Network performance with dynamic workloads with resource demands in hyperconverged environment

Rohit Dhangar, Ravi Bansal, Ken Kumar, Sandip Chakraborty
Department of Computer Science and Engineering, Indian Institute of Technology Kharagpur, INDIA 721302
Email: dhangar.rohit@gmail.com, ravib@iitkgp.ac.in, keniitkgp13@gmail.com, sandipc@cse.iitkgp.ernet.in

***Abstract*—Most of the IT companies used to deal with growth of business applications in a silo-like fashion. They dedicated a set of resources which support a single set of requirements and processes and could not easily be optimized or reconfigured. They used to get away with these type of inefficient, high cost, over provisioned model and still compete. However with the recent boom in technology and unprecedented growth in the number of customers, the *always over provisioning mentality* is no longer acceptable. Hyper Converged architecture is an attempt to simplify IT and reduce the overall cost of infrastructure. The Hyper Converged architecture allows us to establish a pool of virtualized resources that is shared between various applications. In this paper, we analyse the factors and various parameters that affect the network performance in a hyper-converged environment. We identify which parameters are stable metrics while analysing a certain experimental setup and how the variation of network workload affects those parameters. The main objective of conducting these experiments is to get an understanding of how the resources get allocated when different types of applications are running across different VMs so that we can come up with techniques to ensure that quality-of-service is maintained irrespective of the network overload and help cloud service providers and customers to make informed choices while selecting the type of services that they want to provide or use.**

I. Introduction

The over provisioning approach of most of the IT companies over the years have resulted in a phenomenon called IT sprawl. With the rapid growth in technology as well as increase in demand, these companies are now facing maintenance and scalability issues. Most of them are forced to spend a large part of their budget for the maintenance of the resources already deployed, adversely affecting the fundings for innovation and things that would help them to keep the business more competitive. This combined with higher costs of energy, limited space and capacity, and fewer resources severely hurts business agility. It takes weeks or months for these companies to get a new application or service up and running. A converged infrastructure addresses the problem of siloed architectures and IT sprawl by pooling and sharing IT resources. The main philosophy behind it is to somehow establish a pool of virtualized server, storage, and networking capacity that can be shared between many applications instead of providing a dedicated set of resources for each and every service these companies provide. Infact most of the leading names in the IT sector are shifting towards a cloud computing environment. A virtualized cloud computing environment is viewed as a virtualized data center with a network of physical computing nodes (machines), and each physical machine is hosting multiple virtual machines (VMs). The VMM(virtual machine monitor) allocates the resources to the different VMs as per the needs and demands. However, several studies have documented that current implementation of VMMs does not provide sufficient performance isolation to guarantee the effectiveness of resource sharing, especially when the applications running on multiple VMs of the same physical machine are competing for computing and communication resources. As a result, both cloud consumers and cloud providers may suffer from unexpected performance degradation in terms of efficiency and effectiveness of application execution or server consolidation. In this project we analysed how the network performance is affected by network intensive and cpu intensive applications. An in-depth study into how the resource allocation get affected due to different VMs hosted on the same machine, serving different applications provided us an insight into reasons that severely affect the quality-of-service in these cases. We also investigated the effects of co-locating different types of applications on different VMs(a small scale simulation of a practical case) and observed the network performance. We summarize our observations and analysis of the different simulated scenarios in the succeeding sections.

II. Problem Definition

In this paper we present an extensive performance study of network I/O workloads in a virtualized environment. We set up two VMs on a single physical host machine and observe how the performance is affected by the types of applications running on them. One important thing to note here is that we first decided upon the parameters that we were going to use in order to compare one scenario with the other. We need a set of parameters to serve as our stable metric because comparing the different situations based on a single parameter can often be misleading. For example, if we choose the number of requests served per unit time as a parameter, then in situation when the bandwidth is very low, then the network will get saturated very quickly and no difference will be observed on the throughput even if we vary the setup. So we decided upon the following three parameters namely, 1) Throughput of the system 2) Cpu utilization and 3) Network Bandwidth consumption. The reasons for selecting these three metrics as our parameters was driven by the fact that they were easy to measure and provided an overall view about the system's response when it processed different types of applications. In this paper we present an analysis on these factors and how they can impact the throughput performance and resource sharing effectiveness.

III. Testbed Setup

A. Hardware and Software

1) Ovirt: oVirt is free, open-source virtualization management platform. It was founded by Red Hat as a community project on which Red Hat Enterprise Virtualization is based. It allows centralized management of virtual machines, compute, storage and networking resources, from an easy to use web-based front-end with platform independent access.

2) SPICE: Spice XPI is SPICE extension for mozilla allows the client to be used from a web browser (on linux clients). The Spice project aims to provide a complete open source solution for interaction with virtualized desktop devices.The Spice project deals with both the virtualized devices and the front-end. Interaction between front-end and back-end is done using VD-Interfaces. The VD-Interfaces (VDI) enable both ends of the solution to be easily utilized by a third-party component. Spice is an essential part of Ovirt.

3) NetData: Netdata is a highly optimized Linux daemon providing real-time performance monitoring for Linux systems, Applications, SNMP devices, over the web. Blazingly fast and super efficient mostly written in C.

4) Iperf: iPerf3 is a tool for active measurements of the maximum achievable bandwidth on IP networks. It supports tuning of various parameters related to timing, buffers and protocols (TCP, UDP, SCTP with IPv4 and IPv6). For each test it reports the bandwidth, loss, and other parameters.

