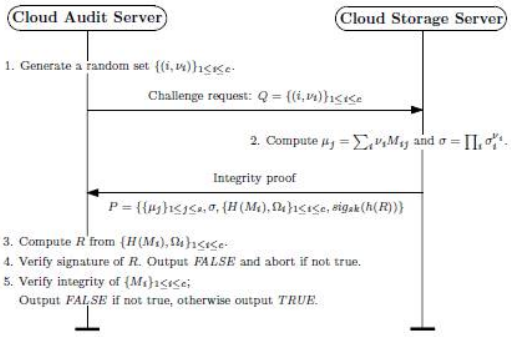

Fig. 3: Protocols for Integrity Verification

a random c-aspect subset I of set [1, n] that denote the positions of the blocks to be checked. For every i ∈I, picks a random detail vi ← f(t, i, τ), denotes the time of question. Let Q be the set {(i, vi)}, which is sent to the cloud storage server. Upon receiving the undertaking question Q = {(i, vi)}1≤i≤c, the cloud storage server computes

$$\mu_j = \sum_{\{(i,\nu_i)\} \in Q} \nu_i M_{ij} \in \mathbb{Z}_p \qquad (2)$$

for $j = 1, \ldots, s$, and

$$\sigma = \prod_{\{(i,\nu_i)\} \in Q} \sigma_i^{\nu_i} \in G. \qquad (3)$$

moreover, the cloud storage server will also provide the cloud audit server with a small amount of auxiliary information. The auxiliary values are the nodes siblings to the nodes {H(Mi)}1≤i≤c to the foundation R. Let {Ωi}1≤i≤c denote the auxiliary knowledge, the the cloud storage server responds the cloud audit server with proof P = {{μj}1≤j≤s, σ, {H(Mi),Ωi}1≤i≤c}. Upon receiving the responses from the cloud storage server, the cloud audit server performs the next computations: (1) generates root R making use of {H(Mi),Ωi}1≤i≤c and tests the consistency; (2) checks if e(t, g) = e(h(R), v). (3) checks whether

$$e(\sigma, g) = e\left(\prod_{\{(i,\nu_i)\} \in Q} H(M_i)^{\nu_i} \cdot \prod_{j=1}^{s} u_j^{\mu_j}, v \right) \quad (4)$$

If all the checking holds, output authentic; or else, output FALSE. The entire protocol methods are illustrated in Fig. 3.

Data Modification:

Consider a patron intends to switch the i-th block Mi to M′ i , then the following techniques must been carried out:

1) The purchaser sends an update request message "update = (M, i,M′$_i$)" to the cloud audit server, the place M denotes the change operation.

2) Upon receiving the request, the cloud audit server generates the corresponding signature $\sigma_i' = \left(H(M_i') \cdot \prod_{j=1}^{s} u_j^{M_{ij}'} \right)^{\alpha}$, and sends update′ = (update, σ′$_i$) to the storage server.

3) Upon receiving update′, the storage server performs the following operations.

– He replaces the block Mi with M′$_i$ and outputs F′.

– Replaces the σi with σ′ i and outputs Φ′.

– Replaces H(Mi) with H(M′ $_i$) in the Merkle hash tree construction and generates the new root R′.

– For the modification operation, replies the client with a proof Pupdate =(Ωi,H(Mi),R′), where Ωi is the AAI of Mi.

4) After receiving the proof Pupdate from the storage server, the cloud audit server operates as follows.

– He generates root R using {Ωi,H(Mi)}.

– Authenticates R by checking if e(t, g) =e(h(R), v).

– Computes the new root value ˆR using {Ωi,H(M′$_i$)} and checks if ˆR = R′.

ISSN (Online) : 2319 - 8753
ISSN (Print) : 2347 - 6710

International Journal of Innovative Research in Science, Engineering and Technology

An ISO 3297: 2007 Certified Organization Volume 6, Special Issue 2, February 2017

National Conference on Recent Advancements in Engineering & Technology and Management 2017 (NCRAETM 2K17)

18th February 2017

Organized by

Sree Rama Engineering College (SRET), Tirupati- 517 507, India

– Signs the new root metadata R′ by
t′ = sigsk(R′) and sends it to the server for storage.

• **Data Insertion**:

Consider the information proprietor needs to insert block M after the i-th block Mi The protocol procedures are just like the information change case. 1) After receiving the proof for insert operation from the storage server, the purchaser first generates root R using {Ωi,H(Mi)} and authenticates R by checking if e(t, g) = e(h(R), v).

