Research Paper

Volume : 2 | Issue : 2 | Feb 2013 • ISSN No 2277 - 8179

A Quasigroup Encryption Based Remote Data Integrity Checking Protocol for Secured Online Storage Services

Computer Science

KEYWORDS : Data Integrity, Data Dynamics, Public Verifiability

N. Saranya — Research Scholar, Department of Computer Science, Gobi Arts & Science College (Autonomous)

Dr.V.Thiagarasu — Associate Professor of Computer Science, Gobi Arts & Science College (Autonomous), Gobichettipalaaym-638453.

ABSTRACT

Cloud computing has become one of the essential tools of computing as a utility, where users can remotely store their data into the cloud such that they can utilize high quality applications and services from a shared pool of configurable computing resources. Remote Data Checking (RDC) is a technique by which clients can establish that data outsourced at untrusted servers remains intact over time. Moreover, the auditing result not only ensures strong cloud storage correctness guarantee, but also simultaneously achieves fast data error localization, i.e., the identification of misbehaving server. The performance of the developed algorithm is evaluated against the RSA algorithm and it is observed that the proposed approach is very significant in terms of execution time, communication cost for checking the data integrity.

I. Introduction

Cloud computing aims to enable end-users to easily create and use software without a need to worry about the technical implementations and nitty-gritties such as the software physical hosting location, hardware specifications, efficiency of data processing [6]. Cloud computing provides the facility to access shared resources and common infrastructure, offering services on demand over the network to perform operations that meet changing business needs [2]. The location of physical resources and devices being accessed are typically not known to the end user. It also provides facilities for users to develop, deploy and manage their applications on the cloud, which entails virtualization of resources that maintains and manages itself [5].

The cloud computing provides users with a long list of advantages, such as provision computing capabilities; broad, heterogeneous network access; resource pooling and rapid elasticity with measured services [14].

Cloud storage enables users to remotely store their data and enjoy the on-demand high quality cloud applications without the burden of local hardware and software management [4].

Over time many big Internet based companies Amazon, Google etc have come to realize that only a small amount of their data storage capacity is being used. This has led to the renting out of space and the storage of information on remote servers or "clouds" [1].

There exist various motivations for cloud service providers to behave unfaithfully towards the cloud users regarding the status of the outsourced data [3].

Recently, the importance of ensuring the remote data integrity has been highlighted in this research work under different system and security models [9]. The proposed algorithm is useful to ensure the storage correctness without having users possessing local data, are all focusing on single server scenario and useful for quality-of-service testing [4].

Literature Survey

Remote data integrity checking is first introduced by Y.Deswarte et al, which independently propose RSA-based methods for solving this problem [9]. After that Shah et al. propose a remote storage auditing method based on pre-computed challenge-response pairs [11]. The auditor verifies both the integrity of the data file and the server's possession of a previously committed decryption key [16].

Privacy preservation and data integrity are two of the most critical security issues related to user data [8].

Problem Formulation

Denote by m the file that will be stored in the untrusted server, which is divided into n blocks of equal lengths:

$m = m_1, m_2 \ldots m_n$, where n= [|m|/1]. Here l is the length of each file block. Denote by fk (.) a pseudo-random function which is defined as:

$$f : \{0,1\}^k * \{0,1\}^{\log 2(n) \rightarrow \{0,1\}d} \quad (1)$$

in which k and d are two security parameters. Furthermore, denote the length of N in bits by |N|.

A remote data integrity checking protocol that includes the following five functions: Setup, TagGen, Challenge, GenProof and CheckProof.

SetUp $(1^k) \rightarrow$ (pk, sk): Given the security parameter k, this function generates the public key pk and the secret key sk. pk is public to everyone, while sk is kept secret by the client.

TagGen (pk, sk, m) → D_m: Given pk, sk and m, this function computes a verification tag D_m and makes it publicly known to everyone. This tag will be used for public verification of data integrity.

Challenge (pk, Dm,) → chal: Using this function, the verifier generates a challenge chal to request for the integrity proof of file m. The verifier sends chal to the server.

GenProof (pk, D_m, m, chal) → R: Using this function, the server computes a response R to the challenge chal. The server sends R back to the verifier.

CheckProof (pk, D_m, chal, R) → {"success", "failure"}: The verifier checks the validity of the response R. If it is valid, the function outputs "success", otherwise the function outputs "failure". The secret key sk is not needed in the CheckProof function.

A. Quasi Group Encryption

The encryption technique used in this approach is the quasi group encryption algorithm. The quasi group encryptor has significant data-scrambling properties and thus, it has effectively used in symmetric cryptography. The main aim of the scrambler is to enhance the entropy at the output, even in scenario where the input is constant.

