ISSN (Online) : 2319 - 8753
ISSN (Print) : 2347 - 6710

IJIRSET

International Journal of Innovative Research in Science, Engineering and Technology

Volume 3, Special Issue 3, March 2014

2014 International Conference on Innovations in Engineering and Technology (ICIET'14)
On 21st & 22nd March Organized by

K.L.N. College of Engineering, Madurai, Tamil Nadu, India

A Migration of Virtual Machine to Remote System

R. Lakshmi Prabha[#1], N. Revathi[*2]

[#]PG Scholar, Department of Information Technology, Sri Venkateswara College of Engineering, Chennai, India.

[*]Assistant Professor, Department of Information Technology, Sri Venkateswara College of Engineering, Chennai, India.

Abstract - Virtualization allows more than one operating systems to run on the same machine. Migration of virtual machine is an important feature in virtualization. Migration of virtual machine from one physical machine to another physical machine allows achieving load balancing and fault tolerance easily. In past migration was done between the hosts which are in the same configuration. However the achievable downtime in a homogeneous migration is not acceptable. A new system model is proposed for achieving the migration of virtual machine in a heterogeneous environment that removes the page retransmissions and an achievable downtime of the order of milliseconds.

Key Words- Virtualization, Migration, Virtual Machine

I. INTRODUCTION

Virtualization plays a major role in cloud computing. Heterogeneity is one of the major characteristics in virtualization. It provides resources for the guest operating system and also used in data centers for managing their workloads and cloud execution environments. It does not depend on the physical machine to start its own operating system. Each virtual machine has its own virtual CPU, RAM and Network Interface Card (NIC). Heterogeneous migration is used for the migration of the running virtual machine from one host to another host without disrupting the virtual machine in the source host. The virtual machine in the target host can be resumed easily. Hypervisor is one part of virtualization that allows operating system to share the single hardware host. Each operating system has its own processor, memory and bandwidth. Live migration of virtual machine is used to decrease the end to end migration time and downtime.

Entire execution environment is migrated to new host that allows easy avoidance of the 'residual dependency' problem, which means that the source host remains available even after the migration is performed. Second, the migration of virtual machine is done in a consistent manner. Third, the user need not know the information about the content in the virtual machine. Overall, migration of virtual machine is more useful for administrators. Suppose if the physical machine is needed to run another application, migration can be used. The running virtual machine is migrated to another physical machine and frees the required physical machine. Virtualization and migration are combined to increase the manageability. For example, the downtime achieved in SPEC Web is only 210ms and in Quake3, it is just 60ms. They use the pre-copy approach, in which pages are continuously copied to target host without halting the virtual machine which is being migrated. Page level protection hardware is used to ensure that the transferred pages are consistent. Rate adaptive algorithm is used to avoid the traffic.

Process migration [7] is the concept in which only the processes are migrated between the two hosts and they do not support the liveliness and heterogeneity. Next work on this topic is collective project. The main goal is to optimize for ADSL [2] links and moreover the virtual machine running in source host is halted and then migrated. Zap [8] introduces the migration of process domains and their results are faster when compared with the above said collective project. Here also the live migration is not supported. In CPU virtualization [1], each virtual machine monitor has its own virtual CPU state which provides the abstraction of CPU to guest operating system. Virtual CPU state differs from vendor to vendor as virtual CPU state in Xen hypervisor has control registers, segment registers and timestamp counter whereas in Xen Para-virtualization has virtual TSS and IDT.

In memory virtualization [5], virtual machine manager monitors the physical memory and is responsible for allocating the memory for each virtual machine. The memory allocated for virtual machine is not contiguous. The guest operating system runs in two modes: direct mode and shadow mode. In direct mode, the guest operating system has only read access to page table entries. In shadow mode, the guest operating system has both read and writes access to page table entries. In I/O devices [3], [6] the model of guest operating system differs in Para-virtualization and full-virtualization. In Para-virtualization, the guest operating system source code can be modified and new device drivers for virtualization. There is no need to use existing device drivers for virtualization. In full-virtualization, existing device drivers are used for virtualization.

In this paper, a new system model is proposed that allows the running virtual machine to be migrated to new host in heterogeneous environment. The model has a common algorithm and abstraction. Source host virtual machine is first converted into one common format. On the target side, the model is again transformed from the common format to its own physical machine supported format. Algorithm is chosen dynamically from the algorithm pool.

