Secured Cloud Storage using Raptor Codes

Anupriya.A.S[1], S.Ananthi[2], Dr. S Karthik[3]

ABSTRACT: Cloud computing has developed as one of the most persuasive paradigms in the IT industry for last few years. To achieve the assertion of cloud data integrity and availability and impose the quality of dependable cloud storage service for users, an effective and flexible distributed scheme with explicit dynamic data support, including block update, delete, and append is designed. Erasure-correcting code in the file distribution preparation provides redundancy parity vectors and guarantees the data dependability. By utilizing the homomorphic token with distributed verification of erasure-coded data, this scheme achieves the integration of storage. The system safeguarded the security and dependability for cloud data storage under the aforementioned adversary model. Analysis shows the proposed scheme is highly efficient and resilient against Byzantine failure, malicious data modification attack, and even server colluding attacks. This paper enhances the work by using Raptor codes, which is an additional pre-coding on an appropriate LT-Code. In asymptotic settings, a class of universal Raptor Codes with linear encode/decode
time for which the failure probability converges to 1 polynomial fast in input size.

Keywords: Cloud computing, Raptor Codes, Security, Confidentiality, Integrity, Erasure-code, Data dependability.

Manuscript received july 1, 2013.

Anupriya.A.S, *M.E Computer Science and Engineering, Anna University/SNS College of Technology, Coimbatore-641035,Tamil Nadu(email:anupriyas191@gmail.com),India ,8870793165*

S.Ananthi, *Computer Science and Engineering, Anna University/SNS College of Technology, Coimbatore-641035, Tamil Nadu (email id:aananthi_s@yahool.com), ,India,9843021703.*

S.Karthik*, Computer Science and Engineering, Anna University/SNS College of Technology, Coimbatore-641035, Tamil Nadu (e-mail:profskarhtik@gmail.com).*

1. INTRODUCTION

Cloud computing is a process in which computing power, memory, infrastructure can be delivered as a service. A Cloud computing is a firm of network enabled services, guaranteed QoS, inexpensive computing infrastructures on demand with an easy and simple access. Cloud security is an emerging sub-domain of computer security, network and information security [8]. Security in cloud can be instrumented remotely by client where the data centres and protocols in the security objectives of the service provider are: i) confidentiality for securing the data access and transfer ii) auditability for checking whether the security aspect of applications has been tampered or not. Dimensions of cloud security have been totaled into three areas like security and privacy, compliance and legal issues.

2. BACKGROUND

2.1 Cloud Computing

Cloud Computing is a technology that uses the Internet and central remote servers to maintain data and applications. Cloud computing allows consumers and businesses to use applications without installation and access their personal files at any computer with internet access.

2.2 Cloud Characteristics

Cloud computing revelations the following key characteristics:

- Reliability is improved if multiple redundant sites are used, which makes well-designed cloud computing suitable

for business continuity and disaster recovery.

- Scalability and Elasticity via dynamic ("on-demand") provisioning of resources on a fine-grained, self-service basis near real-time, without users having to engineer for peak loads.
- Maintenance of cloud computing applications is easier, because they do not need to be installed on each user's computer and can be accessed from different places.
- Virtualization technology allows servers and storage devices to be shared and utilization be increased. Applications can be easily migrated from one physical server to another.

2.3 TPA

In order to unravel the problem of data integrity checking, many schemes are implemented under different systems and security models. In all these works, great efforts are made to design solutions that meet various requirements: high scheme efficiency, stateless verification, unbounded use of queries and retrievability of data, etc. Considering the role of the verifier in the model, all the schemes presented before fall into two categories: private auditability and public auditability. Although schemes with private auditability can achieve higher scheme efficiency, public auditability allows any one, not just the client (data owner), to challenge the cloud server for correctness of data storage while keeping no private information. Then, clients are able to delegate the evaluation of the service performance to an independent TPA, without devotion of their computation resources.

