



## Public Integrity Auditing for Shared Dynamic Cloud Data with Group User Revocation

Ritu A. Rangari<sup>1</sup>, Dr. Dhananjay M. Dakhane<sup>2</sup>

<sup>1</sup>ME Student, Department of Computer Science and Engineering, SIPNA College of Engineering and Technology, Amravati, Maharashtra, India

<sup>2</sup>Associate Professor, Department of Computer Science and Engineering, SIPNA College of Engineering and Technology, Amravati, Maharashtra, India

**Abstract**—The advent of the cloud computing makes storage outsourcing becomes a rising trend, which promotes the secure remote data auditing a hot topic that appeared in the research literature. Recently some researches consider the problem of secure and efficient public data integrity auditing for shared dynamic data. However, these schemes are still not secure against the collusion of cloud storage server and revoked group users during user revocation in practical cloud storage system. In this paper, we figure out the collusion attack in the exiting scheme and provide an efficient public integrity auditing scheme with secure group user revocation based on vector commitment and verifier-local revocation group signature. We design a concrete scheme based on our scheme definition. Our scheme supports the public checking and efficient user revocation and also some nice properties, such as confidently, efficiency, countability and traceability of secure group user revocation. Finally, the security and experimental analysis show that compared with its relevant schemes our scheme is also secure and efficient.

**Keywords**- AGKA, TPA, Cloud Server

### I. INTRODUCTION

The development of cloud computing motivates enterprises and organizations to outsource their data to third-party cloud service providers (CSPs), which will improve the storage limitation of resource constrain local devices. Recently, some commercial cloud storage services, such as the simple storage services(S3) in online data backup services of Amazon, and practical cloud based software Google drive, drop box, mozy, bitcas and memopal have been built for cloud application. There is invalid result in cloud server such as server hardware, software failure, human maintenance and malicious attack. Rabin data dispersion scheme implemented for practical application and overcome above challenges. The limited dynamic scheme cloud only efficiently supports Special field operation (eg. Append).

These applications provide secure file sharing within dynamic group in corporate company with some security features in cloud. Main objectives is to develop secure file storage system on cloud for corporate and to prevent from internal leakage. The static scheme not supports data modification. In publicly verifiable, data integrity check can be performed by data owner and by any third party auditor. Multiple user in group need to share source code they need to access, modify compile and run the shared source code at any time and place. Remote data auditing is only data owner can update its data. Ring signature supports multiple user data operation. The proxy Re-signature is private and authenticated channels exist between each pair of entities. Till today is no solution for above problem in public integrity auditing with group user modification. For providing the integrity and availability of remote cloud store, some solutions [1], [2] and their variants [3], [4], [5], have been proposed. Another attempt to improve the previous scheme and make the scheme efficient, scalable and collusion resistant is Yuan and Yu [6], who designed a dynamic public integrity auditing scheme with group user revocation. A scheme is publicly verifiable means that the data integrity check can be performed not only by data owners, but also by any third-party auditor. Recently, the development of cloud computing boosted some applications where the cloud service is used as a collaboration platform. In these software development environments, multiple users in a group need to share the source code, and they need to access, modify, compile and run the shared source code at any time and place. To support multiple user data operation, Wang et al. [7] proposed a data integrity based on ring signature. In this scheme, the user revocation problem is not considered and the auditing cost is linear to the group size and data size. To further enhance the previous scheme and support group user revocation, Wang et al. [7] designed a scheme based on proxy designators. The authors designed polynomial authentication tags and adopt proxy tag update techniques in their scheme, which make their scheme support public checking and efficient user revocation. It means that, their scheme could efficiently support plaintext data update and integrity auditing, while not cipher text data. In their scheme, if the data owner trivially shares a group key among the group users, the defection or revocation any group user will force the group users to update their shared key.

Also, the data owner does not take part in the user revocation phase, where the cloud itself could conduct the user revocation phase. In this case, the collusion of revoked user and the cloud server will give chance to malicious cloud

server where the cloud server could update the data as many time as designed and provide a legal data finally. To the best knowledge, there is still no solution for the above problem in public integrity auditing with group user modification. The deficiency of above schemes motivates us to explore how to design an efficient and reliable scheme, while achieving secure group user revocation. To the end, here propose a construction which not only supports group data encryption and decryption during the data modification processing, but also realizes efficient and secure user revocation. The group signature will prevent the collusion of cloud and revoked group users, where the data owner will take part in the user revocation phase and the cloud could not revoke the data that last modified by the revoked user.

## II. LITERATURE REVIEW

Plenty of researchers have devoted considerable attention to the problems on to securely outsource local store to remote cloud server. Among which, the problem of remote data integrity and availability auditing attacks the attestation of many researchers. The concepts and solution Provable Data Possession (PDP) and Proofs of Retrievability (PoR) were first proposed by Ateniese et al.[9] And Juels et al[10]. In their scheme, the homomorphic authentication technique was adopted to reduce both the communication and computation cost. Later, a number of variants of PDP and PoR schemes are designed to improve the efficiency and enhance the function of basic schemes, such as allowing public auditing [3],[4], [5], [6] and supporting data update. To enhance the previous works, Wang et al. [5] designed a scheme to support share data integrity auditing, whose scheme adopted ring signature to protect the privacy of users.

