

# **INTEGRITY AUDITING BASED ON THE KEYWORD WITH SENSITIVE INFORMATION PRIVACY FOR ENCRYPTED CLOUD DATA**

## **Guided by:**

Dr. S.Milton Ganesh, M.E., Ph.D.,  
Assistant Professor & HoD, of IT.

## **Presented by:**

1. Dhilip Kumar D (422419205008)
2. Dhina P (422419205009)
3. Yuvaraj S (422419205043)

# LITERATURE SURVEY

S.No.	Title & Author	Journal & Year	Problem	Solution	Parameters Measured	Advantages	Disadvantages
1	<b>Title :</b> Checking Only When It Is Necessary: Enabling Integrity Auditing Based on the Keyword With Sensitive Information Privacy for Encrypted Cloud Data <b>Authors :</b> Xiang Gao , Jia Yu ,Yan Chang, Huaqun Wang , and Jianxi Fan	IEEE transactions on dependable and secure computing, vol. 19, no. 6, Nov/Dec 2022	To check the integrity auditing of an encrypted file in a cloud.	RAL is a label that authenticates file relationships with queried keywords and generates auditing proof while preserving file identity privacy.	Reduce overall time complexity,computation time required for auditing.	Give security analysis that satisfies correctness, auditing soundness and sensitive information privacy.	It can only check the file that contain secure index keywords.
2	<b>Title :</b> Identity-Based Cloud Storage Auditing for Data Sharing With Sensitive Information <b>Authors :</b> Yang Yang, Yanjiao Chen, Fei Chen, Jing Chen	IEEE Internet of Things Journal ( Volume: 9, Issue: 13, 01 July 2022)	The sensitive information should not be exposed to others when the cloud file is shared.	The user hide the data blocks corresponding to original file and the signatures, and then sends them to a receiver.	The proposed scheme is evaluated for its efficiency in terms of computational overhead, communication overhead, and storage overhead.	This method not only realizes the remote data integrity auditing, but also supports the data sharing on the condition that sensitive information is protected in cloud storage.	The data owner can decrypt this file. However, it will make the whole shared file unable to be used by others.
3	<b>Title :</b> Data Integrity Auditing without Private Key Storage for Secure Cloud Storage <b>Authors :</b>  Wenting Shen; Jing Qin, Jia Yu, Rong Hao, Jiankun Hu, Jixin Ma	IEEE Transactions on Cloud Computing ( Volume: 9, Issue: 4, 01 Oct.-Dec. 2021)	The user has to keep hardware token to store his private key and memorize a password to activate this private key. If it lost the current data integrity auditing would be unable to work.	We use biometric data (e.g. iris scan, fingerprint) as private key to avoid using the hardware token.	The result of integrity auditing is more accurate.	Adds reasonable overhead to realize data integrity auditing without private key storage compared with the existing schemes.	The user might need to remember multiple passwords for different secure applications

# PROBLEM STATEMENT

Who does the problem affect?

Cloud users, Cloud service providers, Cloud auditors.

What are the boundaries of the problem?

Data sensitivity, Cloud storage, Auditing, Information privacy.

What is the issue?

Integrity of cloud data, Privacy of sensitive information, unauthorized access

When does the issue occur?

The issue may occur during the auditing of cloud data for integrity, as auditors may require access to sensitive information.

Where does the issue occur?

The issue may occur when the data is initially uploaded to the cloud.

Why is it important that we fix the problem?

Protecting the privacy of sensitive information is essential for maintaining data security, compliance, trust, and cost savings.

What solution to solve this issue?

A combination of encryption, authentication, access controls, monitoring, and audits can help to ensure the integrity of cloud data.