Input data: $d_1, d_2, d_3, \ldots d_n$

Output data: $e_1, e_2, e_3, \ldots e_n$

The two matrices: R, S

Multiplier Elements: $q_1, q_2, q_3 \ldots q_n$

The indices: $I_1, I_2, I_3 \ldots I_n$

It should be that if Q is a quasigroup such that a_1, a_2, a_3... a_n belong to it then the encryption operation QE, which is defined over the defined elements, maps those elements to another vector b_1,b_2,b_3....b_n such that the elements of the resultant vector also belong to the same quasigroup.

The mathematical equation used for encryption (basic level) is defined by:

$E_a (a_1, a_2, a_3... a_n) = b_1, b_2, b_3....b_n$ (2)

where the output sequence is defined by:

$b_1 = a*a_1$

$b_i = b_{i-1} * a_i$

where i increments from 2 to the number of elements that have to be encrypted, and a is the hidden key. Equation (2) describes a typical single level quasigroup encryptor.

It is assumed that the initial input data given by the vector a_1,a_2. a_3.a_4,a_5,a_6. It is mapped to the vector b_1,b_2,b_3,b_4,b_5,b_6 by equation (2). The following steps are used during the process of encryption:

$b_1 = a * a_1 = 2 * 2 = 1$

$b_2 = b_1 * a_2 = 1 * 4 = 4$

$b_3 = b_2 * a_3 = 4 * 1 = 4$

$b_4 = b_3 * a_4 = 4 * 2 = 5$

$b_5 = b_4 * a_5 = 5 * 3 = 1$

$b_6 = b_5 * a_6 = 1 * 3 = 2$

r and s given by the following equations

$E_{h1, h2, h3....hn} (a_1, a_2, a_3... a_n) = e_1, e_2, e_3....e_n$ (3)

where

$e_1 = a*a_1$ and $e_i = e_{i-1} *a_i$ (4)

In the above equation, the incoming data is first mapped through the first multiplier element h_1 then the resultant data is mapped taking into account the second multiplier element h_2. This process continues till all the multiplier elements are exhausted.

$b_1 = h_1*a_1$; $b_2 = b_1*a_2$; ..... $b_n = b_{n-1}*a_n$

$c_1 = h_2*b_1$; $c_2 = c_1*b_2$...$c_n = c_n\text{-}1 *b_n$...

$e_1 = h_n*s_1$; $_{e2} = e_1*s_2$....$e_n = e_n\text{-}1*s_n$ (5)

where the vector (h_1,h_2,h_3...h_n) comprises of all the multiplier elements.

IV.The Proposed Remote Data Integrity Checking Protocol

The proposed protocol is implemented in java which has the following functions SetUp, TagGen, Challenge, GenProof and CheckProof, as well as functions for data dynamics. In the following, the former five functions of the proposed protocol are presented.

SetUp(1^k) → (pk, sk): Let N = pq be one publicly known Quasi group modulus, in which p = 2p' + 1, q = 2q' + 1 are two large primes. The term p' and q' are also primes. In addition, all the quadratic residues modulo N form a multiplicative cyclic group, which is denoted by QR_N. Denote the generator of QR_N by g.1 Since the order of QR_N is p'q', the order of g is also p'q'. Let pk = (N, g) and sk = (p, q). pk is then released to be publicly known to everyone, and sk is kept secret by the client.

TagGen(pk, sk,m) →D_m: For each file block m_i, i ∈ [1, n], the client computes the block tag as $D_i = (g^{mi}) \bmod N$.

Let $D_m = \{D_1, D_2....D_n\}$. After finishing computing all the block tags, the client sends the file m to the remote server, and releases Dm to be publicly known to everyone.

Challenge(pk, D_m) → chal: In order to verify the integrity of the file m, the verifier generates a random key r ∈ [1, $2^k 2^k - 1$] and a random group element s € ZN \ {0}. The verifier then computes $g_s = g^s \bmod N$ and sends Chal = <r,gs > to the server.

GenProof (pk, D_m D_m, m, chal) → R: When the server receives Chal = r, g_s, it generates a sequence of block indexes a_1,a_2....a_n by calling $f_r(i)$ for i ∈ [1, n] iteratively. Then the server computes $R = (g_s)^{\sum_{i=1}^{n} a_i m_i} \bmod N$ and sends R to the verifier.

CheckProof(pk,Dm, chal,R) → {"success", "failure"}: When the verifier receives R from the server, she computes {ai}i=1,...,n as the server does in the GenProof step. Then the verifier computes P and R' as follows:

$P = \prod_{i=1}^{n} (D_i^{ai} \bmod N) \bmod N$

$R' = p^s \bmod N$

After that the verifier checks whether R' =R. If R' =R, output "success". Otherwise the verification fails and the verifier outputs "failure".

V. Correctness And Security Analysis

The proposed protocol is correct in the sense that the server can pass the verification of data integrity as long as both the client and the server are honest. These two theorems together guarantee that, assuming the client is honest, if and only if the server has access to the complete and uncorrupted data, it can pass the verification process successfully. Finally it shows that the proposed protocol is private against third party verifiers.