5) Apache HTTP Web Server: We used apache http web server to setup and host web sites inside VMs. We later on used these websites to test VM performance for our setup.

6) Weighttp: Weighttp (pronounced weighty) is a lightweight and small benchmarking tool for web servers. It was designed to be very fast and easy to use and only supports a tiny fraction of the HTTP protocol in order to be lean and simple. Weighttp supports multi threading to make good use of modern CPUs with multiple cores as well as asynchronous i/o for concurrent requests within a single thread.

B. Setup Installation Steps

1) Install latest centOS on host machine.
2) Install Ovirt hosted setup.
3) Install VDSM(ovirt engine).
4) Create Virtual Data center using Ovirt web API.
5) Create NFS data storage, attach to data center created above.
6) Create ISO Domain (for storing VM ISO images), attach to data center created above.
7) Upload ISO images (fedora 22) to ISO domain of data center.
8) Create two virtual machines of following configuration.
   - Two virtual CPUs per VM
   - 4 GB RAM per VM
   - 50 GB storage pool
   - Common network (default ovirt management network)
9) Install SPICE, web based display for both VMs.
10) Install NetData, for real time monitoring of resources by VMs and host machine.
11) Install Iperf on both VMs and host for network workload generation.
12) Install CPUtest on both VMs and host for CPU workload generation.

IV. Performance Analysis

A. Identical VM and Workload

We started our experiments with 2 VMs with 2 VCPUs each. The objective of these experiments is to test the impact of running application on one VM on performance of other VM. CPUtest tool is used to generate the workload on VM. The experiments are broadly divided into two types of category based on their resource utilization -:

1) CPU Intensive Test
2) Network Intensive Test

1) CPU Intensive Test: CPU Intensive Tests consist of the set of experiments based on CPU Intensive applications. Both the VM's have CPU intensive application and their activeness define the running of the application in the respective VM. We observed CPU utilization of host machine under 3 situations -:

- When both the VM's are idle.
- When one VM is running and other is idle.
- When both the VM's are running.

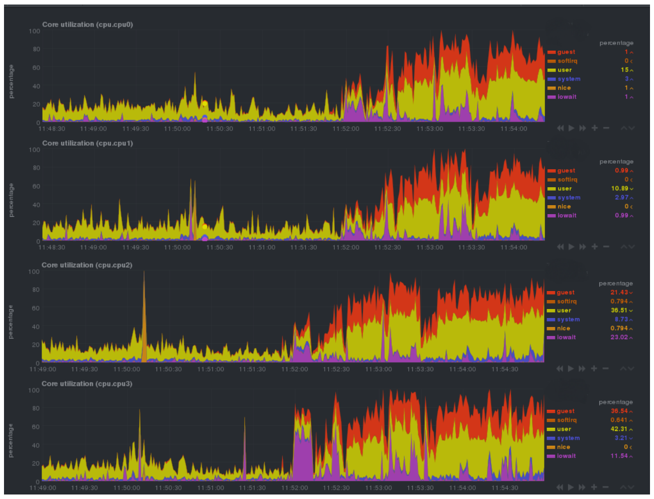

Fig. 1: CPU utilization of host machine when one VM is active

Fig 1 shows the CPU utilization of host machine for the case when only one VM is active. Four graphs in the figure corresponds to the four cores i.e. core0, core1, core2, core3 from top to bottom. Observations from the fig1 -:

- Initial part of the graph corresponds to the idleness of both VM's. The CPU utilization of all the cores is nearly zero with some exceptions of login spikes.

- Later part of the graph corresponds to the situation when one VM is active and the other is idle. We can observe the fact that the utilization of each core rarely cross 80 percent of their full capacity.

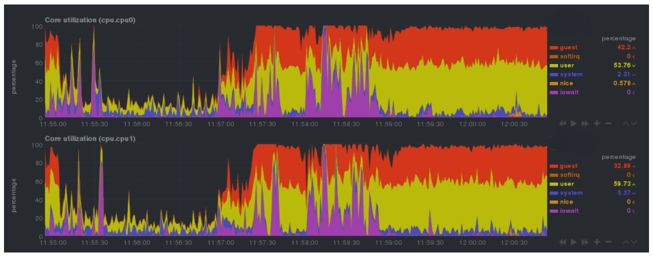

Fig. 2: CPU utilization of host machine when both the VMs are active

Fig 2 shows the CPU utilization of host machine for the case when both the VMs are active. Two graphs in the figure corresponds to core0 and core1 from top to bottom. We can observe from the figure that both both core is utilizing its maximum capacity (almost 100 percent CPU usage).

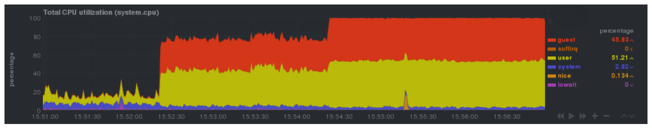

Fig. 3: Aggregated CPU utilization of host machine when VM's are idle, when one is active and when both are active

Fig 3 shows the aggregated status of all 4 CPU cores as single physical CPU status. Initial part of graph represents a scenario when both the VMs are idle and they are only logged in. Middle part of the graph represents a scenario when only one VM is active and last part of the graph represents a scenario when both the VM's are active. From this graph we can see that how running different VMs in different scenario can affect CPU utilization of the host machine.

