CLOUD COMPUTING: USE OF MULTI-CLOUDS

Shaik.Aafreen Naaz¹, Pothireddygari.Ramya², P.Vishunu Vardhan Reddy³

aafreenaaz.13@gmail.com, ramya666@gmail.com, tooclosevishnu.com@gmail.com

S.Vinay kumar* sunkesula.vinay@gmail.com

Department of IT, G.Pullaiah College Of Engineering and Technology

Abstract-
The use of cloud computing has increased rapidly in many organizations. Cloud computing provides many benefits in terms of low cost and accessibility of data. Ensuring the security of cloud computing is a major factor in the cloud computing environment, as users often store sensitive information with cloud storage providers but these providers may be untrusted. Dealing with “single cloud” providers is predicted to become less popular with customers due to risks of service availability failure and the possibility of malicious insiders in the single cloud. A movement towards “multi-clouds”, or in other words, “interclouds” or “cloud-of-clouds” has emerged recently. This paper surveys recent research related to single and multi-cloud and addresses possible solutions. It is found that the research into the use of multi-cloud providers to maintain security has received less attention from the research community than has the use of single clouds. This work aims to promote the use of multi-clouds due to its ability to reduce security risks that affect the cloud computing user.

Index terms-Multi-cloud infrastructure, Storage, Rain cloud system, Cost effective

I. INTRODUCTION
Cloud computing becomes the boom invention of today’s internet world. Through this technology users can consume services at any time by their particular needs. Before cloud computing, users have to buy individual or costly software, hardware resources but now it become easy to access the services on demand over the network [3]. It facilitates the user to access shared resources, common infrastructure or database resources, for as long as they need, without thinking about the cost and maintenance of resources.
It also provides facilities for consumer to develop and manage their own applications „on the cloud”, which enhance the concept of virtualization of resources. Through virtualization the resources are managed itself. The best example of cloud computing is Google docs where any document can be accessed using a browser and it can be shared on thousands of computer through the Internet.

The use of cloud computing has increased rapidly in many organizations. Cloud providers should address privacy and security issues as a matter of high and urgent priority. Dealing with “single cloud” providers is becoming less popular with customers due to potential problems such as service availability failure and the possibility that there are malicious insiders in the single cloud. In recent years, there has been a
move towards “multiclouds”, “intercloud” or “cloud-of-clouds”.

In the following diagram, the architecture of cloud computing is explained that the multiple clients are served by a cloud.

![Figure 1: Architecture of cloud computing](image1)

**II. DEPLOYMENT OF A MULTI-CLOUD VIRTUAL CLUSTER**

The distributed cluster testbed used in this work deployed on top of a multi-cloud infrastructure in Fig. 1. This kind of multi-cloud deployment involves several challenges, related to the lack of a cloud interface standard; the distribution and management of the service master images; and the interconnection links between the service components. A brief discussion of these issues, and the main design decisions adopted in this work to face up these challenges are included in the Appendix A of the supplemental material.

![Figure 2: Experimental framework](image2)

Our experimental testbed starts from a virtual cluster deployed in our local data center, with a queuing system managed by Sun Grid Engine (SGE) software, and consisting of a cluster front-end (SGE master) and a fixed number of virtual worker nodes (four nodes in this setup). This cluster can be scaled-out by deploying new virtual worker nodes on remote clouds. The cloud providers considered in this work are Amazon EC2 (Europe and USA zones) and ElasticHosts. Table 1 shows the main characteristics of in-house nodes and cloud nodes used in our experimental testbed.

| Data Center | In-House Nodes | Cloud Nodes (
| Amazon EC2 (Europe) |  | USA) | ElasticHosts |
|---|---|---|
| CPU | 2.4 GHz | 2.4 GHz |
| RAM | 8 GB | 32 GB |
| Storage | 100 GB | 500 GB |

DRAFT FOR IEEE TPDS (SPECIAL ISSUE ON MANY-TASK COMPUTING), JULY 2010

Besides the hardware characteristics of different testbed nodes, Table 1 also displays the cost per time unit of these resources. In the case of cloud resources, this cost represents the hourly cost charged by the cloud provider for the use of its resources. On the other hand, the cost of the local resources is an estimation based on the model proposed by Walker that takes into account the cost of the computer.
hardware, the cooling and power expenses, and support personnel.

