ISSN: 0976-3104
SPECIAL ISSUE
Kumar et al.

THE IIOAB JOURNAL

ARTICLE OPEN ACCESS

SYSTEM PARAMETER BASED APPROACHES FOR VIRTUAL MACHINE MIGRATION AND DYNAMIC LOAD BALANCING USING OPEN SOURCE XEN VMM

Prakash kumar[1*], Krishna Gopal[1], JP Gupta[2]

[1]*Department of Computer Science Engineering & IT, JIIT, Noida, INDIA*
[2]*Hydrocarbons Education and Research Society, New Delhi, INDIA*

ABSTRACT

Virtualization Technology (VT), re-invented to address most of the computer systems resource utilization challenges especially for Cloud Environment. An important feature of VT is live migration of the Virtual Machine (VM) that consists of Guest Operating Systems and applications running on it.VM migration optimizes the system performance by dynamically balancing the load. This paper proposes two-fold techniques for optimizing system performance: First, A Trigger based VM Migration technique that gets activated when CPU temperature increases beyond an upper threshold value, called Hotspot. Temperature increases due to high computational loads on the physical machine running multiple VMs. Based on the Hotspot threshold, a VM can be live migrated to another best threshold based identified physical machine available. Second, a Network File System (NFS) based dynamic load balancing strategy is proposed for better system resource utilization. This is achieved by selecting the most suitable VM for load allocation.

Received on: 18th-June-2015
Revised on: 20th-July-2015
Accepted on: 03rd– Aug-2015
Published on: 05th –Jan-2016

KEY WORDS

System performance; Virtual Machine Monitor; Hotspot; Coldspot; Migration; Load Balancing; Network File System; Cloud Computing

***Corresponding author: Email:** kprakash91@yahoo.com **Tel:** +91-9810292083

INTRODUCTION

The re-invented Virtualization Technology consists of the system software layer known as the Virtual Machine Monitor (VMM) which controls and facilitates the creation, adoption, implementation and running of separate instances that consists of multiple emulated and separate environments named as Virtual Machines (VMs) on the same underlying physical hardware. This VM has emerged as a need for today's datacenter blocks and system clusters. VMs facilitate the datacenter to handle multi-tenant characteristics in a very reliable, secure, flexible and efficient way, which is the need of exponentially emerging Cloud Paradigm.

The principal characteristic feature of virtualization enabled architecture is its dual state hardware- Privileged and Non-privileged access. The former makes all the instructions available to the user, whereas the latter has to make supervisory calls to the operating system nucleus, in order to have privileged access. The main characteristics of a VM is that it runs and uses only the resources allocated to it and does not go beyond that. Virtualization of the instances of operating systems are highly useful as this feature facilitates the datacenter managers to provide isolated software and hardware environments to balance the user loads in a secure, reliable and fault tolerant way. Multiple VMs own the portion of the underlying hardware resources and each VM run their own separate operating systems which are handled and managed by the Virtual Machine Monitor (VMM). Many privileged and critical instructions are executed by these VMMs on behalf of the VMs running on it [1, 2].

A hypervisor actively encapsulates each and all volatile activities and requests of a VM. So, virtualization architecture can essentially be visualized as a sandbox with various virtual environments each with user defined attributes. One of such attribute is the operating system a VM is carrying called "Guest OS". Similarly, the OS of the system over which para-virtualization is done, is called "Host OS". A VMM maps each of the VMs with a separate file onto its local file system. Every change is reflected in the file on local file system. Such file is generally an image file of the VM. If such a file is carefully copied from a specified path to a similar path of

THE IIOAB JOURNAL
ISSN: 0976-3104

another VMM with similar hardware architecture, then the new hardware can resume the VM running on previous hardware. This process of shifting a VM from one hardware to another is termed as VM migration [1].

The need to migrate the VMs arises because of the overloading of one physical machine that runs multiple VMs onto it. Consequently, there is overheating which may degrade the performance and may also lead to faulty operations and system crashes. Hence a conditional load balancing is required for better manageability of the cluster of servers [2].

There are few advantages of migration of a whole of VM. The narrow interface between a virtualized OS and the virtual machine monitor (VMM) avoids the problem of residual dependencies. There are various ways to migrate a VM from one physical node to another. "Pure stop-and-copy" or "Cold" migration technique halts the VM and copies all its associated memory pages to another pre-identified destination node and then resumes the VM on it. On the contrary, few selected hypervisors like Xen and VMWare do a "Live" or "Hot" migration. The advantage of the latter method is that even the concerned applications and processes are unaware of the VM migration [3].

