ISSN: 0975-766X
CODEN: IJPTFI
Available Online through
www.ijptonline.com
Research Article

REMOTE DATA POSSESSION CHECKING PROTOCOL USING GALOIS FIELD GF (P^N) IN MOBILE CLOUD COMPUTING

Kalpana V*[1], Meena V[1], ShankarSriram V.S[2]
[1]Computer Science and Engineering, [2]Information Technology
[1,2] School of Computing, SASTRA University, Tirumalaisamudram, Thanjavur - 613401. Tamilnadu, India.
Email: kalpana@cse.sastra.edu

Received on 06-08-2016 *Accepted on 10-09-2016*

Abstract

Cloud computing provides enormous amount of storage space for the users. Terabytes of data is stored across the world in the various cloud servers. Resource intensive mobile devices store the data through the Storage as a Service in the remote cloud. Cloud is not trusted for security and integrity of the mobile user's data. So the data is stored in encrypted form to provide confidentiality. Without downloading the contents of the file the integrity has to be checked by the resource intensive mobile devices. To ensure the correctness of the data in the remote cloud server a Remote Checking Protocol is needed. A novel framework is proposed to check the integrity of a data file with optimal computation and minimum storage cost in the mobile device. Mobile client possess only the Meta tag for verification and deletes the file in its local storage. This protocol verifies the data stored in the remote cloud server infinite times and supports the data modification in the cloud server dynamically. A naive approach is proposed to verify the data storage correctness using Galois field GF (p^n) computations.

Keywords: Cloud storage, Data Security, Data Integrity, Optimal computation, Galois Field, Remote checking

1. Introduction

Mobile Cloud Computing is a heterogeneous computing environment where the resource constrained mobile device augments its data and computation to cloud [1]. Mobile devices offload the intensive computing code and data to cloud service provider to save battery energy. Even though the cloud server stores data where as security and reliability were a big concern in mobile cloud computing environment [4]. To realize that the data is not changed in the remote cloud server mobile user needs an efficient protocol to verify the file data contents. The protocol should allow the mobile user to check the correctness, add, modify and delete the data from the file online. In this paper a novel framework is proposed to do remote data checking in a mobile cloud computing environment supporting data

dynamics. Without downloading the data contents the integrity of the file should be checked in the remote cloud storage. As mobile devices lack in its computing power and battery energy file integrity need to be checked with minimum computation cost. A Meta tag is generated in the mobile device to ensure the correctness of the data in the remote cloud. The protocol allows the mobile user to check the integrity infinite number of times. In this paper section 2 discusses the various existing works of the cloud storage correctness methods of different methods. The novel framework and results are presented in the section 3, 4 and the section 5 concludes the paper.

2. Related Work

Data Storage Correctness checking falls in three dimensions based on type, method, protocol followed as shown in Figure 2.1. The two types of Auditability are private and public where the private Auditability uses secret key so only the mobile user alone can verify the integrity where as public Auditability does not involve secret key so any one can verify the data in cloud server behalf of mobile user[2].

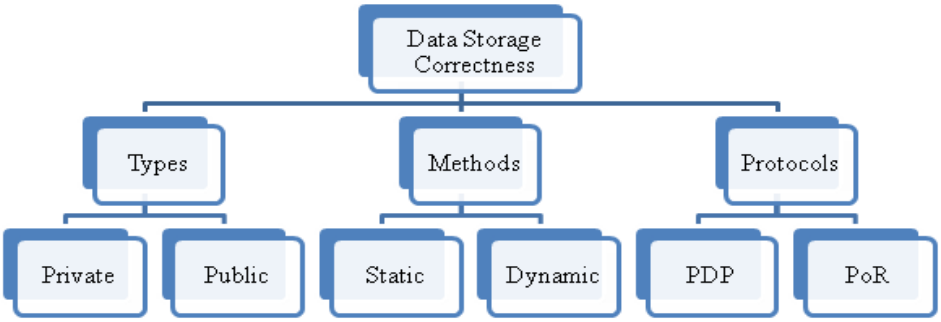

Figure 2.1 Hierarchy of Data Storage Correctness Methods.