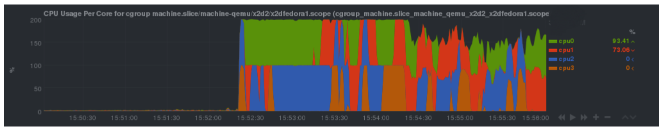

Fig. 4: CPU Usage by VM1 and VM2

Fig 4 shows the graph corresponding to the status of VM1 (Fedora 1). This graph depicts the CPU usage of VMs instead of the host machine. Y-axis of the graph depicts CPU usage corresponding to 400% i.e. 4 cores, one VM can utilize 200% (2 cores) at max as they have been allocated 2 VCPUs each. Upper part of the graph corresponds to VM1 and the lower part corresponds to VM2. First part of the graph from left shows the case when both the VM's are idle and the CPU utilization is nearly zero. Middle part of the graph shows the case when one of the VM is active and it uses 200% out of 400% CPU resource which is its full capacity. Last part of the graph corresponds to the situation when both the VMs are active, we can see that the CPU usage of each VM is dropped from 200% to 150% because another VM is also using the same amount i.e. 150% and host machine is using 50-60% of the CPU resources which sums up to 400%. This experiments depicts that the status of one VM affects the status of other VM.

Conclusion made from the above experiments -:

- When both the VM's are idle, only 15% of host CPU is being utilized. When either one of the VM is active 80% of CPU is being utilized. When both VM's are active 100% of CPU is being utilized.
- When either of the VM is active, mainly CPU core0 is being utilized but when both the VM's are active all the CPU cores are being utilized equally. Also the context switch is much higher in later scenario.
- When both the VM's are active most of CPU is utilized is by VM's and only small chunk i.e 10-15% is being utilized by hypervisor.

2) Network Intensive Test: Network intensive test consist of the set of experiments based on network intensive applications. Both the VM's have network intensive application and their activeness define the running of the application in the respective VM. Both the VMs are connected to common network i.e. ovirt management network. Maximum bandwidth of network is 40 Gbps. We are using iperf as network workload generation tool for these tests. We observed parameters like bandwidth utilization, number of packets sent and received for ovirt management network and each individual VM in following 3 situations -:

- When both the VM's are idle.
- When one VM is running and other is idle.
- When both the VM's are running.

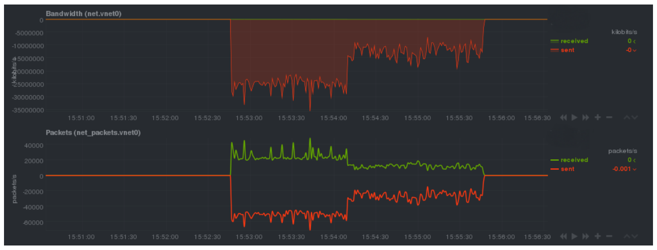

Fig. 5: Utilization of network resources by VM1

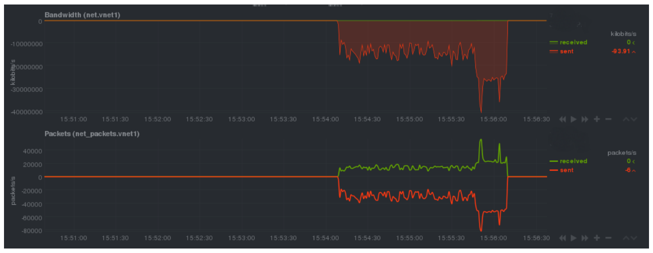

Fig. 6: Utilization of network resources by VM2

First and second graph of Fig 5 corresponds to bandwidth utilization and network I/O (i.e. packets sent and received) by VM1 respectively. Initial part of the graph (i.e. left side) depicts the condition when the VM1 was idle. In the middle part of the graph when the bandwidth utilization is maximum i.e. 40 GB, is the region where iperf network bandwidth test is running on VM1 and VM2 is still idle. As soon as we start iperf network bandwidth test on VM2, depicted in the last part of the graph, bandwidth utilization of VM1 drops by 50% to 20 GB. We can observe here that both the VM is sharing the bandwidth equally. Fig 6 is same as Fig 5 but for VM2 which can be seen as a continuation of the graph of Fig 5.

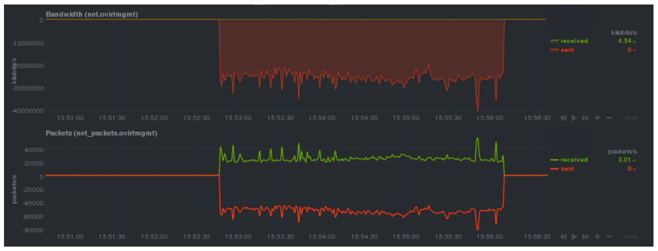

Fig. 7: Utilization of network resources by VM2

Fig 7 shows common network utilization of both the VMs. Here we can see that the maximum bandwidth has been utilized irrespective of whether one VM is active or two VMs are. This graph is for entire ovirt management network. Conclusion made from network intensive experiments -:

- Bandwidth utilization is negligible in the case when both VMs are idle.
- Bandwidth has been utilized fully by the single VM when other is inactive.
- Bandwidth is equally shared when both the VMs are active.
- This kind of bandwidth allocation policy does not guarantee any QOS for VM's.