A new technique to trigger VM migration or any other desired causal effects of temperature and CPU usage variation on VMM Xen 4.2 as a whole and VMs with host OS Ubuntu 13.04 are discussed in following sections.

BACKGROUND AND RELATED WORK

The cloud services providers while focusing on amazing user experience of their services also stress on the need of optimized usage of their resources and data durability of their users. So they developed various algorithms and implementation to migrate a VM in various cases like excessive CPU requirement, memory constraints, Stagnant/idle, etc.

There are few techniques to resolve such issues which include VM migration and Process Migration. On Xen hypervisor, the VM migration feature can easily be run and effects analysed with xm-migrate command. And this command has been highly used and emphasized on since it can be modified with variety of attributes that goes with the command such as whether we want the migration to be live or cold. But the real problem arises on detecting when to fire the VM migration mechanism and detecting which VM is the causing the trouble.

Process migration demonstrates a functionality of transferring a process running on one machine to the other. But, there is an inherent difference in the operating concepts of virtual machine migration and process migration [4]. Though in practice, migrating the process is difficult and quite complex as it should take care of legacy applications and at the same time it should also leverage the currently installed and related large databases of operating systems and maintain independence on different machines. These can be overcome by using a VMM based migration. VMMs such as VMware use Hardware abstraction to encapsulate the complete OS environment in such way that it can be suspended from one machine and resumed onto the other one provided there are inherent similarities in the system architectures of the operating systems.

But VM migration supersedes Process Migration except in some cases that occur due to the narrow interface between a virtualized OS and the virtual machine monitor (VMM) where itavoids the problem of residual dependencies. VM migration has the advantage of transferring internal memory states in a very consistent and efficient way [4].

Another part is that the System Virtual Machines (VMs) [5, 6] are widely used from personnel computers to large organizations. System virtualization acts as powerful means of abstraction for upcoming applications. On the cloud computing platform resources are provided according to need on the principle of pay per use. To accommodate specific requirements of subscribers and how the balance is maintained between Cost, Quality and Resources is mentioned in the Service Level Agreement (SLA). The Cloud Service Providers guarantee a level and quality of service to the users as per the terms and conditions of the SLAs. A lot of challenges are faced while catering the need of the users and at the same time making efficient use of underlying heterogeneous resources in a dynamic and efficient way, which is inherently expected from Cloud Services in terms of Infrastructures, platforms and software.

By mapping the services onto the Virtual Machines (VMs), where multiple VMs can run onto a single physical server, the problems related to heterogeneity in hardware, software and platforms could easily be solved.

THE IIOAB JOURNAL
ISSN: 0976-3104

Consequently the terms and conditions for the SLAs can be agreed upon between the Cloud Service Providers and the Service Users.

VM Load Balancing is crucial characteristics of system virtualization, allocate and shift the running applications dynamically to other physical machines as and when the load increases on any particular machine [7]. Because of the complexity aroused due to the vast heterogeneity in terms of underlying hardware, operating systems environment, platforms and communication technologies, and that too at the run time, it is inevitably important to address the performance and issues related to smooth delivery of services to the end users. Consequently, addressing the resource allocations and related issues, viz. VM Load balancing, VM Scheduling techniques, VM migration [8], VM performance optimization and cross platform operations issues are the need of the hour for Cloud environments, and that too with a guaranteed level of services [9].

This paper addresses two-fold techniques for optimizing system performance: First, A Trigger based VM Migration technique that gets activated when CPU temperature increases beyond an upper threshold value, called Hotspot. Temperature increases due to high computational loads on the physical machine running multiple VMs. Based on the Hotspot threshold, a VM can be live migrated to another best threshold based identified physical machine available. Second, a Network File System (NFS) based dynamic load balancing strategy is proposed for better system resource utilization. This is achieved by creating performance models for VM load balancing. Many experiments were conducted on Xen for Virtual Machines [10] for Network File Systems. Load balancing is done by scheduling the VMs on a particular physical machine that is comparatively having less load. This method computes the load on various virtual machines and then finds the virtual machine which is most suitable for the upcoming load.

Load balancing is the capability of the system which allows the VM hosted applications to be transparently allocated a VM which has the least load dynamically so as the maximum resource utilization of the whole system is achieved.

There have been many approaches to load balancing viz. Static and dynamic. In addition, some hybrid approaches are also adopted. Major difference being in Dynamic Load Balancing, decision is taken at runtime according to the existing situations, whereas in static it is not. Neither of them is superior or inferior but the selection of algorithm depends on the application requirements.

