Providing a Solution for Live Migration of Virtual Machines in Eucalyptus Cloud Computing Infrastructure without Using a Shared Disk

Shayan Zamani Rad
Mazandaran University of Science and Technology
Computer Engineering and IT Department
Babol, Iran
sh.zamani@ustmb.ac.ir

Morteza Sargolzai Javan
Amirkabir University of Technology
Computer Engineering and IT Department
Tehran, Iran
msjavan@aut.ac.ir

Mohammad Kazem Akbari
Amirkabir University of Technology
Computer Engineering and IT Department
Tehran, Iran
akbarif@aut.ac.ir

Abstract— **Today, cloud computing is used as a model in most scientific, commercial, military, other fields. In this model, the main body of the system are virtual servers, which currently provide services to customers around the world. In these circumstances, since the servers are virtual, they can be transferred as a file from one machine to another, which is known as migration. Migration practice is done for a variety of purposes, including load balancing, fault tolerance, power management, reducing response time, increasing quality of service, and server maintenance. Because the use of this technique is highly dependent on cloud computing infrastructure architecture, in some cloud infrastructures, such as Eucalyptus, the virtual machine migration technique has not been used yet. In this paper, we propose a solution for VM migration technique on Eucalyptus Cloud environment. The experiments show the validity of the proposed solution in non-shared disk Cloud environments, where the total migration time and transferred data have been significantly increased.**

Keywords-Eucalyptus; Cloud computing infrastructure; Virtual Machine; Migration.

I. INTRODUCTION

Cloud computing has been considered as a new way of providing Information Technology services to individuals and organizations. In addition, given the increase tendency of users and companies, academic and research centers strive to provide solutions and new tools in the Cloud computing. While commercial products are offered with the goal of cost reduction and customer satisfaction, productivity tools in academic centers are to discover new solutions based on open source technologies. Eucalyptus is an open-source Cloud-computing framework that uses computational and storage infrastructure which is commonly available to academic research groups to provide a platform that is modular and open to experimental instrumentation and study [7]. One of the weaknesses in this Cloud framework is the lack of virtual machine (VM) migration technique. Given that migration technique is done for different purposes, such as load balancing, fault tolerance, power management, reducing response time and increasing quality of service, server maintenance, etc. Therefore, there is not any mentioned algorithm in Eucalyptus. In this paper, we implemented migration technique by presenting solution in the Eucalyptus and establish a basis for providing other security and management algorithms.

The rest this paper is organized as follow: In Section 2, we will describe different migration methods. In Section 3, we examine the architecture of Eucalyptus with its components. In Section 4, we will describe the proposed method. Finally, in Section 5, we evaluate our methods.

II. RELATED WORKS

Currently, there are some works on the migration which can be referred to Pre-copy method in [2] [3] [4]. This method has three steps for migration a Virtual Machine: in the first step, the virtual machine memory pages are transferred in several rounds and then, in step two, the virtual machine CPU states are sent to the destination. After that in step three, the memory pages in source and destination are synched with each other.

Also, Hines and Gopalan [5] present a Post-copy technique with optimization methods. In [6], the method of CR/TR has been presented, which aims to send Logs file instead memory pages toward the destination.

Considering the benefits of Pre-copy approach, this is the main migration method, which is supported in most Hypervisors such as XEN and KVM. Thus, we used Pre-copy for sending memory pages and CPU states of virtual machines. However, Pre-copy approach has some shortcomings; one of these shortcomings is the lack of disk migration (transfer) algorithm. Hence, we cannot use default Pre-copy method in the non-shared disk environments, because the virtual machine disk must be transferred.

In this condition, our method has a disk transmission algorithm which is not dependent on the shared disk and can be used in the above-mentioned environments. In addition,

 ISBN: 978-1-61208-216-5

use of Pre-copy and other migration methods in Eucalyptus is impossible, because it has certain challenges; therefore, in our method, these challenges have been solved and this is an important difference between our method and that of others.

III. EUCALYPTUS

Eucalyptus is an open source implementation of Cloud computing infrastructure that has a particular architecture. By using Eucalyptus, we can make public and private Clouds. The architecture of the Eucalyptus system is simple, flexible and modular with a hierarchical design, reflecting common resource environments found in many academic settings. In essence, the system allows users to start, control, access, and terminate entire virtual machines using an emulation of Amazon EC2's SOAP and "Query" interfaces [7]. That is, users of Eucalyptus interact with the system using exactly the same tools and interfaces that they use to interact with Amazon EC2 [8].