2) If it isn't proper, output FALSE, or else the customer can now verify whether the server has participate in the insertion as required or now not, with the aid of further computing the new root value making use of {Ωi,H(H(Mi) //H(M*))} and comparing it with R′.

3) If now not, output FALSE, otherwise output proper.

4) The cloud auditor server indicators the new root metadata R′ by sigsk(R′)and sends it to the server for storage.

DATA DELETION:

information deletion is solely the reverse operation of information insertion. For single block deletion, it refers to deleting the exact block and moving all the latter blocks one block ahead. Consider the server receives the replace request of deleting block Mi, it will delete Mi from its cupboard space, delete the leaf node H(Mi) in the MHT and generate the new root metadata R′.The small print of the protocol methods are much like those of data modification and insertion, which might be therefore omitted right here.

V. SECURITY ANALYSIS

A. SECURITY MODELING

We analyze the security of our scheme under a variant of Shacham and Waters' PoR model which supports public verifiability and dynamic update operations. Besides, our model offers strengthened security by allowing a malicious storage server to perform reset attack against the client (the cloud audit server) in upload phase. The basic goal of PoR model is to achieve proof of retrievability. Informally, this property ensures that if an adversary can generate valid integrity proofs of any file F for a non-negligible fraction of challenges, we can construct a PPT machine to extract F with overwhelming probability. It is formally defined by the following game between a challenger C and an adversary A, where C plays the role of the audit server (the client) and A plays the role of the storage server:

• Setup Phase: The challenger C runs the Setup algorithm to generate its key pair (pk, sk), and forwards pk to the adversary A.

• Upload Phase: C initiates an empty table called Rlist.

A can adaptively query an upload oracle with reset capability as follows:

– Upload: When a query on a file F and a state index i comes, C checks if there is an entry (i, ri) in the R-list. If the answer is yes, C overwrites ri onto its random tape; otherwise, C inserts (i, ri) into R-list where
ri is the content on its random tape. Then C
runs (F* , t) ← Upload(sk, F; ri), and returns
the stored file F* and the file tag t. Here Upload(· ; ri) denotes an execution of the upload algorithm using randomness ri.

• Challenge Phase: A can adaptively make the following two kinds of oracle queries:

– IntegrityVerify: When a query on a file tagt comes, C runs the integrity verification protocol IntegrityVerify{A C(pk, t)} withA.

– Update: When a query on a file tag ˆt and a data operation request "update" comes, C runs the update protocol Update{A C(sk, ˆt, update)} with A.

• Output Phase: A outputs a file tag t and the
description of a prover Pt. We say that a prover Pt on t is β-admissible, if the following two conditions hold:

(1) t is a file tag output by a previous upload query.

(2) Pr[IntegrityVerify{Pt C(pk, t)} = 1] ≥ β.

Then we can define the soundness of PoR scheme.

Definition 1: (Proof of Retrievability) A PoR scheme is (β, γ)-sound if for any β-admissible prover Pt output by A in the above game, there exists an extractor E that can recover the original file of tag t with probability at least 1 − γ.

Since our model considers reset attack in upload phase, it provides higher security for PoR schemes. To prove this result, we show that Shacham and Waters' PoR scheme, which is proven secure in previous PoR model, is insecure in the new model.

Theorem 1: Shacham and Waters' PoR scheme is not sound in our new model.

Proof: In Shacham andWaters' PoR scheme, the tag t of an uploaded file F is computed as t0 //SSigssk(t0). Here t0 = name //n //u1 // . . . //us, where name R← Zp,n is the number of blocks in F, u1, . . . , us
R←G;

SSigssk(·) is the signing algorithm of a signature scheme, where the signing key ssk is included in sk. We construct an adversary A that wins the soundness game with non-negligible probability as follows:

• Upload Phase: A chooses two different files of n blocks, say F = {M1, . . . ,Mn} and ˆ F ={M′1, . . . ,M′n}. Then A makes an Upload queryon (F, 1), gets (F* , t), but stores t only. Finally Amakes an Upload query on (ˆ F, 1), gets and stores(ˆ F* , ˆt) honestly.