Static method does not allow any modification in the remote cloud server and the data is retained as static. In dynamic data auditing method mobile user is allowed to change the data contents in the remote cloud server without downloading the file. The protocols used in auditing technique are Provable Data Possession [3] and Proof of Retrieval. Data Storage correctness framework in the previous work has been done in three ways using Signatures, Message Authentication Code and Hash generation. Kan Yang [5] proposed an auditing protocol for cloud storage with data dynamics. This protocol is efficient for checking the integrity and prevents the replay attack by including the index of the data block in its tag generation. Data insertion and deletion needs an enormous amount of change in tag generation because of the index inclusion in the tag generation. It has to recompute all the tags following the insertion or deletion data block. If the data has to be inserted in the beginning then this protocol will take more computation cost. Chen et al., [6] proposed two PDP schemes with limited number of verification and unlimited number of verification with probabilistic checking. Using algebraic signature method integrity is checked by selecting

random blocks in the cloud server. Block Tags are computed individually for the random blocks that are chosen by the mobile user and the sum of the tag is stored in the mobile device. Cloud server receives the random indexing of the file blocks from the mobile device and computes the sum of all the selected random blocks. Mobile user computes the signature for the received sum value and checks whether it is the same as the stored tag. Both schemes upload the file without encryption and it does not support confidentiality. Data dynamics not incorporated in this Chen method. Yong et al., [7] proposed data auditing using Zero Knowledge Proof. It supports private Auditability and data dynamics. But this method uses modular arithmetic over Z_n. Y. Yu et al. [8] proposed a Remote Data Possession Checking in the cloud storage as an improvement of the algebraic signature scheme of Chen et al., [6]. Replay attack is prevented in this method. This protocol uses index of the block for Tag Generation so block addition, deletion and modification needs enormous amount of change in the existing tags. This method is not suitable for data dynamics in Cloud environment. Kalpana et al., [9] compares the various Data Storage Correctness Methods in Mobile Cloud Computing based on public and private Auditability. In this paper the public Auditability is chosen as the optimized method for Mobile Cloud Computing. Chen et al., [10] incorporated data auditing in the cloud storage with dynamic data update. Homomorphic hashing is used to verify the storage correctness and Merkle Hash Tree is implemented to support data update in remote cloud server. Here the cloud server can do the replay attack during the update operation by sending the previous computed tag data blocks as the Auxiliary Authentication Information (AAI).

Mehdi et al., [11] used Divide and Conquer Table (DCT) to implement remote data auditing supporting data dynamics. They have used Logical index number and version number as the entry in the DCT. Version number keeps track of the updating information and the index number is not changed for modification.

3. Proposed Method

3.1 Preliminaries

Galois Field GF (p^n) is a finite field where p denotes a prime number and n denotes an positive integer. To store the data efficiently without wasting the bit in the storage area GF(p^n) is used instead of GF(p). The data will be in the range of 0 to $2^n - 1$ without wasting the bit patterns. All the operations are performed in the Galois Field GF (P^n). Addition and subtraction will be XOR of the bytes and multiplication as well as division is done using irreducible polynomial. A generator of the Galois Field plays a vital role in arithmetic operations. A generator g with its successive power generates all the elements in the finite field without repetition. For $r = g^m$ it is very hard to find the m where g and r is known. It will generate unique elements from 0 to 2^{n-1}. Cloud Computing framework is shown in

the Figure 3.1. User holds the mobile devices like cell phones, Personal Digital Assistants and laptops. User encrypts the file using Elliptic Curve Cryptography over finite field F_p and sends to the Cloud Server for storage along with the tag. The encrypted file is uploaded to the cloud and then mobile client deletes the file from its storage space. Both in Cloud Server and Client side the computations for integrity check are made using Galois Field GF (p^n).

