PROVIDING BACKUP STORAGE FOR MIGRATING DATACENTER SERVICES

Neela.K.L[1], AyshaNazrin.A.J[2], Brishiba Reshma.L[3], Medlin Mejo.J[4]
[1] Assistant Professor, [2,3,4] Student
[1,2,3,4] Department of Computer Science
University College of Engineering, Nagercoil, India
Email:[1]klneela@yahoo.com, [2]ayshanazrin96@gmail.com, [3]brishibareshma@gmail.com, [4]mejojm5@gmail.com

Abstract: *Cloud computing is a method for delivering Information technology (IT) services in which resources are retrieved from the internet through web-based tools and applications, as opposed to a direct connection to a server. Rather than keeping files on a proprietary hard drive or local storage device, cloud- based storage makes it possible to save them to a remote database. Service migration between datacenters can reduce the network overhead within a cloud infrastructure, thereby improving the quality of services for the client. The System employs the Network Overhead Minimization (NOM) algorithm to solve the time complexity in deciding at what point in a time a service needs to be migrated to reduce the Network Overhead. The Network Overhead Minimization algorithm determines whether (or not) to migrate a given service from its current host to another datacenter to minimize the total network overhead. Current System does not provide any backup plan for storing user data's while migrating between datacenters, so if there is a failure in migration the user's data will be lost and it can't be retrieved. Therefore, in order to protect the user's data, this project proposes the concept of having a Remote cloud (i.e., a cloud far away from the main cloud) for storing the data's of main cloud by employing Seed Block Algorithm (SBA), so that even if the central repository (main cloud) lost its data under any circumstances either a failure during migration between datacenters or by human attack or deletion that has been done mistakenly the user's data can be retrieved easily from the remote cloud.*

Keywords: *Network Overhead Minimization algorithm, Seed Block algorithm, Cloud Computing, Datacenter.*

1. INTRODUCTION:

In the early 1940s Datacenters have their roots in the huge computer rooms typified by ENIAC, one of the earliest examples of a datacenter. Early Data centers are complex to operate and maintain and required a special environment to operate and they are lot expensive too. It required a great deal of power so that it needs to be cooled in order to avoid overheating. The boom of data centers came during the dot-com bubble of 1997-2000. Companies needed fast Internet connectivity and non-stop operation to deploy systems and to establish a presence on the Internet. Installing such equipment was not viable for many smaller companies. Many companies started building very large facilities, called Internet data centers (IDCs), which provide commercial clients with a range of solutions for systems deployment and operation. A datacenter is a facility composed of networked computers and storage that businesses or other organization use to organize, process, store and disseminate large amounts of data.

The Requirement for modern Datacenters is only due to maintain business continuity because each companies rely on their information systems to run their operations. If a system becomes unavailable, company operations may be impaired or stopped completely. It is necessary to provide a reliable infrastructure for IT operations, in order to minimize any chance of disruption. Information security is also a concern, and for this reason
a datacenter has to offer a secure environment which minimizes the chances of a security breach. A datacenter must therefore keep high standards for assuring the integrity and functionality of its hosted computer environment. This is accomplished through redundancy of mechanical cooling and power systems (including emergency backup power generators) serving the datacenter along with fiber optic cables. Service migration is a concept used in cloud computing implementation models that ensures that an individual or organization can easily shift between different cloud vendors without encountering implementation, integration, compatibility and interoperability issues.

The need for service migration between datacenters is due to reduce the total network overhead thereby improving the quality of services for the client. Overhead is any combination of excess or indirect computation time, memory, bandwidth, or other resources that are required to perform a specific task. In order to reduce the time complexity in deciding at what point in a time a service needs to be migrated NOM algorithm is used. This algorithm determines whether to migrate a given service from its current host to another datacenter or not. While migrating services between datacenters if there is a failure in migration then the user's data will be lost. So in order to protect user's data a remote cloud is set up by employing Seed Block Algorithm (SBA) so even if the main cloud lost its data under any circumstances the lost data can be retrieved easily thereby achieving Fault tolerance. Security is provided for the backup files by simply doing EX-OR operation, so even if the remote cloud is hacked the data can't be viewed thus providing enhanced security.

2. RELATED WORK:

A lot of work have been studied the Service migration between datacenter under various objectives. Specially for taking a replica of the complete state of the main cloud various techniques/Algorithms are used.

Bienkowskiet al. [3] first proposes the Ability to migrate services closer to the client location and allowing the virtual network to adapt to the needs of the services for finding the best migration path. However, it is difficult to balance the number of replicas of each virtual server that exist at any given time.

Meng et al. [7] proposes that network overhead is minimized by co-locating VM which heavily communicate with each other and VM's with large usage are assigned to host machines in close proximity. However, this type is not ideal for large datacenters.

Yoichiro Ueno et al. [12] proposes an HS-DRT algorithm for securing the data, it does not only depend on the cryptographic encryption technique but also the combined method such as spatial scrambling, fragmentation/duplication and shuffling algorithm. However, it is not a suitable solution for backup and recovery because of lack of consistency.

Beloglazovet al. [2] proposes Energy aware resource allocation heuristics for efficient management of data centers for cloud computing, which reduces the energy consumption by switching idle servers to power saving modes. However, this type of optimization is not suitable for large scale data center environment due to complex computations.