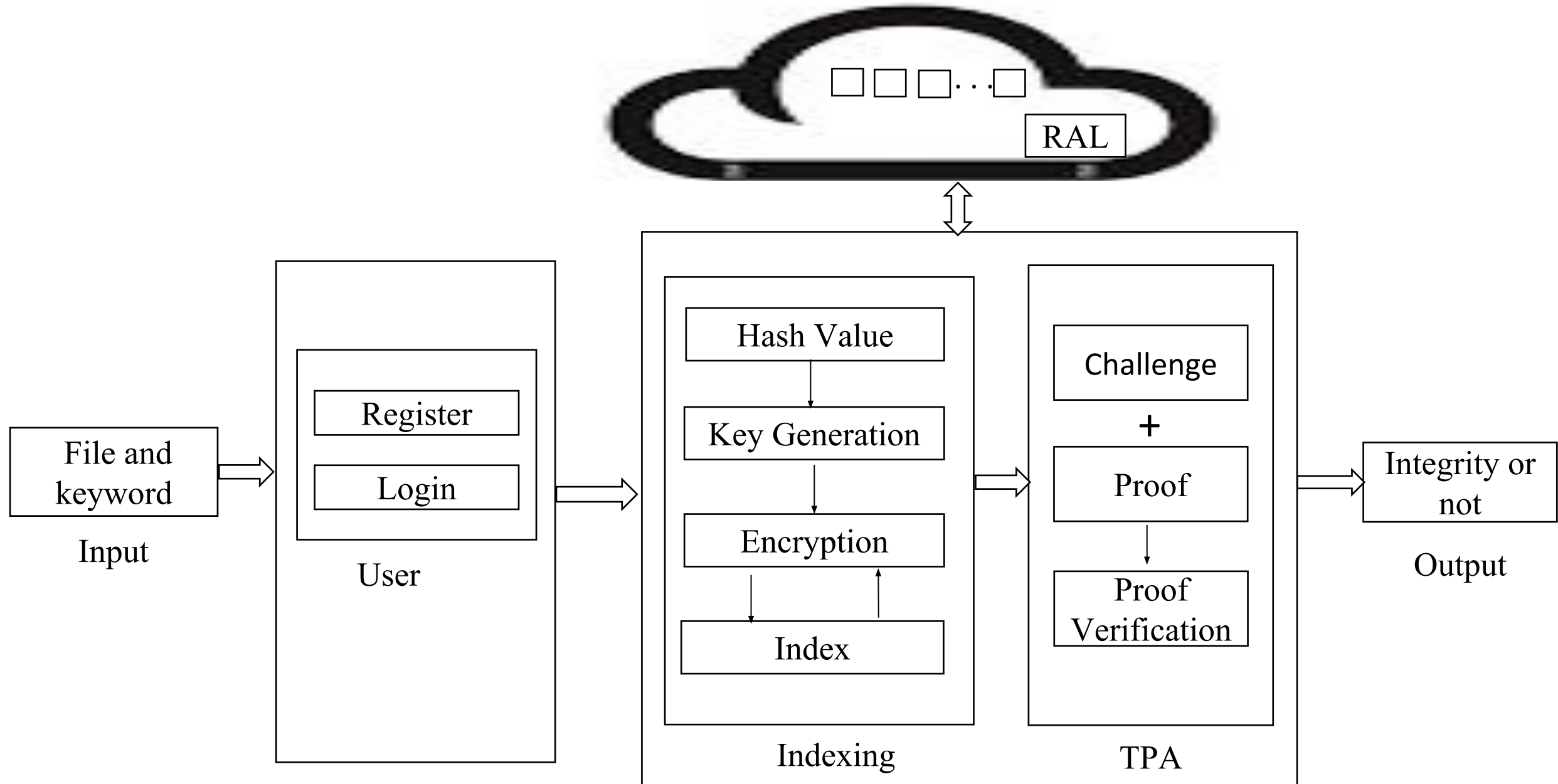
What methodology used to solve the issue?

Encryption, Keyword-based auditing, Relation Authentication Label, Auditing Proof Generation.