II. HOMOGENEOUS MIGRATION

Homogeneous migration takes place between the two systems in same configuration. First migration takes place between the Xen Hypervisor. Live migration takes place in domU, the back storage of virtual machine has to be on a shared storage, like iSCSI. Before performing live migration Xen must be configured using,

```
/etc/xen/xend-config.sxp
```

The above command is used for editing options in the configuration file. Four lines are displayed after using the above command, which have a # character infront of each line. Erase the # character to uncomment the lines for performing live migration. The four lines are,

```
(xend-relocation-hosts-allow)

(xend-relocation-port 8002)

(xend-relocation-address)

(xend-relocation-server yes)
```

The two hosts must be configured after live migration. The code in Xen virtual machine is replaced with the required target virtual machine code using the following command,

```
Disk=[‘phy:/dev/disk-1/debian-base-xen,sda1,w’,‘phy:/dev/disk-1/debian-base-xen-swap,sda2,w’]
```

Copy the configuration file on the other host to run the live migration. Create a logical volume for two hosts and migrate a virtual machine from host 1 to host 2 by providing the virtual machine id in the command.

```
root@host_1 # scp /etc/xen/debian-base-xen.cfg root@host_2:/etc/xen/

root@host_2 # vgscan

root@host_2 # vgchange –a y

root@host_2 # vgdisplay vg_xen

root@host_1 # xm migrate –live 2 host_2
```

Live migration of virtual machine in KVM is simple. Virsh and SSH keyword is used for performing the migration. The main goal here is the migration of virtual machines without halting them and with almost no downtime. It uses pull phase for migration, during migration KVM monitors the source virtual machine for changes in pages it has already transferred, and begins to transfer these changes when all of the initial pages have been transferred.

III. PROPOSED HETEROGENEOUS LIVE MIGRATION

3.1 Live Migration

Live migration of virtual machine is possible now days due to the arrival of Virtual Machine Monitor (VMM). It provides the execution environment for operating systems running on same computer. Virtual machine from source computer is migrated to target host without halting. The memory transformation takes place in three phases.

Push Phase: The source virtual machine runs continuously on transferring the memory pages to a new machine. To maintain the consistency, the modified pages are sent again through the network.

Stop-and-copy phase: The running virtual machine is halted, and then the memory pages are transferred through network to the new virtual machine. Then the virtual machine is started in the new VMM.

Pull phase: When the pages that are not transferred from the source machine are accessed, it is called a fault. If a fault occurs, the VM sends a request for the missing page to the source machine.

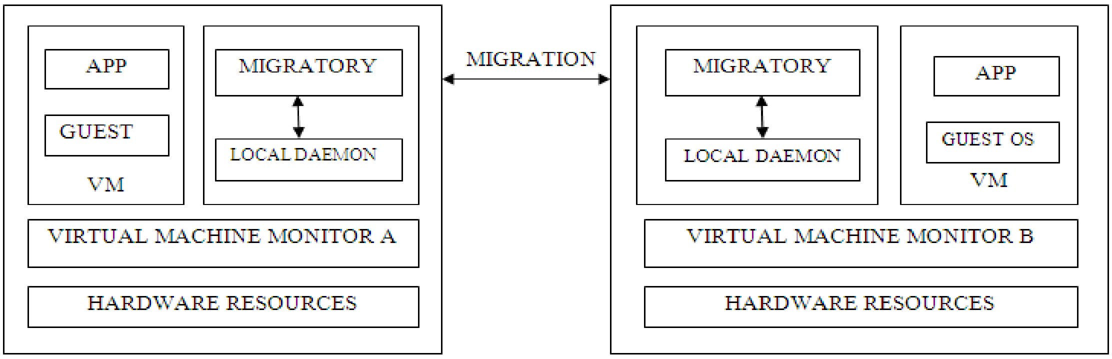


Fig 1: Heterogeneous Migration

3.1.1 Stages in Live Migration

Stage 0 Pre-migration: Alternate physical host is pre selected for migration.

Stage 1 Reservation: Initialize a container on the target host.

Stage 2 Iterative pre-copy: Shadow paging is used for copying the dirty pages in successive rounds.

Stage 3 Stop and Copy: The running VM on the source host is halted and an ARP is generated to redirect traffic to target host.

Stage 4 Commitment: Virtual machine on source host is released.

Stage 5 Activation: The virtual machine in the target host is resumed.