Static load balancing

Static Load Balancing (SLB) refers to the load balancing algorithm that distributes the load strictly on the basis of certain predefined rules relating to the nature of input loads. It does not consider which node is receiving more or less load. In all static algorithms final selection of the virtual machine is done immediately after creation of application. Further it cannot be changed while in execution. These static load balancing techniques are suitable for a system in which load is limited and request of the clients is also limited. But nowadays load on cloud servers is also increasing and also the load is not static hence we need more efficient algorithms then static load balancing algorithms.

Subsequently we describe some of the basic algorithms for static load balancing as follows:

Round Robin Algorithms

Whenever a new application comes it is assigned a virtual machine in a round robin fashion. In general, basic idea for Round Robin [11] is to reduce message passing between various virtual machines and reduce communication delay. Thus it is independent of the state of the system. When coming applications are of similar load then Round Robin works very well as it reduces the communication delay due to inter-process communication. Thus Round Robin has best performance for this special purpose application of similar load, but does not give a good performance for general cases.

Randomised Algorithm

Random numbers are distributed on a basis of a statistical distribution and assigned to virtual machines. Incoming applications are distributed according to these randomly generated numbers. This algorithm is applicable when we have many virtual machines as compared to processes

THE IIOAB JOURNAL
ISSN: 0976-3104

Central Manager Algorithm

In this algorithm (Huang, 2012), there is a central virtual machine and others are slave virtual machines which are assigned applications to be executed. Central virtual machine's task is to gather load information of all the slave virtual machines and assign the coming application to the least loaded slave virtual machine as shown in **Figure-1**.

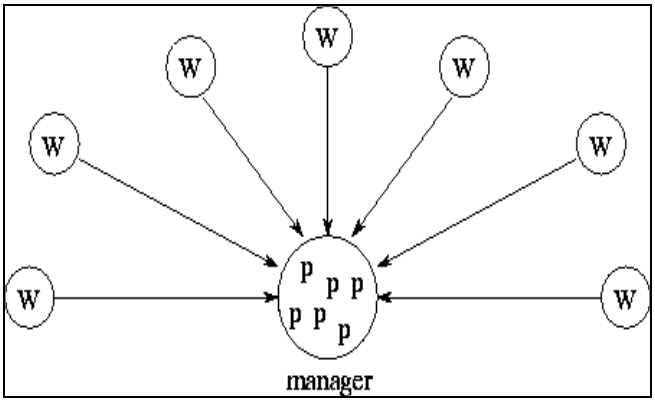

Fig: 1. Manager collecting information of slaves

Threshold Algorithm

In this algorithm the current virtual machine is decided on the basis of two values of upper (t_upper) and lower (t_lower) threshold. Virtual machine is assigned a state depending on its current load compared to these threshold values. If current load is less than the t_lower virtual machine is assigned a Under Loaded State, if its greater than t_upper the state is overloaded, if its between the two threshold values then the state is Medium.

Initially all virtual machines have under loaded state. But as the system advances the load level limit of a virtual machine may change and its state may change. If the state changes then it send message to all other virtual machines notifying the change, so that they can maintain the load state of entire system. When a process arrives and the local virtual machine is under loaded then it executes the application else calls for the remote virtual machine. If no under loaded virtual machine exists then application is executed locally only.

Dynamic load balancing

Dynamic Load Balancing (DLB) techniques provide a method to dynamically allocate load based on self-adapting distribution and intelligent distribution. Here, it is distributed at runtime based on the new information collected. Mainly these techniques are based on greedy algorithmic approaches. Basic algorithms for dynamic load balancing are:

Central Queue Algorithm

The host virtual machine maintains a central queue of all the applications. This queue is shared by all the processes. New applications are added and pending applications are maintained in a cyclic FIFO order in the queue. When a virtual machine is free it will request for application for executing to the host and host assigns the application next in queue to the virtual machine which is demanding the request. If there is no application for execution in the queue then the request is buffered in queue form. And request is answered when new application arrives.

Load Queue Algorithm

Here the applications are assigned virtual machines similar to the static algorithm but a virtual machine here can initiate application migration. Initially all under loaded virtual machines are assigned the applications. The applications are assigned following some static algorithm. Then if a virtual machine's load goes below the lower threshold value then it asks for load from other machines and initiates migration process.

THE IIOAB JOURNAL
ISSN: 0976-3104

A lot of scheduling algorithms have been re-invented and tested for cloud environment, namely intelligent scheduling algorithms, autonomous scheduling algorithms, agent-based negotiable scheduling algorithm, centralized scheduling algorithms. These algorithms are used by many popular Cloud Service Providers for balancing the loads on their virtual machines; a popular example could be VM Ware Distributed Resource Scheduler (VMDRS).

Analysing all these algorithms our paper presents a balanced techniques for balancing load on Xen Hypervisor using the technique of Network File System for file Sharing.

