ISSN:1991-8178

Australian Journal of Basic and Applied Sciences

Journal home page: www.ajbasweb.com

Optimal Cloud Storage and Access Methods for Temporal Cloud Databases

G. Ambikesh, S. Muthurajkumar, M. Vijayalakshmi, A. Kannan

Department of Information Science and Technology, College of Engineering Guindy, Anna University, Chennai, India.

ARTICLE INFO

Article history:
Received 12 November 2014
Received in revised form 26 December 2014
Accepted 29 January 2015
Available online 10 February 2015

Keywords:
Cloud computing, data staging, temporal data storage.

ABSTRACT

Cloud data storage has the biggest challenge on the maintenance of data integrity at untrusted servers. Moreover in such systems, failures may occur frequently and data errors by clients may also be made for their own benefit. In order to provide effective data integrity and security, it is necessary to propose new storage and retrieval algorithms for cloud databases. In addition, the temporal nature of cloud data necessitates the use of temporal constraints for providing effective database services. In this paper, we propose a new time oriented data staging algorithm for effective storage of application data in the cloud using merkle tree with temporal constraints. Also, we propose a new data retrieval algorithm based on temporal constraints. The main advantage of these proposed algorithms is that they reduce the retrieval time and also the storage cost.

To Cite This Article: G. Ambikesh, S. Muthurajkumar, M. Vijayalakshmi, A. Kannan., Optimal Cloud Storage and Access Methods for Temporal Cloud Databases. ***Aust. J. Basic & Appl. Sci.,*** *9(6): 24-29, 2015*

INTRODUCTION

Cloud computing helps to provide cost effective services by providing additional processor and memory features. Storing data in cloud computing environment provides more convenience to the database users. Moreover, cloud services are accessed through World Wide Web and provide hence a cloud database a large amount of storage space. The cloud computing platform eliminates the responsibility of local database administrations from the complex task to maintain data. In such a scenario, the users depend only on the cloud provides for the availability and integrity of the data. The advantages of cloud databases include reducing cost and storage. In addition, it motivates to focus on core competencies instead of infrastructures management. However, the existing storage structures are not sufficient store the temporal data obtained from many applications in the cloud. Therefore, it is necessary to propose a new storage structures along with effective retrieval methods to efficiently store and retrieval the temporal data.

Another important challenge to be addressed in cloud databases is the lack of security in untrusted environments including servers. In such a scenario, it is necessary to provide a trust based storage and retrieval technique which can handle temporal data effectively. In this paper, we propose a new Merkle tree based storage structure for storing the temporal data effectively by applying temporal constraints. In addition, we propose a new search technique for effective retrieval of temporal data from the cloud. Finally, we propose a trust based security maintenance algorithm for enhancing the security of cloud databases. This includes the propose it trust based temporal access control techniques in data retrieval.

Literature survey:

There are many existing works that discuss about data staging algorithm (Jakobsson, M., 2003; Wang, C., 2010; Armbrust, M., 2009; Wang, Y., 2013) for secure data storage which are used to send a common message to a group of users. Ateniese *et al*. (2007) proposed a "provable data possession" model that considers the public auditability in their to ensure the possession of data in untrusted storage servers for providing secured storage. However, the system has a limitation by imposing priori bound on the number of queries. In addition data security algorithms (Juels, A., B.S. Kaliski, 2007; Szydlo, M. Merkle, 2004; Nurmi, D., 2009) described a "proof of retrievability" model in which spot-checking and error-correcting codes have been used to ensure both "possession" and "retrievability" of data files on archive service systems. In their model, some special blocks termed as "sentinels" are randomly embedded into the data files which are used for detection. The data file is further encrypted to protect the positions

Corresponding Author: G. Ambikesh, Department of Information Science and Technology, College of Engineering Guindy, Anna University, Chennai, India.

of these special blocks. In addition, public auditability is not supported in that scheme.

Shacham and Waters (2008) developed two schemes for secured cloud storage in which they redundantly encode a file with an erasure code. Then they applied an audit that probabilistically ensures that enough blocks are retrievable to reconstruct the file. In their schema, the encoder embeds special blocks into the data file before it is encrypted. Moreover, these blocks are derived from a file containing used details for which the verifier keeps the key. During an audit, the verifier requests a set of special blocks that are not used before and checks them for correctness. However, in their model the verifying party must be trusted and even multiple trusted machines cannot independently verify the blocks.