• Output Phase: A outputs t and a prover

ISSN (Online) : 2319 - 8753
ISSN (Print) : 2347 - 6710

International Journal of Innovative Research in Science, Engineering and Technology

An ISO 3297: 2007 Certified Organization Volume 6, Special Issue 2, February 2017

National Conference on Recent Advancements in Engineering & Technology and Management 2017 (NCRAETM 2K17)

18th February 2017

Organized by

Sree Rama Engineering College (SRET), Tirupati- 517 507, India

Pt =P(pk, ˆ F□, ˆt), where P is the prover algorithm ofan honest storage server.A does nothing in Setup Phase and Challenge Phase, so we skip these two phases above. Notice the following two observations:
(1) The tag generation only depends on the randomness used for choosing name, u1, . . . , us and the number of file blocks n. The uploading queries on (F, 1) and (ˆ F, 1) use the same randomness r1 (an entry (1, r1) will be inserted into R-list after querying (F, 1) to the upload oracle). Besides, F and ˆ F□ both have n blocks. Therefore, we haveˆt = t.
(2) (ˆ F□, ˆt) is the honestly stored file of ˆ F, so Pt is a valid proof of ˆ F with tag ˆt, i.e. IntegrityVerify{P(pk, ˆ F□, ˆt) C(pk, ˆt)} =1.
From (1) and (2), we have IntegrityVerify{Pt C(pk, t)} = 1 with probability β = 1, by correctness of the scheme. That is, A outputs a 1-admissible prover Pt on the file tag t. Since that all the information A stores are ˆ F□ and ˆt (where t = ˆt), only ˆ F can be extracted from A's storage. However, the original file of tag t is F ̸= ˆ F. By the knowledge of ˆ F, we can know nothing about F but that F has n blocks. Therefore, given Pt = P(pk, ˆ F□, ˆt), it is impossible to construct a (1, γ)-extractor of the file F (even when assume that the extractor has unlimited computing power) for any γ. This completes the proof.

B. SECURITY PROOFS

Definition 2: (CDH Problem) The Computational Diffie-Hellman problem is that, given g, gx, gy ∈ G1for unknown x, y ∈ Z* p, to compute gxy. We say that the (t, ϵ)-CDH assumption holds in G1 if no t-time algorithm has the non-negligible probability ϵ in solving the CDH problem.
Theorem 2: If the computational Diffie-Hellman problem is hard in bilinear groups, no adversary against the soundness of our public-verifiable PoR scheme could cause the verifier to accept an integrity proof of any file F with non-negligible probability in the random oracle model, except by responding with correctly computed values.
Proof:
The security of soundness is given by reduction. We assume that there is an adversary who can break the soundness. Another simulator will be constructed by interacting with the adversary. The simulator also answers all the queries for PoR protocol, including the tag generation and integrity proof. After the simulation, if the adversary outputs a valid tag without the help of client, the simulator breaks the assumption of computational Diffie-Hellman problem. Suppose that an adversary
outputs the description of a prover that causes the verifier to accept an integrity proof with non-negligible probability, by responding with values that are not correctly
computed. Let F = (M1, . . . ,Mn) be the file for integrity verification, Φ = {σi}1≤i≤n be the signatures of file blocks,
Q = {(i, vi)}1≤i≤c be the verification query. Denote the expected response from a honest
prover by
P = {{μj}1≤j≤s, σ, {H(Mi),Ωi}1≤i≤c}, and
denote the proof generated by the adversary be P′ ={{μ′j}1≤j≤s, σ′, {H(M′i),Ω′i}1≤i≤c} where P′ ̸= P.
First, we show that
{H(M′i),Ω′i}1≤i≤c ={H(Mi),Ωi}1≤i≤c if the hash functions H and h are collision resistent. Denote by R′ the Merkle Hash
Tree root generated from {H(M′i),Ω′i}, and denoteby R the MHT root R generated from {H(Mi),Ωi}.