3.2 Heterogeneous Migration

Live migration is performed between two different hosts, the Xen and KVM. Each VMM provides its own abstraction to its underlying hardware resources. Many migration algorithms such as stop-and-copy, pre-copy, pull-copy are used and a framework is designed for performing migration. Migratory in the source host first transforms the own virtual machine into a common format. The virtual machine is transformed to the target host common format in the next step. The algorithm is used for changing the virtual machine to specific format that is supported on that machine and then the virtual machine is run on the target machine. Migration downtime is very small almost in the order of milliseconds.

3.2.1 Memory Migration Algorithm

In the proposed model, algorithm pool is used which supports various memory migration algorithms. The precondition is that at least one memory migration algorithm is common for both source and target hosts. If multiple users are used, the algorithm is chosen randomly or by the user.

3.2.2 Migration Protocol

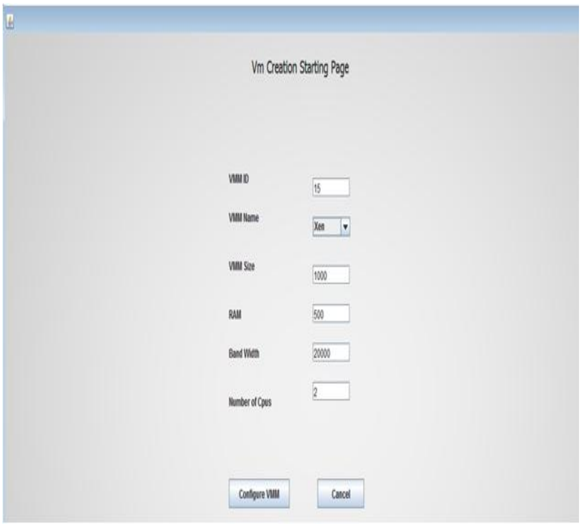
Protocol is mainly used to control the live migration process. Source and destination VMMs can establish a connection using the port numbers. After establishing the connection, commands are sent to the destination by source host. Commonly used control commands are shown below,

TABLE 1 CONTROL COMMANDS

Usage	Source	Destination
Request	MIG_REQ	REQ_OK, REQ_FAIL
Send configuration	MIG_CFG	CFG_OK,CFG_FAIL
Begin pre-copy	MEM_BEG	No response
Send CPU state	CPU_BEG	No response
Send I/O devices	DEV_BEG	No response

IV. EXPERIMENTS AND RESULTS

Cloudsim tool is used for live migration in heterogeneous migration. The major issues in cloudsim are provisioning of hosts to VMs, managing application execution, and monitoring dynamic system state. It has three layers, the bottom most layer is cloudsim core simulation engine, the middle layer is cloudsim and the top most layer is user code. In the user code, scheduling policy is used for scheduling tasks. The cloudsim layer has user interface structure, VM services, cloud services, cloud resources and network. The virtual machine with the necessary VMM size, RAM and bandwidth is created. It is used for configuring the virtual machine in the hypervisor. Many virtual machines can be created using unique ID.



Vm Creation Starting Page

VMM ID: 15

VMM Name: Xen ▾

VMM Size: 1000

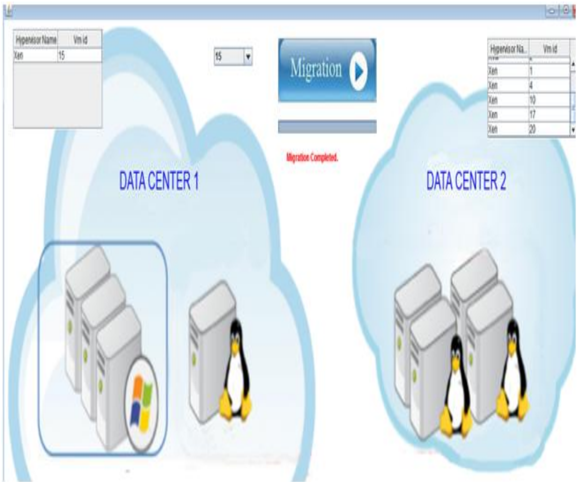
RAM: 500

Band Width: 20000

Number of Cpus: 2

Fig 2: VM creation

VMM type, VMM size, bandwidth are allocated for the newly created virtual machine. After providing the parameters, configure VM shown in Fig. 2 is used to deploy the new virtual machine to VMM. For performing migration, select the virtual machine id which is needed to be migrated. The virtual machine is migrated from datacenter1 to datacenter2. In Fig. 4 the newly created virtual machine is migrated to datacenter2 and is no more present in datacenter1. RAM utilization is shown in Fig 5.