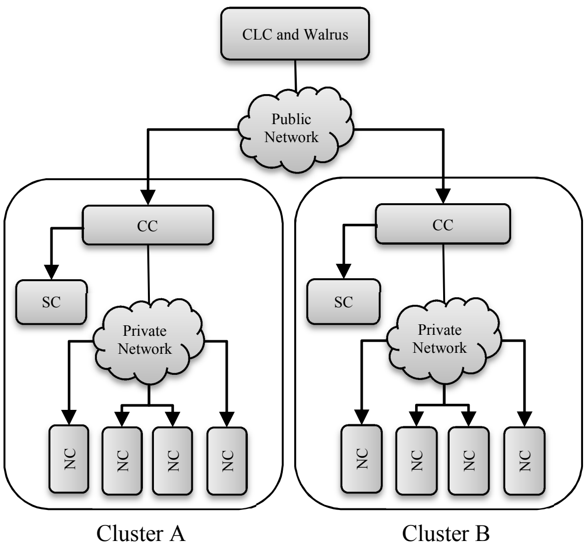

Figure 1. Eucalyptus Architecture [7]

The Eucalyptus is composed of five main components [7]:

- Cloud Controller (CLC): This component is frontend of the infrastructure and through a Web interface interacts with users and provides possibility of controlling virtual machine.
- Walrus: It is a put/get storage service that implements Amazon's S3 interface, providing a mechanism for storing and accessing virtual machine images and user data.
- Cluster Controller (CC): This component is responsible for the management of one or more node controllers. It also manages and sends the order of running of the instances on them. In Eucalyptus, Virtual Machine is known as Instance.
- Storage Controller (SC): This component provides virtual disks for instances, allowing them to permanently store and keep the information. This is very similar to the EBS service.
- Node Controller (NC): It controls the execution, inspection, and terminating of VM instances on the host where it runs.

The Eucalyptus architecture and its components are shown in Figure 1.

IV. CHALLENGES

There are some challenges in implementing the migration technique in Eucalyptus Cloud Infrastructure. These challenges do not allow to implementing ordinary VM migration methods; in fact, these challenges are the properties of Eucalyptus.

A. *Clearing the Instance Data after Turning it off*

In Eucalyptus, when the instance is turned off even temporarily, its information would be completely removed from CC, NC and CLC. However, the instance would enter in suspend mode for a short time (60 milliseconds) in all migration methods.

B. *Operations Management is Performed by CLC*

All operations and activities must be performed under the CLC and the CC, and if any actions get implemented without these two components, the structure of Eucalyptus will change. But, in all migration methods, migration operations are done under the hypervisor. Now, the NC and the hypervisor are executive components in Eucalyptus, and all operations will report to the higher administrative units.

C. *Lack of Shared Disk*

In most migration methods, the disk of virtual machine is considered as a shared disk that is the source and destination hypervisors have access to it. Therefore, during the migration process, only memory pages and CPU states are displacing, but if there is no shared disk available in Eucalyptus, the disk must be transferred when moving a VM.

D. *Some Common Mistakes*

In Eucalyptus, when instances are displaced, some information must be updated and changed, e.g., available resources, the number of being established instances, the number of running instances, etc. If any of this information has incorrect content, Eucalyptus performance and overall cloud would decrease and its structure might be out of control.

E. *Instance Death Zone Time*

In Eucalyptus, if the running instance cannot send any response (heart beat) to NC in ranged 20 to 24 seconds for any reasons, CLC will assume that instance is terminated. Then, CLC release it from the list of running instances and delete all information about it. Consequently, API functions will not be used for this instance, because the CLC would not know an instance with this name. Therefore, migration operation should be less than the mentioned time period which we have named it the "Instance Dead Zone Time".

 ISBN: 978-1-61208-216-5

V. SYSTEM ARCHITECTURE

In Eucalyptus, all commands are issued by the CLC, and the CC and NC behave as an observer and a worker (commands runner), respectively. Furthermore, the user (client) and cloud administrator input their requests to the CLC through running API functions. Therefore, in order to create the migration capability in Eucalyptus, we first created an API function that is responsible for the migration. This function has the following format:

euca-migrate-instance -i instance_id –d destination_node

Through running *euca-migrate-instance* API, first, the CLC finds node's IP address that the instance is currently running there. Then, these three values are transferred to the related cluster controller (where the cluster is running the instance). Figure 2 shows the process in the CLC.

