ISSN: 0975-766X
CODEN: IJPTFI
Available Online through *Research Article*
www.ijptonline.com

PERFORMANCE MEASURE OF NETWORK I/O WORKLOAD IN VIRTUALIZED DATA CENTER USING HYPERVISOR

[1]M.Rajesh, [2]Dr .G.Singaravel.
[1]Research Scholar, Bharath Institute of Higher Education and Research
Bharath University, Chennai.
[2] Professor and Head, K.S.R college of Engineering (Autonomous), Tiruchengode.
Email: goldmraja@gmail.com

Received on: 15.10.2016 *Accepted on: 22.11.2016*

Abstract

Cloud computing is gaining popularity as it's the way to virtualize the datacenter and increase flexibility in the use of computation resources. This virtual machine approach can dramatically improve the efficiency, power utilization and availability of costly hardware resources, such as CPU and memory. Virtualization in datacenter had been done in the back end of Eucalyptus software and Front end was installed on another CPU. The operation of performance measurement had been done in network I/O applications environment of virtualized cloud. Then measurement was analyzed based on performance impact of co-locating applications in a virtualized cloud in terms of throughput and resource sharing effectiveness, including the impact of idle instances on applications that are running concurrently on the same physical host. This project proposes the virtualization technology which uses the hypervisor to install the Eucalyptus software in single physical machine for setting up a cloud computing environment. By using the hypervisor, the front end and back end of eucalyptus software will be installed in the same machine. The performance will be measured based on the interference in parallel processing of CPU and network intensive workloads by using the Xen Virtual Machine Monitors. The main motivation of this project is to provide the scalable virtualized datacenter.

Keywords: Cloud computing; Virtualization; Hypervisor; Eucalyptuin.

Introduction

Cloud Computing [1] is more than a collection of computer resources as it provides a mechanism to manage those resources. It is the delivery of computing as a service rather than a product, whereby shared resources, software, and information are provided to computers and other devices as a utility over a network. A Cloud Computing platform

supports redundant, self-recovering, highly scalable programming models that allow workloads to recover from many inevitable hardware/software failures. The concept of cloud computing and virtualization offers so many innovative opportunities that it is not surprising that there are new announcements every day.

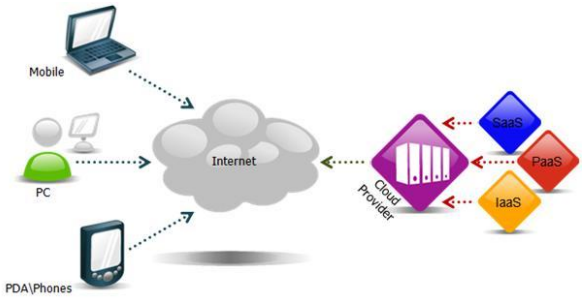


Fig. 1 Cloud Computing Architecture.

It is defined as a pool of virtualized computer resources. Based on this virtualization the Cloud Computing paradigm allows workloads to be deployed and scaled-out quickly through the rapid provisioning of virtual machines or physical machines. In a Cloud Computing platform software is migrating from the desktop into the "clouds" of the Internet, promising users anytime, anywhere access to their programs and data. The innovation will continue and there will be massive value created for customers over the coming years. Most technologists are sold on the fundamentals of cloud computing and it's a realization of the last 20 years of architecture development. It describes a new supplement, consumption, and delivery model for IT services based on Internet protocols, and it typically involves provisioning of dynamically scalable and often virtualized resources. It provides computation, software, data access, and storage services that do not require end-user knowledge of the physical location and configuration of the system that delivers the services. Cloud computing providers deliver applications via the internet, which are accessed from a web browser, while the business software and data are stored on servers at a remote location. In some cases, legacy applications (line of business applications that until now have been prevalent in thin client Windows computing) are delivered via a screen-sharing technology, while the computing resources are consolidated at a remote data center location; in other cases, entire business applications have been coded using web-based technologies such as AJAX. In this paper, performance interference among different VMs is running on the same hardware platform with the focus on network I/O processing. The main motivation for targeting our measurement study on performance interference of processing concurrent network I/O workloads in virtualized environment is simply because network I/O applications are becoming dominating workloads in current cloud computing systems. By carefully design of our measurement study and the set of performance

metrics we use to characterize the network I/O workloads, we derive some important factors of I/O performance conflicts based on application throughput interference and net I/O interference. Our performance measurement and workload analysis also provide some insights on performance optimizations for CPU scheduler and I/O channel and efficiency management of workload and VM configurations.