TPA is the third party auditor who will audit the data of data owner or client so that it will let off the burden of management of data of data owner. TPA eliminates the involvement of the client through the auditing of whether the data stored in the cloud are indeed intact, which can be important in achieving economies of scale for Cloud Computing. The released audit report would not only help owners to evaluate the risk of their subscribed cloud data services, but also be beneficial for the cloud service provider to improve their cloud based service platform .This public auditor will help the data owner that his data are safe in cloud. With the use of TPA, management of data will be easy and less burdening to data owner but without encryption of data, how data owner will ensure that his data are in a safe hand.

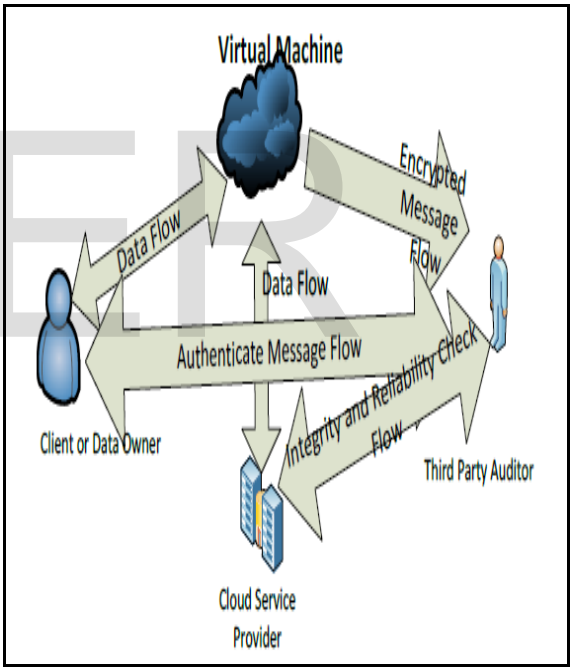

Figure 2.3: Architecture of Client, Third Party Auditor, Client service provider

2.4 Raptor Code

Most advanced forward error correction (FEC) code for data networks; Raptor codes provide protection against packet loss by sending additional repair data used to reconstruct “erased” or “lost” data. Erasure codes provide data recovery by transforming a message into a longer message,

allowing the original message to be recovered from a subset of the expanded message. Raptor recovers missing data packets with only minimal amounts of additional repair data and without requiring retransmission from the sender an efficiently and effectively providing reliability in data networks.

Using a Raptor code, an application can send and receive encoded data, and the fountain properties of the solution obviate, or greatly reduce the usage of feedback and retransmission protocols. This allows simpler, more scalable, and more efficient solutions. Raptor codes may be systematic or non-systematic.

In the systematic case, the symbols of the original message are included within the set of encoding symbols. Raptor codes are formed by the concatenation of two codes. A fixed rate erasure code, usually with a fairly high rate, is applied as a 'pre-code' or 'outer code'. the definition of the encoding cost of a Raptor Code differs slightly { it is the sum of the encoding cost of the pre-code divided by *k*, and the encoding cost of the LT code. Raptor Codes also require storage for intermediate symbols, so space consumption is another important performance parameter.

Need of Study

Moving data into the cloud offers great convenience to users since they don't have to care about the complexities of direct hardware management[3]. In order to achieve the assurances of cloud data integrity and availability and enforce the quality of cloud storage service, efficient methods that enable on-demand data correctness verification on behalf of cloud users have to be designed. However, while providing efficient cross server storage verification and data availability insurance, these schemes are all focusing on static or archival data.

This work proposes an effective and flexible distributed storage verification scheme with explicit dynamic data support to ensure the correctness and availability of users' data in the cloud[7]. Erasure correcting code in the file distribution preparation is to provide redundancies and guarantee the data dependability against Byzantine servers, where a storage server may fail in arbitrary ways. This construction drastically reduces the communication and storage overhead as compared to the traditional replication-based file distribution techniques.

2.5 OBJECTIVE AND SCOPE OF WORK

To develop a model that ensures the security and dependability for cloud data storage under the aforementioned adversary model, which aims to design efficient mechanisms for dynamic data verification and operation and achieve the following objectives:

- Storage correctness: to ensure users that their data are indeed stored appropriately and kept intact all the time in the cloud.
- Fast localization of data error**:** to effectively locate the malfunctioning server when data corruption has been detected.
- Dynamic data support: to maintain the same level of storage correctness assurance even if users modify, delete, or append their data files in the cloud.
- Dependability: to enhance data availability against Byzantine failures, malicious data modification and server colluding attacks, i.e., minimizing the effect brought by data errors or server failures.
- Lightweight: to enable users to perform storage correctness checks with minimum overhead.