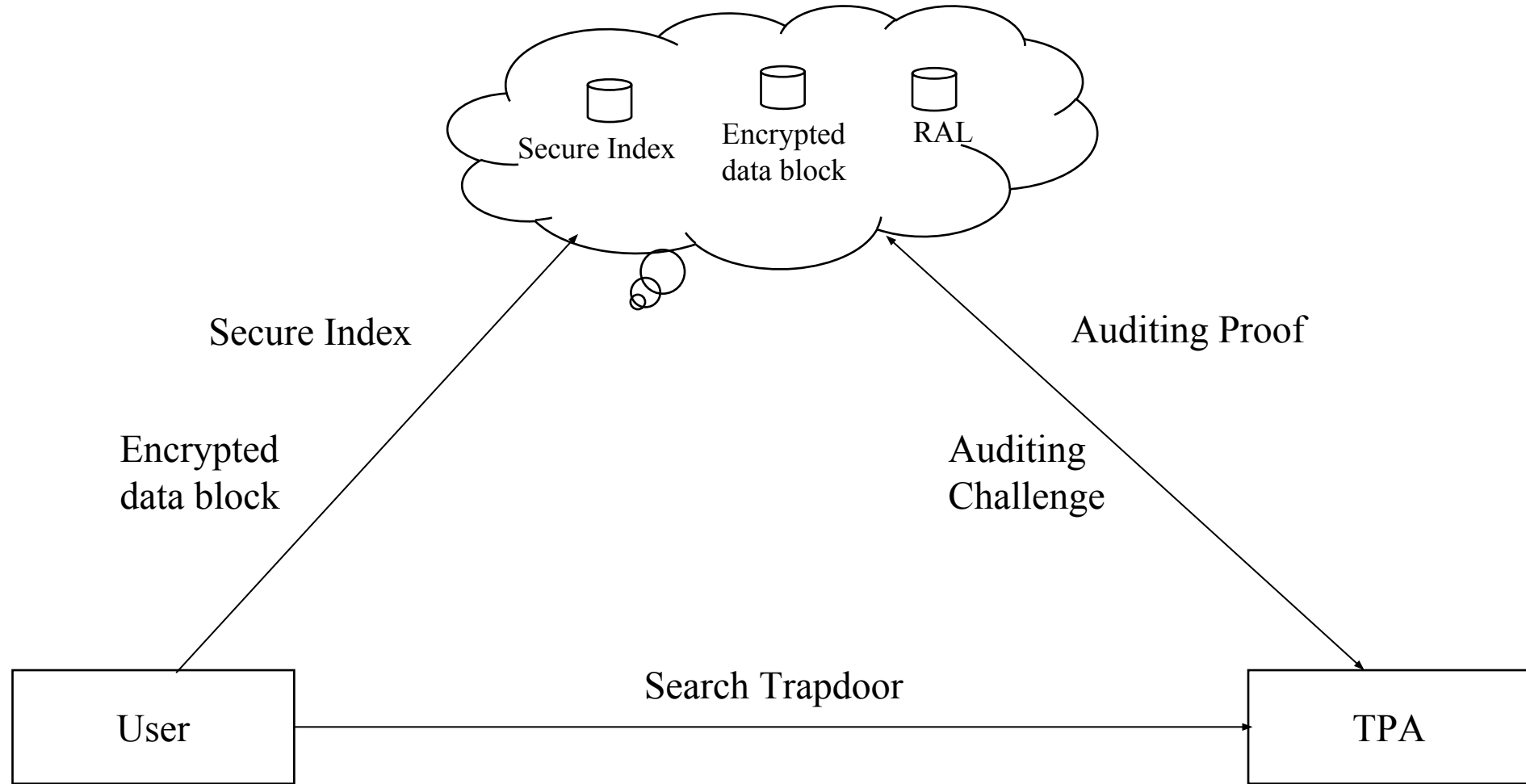
# PROPOSED SOLUTION

Parameter	Description
Problem statement	To ensure the integrity of encrypted cloud data
Idea/Solution description	Solution is to check Integrity based on the queried keyword using RAL(Relational Authentication Label).
Novelty/Uniqueness	The uniqueness is keyword-based auditing approach.
Customer satisfaction	Ensures the privacy and security of sensitive information. This can increase customer satisfaction and trust.
Business model (Revenue model)	The company could charge a fee for each auditing request from the organizations.
Scalability of the solution	It should be able to handle large datasets and a high volume of auditing requests without a significant increase in computational overhead.

