Effects of Virtualization on Network and Processor Performance using Open vSwitch and XenServer

Adnan Noor Mian
Department of Computer Science
Information Technology University
Arfa Software Technology Park,
Ferozpur Road,
Lahore, Pakistan.
Email: adnan.noor@itu.edu.pk

Ali Mamoon, Raees Khan
Department of Computer Science
Information Technology University
Arfa Software Technology Park,
Ferozpur Road,
Lahore, Pakistan.
Email: {alimamoon@gmail.com, raees.khan@itu.edu.pk}

Ashiq Anjum
School of Computing and Mathematics
University of Derby,
Kedleston Road,
Derby, UK.
Email: a.anjum@derby.ac.uk

Abstract—**Cloud computing is based on virtualization, where a single physical resource is virtualized into multiple virtual resources. Processor and network virtualization offer many advantages like saving in hardware cost, energy consumption, human effort and management of resources. In this paper we have evaluated the effect of network and processor virtualization using popular open source tools, Open vSwitch and XenServer. Based on a number of experiments in which we compared virtualized with non virtualized scenarios, we found that the virtual network using Open vSwitch is secure. Moreover TCP and UDP throughput is not much effected but there is an increase in average Round Trip Time (RTT). Similarly, processor virtualization on XenServer does not affect much the average schedule time in comparison with a non-virtualized machine. We thus conclude that in general a slight decrease in the performance in case of virtualization is not significant as compared with the advantages we get from virtualization when using Open vSwitch with XenServer. This work motivates for the application of virtualization using Open vSwitch and XenServer instead of using non-virtualized environments for setting up data centers.**

Keywords—Open vSwitch; XenServer; network virtualization; cloud computing; data center

I. Introduction

Cloud services are provided by a system which consists of a large pool of hardware and software resources. These resources can be dynamically re-configured to adjust to the variable load and allow for an optimum resource utilization [1]. This scalable and efficient utilization of resources is not possible without the virtualization of hardware and software resources. With the help of virtualization, a single physical processor, memory, storage and network can be used as multiple virtualized resources. These virtualized resources are able to run in parallel resulting in better efficiency of the hardware [2]. Virtualization enables businesses to reduce IT expenses while increasing the efficiency, utilization and flexibility of their existing computer hardware. Figure 1 represents a very basic case of a virtualization environment where multiple virtual resources are being used to provide services to the users. These virtual resources work collaboratively in order to provide services. For collaboration between the virtual resources, there must exist a communication mechanism and a way to manage and control this communication. Such a communication is achieved through network virtualization [2]. In physical networks we can control, track and manage network communication between nodes but in virtual environments these tasks are not easy. All of the communication is carried out at virtual layer where it may or may not pass from any physical network hardware, making it much more difficult to track, secure and isolate the virtual networks [2] [3] [4]. These challenges are handled by network virtualization.