Vijaykumar et al. [11] here a simple backup software and hardware is attached to the Linux box, so that one can opt for routine backup's easily Low cost for implementation data transmission is Secure because of Encryption. However, it is suitable only for small and medium business not for large datacenters.

Chi-won Song et al. [4] proposes a virtual disk in user system for private data backup. Each user is enough to back up their files to their own virtual disk for future data recovery and it is highly reliable privacy based recovery services. However, implementation complexity is too high.

Sheheryar Malik et al. [9] here specifically targets the home users and small enterprises, Cost depends on the infrastructure utilization. Since, it is based on rented concept resources must be kept under special attention and hence no reliability.

Tziritas et al. [8] helps in Deciding at what point in time an agent should be migrated on which node in order to reduce the network cost and providing infinite storage to keep information about the message exchange history of agents. However, it does not support code migration in an online fashion and data will be lost if there is a failure in migration between datacenters.

3. SYSTEM MODEL AND PROBLEM DEFINITION

Let a cloud infrastructure be captured as a graph $G = (V, E)$, where each vertex $u \in V$ represents a datacenters n_u and each edge $e = (u, v) \in E$ represents a communication link between n_u and n_v in the intra-cloud network. From now on we will use interchangeably the intra-cloud and inter-datacenters network. Each communication link e is characterized by a possibly different data transfer overhead weight w_e. The data transfer overhead is computed by the average delay experienced when transferring data over that link. The aggregated network overhead for a path p between two datacenters n_u and n_v is denoted by $w_{uv}^{p} = \sum_{\forall e \in p} w_e$, and the minimum network overhead over all such paths is denoted by $W_{uv} = \min_{\forall p} w_{uv}^{p}$. Let us assume that the network is direction-neutral ($W_{uv}= W_{vu}$) and that the network overhead of local data transfers is zero ($W_{uu}= 0$).

A datacenter can host several services; however, each service is hosted on a single datacenter. Each client connects to the cloud through an entry-point datacenter, e.g., the one closest to its physical location. Client requests are forwarded over the inter-datacenter network to the datacenter hosting the respective services, and responses are sent back to the clients in the same way. Without the loss of generality, we assume that a reply follows the same path that was used for the requests (in reverse direction). Client mobility is captured implicitly, through the dynamically changing client access pattern.

Let s_j be the j^{th} service of the system, hosted on datacenter n_v. Moreover, let q_u be the set of clients that connect to the cloud through n_u, and let $\sigma_{uj}{}^{t}$ denote the data volume of the requests/replies sent/received by clients in q_u through n_u for service s_j at time t. Such traffic must be forwarded between n_u and n_v (the host of s_j). It is noteworthy to mention that no such forwarding is required if s_j is hosted on n_u. Fig. 1[8] depicts an example of the three different sets of clientsq_u, q_m, and q_v that connect to the cloud via datacenters n_u, n_m, and n_v, respectively. The clients interact with the service s_j hosted on n_u with the respective client volume being $\sigma_{uj}{}^{t}$, ${}_{j}{}^{t}$and $\sigma_{vj}{}^{t}$.