Our Contribution

The technique of Virtual Machine migration has been very helpful in maintaining VMs in a way more resource optimized but the cause for migration is very significant that we have been ignoring for a long time. In This paper as already discussed, focus will be on the importance of various trigger options that initiate the migration procedure. It has been focused on the thermal and CPU usage of the data server as the trigger for VM migration. This leads to a scenario where the user can choose what to do in case of various faults. The `"Cold Spot", "Hotspot" as thermal issue of data server hardware have been looked into. Similarly, "Overload" and "Under load" are the issues alarming CPU usage issues. Shell scripting was used as a tool for determining the boundary conditions for each case and then the triggering part comes into play. Python socket connection was also used so as to facilitate the communication between two PCs acting as data servers independently.

The connection so set up will ask the data server 2 to get ready to receive an image file. On data server 1, an image of VM to be migrated will be selected from the Xen VM image repository and sharing the file with data server 2 over the network. The server will now install this image over itself.

The VM migration in general can be classified into three phases:

1. Recognizing trigger: This phase implies the detection of a trigger that may cause the VM to crash or harm the memory space. This detection can be based on the various parameters like CPU utilization, Process throughputs etc.
2. Image packing: Creating the image file of the memory space and packing the image with headers.
3. File Sharing: Sharing this image file over the hosts/data server network using Network File Sharing.
4. VM replication: Extracting the VM over the data server from the image.
5. Stability test of newly formed VM
6. Instant transfer of workload to new VM.
7. Deletion of old existing VM.

A hotspot/ cold spot is an undesirable hardware temperature fluctuation which generally occurs when data centre is improperly cooled.Hotspot is dangerous problem since it can cause serious trouble to the hardware. A typical data server contain significant amount of power consuming components this producing so much heat like a furnace if not properly cooled.Cold spot on the other hand occurs when the equipments installed in a data server receive too much of the cooling thus posing the moisture trouble and causing disruptions in electronic circuits.

The ASHRAE (American Society for Heating, Refrigerating and Air-Conditioning Engineers) standard cold spot for a typical data server is 64.4 degree Fahrenheit or 18 degree Celsius. And hotspot depends on the quality of hardware use but generally 85+ degrees Celsius is considered highly critical.

Functional Requirements

Functional requirements define the fundamental actions that must take place in the software in accepting and processing the inputs and in processing and generating the outputs as shown in **Figure-2**.

Table: 1. Compatible Paring (source: Linux Foundation, 2014)

Ubuntu Version	Xen Release
Ubuntu 12.10	Xen 4.1
Ubuntu 13.04	Xen 4.2

Compatibilities Issues:

1. Enough Memory space so as to create an image of data to migrated.
2. Smooth Internet connection to eliminate any data losses or interruption
3. All the data servers must be in the same subnet network.
4. Python 2.7+ packages must be installed.

THE IIOAB JOURNAL
ISSN: 0976-3104

5. Libvirt and libvirsh must configured
6.Live migration settings must be configured.

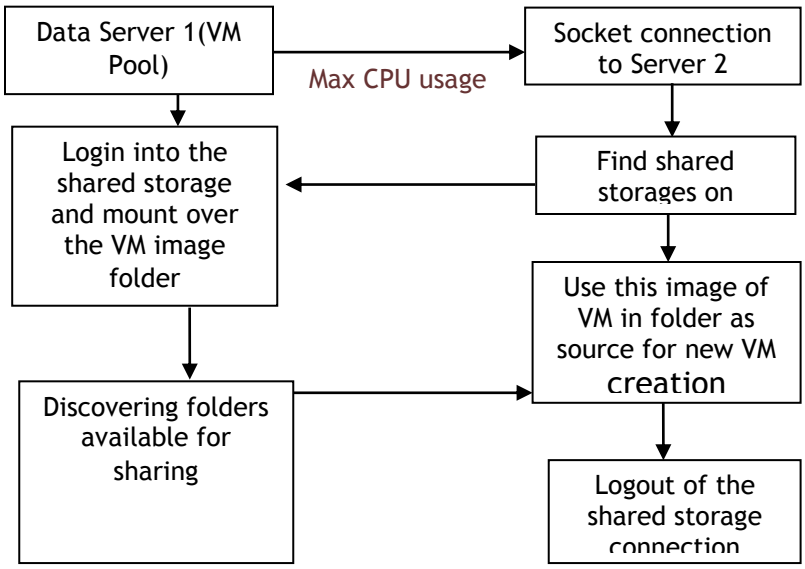

Fig: 2: Steps of Execution

Non-Functional Requirements

1. *Error handling*
   - Expected and non-expected errors shall be handled in such a way that it prevents loss of information with minimum downtime.