To further support user revocation, Wang et al. [5] designed another scheme based on the assumption that no collusion occurs between cloud servers and revoked user. Group signatures without revocation. The limitation of the scheme is that it does not support dynamic group and also suffers from a computational overhead linear to the group size and the number of data auditing were proposed by Tao Jiang, Xiaofeng Chen, and Jianfeng Ma[8]. At that time, the security of group signatures was not totally understood and proper security definitions were given later on by Bellare, Micciancio and Warinschi [9] (BMW) whose model captures all the requirements of group signatures in three properties. In (a relaxation of) this model, Boneh, Boyen and Shacham [9] obtained a construction in the random oracle model with signatures shorter than 200 bytes .

In the BMW model, the population of users is frozen after the setup phase beyond which no new member can be added. In these models, pairing-based schemes with relatively short signatures were put forth in Ateniese et al. [9] also gave a construction without random oracles using interactive assumptions. In the BMW model [9], Boyen and Waters independently came up with a different standard model proposal using more classical assumptions and they subsequently renamed their scheme to obtain constant-size signatures. The provably coalition-resistant scalable group signature was described by Ateniese, Camenisch, Joye and Tsudik[10][11][12]. An alternative approach taken by Bresson and Stern is to have the signature. The group manager maintains a revocation list (RL) which is used by varies to make sure that signatures were not generated by a revoked member Prove that this membership certificate does not appear in a public list or revoked certificate.

For providing the integrity and availability of remote cloud store, some solutions [10], [11] and their variants [12], [13], [14], [15], [16], have been proposed. In these solutions, when a scheme supports data modification, we call it *dynamic* scheme, otherwise *static* one (or limited dynamic scheme, if a scheme could only efficiently support some specified operation, such as append) In group signatures, membership revocation has received much attention in the last decade since revocation is central to digital signature schemes[13][14]. One simple solution is to generate a new group public key and deliver a new signing key to each unrevoked member propose by Boyang Wang, Baochun Li, and Hui Li[17][18]. However, in large groups, it may be inconvenient to change the public key and send a new secret to signers after they joined the group. where group membership can be disabled without affecting the signing ability of unrevoked users. Boneh and Shacham [21][22][23] proposed an efficient group signature with verifier-local revocation.

## III. METHODOLOGY

Systems Model explores on the secure and efficient shared data integrate auditing for multi-user operation for cipher text database. By incorporating the primitives of victor commitment, asymmetric group key agreement and group signature and an efficient data auditing scheme while at the same time providing some new features, such as traceability and countability. Also provide the security and efficiency analysis and the analysis results show that it is secure and efficient. A system model for the cloud storage architecture, which includes three main network entities: users, a cloud server, and a trusted third party.

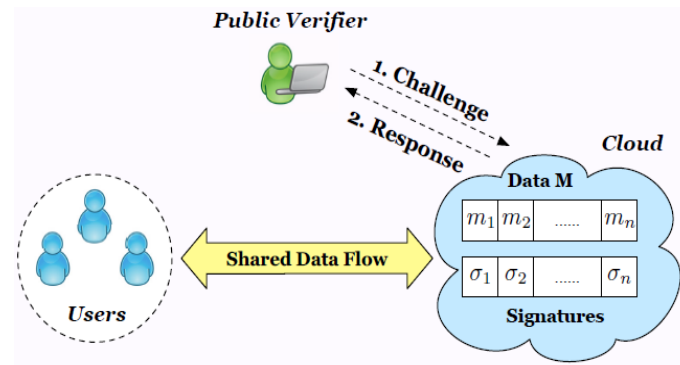


Fig 1 Architecture

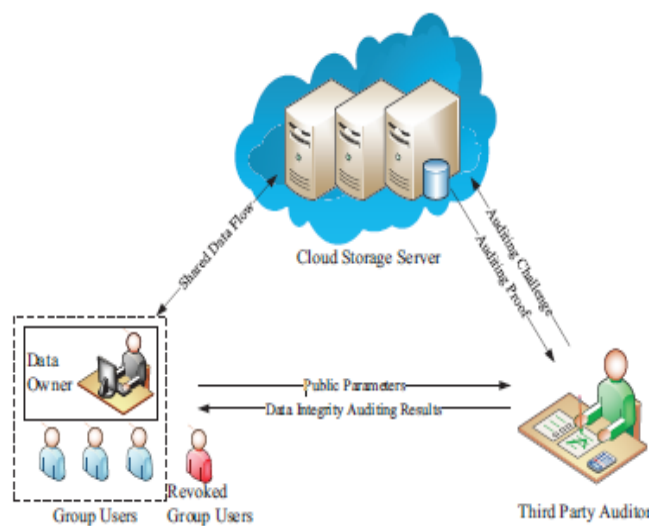
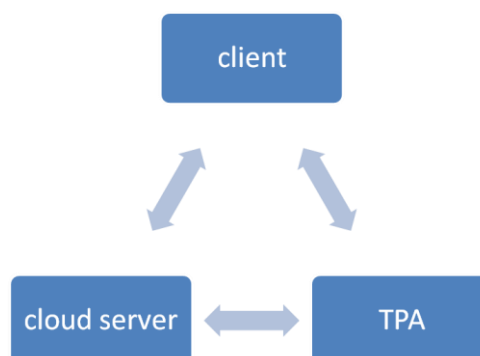