The "provable data possession" model proposed by Wang *et al*. 2010 describes a scheme to address the concerns in Juels and Kaliski (2007) schema. In this system each block of the redundantly encoded data is stored along with a Message Authentication Code (MAC). They also used a third parity audited for providing additional security. Even though, their model is cost effective, the security needs further improvements. In order to overcome the limitation of the existing cloud databases, a new intelligent and secure approach is proposed in this paper for effective storage and retrieval of cloud data.

System architecture:

Fig.1, shows the overall architecture of the system discussed in this paper. It consists of eight major components namely client, server, trusted TPA, security manager, temporal constraint manager, cloud server, knowledge base and cloud database. Moreover, N number of clients and M number of servers are present in this model in such a way that the N servers can interact with M clients. These interactions take place in the cloud environment in the presence of cloud databases and a cloud data manager. All the data stored in the cloud database are accessed from the server through the client. The client communicates to the cloud database through server.

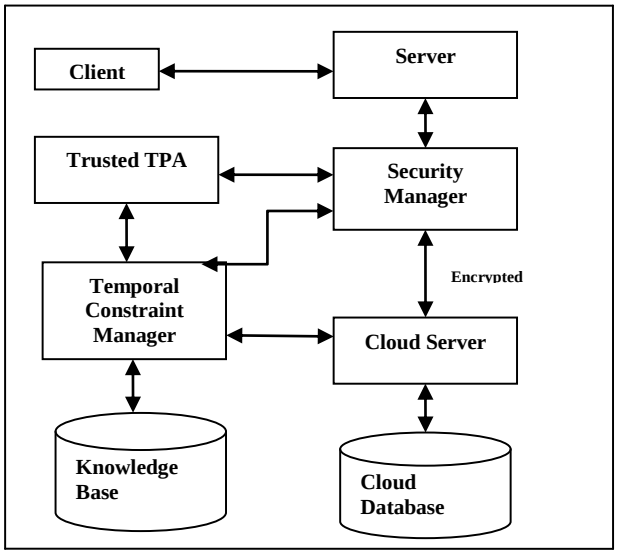

Fig. 1: Architecture Diagram.

A server may receive request from many different clients in very short period of time. A server host runs one or more server programs which share the resources with clients. Database can access to clients from the cloud and delivered to users on demand through server provided by cloud database servers. By using cloud computing facilities, cloud database can achieve optimized scaling, high availability, multi-tenancy and effective resource allocation. The server communicates to security manager for data storing and retrieving.

Security manager communicates with server and at the same time it can also communicate with cloud server, trusted TPA and temporal constraint manager. Trusted TPA is used to compute and maintain trust to secure data. Temporal constraint manager is used not only to check data integrity ans security but also to store and retrieve data from knowledge base to cloud database via cloud server.

A cloud server is a logical server that is built, hosted and delivered through a cloud computing platform over the Internet. Cloud server communicates to server through security manager and it also communicates with knowledge base through temporal constraint manager for data transmission. A cloud server may also be called a virtual server or virtual private sever.

Cloud database is connected to cloud server for storing data. Cloud databases can offer significant advantages over their traditional counterparts, including increased accessibility, automatic failover and fast automated recovery from failures, automated

on-the-go scaling, minimal investment and maintenance of in-house hardware, and potentially better performance. At the same time, cloud databases have their share of potential drawbacks, including security and privacy issues as well as the potential loss of or inability to access critical data in the event of a disaster or bankruptcy of the cloud database service provider.

Proposed intelligent temporal secure data staging algorithm:

The proposed Intelligent Temporal secure Data Staging Algorithm contains three phases namely Intelligent Temporal Multiple Copies without Constraints, Temporal Constraints on database tables and Intelligent Temporal and security constraints for entire database.

Intelligent temporal multiple copies without constraints:

In this phase, a new intelligent multicopy algorithm is proposed in this paper to handle multiple distinct data items. In this algorithm, n is the number of requests in sequence, m is the number of processing nodes, k means number of distinct data items, j is the j^{th} data item, and t_i is the i^{th} time stage. This algorithm find the shortest path from source node p_u^j to designation node p_v^j that has the request point for item j at stage t_u. The shortest path is defined on a subset instead of the fully connected network. It is necessary to update the costs for the request points in the affected set due to of the side effects of the shortest path computation. The steps of the proposed algorithm are as fellow's

Step1: Find the shortest path $sp(p_u^j, p_v^j)$ from p_u^j to p_v^j in which based on edges of the graph is the node that has the request point for item j at stage t_u.