Virtualization

Virtualization [9],[6],[5] in computing is a process of creating virtual (rather than actual) version of something, such as a hardware platform, operating system, a storage device or network resources. It is a software acts like hardware. The software used for virtualization is known as hypervisors. There are different types of hypervisors which are used for virtualization like, XEN, VMware, and KVM.

Xen Hypervisor

Xen is a Virtual-Machine Monitor (VMM) providing services that allow multiple computer operating systems to execute on the same computer hardware concurrently. The Xen community develops and maintains Xen as free software, licensed under the GNU General Public License (GPLv2). It is available for the IA-32, x86-64, Itanium and ARM computer architectures.

VM Ware

VMware is proprietary software developed in 1998 and based in Palo Alto, California, USA. It is majorly owned by Corporation. VMware's desktop software runs on Microsoft Windows, Linux, and Mac OS X, while VMware's enterprise software hypervisors runs for servers, VMware ESX and VMware ESXi, are bare-metal embedded hypervisors that run directly on server hardware without requiring an additional underlying operating system.

Kernel-based Virtual Machine (KVM)

KVM provides infrastructure virtualization for the Linux kernel. KVM supports native virtualization on processors with hardware virtualization extensions. This supports a paravirtual Ethernet card, a paravirtual disk I/O controller, a balloon device for adjusting guest memory-usage, and VGA graphics interface using SPICE or VMware drivers.

Paravirtualization

Paravirtualization is a virtualization technique that provides a software interface to virtual machines that is similar but not identical to that of the underlying hardware. The intent of the modified interface is to reduce the portion of the guest's

execution time spent for performing operations which are substantially more difficult to run in a virtual environment compared to a non-virtualized environment. The paravirtualization provides specially defined 'hooks' to allow the guest(s) and host to request and acknowledge these tasks, which would otherwise be executed in the virtual domain. A successful paravirtualized platform may allow the Virtual Machine Manager (VMM) to be simpler and reduce the overall performance degradation of machine-execution inside the virtual-guest.

Credit-Based CPU Scheduler

Introduction

The credit scheduler[4] is a proportional fair share CPU scheduler built from the ground up to be work conserving on SMP hosts. It is now the default scheduler in the Xen-unstable trunk. The SEDF and BVT schedulers are still optionally available but the plan of record is for them to be phased out and eventually removed.

Description

Each domain (including Host OS) is assigned a weight and a cap.

Weight: A domain with a weight of 512 will get twice as much CPU as a domain with a weight of 256 on a contended host. Legal weights range from 1 to 65535 and the default is 256.

Cap: The cap optionally fixes the maximum amount of CPU a domain will be able to consume, even if the host system has idle CPU cycles. The cap is expressed in percentage of one physical CPU: 100 is 1 physical CPU, 50 is half a CPU, 400 is 4 CPUs, etc... The default, 0, means there is no upper cap.

Smp Load Balancing

The credit scheduler performs automatic load balances among guest VCPUs across all available physical CPUs on an SMP host. The administrator does not need to manually pin VCPUs to load balance the system.