Fig 2 System Model

In above system model the data owner could encrypt and upload its data to the remote cloud storage server. Also, the access and modify privileges is shared to a number of group users. Even if the data is frequently updated by the group users, the TPA efficiently verifies the integrity of the data stored in the cloud storage server. The owner of data is different from other group users. When a group user is found malicious or the contract of the user is expired, he/she could securely revoke a group user.

1. Text.txt - Authentication
2. Text. Key – private key
- 3 cloud metadata - metadata
4. Tpa\_metadata -TPA



1. Text.txt
2. Text. Key
3. Cloud metadata

- 1.Text.txt
- 2.Tpa\_metadata

Fig 3 Data Flow Diagram

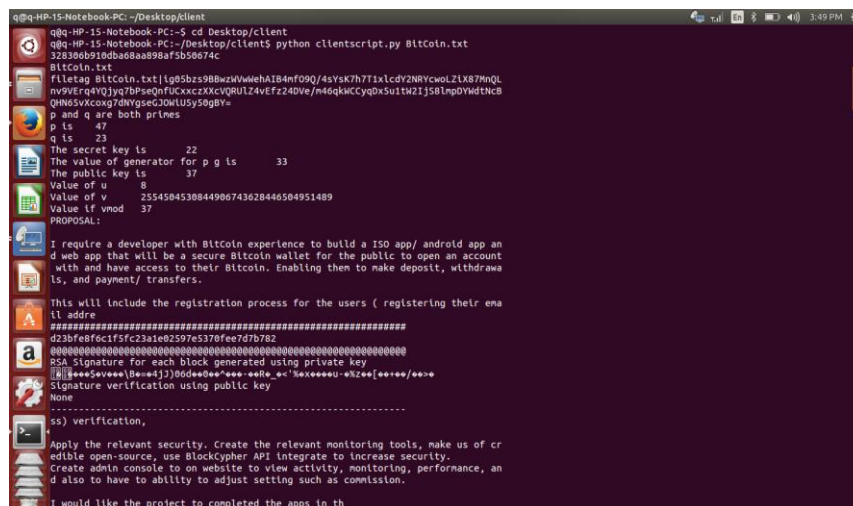
### 3.1 Propose Model Design

Here, there are three modules which are as follows,

- User
- Cloud server
- Third Party Auditor

**3.1.1 User:** An individual or group entity, which owns its data stored in the cloud for online data storage and computing. Group users consist of a data owner and a number of users who are authorized to access and modify the data by the data owner.

1. Run clientscript.py
2. Enter name of files as comma separated list. Whole list within single inverted comma as 'f1.py,f2.py'
3. It will generate the f1.crt,f1.key,f2.crt,f2.key,tpa\_metadata.txt and cloud\_metadata.txt



```
q@q-HP-15-Notebook-PC: ~/Desktop/client
q@q-HP-15-Notebook-PC:~$ cd Desktop/client
q@q-HP-15-Notebook-PC:~/Desktop/client$ python clientscript.py Bitcoin.txt
Bitcoin.txt
Filetag Bitcoin.txt|g05bz98BwzVwMehA184nf09Q/4sYsk7h7T1xldY2NRYcwoL2LX87nnQL
nv9Pqrq4VQjyq7bPseQnFUCxczXKcVQRULZ4vEfz24DVe/m46qkKCyqDxSutN21j501mpYwdTncB
QMH6svKcong7dWgseC5Wu1lly50gVv
p and q are both primes
p is 47
q is 23
The secret key is 22
The value of generator for p g is 33
The public key is 37
Value of u 8
Value of v 2554504530844906743628446504951489
Value of vm0d 37
PROPOSAL:
I require a developer with Bitcoin experience to build a ISO app/ android app an
d web app that will be a secure Bitcoin wallet for the public to open an account
with and have access to their Bitcoin. Enabling them to make deposit, withdraw
ls, and payment/ transfers.
This will include the registration process for the users ( registering their ema
il addre
#####
d23bfef6cf5fc23a1e02597e5370fee7d7b782
#####
25A Signature for each block generated using private key
[#####]
Signature verification using public key
None
-----
ss) verification,
Apply the relevant security, Create the relevant monitoring tools, make us of cr
edible open-source, use BlockCypher API integrate to increase security.
Create admin console to on website to view activity, monitoring, performance, an
d also to have to ability to adjust setting such as commission.
I would like the project to completed the apps in th
```

Fig 4. User Text File Run With Clientscript.Py

After running user text file with clientscript.py three supportive files are generated. Which are as follows?