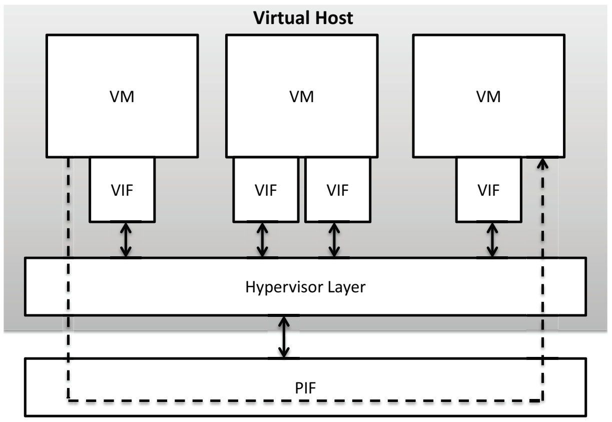

Fig. 1. Virtual Network Data Transfer and Hypervisor

Network virtualization is a networking environment where multiple virtual networks are established on top of a physical network. Each virtual network in a network virtualized environment is a collection of virtual nodes and virtual links. In such a network, we can create and manage multiple virtual networks at software level which can co-exist in isolation without interfering with each other. These networks are then used by service provider to host services for the users [5]. Network virtualization is based on the model of standard L2 switch or IP Router functionality and is included in hypervisor or hardware management layer [3]. This software layer works as a bridge between physical network interfaces (PIFs) and virtual interfaces (VIFs) of virtual machines (VMs). In cloud environment, VMs are not given direct access to the physical network interfaces. Instead these are connected via VIFs to the software layer which is responsible for interconnection of VMs within the same host or to the outer world via PIF, as shown in Figure 1. A virtual machine may have one or more VIFs connected to the software switches in the hypervisor layer.

An emerging application of virtualization is setting up clouds in data centers. Traditionally data centers use dedicated servers to run applications. Such approaches have a number of limitations like not providing dynamic resource allocation, customized applications and network protocols, management policies and performance isolation [6]. Most of these limitations are overcome if processor and network virtualization are used for the implementation of clouds in data centers. For processor and network virtualization, a popular combination is to use XenServer using Open vSwitch for designing clouds. In this paper we evaluate the effect of virtualization on network and processor performance using the combination of XenServer and Open vSwitch. We have found that by using virtualization, there is no significant loss in major network and processor performance. This work motivates for developing virtualized data centers instead of using non virtualized solutions. To the best of our knowledge we have not found such a detailed experimental study on investigating the performance of using Open vSwitch and XenServer.

In rest of the paper, we discuss related work in section II. We then describe the experimental set up and the experiments performed in section III. In section IV we discuss the results and finally in section V we conclude and give a future direction of this work.

II. Related work

Open vSwitch (OVS) is an open source software switch used for network virtualization and communication between virtual machines. We can incorporate this switch in multiple supported hypervisors. Pfaff et al. [3] described that open vSwitch acts as a bridge between VMs and PIFs and replicates the Ethernet (Layer-2) switching behavior. They focused on the implementation aspect of Open vSwitch and how it can be used to handle problems like isolation in joint-tenant environments, mobility across subnets, and distributing configuration and visibility across hosts. We, on the other hand, have focused on the evaluation of Open vSwitch based on multiple parameters.

Pettit et al. [4] incorporated OpenVswitch in XenServer and tested its various capabilities. They argue that Open vSwitch provides most of the capabilities provided by an advance hardware network switch when used with a hypervisor like XenServer. We believe their arguments are correct and we take it further to evaluate the performance of Open vSwitch in XenServer for Network and CPU virtualization.

In [7], authors incorporated Open vSwitch with an open source hypervisor OpenNebula. A single-root I/O virtualization (SR-IOV) based architecture has been proposed in [8]. The authors also proposed a dynamic network interface switching (DNIS) scheme to address the migration challenges and suggest that SR-IOV provides a good solution for high performance I/O virtualization without sacrificing migration. In our paper, the focus is not on the migration aspect in network virtualization. For data center designs using Open vSwitch and XenServer, [8] can serve as a guide for handling migration challenges.

Lee et al. [9] proposed a network services deployment framework i.e. In-Network Processing (INP) which could proficiently shift the network services processing from separate hardware to network devices and computing resources. For switch management, they used OpenFlow and XenServer hypervisor as processing module and Open vSwitch to control the network flow. In our work, the focus is not on the policy implementation, processing, controlling or access management.

In [10] Duan et al. proposed a service-oriented architecture (SOA) in network virtualization environment and evaluated its performance. When SOA was applied within network virtualization environment, the resultant behavior greatly depicted the benefits of merging networks and cloud. They named it as "Network as a Service" (NaaS) [11] and also proposed a framework for NaaS whose main focus is on network service description, discovery and composition. NaaS is also discussed for hybrid clouds in [12]. A joint virtualization of network and cloud resources is been proposed in [13] for addressing resource allocation problem. In our paper, primarily the focus is on the evaluation of network virtualization but the results we got can serve as a guidance for setting up a data centers which can be used for providing network oriented services.