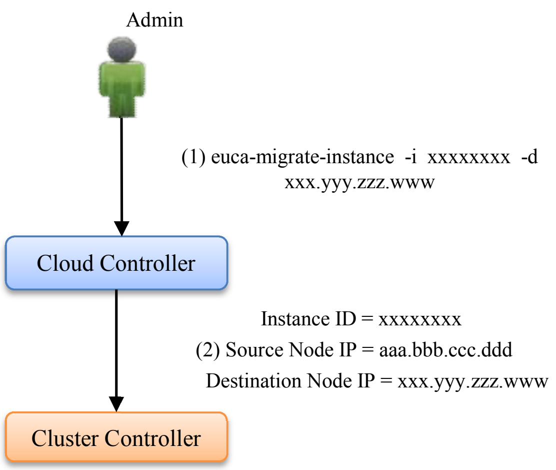

Figure 2. CLC with defined API Function

After CC's stub receives commands, it checks the needed resource amounts and makes decisions about sending migration command to the appropriate nodes.

After sending the migration command from CC to source node (NC), created stub in the NC receives related information (Figure 3). Next, it tries to communicate with the destination node and its hypervisor. After establishing relations, the source hypervisor would send memory pages and CPU states with uses Pre-copy method (Figure 4). After a few seconds, the migration process ends. Now, the instance is running on the destination node. Finally, the instance's information must be updated on the source and destination NCs, the CC and the CLC.

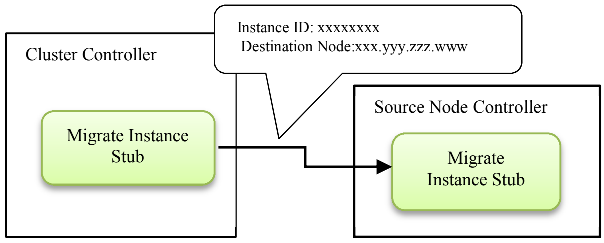

Figure 3. Relationship of CC and NC through stubs

At this point, the destination NC updates instance's information, which is running on it. Afterward, source NC removes instance's name and information from the own list of running instances. Next, the CC updates its information and changes instance location. This change causes the following executive orders to be sent to the destination NC. These orders include terminate, reboot, attach and detach volumes to instance.

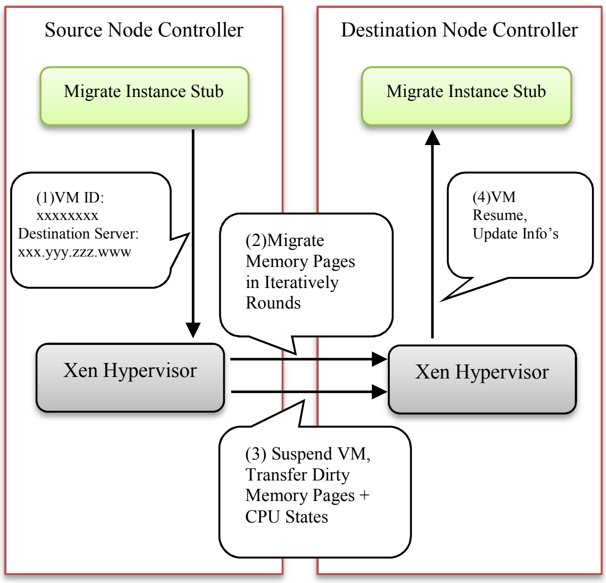

Figure 4. Relationship between Source and Destination NCs for Migration Operation

A. Disk Transfer Algorithm

In Eucalyptus, if there is no available shared disk, when the instance is migrated, its disk must be transferred. So, if the Xen hypervisor [9] is used without any changes for instance migration, instance moves with memory in destination node and its disk is located in source node. As a result, this is a fault in migration process. Therefore, it must be used an algorithm to transfer instance's disk. This algorithm must be written within the hypervisor. At the beginning of migration process, exactly before transferring memory pages and CPU states, the instance's disk must be transferred to the destination node. In fact, transmission of disk blocks lasts usually much longer than the transferring memory pages and CPU states. Thus, the disk transfer must be done before memory pages and CPU states transfer begin. In the following, we illustrate our proposed disk transferring algorithm:

```
1. for i=min to max (number of disk blocks)
2.    Begin to Transfer Blocks
3.    if I/O Request is coming and  I/O = WRITE then{
4.       block-bitmap[i] = 1}
5. While bock-bitmap[i]=1 {
6.    Transfer Dirty Block_i to Destination}
7. End
```

Pseudo-code of Disk Transfer Algorithm

Also, we have used LZO [10] compression algorithm in our disk transfer algorithm presented. The disk blocks in the source node are compressed before transmission. And then, they are sent to the destination node to be decompressed. This action causes optimal use of available bandwidth and also the data will be transferred in much less time than it is usually the case (without compression). Figure 5 shows the steps of our migration method.