This project is to develop an effective and flexible distributed storage verification scheme with explicit dynamic data support to ensure the correctness and availability of users' data in the cloud which rely on erasure correcting code in the file distribution preparation to provide

redundancies and guarantee the data dependability against Byzantine servers. This construction drastically reduces the communication and storage overhead as compared to the traditional replication-based file distribution techniques[2]. By utilizing the homomorphic token with distributed verification of erasure-coded data, this scheme achieves the storage correctness insurance as well as data error localization. Whenever data corruption has been detected during the storage correctness verification, this scheme can almost guarantee the simultaneous localization of data errors, i.e., the identification of the misbehaving server(s).

In order to strike a good balance between error resilience and data dynamics, the algebraic property of the token computation and erasure-coded data is explored and demonstrate how to efficiently support dynamic operation on data blocks, while maintaining the same level of storage correctness assurance. In order to save the time, computation resources, and even the related online burden of users, the extension of the proposed main scheme to support third-party auditing is provided, where users can safely delegate the integrity checking tasks to TPA and be worry-free to use the cloud storage services.

3. System Design

In the proposed use the raptor code instead of erasure code. Encode the input symbols using a traditional erasure correcting code, and then apply an appropriate LT-code to the new set of symbols in a way that the traditional code is capable of recovering all the input symbols even in face of a fixed fraction of erasures. To deal with the first issue, need to design the traditional code and the LT-code appropriately. Let Ω(x) be a linear code of block length and dimension, and let be a degree distribution. A Raptor code with parameters (k, C, Ω(x)) is an LT-code with distribution Ω(x) on symbols which are the coordinates of code words in C. The code C is called the pre-code of the Raptor code[15]. The input symbols of a Raptor code are the symbols used to construct the codeword in C consisting of n intermediate symbols.

The output symbols are the symbols generated by the LT-code from the n intermediate symbols. Typically, assume that is equipped with a systematic encoding, though this is not necessary. The definition of the encoding cost of a Raptor code differs slightly from that of a Fountain code. This is because the encoding cost of the pre-code has to be taken into account. The encoding cost of a Raptor code as E(c)/k+ Ω′(1), where E(c) is the number of arithmetic operations sufficient for generating a codeword in from the input symbols. The encoding cost equals the per-symbol cost of generating k output symbols[13]. The decoding cost of a decoding algorithm for a Raptor code is the expected number of arithmetic operations sufficient to recover the k input symbols, divided by k. As with the Fountain codes, this cost counts the expected number of arithmetic operations per input symbol.

Advantages of proposed system

1. Space: Since Raptor codes require storage for the intermediate symbols, it is important to study their space consumption. Count the space as a multiple of the number of input symbols. The space requirement of the Raptor code is 1/R, where R is the rate of the pre-code.
2. Overhead: The overhead is a function of the decoding algorithm used, and is defined as the number of output symbols that the decoder needs to collect in order to recover the input symbols with high probability, minus the number of input symbols. Measure the overhead as a multiple of the number of input symbols, so an overhead of ε, for example, means that (1+ ε)K output symbols need to be

collected to ensure successful decoding with high probability.

