Enhancing Security and Performance by Fragmenting and Deduplication in cloud

Mr. Mahesh Kharde, Prof. Ashish Kumar
G.H.Raisoni College of Engineering and Management, Ahmednagar.

Abstract **- From last two decades, the appealing features of cloud computing has fueled integration of cloud in the human life. In cloud computing, data is been outsourced to 3rd party, which gives rise to security concerns. Due to attacks by other users and node within cloud, data compromise may occur. In this paper we propose Enhancing technique for cloud Security by Fragmenting and replicating data that collectively approaches the security and performance issues. In this methodology, we break file into pieces (Fragments), and replicate this fragments over the nodes in the cloud. Each nodes consist of single fragment of a particular file that relies us that even any attacker attacks a node, no meaningful information should be exposed to him. We also restrict attackers from guessing location of fragments by separating them using T-coloring algorithm. Data Deduplication plays an important role in reducing storage overhead in our proposed methodology. Since we do not use any traditional cryptography, we avoid performance overhead on data. We show that probability to identify any file is extremely low.**

Index Terms **- cloud security, Centrality, fragmentation, performance, replication.**

1 INTRODUCTION

CLOUD computing plays an important role in internet. It provides many features flexibility, on-demand services, elasticity, ubiquitous network accesses etc. But, the advantages of low-cost, flexibility and negligible management comes with security concerns. In cloud data is outsourced to third party which must be secure. Unauthorized access of data access by any other users and processes (whether accidental or deliberate) should be prevented [2]. Security plays very crucial role in adoption of cloud computing [2, 3]. Any weak entity can put the whole cloud at risk.

Issues in cloud computing may be due to cloud service offerings (weak authentication schemes, structured query language injection, etc.), core technology's implementation (session riding, VM (virtual machine) escape, etc.) and arising from cloud characteristics (Internet protocol vulnerability, data recovery vulnerability, etc.). In cloud, the storage space is shared among different entity (users). A multi-user virtualized environment may result in different issues like VM to escape the bounds of virtual machine monitor (VMM). The escaped VM can interfere with other VMs to have access to unauthorized data [4]. Similarly, cross-user virtualized network access may also compromise data privacy and integrity. Customer's private data can also leak due to improper media sanitization [5].

A cloud must ensure reliability, throughput and security. A key factor determining the throughput of a cloud that stores data is the retrieval time of the data [6]. In large-scale systems, the problems of data availability, data reliability and response time are dealt with strategies such as data replication [7]. However, placing replicas data all over the nodes increases the attack surface for that particular data. For instance, storing k replicas of a file in cloud instead of one replica increases the probability of a node holding file to be chosen as attack victim, from 1 n to k n, where n is the total number of nodes.

Attack	Description
Data Recovery	Rollback of VM to some previous state. May expose previously stored data.
Cross VM attack	Malicious VM attacking co-resident VM that may lead to data breach.
Improper media sanitization	Data exposure due to improper sanitization of storage devices.
E-discovery	Data exposure of one user due to seized hardware for investigations related to some other users.
VM escape	A malicious user or VM escapes from the control of VMM. Provides access to storage and compute devices.
VM rollback	Rollback of VM to some previous state. May expose previously stored data.

Fig. 1. Various attacks handled by methodology.

From the above discussion, we can summarized that both performance and security are critical for large-scale systems, such as clouds. In this paper, we collectively approach the concerns of performance and security. We present Enhancing cloud Security by Fragmenting and replicating data that fragments user files into pieces, replicate them at algorithmically determined location within cloud. To increase the performance we use deduplication algorithms. We take care that a successful attack on a single node will not reveal the locations of fragments in file sequence within the cloud. To keep the attacker unknown about the location of the file fragments and further enhance security, we select nodes in such a manner that they are not adjacent and are at certain distance from each other. We used T-coloring algorithm [8] for node separation.