# SYSTEM ARCHITECTURE



# SYSTEM MODEL



# MODULE SPLITS-UP

## REGISTRAION

**Input:** Files and keyword

**Output:** Encrypted file block and keyword set

### Algorithm:

1. The user register and creates account.
2. Then login to the account.
3. Generate the secret key  $x$  and public key  $y$ .
4. The file is splits into  $s$  blocks and then the blocks are encrypted by symmetric encryption.
5. Build the keyword set  $w_k$  from the files.

# INDEXING

- **Input:** Encrypted file block and keyword set  
**Output:** Secure index, Authenticator, Trapdoor

## Algorithm:

1. Generate three secure Hash functions  $H_1: \{0,1\}^* \rightarrow G_1$ ,  $H_2: \{0,1\}^* \rightarrow G_1$ ,  $H_3: \{0,1\}^* \rightarrow G_1$
2. Computes  $\pi(w_k)$  as the address of each row in the secure index.
3. Encrypts the index vector.  $ev_{\pi}(w_k) = vw_k \oplus f(\pi(w_k))$ .
4. Computes the RAL  $\Omega_{\pi(w_k)} = \{\Omega_{w_{k,1}}, \Omega_{w_{k,2}}, \dots, \Omega_{w_{k,s}}\}$ .
5. Computes the Secure index  $I = \{\pi(w_k), ev_{\pi(w_k)}, \Omega_{\pi(w_k)}\} k = 1, 2, \dots, m$ .
6. Computes the authenticator  $\sigma_{ij} = [H_1(ID_i || j) \cdot u^{e_{ij}}]^x$ .
7. Computes the search trapdoor as  $T_{w'} = \{\pi(w'), f(\pi(w'))\}$ .



# CLOUD

**Input:** Challenge, Secure index, Encrypted data blocks and Authenticators

**Output:** Proof

## Algorithm:

1. It stores Auditing Challenge, Secure index I, encrypted data blocks and Authenticators in cloud.

$$Chal = \left\{ T_{w'}, \{j, v_j\}_{j \in Q} \right\} \quad T_{w'} = \{\pi(w'), f(\pi(w'))\} \quad v_{wk} = ev_{\pi(w_k)} \oplus f(\pi(w_k))$$

2. Using RAL, The cloud generate Auditing Proof and it sends to TPA.

$$T = \prod_{i \in S_{w_K}} \prod_{i \in Q} \sigma_{ij}^{v_j} \cdot \prod_{j \in Q} \Omega_{w_k} j^{v_j}, \mu = \sum_{i \in S_{w_k}} \sum_{j \in Q} C_{ij} \cdot v_j.$$

# TPA(Third Party Auditor)

**Input:** Trapdoor and proof

**Output:** Challenge and proof verification

## Algorithm:

1. It takes Trapdoor as input.
2. And it generate the Auditing Challenge then TPA sends challenge to Cloud.

$$Chal = \left\{ T_{w'}, \{j, v_j\}_{j \in Q} \right\}.$$

3. TPA receives the Auditing proof from the cloud.
4. TPA verifies proof using Auditing Challenge sent by TPA and Proof generated by the cloud.

$$e(T, g) \stackrel{?}{=} e \left( \left( \prod_{j \in Q} (H_3(j) \cdot H_2(\pi(w') || j))^{v_j} \right) \cdot u^\mu, y \right)$$

if Challenge == Proof

    The file is intact

else

    The file is corrupted.