<table>
<thead>
<tr>
<th>Site</th>
<th>Arch.</th>
<th>Processor (single core)</th>
<th>Mem. (GB)</th>
<th>Cost (USD/hour)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Local data center (L)</td>
<td>i686 32-bits</td>
<td>Xeon 2.0GHz</td>
<td>1.0</td>
<td>0.04</td>
</tr>
<tr>
<td>Amazon EC2 Europe (AE)</td>
<td>i686 32-bits</td>
<td>Xeon 1.2GHz</td>
<td>1.7</td>
<td>0.11</td>
</tr>
<tr>
<td>Amazon EC2 USA (AU)</td>
<td>i686 32-bits</td>
<td>Xeon 1.2GHz</td>
<td>1.7</td>
<td>0.10</td>
</tr>
<tr>
<td>ElasticHosts (EH)</td>
<td>AMD 64-bits</td>
<td>Opteron 2.1GHz</td>
<td>1.0</td>
<td>0.12</td>
</tr>
</tbody>
</table>

### Performance Analysis

In this section we analyze and compare the performance offered by different configurations of the computing cluster, focused in the execution of loosely-coupled applications. In particular, we have chosen 9 different cluster configurations (with different number of worker nodes from the three cloud providers), and different number of jobs (depending on the cluster size), as shown in Table 2. In the definition of the different cluster configurations, we use the following acronyms: L: Local infrastructure; AE: Amazon EC2 Europe cloud; AU: Amazon EC2 USA cloud; EH: ElasticHosts cloud. The number preceding the site acronym represents the number of worker nodes. For example, 4L is a cluster with four worker nodes deployed in the local infrastructure; and 4L+4AE is an eightnode cluster, four deployed in the local infrastructure and four in the Amazon EC2 Europe cloud. To represent the execution profile of loosely-coupled applications, we will use the Embarrassingly Distributed (ED) benchmark from the Numerical Aerodynamic Simulation (NAS) Grid Benchmarks [7] (NGB) suite. The ED benchmark consists of multiple independent runs of a flow solver, each one with a different initialization constant for the flow field. NGB defines several problem sizes (in terms of mesh size, iterations, and number of jobs) as classes S, W, A, B, C, D and E. We have chosen a problem size of class B, since it is appropriate (in terms of computing time) for middle-class resources used as cluster worker nodes. However, instead of submitting 18 jobs, as ED class B defines, we have submitted a higher number of jobs (depending on the cluster configuration, see Table 2) in order to saturate the cluster and obtain realistic throughput measures. As we have proven in a previous work [8], when executing loosely coupled high throughput computing applications, the cluster performance (in jobs completed per second) can be easily modeled using the following equation:

\[
r(n) = \frac{r_1}{1 + n^{1/2}/n}
\]

where \(n\) is the number of jobs completed, \(r_1\) is the asymptotic performance (maximum rate of performance of the cluster in jobs executed per second), and \(n^{1/2}\) is the half-performance length. For more details about this performance model see Appendix C of the supplemental material.
Fig. 2 shows the experimental cluster performance and that predicted by (1). As can be observed from these plots the performance model defined in equation (1) provides a good characterization of the clusters in the execution of the workload under study. If we compare 4L and 4EH configurations, we observe that they exhibit very similar performance. This is because of two main reasons: first, worker nodes from both sites have similar CPU capacity (see Table 1); and second, communication latencies for this kind of loosely coupled applications do not cause significant performance degradation, since data transfer delays are negligible compared to execution times, mainly thanks to the NFS file data caching implemented on the NFS clients (worker nodes), which notably reduces the latency of NFS read operations. On the other hand, the lower performance of 4AE and 4AU configurations is mainly due to the lower CPU capacity of the Amazon EC2 worker nodes (see Table 1).

An important observation is that cluster performance for hybrid configurations scales linearly. For example, if we observe the performance of 4L+4AE configuration, and we compare it with the performance of 4L and 4AE configurations separately, we find that the sum of performances of these two individual configurations is almost similar to the performance of 4L+4AE configuration. This observation is applicable to all of the hybrid configurations, as shown in Fig. 3. This fact proves that for the particular workload considered in this work, the use of a multi-cloud infrastructure spanning different cloud providers is totally viable from the point of view of performance and scalability, and does not introduce important overheads, which could cause significant performance degradation.

III. DATA DISTRIBUTION FOR MULTI-CLOUDS

Cloud data storage redefines the issues targeted on customer’s out-sourced data (data that is not stored/retrieved from the customer’s own servers). In this work we observed that, from a customer’s point of view, relying upon a solo SP for his outsourced data is not very promising. In addition, providing better privacy as well as ensure data availability and reliability can be achieved by dividing the user’s data block into data pieces and distributing them among the available SPs in such a way that no less than a
threshold number of SPs can take part in successful retrieval of the whole data block. In this paper, we propose a traditional technique followed to distribute data to multi-cloud storage model in cloud computing which holds an economical distribution of data among the available SPs in the market, to provide customers with data availability as well as reliability. Data fragmentation plays an important role in data distribution.