Suppose $\{H(M_i'),\Omega_i'\}_{1\le i\le c} \neq \{H(M_i),\Omega_i\}_{1\le i\le c}$, there must be R ̸= R′ except with negligible probabilitydue to the collision resistance of H. The signature on R′can pass the verification, so (1) e(t, g) = e(h(R′), v).Since t = h(R)_, we have (2) e(t, g) = e(h(R), v).From (1) and (2), h(R′) = h(R), implying a collision(R,R′) of h which occurs with negligible probability.So the adversary outputs the correct hashes of fileblocks {H(Mi),Ωi}1≤i≤c in the proof P′ except withprobability 1 − (1 − 1/p)lgn+1, when H and h aremodeled as random oracles.Then we show that σ′ = σ if the computational Diffie-Hellman problem is hard. Otherwise, we construct a simulator,
that given g, ˜g = g_, h ∈ G where α is unknown,outputs h_. In the setup phase, the simulator sets v as ˜g,chooses two vectors of randomness β1, . . . , βs ∈ Zp andγ1, . . . , γs ∈ Zp, and sets uj = g_jhj for j = 1, . . . , s.
It initiates three empty hash tables H-table, h-table, ftable, and simulates the oracle queries as follows:
• H oracle: When a query of Mi comes where there exists (Mi, ri) in the H-table for some ri, returns ri. The h oracle and f oracle process such queries similarly. When a query of a fresh Mi comes, performs the following:
(1) Randomly chooses ri ∈ Zp.
(2) If there exists (M′, ri) in the H-table, go tostep (1).
(3) Inserts (Mi, ri) in the H-table and returns
H(Mi) = gri/(gΣsj=1 _jMijhΣsj=1 jMij).
• h oracle: when a oracle query of a fresh R comes,performs the following:
(1) Randomly chooses r ∈ Zp.
(2) If there exists (R′, r) in the h-table, go to step(1).
(3) Inserts (R, r) in the H-table and returns
h(R) = gr.The step (2) in the simulation of H oracle and h oracle ensures that H, h are collision resistant, and the reset attack by uploading different files using the same randomness will not work.
• f oracle: when a oracle query of a fresh ξ comes, performs the following:

ISSN (Online) : 2319 - 8753
ISSN (Print) : 2347 - 6710

International Journal of Innovative Research in Science, Engineering and Technology

An ISO 3297: 2007 Certified Organization Volume 6, Special Issue 2, February 2017

National Conference on Recent Advancements in Engineering & Technology and Management 2017 (NCRAETM 2K17)

18th February 2017

Organized by

Sree Rama Engineering College (SRET), Tirupati- 517 507, India

(1) Randomly chooses ν ∈ Zp.
(2) If there exists (ξ, ν) in the f-table, go to step(1).
(3) Inserts (ξ, ν) in the f-table, and returns
f(ξ) = ν.
• Upload Oracle: when a oracle query of F =
{M1, · · · ,Mn} and a state index i comes, sets the random tape as the challenger does in the game defined, and performs the following computation:
(1) Query Mi to H oracle for i = 1, · · · , n;
(2) Construct MHT root R from {Mi}, and query R to h oracle;
(3) Compute Φ = {σi}1≤i≤n where σi =
(H(Mi) · Πs j=1 uMij j)_ = (gri)_ = (g_)ri=
˜gri and (Mi, ri) is an entry in H-table;
(4) Compute t = h(R)_ = gr_ = ˜gr where (R, r) is an entry in h-table;
(5) Return F∗ = (F, Φ) and t.
• Integrity Verify Oracle: when a oracle query of a file tag t, starts an execution of IntegiryVerify protocol with the adversary, and performs as an honest verifier.
• Update Oracle: when a oracle query of a file tag t for an operation "update" comes, the simulation is similar to Upload oracle. For instance, when update = (M, i,M′
i), the simulator computes
σ′ i and produces the new file tag t′ similarly as answering an Upload query. Eventually the adversary outputs σ′ ≠σ, such that:

$$e(\sigma', g) = e\left(\prod_{\{(i,\nu_i)\}\in Q} H(M_i)^{\nu_i} \cdot \prod_{j=1}^{s} u_j^{\mu'_j}, v\right). \quad (5)$$

Combining with the equation 4, we have:

$$\begin{aligned} e(\sigma'/\sigma, g) &= e\left(\prod_{j=1}^{s} u_j^{\mu'_j-\mu_j}, v\right) \\ &= e\left(\prod_{j=1}^{s} (g^{\beta_j} h^{\gamma_j})^{\mu'_j-\mu_j}, v\right). \quad (6) \end{aligned}$$

We denote $\Delta\mu_j = \mu'_j - \mu_j$ where at least one $\Delta\mu_j \neq 0$, then above equation yields:

$$e(h, v)^{\sum_{j=1}^{s}\gamma_j\Delta\mu_j} = e(\sigma'\sigma^{-1}v^{-\sum_{j=1}^{s}\beta_j\Delta\mu_j}, g). \quad (7)$$

The simulator computes $h^{\alpha} = (\sigma'\sigma^{-1}v^{-\sum_{j=1}^{s}\beta_j\Delta\mu_j})^{\frac{1}{\sum_{j=1}^{s}\gamma_j\Delta\mu_j}}$ as the solution to the given CDH instance. Since that at least one $\Delta\mu_j \neq 0$ and the random values γ_j are information theoretically hiding, the probability that $\sum_{j=1}^{s}\gamma_j\Delta\mu_j = 0$ is about $1/p$.