B. VMs with varying workloads and scenarios

Previous set of experiments were performed on VM's with identical configuration. We now changed the VCPUs allocated to each VM to mimic real life scenario. In this set of experiments, VM1 is allocated 1 VCPU and VM2 is allocated 3 VCPUs. Common network bus of 100 Mbps is used for our experiments in this section. We have used Apache HTTP server on both the VMs to host website which host CPU and network intensive application. We have considered 4 types of application i.e. 1 Kb, 10 Kb, 100 Kb, 500 Kb for our experiments. 1 Kb and 10 Kb applications are considered as CPU intensive as overhead will be more to process large number of packets for the same amount of data as compared to other applications. 100 Kb and 500 Kb application are considered as network intensive application as the packet size is large which results in more use of bandwidth for the same number of packets as compared to other applications. Web server performance is tested by weighttp tool. Before moving on to the collocation of CPU intensive applications, we observed the CPU utilization of VM1 and VM2 when they run one at a time. This scenario has been explained in identical VMs section in detail. We want to observe the CPU percentage so that we can relate to it in the next section while comparison.

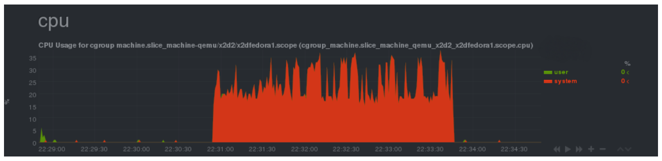

Fig. 8: Utilization of CPU by VM1 when 1Kb application is running on it and VM2 is idle

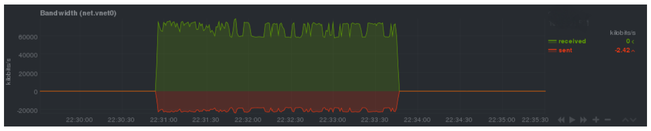

Fig. 9: Bandwidth utilization by VM1 when 1Kb application is running on it and VM2 is idle

Fig 8 shows the CPU utilization by VM1 which corresponds to 25% CPU usage. Fig 9 shows that VM1 is utilizing 60 Mbps of 100 Mbps available. We can observe that the CPU allocated to VM1 is bottleneck here as it is fully utilized.

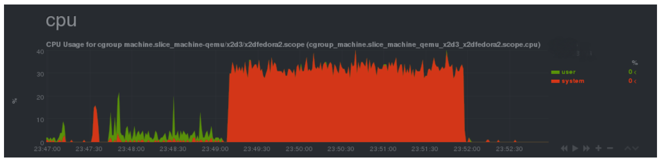

Fig. 10: Utilization of CPU by VM2 when 1Kb application is running on it and VM1 is idle

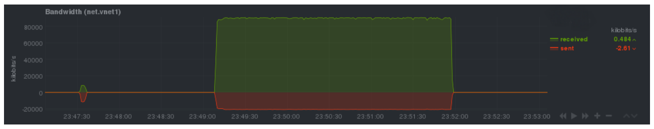

Fig. 11: Bandwidth utilization by VM2 when 1Kb application is running on it and VM1 is idle

Fig 10 shows the CPU utilization by VM2 which corresponds to 35% CPU usage, here as VM2 has more cores, it is using more CPU than VM1. Fig 11 shows the bandwidth utilized by VM2. Due to the extra CPU utilization, VM2 is able process more packets than VM1 which is the cause for achieving more bandwidth utilization i.e 90Mbps compared to 60Mbps in VM1.

1) Testing VM1 and VM2 with CPU Intensive application simultaneously: We started with collocating two CPU intensive application by running 1Kb application on both the VMs simultaneously. We want to observe the performance of VM and how it is affected by neighboring VM. Our objective is to test CPU utilization of VM1, VM2 and hypervisor. We also want to test network utilization of VM1 and VM2 by observing bandwidth utilization and number of HTTP requests served.

Fig 12 shows the CPU utilization of VMs, CPU utilization of hypervisor is not considered here. First graph in the Fig

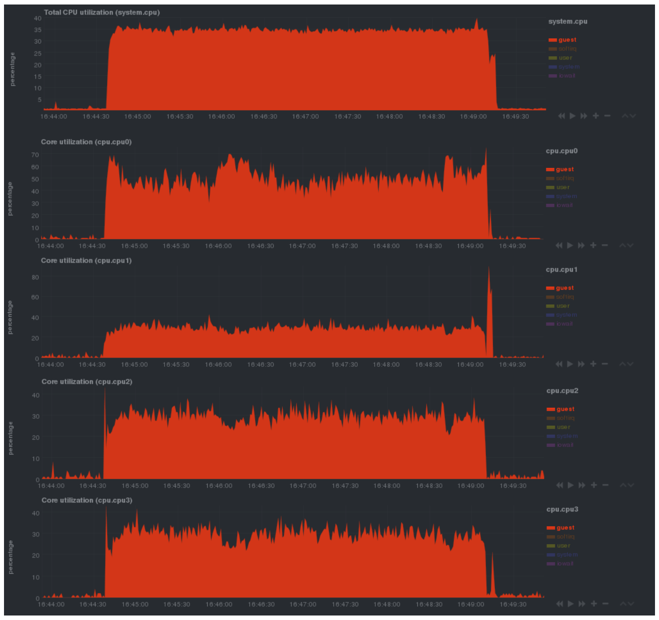

Fig. 12: Host CPU utilization when 1Kb application is running on VM1 and VM2 simultaneously

12 shows the total CPU usage of host CPU in percentage which corresponds to 35% of total CPU resources. Second, third, fourth and fifth corresponds to CPU utilization of vcpu0, vcpu1, vcpu2 and vcpu3 respectively. These four graph represents the units of CPU usage of individual vcpu, where 100 units is the maximum CPU resource of each vcpu which implies 400 units as total CPU resources availability by 4 VCPUs. VM1 is allocated 1 VCPU and utilize vcpu0 for its computation which corresponds to 60 units out of total 400 units of CPU resource available according to second graph. VM2 is allocated 3 VCPUs and utilize 25 units each of vcpu1, vcpu2 and vcpu3 according to the 3rd, 4th and 5th graph in Fig 12. This concludes that VM1 uses 15% and VM2 uses 20% of total CPU resources. It can be verified by the initial statement made by us about CPU utilization of host CPU i.e. 35% (15% by VM1 + 20% by VM2). It is evident that the CPU utilization of VM1 and VM2 decreases by 10% with respect to when they ran independently as seen earlier.