Step2: Find the minimum cost considering the temporal aspect and using the formula

Step3: Find the total cost F (i) using the formula $F(i) = \sum_{j=1}^{k} F_j(i)$.

Step4: Find the best path using decision rules based on length and time.

This algorithm finds the shortest path between source and designation using Dijkstra's algorithm. The decision rules used in step 4 of the algorithm are stored in the Knowledge. They are useful for finding the best path by optimizing distance and cost. This algorithm is used for find the shortest path based on temporal aspects and it is used in intelligent temporal multiple copies without constraints.

Temporal constraints on database tables:

In this phase, the temporal constraints on database tables for the distinct data items are applied. E_i^j is a request point node, j is the j^{th} data item and t_i is the i^{th} time stage.

Step1: Select the affected point set for all the affected request nodes.

Step2: Compute the shortest path of moving item j from node to.

Step3: Find the request point using the formula $E_i^j = \{E_\theta^j j \neq C$

Step4: Minimize the request between the source and destination node using shortest path *SP new* source selection.

Step5: Minimize the request $E_h^j, h \in A_1^j$, using E_i^j, *SP new* and constraints

Intelligent temporal and security constraints for entire database:

In this phase, intelligent temporal and security constraint for entire database are applied on the distinct data items. F is a data file to be outsourced, denoted as a sequence of n blocks $m_1, m_2, \ldots, m_i, \ldots, m_n \in z_p$ for some large prime *p* and t is a time point from a set of time point T. In this algorithm *MAC (.)* — message authentication code (MAC) function, defined as $k \times \{0,1\} * \rightarrow \{0,1\}^1$ where k denotes the key space. *H (.), h (.)* is a cryptographic hash functions.

Step 1: Find the bilinear map for storing the data items in Mekle tree blocks.

Step 2: Let G_1, *G, and* G_T be multiplicative cyclic groups of prime order *p*. Let g_1, g_2 *and* g_t be generators of G_1, *G, and* G_T respectively.

Step 3: A bilinear map is a map e: $G_1 X G \rightarrow G_T$ such that for all

Step 4: This bilinearity implies that for any $u_1, u_2 \in G_1, v \in G_2, t \in T$, $e(u_1.u_2, v, t) = e(u_1, v, t).e(u_2, v, t)$.

Step 5: Find the computable algorithm for computing e and the map should be nontrivial, i.e., e is nondegenerate: $e(g1, g2) \neq 1$

Hash functions have been used to verify any kind of data stored, handled and transferred between computers. The prominent goal of using hash functions on Temporal Merkle trees is to make sure that data blocks received are undamaged and unaltered, and even to check that the other host do not send fake blocks.

Performance analysis:

In this section, we discuss about the performance analysis of our proposed algorithm used in intelligent temporal with other existing algorithms. The proposed and existing algorithms have been implemented in JAVA for measuring the actual computation time to perform with constraint and without constraint operations. In order to measure the actual computation time taken for performing the intelligent temporal operation, we have used various

numbers of CPUs, Memory in the third parity side. Similarly, the same number of hardware components is used in the cloud service provider side also for measuring the computation time. The measured computation time for all the existing and proposed algorithm are included in Table 1 and Table 2.

Table 1, shows the comparison on the amount of storage by Data Staging without Temporal Constraints Algorithm (DSWOTSA) and Data Staging with Temporal Constraints Algorithm (DSWTSA) in 5 experiments with different number of request. The requests included genuine and malicious user request with a proportion 19:1. From Table 1, it can be observed that the proposed DSWOTSA model performs better when compared with DSWTSA model in restricting the users and provides more than 90% detection and prevention accuracy. This is due to the use of intelligent agents and effective key sharing techniques proposed and used in this model.

Table 1: Storage requirements analysis.