Now assume that σ′ = σ in the proof P′, we show that{μ′j} = {μj} for j = 1, . . . , s if the discrete logarithmproblem is hard. Otherwise, we construct a simulator,
that given g, h ∈ G, outputs x such that h = gx, by acting as the verifier to play with the adversary. The simulation for the oracle queries is similar as above except that v = g_ is honestly generated in the Setup phase, such that the simulator can produce signatures of file blocks by itself with the knowledge of α to answer an Upload query.
Eventually the adversary outputs {μ′j} for j = 1, . . . , s, such that at least one of them is not equal to the corresponding value in {μj}. Since that e(σ′, g) =e(σ, g), the following equation holds:

ISSN (Online) : 2319 - 8753
ISSN (Print) : 2347 - 6710

International Journal of Innovative Research in Science, Engineering and Technology

An ISO 3297: 2007 Certified Organization Volume 6, Special Issue 2, February 2017

National Conference on Recent Advancements in Engineering & Technology and Management 2017 (NCRAETM 2K17)

18th February 2017

Organized by

Sree Rama Engineering College (SRET), Tirupati- 517 507, India

$$e\left(\prod_{\{(i,\nu_i)\}\in Q} H(M_i)^{\nu_i} \cdot \prod_{j=1}^{s} u_j^{\mu'_j}, v\right) = e\left(\prod_{\{(i,\nu_i)\}\in Q} H(M_i)^{\nu_i} \cdot \prod_{j=1}^{s} u_j^{\mu_j}, v\right). \quad (8)$$

So we have $\prod_{j=1}^{s} u_j^{\mu'_j} = \prod_{j=1}^{s} u_j^{\mu_j}$, therefore:

$$\prod_{j=1}^{s} u_j^{\Delta\mu_j} = \prod_{j=1}^{s} (g^{\beta_j} h^{\gamma_j})^{\Delta\mu_j} = g^{\sum_{j=1}^{s} \beta_j \Delta\mu_j} h^{\sum_{j=1}^{s} \gamma_j \Delta\mu_j} = 1. \quad (9)$$

The simulator computes $x = -\frac{\sum_{j=1}^{s} \beta_j \Delta\mu_j}{\sum_{j=1}^{s} \gamma_j \Delta\mu_j}$ as the solution to the given DL instance. The simulation fails when $\sum_{j=1}^{s} \gamma_j \Delta\mu_j = 0$ with probability about $1/p$.

In conclusion, if the proof P′ ≠ P causes the verifier to accept, we can either break the computational Diffie- Hellman assumption, or the discrete logarithm assumption, or collision resistance of the hash functions H or h. Notice that the hardness of discrete logarithm assumption is implied by the computational Diffie-Hellman assumption, and H, h are modeled as random oracles that are inherently collision resistant. Hence we complete the proof for this theorem in the random oracle model based on the computational Diffie Hellman assumption.

Theorem 1 ensures that all the provers winning the adversarial game are well-behaved, i.e. any valid proof must be exactly the same as the proof computed by an honest storage server except with a negligible probability.

Theorem 3: For any well behaved β-admissible prover Pt on a n-block file of tag t, there exists an extractor that can recover the original file in time

O(n2(s + 1) + (1 + βn2)n/ω) by running O(n/ω)

interactions with Pt, where

ω = β−1/p−(ρn/n−c+1)c.

Following Theorem 1, we can construct an extractor to interact with the prover Pt and get correctly computed proofs for β-fraction of the verification queries in the query space Zp. Applying combinatorial techniques, a

ρ-fraction of encoded file blocks are retrievable after at most O(n/ω) interactions. Then these file blocks suffice for recovering the original file using the decoding procedure of rate-ρ error correcting codes. Since that our modeling for integrity verification protocol is the same as in Shacham and Waters' PoR model, the simulation for extracting the original file is similar to that in [3], so we omit the details of the proof here. The following theorem is straightly forward combining

Theorem 1 and Theorem 2.