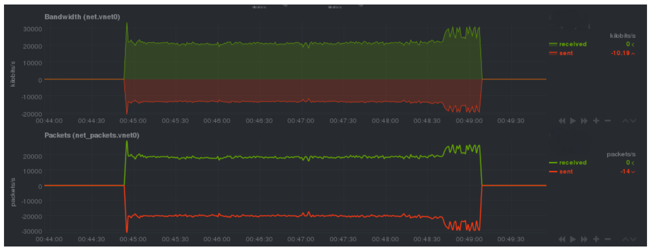

Fig. 13: Bandwidth utilization by VM1 when 1Kb application is running on VM1 and VM2 simultaneously

Fig 13 and Fig 14 corresponds to the bandwidth utilization of VM1 and VM2 respectively. We can observe that the VM2 is

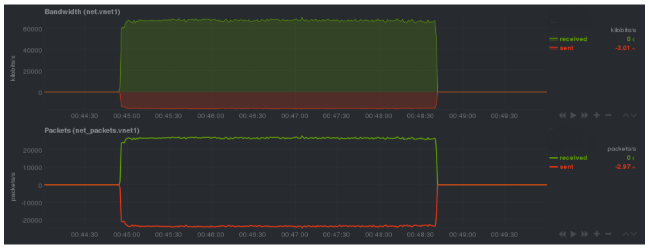

Fig. 14: Bandwidth utilization by VM2 when 1Kb application is running on VM1 and VM2 simultaneously

getting preference or more bandwidth as compared to VM1 i.e. bandwidth utlized by VM1 is 20mbps and VM2 is 60mbps. This difference can be explained by high computation power of VM2, as it has been allolcated 3 VCPUs against 1 VCPU to VM1, it is able to server more number of requests per second. VM1 is serving 3936 req/sec and VM2 is serving 4427 req/sec. Conculsions from the above experiments -:

- VM1 is using 15% of total CPU and only core0 is being utilized by VM1.
- VM2 is using 20% of total CPU and core1, core2, core3 is being utilized by VM2.
- Excluding hypervisor usage total host CPU utilization is 35% .
- Hypervisor is using around 8-10% of CPU and utilizing all four cores.
- VM1 is using 20 Mbps out of total 100 mbps of bandwidth while VM2 is using 60 Mbps out of total 100 Mbps of bandwidth.
- Due to extra core allocated to VM2, it is able to consume more resources than VM1.
- When both the VM's are running CPU intensive applications, their CPU utilization drops by 50% compared to the scenario when only 1 VM is running CPU intensive application.

2) Testing VM1 and VM2 with Network Intensive application: Similar to previous case, we performed test of CPU and network utilization, when only network intensive application runs on VMs. We have already considered the case when the VMs run independently in earlier sections.

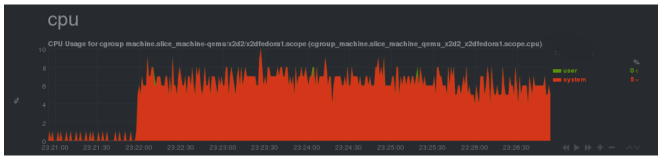

Fig. 15: CPU utilization by VM1 when 500Kb application is running on it and VM2 is idle

In Fig 15 and Fig 16 we have shown the CPU utilization of VM1 and Bandwidth utilization of VM1 respectively, as we have VMs with different VCPUs allocation compared to identical VMs case. We can observe from the graphs in those figures that the average CPU utilization of VM1

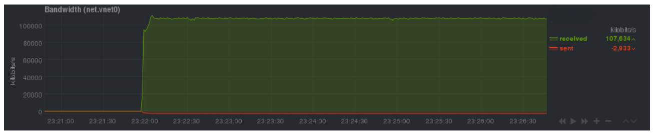

Fig. 16: Bandwidth utilization by VM1 when 500Kb application is running on it and VM2 is idle

is 6% and network is saturated i.e. it is fully utilizing the bandwidth. CPU and network utilization of VM2 is similar to VM1, because, network resources are bottleneck here and we can see from the graphs that network is saturated. Our objective from this set of experiments is to check impact on CPU utilization and bandwidth allocation when both the VM's are running network intensive application and to check how QOS is affected when there is high demand on network bandwidth.