S. No.	Amount of storage (GB)	Time in data staging without temporal constraints (ms)	Time in data staging with temporal constraints (ms)
1	500	11.142	11.124
2	1000	11.148	11.126
3	1500	11.149	11.131
4	2000	11.153	11.136
5	2500	11.159	11.142

Table 2, shows the comparison on the amount of storage by Data Staging without agents and with agents 5 experiments with different number of request. The requests included genuine and malicious user request with a proportion 19:1. From Table 2, it can be observed that the proposed DSWOTSA model performs better when compared with DSWTSA model in restricting the users and provides more than 90% detection and prevention accuracy.

Table 2: Storage analysis for agents.

S. No.	Storage (GB)	Data staging time (ms)	Data staging time with agents (ms)
1	500	10.115	9.454
2	1000	10.448	9.121
3	1500	10.149	9.311
4	2000	10.155	9.153
5	2500	10.459	9.122

Fig. 3, shows the data stored by the users who were permitted by the Cloud Data Staging without Constraint Algorithm. From this figure, it is observed that the security level of data storage in Data Staging with Temporal Constraints Algorithm (DSWTCA) is 5% more than the security level of data storage in Data Staging without Temporal Constraints Algorithm (DSWOTCA). Moreover, our DSWTCA is more useful for securing data storage than DSWOTCA.

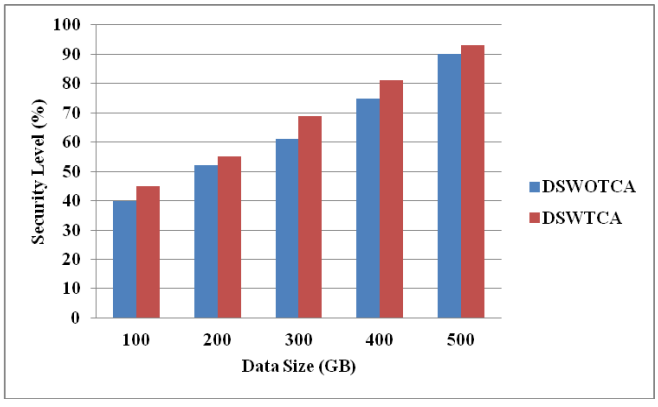

Fig. 3: Data storage security level using DSWOTCA and DSWTCA.

Fig. 4, shows the data stored by the users who were permitted by the Cloud Data Staging without Constraint Algorithm. From this figure, it is observed that the security level of data storage in Data Staging with Temporal Agent Temporal Constraints algorithm (DSWTATCA) is 10% more than the security level of data storage in Data Staging without Agent with Temporal Constraints Algorithm (DSWOAWTCA). Moreover, 10% of data store is more in comparison with the existing algorithm and hence the security is enhanced. This is due to the fact that temporal constraints are used effectively to check the abnormal users.

Fig. 4: Data storage security level using DSWTATCA and DSWOAWTCA.

Fig. 5, shows comparison between Conventional Application Time by Existing Data Staging without Temporal Constraint Algorithm (DSWOTCA) and Data Staging with Temporal Constraint Algorithm (DSWTCA) when the request is sent during time interval (t_1, t_2). From the implementation carried out in this model, it is observed that the time is reduced by 7% in DSWTCA when it is compared with DSWOTCA.

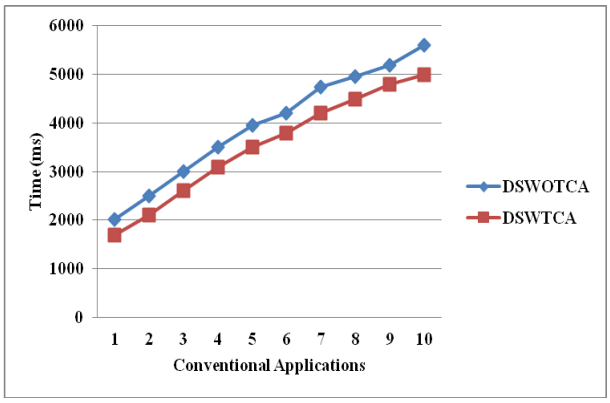

Fig. 5: Conventional applications time using DSWOTCA and DSWTCA.