To decrease data retrieval time, the nodes are selected by centrality algorithm which ensure decrease in access time. We use two phases for selection of the node. In phase one we select node for initial placement of fragments based on the centrality measures. In phase two, we replicate the fragments so that it is available even if any node fails

Our major contributions in this paper are as follows:

- We develop a scheme which fragments and replicates the data file over cloud nodes and increase both the security and performance.
- We provide deduplication scheme for minimizing storage cost.
- We ensure a controlled replication of the file fragments, where each of the fragments is replicated only once for the purpose of improved security.
- We do not rely on traditional cryptographic techniques for data security. The non-cryptographic nature of the proposed scheme makes it faster to perform the required operations (placement and retrieval) on the data.
- The proposed scheme ensures that even in the case of a successful attack, no meaningful information is revealed to the attacker.

2 RELATED WORKS

In [10], Author approach the multi user and virtualized related issues in cloud by utilizing the native access control and consolidated storage. It consist of Dike authorization architecture that combines tenant name space isolation and native access control. The proposed system works for object based file tenant file system, however in case of improper sanitization and malicious VM, there can be leakage of critical information. Our proposed system handles the leakage of critical information by fragmenting and storing fragments at multiple nodes.

Juels et at [9] in his paper presented technique to ensure freshness, availability and integrity of data in cloud. He used Iris file system for performing data migration. In his technique, file blocks, MAC codes and version numbers were stored at various level of tree. Authors proposed system heavily depend upon user for data confidentiality. Also in case of data tempering due to intrusion by other VM, loss was not decreased. Our methodology not stores all data in single node so we avoid and protect it from data compromise in case of successful attach. Also we do not rely on traditional cryptography for data security.

An optimal and secure placement of data object is presented in [6]. Here we divide encryption key into n shares and distribute within network. Here network is divided into clusters. A primary site is selected in each cluster that performs allocation within cluster. The scheme presented in [6] checks replication problem with security and access time improvement. This scheme only focuses on the security of encryption key. The file is not fragments and is handled as a single file. On other hand, in our methodology we fragment file and store it at multiple nodes.

In [16] Author explained, how client side deduplication helps in reducing consumption of storage space along with bandwidth used to transfer duplicated file. The author has used cryptographic techniques for encryption. Both symmetric and asymmetric techniques if encryption was used. The technique resisted to unauthorized access to data and it maintain privacy during sharing process. But it still had challenge to outsourcing the data into cloud.

3 PRELIMINARIES

3.1 Data Fragmentation

The security in cloud depends on security of nodes present in it. A successful intrusion in a single node may cause severe consequences, not only for specified node but also other nodes. If whole file is present in single node then it can reveal user data [11]. In case of homogeneous systems, the same flaw can be utilized to target other nodes within the system. We can reduce this risk by making fragments of a file and storing each fragments on different node [11 13]. So now intrusion on single node will provide access the certain portion of file which might not be of any significance. Moreover, if an attacker is unknown about the locations of the fragments, the probability of finding fragments on all of the nodes is very less. Let us consider a cloud with M nodes and a file with z number of fragments. Let s be the number of successful intrusions on distinct nodes, such that $s > z$. The probability [1] that s number of victim nodes contain all of the z sites storing the file fragments (represented by P(s,z)) is given as:

$$P(s,z) = \frac{\binom{s}{z}\binom{M-z}{s-z}}{\binom{M}{s}} \qquad (1)$$

Using Fragmenting technique in cloud, there are thousands of nodes, the probability for an attacker to obtain a considerable amount of data, reduces significantly. However, placing each fragment once in the system will increase the data retrieval time. [1] To improve the data retrieval time, fragments can be replicated in a manner that reduces retrieval time to an extent that does not increase the aforesaid probability.

3.2 Centrality

The objective of improved retrieval time in replication makes the centrality measures more important. The centrality of a node in a graph provides the measure of the relative importance of a node in the network. There are various centrality measures; for instance, closeness centrality, degree centrality, betweenness centrality, eccentricity centrality, and eigenvector centrality.