Theorem 4: The proposed PoR scheme is

(β, γ)- sound for any β-admissible prover Pt output by A in the soundness game, where

γ = 1 − (1 − 1/p)lgn+1 + 1/p.

VI. PERFORMANCE ANALYSIS

On this section, we will provide an intensive experimental analysis of the development proposed. We build our testbed via making use of 64-bit M2 excessive-memory quadruple extra huge Linux servers in Amazon EC2 platform because the auditing server and storage server, and a Linux desktop with Intel(R) Core(TM)2 Duo CPU clocked at 2.40 GHz and 2 GB of system memory as the user.

In order to achieve λ = 80 bit security, the prime order p of the GDH workforce G of the bilinear mapping should be a hundred and sixty bits in length. Word that in all of the reviews, the corporations G and GT are chosen in 160-bit and 512- bit length respectively. Think there is a 4 GB file with block measurement four KB, then it has n = 1000000 blocks and s = 25 sectors each block. When it is uploaded onto the storage server, the set of signatures on the file blocks only requires for yet another storage of 20 MB for information integrity verification.

Additionally, we assessment the conversation fee for each and every users communication cost in Amazon EC2 cloud atmosphere, which is 92 ms. Note that such an overhead involves the time consuming for transmission and authentication at Amazon EC2 cloud platform.

ISSN (Online) : 2319 - 8753
ISSN (Print) : 2347 - 6710

International Journal of Innovative Research in Science, Engineering and Technology

An ISO 3297: 2007 Certified Organization Volume 6, Special Issue 2, February 2017

National Conference on Recent Advancements in Engineering & Technology and Management 2017 (NCRAETM 2K17)

18th February 2017

Organized by

Sree Rama Engineering College (SRET), Tirupati- 517 507, India

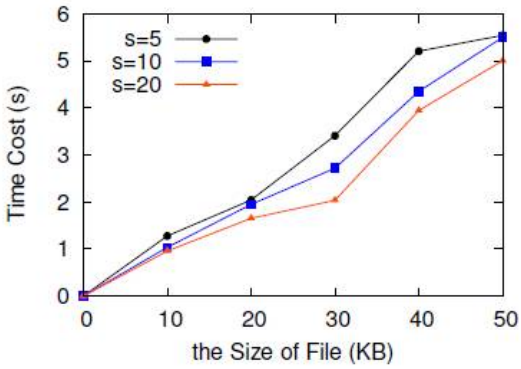

Fig. 4: Tag generation time

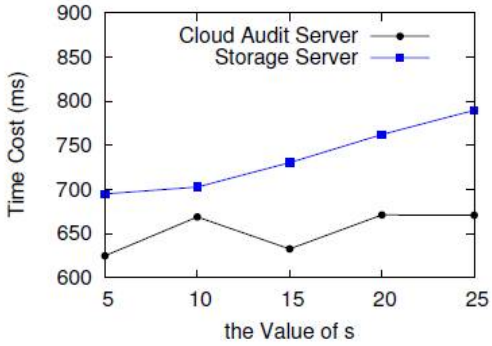

Fig. 5: Verification time

Within the first experiment, the computational overhead for the tag iteration of records on the cloud audit server is evaluated. We now have no longer checked the computational overhead at customers considering it most effective desires the computation of a digital signature, which may be very small when put next with the computation of the tags. The cause is that probably the most overhead computation has been delivered to the cloud audit server. Three distinct numbers of s are chosen within the scan to show the outcome on the efficiency of the time price. From Fig. 4, we can see that the time rate grows when the quantity of s decreases. The natural time price for file with dimension 50KB is 5s. When put next with the previous related work, the computational overhead at users in is outsourced to the cloud audit server. The response time at the person facet together with the time price of uploading documents, tag new release on the audit server, building of Merkle-hash tree, the conversation rate and signature generation. To assess the response time, the fee of importing file to the cloud audit server is confirmed. For file with 10MB, the ordinary time price is 25s. The time cost of development for the Merklehash tree is 27s for the file with dimension 10MB. The signature new release for the basis is 3ms, which can be disregarded in comparison with the time fee of importing and development of Merkle-hash tree. Notice that the time rate of uploading documents are not able to be kept away from in any applications. Although the tag iteration price can be close to the time rate of importing records, the cloud audit server can system these tag generation for the period of the file importing. Thus, the further time for the response is very small. That is perfect for the users due to the fact the time cost would be double at the user facet if the tag is computed by using the customers. We additional evaluate the efficiency for verification at each cloud audit server and cloud storage server in a scalable process in Fig. 5. Obviously, as the development of the quantity of s in procedure, the time cost for response value at cloud storage server is growing. This is given that it desires to compute the entire exponentiations for each and every block in a tag. Whereas, such rate at cloud audit server is just about consistent (practically 650 ms) due to the fact that 460 blocks are ample for the integrity verification it doesn't matter what is the size of file to be checked.