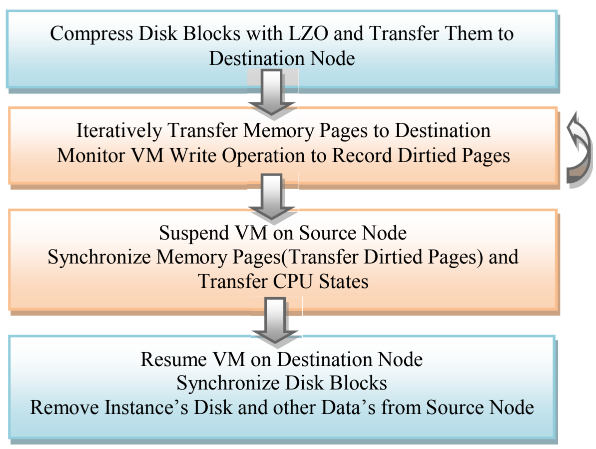

Figure 5. Steps of our migration method

VI. IMPLEMENTATION AND EVALUATION

In Eucalyptus environment, each running instance gives service to one or more customers; so, in instance migration process, the total migration time, network throughput and response time (to customers) are very important and must be evaluated. We could use other experiments such as disk performance evaluate (with the bonnie++ benchmark), but when the instance's disk must be transferred, this evaluation would not be helpful. We used two scenarios to evaluate our migration technique with above criteria. Each experiment was performed three times and average values were recorded.

A. Evaluation Environment

In order to implement Eucalyptus components and create a Cloud environment, we used two machines with AMD Quadro Core 800 MHz processor with disk capacity 500 GB and 8GB memory as NCs. Furthermore, we installed Xen 3.4 hypervisor on the NCs. Also, we have a machine with Intel Core2Duo 2.66GHz CPU, 320 GB disk capacity and 4 GB of memory as CLC and CC. These three machines are connected through a LAN network with 100 Mbps bandwidth. The migration operation is performed on the instance; it has one vCPU, 2GB disk space with 128 MB memory. On all components, the Linux Centos 5.6 OS is also installed. You see evaluation environment in Figure 6.

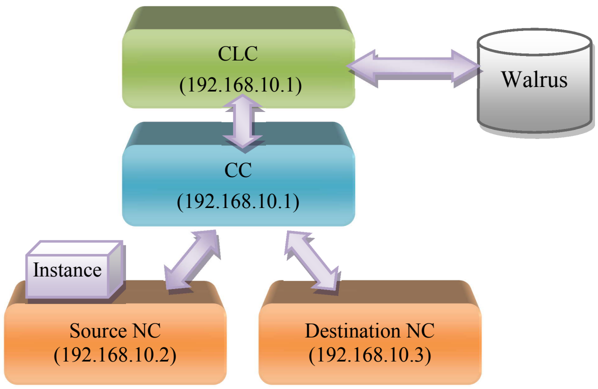

Figure 6. The Cloud Implemented Environment

B. First Test: Network Throughput

In the first experiment, we evaluated the network bandwidth of instance during migration in terms of throughput. In this test, we have used Netperf benchmark, version 2.5.0 in order to measure the network bandwidth in normal conditions (not using migration technique) and when the instance is migrating. You see the test results in Figure 7. As one can see in the figure, when the instance is suspended and moved from origin to destination, throughput rate of the network is reduced by half.

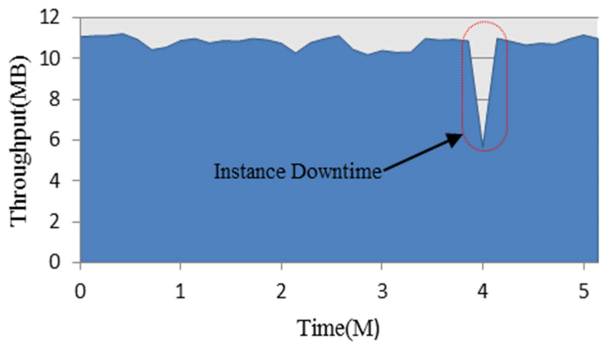

Figure 7. The Throughput of the Network during Instance Migration

C. Second Test: Response Time

The aim of this test is to evaluate response time to incoming requests by the instance during the migration operation. In this test, the instance functions as a server that provides a service to the users and begins migrating from a place to another place.