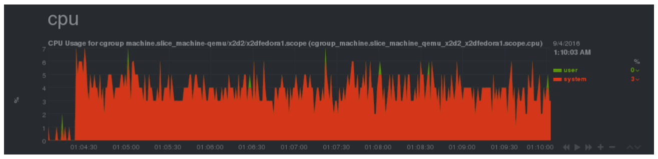

Fig. 17: CPU utilization by VM1 when 500Kb application is running on both VM1 and VM2 simultaneously

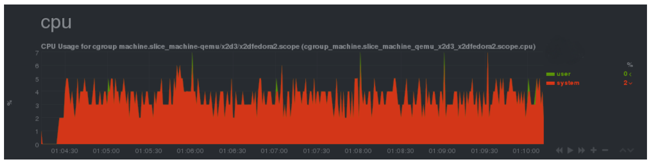

Fig. 18: CPU utilization by VM2 when 500Kb application is running on both VM1 and VM2 simultaneously

Fig 17 and Fig 18 corresponds to the CPU utilization VM1 and VM2 respectively. We can observe that the CPU utilization of both the VM's has been decreased to 3-4% compared to 6% when only one VM was running.

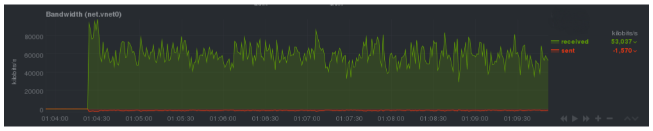

Fig. 19: Bandwidth utilization by VM1 when 500Kb application is running on both VM1 and VM2 simultaneously

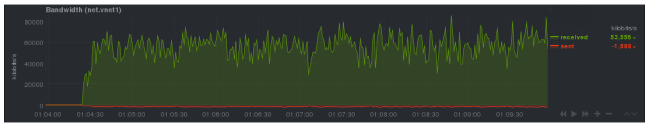

Fig. 20: Bandwidth utilization by VM2 when 500Kb application is running on both VM1 and VM2 simultaneously

Fig 19 and Fig 20 corresponds to the network utilization for VM1 and VM2 respectively. As both VM's are running network intensive application bandwidth is getting divided equally among both the VMs, i.e, 50-60 Mbps as compared to running only 1 VM, this is a 50 percent drop. So, we can say that similar to CPU intensive applications when network intensive applications run simultaneously they affect each others performance and there is a huge drop here i.e 50 percent in terms of bandwidth as well as number of request served. Conclusions from the above set of experiments -:

- When both the VM's are running network intensive application, CPU utilization of both VM's becomes very less due to congestion in network.
- As both VM's are running network intensive application extra CPU resources allocated to VM2 became useless and both VM's were able to process same number of HTTP requests.
- When both VM's are running network intensive application, then due to network chocking or congestion both VM's were able to serve very small number of request. Due to this QOS of web server is affected.

3) Collocating CPU intensive application with network intensive application: From our experiments till now, we can say that the collocation of same type of applications results in decrement of performance. Can we say the same in the case of collocation of different type of application? To answer this question, we performed one more experiment with 1 Kb application running on VM1 and 500 Kb application on VM2 simultaneously.

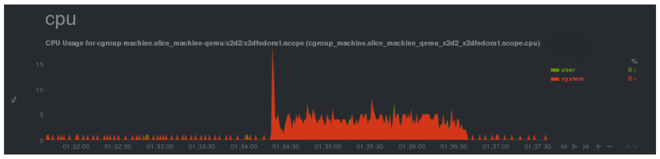

Fig. 21: CPU utilization by VM1 when 1Kb application is running on VM1 and 500 Kb application on VM2 simultaneously

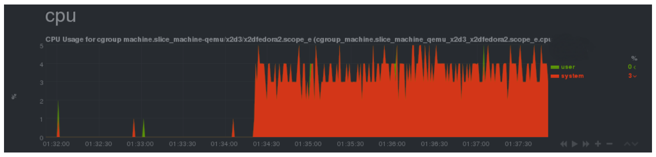

Fig. 22: CPU utilization by VM2 when 1Kb application is running on both VM1 and 500 Kb on VM2 simultaneously

Fig 21 and Fig 22 corresponds to CPU utilization of VM1 and VM2 respectively. Here we can observe that there is huge drop in CPU utilization of VM1 i.e. from 25% when only one VM was active to 5% when it is collocated with network intensive application. CPU utilization of VM2 decreases from 6% to 5% which is very less compared to VM1. This relatively less decrement can be explained by saying that the network is already saturated and further decrement of 10 Mbps that can be seen in the Fig 24, causes 1% decrement. While in the case of VM1, this huge decrement can be explained by the drop in bandwidth allocation from 60 Mbps in the one VM case to 10 Mbps, Fig 23, this implies that the number of packets served will also decrease resulting in less CPU utilization.

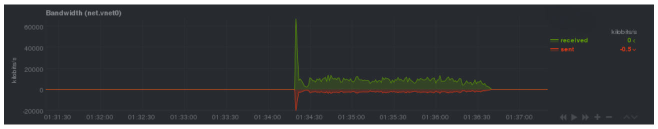

Fig. 23: Bandwidth utilization by VM1 when 1Kb application is running on VM1 and 500 Kb application on VM2 simultaneously

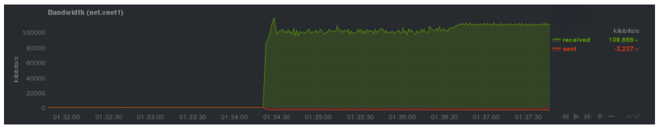

Fig. 24: Bandwidth utilization by VM2 when 1Kb application is running on both VM1 and 500 Kb on VM2 simultaneously

Conclusion from above experiments-:

- Network intensive application captures the bandwidth which affects on QOS of other application.
- Even though we are co-locating different types of application, they can impact on each other depending on resource requirement.
- There must be some kind of resource reservation to achieve desire QOS.