VII. CONCLUSION

This paper proposes OPoR, a new proof of retrievability for cloud storage, in which a trustworthy audit server is introduced to preprocess and upload the information on behalf of the customers. In OPoR, the computation overhead for tag iteration on the patron part is reduced tremendously. The cloud audit server additionally performs the data integrity verification or updating the outsourced

ISSN (Online) : 2319 - 8753
ISSN (Print) : 2347 - 6710

International Journal of Innovative Research in Science, Engineering and Technology

An ISO 3297: 2007 Certified Organization Volume 6, Special Issue 2, February 2017

National Conference on Recent Advancements in Engineering & Technology and Management 2017 (NCRAETM 2K17)

18th February 2017

Organized by

Sree Rama Engineering College (SRET), Tirupati- 517 507, India

information upon the customers request. Besides, we construct another new PoR scheme tested relaxed below a PoR mannequin with stronger safety in opposition to reset assault within the upload phase. The scheme also supports public verifiability and dynamic knowledge operation concurrently.

REFERENCES

[1] G. Ateniese, R. Burns, R. Curtmola, J. Herring, L. Kissner, Z. Peterson, and D. Song, "Provable data possession at untrusted stores," in CCS '07: Proceedings of the 14th ACM conference on Computer and communications security. New York, NY, USA: ACM, 2007, pp. 598–609.
[2] A. Juels and B. S. K. Jr., "Pors: proofs of retrievability for large files," in CCS '07: Proceedings of the 14th ACM conference on Computer and communications security. New York, NY, USA: ACM, 2007, pp. 584–597.
[3] H. Shacham and B. Waters, "Compact proofs of retrievability," in ASIACRYPT '08: Proceedings of the 14th International Conference on the Theory and Application of Cryptology and Information Security. Berlin, Heidelberg: Springer-Verlag, 2008, pp. 90–107.
[4] K. D. Bowers, A. Juels, and A. Oprea, "Proofs of retrievability: theory and implementation," in Proceedings of CCSW 2009. ACM, 2009, pp. 43–54.
[5] M. Naor and G. N. Rothblum, "The complexity of online memory checking," J. ACM, vol. 56, no. 1, pp. 2:1–2:46, Feb. 2009. [Online]. Available: http://doi.acm.org/10.1145/1462153.
1462155
[6] E.-C. Chang and J. Xu, "Remote integrity check with dishonest storage server," in Proceedings of ESORICS 2008, volume 5283 of LNCS. Springer-Verlag, 2008, pp. 223–237.
[7] M. A. Shah, R. Swaminathan, and M. Baker, "Privacy-preserving audit and extraction of digital contents," Cryptology ePrint Archive, Report 2008/186, 2008, http://eprint.iacr.org/.
[8] A. Oprea, M. K. Reiter, and K. Yang, "Space-efficient block storage integrity," in In Proc. of NDSS 2005, 2005.
[9] T. S. J. Schwarz and E. L. Miller, "Store, forget, and check: Using algebraic signatures to check remotely administered storage," in ICDCS '06: Proceedings of the 26th IEEE International Conference on Distributed Computing Systems. Washington, DC, USA: IEEE Computer Society, 2006.
[10] Q. Wang, K. Ren, S. Yu, and W. Lou, "Dependable and secure sensor data storage with dynamic integrity assurance," ACM Transactions on Sensor Networks, vol. 8, no. 1, pp. 9:1–9:24, Aug. 2011. [Online]. Available: http:
//doi.acm.org/10.1145/1993042.1993051
[11] L. V. M. Giuseppe Ateniese, Roberto Di Pietro and G. Tsudik, "Scalable and efficient provable data possession," in International Conference on Security and Privacy in Communication Networks (SecureComm 2008), 2008.
[12] C. Wang, Q. Wang, K. Ren, and W. Lou, "Privacy-preserving public auditing for data storage security in cloud computing," in INFOCOM, 2010, pp. 525–533.