Algorithm

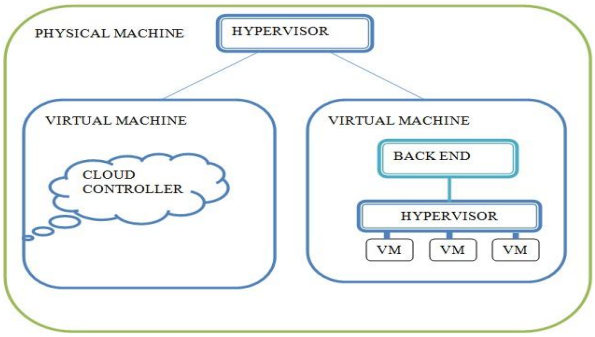
Each CPU manages a local run queue of runnable VCPUs. This queue is sorted by VCPU priority. A VCPU's priority can be one of the two value, they are over or under representing whether this VCPU has or hasn't yet exceeded its fair share of CPU resource in the ongoing accounting period. When inserting a VCPU onto a run queue, it is put after all other VCPUs of equal priority to it. As a VCPU runs, it consumes credits. Every so often, a system-wide accounting thread recomputes how many credits each active VM has earned and bumps the credits. Negative credits imply a priority

of over. Until a VCPU consumes its allotted credits, it priority is under. On each CPU, at every scheduling decision, the next VCPU to run is picked off the head from the run queue. The scheduling decision is the common path of the scheduler and is therefore designed to be light weight and efficient. No accounting takes place in this code path. When a CPU doesn't find a VCPU of priority under on its local run queue, it will look on other CPUs for one. This load balancing guarantees that each VM receives its fair share of CPU resources system-wide. Before a CPU goes idle, it will look on other CPUs to find any runnable VCPU. This guarantees that no CPU remains idles when there is runnable work in the system.

Related work

Most of the efforts to date can be classified into three main categories: (1) performance monitoring and enhancement of VMs hosted on a single physical machine (2) performance evaluation, enhancement, and migration of VMs running on multiple physical hosts(3) performance comparison conducted with different platforms or different implementations of VMMs, such as Xen and KVM , as well as the efforts on developing benchmarks Given that the focus of this paper is on performance measurement and analysis of network I/O applications in a virtualized single host, in this section we provide a brief discussion on the state of art in literature to date on this topic. Most of the research on virtualization in a single host has been focused on either developing the performance monitoring or profiling tools for VMM and VMs, represented by or conducting performance evaluation work by varying VM configurations on host capacity utilization or by varying CPU scheduler configurations, especially for I/O related performance measurements. For example, some work has focused on I/O performance improvement by tuning I/O related parameter such as TCP Segmentation Offload, network bridging.

Proposed System Architecture



Proposed Architecture.

Here the virtualization technology which uses the hypervisor to install the Eucalyptus software in single physical machine for setting up a cloud computing environment. By using the hypervisor, the front end and back end of eucalyptus software will be installed in the same machine. The performance will be measured based on the interference in parallel processing of CPU and network intensive workloads by using the Xen Virtual Machine Monitors.

Experimental Setup

A new open-source framework called Eucalyptus has been released that allows users to create private cloud computing grids that are API-compatible with the existing Amazon standards. Eucalyptus leverages existing virtualization technology (the KVM or Xen hypervisors) and popular Linux distributions. To test Eucalyptus, a simplified two-node cluster was used. A front-end node with two network interfaces was connected to both the campus network and a private test network, and a back-end node was connected only to the private network. Both networks ran at gigabit speeds. The private network was configured to support jumbo frames with a 9000 byte MTU to accelerate EBS performance. The ATA over Ethernet protocol used as the transport mechanism behind EBS limits the disk request size to a single Ethernet frame for simplicity, and fragments larger client requests. Thus, using the largest Ethernet frame size possible for both uses the disk more efficiently and reduces the protocol overhead in relation to the payload. The front-end node was equipped with two AMD Opteron processors running at 2.4GHz with 4GB of RAM and a 500GB hard drive. It was configured to run the CLC, CC, EBS, and WS3 services. The back-end node was equipped with two Quad-Core AMD Opteron processors running at 3.1GHz with 16GB of RAM and a 500GB hard drive. These processors support the AMD-V virtualization extensions as required for KVM support in Linux.

The back-end node was configured to run the NC service and all virtual machine images. To provide a performance baseline, the storage and network components were profiled outside of the virtual machine. For storage, the Seagate Barracuda 7200.11 500GB hard drive has a peak read and write bandwidth of approximately 110MB/s, assuming large block sizes (64kB+) and streaming sequential access patterns. For networking the virtual switch was provided for bridging. In an ideal cloud computing system, this performance would be available to applications running inside the virtual environment.