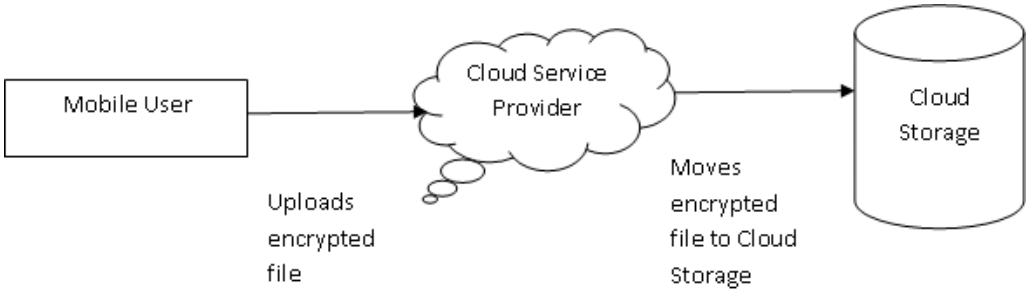

Figure 3.1. Cloud Computing Framework.

3.2 Algorithm for Storage Correctness Method in Mobile Cloud Computing

The algorithm composes of four phases as Tag Generation, Challenge, Response and Verification. The mobile client computes the tag for the file and stores the Meta tag in its storage. To check the data integrity client challenges the cloud server by selecting random blocks. The cloud server in turn gives the proof as the response. Mobile client verifies the proof in its verification phase.

3.2.1 Tag Generation

1. Split the text file in to n blocks $m_1, m_2 \ldots\ldots m_n$.
2. Each block is divided in to m sub blocks where each sub block consists of 2^l bits where l=8 or 16.
3. Each sub block is composed of the symbols $x_{i1}, x_{i2} \ldots x_{im}$ where $1 \leq i \leq n$ and the symbols are the elements of GF(p^n).
4. A tag is generated for each block using the Galois exponentiation function $T_i = \prod_{j=1}^{m} g^{xij}$ where g is the generator of the Galois Field GF (p^n).
5. Client generates the file tag by applying $\prod_{i=1}^{n} T_i$ and stores the value in its local storage.
6. Client stores the Ti as the leaves in the binary tree.
7. Each level the parent of the binary tree is computed by concatenating the two children nodes and taking exponentiation of that value in the Galois Field GF (p^n).
8. The above step is repeated up to the root of the binary tree as shown in Figure 3.2.
9. Client stores the Root Tag and deletes all the T_i, binary tree, and file in its local storage.

3.2.2 Challenge

1. Client chooses a random number r and computes $R = g^r$ in the Galois Field GF (p^n).
2. Client selects random subset of the index of the blocks b_1, b_2…… b_k for verification.
3. Client sends the g, R and index of the selected data blocks to the server.

3.2.3 Response

1. Server computes the sum of the data blocks b_1, b_2…… b_k as SB.
2. Server generates Proof $Pr = R^{SB}$ in GF(p^n) .
3. Server sends the selected subset of data blocks, Pr, siblings in the path of the root, Root Tag to the client.

3.2.4 Verification

1. Mobile Client receives the subset of data blocks and computes g^{bi} for each data block.
2. Constructs the root of the binary tree by the values obtained from the step 1 and from the Cloud server.
3. Mobile Client receives the Pr and calculates the $Vr = FT^r$ in GF(p^n).
4. Mobile Client checks whether Pr = Vr as shown in Figure 3.2.
5. If the above condition is true then client concludes that integrity of the file is preserved.
6. Mobile checks whether the root of the binary tree is same as Root Tag as shown in Figure 3.3 for authentication of the position of the data blocks in the file.

3.3 Data Dynamic Update methods

Data dynamics is supported by insert delete and modify operations in the cloud server by selecting only the part of the data without downloading the whole data contents.

3.3.1 Block Insertion

To insert a new block of the file in the cloud server the Mobile Client performs the following steps,

i. Computes the tag of the new data block $T_{ins} = \prod_{i=1}^{n} g^{xij}$ and computes the updated file tag as $T_{new} = T_{old} * T_{ins}$.

ii. Mobile client sends the data block to be inserted to the cloud.

iii. Cloud in turn returns the sum of the data blocks after updating.