## **CODE LINK:**

<https://github.com/Dhina8801/Integrity--Auditing.git>

# REGISTER

## Register

Name:

Email:

Password:

Confirm Password:

# SYSTEM INITIALIZATION

## Integrity Auditing Based on the Keyword With Sensitive Information Privacy for Encrypted Cloud Data

### System Initialization

U is :135937237065549965603754091341461620026342011511327449027823544470067083249227  
2001641677649125315065675307617186024846479989268427046323859536155135372730,488048  
3331294021137286939863872551852163454553112013051082411175612574223586633080391762  
074184020012351576704810024736492294163172812838607042347234254081,0  
G is:649206173080627102164898984644225445989442004927323878826403405298434867581759  
8167811080245436275761124212951618108327476109344948373660727360545845824151,164308  
2028410098956113199697716389696900788555358905504364154078813188192504663594264793  
913798914682123764205611749833740623128426236454974966719532821928,0

x is :729228598578030141662714149862932118592615666024

Y is:532796992485918904718578552701598970083446037950886990537698429597209893786215  
4262175245899261070443149120749890282101388593622815648544945153303553843827,148492  
3337606696184043594373239940404565819672912298247472978792606692786634256410825041  
166412544523655933562094072083904583191689884708103025858032202149,0

### Setup

b1 is :337453837815800294406828201848924868842873950448

b2 is:269887232545931482285721176523167749760173030530

b3 is:691986362767550665046120162217673703478592788706

c1 is :642376443313883807332498049119015033565285696450

c2 is:487095086201965455489953604389957780524649153456

c3 is: 187692890643663819234079394542884846265848658927

Keyword 1 is :541241113329492272946960723993304579070415746346

Keyword 2 is :188243625919875793030691398584761639860636952433

keyword 3 is: 518093855990042210867477182952170648655890853462

System Initialization

Setup

Index generation

# INDEX GENERATION OUTPUT

Index Generation

pik0 is :307093833617915601768127651863447936352641658845  
pik1 is :530519505773101164302708710302935960904923049089  
pik2 is :197215190315890027058111182105160591506223803807

evpiwko is :4  
evpiwk1 is :5  
evpiwk2 is :3

ohm\_wk1 is :56541623980245468330616200464720403789081411959169666034711962147406621957957915914752596273925169024666436976785755255203050418674  
48775104149413589993644,58429103744602287326015453029068109246385487016911058101238875607475000458702602536629440296570155470954956763282398690  
62663871604913764572070456273942884,0

ohm\_wk2 is :17515613588791567115035770026755076625009373188039600590573147889278952726338936785948000774262607972014492289730232379009692206640  
10484823993952031970953,3596134005345495367955470840594553887674515147129933362063336359197387613815445553758249595611150634643056041336676910  
02377039195717761032478655569053350,0

ohm\_wk3 is :31891824643619845274080779002420581403613823030448171242205599934488610382973039634190255704320457285124059031484615544832997296354  
37769140862068434666369,34883409817175659002214043700058401920520330894798845659556132087632186203078380052565982990910828304368931834282538741  
82830781903547447082542930126523752,0

Indexing

Authenticator generation



# AUTHENTICATOR AND TRAPDOOR OUTPUT

**Authenticator**

sigma\_file1\_block1 is :3011139061153026875403478770077408266267842849542051051304124263704340552049313352111456265839894046252211987814993026941444112626781809113283016724940084,2662274945582620951738710096289731000640974197449217866330455708883495405006461247661981967268827809868419911323170860419205917084746322887716248036973452,0

sigma\_file1\_block2 is :589040784921649017746607479461667232051589695133144373202195867851431442391030846026443936189087703405057508912877581082490153555641904906423002524886046,2094045951689692234641719426629945072874550805813591737309331300559194149753940631383247781868789522695758301464080658675202369670499421437548217576745141,0

sigma\_file1\_block3 is :4969521037402841377432265589742556596734641937589605250736415137274274785408989718107323169000458409288066931215913083189144456133897514698982004668688569,5382896945143842281598397991920931368628047312056363368999590138835074528299411375323849906941561501889156108041792918054902218607841638343975248805102729,0