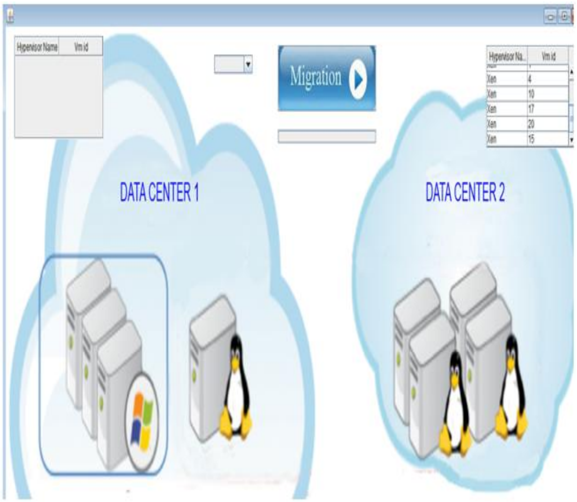
Hypervisor Name	Vm id
Xen	15

15 ▾

Hypervisor Na...	Vm id
Xen	1
Xen	4
Xen	10
Xen	17
Xen	20

Migration Completed.

Fig 3: Before performing migration



Hypervisor Name	Vm id

Hypervisor Na...	Vm id
Xen	4
Xen	10
Xen	17
Xen	20
Xen	15

Fig 4: After performing migration

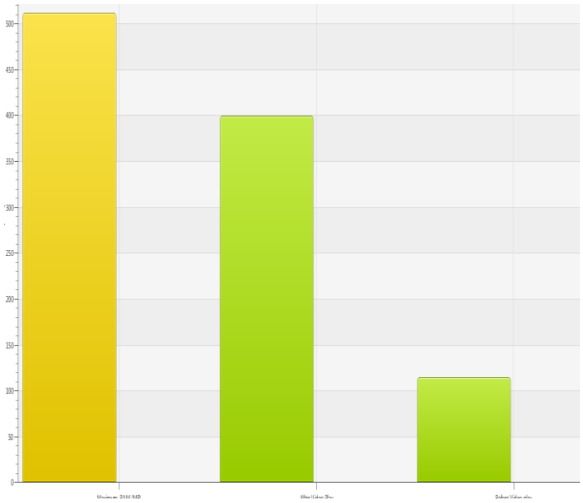


Fig 5: Utilization of RAM

V. CONCLUSION & FUTURE WORK

Live Migration, the moving of virtual machine from one physical machine to another physical machine without halting helps in decreasing downtime, almost to zero. Live migration in heterogeneous environment is also applicable for different virtual machine migrations. Performance measurements indicate that the downtime and end-to-end time in heterogeneous environments are comparable to homogeneous live virtual machine migration. In future work, plan to extended migration on other virtual machines, such as virtual boxes and VMware and also planned to add some Qos parameter to make migration more flexible and robust.

REFERENCES

[1] C. Clark, K. Fraser, S. Hand, J. Hansen, E. Jul, C. Limpach, and A. Pratt, I and Warfield, "Live migration of virtual machines", In Proc. NSDI, 2005.

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

[3] Eric Kellery, Soudeh Ghorbani, Matt Caesar, Jennifer Rexford, "Live migration of an entire network", In Proc. of 6 th ACM Hotnets'12 workshop. 2012.

[4] Wenjin Hu, Andrew Hicks, Long Zhang, Eli M. Dow, Vinay Soni, Hao Jiang, Ronny Bull, Jeanna N. Matthews, "A Quantitative study of Virtual machine live migration", In Proceedings of ACM CAC'13 workshop. 2013.

[5] Y. Wang, E. Keller, B. Biskeborn, J. van der Merwe, and J. Rexford, "Virtual routers on the move: Live router migration as a network-management primitive", In ACM SIGCOMM, 2008.

[6] Zhou, R., Liu, F., Li, T.2013, "Optimizing Virtual Machine Live Storage Migration in Heterogeneous Storage Environment", VEE'13 Proceedings of the 9th ACM SIGPLAN/SIGOPS International Conference on Virtual Execution Environments, 2013, pp. 73-84.

[7] Michael L. Powell and Barton P. Miller, "Process migration in DEMOS/MP", In Proceedings of the ninth ACM Symposium on Operating System Principles, pages 110.119. ACM Press, 1983.

[8] S. Osman, D. Subhraveti, G. Su, and J. Nieh. "The design and implementation of zap: A system for migrating computing environments", In Proc. 5th USENIX Symposium on Operating Systems Design and Implementation (OSDI-02), pages 361.376, December 2002.