3. Cost: The cost of the encoding and the decoding process.

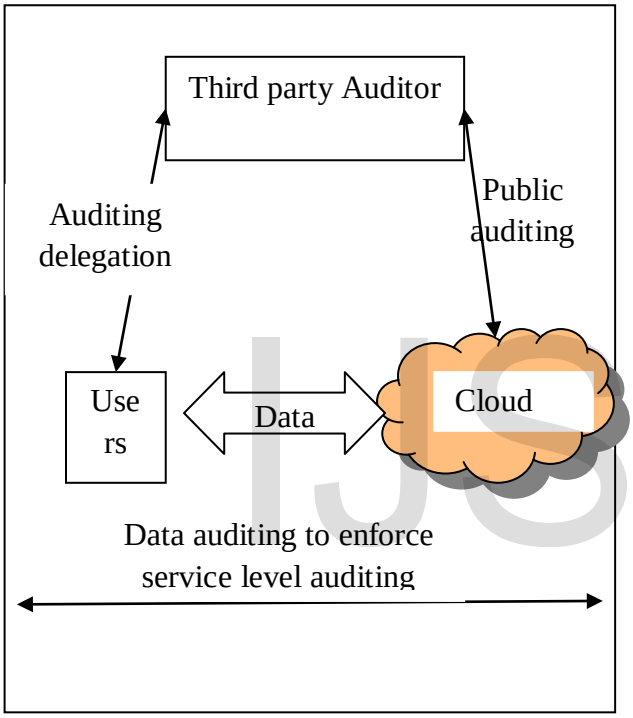

Figure 3.1: Cloud service architecture

4. MODULES DESCRIPTION

4.1 Developing a Cloud Network

Initially the basic network model for the cloud data storage is developed in this module. Three different network entities can be identified as follows: User: an entity, who has data to be stored in the cloud and relies on the cloud for data storage and computation, can be either enterprise or individual customers[1]. Cloud Server (CS): an entity, which is managed by cloud service provider (CSP) to provide data storage service and has significant storage space and computation resources (we will not differentiate CS and CSP hereafter). Third-Party Auditor: an optional TPA, who has expertise and capabilities that users may not have, is trusted to assess and expose risk of cloud storage services on behalf of the users upon request.

4.2 Implementing the file distribution and the token pre-computation

In this module erasure-correcting code to tolerate multiple failures in distributed storage systems is used[8]. The data file F redundantly across a set of n = m + k distributed servers. An (m; k) Reed-Solomon erasure-correcting code is used to create k redundancy parity vectors from m data vectors in such a way that the original m data vectors can be reconstructed from any m out of the m + k data and parity vectors[3]. By placing each of the m + k vectors on a different server, the original data file can survive the failure of any k of the m + k servers without any data loss, with a space overhead of k = m. For support of efficient sequential I/O to the original file, file layout is systematic, i.e., the unmodified m data file vectors together with k parity vectors is distributed across m + k different servers[5]. After performing the file distribution operation need to precompute the token. Token precomputation is the process for assuring the data storage correctness and data error localization simultaneously, this scheme entirely relies on the precomputed verification tokens. The main idea is as follows: before file distribution the user precomputes a certain number of short verification tokens on individual vector $G^{(j)}$ ($j \in \{1; \ldots; n\}$), each token covering a random subset of data blocks. Later, when the user wants to make sure the storage correctness for the data in the cloud, he challenges the cloud servers with a set of randomly generated block indices.

Algorithm for Token Precomputation.

1: procedure

2: Choose parameters l; n and function f, Ø;

3: Choose the number t of tokens;

4: Choose the number r of indices per verification;

5: Generate master key K_{PRP} and challenge key k_{chal};

6: for vector $G^{(j)}$, j ←1; n do

7: for round i ←1; t do

8: Derive $\alpha_i = fk_{chal}$ (i) and $k^{(i)}_{prp}$ from K_{PRP} .

9: Compute $v^{(j)}_i = \Sigma^r_{q=1} \alpha^{q_i} * G^{(j)}[\phi k^{(i)}_{prp}(q)]$

10: end for

11: end for

12: Store all the v_i's locally.

13: end procedure

4.3 Implementation of Correctness Verification and Error Localization

In this module, integrate the correctness verification and error localization (misbehaving server identification) in challenge-response protocol: the response values from servers for each challenge not only determine the correctness of the distributed storage, but also contain information to locate potential data error(s).

