International Research Journal of Advanced Engineering and Science
ISSN: 2455-9024

Division and Replication of Data in Cloud for Optimal Performance and Security using Fragment Placement Algorithm

Ms. A. Sivasankari[1], Ms. D. Abirami[2], Mrs. K. Ayesha[3]
[1]Head of the Department, Department of Computer Science DKM College for women, Vellore, TamilNadu, India
[2]Department of Computer Science DKM College for women, Vellore, TamilNadu, India
[3]Assistant Professor Department of Computer Science, DKM College for Women (Autonomous), Vellore, TamilNadu, India

***Abstract*— *Outsourcing data to a third-party managerial control, as is done in cloud compute, gives rise to security concerns. The data compromise may occur due to attacks by other users and nodes within the cloud. Therefore, high security measures are required to protect data within the cloud. However, the employed security strategy must also take into account the optimization of the data retrieval time. In this paper, we propose Division and Replication of Data in the Cloud for Optimal Performance and Security (DROPS) that collectively approaches the security and presentation issues. In the DROPS methodology, we divide a file into fragments, and replicate the fragmented data over the cloud nodes. Each of the nodes stores only a single fragment of a particular data file that ensures that even in case of a successful attack, no meaningful information is revealed to the attacker. Moreover, the nodes storing the fragments, are separated with certain distance by means of graph T-coloring to prohibit an attacker of guessing the locations of the fragments. Furthermore, the DROPS methodology does not rely on the traditional cryptographic techniques for the data security; thereby relieving the system of computationally expensive methodologies. We show that the probability to locate and compromise all of the nodes storing the fragments of a single file is extremely low. We also compare the performance of the DROPS methodology with ten other schemes. The higher level of security with slight performance overhead was observed.*

***Keywords*— *Centrality, cloud security, fragmentation, replication, performance.*

I. INTRODUCTION

Cloud computing is characterized by on-demand self-services, ubiquitous net- work accesses, resource pooling, elasticity, and measured services [22, 8]. The aforementioned characteristics of cloud computing make it a striking candidate for businesses, organizations, and individual users for adoption [25].

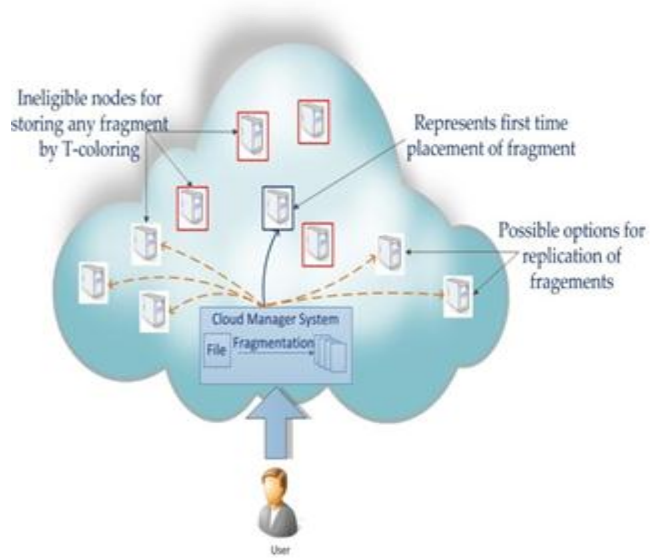
However, the benefits of low-cost, negligible management (from a users perspective), and greater flexibility come with increased security concerns [7].Security is one of the most crucial aspects among those prohibiting the wide-spread adoption of cloud computing [14, 19].

Cloud security issues may stem. For a cloud to be secure, all of the participating entities must be secure. In any given system with multiple units, the highest level of the system′s security is equal to the security level of the weakest entity [12]. Therefore, in a cloud, the security of the assets does not solely depend on an individual's security measures [5]. The neighboring entities may provide an opportunity to an attacker to bypass the users defenses.

The off-site data storage cloud utility requires users to move data in cloud's virtualized and shared environment that may result in various security concerns. Pooling and elasticity of a cloud, allows the physical resources to be shared among many users [22]. Moreover, the shared resources may be reassigned to other users at some instance of time that may result in data compromise through data recovery methodologies [22].