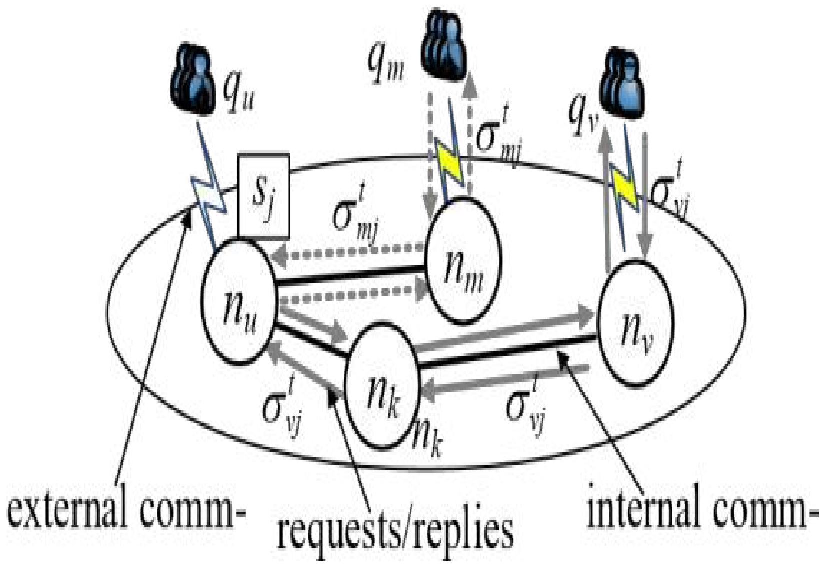


Fig. 1: An example with datacenters and clients

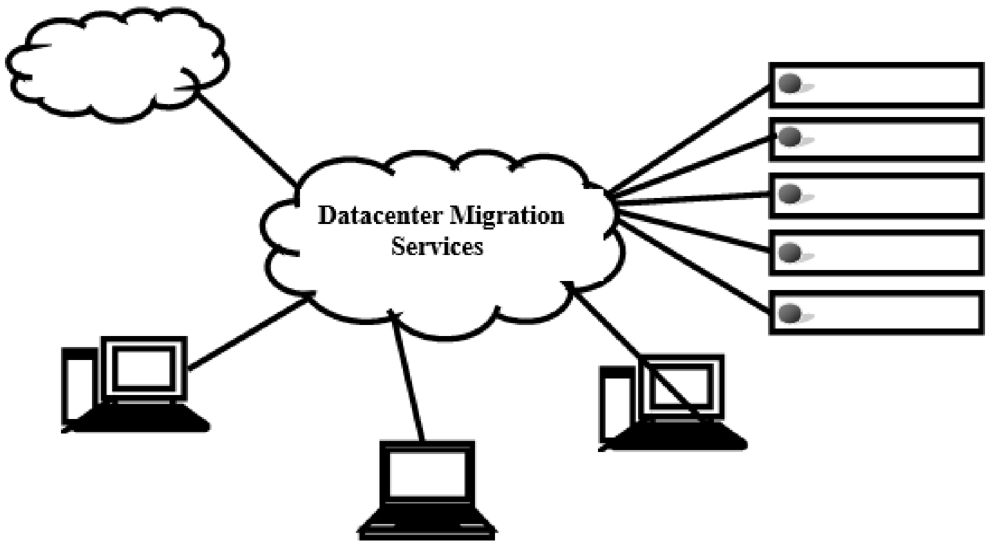


Fig. 2: Online Inter data center Service migration

To deal with a dynamically changing client request/reply volume, a service s_j can be migrated from its current host n_u to another datacenter n_v. Let $M_{uvj}(t)$ represent such a migration being performed at time t. Moreover, let MC_j be the data transferred associated with s_j. Finally, let h be a service hosting scheme/function, where $h(j, t) = u$ means that $s_j$is hosted on n_u at time t. Due to service migrations, it could be that $(j, t) \neq (j, t')$. That is to say that a service may be hosted on various datacenters at different points in time. Let also $\xi_j(t, t')$ reflect the time it takes for sj to migrate from (t, j) towards (t', j), while φ denote the maximum time it takes to migrate any service considering any pair of source/destination.The most commonly used metric to determine the energy efficiency of a data center is *power usage effectiveness,* or PUE. This simple ratio is the total power entering the data center divided by the power used by the IT equipment.

$$\text{PUE} = \frac{\text{Total Facility Power}}{\text{IT Equipment Power}}$$

4. SYSTEM DESIGN:

This section describes the system design in detail. The overall functionality of the system can be divided into four steps.Fig.2 provides an overview of the work-flow of our approach. Algorithm (NOM & SBA) explains how the system works and we will describe each step in detail. The system design clearly depicts how and when the migration between data centers take place and it also clearly explains about the Remote Cloud which contains the replica of the complete state of the Main Cloud.

4.1. NETWORK OVERHEAD MINIMIZATION (NOM) ALGORITHM

In this section, NOM algorithm decides that, in an online fashion, whether (or not) to migrate a given service from its current host to another datacenter to minimize the total network overhead. It is noteworthy to mention that the Network Overhead Minimization is designed for tree-based networks. Because each datacenter is responsible for the service migration it hosts, there is no way of a conflict between service migrations. Overhead is any combination of excess or indirect computation time, memory, bandwidth, or other resources that are required to perform a specific task. A key parameter of NOM is the extent to which a datacenter knows the topology of the neighborhood, referred to as network awareness radius R.

To reduce the time-complexity of the algorithm, the following technique is employed. Initially, the algorithm calculates the benefit of migrating s_j only to 1-hop neighbours of n_u, as if R were equal to 1, at most one 1-hop neighbour, say n_m, can have a positive benefit B_{umj}. If B_{umj} is greater than the migration threshold (see next), then the calculation is repeated by considering only the datacenters that are one hop further away from n_m. The iteration stops when: (a) no beneficial migration can be found for s_j; or (b) n_m is R hops away from n_u. The algorithm decides to migrate s_j to the last datacenter (if any) for which the benefit was greater than the threshold. The pseudocode of the Network Overhead Minimization Algorithm is given in Fig. 3[8]. Note that the Network Overhead Minimization is distributed, and runs periodically at every datacenter.

Algorithm 1 NOM

```
1:  for each s_j hosted by n_u
2:  newHost = n_u; //the current host of s_j
3:  do
4:  b = 0;  //the current benefit of migrating s_j
5:  candHost = newHost;// init best candidate for   migrating s_j
6:  for each (n_m€ R(n_u): hops(newHost, n_m) == 1) {
7:  if ( B_umj> 0 ) //pick this host
8:  b = B_umj;
9:  candHost = n_m;
10: break;
11: end if
12:  end for
13: if ((candHost == newHost) ‖ (Eq. (6) is false)) break;
14: else newHost = candHost;
15: while (hops(newHost, n_u)<R);
16: if (b!= 0) migrate s_j on newHost; // the migration decision
17:  else do nothing;
18:  if (Eq. (7) is true) then reset the load variables s_j
19: End for                          ▷NOM Algorithm ends
```