3.2.1 Eccentricity

The eccentricity of a node n is the maximum distance to any node from a node n [14]. A node is more central in the network, if it is less eccentric. Formally, the eccentricity can be given as:

$$E(v_a) = max_b d(v_a,v_b) \qquad (2)$$

Where $d(v_a,v_b)$ represents the distance between node *va* and node *vb*. It may be noted that in our evaluation of the strategies the centrality measures introduced above seem very meaningful and relevant than using simple hop-count kind of metrics.

3.3 T-coloring

Suppose we have a graph $G = (V,E)$ where V is vertex and E is edges and let T be set nonnegative integers including 0. The T coloring is a mapping function f from the vertices of V to the set of non-negative integers, such that $|f(x) - f(y)| \notin T$, where $(x,y) \in E$. The mapping function f assigns a color to a vertex V. In simple words, the distance between the colors of the adjacent vertices must not belong to T. Formulated by Hale [8], the T-coloring problem for channel assignment assigns channels to the nodes, such that the channels are separated by a distance to avoid interference

3.4 Data Deduplication

Data Deduplication is one of the hottest technology in storage area right now because it enables us to save bandwidth cost, storage cost and many more. In deduplication technique we perform following actions

1. Divide the input data into blocks or chunks.
2. Calculate a hash value for each block of data.
3. Use these values to determine if another block of the same data has already been stored.
4. Replace the duplicate data with a reference to the object already in the present.

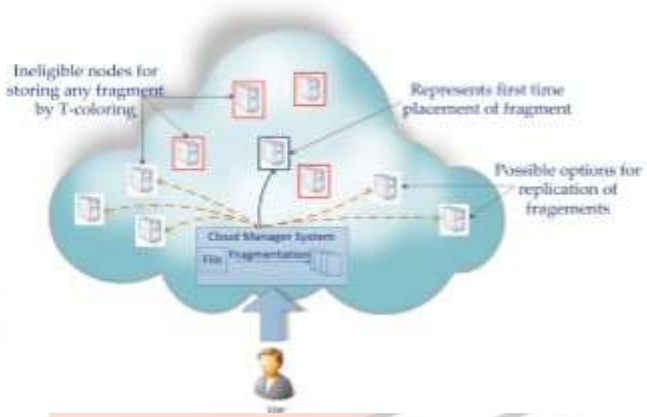

Fig. 2. The Proposed methodology

4 SYSTEM MODEL

Consider a cloud with M number of nodes consisting of its own storage capacity.

Let,

$c(i,j)$: Communication time between S^i and

Sj r_k^i : read request from S^i for O_k

w_k^i : write requests from S^i for O_k

P_k : Primary node

R_k : Replicate node

TABLE 1
Notations and their meanings

Symbols	Meanings
M	Total number of nodes in the cloud
N	Total number of file fragments to be placed
s_i	Size of S^i
O_k	k-th fragment of le
o_k	Size of O_k
S^i	i-th node
cen_i	Centrality measure for S_i
col_{S^i}	Color assigned to S^i
T	A set containing distances by which assignment of fragments must be separated

Our aim [1] is to minimize the overall total network transfer time or replication time (RT) replication cost (RC).The Replication time is composed of two factors: (a) time due to write requests. And (b) time due to read requests.

$$R_k^i = r_k^i o_k c\left(i, NN_k^i\right) \quad (3)$$

The total time due to the writing of O_k by S^i addressed to the P_k is represented as $W_k{}^i$ and is given:

$$W_k^i = w_k^i o_k \left(c\left(i, P_k\right) + \sum_{(j \in R_k), j \neq i} c\left(P_k, j\right) \right) \quad (4)$$

The overall RT is represented by:

$$RT = \sum_{i=1}^{M} \sum_{k=1}^{N} \left(R_k^i + W_k^i\right) \quad (5)$$

Replication increases the number of file copies within the cloud. Thereby, increasing the probability of the node holding the file to be a victim of attack. Replication and Security must be balanced so that any one service not lower the other.

Process: User sends the file to cloud for storing. The cloud manager(CM) after receiving file perform following operations a) Fragmentation b) First phase of node selection and storing one fragment in each selected node and c) second phase of fragment replication. The cloud manager keeps track of the fragments placement and is assumed to be secured.