Furthermore, a multi-tenant virtualized environment may result in a VM to escape the bounds of virtual machine monitor (VMM). The escaped VM can interfere with other VMs to have access to unauthorized data [9]. Similarly, cross-tenant virtualized network access may also compromise data privacy and integrity. Improper media sanitization can also leak customers private data [5].

Our major contributions in this paper are as follows: on a single node must not reveal the locations of other fragments within the cloud. To keep an attacker uncertain about the locations


Ineligible nodes for storing any fragment by T-coloring
Represents first time placement of fragment
Possible options for replication of fragements
Cloud Manager System
File
Fragmentation
User
Fig. 1. The DROPS methodology.

The data outsourced to a public cloud must be secured. Unauthorized data access by other users and processes (whether accidental or deliberate) must be prevented [14]. As discussed above, any weak entity can put the whole cloud at risk. In such a scenario, the security mechanism must substantially increase an attacker's effort to retrieve a reasonable amount of data even after a successful intrusion in the cloud. Moreover, the probable amount of loss (as a result of data leakage) must also be minimized.

A cloud must ensure throughput, reliability, and security [15]. A key factor determining the throughput of a cloud that stores data is the data retrieval time [21]. In large-scale systems, the problems of data re- liability, data availability, and response time are dealt with data replication strategies [3]. However, placing replicas data over a number of nodes increases the attack surface for that particular data. For instance, storing *m* replicas of a file in a cloud instead of one replica increases the probability of a node holding file.

From the above discussion, we can deduce that both security and performance are critical for the next generation large-scale systems, such as clouds. Therefore, in this paper, we collectively approach the issue of security and performance as a secure data replication problem.

Fig. 2. The fragments.

Fragments and to further improve the security, we select the nodes in a manner that they are not adjacent and are at certain distance from each other. The node separation is ensured by the means of the T-coloring [6]. To improve data retrieval time, the nodes are selected based on the centrality measures that ensure an improved access time. To further improve the retrieval time, we judicially replicate fragments over the nodes that generate the highest read/write requests. The selection of the nodes is performed in two phases. In the first phase, the nodes are selected for the initial placement of the fragments based on the centrality measures. In the second phase, the nodes are selected for replication. The working of the DROPS methodology is shown as a high-level work flow in Fig. 1. We implement ten heuristics based replication strategies as comparative techniques to the DROPS methodology. The implemented replication strategies are:

Our major contributions in this paper are as follows:

- We present Division and Replication of Data in the Cloud for Optimal Performance and Security (DROPS) that judicially fragments user files into pieces and replicates them at strategic locations within the cloud. The division of a file into fragments is performed based on a given user criteria such that the individual fragments do not contain any meaningful information. Each of the cloud nodes (we use the term node to represent computing, storage, physical, and virtual machines) contains a A successful attack replicates the data file over cloud nodes.
- The proposed DROPS scheme ensures that even in the case of a successful attack, no meaningful information is revealed to the attacker.
- We do not rely on traditional for data security.
- The non-cryptographic nature of the proposed scheme makes it faster to perform the required operations.

The remainder of the paper is organized as follows. Section 2 provides an overview of the related work in the field. In Section 3, we present the preliminaries. The DROPS methodology is introduced in Section 4. Section 5 explains the experimental setup and results, and Section 6 concludes the paper.

We ensure a controlled replication of the fragments, where each of the fragments is replicated only once for the purpose of improved security.

II. Related Work

Juels *et al.* [10] presented a technique to ensure the integrity, freshness, and availability of data in a cloud. The data migration to the cloud is performed by the Iris file system. A gateway application is designed and employed in the organization that ensures the integrity and freshness of the data using a Merkle tree. The file blocks, MAC codes, and version numbers are stored at various levels of the tree.

The proposed technique in [10] heavily depends on the user′s employed scheme for data confidentiality. Moreover, the probable amount of loss in case of data tempering as a result of intrusion or access by other VMs cannot be decreased. Our proposed strategy does not depend on the traditional cryptographic techniques for data security. Moreover, the DROPS methodology does not store the whole file on a single node to avoid compromise of all of the data in case of successful attack on the node.