Fig. 3: Pseudocode of NOM for datacenter n_u

In addition to the aforementioned steps, NOM makes two important checks or actions. The first check (line 13 of the pseudocode) is to decide for a service migration only if Eq. (6)[8] holds (if the benefit is at least twice the network overhead for actually performing the migration). The second check (line 18 of the pseudocode) is to reset the load variables when Eq. (7)[8] holds (when the aggregate reaches the so-called reset threshold RT_j).

$$B_{umj} \geq W_{um} \times 2MC_j \qquad (6)$$

$$\sum_{n_m \in R(n_u)} rmj \geq RTj \qquad (7)$$

4.2. SEED BLOCK ALGORITHM (SBA)

A. REMOTE DATA BACKUP SERVER

When we talk about Backup server of main cloud, we only think about the copy of main cloud. When this Backup server is at remote location (i.e. far away from the main server) and having the complete state of the main cloud, then this remote location server is termed as Remote Data Backup Server. The main cloud is termed as the central repository and remote backup cloud is termed as remote repository.

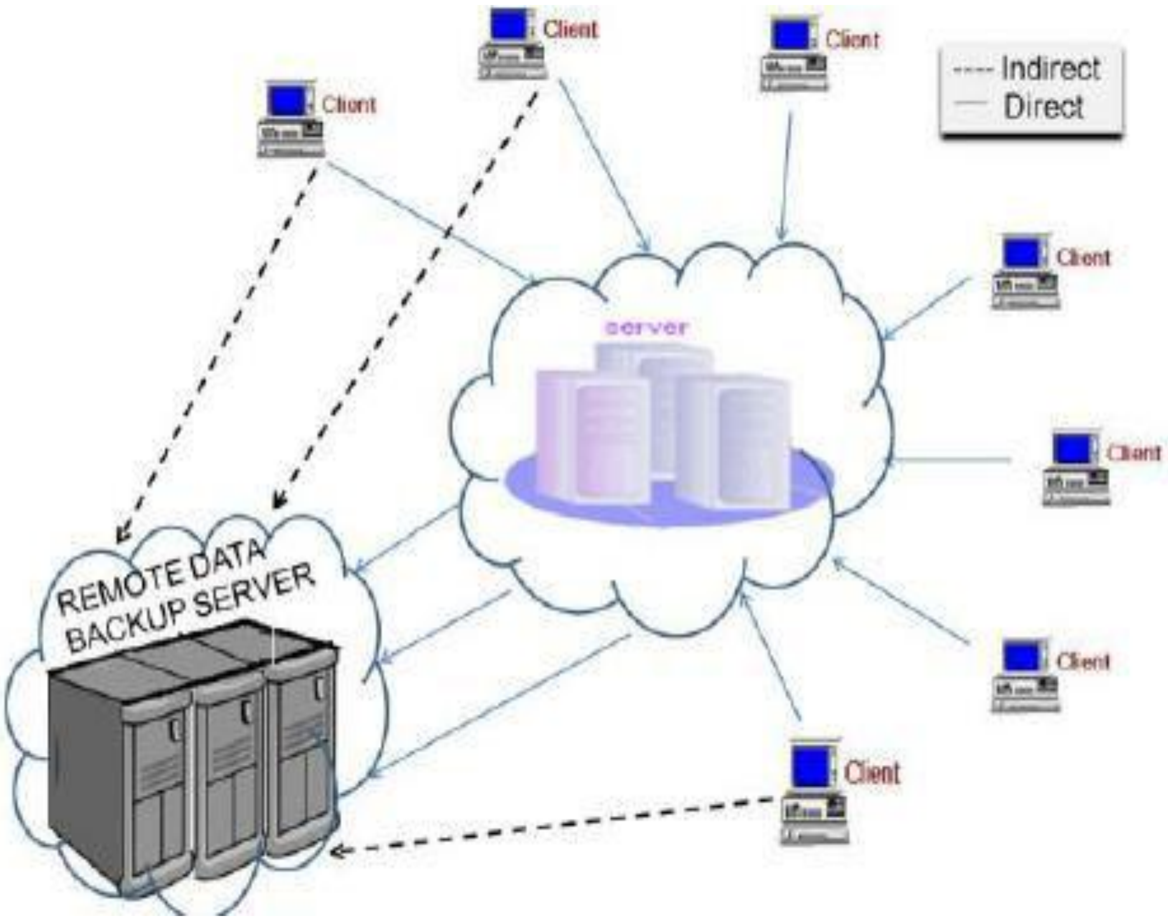


Fig. 4: Remote data backup server

And if the central repository lost its data under any circumstances either of any natural calamity (for ex - earthquake, flood, fire etc.) or by human attack or deletion that has been done mistakenly and then it uses the information from the remote repository. The main objective of the remote backup facility is to help user to collect information from any remote location even if network connectivity is not available or if data not found on main cloud. As shown in Fig. 4[6] clients are allowed to access the files from remote repository if the data is not found on central repository (i.e. indirectly).