Data preservation and data integrity are two of the most critical security issues related to user data. In conventional paradigm, the organizations had the physical possession of their data, and thus have an ease of implementing better data availability policies. But in case of cloud computing, the data is stored on an autonomous business party, that provides data storage as a subscription service. The users have to trust the cloud service provider (SP) with security of their data. In the author discussed the criticality of the privacy issues in cloud computing, and pointed out that obtaining an information from a third party is much more easier than from the creator himself. One more bigger concern that arises in such schemes of cloud storage services, is that, there is no full-proof way to be certain that the service provider does not retains the user data, even after the user opts out of the subscription. With enormous amount of time, such data can be decrypted and meaningful information can be retrieved and user privacy can easily be breached. In order to stop the SP to observe the data, data can be fragmented and distributed to several SP’s.

**Why Fragment?**
- Usage
- Applications work with views rather than entire relations.
- Efficiency

- Data is stored close to where it is most frequently used.
- Data that is not needed by local applications is not stored
- Parallelism
- With fragments as unit of distribution, transaction can be divided into several sub-queries that operate on fragments.
- Security
  - Data not required by local applications is not stored and so not available to unauthorized users

**Types of Fragmentation**

Four types of fragmentation:
- Horizontal,
- Vertical,
- Mixed,
- Derived.
- Other possibility is no fragmentation

If relation is small and not updated frequently, may be better not to fragment relation

**Horizontal Fragmentation**
- Each fragment consists of a subset of the tuples of a relation R.
- Defined using Selection operation of relational algebra:
  - $\sigma p(R)$
- Example:
- Relation: Sells(pub,address,price,type)
- Fragments:
  - SellsBitter = $\sigma$ type = “bitter”(Sells)
  - SellsLager = $\sigma$ type = “lager”(Sells)
- This strategy is determined by looking at predicates used by transactions.
- Involves finding set of minimal (complete and relevant) predicates.
- Set of predicates is complete, if and only if, any two tuples in same fragment are
referenced with same probability by any application.

- Predicate is relevant if there is at least one application that accesses fragments differently

**Vertical Fragmentation**

- Each fragment consists of a subset of attributes of a relation R.
- Defined using projection operation of relational algebra:
  \[ \Pi_{a_1, \ldots, a_n}(R) \]
- Determined by establishing *affinity* of one attribute to another.
- Example:
  - Relation: \( \text{Bars}(\text{name}, \text{address}, \text{licence}, \text{employees}, \text{owner}) \)
  - Fragments:
    - \( \Pi_{\text{name}, \text{address}, \text{licence}}(\text{Bars}) \)
    - \( \Pi_{\text{name}, \text{address}, \text{employees}, \text{owner}}(\text{Bars}) \)

**Mixed Fragmentation**

- We can also mix horizontal and vertical fragmentation.
- We obtain a fragment that consist of an horizontal fragment that is vertically fragmented, or a vertical fragment that is horizontally fragmented.
- Defined using Selection and Projection operations of relational algebra.
  \[ \sigma_{p}(\Pi_{a_1, \ldots, an} R) \]

**Derived Horizontal Fragmentation**

- A horizontal fragment that is based on horizontal fragmentation of a parent relation.
- Ensures that fragments that are frequently joined together are at same site.
- Defined using Semijoin operation of relational algebra:

\[ \text{Ri} = R \bowtie F S_i, \quad 1 \leq i \leq w \]

- If relation contains more than one foreign key, need to select one as parent.
- Choice can be based on fragmentation used most frequently or fragmentation with better join characteristics.

**Three correctness rules:**

**Completeness:** If relation R is decomposed into fragments \( r_1, r_2, \ldots, r_n \), each data item that can be found in R must appear in at least one fragment. This ensures no loss of data during fragmentation.

**Reconstruction:** we must be able to reconstruct the entire R from fragments. For horizontal fragmentation it is union operation.

\[ R = r_1 \cup r_2 \cup \ldots \cup r_n \]

For vertical fragmentation it is natural join operation.

\[ R = r_1 \bowtie r_2 \bowtie \ldots \bowtie r_n \]

To ensure reconstruction we have to include primary key attributes in all fragments.