The authors in [11] approached the virtualized and multi-tenancy related issues in the cloud storage by utilizing the consolidated storage and native access control. The Dike authorization architecture is proposed that combines the native access control and the tenant name space isolation. The proposed system is designed and works for object based file systems. However, the leakage of critical information in case of improper sanitization and malicious VM is not handled. The DROPS methodology handles the leakage of critical information by fragmenting data file and using multiple nodes to store a single file.

The use of a trusted third party for providing security services in the cloud is advocated in [22]. The authors used the public key infrastructure (PKI) to enhance the level of trust in the authentication, integrity, and confidentiality of data and the communication between the involved parties. The keys are generated and managed by the certification authorities. At the user level, the use of temper proof devices, such as smart cards was proposed for the storage of the keys.

Similarly, Tang *et. al.* have utilized the public key cryptography and trusted third party for providing data security in cloud environments [20]. However, the authors in [20] have not used the PKI infrastructure to reduce the overheads. The trusted third party is responsible for the generation and management of public/private keys.

The trusted third party may be a single server or multiple servers. The symmetric keys are protected by combining the public key cryptography and the *(k, n)* threshold secret sharing schemes. Nevertheless, such schemes do not protect the data files against tempering and loss due to issues arising from virtualization and multi-tenancy.

Pros and Cons
Scale and Cost
Security
Lock-in
Choice and Agility
Cloud Computing
Encapsulated Change Management
Lack of Control
Next-Generation Architectures
Reliability

Fig. 3. Pros and Cons.

The *n* shares is carried out through the *(k, n)* threshold secret sharing scheme. The network is divided into clusters. The number of replicas and their placement is determined through heuristics. A primary site is selected in each of the clusters that allocates the replicas within the cluster. The scheme presented in [21] combines the replication problem with security and access time improvement. Nevertheless, the scheme focuses only on the security of the encryption key. The data files are not fragmented and are handled as a single file. The DROPS methodology, on the other hand, fragments the file and store the fragments on multiple nodes.

Moreover, the DROPS methodology focuses on the security of the data within the cloud computing domain that is not considered in [21]. Before we go into the details of the DROPS methodology, we introduce the related concepts in the following for the ease of the readers.

III. PRELIMINARIES

Data Fragmentation

A secure and optimal placement of data objects in a distributed system is presented in [21]. Will require the effort to penetrate only a single node. The amount of compromised data can be reduced by making fragments of a data file and storing them on separate nodes [17, 21]. A successful intrusion on a single or few nodes will only provide access to a portion of data that might not be of any significance. Moreover, if an attacker is uncertain about the locations of the fragments, the probability of finding fragments on all of the nodes is very low. 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 $s>z$.

Homogenous systems, the same flaw can be utilized to target other nodes within the system. The success of an attack on the subsequent nodes will require less effort as compared to the effort on the first node. Comparatively, more effort is required for heterogeneous systems. However, compromising a single file

a. Betweenness Centrality

The betweenness centrality of a node *n* is the number of the shortest paths, between other nodes, passing through *n* [24]. Formally, the betweenness centrality of any node *v* in a network.

b. Eccentricity

The eccentricity of a node *n* is the maximum distance to any node from a node *n* [24]. A node is more central in the network, if it is less eccentric. Formally, the eccentricity can be given as:

The security of a large-scale system, such as cloud depends on the security of the system as a whole and the security of individual nodes. A successful intrusion into a single node may have severe consequences, not only for data and applications on

Centrality

The centrality of a node in a graph provides the measure of the relative importance of a node in the network. The objective of improved retrieval time in replication makes the centrality measures more important. There are various centrality measures; for instance, closeness centrality, degree centrality, be- tweenness centrality, eccentricity centrality, and eigenvector centrality. We only elaborate on the closeness, betweenness, and eccentricity centralities because we are using the aforesaid three centralities in this work. For the remainder of the centralities, we encourage the readers to review [24].

Whole file [17]. A successful intrusion may be a result of some software or administrative vulnerability [17]. In case of the victim node, but also for the other nodes.

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

where $d(v_a, v_b)$ represents the distance between node. v_a and node v_b. 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 metric P_k denote the primary node that stores the primary copy of O_k.