V. Related Works

A fair number of research and development efforts have been dedicated to the enhancement of virtualization technology in the past few years. 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 [1], [2], [3], [4], [5], [6], [7], [8];
2) performance evaluation, enhancement, and migration of VMs running on multiple physical hosts [9], [10], [11], [12], [13], [14];
3) performance comparison conducted with different platforms or different implementations of VMMs [15], [16], [17], such as Xen [18] and KVM [19], as well as the efforts on developing benchmarks [20], [16].
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 [21], [22], or conducting performance evaluation work by varying VM configurations on host capacity utilization or by varying CPU scheduler configurations [2], [23], [24], [5], [25], especially for I/O related performance measurements [18], [1], [26], [27]. For example, some work has focused on I/O performance improvement by tuning I/O related parameter [28], [3], [4], [6], [7], [8], [29], such as TCP Segmentation Offload (TSO), network bridging. Recently, some study showed that performance interference exists among multiple virtual machines running on the same physical host due to the shared use of computing resources [1], [27], [30], [31] and the implicit resource scheduling of different virtual machines done by VMM in the privileged driver domain [32]. For example, in current Xen implementation, all the I/O requests have to be processed by the driver domain, and Xen does not explicitly differentiate the Dom 0 CPU usage caused by I/O operations for each guest domain. The lacking of mechanism for Domain 0 (Dom 0) to explicitly separate its usage for different VMs is, to some degree, contributing to the unpredictable performance interference among guest domains (VMs) [21].
The Hyper Converged data center architecture is still in its infancy and recent research has mainly focused on how to provide basic functionality and features including partitioning of data center network resources and network performance isolation. Current hyper converged environments are predominantly used for workloads such as VDI. However, hyper converged architecture is set to evolve towards support for diverse enterprise applications with varying workload characteristics and associated scalability. This consequentially drives workload mobility requirements for optimal application workload placement on the hyper converged nodes for efficient resource utilization, or workload migration for Business Continuity. Current control mechanisms for storage and compute resources are controlled by software on per node basis, while network resources are controlled globally.

VI. Conclusion

Based on the experiments in performance analysis section, we can conclude that -:

- When only one VM is running, it is free to use other VM's resource. However, when both VMs are running they only use their allocated CPU share.
- VM with more resources dominates VM with less resources.
- If network is choked (network bandwidth is saturated), then extra CPU allocation is wasted (the additional CPU allocation is not utilized).
- CPU utilization of hypervisor increases with increase in number of active VMs which affects the QOS of application running in those VMs.
- When only network intensive application are running on both the VMs, there is no impact on CPU allocation if we have limited bandwidth.
- In a limited bandwidth situation, performance of CPU intensive application decreases more when they are collocated with network intensive applications than with same type i.e. CPU intensive application.
- In a limited bandwidth situation, performance of Network intensive application decreases more when they are collocated with same type i.e. network intensive applications than with different type i.e. CPU intensive application.

References

[1] L. Cherkasova and R. Gardner, "Measuring cpu overhead for i/o processing in the xen virtual machine monitor." in *USENIX Annual Technical Conference, General Track*, vol. 50, 2005.

[2] L. Cherkasova, D. Gupta, and A. Vahdat, "Comparison of the three cpu schedulers in xen," *ACM Sigmetrics Performance Evaluation Rev.*, vol. 35, no. 2, pp. 42–51, Sept. 2007.

[3] A. Gulati, A. Merchant, and P. Varman, "mclock: Handling throughput variability for hypervisor io scheduling," *Proc. Ninth USENIX Symp. Operating System Design and Implementation (OSDI)*, 2010.

[4] M. Kesavan, A. Gavrilovska, and K. Schwan, "Differential virtual time (dvt): Rethinking i/o service differentiation for virtual machines," *Proc. ACM Symp. Cloud Computing (SOCC)*, pp. 27–38, 2010.

[5] M. Lee, A. Krishnakumar, P. Krishnan, N. Singh, and S. Yajnik, "Supporting soft real-time tasks in the xen hypervisor," *Proc. ACM/USENIX IntâĂŹl Conf. Virtual Execution Environments*, pp. 97–108, 2010.

[6] A. Menon, A. L. Cox, and W. Zwaenepoel, "Optimizing network virtualization in xen," in *USENIX Annual Technical Conference*, no. LABOS-CONF-2006-003, 2006.

[7] A. Menon, J. R. Santos, Y. Turner, G. J. Janakiraman, and W. Zwaenepoel, "Diagnosing performance overheads in the xen virtual machine environment," in *Proceedings of the 1st ACM/USENIX international conference on Virtual execution environments*. ACM, 2005, pp. 13–23.

[8] K. K. Ram, J. R. Santos, Y. Turner, A. L. Cox, and S. Rixner, "Achieving 10 gb/s using safe and transparent network interface virtualization," in *Proceedings of the 2009 ACM SIGPLAN/SIGOPS international conference on Virtual execution environments*. ACM, 2009, pp. 61–70.

[9] C. Clark, K. Fraser, S. Hand, J. G. Hansen, E. Jul, C. Limpach, I. Pratt, and A. Warfield, "Live migration of virtual machines," in *Proceedings of the 2nd conference on Symposium on Networked Systems Design & Implementation-Volume 2*. USENIX Association, 2005, pp. 273–286.

[10] M. R. Hines and K. Gopalan, "Post-copy based live virtual machine migration using adaptive pre-paging and dynamic self-ballooning," in *Proceedings of the 2009 ACM SIGPLAN/SIGOPS international conference on Virtual execution environments*. ACM, 2009, pp. 51–60.