2. *Performance Requirements*
   - Depends on network connection.
   - Processor cycles to execute commands.

3. *Security Requirements*
   - Security of the data involved in migration depends on the network security .
   - Do not apply network security on a private network because it impacts the performance a lot. However, security measures should be implemented when the migration traffic needs to be encrypted

4. *Reliability*
   - The procedure is reliable as long as the data selected is accessible and the migration is nt interrupted.

5. *Correctness*
   - The procedure implemented in the system should be correct which means that they should be performed as required. The testing Phase insures correctness of the software by trying all possible Case and matching their output with the documentation.

IMPLEMENTATION OF TRIGGER BASED MIGRATION

Virtualization can be implemented using any of the readily available hypervisors but choosing the right one and feasible and good support is important. Here open source VMM namely Xen 4.2 over Ubuntu 13.04 is chosen as this is highly in use in most of the commercial and open source freeware VMMs. Xen is a very popular hypervisor among all cloud communities. It has full compatibility and support by x86, IA-32, ITANIUM, VT_x, VT_i and most of the PIC, ARM architectures. It is supported by many operating systems like Solaris, Windows, Linux, etc. as guest operating systems on their CPU architectures. Xen can do full virtualization on systems that support virtualization extensions, but can also work as a hypervisor on machines that don't have the virtualization extensions. Citrix has collaboratively launched a Xen Citrix server.

THE IIOAB JOURNAL
ISSN: 0976-3104

Performing the objective so discussed is crucially dependent on the installation and configuration of the VMM used.

Xen virtualization is ready to deploy but now a trigger is needed to initiate the VM migration. The host Os doesn't know about the Virtual machines running over it except it knows the existence of VMM over it and it is same for VMs as well, they know Xen only not the presence of any host Os. So for host Os to calculate any over loading CPU usage on account of virtual machines, it has to be the xen process on host Os that should be monitored. Hence, the following command will help in retrieving all processes responsible for CPU cycle usage and can be monitored accordingly.

>> top -bn1 | grep "Cpu" | sed "s/., *\([0-9.]*\)%* id.*/\1/"*

Above command can be used to write a shell script that could calculate cumulative CPU usage and hence deal with "Over loading" and "Under use "in order to migrate VM thus optimizing resource use. One shouldn't let a data server host VMs that could be accommodated on other data server thus reducing the expense on a data server. In case of Cold spot/ Hotspot, one needs to continuously monitor the thermals of the hardware involved, thus,
>> install acpi (to handle power and thermal related issues and logs one needs to install this on the machine first). Then only can one monitor using the command:
*>>acpi –t| sed "s/.*ok,\([0-9]*)*\).*/\1/"*(Command for monitoring CPU temperature)

1. *sudo vim /etc/default/iscsitarget:* (To allow discovery of target machine change the value to true.It normally resides in the default directory of iscsi).
2. *Sudo vim /etc/iet/ietd.conf*(In this set the name, path and type of the iscsi storage).
3. *Sudo /etc/init.d/iscsitarget restart* (To make changes take into effect).
4. *Cat /proc/net/iet/session* (Session is used to start the session used to allow the discovery of the storage to be available to the client system).
5. *Iscsiadm –mode discovery –type sendtargets –portal ipaddress*(To discover the list of available storage device on which can be target to migrate our machine).
6. *Iscsiadm –m node –t NameOfTarget –p ipaddress login*(Login to the storage device. In case it is secured then requie password otherwise it will login automatically).(Commands for migration)
7. *Dmesg*(To detach from iscsitarget)
8. *Iscsiadm –m session –logout*(To logout from the target it is required to log out from the target from all the address).

The above procedure should result in an image of VM to be migrated in the destination data server.

VM SELECTION FOR MIGRATION

The tricky part is deciding to choose which VM must be migrated from one data server to another in case a trigger has been fired and VM migration is inevitable. In such critical conditions, it is logical and feasible to propose that such a dilemma can be tackled by listing every VM that particular hardware hosts and then running a script to find the combination of VMs whose CPU usage can add up together to maximise CPU usage thus avoiding CPU overloading, In addition, this also helps in deciding as to which VM should be migrated. Above discussed method can be compared to best fit allotment analogy. However, this algorithm can only be applied if the trigger was CPU overload. The algorithm and the environment is still in studying phase to improve and broaden the parameters to design a better algorithm. The ***"Xm"*** command set's ***"Xmtop"*** instruction is quite useful in the algorithm mentioned.