The difference of our research, in general, from all of the previously mentioned works, is that we have evaluated the effects of virtualization using the combination of Open vSwitch and XenServer. We have not focused on the actual implementation or internal working of Open vSwitch. Instead we have focused on the factors which may affect the performance of the system. The work which is close to what we have done in this paper, is given in [2]. Authors in [2] have conducted experiments in network virtualization using Open vSwitch version 1.0.99 and used XenServer as hypervisor. They evaluated the security, performance and QoS service of the Open vSwitch. In our work we have not only focused on the effects of network virtualization but also on the processor virtualization. Moreover, we have done more extensive evaluations. Similarly in [14], authors have evaluated the network virtualization and virtual machines in Amazon clouds, whereas we have done the same for Open vSwitch and XenServer.

III. Experimental Setup

We first installed XenServer version 6.2.0 on two different physical machines. This version has the Open vSwitch version 1.4.6 pre-installed. Moreover this version of XenServer can only be installed on 64-bit machines. XenServer is based on Linux kernel and hence installs like any other Linux flavor except with few differences. XenSever does not allow to create custom size partitions and also manage the storage drive and file system itself. In our experiments, we used one hard drive on each pc for storing VMs. For network configuration, we had 1 NIC (PIF) per physical machine. We configured the network using static addresses and universally unique ids (UUIDs).

After XenServer installation we installed XenCenter, and connected XenCenter to the XenServer. XenCenter provides a centralized control with which we need just one machine to interact with all of the installed XenServers. We then created two VMs on each of the physical machines and installed Windows 7, 64 bit and Ubuntu 12.04, 64 bit on each of the VM on a physical machine. This setup is shown in figure 2. We also installed XenServer Tools on VMs. XenServer tools

TABLE I. XENSERVER CONFIGURATIONS

	XenServer 1	XenServer 2
Processor	Intel Core i3	Intel Core i3
RAM	4GB	4GB
Network Interface	1000Mbps	1000Mbps
XenServer Version	6.2.0	6.2.0
Open vSwitch Version	1.4.6	1.4.6
IP Address	192.168.2.10	192.168.2.20
Subnet Mask	255.255.255.0	255.255.255.0
Default Gateway	192.168.2.1	192.168.2.1
Number of VMs	2	2

act as drivers for the VMs so that VM and XenServer can interact with each other. XenServer tools are only available for specific operating systems. If the operating system is not supported by XenServer and XenServer tools, we will be able to create VM but will not be able to use it properly. List of supported operating systems can be found at [15].

After the successful installation of XenServer tools, we configured VLAN with all the security, QoS and other features. For this purpose we added VIFs and assign them to the VMs. We created VLANs connections and assigned VLAN tags to these connections. These tags help in providing security, since VMs assigned with a tagged VLAN connection can only communicate with a machine having a connection with the same tag.

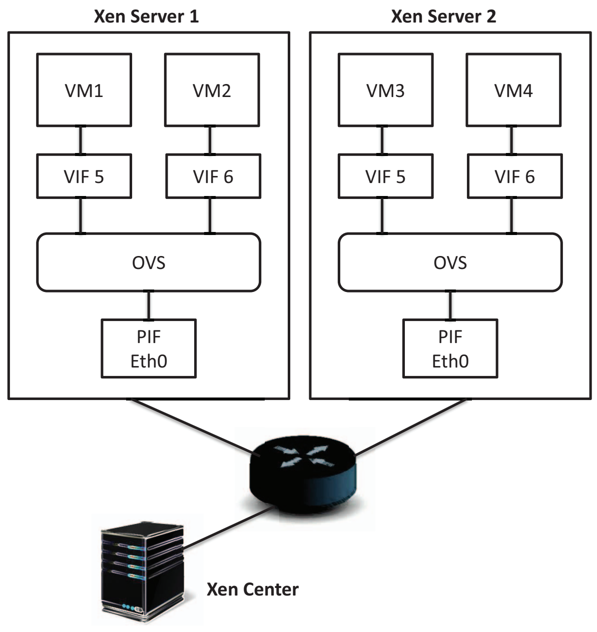

Fig. 2. Experimental Setup

After performing these steps, we have established the virtualized environment by creating XenServers on two different physical machines. The hardware and software configurations for both of these servers are shown in Table I. On each server we have created two VMs running Windows and Linux operating systems respectively. In Figure 2 we have shown the overall view of the experimental setup.