The replication scheme for O_k denoted by R_k is also stored at P_k. Moreover, every S^i contains replication strategies. However, replication increases the number of file copies within the cloud. Thereby, increasing the probability of the node holding the file represents the nearest node storing O_k. Whenever there is an update in O_k, the updated version is sent to P_k that broadcasts the updated version to all of the nodes in R_k. Let *b(i,j)* and *t(i,j)* be the total bandwidth of the link and traffic between sites S^i and S^j , respectively. The centrality measure for S^i that to be a victim of attack as discussed in Section 1.

Let col_{S^i} store the value of assigned color to S^i. The col_{S^i} can have one out of two values, namely: open color and close color. The value open color represents that the node is available for storing the file fragment. The value close color shows that the node cannot store the file fragment. Let *T* be a set of integers starting from zero and ending on a prespecified number

A node is said to be closer$_k$ with respect to all of the other nodes within a network, if the sum of the distances from all of all of *N* is total number of nodes in a network and $d(v, a)$ represents the distance between node *v* and node *a*. other nodes, the more *.ZZT* containing non-negative integers including 0. Consider a cloud that consists of *M* nodes, each with its own storage capacity. the other nodes [24]. The lower the sum of distances from them i.The i-th node and s_i denotes total storage capacity of S^i.

where $(x, y) \in E$. The mapping function *f* assigns a color to a vertex. In simple words, the distance between the colors of the adjacent vertices must not belong to *T*. Formulated by Hale [6], the T-coloring problem for channel assignment assigns channels to the nodes, such that the channels are separated by distance to avoid interference.

In addition to providing primitives for low-level operations, the Spec node also includes primitives to support complex operations such as encryption. Choose the nodes that are most central to the cloud network to provide better access time. For the aforesaid purpose, the DROPS methodology uses the concept of centrality to reduce access the OPEN list. The list is ordered in the ascending order so that the solution with the minimum cost is expanded first. The heuristic used by the DRPA- star is given as $h(n) = max(0, (mmk(n)g(n)))$, where color.

In the aforesaid process, we lose some of the central nodes that may increase the retrieval time but we achieve a higher security level. The neighborhood at a distance belonging to *T* are assigned close color. Once a fragment is placed on the node, all of the nodes within *mmk(n)* is the least cost replica allocation or the max-
min RC

Readers are encouraged to see the details about DRPA-star in [13]. The WA-Star is a refinement of the DRPA-star that implements a weighted func- tion to evaluate the cost. The function is given as: $f(n) = f(n) + h(n) + s(1-(d(n)/D)h(n)$. The variable*d(n)* represents the depth of the node *n* and *D* denotes access layer switches are connected using aggregate layer switches.

The Spec node includes an encryption accelerator that can be used to automatically encrypt and decrypt messages for transmission. LFSRs that are xor-ed together generate a single random sequence. This parallels the encryption methods used in the Bluetooth wireless standard. This collection of 4 LFSR offloads a majority of the overhead

The communication time between S^i and S^j is the total time of all of the links within a selected pathfrom S^i to S^j represented by *c(i, j)*. We consider *N* number of file fragments such that O_k denotes *k*-th fragment of a file while o_k represents *the* size of *k-th* fragment. Let the total read and write requests from *S* then *T* = {0, 1, 2, 3}. The set *T* is used to restrict the node selection to those nodes that are at hop-distances not belonging to *T*. For the ease of reading, the most commonly used notations are listed in fig3 .

Once the file is split into fragments, the DROPS methodology selects the cloud nodes for fragment placement associated with encryption. Unlike in Bluetooth, the processing for seeding the LFSRs is done in software

Algorithm 1 Algorithm for fragment 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
$cen \leftarrow cen_i \forall$ i

Compute:
for each $O_k \in O$ do
    select $S^i \mid S^i \leftarrow$ indexof(max($cen_i$))
    if $col_{S^i} = open\_color$ and $s_i >= o_k$ then
        $S^i \leftarrow O_k$
        $s_i \leftarrow s_i - o_k$
        $col_{S^i} \leftarrow close\_color$
        $S^{i'} \leftarrow distance(S^i, T)$    s at
        distance $T$ from $S^i$
        $col_{S^{i'}} \leftarrow close\_color$