To understand the algorithm in a better way, an example is shown: Assume there are three VMs with CPU usage 25%, 25%, 40%, thus leading the machine to CPU overload. This scenario will cause the machine to fire the trigger and hence migration. Now comes the part when it decides which VM to migrate. Here the maximised combination of CPU usage is 40% and 25% together, if add the other 25% to it, it'll shoot over 85%, which is theupper threshold decided by the standards. Hence, either of the VMs with 25% CPU usage can be migrated to stabilize this host.

THE IIOAB JOURNAL
ISSN: 0976-3104

Setup for VM Load Balancing

VM Model

Initially, the physical machines are mapped to the VMs as shown in fig :3: Three physical machines have been taken; where two VMs are installed on two of them and three VMs on one of them i.e. in total, there are three physical machines and seven VMs. To generalise the abovementioned, there are N physical machines and M Virtual Machines. Then the active Set of existing machines can be given as, PM= $\{PM_1,PM_2,\ldots,PM_N\}$, where, $PM_i(1<=i<=N)$ denotes the No.i for the physical machines. Physical Machine PM_i has m_i VMs on it represented as $V_i= \{V_{i1},V_{i2},\ldots,V_{imi}\}$ and $m_1+m_2+\ldots.+m_N= M$

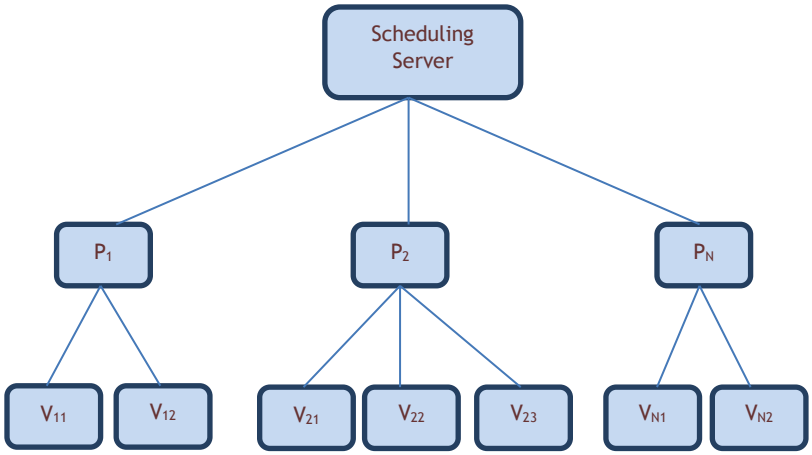

Fig: 3: System Structure

Expression of Load

The load on the physical machine is calculated by adding the total loads of all the Virtual Machines (VMs) running on this physical machine. The load of VMs is measured after a particular period of time say j, where j is the time period between $(t_j – t_{j-1})$. Assuming that the load of VMs is relatively constant in every time period, load of the VM No.i can be defined as V(i, j) for the time period p. Supposed that there are n VMs whose loads needs to be measured in the time period k. Then the loads of each one can be measured accordingly e.g. load of VM_1 at (1*p/n), VM_2 at (2*p/n) and so on. Thus, the load value of a particular VM gets updated once in each time period of duration k.

Therefore, it can be concluded that in cycle T, the average load of VM VM_i on physical machine PM_i [31] can be given as

$$\overline{VMi(i,T)} = \frac{1}{T}\sum_{j=1}^{n} VM(i,j)\times(t_j - t_{j-1})$$

Obviously, the total load of any physical machine is the sum of all running VM loads. Hence, load of physical machine PM_i is given by:

$$PM(i,T) = \sum_{j=1}^{mi} \overline{VMi(j,T)}$$

IMPLEMENTATION

For implementing a load balancer on Xen the specification of the system used were:

- 64bit x 86 computers with 3GB of RAM, 320GB of storage space.

THE IIOAB JOURNAL
ISSN: 0976-3104

- Processor with VT_D , VT_X enabled
- Ubuntu 12.04 as Operating System

Initially Ubuntu12.04 was installed as host operating system. A hypervisor was created using Xen project, which enabled executions of multiple guest operating system simultaneously on a single physical machine. In particular there are two types of hypervisors, these are type1(native or bare metal) and type2(hosted).Our project is using bare metal hypervisor meaning the hypervisor layer is created directly on the host hardware, which allows the hypervisor to control the hardware. There is a concept of domains in Xen Hypervisor. There are two type of domains, domo-which controls the functioning of the hypervisor and starting the operating guest operating system.. For our work we have chosen Ubuntu 12.04 as dom0 machine. other guest operating systems are called domUs, this is because these domains are "unprivileged" in the sense they cannot control the hypervisor or start/stop other domains For our work we have created 3 machines as domUs in two of them Ubuntu 12.04 is installed and in one of the DSL in installed. For the purpose of communication between different domains, and also intra domain communication, a NFS is configured. NFS is a distributed file system protocol which allows sharing and transfer of files across various nodes on a particular network so to share files across domains we have to use NFS.