Algorithm 1 Algorithm for finding arrays for fragments placement

Inputs and initializations:
$O = \{O_1, O_2, ..., O_N\}$
$o = \{sizeof(O_1), sizeof(O_2), ..., sizeof(O_N)\}$
$col = \{open\_color, close\_color\}$
$cen = \{cen_1, cen_2, ..., cen_M\}$
$col \leftarrow open\_color \forall i$
Compute:
For Graph G do
Calculate shortest Distance
using Dijkstra's algorithm
Split this array Graph in two arrays
{ open_color, close_color }
using Graph Coloring using 2 colors
open_color : nodes having color 1
close_color : nodes having color 2
foreach color array do
sort by shortest distance
store fragments in this two arrays
namely open$_c$olor and close$_c$olor
end if
End for =0

Algorithm 2 Algorithm for fragment placement

Compute:
for each $O_k \in O$ do
  Select $S^i | S^i \leftarrow indexof(max(cen_i))$
  if $col_{S^i} = open\_color$ *and* $s_i >= o_k$ then
    $S^i \leftarrow O_k$
    $col_{S^i} \leftarrow close\_color$
    $S^{i'} \leftarrow distance(S_i, T)$
    $col_S^{i'} \leftarrow close\_color$
  end if
End for =0

When the file is split into fragments, we perform fragment placement. While performing fragment placement we focus on both security and performance in terms of access time. If we place file to the central of all node, we will get better access time. So to reduce access time we use the concept of centrality. In current paper we use three types of centrality algorithm: (a) eccentricity centrality, (b) closeness, and (c) betweenness. However if all fragments are placed based on centrality in descending order, then there is possibility that adjacent node get selected for fragment placement. This can give attackers chance to identify next fragment of file. To avoid this we use the concept of T- coloring [8]. In this process we may loss some central nodes but we will achieve high security. So now if any attacker gets access to any node and obtain fragments , still he will not be able to determined other fragments of the file. The process of fragment placements is repeated until all of the fragments are placed at the nodes.

Now we are ready with fragments placed at their positions, now we require to replicate this fragments so that they are available even if concerned particular node fails. While placing replicate we take care of coloring also, so that adjacent node doesnt get sequenced fragment. The replication strategy is presented in algorithm 2

Algorithm 3 Algorithm for fragments replication

for each $O_k \in O$ do
  Select S^i *that has* $max(R_k^i + W_k^i)$
  if $col_{S^i} = open\_color$ *and* $s_i >= o_k$ then
    $S^i \leftarrow O_k$
    $s_i \leftarrow s_i - o_k$
    $col_{S^i} \leftarrow close\_color$
    $S^{i'} \leftarrow distance(S_i, T)$
    $col_S^{i'} \leftarrow close\_color$
  end if
End for

To handle the download request from user, the cloud manager collects all the fragments from the nodes and re-assemble them into a single file. Afterwards, the file is sent to the user.

5 RESULTS AND DISCUSSION

We compared the performance of the methodology with the algorithms discussed above. The behavior of the algorithms was studied by: (a) increasing the number of objects keeping number of nodes constant, (b) varying the read/write ratio. The aforesaid parameters are significant as they affect the problem size and the performance of algorithms [12].

5.1 Impact of increase in number of file fragments

When there is increase in number of fragments then there is a strain on the storage capacity which in turn impacts the selection if the nodes. To check impact on performance we increased the fragments to 50, 100, and 200. In general the workload generated showed that, increase in number of file fragments reduced the performance of the algorithms. So far methodology with eccentricity centrality maintains the supremacy on the other two centralities.

5.2 Impact of increase in the read/write ratio

Change in Read/Write ration affected the performance. As we increase number of reads, it would lead to need of more replicas of fragments in cloud. Increased in number of replicas decreased the communication cost associated with the reading of the fragments. However increased in number of writes demanded the replicas to be placed nearer to primary node.