TABLE II. PING AND ARP TEST WITHOUT VLAN TAGS

	VM 1	VM 2	VM 3	VM 4
VM 1	-	Y	Y	Y
VM 2	Y	-	Y	Y
VM 3	Y	Y	-	Y
VM 4	Y	Y	Y	-

TABLE III. PING AND ARP TEST WITH VLAN TAGS

	VM 1 (T-5)	VM 2 (T-6)	VM 3 (T-5)	VM 4 (T-6)
VM 1 (T-5)	-	N	Y	N
VM 2 (T-6)	N	-	N	Y
VM 3 (T-5)	Y	N	-	N
VM 4 (T-6)	N	Y	N	-

IV. RESULTS AND DISCUSSION

Now we discuss different experiments including security, network packets monitoring, throughput measurement, RTT, processor utilization on network activity and processor sharing among virtual resources.

A. Network Virtualization

In this subsection we evaluate the effect of virtualization on different network parameters like enabling security, network throughput and RTT.

Enabling security/traffic isolation: Physical isolation is an important mechanism to improve the security of the network. In physical environment, networks cannot be isolated into different physical subnets. In VLANs we can isolate the communication between machines in the same physical network. In fact machines in the same virtual subnets can communicate with each other whereas those on different virtual subnets can not. This is because that ARP request sent by one machine will not reach the other machine if they are in different virtual subnets [2]. To test the security of the network we have conducted the ping and Address Resolution Protocol (ARP) tests as described below.

We conducted the VLAN test to see whether Open vSwitch is able to isolate VM traffic using VLANs. In this test, we sent the ping request from one VM to other VM. The ping results are shown in table II. Interestingly we found that every VM was able to send and receive packets from all other VMs which was conflicting with statement of isolation. The reason is, that at time of XenServer host installation, one network per physical NIC is created automatically. When XenServer tools are installed on VMs, network is established with this default feature. This network does not involve any VLAN tags.

We setup a new VLAN using an existing NIC. This is represented with VIF (virtual network interface) as NIC with VLAN tag and attached this connection to the VM. XenServer supports up to 4 physical network interfaces per XenServer host and up to 7 virtual network interfaces. There are 3 server side objects that represent network entities of XenServer: Physical Interface (PIF), Virtual Interfaces (VIFs) and Network (Virtual Ethernet switch). Multiple PIFs can be assigned to single network interface card but separate VLANs are assigned to separate PIFs. VIF is the virtual network interface on virtual machine. Network consists of a name, description, universally unique id UUID and collection of PIFs and VIFs. Using this information, we established a network connection with VLANs, using the server on XenCenter.

TABLE IV. TCP AND UDP THROUGHPUT

	Min (MB/s)	Max (MB/s)	Avg (MB/s)
TCP with no VLAN Tag	11.28	11.33	11.31
TCP with VLAN Tag	11.25	11.28	11.27
TCP without VLAN	11.30	11.32	11.31
UDP with no VLAN Tag	28.04	30.28	29.58
UDP with VLAN Tag	22.28	29.90	28.70
UDP without VLAN	7.36	7.55	7.42

TABLE V. ROUND TRIP TIME (RTT)

	Min (ms)	Max (ms)	Avg (ms)
VMs on same physical Machine	0.15	4.42	0.38
VMs on different physical Machine	0.34	12.44	0.64
Non-virtual machines	0.11	12.34	0.23

After assigning newly created configuration to VMs, we again performed the VLAN test. Table III shows result with VLAN tag configuration. When we used the default connection without any VLAN tag, every VM regardless of the hosting server was able to access other VMs. But when VLAN tagged connections were used, only those VMs were able to access other VMs which are connected using connection having same VLAN tag and subnets otherwise connection was failed to established. With these results we can deduce that using Open vSwitch, we can achieve network isolation and thus can assure secure communication using virtual networks in XenServer.