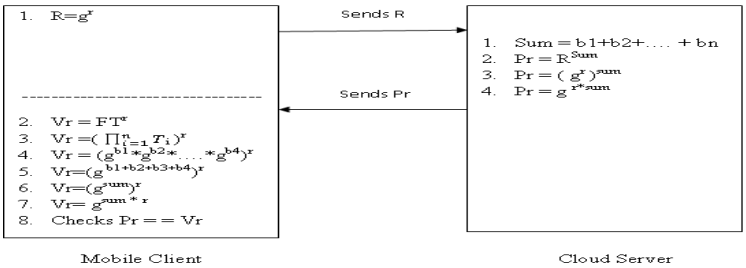

Figure 3.2 Verification of Proof between Mobile Client and Cloud Server.

iv. Mobile client computes the $T'_{new} = g^{newsum}$.

v. If $T_{new} = T'_{new}$ then the update is success.

vi. To check the position of the newly inserted data block mobile client randomly requests the subset of the data blocks from the cloud server.

vii. Cloud Server returns the data of the selected data blocks and the siblings up to the root to the Mobile Client and Mobile Client computes the root of the binary tree.

viii. If both roots are equal then position of the data blocks are verified.

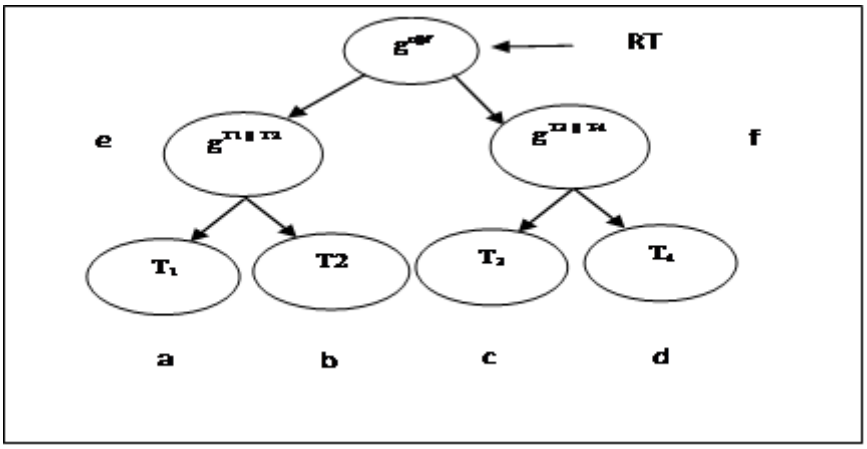

Figure 3.3 Binary trees for verification.

3.3.2 Block Deletion

i. Find the multiplicative inverse of the tag value (T_{Inv}) of the data block that need to be deleted.

ii. Compute the new tag value after the deletion of the file data block as $T_{del} = T * T_{Inv}$.

iii. Mobile user requests the remote cloud server for the sum of the data blocks.

iv. Mobile client computes $T'_{new} = g^{sumdel}$ checks $T'_{del} = T_{del}$.

v. If the step (iv) succeeds then the data block has been deleted from the cloud storage successfully.

3.3.3 Block Modification

To modify a particular file block Mobile Client does the following steps,

i. Mobile client requests the data block to be modified from the cloud server.

ii. Cloud server sends the requested data block to the Mobile Client.

iii. Mobile Client computes the tag of the received block as T_{mold}.

iv. Mobile Client modifies the data block and computes the tag of the modified data block T_{mnew}.

v. Mobile Client computes the Tag of the file as $T_{modnew} = T_{old} * T_{mold} * T_{mi} * T_{mnew}$.

vi. Mobile client transfers the data block to be inserted in the file to the cloud.

vii. Cloud in turn returns the sum of the data blocks after updating.

viii. Mobile client computes the $T'_{modnew} = g^{modsum}$.

ix. If $T_{modnew} = T'_{new}$ then the update is success.