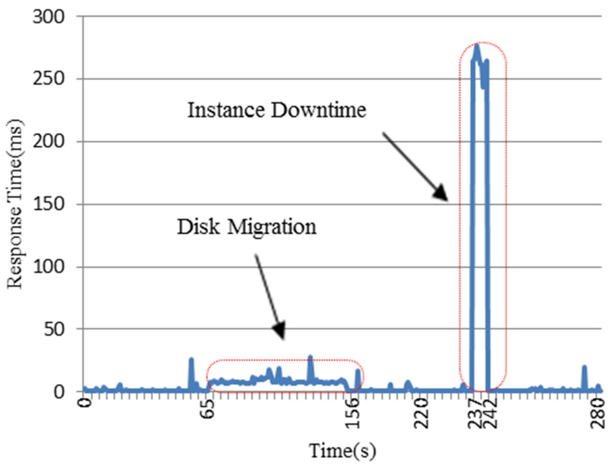

Figure 8. Response Time to Incoming Requests during Instance Migration

The results of this test have been shown in Figure 8. At first, when the migration process is starting, disk blocks compress. Afterward, disk transfer starts in second 65 and end on second 156. Now, destination node decompresses received disk blocks and then sends a message based on preparing to receive memory pages to source node. Next, in second 220, the operation of transferring memory pages is started. After a few seconds, in second 237, the instance was entered in suspend state. Now, memory pages and CPU states are transferred to destination. After a few seconds, in second 242, instance will resume working on the destination node and synchronization operation of disk blocks between source and destination was began from second 243 to 280.

VII. CONCLUSION

In this paper, we proposed a way to implement the migration of virtual machines on Eucalyptus Cloud Infrastructure. As mentioned, Eucalyptus has a unique architecture; thus, our method should be compatible with this architecture, so our main purpose is compatibility. Considering that there has been no migration feature in this cloud environment yet, other management features and algorithms such as load balancing and power management have not been implemented. Through using the method, we provided the field with further development and empowerment of the Cloud environment, and paved the way for the creation of more powerful algorithms in future.

REFERENCES

[1] Sh.Z. Rad, M.S. Javan, and M.K, Akbari, “A survey on virtual machine migration methods and performance evaluations”, First CSUT Conference on Computer, Communication, Information Technology (CSCCIT), Tabriz, Iran, 2011.

[2] C.P. Sapuntzakis, R. Chandra, B. Pfaff, J. Chow, M.S. Lam, and M. Rosenblum, "Optimization the migration of virtual computers", In Proceeding of 5th USENIX Symposium on Operating Systems Design and Implementation (OSDI-02), December 2002.

[3] C. Clark, K. Fraser, S. Hand, J.G. Hansen, E. Jul, C. Limpach, I. Pratt, and A. Warfield, “Live migration of virtual machines”, In Proc. of the second USENIX Symposium on Networked Systems Design and Implementation (NSDI), Boston, MA, USA, May 2005.

[4] R. Bradford, E. Kotsovinos, A. Feldmann, and H. Schioberg, “Live wide-area migration of virtual machines with local persistent state”, VEE’07, June 2007.

[5] M. R. Hines and K. Gopalan, “Post-copy based live virtual machine migration using adaptive pre-paging and dynamic self-ballooning”, Proceeding of the 2009 ACM SIGPLAN/SIGOPS International Conf. on Virtual Execution Environments, Binghamton University (State University of New York), Feb 2009.

[6] H. Liu, H. Jin, X. Liao, L. Hu, and C. Yu, “Live migration of virtual machine based on full system trace and replay”, in Proceedings of the 18th International Symposium on High Performance Distributed Computing (HPDC’09), 2009, pp.101–110.

[7] D. Nurmi, R. Wolski, C. Grzegorczyk, G. Obertelli, S. Soman, L. Youseff, and D. Zagorodnov, “Eucalyptus open source Cloud-computing system”, In CCA08: Cloud Computing and Its Applications, 2008.

[8] Amazon Elastic Compute Cloud (Amazon EC2), http://aws.amazon.com/ec2/, [retrieved: May, 2012].

[9] P. Barham, B. Dragovic, K. Fraser, S. Hand, T. Harris, A. Ho, R. Neugebauer, I. Pratt, and A. Warfield. “Xen and the art of virtualization.” In SOSP ’03: Proceedings of the nineteenth ACM symposium on Operating systems principles, pages 164–177, New York, NY, USA, 2003. ACM.

[10] M.F.X.J. Oberhumer, “LZO – a real-time data compression library”, http://www.oberhumer.com/opensource/lzo/, [retrieved: May, 2012].