B. DESIGN OF SEED BLOCK ALGORITHM

As discussed in literature, many techniques have been proposed for recovery and backup such as HSDRT, PCS, ERGOT, Linux box, Cold/Hot backup strategy etc. But due to high implementation complexity, lack of Reliability and Consistency, less data security and other time related issues these techniques have become a failure in the backup and recovery process and it has become a challenge in the field of cloud computing. To tackle these issues, this system proposes Seed Block Algorithm which uses the simple EXOR operation instead of the high complex encryption techniques to backup user's data thereby solving time related issues. It basically uses the concept of Exclusive– OR (XOR) operation of the computing world. For ex: - Suppose there are two data files: A and B. When we XOR A and B it produced X. If suppose A data file get destroyed and we want our A data file back, then it is very easy to get back it with the help of B and X data file. Similarly, the Seed Block Algorithm works to provide the simple Back-up and recovery process. Its architecture is shown in Fig. 5 consists of the Main Cloud and its clients and the Remote Server. Here, first we set a random number in the cloud and unique client id for every client. Second, whenever the client id is being register in the main cloud; then client id and random number is getting EXORed with each other to generate seed block for the particular client. The generated seed block corresponds to each client is stored at remote server.

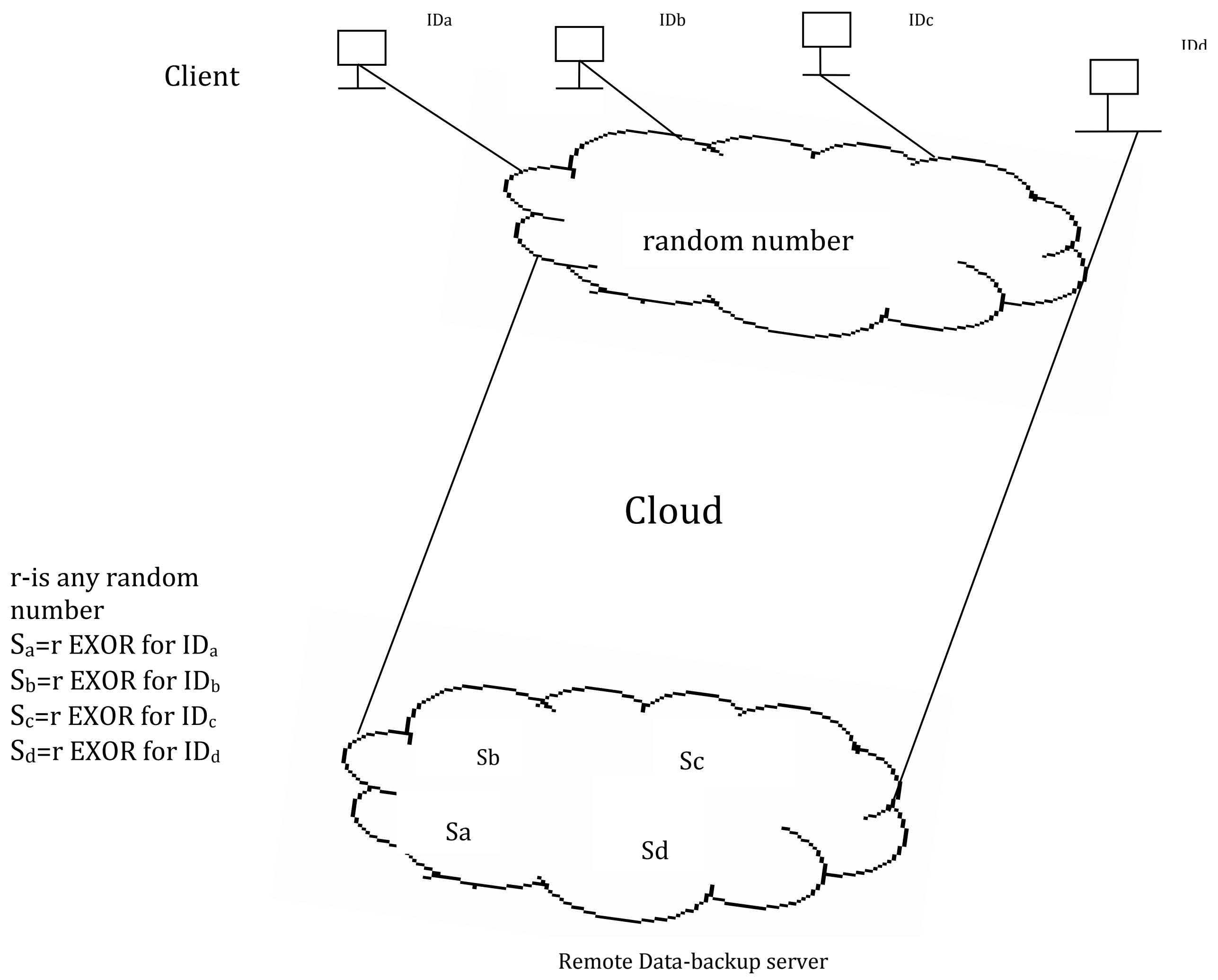


Fig. 5: SBA Architecture

This algorithm focuses on simplicity of the back-up and recovery process. The Pseudocode of Seed Block Algorithm is given in Fig .6[6] Whenever client creates the file in cloud first time, it is stored at the main cloud. When it is stored in main server, the main file of client is being EXORed with the Seed Block of the particular client. And that EXORed file is stored at the remote server in the form of file' (pronounced as File dash). If either unfortunately file in main cloud crashed / damaged or file is deleted mistakenly, then the user will get the original file by EXORing file' with the seed block of the corresponding client to produce the original file and return the resulted file i.e. original file back to the requested client. Thereby Fault tolerance can be achieved. Since the backup data file is EXORed even if the hacker hacks the he can't view the file thus providing enhanced security for the user's private data.