  end if
```

IV. DROPS

However, as stated previously in Section 3.1, the probability of a successful coordinated attack is extremely minute. The process is repeated until all of the fragments are placed at the nodes. Algorithm 1 represents the fragment placement methodology.

In addition to placing the fragments on the central nodes, we also perform a controlled replication to increase the data availability, reliability, and improve data retrieval time. We place the fragment on the node that provides the decreased access cost with an objective to improve retrieval time for accessing the fragments for reconstruction of original file. While replicating the fragment, the separation of fragments as explained in the placement technique through T- coloring, is also taken care off.

In case of a large number of fragments or small number of nodes, it is also possible that some of the fragments are left without being replicated because of the T-coloring. As discussed previously, T-coloring prohibits storing the fragment in neighborhood of a node storing a fragment, resulting in the elimination of a number of nodes to be used for storage.

In such a case, only for the remaining fragments, the nodes that are not hold ing any fragment are selected for storage randomly. The replication strategy is presented in Algorithm 2. To handle the download request from user, the cloud manager collects all the fragments from the nodes and re-assemble them into a single.

We implement DROPS with three centrality measures, namely: **(a)** betweenness, **(b)** closeness, and **(c)** eccentricity centrality. However, if all of the fragments are placed on the nodes based on the descending order of centrality, then there is a possibility that adjacent nodes are selected for fragment placement. Such a placement can provide clues to an attacker as to where other fragments might be present, reducing the security level of the data. To deal with the security aspects of placing fragments, we use the concept of T-coloring that was originally used for the channel assignment problem [6]. We generate a non-negative random number and build the set T starting from zero to the generated random number. The set T is used to restrict the node selection to those nodes that are at hop-distances not belonging to T. For the said purpose, we assign colors

V. Experimental Setup and Results

The communicational backbone of cloud computing is the Data Center Network (DCN) [2]. In this paper, we use three DCN architectures namely: **(a)** Three tier, **(b)** Fat tree, and **(c)** DCell [1]. The Three tier is the legacy DCN architecture. However, to meet the growing de- mands of the cloud computing, the Fat tree and Dcell architectures were proposed [2]. Therefore, we use the aforementioned three architectures to evaluate the performance of our scheme on legacy as well as state of the art architectures. The Fat tree and Three tier architectures are switch-centric networks. The nodes are connected with the access layer switches fully connected. For details about the aforesaid architectures and their build the higher level dcells

Comparative Techniques:

We compared the results of the DROPS methodology with fine-grained replication strategies, namely:
(a) DRPA-star, **(b)** WA-star, **(c)** A*s*-star, **(d)** SA1, **(e)** SA2, **(f)** SA3, **(g)** Local Min-Min, **(h)** Global Min- Min, **(i)** Greedy algorithm, and **(j)** Genetic Replication Algorithm (GRA). The DRPA-star is a data replication algorithm based on the A-star best-first search algorithm. The DRPA-star starts from the null solution that is called a root node. The communication cost at each node n is computed as: $cost(n) = g(n) + h(n)$, where $g(n)$ is the path cost for reaching n and $h(n)$ is called the heuristic cost and is the estimate of cost from n to the goal node. The DRPA-star searches all of the solutions of allocating a fragment to a node. The solution that minimizes the cost within the constraints is explored while others are discarded. The selected solution is inserted into a list called the expected depth of the goal node [13]. The A*s*-star is also a variation of the DRPA-star that uses two lists, OPEN and FOCAL.

The FOCAL list contains only those nodes from the OPEN list that have f greater than or equal to the lowest f by a factor of $1 + s$. The node expansion is performed from the FOCAL list instead of the OPEN list. Further details about WA- Star and A*s*-star can be found in [13]. The SA1 (sub-optimal assignments), SA2, and SA3 are DRPA-star based heuristics.

In SA1, at level R or below, only the best successors of node n having the least expansion cost are selected. The SA2 selects the best successors of node n only for the first time when it reaches the depth level R. All other successors are discarded. The SA3 works similar to the SA2, except that the nodes are removed from OPEN list except the one with the lowest cost. Readers are encouraged to read [13] for further details about SA1, SA2, and SA3. The LMM can be considered as a special case of the bin packing algorithm.