**Trapdoor**

Trapdoors are :

wpik0 is :307093833617915601768127651863447936352641658845

fpik0 is :55775095427622207636301035504698428519545230434

Authenticator

Trapdoor

Challenge generation

# CHALLENGE GENERATION AND PROOF GENERATION OUTPUT

## Challenge Generation

Challenge generation is :150463270777944276481170676165705953515624350826

604262911208618535996976747919148039621971429412

## Proof Generation

vw0 is :1

T is : 620888033280182179430221839339129650277235335021019631572537243486720746026760982710100112583302277799807823  
6237960172468225594871839362775001156428798592,2590402298823812538284190629135969220795598147313556620896671281871  
78652796317941761146541354939577424055225518151978880201223087004386188734852647948318,0

meu is : 659142028705178070505797334035775931857924776783

Challenge Generation

Proof generation

Proof Verification



# PROOF VERIFICATION OUTPUT

Proof Verification

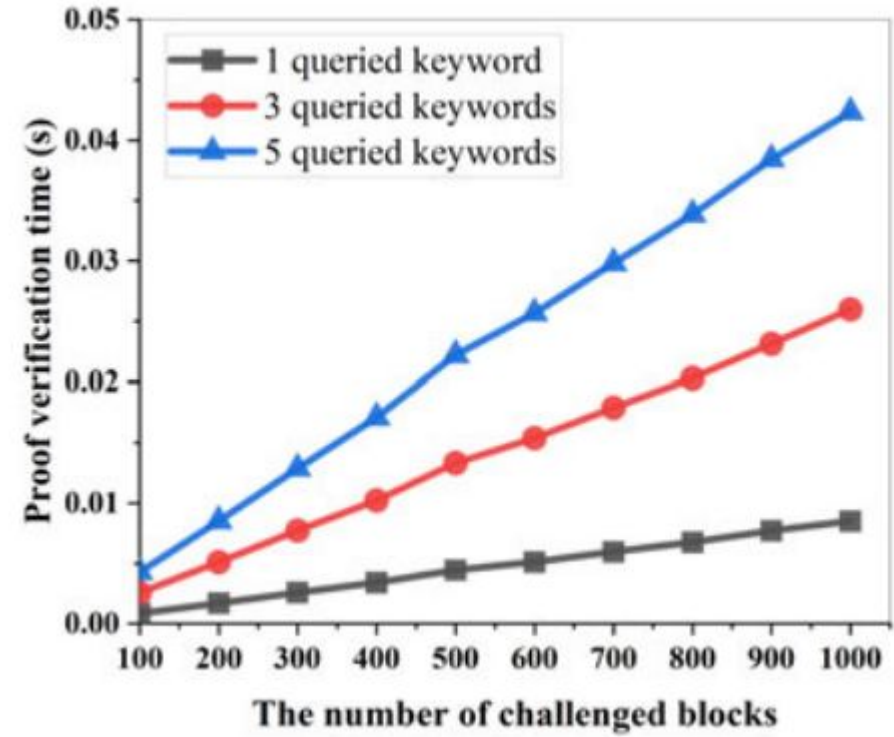
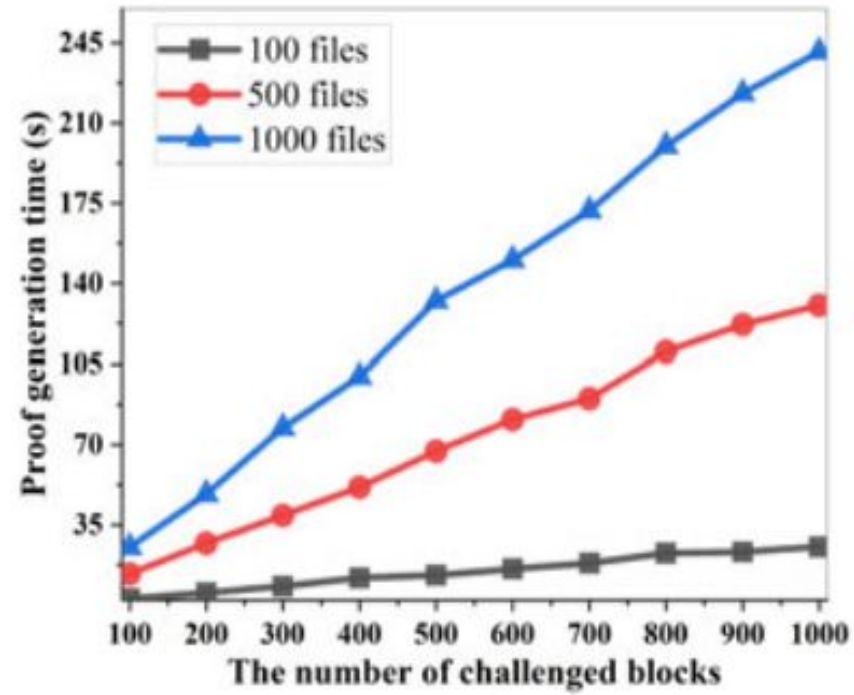
LHS is :{x=1158345588646789296702401035736027869730811201015807567570124  
29507988314778931712162966135092375332719241693578794661318903306093957  
677128486089107561078,y=79412477520504842064178972301776725105200635035  
47791632454178240269445899097156057181413499355693360556517153593000832  
655405598447148311526936275107746495}