[11] P. Padala, X. Zhu, Z. Wang, S. Singhal, K. G. Shin *et al.*, "Performance evaluation of virtualization technologies for server consolidation," *HP Labs Tec. Report*, 2007.

[12] A. Ranadive, M. Kesavan, A. Gavrilovska, and K. Schwan, "Performance implications of virtualizing multicore cluster machines," in *Proceedings of the 2nd workshop on System-level virtualization for high performance computing*. ACM, 2008, pp. 1–8.

[13] C. P. Sapuntzakis, R. Chandra, B. Pfaff, J. Chow, M. S. Lam, and M. Rosenblum, "Optimizing the migration of virtual computers," *ACM SIGOPS Operating Systems Review*, vol. 36, no. SI, pp. 377–390, 2002.

[14] T. Wood, P. J. Shenoy, A. Venkataramani, and M. S. Yousif, "Black-box and gray-box strategies for virtual machine migration." in *NSDI*, vol. 7, 2007, pp. 17–17.

[15] T. Deshane, Z. Shepherd, J. Matthews, M. Ben-Yehuda, A. Shah, and B. Rao, "Quantitative comparison of xen and kvm," *Xen Summit, Boston, MA, USA*, pp. 1–2, 2008.

[16] J. N. Matthews, W. Hu, M. Hapuarachchi, T. Deshane, D. Dimatos, G. Hamilton, M. McCabe, and J. Owens, "Quantifying the performance isolation properties of virtualization systems," in *Proceedings of the 2007 workshop on Experimental computer science*. ACM, 2007, p. 6.

[17] T. Wood, L. Cherkasova, K. Ozonat, and P. Shenoy, "Profiling and modeling resource usage of virtualized applications," in *Proceedings of the 9th ACM/IFIP/USENIX International Conference on Middleware*. Springer-Verlag New York, Inc., 2008, pp. 366–387.

[18] 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 *ACM SIGOPS Operating Systems Review*, vol. 37, no. 5. ACM, 2003, pp. 164–177.

[19] KVM, "http://www.linux-kvm.org/page/Main_Page," 2012.

[20] P. Apparao, R. Iyer, X. Zhang, D. Newell, and T. Adelmeyer, "Characterization & analysis of a server consolidation benchmark," in *Proceedings of the fourth ACM SIGPLAN/SIGOPS international conference on Virtual execution environments*. ACM, 2008, pp. 21–30.

[21] D. Gupta, L. Cherkasova, R. Gardner, and A. Vahdat, "Enforcing performance isolation across virtual machines in xen," in *ACM/IFIP/USENIX International Conference on Distributed Systems Platforms and Open Distributed Processing*. Springer, 2006, pp. 342–362.

[22] D. Gupta, R. Gardner, and L. Cherkasova, "Xenmon: Qos monitoring and performance profiling tool," *Hewlett-Packard Labs, Tech. Rep. HPL-2005-187*, pp. 1–13, 2005.

[23] S. Govindan, A. R. Nath, A. Das, B. Urgaonkar, and A. Sivasubramaniam, "Xen and co.: communication-aware cpu scheduling for consolidated xen-based hosting platforms," in *Proceedings of the 3rd international conference on Virtual execution environments*. ACM, 2007, pp. 126–136.

[24] H. Kim, H. Lim, J. Jeong, H. Jo, and J. Lee, "Task-aware virtual machine scheduling for i/o performance." in *Proceedings of the 2009 ACM SIGPLAN/SIGOPS international conference on Virtual execution environments*. ACM, 2009, pp. 101–110.

[25] N. Nishiguchi, "Evaluation and consideration of the credit scheduler for client virtualization," *Xen Summit Asia*, 2008.

[26] B. Clark, T. Deshane, E. M. Dow, S. Evanchik, M. Finlayson, J. Herne, and J. N. Matthews, "Xen and the art of repeated research." in *USENIX Annual Technical Conference, FREENIX Track*, 2004, pp. 135–144.

[27] Y. Mei, L. Liu, X. Pu, and S. Sivathanu, "Performance measurements and analysis of network i/o applications in virtualized cloud," in *2010 IEEE 3rd International Conference on Cloud Computing*. IEEE, 2010, pp. 59–66.

[28] Y. Dong, X. Yang, J. Li, G. Liao, K. Tian, and H. Guan, "High performance network virtualization with sr-iov," *Journal of Parallel and Distributed Computing*, vol. 72, no. 11, pp. 1471–1480, 2012.

[29] J. Wang, K.-L. Wright, and K. Gopalan, "Xenloop: a transparent high performance inter-vm network loopback," in *Proceedings of the 17th international symposium on High performance distributed computing*. ACM, 2008, pp. 109–118.

[30] S. Sivathanu, L. Liu, M. Yiduo, and X. Pu, "Storage management in virtualized cloud environment," in *2010 IEEE 3rd International Conference on Cloud Computing*. IEEE, 2010, pp. 204–211.

[31] G. Somani and S. Chaudhary, "Application performance isolation in virtualization," in *2009 IEEE International Conference on Cloud Computing*. IEEE, 2009, pp. 41–48.

[32] Y. Koh, R. Knauerhase, P. Brett, M. Bowman, Z. Wen, and C. Pu, "An analysis of performance interference effects in virtual environments," in *2007 IEEE International Symposium on Performance Analysis of Systems & Software*. IEEE, 2007, pp. 200–209.