The LMM sorts the file fragments based on the RC of the fragments to be stored at a node. The LMM then assigns the fragments in the ascending order. In case of a tie, the file fragment with minimum size is selected for assignment (name local Min-Min is derived from such a policy). The GMM selects the file fragment with global minimum of all the RC associated with a file fragment. In case of a tie, the file fragment is selected at random. The Greedy algorithm first iterates through all of the M cloud nodes to find the best node for allocating a file fragment. The node with the lowest replication cost is selected. The second node for the fragment is selected in the second iteration.

However, in the second iteration that node is selected that produces the lowest RC in combination with node already selected. The process is repeated for all of the file fragments. Details of the greedy algorithm can be found in [18]. The GRA consists of chromosomes rep- resenting various schemes for storing file fragments over cloud nodes. Every chromosome consists of M genes, each representing a node. Every gene is a N bit string. If the k-th file fragment is to be assigned to S^i, then the k-th bit of i-*th* gene holds the value of one. Genetic algorithms perform the operations of selection, crossover, and mutation. The value for the crossover rate (μ_C) was selected as 0.9, while for the mutation rate (μ_m) the value was 0.01. The use of the values for μ_C and μ_m is advocated in [16]. The best chromosome represents the solution. GRA utilizes mix and match strategy to reach the solution. More details about GRA can be obtained from [16].

Workload

The size of files were generated using a uniform distribution between 10Kb and 60 Kb. The primary nodes were randomly selected for replication algorithms. For the DROPS methodology, the $S^{i'}$s selected during the first cycle of the nodes selection by Algorithm 1 were considered as the primary nodes.

The capacity of a node was generated using a uniform distribution between $(\frac{1}{2}CS)C$ and $(\frac{3}{2}CS)C$, where $0 \leq C \geq 1$. For instance, for $CS = 150$ and $C = 0.6$ the capacities of the nodes were uniformly distributed between 45 and 135. The mean value of *g* in the OPEN and FOCAL lists was selected as the value of *s*, for WA-star and A*s*-star, respectively. The value for level *R* was set to [$\underline{d}$], where d is the depth of the search tree(number of fragments).

The read/write (R/W) ratio for the simulations that used fixed value was selected to be 0.25 (The R/W ratio reflecting 25% reads and 75% writes within the cloud). The reason for choosing a high workload (lower percentage of reads and higher percentage of writes) was to evaluate the performance of the techniques under extreme cases.

The simulations that studied the impact of change in the R/W ratio used various workloads in terms of R/W ratios. The R/W ratios selected were in the range of 0.10 to 0.90. The selected range covered the effect of high, medium, and low workloads with respect to the R/W ratio.

VI. RESULTS AND DISCUSSION

We compared the performance of the DROPS methodology with the algorithms discussed in Section 5.1. The behavior of the algorithms was studied by: **(a)** increasing the number of nodes in the system, **(b)** increasing the number of objects keeping number of nodes constant, **(c)** changing the nodes storage capacity, and **(d)** varying the read/write ratio. The aforesaid parameters are significant as they affect the problem size and the performance of algorithms [13].

Impact of increase in number of file fragments:

The increase in number of file fragments can strain the storage capacity of the cloud that, in turn may affect the selection of the nodes. To study the impact on performance due to increase in number of file fragments, we set the number of nodes to 30,000.

The numbers of file fragments selected were 50, 100, 200,300, 400, and 500. The workload was generated with C = 45% to observe the effect of increase number of file fragments with fairly reasonable amount of memory and to discern the performance of all the algorithms. The results are shown in Fig. 5 (a), Fig. 5 (b), and Fig. 6 (a) for the Three tier, Fat tree, and Dcell architectures, respectively. It can be observed from the plots that the increase in the number of file fragments reduced the performance of the algorithms, in general. However, the greedy algorithm showed the most improved performance. The LMM showed the highest loss in performance that is little above 16%. The loss in performance can be attributed to the storage capacity constraints that prohibited the placements of some fragments at nodes with optimal retrieval time.