EXPERIMENTAL RESULTS

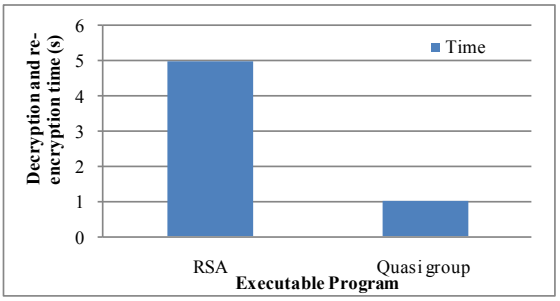

Figure 1 Comparison of Decryption and Re-Encryption Time

Figure 1 represents the comparison of decryption and re-encryption time. It is observed from the figure 1 that the Decryption and re-encryption time by the proposed quasi group encryption scheme is very less when compared with Reed Solomon code encryption scheme.

VI. CONCLUSION

This paper proposes a new remote data integrity checking protocol for cloud storage. The protocol is suitable for providing integrity protection of customers' important data. The protocol supports data insertion, modification and deletion at the block level, and also supports public verifiability. The protocol is proved to be secure against an untrusted server. It is also private against third party verifiers. Both theoretical analysis and experimental results demonstrate that the proposed protocol has very good efficiency in the aspects of communication, computation and storage costs.

REFERENCE

[1] Yashaswi Singh, Farah Kandah, Weiyi Zhang, "A Secured Cost-effective Multi-Cloud Storage in Cloud Computing", IEEE INFOCOM Workshop on Cloud Computing, 2011. | [2] Mache Creeger, "Cloud Computing: An Overview", ACM Queue, vol. 7, no. 5, pp. 2-4, 2009. | [3] P. Mell, T. Grance, "Draft NIST working definition of cloud computing", Referenced on June. 3rd, 2009, Online at http://csrc.nist.gov/groups/SNS/cloud-computing/index.html, 2009. | [4] Cong Wang, Qian Wang, Kui Ren, Ning Cao, and Wenjing Lou, "Towards Secure and Dependable Storage Services in Cloud Computing", 7th IEEE International Workshop on Quality of Service (IWQoS'09). | [5] Srinivasa Rao V, Nageswara Rao N K and E Kusuma Kumari, "Cloud Computing: An Overview", Journal of Theoretical and Applied Information Technology, 2009. | [6] F. Sebe, J. Domingo-Ferrer, A. Martinez-Balleste, Y. Deswarte, and J.-J. Quisquater, "Efficient remote data possession checking in critical information infrastructures," IEEE Trans. on Knowledge and Data Engineering, vol. 20, pp. 1034 –1038, 2008. | [7] A. Juels and J. Burton S. Kaliski, "Pors: Proofs of retrievability for large files," in Proc. of CCS'07, Alexandria, VA, pp. 584–597, 2007. | [8] Z. Hao, S. Zhong, and N. Yu, "A privacy-preserving remote data integrity checking protocol with data dynamics and public verifiability," Suny Buffalo CSE department technical report 2010-11,2010. http://www.cse.buffalo.edu/tech-reports/2010-11.pdf. | [9] Y. Deswarte and J.-J. Quisquater, "Remote Integrity Checking," in IICIS'04, pp. 1–11, Kluwer Academic Publishers, 1 2004. | [10] D. L. G. Filho and P. S. L. M. Barreto, "Demonstrating data possession and uncheatable data transfer." Cryptology ePrint Archive, Report 2006/150, 2006. http://eprint.iacr.org/. | [11] M. A. Shah, M. Baker, J. C. Mogul, and R. Swaminathan, "Auditing to keep online storage services honest," in HotOS XI., Usenix, 2007. | [12] C. Wang, S. S.-M. Chow, Q. Wang, K. Ren, and W. Lou, "Privacy preserving public auditing for secure cloud storage." Cryptology ePrint Archive, Report 2009/579, 2009. http://eprint.iacr.org/. | [13] Y. Zhu, H. Wang, Z. Hu, G.-J. Ahn, H. Hu, and S. S. Yau, "Cooperative provable data possession." Cryptology eprint Archive, Report 2010/234, 2010. http://eprint.iacr.org/. | [14] Q. Wang, C. Wang, J. Li, K. Ren, and W. Lou, "Enabling public verifiability and data dynamics for storage security in cloud computing," in 14th ESORICS, Springer, September 2009. | [15] C. Wang, Q. Wang, K. Ren, and W. Lou, "Privacy-preserving public auditing for data storage security in cloud computing," in InfoCom2010, IEEE, March 2010. | [16] C. Wang, Q. Wang, K. Ren, and W. Lou, "Ensuring data storage security in cloud computing," in IWQoS'09, pp. 1 –9, july 2009. |