1. BitCoin.txt file
2. BitCoin.key file
3. BitCoin.crt file
4. Cloud\_metadata
5. TPA\_metadata

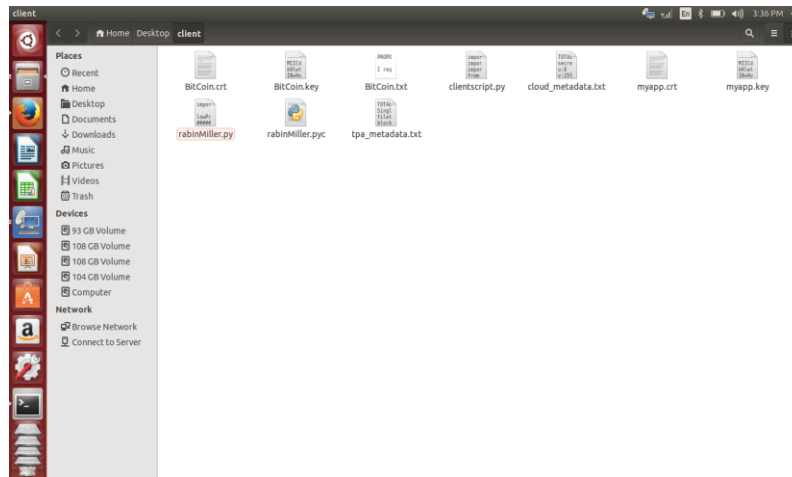


Fig 5 generating supportive files

**3.1.2 Cloud Server:** An entity, which is managed by a particular cloud service provider or cloud application operator to provide data storage and computing services. The cloud storage server is semi-trusted, who provides data storage services for the group users. The cloud server is regarded as an entity with unrestricted storage and computational resource.

Working of cloud server after receiving challenge is as follows:

It parse the parameter and extracts the information as filename=filename  
block number =i and block\_associate=vi.

It calls get Challenged Block (int(i),adder) where adder is required to point to correct line number within the cloud\_metadata.txt file. So adder is passed. While i is th ith block Now it extracts the correct block and pass it to the generate Proof (str(block\_extracted),key\_file\_name,secret\_key):: this methods take the hash on the block and the computes its signature and then encode it using the base64 and returns the encoded is to get Challenged Block () method.

Now this encoded challenge is return to the testkishan (). Testkishan () is called on /challenge route.

Now the result is constructed as :

```
reply=get Challenged Block(int(i),adder)  
filename: i:reply::i:reply::filename.....
```

Now total\_reply is return to the client (which is tpa[because tpa has send the request])

Now upload following files to the cloud server:

- f1.key
- f1.txt
- f2.key
- f2.txt
- cloud\_metadata.txt

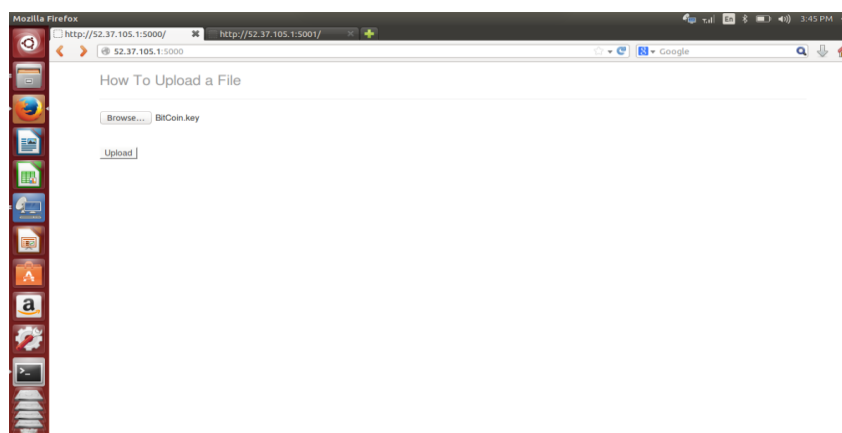


Fig 6 Uploading Text.Key File On Cloud Server

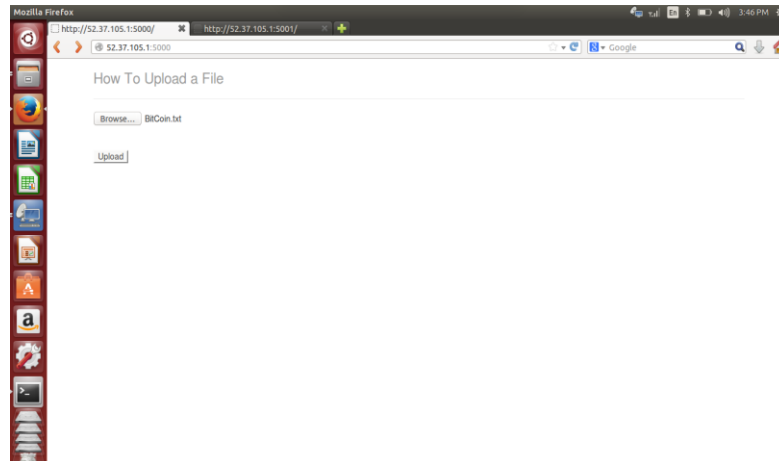


Fig 7 uploading text.txt files on cloud server

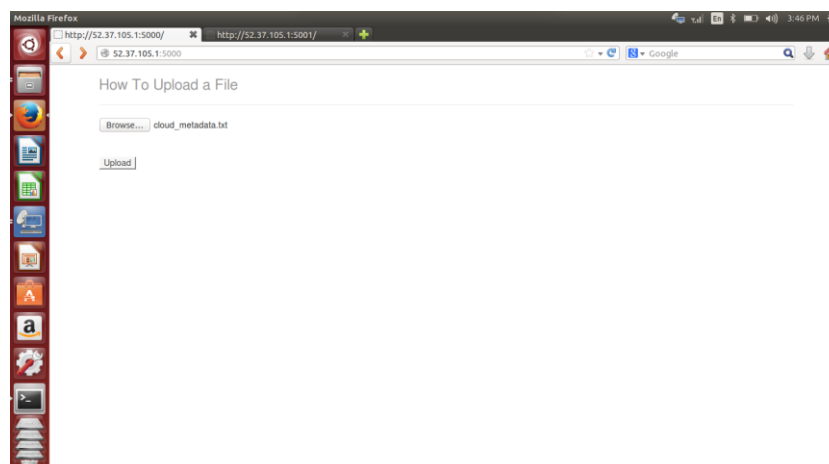


Fig 8 uploading cloud metadata on cloud server

**3.1.3. Trusted Third Party:** An optional and neutral entity, which has advanced capabilities on behalf of the users, to perform data public auditing and dispute arbitration. In the cloud storage, a user remotely stores its data via online infrastructures, platforms, or software for cloud services, which are operated in the distributed, parallel, and cooperative modes. TPA could be any entity in the cloud, which will be able to conduct the data integrity of the shared data stored in the cloud server. During cloud data accessing, the user autonomously interacts with the cloud server without external interferences, and is assigned with the full and independent authority on its own data fields. It is necessary to guarantee that the users' outsourced data cannot be unauthorized accessed by other users.

Now upload following files to tpaserver:

f1.crt

f2.crt

tpa\_metadata.txt

Now click the verify button on the page of tpa server.

Get the verified result.

TPA reads the tpa\_metadata.txt file : read\_metadata(): returns the metadata of file.

It extracts the total number of files details mention in tpa\_metadata.txt: get\_tag\_values(metadata) it extracts the name of all files, file tag and total number of blocks contain in each of the file.

Now as per the paper it first verifies whether the filename extracted verified the signatuer of the filename: thus calls the verifyFileTag()

After knowing that the all files are correct it proceeds below

It returns the list of filename and finally the totalnumber of blocks as last entry of list.

Now generate\_challenge(tag\_values) is called passing the result of previous call that is list Here the challenge is created as single\_chal=[filename,i,vi] and chal=[single\_chal1,single\_chal2] as total number of files mention in tpa\_metadata.txt  
Now the challenge is send to cloud using challenge\_cloud(chal) where chal is list  
Now challenge is moulded properly as filename:i:vi::i:vi:::filename2..  
proof =urllib2.urlopen('http://kishancsp-rtpa.rhcloud.com/challenge?chal='+vii).read()  
challenge is send.  
verification\_result=verifyProof(proof) is called. At verifyProof the corresponding reply is  
parse and filename=filename is extracted block number is extracted signaturer is extracted. Now it is passed to  
verified=verifySignature(file\_name,int(vi),sig): Here the tpa\_metadata.txt is read and parse the filename is searched and the 'i' th signature is extracted. Now signature is matched as if itis a string. And the reult is return as html data. And send.  
Result is received and response is shown in index.html page.