As discussed earlier, the DROPS methodology produced similar results in three tier and fat tree architectures. However, from the Dcell architecture, it is clear that the DROPS methodology with eccentricity centrality maintains the supremacy on the other two centralities.

The fragments are dispersed over multiple nodes. The nodes were separated by means of T-coloring. The fragmentation and dispersal ensured that no significant because of violation of storage capacity constraints.

We proposed the DROPS methodology, a cloud storage security scheme that collectively deals with the security and performance in terms of retrieval time. The data file was some optimal nodes to be selected for replication node storage capacity may result in the elimination of capacity constraints. Intuitively, a lower

VII. CONCLUSIONS

Work will save the time and resources utilized in downloading, up- dating, and uploading the file again. Moreover, the implications of TCP incast over the DROPS methodology need to be studied that is relevant to distributed data storage and access. information was obtainable by an adversary in case of a successful attack. No node in the cloud, stored more than a single fragment of the same file. The performance of the DROPS methodology was compared with full-scale replication techniques. The results of the simulations revealed that the simultaneous focus on the security and performance, resulted in increased security level of data accompanied by a slight performance drop.

Currently with the DROPS methodology, a user has to download the file, update the contents, and upload it again. It is strategic to develop an automatic update mechanism that can identify and update the required fragments only. The aforesaid future

REFERENCES

[1] K. Bilal, S. U. Khan, L. Zhang, H. Li, K. Hayat, S. A. Madani, N. Min-Allah, L. Wang, D. Chen, M. Iqbal, C. Z. Xu, and A. Y. Zomaya, “Quantitative comparisons of the state of the art data center architectures,” *Concurrency and Computation: Practice and Experience*, vol. 25, no. 12, pp. 1771-1783, 2013.

[2] K. Bilal, M. Manzano, S. U. Khan, E. Calle, K. Li, and A. Zomaya, “On the characterization of the structural robustness of data center networks,” *IEEE Transactions on Cloud Computing*, vol. 1, no. 1, pp. 64-77, 2013.

[3] 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.

[4] Y. Deswarte, L. Blain, and J-C. Fabre, “Intrusion tolerance in distributed computing systems,” *In Proceedings of IEEE Computer Society Symposium on Research in Security and Privacy, Oakland CA*, pp. 110-121, 1991.

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

[6] W. K. Hale, “Frequency assignment: Theory and applications,” *Proceedings of the IEEE*, vol. 68, no. 12, pp. 1497-1514, 1980.

[7] K. Hashizume, D. G. Rosado, E. Fernndez-Medina, and E. B. Fernandez, “An analysis of security issues for cloud computing,” *Journal of Internet Services and Applications*, vol. 4, no. 1, pp. 1-13, 2013.
[8] M. Hogan, F. Liu, A. Sokol, and J. Tong, *NIST Cloud Computing Standards Roadmap*, NIST Special Publication, 2011.
[9] W. A. Jansen, “Cloud hooks: Security and privacy issues in cloud computing,” *In 44th Hawaii IEEE International Conference on System Sciences (HICSS)*, pp. 1-10, 2011.
[10] A. Juels and A. Opera, “New approaches to security and availability for cloud data,” *Communications of the ACM*, vol. 56, no. 2, pp. 64-73, 2013.
[11] G. Kappes, A. Hatzieleftheriou, and S. V. Anastasiadis, “Dike: Virtualization-aware Access Control for Multitenant File systems,” University of Ioannina, Greece, Technical Report No. DCS2013-1, 2013.
[12] L. M. Kaufman, “Data security in the world of cloud computing,” *IEEE Security and Privacy*, vol. 7, no. 4, pp. 61-64, 2009.
[13] S. U. Khan, and I. Ahmad, “Comparison and analysis of ten static heuristics-based internet data replication techniques,” *Journal of Parallel and Distributed Computing*, vol. 68, no. 2, pp. 113-136, 2008.
[14] 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, pp. 1278, 2013
[15] A. N. Khan, M. L. M. Kiah, S. A. Madani, and M. Ali, “Enhanced dynamic credential generation scheme for protection of user identity in mobile-cloud computing,” *The Journal of Supercomputing*, vol. 66, no. 3, pp. 1687-1706, 2013.
[16] T. Loukopoulos and I. Ahmad, “Static and adaptive distributed data replication using genetic algorithms,” *Journal of Parallel and Distributed Computing*, vol. 64, no. 11, pp. 1270-1285, 2004.
[17] 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, pp. 885-896, 2003.
[18] L. Qiu, V.N. Padmanabhan, and G. M. Voelker, “On the placement of web server replicas,” *In Proceedings of INFOCOM 2001, Twentieth Annual Joint Conference of the IEEE Computer and Communications Societies*, vol. 3, pp. 1587-1596, 2001.
[19] 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, pp. 2852-2856, 2011.