RHS is :{x=115834558864678929670240103573602786973081120101580756757012  
42950798831477893171216296613509237533271924169357879466131890330609395  
7677128486089107561078,y=7941247752050484206417897230177672510520063503  
54779163245417824026944589909715605718141349935569336055651715359300083  
2655405598447148311526936275107746495}

LHS

RHS

# PERFORMANCE EVALUATION



# CONCLUSION

Cloud data integrity auditing techniques are important for ensuring the security of outsourced data, but auditing all files in the cloud can be costly and inefficient. Third Party Auditor (TPA) to audit the integrity of all encrypted cloud files containing a specific keyword, without knowing which files contain the keyword or how many files contain it. The scheme uses a newly proposed Relation Authentication Label (RAL) to authenticate the relation that files contain the queried keyword and generate the auditing proof without exposing sensitive information.

## **FUTURE WORK**

- Support multiple keyword queries or SQL-like queries without exposing sensitive information
- Investigate ways to reduce the computation and storage overhead of the RAL in the proposed scheme.
- Design new schemes that can protect forward and backward privacy in the context of data dynamics

# REFERENCES

1. G. Yang, J. Yu, W. Shen, Q. Su, Z. Fu, and R. Hao, ‘Enabling public auditing for shared data in cloud storage supporting identity privacy and traceability’, *J. Syst. Softw.*, vol. 113, pp. 130–139, 2016.
2. R. Bost, P.-A. Fouque, and D. Pointcheval, ‘Verifiable dynamic symmetric searchable encryption: Optimality and forward security’, *IACR, Lyon, France, Rep. 2016/062*, 2016.
3. R. Curtmola, J. A. Garay, S. Kamara, and R. Ostrovsky, ‘Searchable symmetric encryption: Improved definitions and efficient constructions’, *J. Comput. Secur.*, vol. 19, no. 5, pp. 895–934, 2011.
4. X. Ge, J. Yu, C. Hu, H. Zhang, and R. Hao, ‘Enabling efficient verifiable fuzzy keyword search over encrypted data in cloud computing’, *IEEE Access*, vol. 6, pp. 45725–45739, 2018.
5. X. Zhu, Q. Liu, and G. Wang, ‘A novel verifiable and dynamic fuzzy keyword search scheme over encrypted data in cloud computing’, in *Proc. IEEE Trustcom/BigDataSE/ISPA*, 2016, pp. 845–851.
6. Y. Yu, J. Ni, M. H. Au, Y. Mu, B. Wang, and H. Li, ‘Comments on a public auditing mechanism for shared cloud data service’, *IEEE Trans. Serv. Comput.*, vol. 8, no. 6, pp. 998–999, Nov./Dec. 2015.

**THANK YOU**