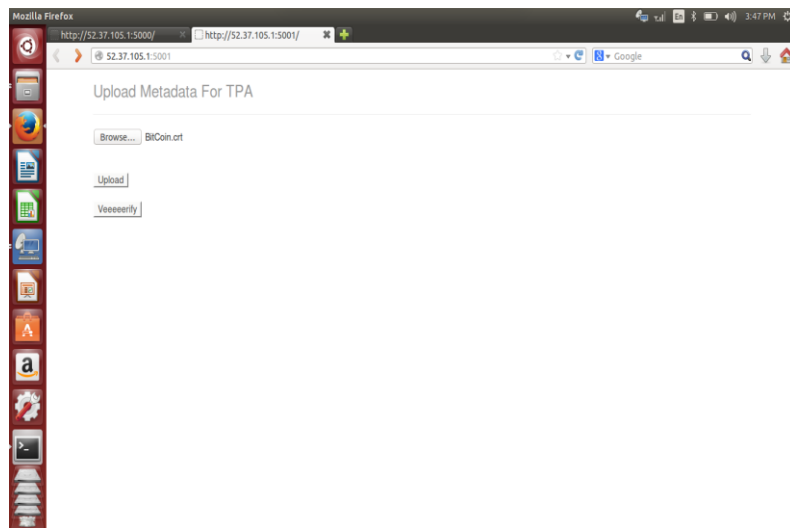


Fig 9 uploading txt.crt file on TPA

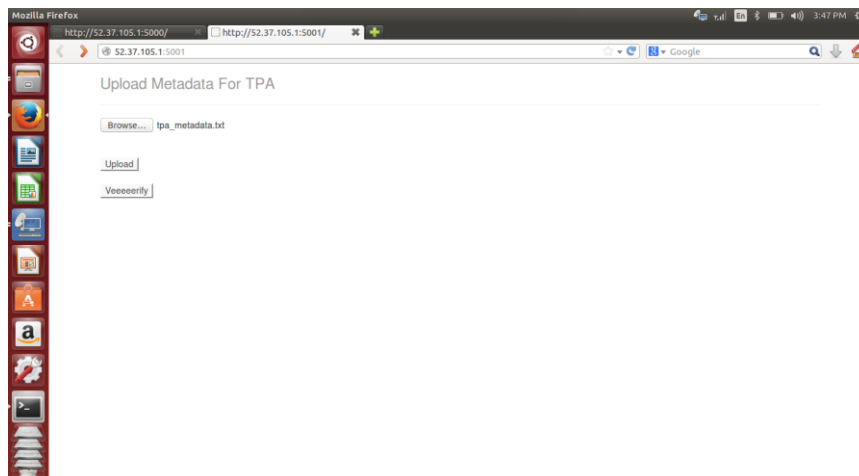


Fig 10 uploading tpa\_metadata on TPA

#### IV. EXPERIMENTAL RESULT

Result generated in two conditions which are as follows,

**4.1 True condition:** - here if integrity and security of files which are uploaded on cloud server is maintained then result is shown as follows:



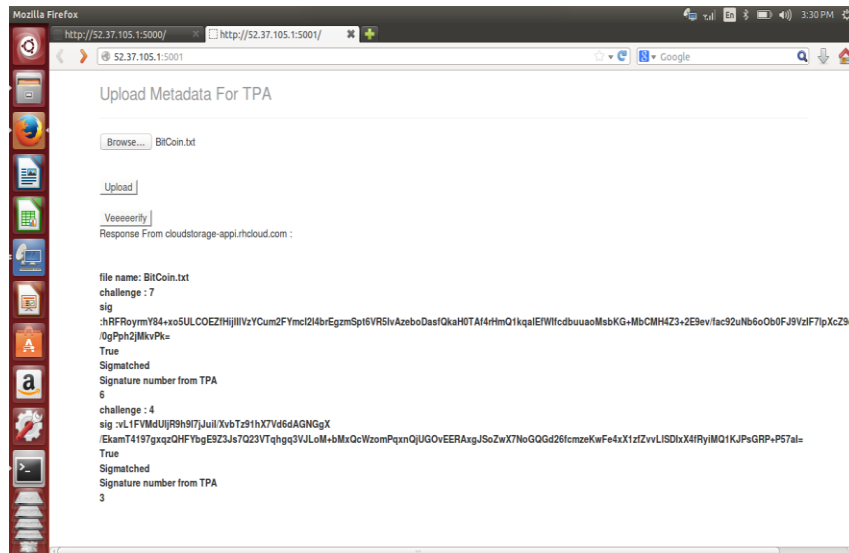


Fig 11. True

**4.2 False condition:-** here if integrity & security of files which are uploaded on cloud server is not maintained or access by unauthorized person then result is shown as follows.