**Disjointness:** If data item x appears in fragment \( r_i \), then it should not appear in any other fragment.

**Exception:** vertical fragmentation, where primary key attributes must be repeated to allow reconstruction. For horizontal fragmentation, data item is a tuple. For vertical fragmentation, data item is an attribute.

Correctness of Horizontal Fragmentation

**Relation:** \( \text{Sells}(\text{pub}, \text{address}, \text{price}, \text{type}) \)

**type=\{Bitter, Lager\}**

- **Fragments:**
  - \( \text{SellsBitter} = \sigma_{\text{type} = "bitter"}(\text{Sells}) \)
  - \( \text{SellsLager} = \sigma_{\text{type} = "lager"}(\text{Sells}) \)

**Correctness rules**

**Completeness:** Each tuple in the relation appears either in \( \text{SellsBitter} \), or in \( \text{SellsLager} \)

**Reconstruction:** The \( \text{Sells} \) relation can be reconstructed from the fragments

\[ \text{Sells} = \text{SellsBitter} \bowtie \text{SellsLager} \]
**Disjointness:** The two fragments are disjoint, there can be no beer that is both “Lager” and “Bitter”

Correctness of Vertical Fragmentation Relation:
Bars(name,address,licence,employees,owner)

Fragments:
r1=Πname,address,licence (Bars)
r2= Πname,address,employees,owner(Bars)

Correctness rules
Completeness: Each attribute in the Bars relation appears either in r1 or in r2

**Reconstruction:** The Bars relation can be reconstructed from the fragments

**Disjointness:** The two fragments are disjoint, except for the primary key, name, which is necessary for reconstruction

Our model distributes the data pieces among more than one service providers, in such a way that no one of the SP s can retrieve any meaningful information from the pieces of data stored on its servers, without getting some more pieces of data from other service providers. Therefore, the conventional single service provider based techniques does not seem too much promising.

Distributing the data over multiple clouds or networks in such a way that if an adversary is able to intrude in one network, still he cannot retrieve any meaningful data, because its complementary pieces are stored in the other network. Our approach is similar to this approach, because both aim to remove the centralized distribution of cloud data. Although, in their approach, if the adversary causes a service outage even in one of the data networks, the user data cannot be retrieved at all. This is why in our model; we propose to use a redundant distribution scheme in which at least a threshold number of pieces of the data are required out of the entire distribution range, for successful retrieval.

Meaningful information from the data pieces allocated at their servers. Also, in addition, we provide the user with better assurance of availability of data, by maintaining redundancy in data distribution. In this case, if a service provider suffers service outage or goes bankrupt, the user still can access his data by retrieving it from other service providers. From the business point of view, since cloud data storage is a subscription service, the higher the data redundancy, the higher will be the cost to be paid by the user

**IV. SECURED COST EFFECTIVE MULTI-CLOUD STORAGE**

Privacy preservation and data integrity are two of the most critical security issues related to user data. In conventional paradigm, the organizations had the physical possession of their data and hence have an ease of implementing better data security policies. But in case of cloud computing, the data is stored on an autonomous business party, that provides data storage as a subscription service. The users have to trust the cloud service provider (SP) with security of their data.

Following the pattern of paradigm shift, the security policies also evolved from the conventional cryptographic schemes applied in centralized and distributed data storage, for enabling the data privacy. Many of the cryptographic approaches have been proposed for hiding the data from the storage provider and hence preserving data privacy. In , the authors proposed a scheme in which, the user’s identity is also detached from the data, and claim to provide public auditing of data. These approaches concentrate on one single cloud service provider, that can easily become a bottleneck for such services. In , the authors studied and proved that sole cryptographic measures are insufficient for ensuring...
data privacy in cloud computing. They also argued that the security in cloud storage needs a hybrid model of privacy enforcement, distributed computing and complex trust ecosystems. One more bigger concern that arises in such schemes of cloud storage services, is that, there is no full-proof way to be certain that the service provider does not retain the user data, even after the user opts out of the subscription. With enormous amount of time, such data can be decrypted and meaningful information can be retrieved and user privacy can easily be breached. Since, the user might not be availing the storage services from that service provider, he will have no clue of such a passive attack. The better the cryptographic scheme, the more complex will be it’s implementation and hence the service provider will ask for higher cost. This could also lead to a monopoly over cloud services in the market. To provide users with better and fair chances to avail efficient security services for their cloud storage at affordable costs, our model distributes the data piece among more than one service providers, in such a way that no one of the SPs can retrieve any meaningful information from the pieces of data stored on its servers, without getting some more pieces of data from other service providers. Therefore, the conventional single service provider based cryptographic techniques does not seem too much promising. In, the authors discussed distributing the data over multiple clouds or networks in such a way that if an adversary is able to intrude in one network, still he can not retrieve any meaningful data, because its complementary pieces are stored in the other network. Our approach is similar to this approach, because both aim to remove the centralized distribution of cloud data. Although, in their approach, if the adversary causes a service outage even in one of the data networks, the user data can not be retrieved at all. This is why in our model, we propose to use a redundant distribution scheme, such as in, in which at least a threshold number of pieces of the data are required out of the entire distribution range, for successful retrieval.