Algorithm 2 SBA (M_c,R_s)

```
1: Initialization: Main Cloud: $M_c$
2: Remote server: $R_s$
3: Clients of Main cloud: $C_i$
4: Files : $a_1$and $a`_1$
5: Seed block: $S_i$
6: Random number: r
7: Client's id: $Client\_id_i$
8: Generate a random number
   int R= rand();
9: Create a seed block $S_i$ for each $C_i$ and store $S_i$ at $R_s$
   $S_i = r \oplus Client\_id_i$;(Repeat step 9 for all Clients)
```

```
10: if $C_i$/Admin creates/modifies a $a_1$ and stores at $M_C$, then $a`_1$ create as
    $a`_1 = a_1 \oplus S_i$ ;
11: Store $a`_1$ at $R_s$.
12: if Server crashes $a_1$ deleted from $M_c$,
    then we do EXOR to retrieve the original $a_1$
    $a_1 = a`_1 \oplus S_i$ ;
13: Return $a_1$ to $C_i$ ;
14: End For                    ▷SBA Algorithm ends
```

Fig. 6: Pseudocode of SBA algorithm

5. MATHEMATICAL MODEL:

1. Client Module:
   Set (C) = c0, c1, c2, c3, c4
   C0-registration, C1-upload files, C2-download files,
   C3-delete files, C4-request files

2. Main Server Module
   Set (M) = c3, m0, m1, m2, m3
   M0-authentication, M1-generate random number,
   M2-create seed block, M3-send files

3. Remote Server Module
   Set (R) = m3, r0, r1, r2
   R0-get random number,
   R1-store backup, R2-recover files.

Venn Diagram:

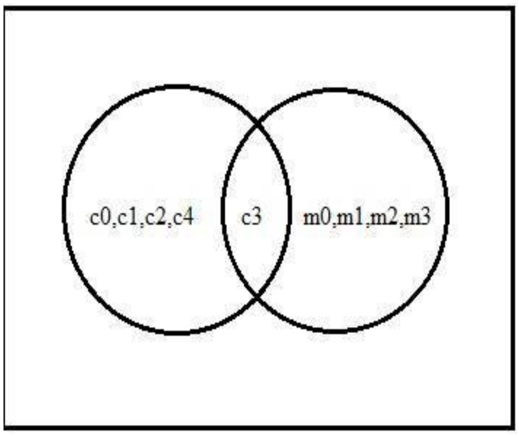


Fig. 7: *C Intersection M*

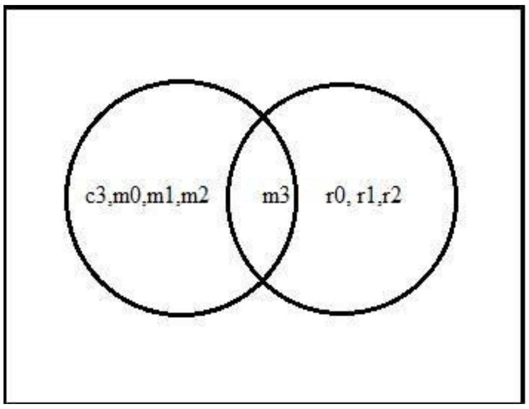


Fig. 8: *M Intersection R*

- **Success Conditions**: Our system will give the expected result
- **Failure Conditions**: NA

Here Fig. 7[3] represents the intersection between client and main server, it shows that both client and admin can delete the file. The clients also have the right to request file, upload file, and download file.

Fig. 8[3] represents the intersection between main server and remote server, it shows that the main server sends a file as a backup to the remote server by doing Ex-or operation and the remote server sends that original file back to the central repository as a part of recovery process if the data is lost in the main cloud either due to a failure during migration or human attack.

6. EXPERIMENTATION AND RESULT ANALYSIS

In this section, an experimental study of the time complexity, performance and comparison results while using SBA algorithm for providing backup storage during service migration between data centers was carried out. The evaluation results show the good performance of the proposed system in different aspects when compared to other data backup techniques.

A. Performance analysis for different types of files

During experimentation, it is found that size of original data file stored at main cloud is exactly similar to the size of Back-up file stored at Remote Server as depicted in Table 1. In order to make this fact plausible, experiment evaluation for different types of files is carried out. Results are tabulated in Table 1. for this experiment shows that proposed SBA is very much robust in maintaining the size of the recovery file same as that of the original data file. From this the system conclude that proposed Seed Block Algorithm recover the data file without any data loss.

Table 1. Performance analysis for different types of files

Type	Size of Main file in Server	Size of backup file in Remote Server	Size of Recovered file
Text (.txt/ .doc/ .docx/ .xl/ .pdf)	550 KB	550 KB	550 KB
	2.7 MB	2.7 MB	2.7 MB
Image (.jpeg/ .gif/ .png/ .bitmap)	70 KB	70 KB	70 KB
	4.2 MB	4.2 MB	4.2 MB

Here the complete state of the central repository (Main Cloud) is maintained at the remote server and from the performance analysis it is clear that the data is unaltered during the transmission and Reception, hence Data Integrity is achieved.