Network Throughput: We conducted the following tests to measure the performance of the virtual networks using both TCP and UDP data streams. We used two different connections with and without VLAN tags. We also applied QoS rate of 100MB to the connection with VLAN tag. For measuring the performance, we used Microsoft NTttcp tool and windows VMs. The results were taken for 20 different measurements by changing the time in each measurement calculation. The first test was run for 10 seconds and in each following test this time span was increased by 1 second.

Figure 3 shows the TCP throughput results. The average frame size was ranging from 1455-1456 during TCP throughput calculation. We observe that there is not much difference in with and without VLAN tag results.

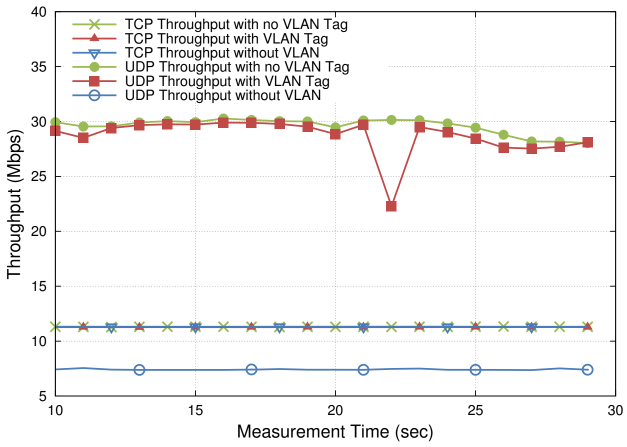

Fig. 3. TCP &UDP Throughput for Open vSwitch (using VLAN tags and without VLAN tags)

Figure 3 shows the plots for TCP and UDP throughput. We see that in case of UDP the throughput is higher than the TCP case. In table IV, we can see that the average TCP throughput with and without VLAN tag is 11.27 MB/sec and 11.31 MB/sec respectively which are quite close. Similarly, the average UDP throughput with and without VLAN tag is 28.71 MB/sec and 29.58 MB/s respectively, which are very much close too . Thus we find that enabling security and network virtualization has virtually no effect on the throughput. Moreover we see that in older version of XenServer and Open vSwitch [2], the TCP throughput variation is high but in newer version of Open vSwitch this variation is small.

Round-Trip Time (RTT): In order to measure the delay in network communication we calculated the RTT. Figure 4 shows the RTT plots when (1) both VMs are on the same physical server, (2) when VMs are on two different physical servers and (3) when the VMs are placed on different non-virtual machines. For each of the cases, we sent 5000 ping probes with the gap of 0.2 seconds. It is quite obvious to see that RTT for case (2) is higher than that of case (1). This is due to the latency involved in the management overhead of Open vSwitch. In network virtualization, request from VIF is transferred to the Open vSwitch which redirects it to the PIF. The summary for RTT is given in table V. The average RTT between VMs on same physical machine is 0.38ms, it is 0.65ms between VMs on different physical machines and 0.23ms between two physical (non-virtual) machines.

We find that the average RTT is maximum for the case of VMs on different physical machines and this increase is about 2 times. The increase is due to the network virtualization in which the processor not only schedules the VMs but also has to do the task of switching.

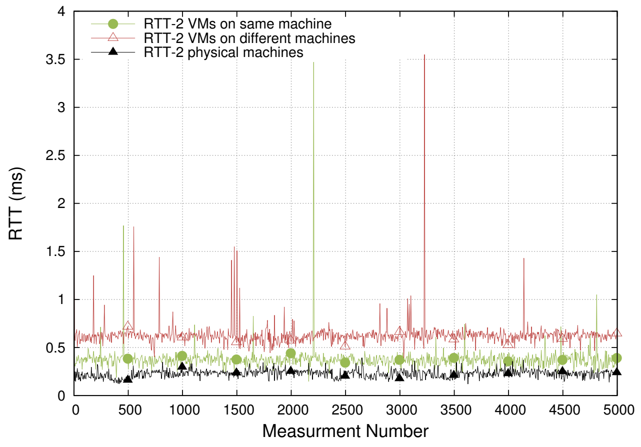

Fig. 4. Round Trip Time (RTT)

B. Processor Virtualization

In this subsection we shall evaluate the effect of virtualization on CPU performance in terms of network activity and processor/CPU sharing.