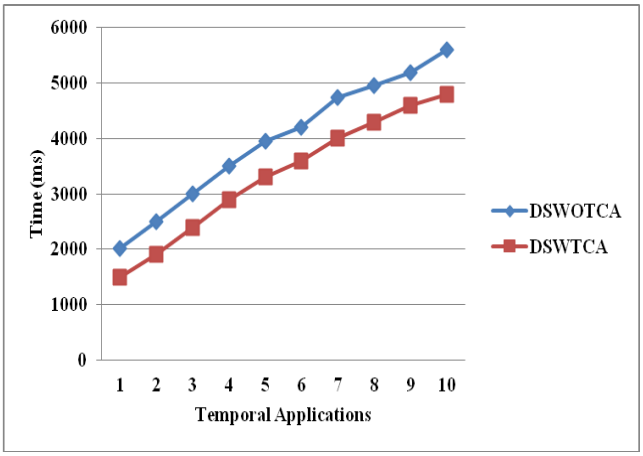

Fig. 6: Temporal applications time using DSWOTCA and DSWTCA.

Fig. 6, shows comparison between Temporal Application Time by Existing Data Staging without Temporal Constraint Algorithm (DSWOTCA) and Data Staging with Temporal Constraint Algorithm (DSWTCA) when the request is sent during time interval (t_1, t_2). From the implementation carried out in this model, it is observed that there is a difference in number of users by 25% in time who were data

storage and retrieval in comparison with the Temporal Data Staging Algorithm.

Conclusions:

In this paper, new algorithms are proposed for effective data staging of different data items in a fully connected cloud network environment so that it is possible to make easy cloud-based services with least cost. This optimal data storage and staging strategies are based on the temporal constraints which are used to minimize the total staging cost. We have validated the proposed algorithms by implementing these algorithms on a private cloud by conducting -experiments. From his implementation, it has been observed that the proposed data staging algorithms provide better storage and retrieval facilities. In addition, security is maintained in this work using trust management and temporal constraints. Further works in this direction can be the use of computations intelligent techniques for effective decision making.

REFERENCES

Ateniese, G., G.R. Burns, R. Curtmola, J. Herring, L. Kissner, Z. Peterson, D. Song, 2007. Provable Data Possession at Untrusted Stores. Proc. 14th ACM Conference Computer and Communication Security (CCS'07), pp: 598-609.

Juels, A., B.S. Kaliski, 2007. Proceedings: Proofs of Retrievability for Large Files. Proc. 14th ACM Conference Computer and Comm. Security (CCS '07), pp: 584-597.

Wang, Q., C. Wang, J. Li, K. Ren, W. Lou, 2009. Enabling Public Verifiability and Data Dynamics for Storage Security in Cloud Computing. Proceedings. 14th European ymp. Research in Computer Security (ESORICS '09), pp: 355-370.

Jakobsson, M., T. Leighton, S. Micali, M. Szydlo, 2003. Fractal Merkle Tree Representation and Traversal.

Shacham, H., B. Waters, 2008. Compact Proofs of Retrievability. Proceeding 14th International Conference. Theory and Application of Cryptology and Information Security: Advances in Cryptology (ASIACRYPT '08), pp: 90-107.

Szydlo, M. Merkle, 2004. Tree Traversal in Log Space and Time. Proceeding International Conference on the Theory and Application of Cryptographic Techniques.

Nurmi, D., R. Wolski, C. Grzegorczyk, G. Obertelli, S. Soman, 2009. The Eucalyptus Open-source Cloud-computing System. Proceeding International Conference on the Cluster Computing and the Grid.

Shacham, H., B. Waters, 2008. Compact Proofs of Retrievability. Proceeding International Conference on the Theory of Cryptography, February.

Wang, C., Q. Wang, K. Ren, W. Lou, 2010. Privacy-Preserving Public Auditing for Storage Security in Cloud Computing. Proceedings IEEE INFOCOM '10.

Armbrust, M., A. Fox, R. Griffith, V. Joseph, V. Katz, V. Konwinski, V. Lee, D.A. Patterson, A. Rabkin, I. Stoica, M. Zaharia, 2009. Above the Clouds: A Berkeley View of Cloud Computing. Technical Report UCB-EECS-2009-28, Univ. of California, Berkeley.

Wang, Y., B. Veeravalli, C. Tham, 2013. On Data Staging Algorithms for Shared Data Accesses in Clouds. IEEE Transactions on Parallel and Distributed Systems, 24(4): 825-838.