Algorithm

This algorithm repeats itself after an interval of time 'k'.

1) After time k the dom0 as updated load status of each VM
2) Dom0 machine will read the load
3) A:= load of vm1
4) B:=load of vm2
5) C:=load of vm3
6) If(A<B && A<C)
   then
      allocate task to vm1
   else if(B<A && B<C)
   then
      allocate task to vm2
   else
      allocate task to vm3
   end if
   end if

ANALYSIS

Our experimental work mainly analyzes the effect of implemented load balancing strategy and compares this method with the performance of a system without load balancing. Here we draw a comparison between three kinds of systems. Firstly, a system running on a single operating system. Second, a system in which Xen hypervisor is implemented without using the technique of load balancing thirdly, a system with Xen hypervisor installed and also load balancing techniques implemented.
Figure 4 shows the measured comparison of response time when number of request increases. The improvement done by our work is clearly visible through this comparison.

Figure 5 shows that when there is only one operating system installed then the performance of system decreases when load increases, while if a hypervisor is installed then initially the performance is lower than native performance due to certain overheads, but as load increases performance becomes better. Performance is best when a load balancer is also implemented. In these plots, Y-axis depicts the performance of the system whereas X-axis depicts the number of tasks.

THE IIOAB JOURNAL
ISSN: 0976-3104

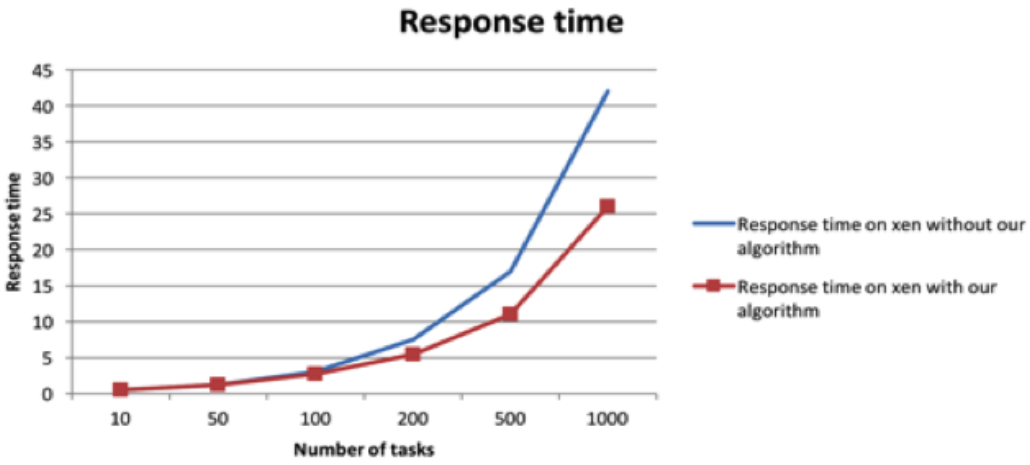

Fig: 4. Comparison of response times

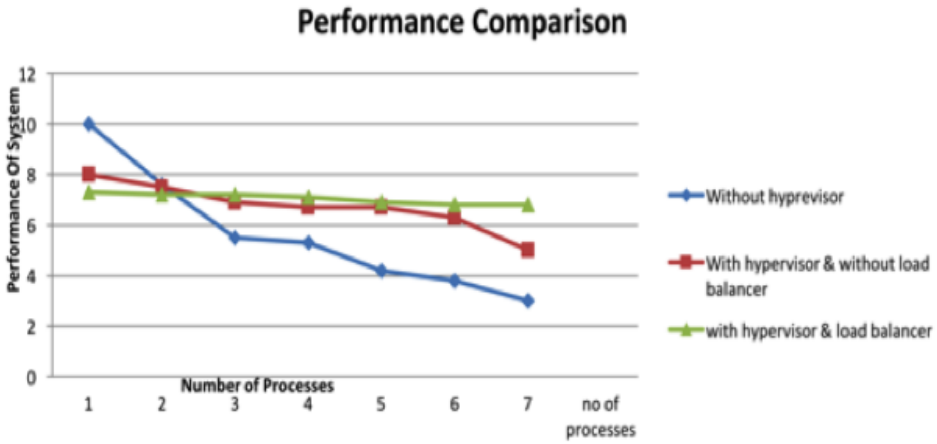

Fig: 5. Comparison of System performance (CPU usage) with and without load balancer in systems with and without hypervisor

CONCLUSION

The migration of a VM is a tricky and complex process. It may lead to loss of VM or corruption of the same if not handled properly. This innovative technique of trigger based VM migration will surely provide users to take more control over its VM and help them to better manage their VMs,consequently, a better and efficient way of managing the underlying system resources, hence bolstering the core idea of virtualization, which is the technical backbone of cloud computing.