B. Effect of data size on processing time

Processing Time means the time taken by the process when client uploads a file at main cloud and this process includes the assembling of data such as the random number from main cloud, seed block of the corresponding client from the remote server for EXORing operation; after assembling, performing the EXORed operation of the contents of the uploaded file with the seed block and finally stored the EXORed file onto the remote server. Processing time of both the main and the remote cloud in this experiment is tabulated in Table 2[6]. We also have observed that even if the size of the data increases, the processing time is still faster in the Remote server than in main cloud.

Table 2. Effect of data size on processing time

File Size	Processing time on Main Cloud (in ms) (Approx.)	Processing time on Remote Cloud (in ms) (Approx.)
75 KB	22.8	12

15 MB	5600	1800
32 MB	8400	3000
1 GB	16200	7500
4 GB	32100	15400
6 GB	52000	26300

From the Fig. 9 it is clear that the time taken to process the data in the backup cloud is lesser when compared to the main cloud. Thus we can say that SBA algorithm takes less time to access or process the data when compared to other backup techniques.

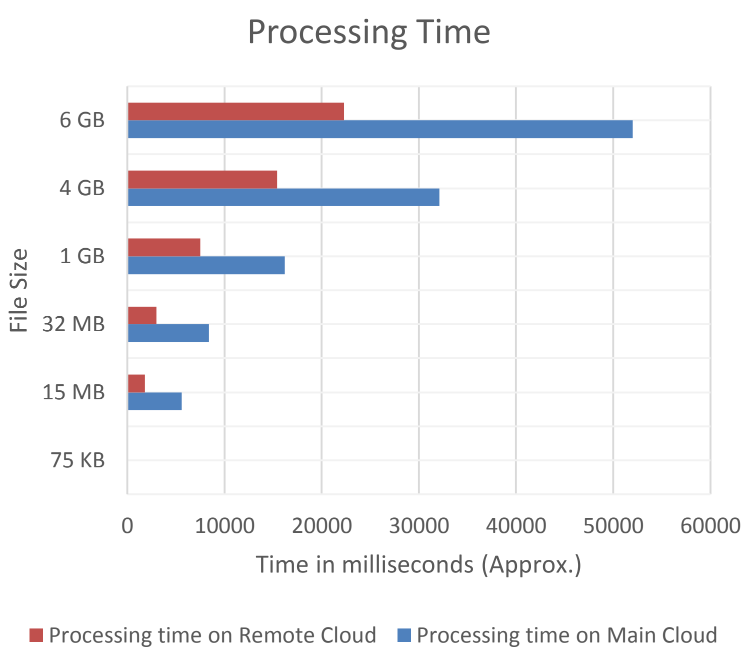


Fig. 9. Comparison between main and remote cloud

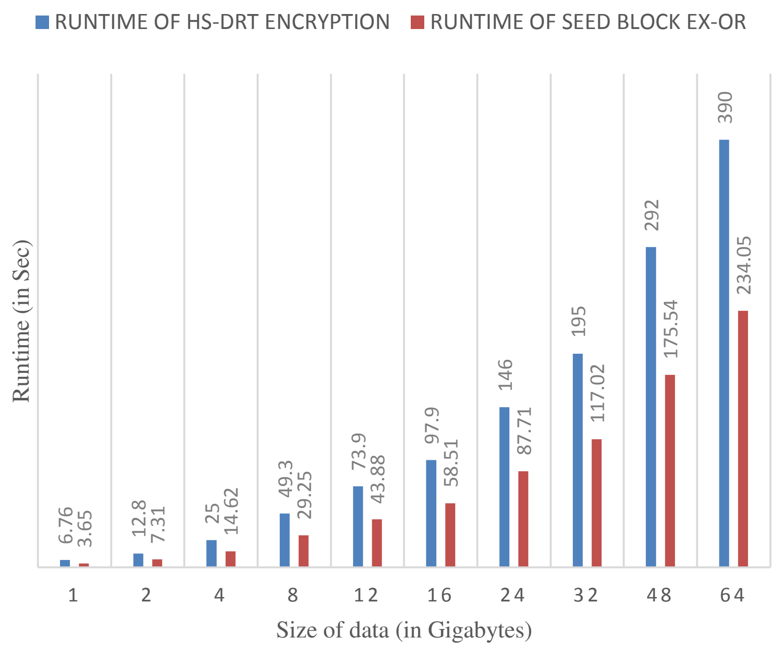
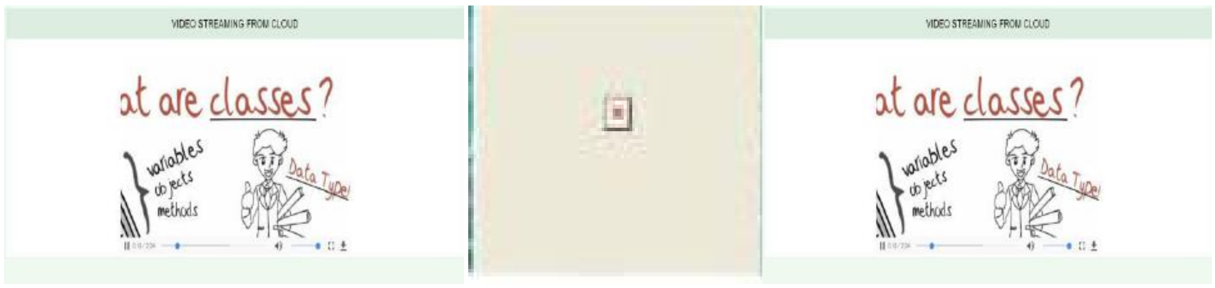