TABLE VI. CPU UTILIZATION DURING TCP AND UDP

	Min (%)	Max (%)	Avg (%)
TCP with no VLAN Tag	4.34	6.98	5.42
TCP with VLAN Tag	2.65	3.89	3.34
TCP without VLAN	3.06	6.03	3.75
UDP with no VLAN Tag	55.90	58.69	57.56
UDP with VLAN Tag	55.68	59.78	57.47
UDP without VLAN	2.82	5.41	4.22

Virtualized processor utilization for throughput test: Figure 5 shows the plots of processor utilization during the throughput calculations. These plots are obtained with the help of Microsoft NTttcp. The summary of the experiment results has been given in table VI. We see that in case of TCP, the processor utilization is minimal but during UDP data stream, it is up to 60% higher. The average processor utilization in case of TCP with and without VLAN tag is 3.34% and 5.42% respectively. Similarly the average processor utilization in case of UDP with and without VLAN tag is 57.47% and 57.56% respectively. Thus we see that the processor utilization almost remains the same in case of enabling security in network or in other words using VLANs. Though we also find that the processor utilization is higher in case of UDP in comparison with the TCP. This is because in case of UDP, the throughput is higher leading to a higher utilization of processor.

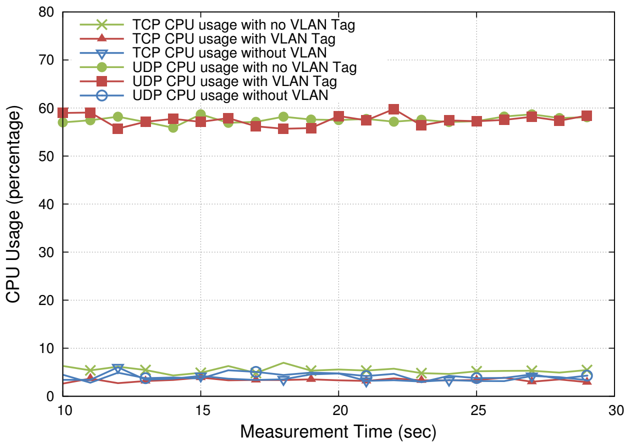

Fig. 5. TCP & UDP Processor Usage (using VLAN tags and without VLAN tags)

Processor virtualization using VM: Virtual machines are being executed on top of XenServer hypervisor. In order to determine processor utilization with XenServer, we conducted this experiment. For XenServer, VMs execution is no more different than scheduling a VM process on physical server. A VM is being executed as a process in XenServer and we compare this with the processor sharing among processes on a non-virtualized machine. For this purpose, we wrote a small application in C++ and executed it on the VMs and physical machine. This application reported the time on which it is scheduled to the processor. While performing this experiment we made sure that no other user process is running during the execution of this test.

We did the test for two cases. In the first case, only one VM was running on XenServer hypervisor which is compared with the non-virtual machine where only one user process

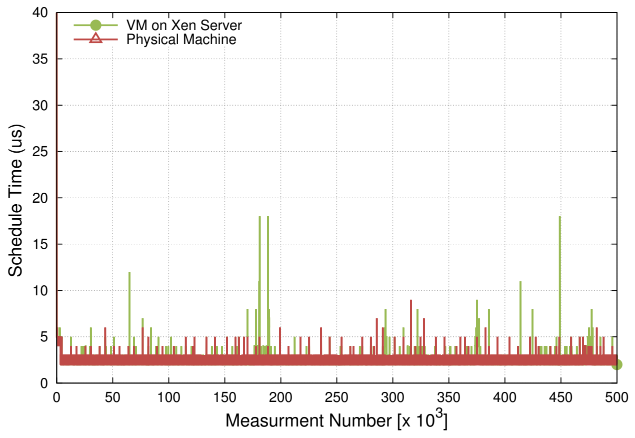

Fig. 6. Processor Sharing between 1 VM with 1 User process and 1 Physical Machine with 1 User Process