[13] Y. Zhu, H. Wang, Z. Hu, G.-J. Ahn, H. Hu, and S. S. Yau, "Dynamic audit services for integrity verification of outsourced storages in clouds," in SAC, 2011, pp. 1550–1557.
[14] Q. Zheng and S. Xu, "Fair and dynamic proofs of retrievability," in CODASPY, 2011, pp. 237–248.
[15] J. Li, X. Chen, J. Li, C. Jia, J. Ma, and W. Lou, "Fine-grained access control system based on attribute-based encryption," ESORICS, 2013.
[16] J. Li and K. Kim, "Hidden attribute-based signatures without anonymity revocation," Information Sciences, vol. 180, no. 9, pp. 1681–1689, 2010.
[17] J. Li, C. Jia, J. Li, and X. Chen, "Outsourcing encryption of attribute-based encryption with mapreduce," ICICS, 2012.
[18] X. Chen, J. Li, J. Ma, Q. Tang, and W. Lou, "New algorithms of outsourcing modular exponentiations," ESORICS, pp. 541–556, 2012.
[19] Y. Zhu, H. Hu, G.-J. Ahn, and M. Yu, "Cooperative provable data possession for integrity verification in multicloud storage," IEEE Trans. Parallel Distrib. Syst., vol. 23, no. 12, pp. 2231–2244, 2012.
[20] H. Xiong, X. Zhang, D. Yao, X. Wu, and Y. Wen, "Towards endto- end secure content storage and delivery with public cloud," in CODASPY, 2012, pp. 257–266.
[21] Q. Zheng and S. Xu, "Secure and efficient proof of storage with deduplication," in CODASPY, 2012, pp. 1–12.
[22] C. Wang, Q. Wang, and K. Ren, "Ensuring data storage security in cloud computing," in Proceedings of IWQoS 2009, Charleston, South Carolina, USA, 2009.
[23] C. Erway, A. Kupcu, C. Papamanthou, and R. Tamassia, "Dynamic provable data possession," Cryptology ePrint Archive, Report 2008/432, 2008, http://eprint.iacr.org/.
[24] X. Lei, X. Liao, T. Huang, H. Li, and C. Hu, "Outsourcing large matrix inversion computation to a public cloud,," in IEEE Transactions on Cloud Computing, 2013, pp. vol. 1, no. 1.
[25] Q. Wang, C. Wang, J. Li, K. Ren, and W. Lou, "Enabling public verifiability and data dynamics for storage security in cloud computing," in ESORICS, 2009, pp. 355–370.
[26] H. Li, B. Wang, and B. Li, "Oruta: Privacy-preserving public auditing for shared data in the cloud," in IEEE Transactions on Cloud Computing, 2014, pp. vol. 2, no. 1, 43–56.
[27] J. Li, X. Tan, X. Chen, and D. S. Wong, "An efficient proof of retrievability with public auditing in cloud computing," in INCoS, 2013, pp. 93–98.
[28] D. Boneh and C. Gentry, "Aggregate and verifiably encrypted signatures from bilinear maps," in Proceedings of Eurocrypt 2003, volume 2656 of LNCS. Springer-Verlag, 2003, pp. 416– 432.
[29] D. Boneh, B. Lynn, and H. Shacham, "Short signatures from the weil pairing," in ASIACRYPT '01: Proceedings of the 7th International Conference on the Theory and Application of Cryptology and Information Security. London, UK: Springer- Verlag, 2001, pp. 514–532.
[30] R. C. Merkle, "Protocols for public key cryptosystems," in Proceedings of IEEE Symposium on Security and Privacy 1980. IEEE, 1980, pp. 122–133.
[31] Z. Liu, Y. Hu, X. Zhang, and H. Ma, "Provably secure multiproxy signature scheme with revocation in the standard model," Computer Communications, vol. 34, no. 3, pp. 494–501, 2011.
[32] T. Malkin, S. Obana, and M. Yung, "The hierarchy of key evolving signatures and a characterization of proxy signatures," in Proceedings of Eurocrypt 2004, volume 3027 of LNCS. Springer- Verlag, 2004, pp. 306–322.
[33] J. C. Schuldt, K. Matsuura, and K. G. Paterson, "Proxy signatures secure against proxy key exposure," in Proceedings of PKC 2008, volume 4939 of LNCS. Springer-Verlag, 2008, pp. 141–161.