A dynamic load balancing strategy and algorithm is developed and implemented on Xen VMM for VM load balancing based on Network File System (NFS). According to the current states of VMs, it computes and identifies in advance, the most suitable VM where the upcoming application can be allocated. The overhead involved for identifying suitable VMs for successive allocations is drastically reduced as the previous and current states are already available. Additionally, this strategy is quite efficient and requires less computational overhead as compared to normally used Migration strategies and other traditionally adopted load balancing techniques and hence better resource utilization comparatively.

THE IIOAB JOURNAL
ISSN: 0976-3104

CONFLICT OF INTEREST

Authors declare no conflict of interest.

ACKNOWLEDGEMENT

None.

FINANCIAL DISCLOSURE

No financial support was received to carry out this project.

REFERENCES

[1] Abels T, Dhawan P. [2005] An Overview of Xen Virtualization. *Dell Power Solutions.*

[2] Anderson AV, Benett S M, Kagi A, et al. [2005] Intel Virtualization Technology. *IEEE*. http://doi.ieeecomputersociety.org/10.1109/MC.2005.163.

[3] Shenoy P, Venkataramani A, Wood T, Yousif M. [2007]Black-box and Gray-box Strategies for Virtual Machine Migration, 4th USENIX Symposium on Networked Systems Design & Implementation, 229–242.

[4] Sandhu S,Venkatesh, S. [2009] Survey of VM migration Techniques. *IEEE.*

[5] Carl A, Waldspurger. [2002] Memory Resource Management in VMware ESX Server. Proceedings of the 5th Symposium on Operating Systems Design and Implementation , 181–194.

[6] Dragovic B, FraserD, Hand S, et al. [2003] Xen and The Art of Virtualization. Proceedings of the 19th ACM Symposium on Operating Systems Primciples. 164–177.

[7] Freeman T, Foster I, Keahey, K, Sotomayor, B. [2007] Enabling cost-effective resource leases with virtual machines. *International Symposium on High Performance Distributed Computing.*

[8] Clark C, Hand S. [2005] Live Migration of virtual machines. 2nd Symposium on Networked Systems Design & Implementation.

[9] Cherkasova L, Gupta D, Vahdat A. [2007] When virtual is harder than real: Resource allocation challenges in virtual machine based it environments. *Technical Report HP*, 1–10.

[10] Ballani H, Costa P, Karagiannis T, Rowstron A. [2011] Towards predictable datacenter networks. Proceedings of ACM SIGCOMM, 242–253.

[11] Huang R, Xu Z. [2012] Performance Study of Load Balancing Algorithms in Distributed Web Server Systems. CS213 Parallel and Distributed Processing Project Report, 3679-3683.

ABOUT AUTHORS

Prof. Prakash Kumar *has received his B. Tech. in Electronics and Communication Engineering, and M. Tech in Computer Science and Technology from University of Roorkee (Now Indian Institute of Technology, Roorkee), India. He is currently an Assistant Professor (Senior Grade) in Jaypee Institute of Information Technology,(Deemed University), Noida, India. His area of interest is in Computer Networks and Communications, Distributed Computing, Cloud Computing and Virtualization,. He is currently pursuing his Ph D in the field of virtualization of resources viz. Systems and network resources. His main focus is on Trust, Reliability and Fault Tolerant networks and systems for distributed and cloud environments.*

Prof. Krishna Gopal *is currently Dean Academic and Research at JIIT, Noida, India since 2011. He is Ph. D. from REC Kurukshetra, Kurukshetra, India. He is having 45 years of teaching and R&D experience. He received his Bachelor, Master and PhD in Electronics engineering from IIT, Madras, NIT Kurukshetra in 1966, 1972, 1979 respectively. He published more than 100 papers in different journals, conferences, patents etc. He is member of various professional bodies like: Life Member System Society of India, Indian Society for Technical Education, senior member of IEEE etc. His area of interest is Reliabilitv and Fault Tolerant Networks and communication Svstems.*

Prof. J P Gupta *has received his Ph D degree from University of Westminster, UK. He is currently the Director Emeritus (QA) at Hydrocarbons Education and Research Society, New Delhi, India. He is an academician having more than 35 years experience including Professor at IIT Roorkee, India, Vice Chancellor at JIIT Noida,India, Galgotia University, and Sharda University, India. He is author of more than 70 research papers published in International journals and conferences. He has received Commonwealth Fellowship and many more awards and memberships to his credit.*