Eucalyptus with KVM: In the first configuration, Eucalyptus with the KVM hypervisor was used. This is a default installation of Ubuntu Enterprise Cloud (UEC), which couples Eucalyptus 1.60 with the latest release of Ubuntu 9.10.

The key benefit of UEC is ease-of-installation as it took less than 30 minutes to install and configure the simple two-node system.

Eucalyptus with Xen

Eucalyptus was used with the Xen hypervisor. Unfortunately, Ubuntu 9.10 is not compatible with Xen when used as the host domain (only as a guest domain). Thus, the Centos 5.4 distribution was used instead because of its native compatibility with Xen

Performance Metrics

The following metrics are used in our measurement study. They are collected using Xenmon [8] and Xentop [15].

• **Server throughput** (#req/sec). It quantitatively measures the maximum number of successful requests served per second when retrieving web documents.

• **Normalized throughput**. We typically choose one measured throughput as our baseline reference throughput and normalize the throughputs of different configuration settings in order to make adequate comparison.

• **Aggregated throughput** (#req/sec). We use aggregated throughput as a metric to measure the impact of using varying number of VMs on the aggregated throughput performance of a physical host.

• **CPU utilization** (%). To understand the CPU resource sharing across VMs running on a single physical machine, we measure the average CPU utilization of each VM, including Domain0 CPU usage and guest domain CPU usage respectively.

• **Network I/O per second** (Kbytes/sec). We measure the amount of network I/O traffic in KB per second, transferred from a remote web server for the corresponding workload.

Impact Analysis

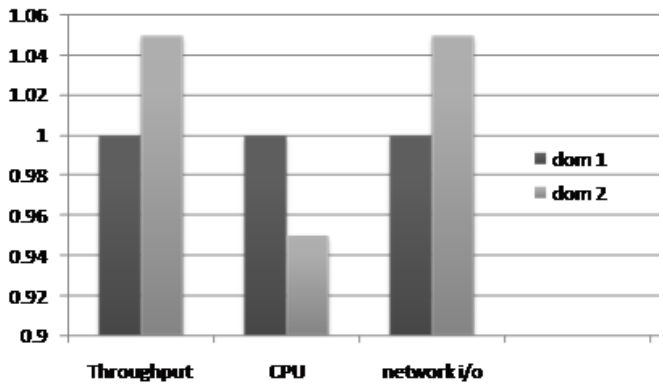
In this section we provide a detailed performance analysis of maintaining idle VM instances, focusing on the cost and benefit of maintaining idle guest domains in the presence of network I/O workloads on a separate VM sharing the same physical host.

Concretely, we focus our measurement study on addressing the following two questions: First, we want to understand the advantages and drawbacks of keeping idle instances from the perspectives of both cloud providers and cloud consumers. Second, we want to measure and understand the start-up time of creating one or more new guest domains on a physical

host, and its impact on existing applications. Consider a set of n (n>0) VMs hosted on a physical machine, at any given point of time, a guest domain (VM) can be in one of the following three states: (1) execution state, namely the guest domain is currently using CPU; (2) runnable state, namely the guest domain is on the run queue, waiting to be scheduled for execution on the CPU; and (3) blocked state, namely the guest domain is blocked and is not on the run queue. A guest domain is called idle when the guest OS is executing idle-loop.

Measurement

We set up our environment with one VM (VM1) running one of the six selected network I/O workloads of 1 KB, 4 KB, 30 KB, 50 KB, 70 KB and 100 KB. The value of each I/O workload characteristics is measured at 100% workload rate for the given workload type. Comparing with network-intensive workloads of 30 KB, 50 KB, 70 KB, and 100 KB files, the CPU-intensive workloads of 1 KB and 4 KB files have at least 30% and 60% lower event and switch costs respectively because the network I/O processing is more efficient in these cases. Then normalized throughput, CPU utilization and Network I/O between Domain1 and Domain2, both with identical 1kB application at 50% workload rate.