Fig. 10. Runtime of HS-DRT Encryption vs SBA EX-OR

From the above Fig. 10[4]. Evaluation result it is clear that the EX-OR operation in Seed block algorithm has comparatively less time complexity when compared with HS-DRT Encryption operation. Thus from this we can say that Seed block algorithm takes lesser time for the backup and recovery process comparing to the current existing system.

C. Sample output image of seed block algorithm

The Fig. 11 shows the experimentation result of proposed SBA. As fig. 11(a) shows the original file which is uploaded by the client on main cloud. Fig. 11 (b) shows the EXORed file which is stored on the remote server. This file contains the secured EXORed content of original file and seed block content of the corresponding client. Fig. 11(c) shows the recovered file; which indirectly sent to client in the absence of network connectivity and in case of the file deletion or if the cloud gets destroyed due to any reason.



(a) Original file **(b)** EX-ORed file **(c)**Recovered file

Fig. 11 Sample output image of Seed Block Algorithm

7. CONCLUSION:

This paper provides a solution for the failure in migrating services between datacenters by providing a backup storage at a remote location so that user lost data can be retrieved easily. Here Seed Block Algorithm is used which provides a Remote cloud (i.e., a cloud far away from the main cloud) for storing the data's of main cloud proposed for protecting user's data in which simple EX-OR operation is performed. From the comparison it is clear that recovery process is done in minimal time by using this algorithm, also the quality of services is improved. The Evaluation result clearly depicts that the system achieves data integrity and data security.

REFERENCES:

1. R. W. Ahmad, A. Gani, S. H. A. Hamid, M. Shiraz, A. Yousafzai, F. Xia,(2015) "A Survey on Virtual Machine Migration and Server Consolidation for Cloud DataCenters," Elsevier Journal of Net-work and Computer Applications, vol.52,11-25,
2. A.Beloglazov, J.Abawajy, R.Buyya (2012),"Energy-aware Resource Allocation Heuristics for Efficient Management of DataCenters for Cloud Computing," Future Generation Computer Systems, vol.28 (5), 755-768,.
3. M.Bienkowski, A. Feldman, D. Jurca, W. Kellerer, G. Schaffrath, S. Schmid, J. Widmer (2010), "Competitive Analysis for Service Migration in Vnets," SIGCOMM (workshop), 17-24,.
4. Chi-won Song, Sungmin Park, Dong-wook Kim, Sooyong Kang (2011), "Parity Cloud Service: A Privacy-Protected Personal Data Recovery Service," International Joint Conference of IEEE TrustCom-11/IEEE ICESS-11/FCST- 11,.
5. B. Dietrich, D. Goswami, S. Chakraborty, A. Guha, M. Gries (2015), "Time Series Characterization of Gaming Workload for Runtime Power Management," IEEE Transactions on Computers, 64(1), 260-273,.
6. Ms. Kruti Sharma, Prof. Kavita R Singh (2014), "A Remote Smart Data Back-up Technique for Cloud Computing" International Conference on Communication Systems and Network Technologies,.
7. X.Meng, V.Pappas, and L.Zhang, "Improving the Scalability of DataCenter Networks with Traffic-aware Virtual Machine Placement," INFOCOM,1-9,2010
8. Nikos Tziritas, Samee U.khan, Thanasis Loukopoulous, Spyros Lalis, Cheng-Zhong Xu, Senior Keqin Li, Albert Y.Zomaya (2017), "Online Inter-Datacenter Service Migrations" IEEE Transaction on cloud computing, 2168-7161,.
9. Sheheryar Malik, FabriceHuet,(December 2011), "Virtual Cloud: Rent Out the Rented Resources," 6th International Conference on Internet Technology and Secure Transactions,11-14, Abu Dhabi, United Arab Emirates,.
10. N. Tziritas, S. U. Khan, T. Loukopoulos, S. Lalis, C.-Z. Xu, P. Lampas (2013), "Distributed Online Algorithms for the Agent Migration Problem in WSNs," ACM / Springer Mobile Networks and Applications, 18(15), 622-638,.
11. Vijaykumar, Javaraiah Brocade, Advanced Networks and Telecommunication Systems (ANTS), 2011, "Backup for Cloud and Disaster Recovery for Consumers and SMBs," IEEE 5th International Conference, 2011.
12. Yoichiro Ueno, NoriharuMiyaho, ShuichiSuzuki, MuzaiGakuendai, Inzaishi, Chiba, Kazuo Ichihara,(2010), "Performance evaluation of a Disaster Recovery System and Practical Network System Applications," Fifth International Conference on Systems and Networks Communications, pp 256-259.