V. RAIN-CLOUDS SYSTEM

The term „Rain Cloud System” is simply a collection of several clouds. These clouds collectively form a group of excess resources like rain drops and reduce the drought or lack of resources in a network [1]. The condition of drought occurs mainly in private clouds. Mostly the private firms, government organization or universities have chosen „private clouds”, because they do not allow crucial information to be exposed. There could be several problems occurred in private clouds such as lacking of hardware and software resources, network, congestion of packets, data become bottleneck etc. These problems always either slow down the network, loss of packets or information. To resolve such kind of problems „multi-clouds” or „rain clouds” are introduced. These clouds provide access to resources in emergency situation like pouring rain drops to clients or users.

Rain computing deal with the large volume of data traffic or services that cannot be dealt with the single cloud system. now a days, as the number of smart devices or PDA’s increases with a rapid rate, the provider have to concern more about the security and the availability of the services. When a firm enters the cloud and put into a service, it first starts with IaaS (Infrastructure as a service) and later transform into PaaS (Platform as a service). In case of online examination or online banking, the demand of service or resources (such as memory, server and network) rapidly increases. For this instance the lack of services results in bottleneck of data or network.
congestion which slows down the network or incomplete transactions. For such circumstances, the cloud will not recover the damage or expanding the capacity within the transaction time.

To solve such problems, the only one solution is to connecting to another cloud.

A. Characteristics of Rain-Clouds
1. Rapid Elasticity
2. Broad network Access
3. Rapid Connectivity within clouds

VI. RAIN-CLOUDS ARCHITECTURE

The design architecture of Rain cloud consist of number of nodes (such as users) connected with the number of clouds within the network. Every node is connected to its owner cloud and also able to connect to its neighbor cloud whenever required. As in the figure, there are four clouds and four clients that are interconnected through the communication links. The whole rain cloud system deals with different cloud providers but they exist in a particular organization as it is collection of several private clouds. The main point of rain cloud system is that it works only in the single or private organization[1]. It is not reliable under public cloud system.

The rain cloud system uses three Messages to communicate.

a) Service Request Message (SRQM)
b) Service Response Message (SRSM)
c) Service Level Agreement(SLA)

Suppose Client 4 sends SRM to the cloud 1 and waits for the acknowledgement. Cloud 1 also serves the client 2 and client 3 at that time. Client 4 repeatedly sends request messages to the cloud 1. Atlast cloud 1 is congested with the number of request messages and it temporarily fails down. In the case, neighbour clouds provides the essential services to the client 4. The Service Level Agreement (SLA) messages are used to linked the rain clouds with various public clouds that allow them to receive services, if all clouds are unable to serve then send request to other public clouds. The following diagram shows the architecture design of a Rain-Cloud system

Figure 2: Rain cloud Architecture

The rain cloud architecture always try to maintain the network congestion free because in small organizations the load of data traffic rapidly increases that causes severe problems. Such as,

If (n=f) then
n=n+1 where n= number of storage clouds
f= faulty clouds

If current storage cloud becomes faulty then control shift to another cloud. Rain cloud system act as a parallel files system allowing reading and writing of data in the back-end simultaneously.
VII. CONCLUSION

In this paper we have analyzed the challenges and viability of deploying a computing cluster on top of a multi-cloud infrastructure. We also proposed different data fragmentation schemes for multi cloud storage in cloud computing, which seeks to provide each customer with reliability, availability and better cloud data storage decisions. We proposed a secured cost-effective multicloud storage (SCMCS) in cloud computing, which seeks to provide each customer with a better cloud data storage decision, taking into consideration the user budget as well as providing him with the best quality of service (Security and availability of data) offered by available cloud service providers. The main point is to be focused that Rain-cloud system is a service necessary in emergency situations, when single cloud cannot process properly due to overload or congested from data traffic. The Further studies states to securely obtain the additional resources and availability of data is necessary. Security problems must be the first priority and thus switching of connection between the clouds must be with immediate reaction without any delay.

VIII. REFERENCES