Correctness Verification and Error Localization

1: procedure CHALLENGE (i)

2: Recompute $\alpha_i = fk_{chal}$ (i) and $k^{(i)}_{prp}$ from K_{PRP} ;

3: Send $\{\alpha_i, k^{(i)}_{prp}\}$ to all the cloud servers;

4: Receive from servers: $\{R_i^{(j)} = \Sigma^r_{q=1} \alpha^{q_i} * G^{(j)}[\phi k^{(i)}_{prp}(q)] I 1<j<n\}$

5: for (j← m + 1; n) do

6: $R_i^{(j)} \leftarrow R_i^{(j)} - \Sigma^r_{q=1} f_{kj}(s_{iq,j}). \alpha^{q_i}, I_q \neq \phi k^{(i)}_{prp}(q)$

7: end for

8: if $((R_i^{(j)} \ldots\ldots R_i^{(m)}).P == (R_i^{(m+1} \ldots\ldots R_i^{(n))}))$ than

9: Accept and ready for the next challenge.

10: else

11: for (j ← 1; n) do

12: if $(R_i^{(j)} != V_i^{(j)})$ than

13: return server j is misbehaving.

14: end if

15: end for

16: end if

17: end procedure

Finally in this module we can identify any number of misbehaving servers for b ≤ (m + k). Also note that, for every challenge, each server only needs to send back an aggregated value over the specified set of blocks

4.4 Implementation of Error Recovery and Third party auditor

After identifying the misbehaving server from among all other servers need to recover those files. The user can always ask servers to send back blocks of the r rows specified in the challenge and regenerate the correct blocks by erasure correction, shown in Algorithm, as long as the number of identified misbehaving servers is less than k. The newly recovered blocks can then be redistributed to the misbehaving servers to maintain the correctness of storage[7].

Algorithm for Error Recovery

1: procedure

Assume the block corruptions have been detected among the specified r rows;

Assume s ≤ k servers have been identified misbehaving

2: Download r rows of blocks from servers;

3: Treat s servers as erasures and recover the blocks.

4: Resend the recovered blocks to corresponding servers.

5: end procedure

The TPA design is based on the observation of linear property of the parity vector blinding process[16]. However, this can be achieved either by blinding the parity vector or by blinding the data vector (we assume k < m). Thus, if blind data vector before file distribution encoding, then the storage verification task can be successfully delegated to third party auditing in a privacy-preserving manner. As TPA does not

know the secret blinding key k_j($j \in (1; . . .;m)$, there is no way for TPA to learn the data content information during auditing process. Therefore, the privacy-preserving third party auditing is achieved.

4.5 Providing dynamic data operation support

In this module provided the dynamic data operation support to user. Normally there are four categories of operation available Update operation, delete operation, append and insert operation. In update operation to update the existing or already available blocks of data in servers in this operation the user must priory know about the data block which is going to modify or alter[9]. Use the master key to perform that action to update the existing file.

In Delete operation first define the data blocks that are need to remove from the data server and after remove such file from the server we have to rearrange the remaining data blocks in the storage. The Append and the Insert are the same operation but in append add new data's to already existing server and the insert operation is we embedding the data for already existing data[10]. Master key is the basic need for all dynamic data support operations performing in data servers.

4.6 Implementation of Raptor code

Encode the input symbols using a traditional erasure correcting code, and then apply an appropriate LT-code to the new set of symbols in a way that the traditional code is capable of recovering all the input symbols even in face of a fixed fraction of erasures. To deal with the first issue, design the traditional code and the LT-code appropriately. Let $\Omega(x)$ be a linear code of block length and dimension, and let be a degree distribution. A Raptor code with parameters (k, C, $\Omega(x)$) is an LT-code with distribution $\Omega(x)$ on symbols which are the coordinates of code words in C[11]. The code C is called the pre-code of the Raptor code. The input symbols of a Raptor code are the symbols used to construct the codeword in C consisting of n intermediate symbols. The output symbols are the symbols generated by the LT-code from the n intermediate symbols.

Typically, assume that is equipped with a systematic encoding, though this is not necessary. The definition of the encoding cost of a Raptor code differs slightly from that of a Fountain code. This is because the encoding cost of the pre-code has to be taken into account. Define the encoding cost of a Raptor code as $E(c)/k+ \Omega'(1)$, where E(c) is the number of arithmetic operations sufficient for generating a codeword in from the input symbols. The encoding cost equals the per-symbol cost of generating k output symbols[15]. The decoding cost of a decoding algorithm for a Raptor code is the expected number of arithmetic operations sufficient to recover the k input symbols, divided by k. As with the Fountain codes, this cost counts the expected number of arithmetic operations per input symbol.