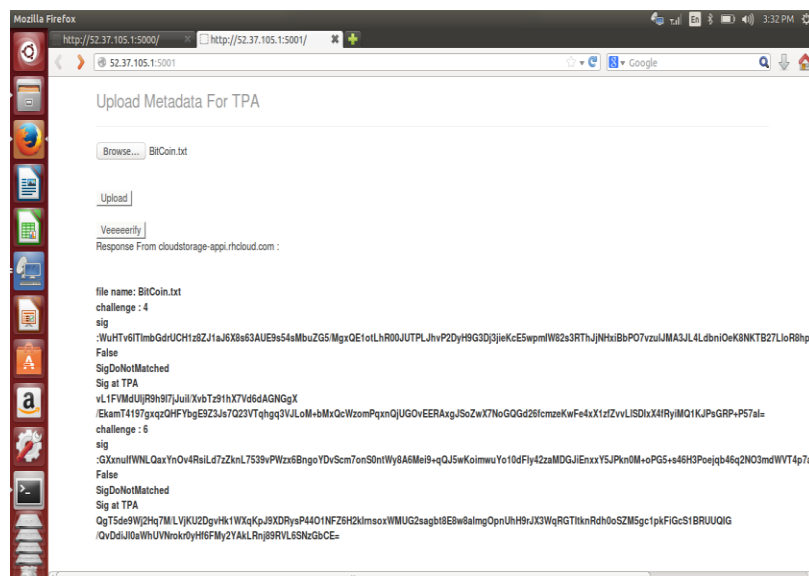


Fig 12. False

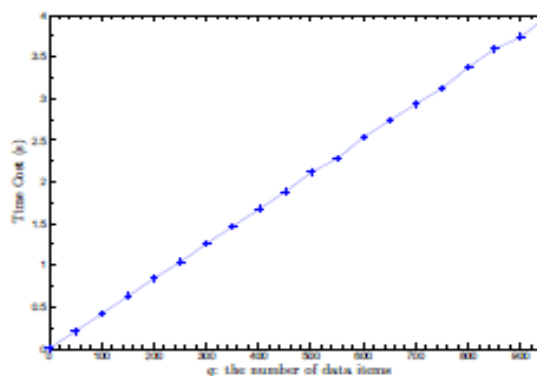


Fig 13 Query Time Cost



In above Figure, the Query time cost of is linear with the data items number  $q$ , which will take approximately 4 seconds to query about 1000 data items. However, here need to emphasize that the computation cost is at the cloud storage server side, which is very powerful compare with the Linux system running on our laptop.

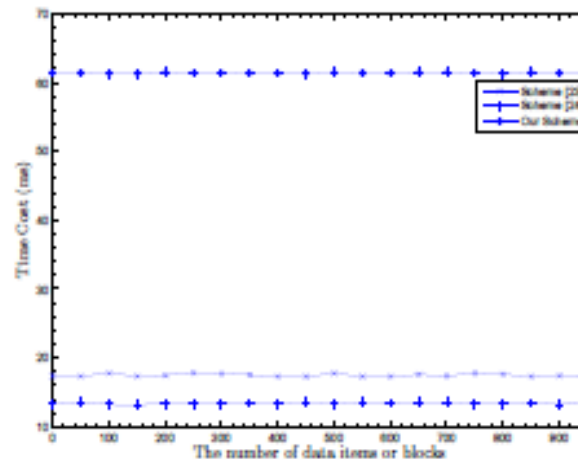


Fig 14 Verify Time Cost

Actually, in the Verify algorithm, the computation overhead mostly comes from the group signature scheme. More precisely, to verify the validity of this phase, here need firstly to verify the integrity of the signature, which means that need to generate the time costed parameters such as  $R_1$ ,  $R_2$ , and  $R_3$ . Actually, the computation time cost that is a constant number.

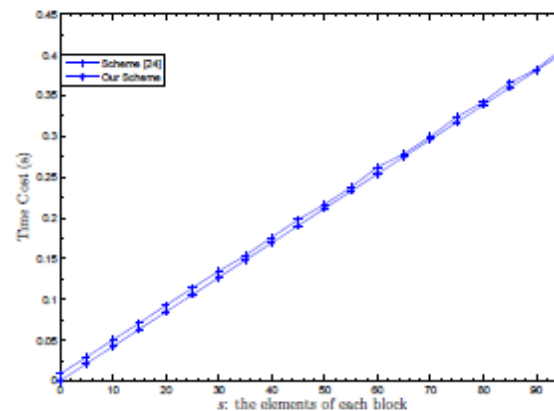


Fig 15 Update Time Cost

In above Figure , here show the data update computation comparison and both their computation overheads grow with the increase of the element number in each Block.