Concretely, for 1 KB and 4 KB workloads, driver domain has to wait about 2.5 times longer on the CPU run queue for being scheduled into the *execution state* and the guest domain (VM1) has 30 times longer waiting time. However, they are infrequently blocked for acquiring more CPU resource, especially in the guest domain the block time is less than 6%. We notice that the borderline network-intensive 10 KB workload has the most efficient I/O processing ability with 7.28 pages per execution while the event and switch numbers are 15% larger than CPU-intensive workloads. Interesting to note is that initially, the I/O execution is getting more and more efficient as file size increases. However, with file size of workload grows larger and larger, more and more packets need to be delivered for each request. The event and switch number are increasing gradually as observed. Note that the VMM events per second are also related to request rate of the

workload. Though it drops slightly for the workload of file size 30-100 KB, the overall event number of network-intensive workloads is still higher than CPU-intensive ones. With increasing file size of shorter workloads (1 KB, 4 KB and 10 KB), VMs are blocked more and more frequently. Finally, I/O execution starts to decline when the file size is greater than 10 KB. The network I/O workloads that exhibit CPU bound are now transformed to network bounded as the file size of the workloads exceeding 10 KB and the contention for network resource is growing higher as the file size of the workload increases. In our experiments, the 100 KB workload shows the highest demand for the network resource. These basic system level characteristics in our *basecase* scenario can help us to compare and understand better the combination of different workloads and the multiple factors that may cause different levels of performance interferences with respect to both throughput and net I/O.

Conclusion

Cloud computing offers scalable infrastructure and software off site, saving labor, hardware, and power costs. Financially, the cloud's virtual resources are typically cheaper than dedicated physical resources connected to a personal computer or network. This project proposes the virtualization technology of using the hypervisor to install the Eucalyptus software in single physical machine. By using the hypervisor, the front end and back end will be installed in the same machine. Then performance will be measured based on the interference in parallel processing of CPU and network intensive workloads by using the Xen Virtual Machine Monitors. The main motivation of this project is to provide the scalable virtualized data center.

References

1. Chen.V,Kaeli.D, and Murphy.D, 'Performance Evaluation of Virtual Appliances,' Proc. First International Workshop on Virtualization Performance: Analysis, Characterization, and Tools (VPACT 08), April, 2008.
2. Chinni.s and Hiremane .R, 'Virtual Machine Device Queues: An Integral Part of Intel Virtualization Technology for Connectivity that Delivers Enhanced Network Performance,' white paper, Intel, 2007.
3. Gupta, R. Gardner, L. Cherkasova, XenMon: QoS Monitoring and Performance Profiling Tool, http://www.hpl.hp.com/techreports/2005/HPL-2005-187.html
4. www.xen.org/files/summit_3/sched.pdf

5. P. Padala, X. Zhu, Z. Wang, S. Singhal and K. Shin, "Performance evaluation of virtualization technologies for server consolidation," HP Laboratories Report, NO. HPL-2007-59R1, September 2008.
6. R. Rose, "Survey of system virtualization techniques", Technical report, March 2004.
7. S. Wardley, E. Goyer, and N. Barcet. Ubuntu enterprise cloud architecture. Technical report, Canonical, August 2009.
8. http://linux.die.net/man/1/xentop
9. P. Padala, X. Zhu, Z. Wang, S. Singhal and K. Shin, "Performance evaluation of virtualization technologies for server consolidation," HP Laboratories Report, NO. HPL-2007-59R1, September 2008.
10. http://en.wikipedia.org/wiki/Cloud_computing
11. KVM http://www.linux-kvm.org/page/Main_PageP.
12. Yiduo Mei, Ling Liu, Senior Member, IEEE, Xing Pu, Sankaran Sivathanu, and Xiaoshe Dong ,' Performance Analysis of Network I/O Workloads in Virtualized Data Centers' IEEE transactions on service computing, 2010.
13. Single Root I/O Virtualization and Sharing Specification revision 1.0, specification by Peripheral Component Interconnect Special Interest Group, Sept. 2007.
14. D. Gupta, R. Gardner, L. Cherkasova, XenMon: QoS Monitoring and Performance Profiling Tool, http://www.hpl.hp.com/techreports/2005/HPL-2005-187.html.