6 CONCLUSION

We proposed methodology for cloud storage security, that collectively deals with performance and security with respect to retrieval time. The file was fragmented and replicated over multiple nodes. The node were separated by Coloring algorithm (T coloring algorithm). The fragmentation and replication with deduplication ensures no significant information is obtainable in case of successful attack. We also ensure that no single node stores more than one fragments of the file. Currently with this methodology, user has to upload a file, where fragments, replication and deduplication takes place.

REFERENCES

[1] Mazhar Ali, Kashif Bilal, Samee U. Khan, Bharadwaj Veeravalli, Keqin Li, Albert Y. Zomaya, *DROPS: Division and Replication of Data in Cloud for Optimal Performance and Security*, IEEE Transactions on Cloud Computing, Volume:PP , Issue: 99.

[2] A. N. Khan, M. L. M. Kiah, S. U. Khan, and S. A. Madani, *Towards Secure Mobile Cloud Computing: A Survey*, Future Generation Computer Systems, Vol. 29, No. 5, 2013, pp. 1278-1299.

[3] D. Sun, G. Chang, L. Sun, and X. Wang, *Surveying and analyzing security, privacy and trust issues in cloud computing environments*, Procedia Engineering, Vol. 15, 2011, pp. 2852 2856.

[4] W. A. Jansen, *Cloud hooks: Security and privacy issues in cloud computing*, In 44th Hawaii IEEE International Conference onSystem Sciences (HICSS), 2011, pp. 1-10.

[5] B. Grobauer, T.Walloschek, and E. Stocker, *Understanding cloud computing vulnerabilities*, IEEE Security and Privacy, Vol. 9, No. 2, 2011, pp. 50-57.

[6] M. Tu, P. Li, Q. Ma, I-L. Yen, and F. B. Bastani, *On the optimal placement of secure data objects over Internet*, In Proceedings of 19th IEEE International Parallel and Distributed Processing Symposium, pp. 14-14, 2005.

[7] D. Boru, D. Kliazovich, F. Granelli, P. Bouvry, and A. Y. Zomaya, *Energy-efficient data replication in cloud computing datacenters*, In IEEE Globecom Workshops, 2013, pp. 446-
451.

[8] W. K. Hale, *Frequency assignment: Theory and applications*, Proceedings of the IEEE, Vol. 68, No. 12, 1980, pp. 14971514.

[9] A. Juels and A. Opera, *New approaches to security and availability for cloud data*, Communications of the ACM, Vol. 56, No. 2, 2013, pp. 64-73.

[10] G. Kappes, A. Hatzieleftheriou, and S. V. Anastasiadis, *Dike: Virtualization-aware Access Control for Multitenant Filesystems*, University of Ioannina, Greece, Technical Report No. DCS2013-1, 2013.

[11] A. Mei, L. V. Mancini, and S. Jajodia, *Secure dynamic fragment and replica allocation in large-scale distributed file systems*, IEEE Transactions on Parallel and Distributed Systems, Vol. 14, No. 9, 2003, pp. 885-896.

[12] S. U. Khan, and I. Ahmad, *Comparison and analysis often static heuristics-based Internet data replication techniques*, Journal of Parallel and Distributed Computing, Vol. 68, No. 2, 2008, pp. 113-136.

[13] Y. Tang, P. P. Lee, J. C. S. Lui, and R. Perlman, *Secure overlay cloud storage with access control and assured deletion*, IEEE Transactions on Dependable and Secure Computing, Vol. 9, No. 6, Nov. 2012, pp. 903-916.

[14] M. Newman, *"Networks: An introduction, Oxford University Press,"* 2009.Press, 2009.

[15] A. N. Khan, M. L. M. Kiah, S. U. Khan, and S. A. Madani, *Towards Secure Mobile Cloud Computing: A Survey*, Future Generation Computer Systems, Vol. 29, No. 5, 2013, pp. 1278-1299.

[16] Amaresh, Manjunath G. S., *An Efficient, Secure Deduplication Data Storing In Cloud Storage Environment*, International Journal of Research in Engineering and Technology, Vol. 4, No. 4, 2015, pp. 2321-7308.