## V. CONCLUSION

There will be no any type of attack or data loss caused by the attackers and the intruders. A very high security will be provided to the users who outsourced there data to the cloud services providers and thus the use cloud services will be increased tremendously. Another research direction would be to give the data owner physical access control over his data. Instead of accountability the data owner can create a set of access control rules on his data and send the data along with the access control policy. The preserving of verifiable database with efficient and secure updates is an important way to solve the problem of verifiable data storage. The proposed system securely share the data file among the dynamic groups without revealing their identity members in the same group can share the data efficiently. Cryptography is used for over all security. It is used for efficient revocation without updating private keys of remaining users. In future, concentrate on key management, how to revoke the private keys from the group members. Here provide security analysis, and it shows that propose system model provide data confidentiality for group users, and it is also secure against the collusion attack from the cloud storage server and revoked group users

## REFERENCES

- [1] G. Ateniese, R. Burns, R. Curtmola, J. Herring, L. Kissner, Z. Peterson, and D. Song, "Provable data possession at untrusted stores," in *Proc. of ACM CCS*, Virginia, USA, Oct. 2007, pp. 598–609.
- [2] A. Juels and B. S. Kaliski, "Pors: Proofs of retrievability for large files," in *Proc. of ACM CCS*, Virginia, USA, Oct. 2007, pp. 584–597.
- [3] C. Wang, Q. Wang, K. Ren, and W. Lou, "Privacy-preserving public auditing for data storage security in cloud computing," in *Proc. of IEEE INFOCOM 2010*, CA, USA, Mar. 2010, pp. 525–533.
- [4] J. Yuan and S. Yu, "Proofs of retrievability with public verifiability and constant communication cost in cloud," in *Proc. of International Workshop on Security in Cloud Computing*, Hangzhou, China, May 2013, pp. 19–26.
- [5] B. Wang, B. Li, and H. Li, "Oruta: Privacy-preserving public auditing for shared data in the cloud," in *Proc. of IEEE CLOUD2012*, Hawaii, USA, Jun. 2012, pp. 295–302.
- [6] B. Wang, L. Baochun, and L. Hui, "Public auditing for shared data with efficient user revocation in the cloud," in *Proc. Of IEEE INFOCOM 2013*, Turin, Italy, Apr. 2013, pp. 2904–2912.
- [7] J. Yuan and S. Yu, "Efficient public integrity checking for cloud data sharing with multi-user modification," in *Proc. of IEEE INFOCOM 2014*, Toronto, Canada, Apr. 2014, pp. 2121–2129.
- [8] Public Integrity Auditing for Shared Dynamic Cloud Data with Group User Revocation Tao Jiang, Xiaofeng Chen, and Jianfeng Ma.
- [9] G. Ateniese, R. Burns, R. Curtmola, J. Herring, L. Kissner, Z. Peterson, and D. Song, "Provable data possession at untrusted stores," in *Proc. of ACM CCS*, Virginia, USA, Oct. 2007, pp. 598–609.
- [10] A. Juels and B. S. Kaliski, "Pors: Proofs of retrievability for large files," in *Proc. of ACM CCS*, Virginia, USA, Oct. 2007, pp. 584–597.
- [11] Tao Jiang, Xiaofeng Chen, and Jianfeng Ma, "Public Integrity Auditing for Shared Dynamic Cloud Data with Group User Revocation", *IEEE Transactions on Computers* 2015
- [12] B. Wang, B. Li, and H. Li, "Public Auditing for Shared Data with Efficient User Revoation in the Cloud," in the *Proceedings of IEEE INFOCOM 2013*, 2013, pp. 2904–2912.
- [13] 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, April 2010.
- [14] Boyang Wang, Baochun Li, and Hui Li, "Panda: Public Auditing for Shared Data with Efficient User Revocation in the Cloud" *IEEE TRANSACTIONS ON SERVICE COMPUTING VOL. PP, NO. 99*, December 2013.
- [15] B. Libert, T. Peters, and M. Yung, "Scalable group signatures with revocation," in *Proc. of EUROCRYPT 2012*, CA, USA, Aug. 2002, pp. 61–76
- [16] E. Bresson and J. Stern, "Efficient revocation in group signatures," in *Public-Key Cryptography - PKC 2001*, Cheju Island, Korea, Feb. 2001, pp. 190–206.
- [17] D. Catalano and D. Fiore, "Vector commitments and their applications," in *Public-Key Cryptography - PKC 2013*, Nara, Japan, Mar. 2013, pp. 55–72.
- [18] Q. Wu, Y. Mu, W. Susilo, B. Qin, and J. Domingo-Ferrer, "Asymmetric group key agreement," in *Proc. of EUROCRYPT 2009*, Cologne, Germany, Apr. 2009, pp. 153–170.
- [19] D. Boneh and H. Shacham, "Group signatures with verifierlocal revocation," in *Proc. of ACM CCS*, DC, USA, Oct. 2004, pp. 168–177.
- [20] D. Boneh, B. Lynn, and H. Shacham, "Short signatures from the weil pairing," in *Proc. of Asiacrypt 2001*, Gold Coast, Australia, Dec. 2001, pp. 514–532.
- [21] D. Boneh and X. Boyen, "Collision-free accumulators and failstop signature schemes without trees," in *Proc. of EUROCRYPT 2004*, Interlaken, Switzerland, May 2004, pp. 56–73.
- [22] N. Baric and B. Pfitzman, "Collision-free accumulators and fail-stop signature schemes without trees," in *Proc. of EURO-CRYPT 1997*, Konstanz, Germany, May 1997, pp. 480–494.