4. Results & Discussion

4.1 Storage and computation cost of mobile client

The mobile client has to store only l bits (where l=8/16) of the tag regardless of file size. Root Tag is used to verify integrity of the data. The binary tree is used to check the position of the data blocks. For example if the client has to store 1GB file in the cloud storage. The file is split in to 256 blocks of 4MB size ($2^{30}/2^{8}$). Each block is subdivided into 256 sub blocks of size 4KB. Each sub block is split into l bits of symbol in the Galois Field.

The symbols will map to the integer value between 0 and $2^{l} - 1$. The exponentiation over the generator generates the unique values from 1 to $2^{l} - 1$ Mapping is done with unique symbols so it is very hard to replace x in this equation $a = g^{x}$ by any other symbol to have the same answer a. To avoid the replay attack by the cloud server mobile user generates random key for each time of verification.

Table 4.1 Time complexity of the algorithms in Client side and Server Side.

Algorithm	**Client computation**	**Server Computation**
Tag generation	O(logn)	-
Challenge	O(1)	-
Proof	-	O(n)
Verification	O(1)	-

The framework is implemented in Intel Core i5 – 2450M CPU with 2.50 GHZ. We used Dev C++ for Elliptic Curve Cryptography encryption and MATLAB 7.0 for Galois Field Computations.

5. Conclusion

In this paper a novel framework has been proposed for integrity checking of the remote files in the cloud server using Galois Field Computations. The framework needs only l bits of the tag for verification and resistant to replay attack by any intruder. This protocol saves the computation cost of the mobile client in terms of verification. In future enhancements the part of the protocol can be migrated to the cloud server thus by saving the computation cost of the mobile client.

6. References

1. Dinh HT, Lee C, Niyato D, Wang P, A survey of mobile cloud computing: architecture, applications, and approaches. Wireless Communication Mobile Computing, 2011.
2. Mehdi Sookhak, HamidTalebian, EjazAhmed, AbdullahGani, Muhammad Khurram Khan, A review on remote data auditing in single cloud server: Taxonomy and open issues, Journal of Network and Computer Applications 43(2014)121–141
3. Ateniese G,Burns R,Curtmola R,Herring J,Kissner L,Peterson Z, Provabledata possession at untrusted stores. In Proceedings of the 14th ACM conference on computer and communications security, Alexandria,Virginia, USA:ACM;2007. p. 598–609.
4. Fernando.N, Loke.SW, Rahayu.W. Mobile cloud computing: A survey. Future Generation Computer Systems 2013. (29) 84–106.
5. Kan Yang, Xiaohua Jia, An Efficient and Secure Dynamic Auditing Protocol for Data Storage in Cloud Computing, IEEE Transactions On Parallel And Distributed Systems, Vol. 24, No. 9, September 2013.
6. Lanxiang Chen, Using algebraic signatures to check data possession in cloud storage, Future Generation Computer Systems 29 (2013) 1709–1715.
7. Yong Yu Man Ho Au Yi Mu Shaohua Tang Jian Ren Willy Susilo Liju Dong, Enhanced privacy of a remote data integrity-checking protocol for secure cloud storage,
8. Yong Yu, Yafang Zhang, Jianbing Ni , Man Ho Au, Lanxiang Chen, Hongyu Liu, Remote data possession checking with enhanced security for cloud storage
9. Kalpana.V, Meena.V, Study on Data Storage Correctness Methods in Mobile Cloud Computing Indian Journal of Science and Technology, Volume 8(6), 495–500, March 2015.
10. Lanxiang Chen ,Shuming Zhou, Xinyi Huang, Li Xu , Data dynamics for remote data possession checking in cloud storage, Computers and Electrical Engineering 39 (2013) 2413–2424.
11. Mehdi Sookhak , Abdullah Gani , Muhammad Khurram Khan , Rajkumar Buyya , Dynamic remote data auditing for securing big data storage in cloud computing, Information Sciences (2015), doi: 10.1016/j.ins.2015.09.004.

Corresponding Author:

Kalpana V*,

Email: *kalpana@cse.sastra.edu*