CONCLUSION

In the cloud data storage, users store their data and no longer posses the data locally. In the distributed cloud servers, the correctness and availability of the data files being stored. One of the key issues is to effectively detect any unauthorized data modification and corruption. The Third Party Auditing allows to save the time and computation resources with reduced online burden of users. In the proposed system the raptor code is used instead of erasure code. Encode the input symbols using a traditional erasure correcting code, and then apply an appropriate LT-code to the new set of symbols in a way that the traditional code is capable of recovering all the input symbols even in face of a fixed fraction of erasures.

REFERENCES

[1]Ateniese G, Burns R, Curtmola R, Herring J, Kissner L, Peterson Z, and Song D (2007), "Provable data possession at untrusted stores," in Proc. of CCS'07. New York, NY, USA: ACM, pp. 598–609

[2] Ateniese G, Pietro R.D, Mancini L.V, and Tsudik G, "Scalable and efficient provable data possession," in Proc. of SecureComm'08. New York, NY, USA: ACM, 2008, pp.1-10.

[3]Black.J,Halevi.J, Krawczyk.K, Krovetz.T, and Rogaway.P(1999), UMAC: Fast and secure message authentication.In *CRYPTO*, volume 1666 of *LNCS*, pages 216–233.

[4] Bowers K.D, Juels A, and Oprea A (2009), "Hail: A high-availability and integrity layer for cloud storage," in Proc. of CCS'09. Chicago, IL, USA: ACM, pp. 187–198.

[5]Castro.M and Liskov.B(2002), "Practical Byzantine Fault Tolerance and Proactive Recovery," ACM Trans. Computer Systems, vol. 20, no. 4,pp. 398-461.

[6] Chang E.C, and Xu J (2008), "Remote integrity check with dishonest storage server," in Proc. of ESORICS'08. Berlin, Heidelberg: Springer-Verlag, pp. 223–237.

[7] Chandran S. and Angepat M.(2010), "Cloud Computing: Analyzing the risks involved in cloud computing environments," *in Proceedings of Natural Sciences and Engineering*, Sweden, pp. 2-4.

[8] Cong Wang,Qian Wang,Kui Ren Ninig Cao and Wenjing Lou(2012),"Towards Secure and Dependable storage services in cloud computing",IEEE Transaction on service computing,vol 5,no 2.

[9] Dalia Attas and Omar Batrafi (2011)," Efficient integrity checking technique for securing client data in cloud computing".

[10]Jaison Vimalraj.T,M.Manoj (2011),"Enabling Public Verifiability and Data Dynamics for Storage Security in Cloud Computing".

[11] Kayalvizhi S,Jagadeeswari (2012),"Data Dynamics for Storage Security and Public Auditability in Cloud Computing", February 10.

[12] Metri P. and Sarote G. (2011), "Privacy Issues and Challenges in Cloud computing," International Journal of Advanced Engineering Sciences and Technologies, vol. 5, no. 1, pp. 5-6.

[13]Ren.K, Wang.C, and Wang.Q (2012), "Security Challenges for the Public Cloud," IEEE Internet Computing, vol. 16, no. 1, pp. 69-73.

[14]Srinivas.D(2011), "Privacy-Preserving Public Auditing In Cloud Storage Security", November .

[15] Shah M. A, Baker. M, Mogul J. C, and Swaminathan.R (2007), "Auditing tokeep online storage services honest," in *Proc. Of HotOS'07*., CA, USA: USENIX Association, pp. 1–6.

[16] Wang .C, Wang .Q, Ren .K, and Lou. W (2009), "Ensuring Data Storage Security in Cloud Computing," Proc. 17th Int'l Workshop Quality ofService (IWQoS '09), pp. 1-9, July.

LIST OF PUBLICATIONS

1.Anupriya.A.S,S.Ananthi,Dr.S.Karthick,"TPA Based Cloud Storage Security Techniques"in International Journal of Advanced Research in Computer Engineering and Technology(IJARCET),Volume 1,Issue 8,October 2012,ISSN:2278-1323