TABLE VII. PROCESSOR SHARING BETWEEN 1 VM WITH 1 USER PROCESS AND 1 PHYSICAL MACHINE WITH 1 USER PROCESS

	Min (μs)	Max (μs)	Avg (μs)
VM on XenServer	2	3156	2
Physical Machine	2	68	2

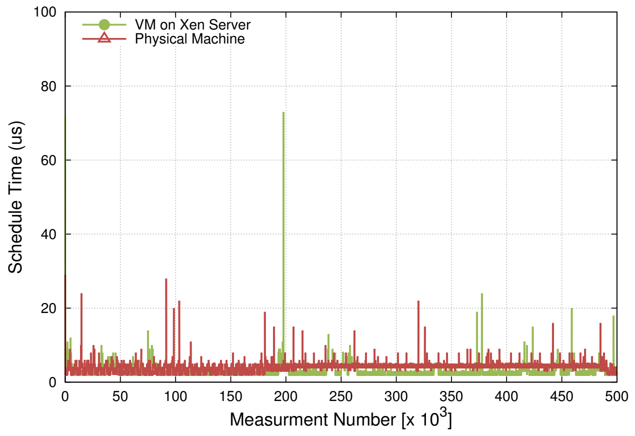

Fig. 7. Processor Sharing between 2 VM with 1 User process each and 1 Physical Machine with 2 User Processes

TABLE VIII. PROCESSOR SHARING BETWEEN 2 VM WITH 1 USER PROCESS EACH AND 1 PHYSICAL MACHINE WITH 2 USER PROCESSES

	Min (μs)	Max (μs)	Avg (μs)
VMs on XenServer	2	6729	3
Physical Machine	2	342	4

was executed. In the second case two VMs were executed in XenServer and compared with non-virtual environment by executing two user processes. Figure 6 shows the plots for the first case. We note that scheduling time in non-virtual machine is negligible. This is because there is only one user process was running while scheduling time for the VM varies. Figure 7 shows the results for the second case. In this case we executed two processes but result is taken for one process. The result for this case is surprising. The schedule time for non-virtual machine did not change and followed the same behavior as in the first case with minimal time. But schedule time for the virtualized machines also has reduced at majority of instants, depicting better processor sharing among VMs. In case of processor sharing, the average sharing time for VMs and physical machine with one process is $2\mu s$ seconds as shown in table VII. The table VIII shows that with two processes, the average sharing time for VMs and physical machine with one process is $3\mu s$ seconds and $4\mu s$ seconds. We thus find that virtualization has negligible effect on processor sharing.

V. Conclusion and Future Work

In network virtualization with Open vSwitch we can achieve network security and scalability, as we can assign multiple VIF to the VMs. A VM can only communicate with those VMs over virtual network which are having same VLAN tags otherwise communication is not possible. The network throughput that we can achieve for TCP or UDP are almost same with or without VLAN tag but the added advantage is the security benefit we achieve using VLAN tags without affecting network performance. The RTT results in virtual network is slightly higher than those in non-virtual environment. This behavior is due to the involvement of hypervisor layer which is acting as a bridge between VIFs and PIFs. Same is the case with processor sharing, where multiple VMs are sharing the same processor as in comparison with non-virtual environment where no processor sharing is actually done. This also adds some latency in process scheduling whereas it is almost equal to zero in non-virtual system.

We thus find the effect of virtualization on network and processor performance using Open vSwitch and XenServer. The effect is negligible for most of the performance parameters as compared with non-virtualized environments. Though there may be an increase in the cost for setting up the virtualized environment but the benefits we get is far more than using a non-virtual environment. This work thus motivates for setting up clouds for data centers based on virtualization. In future we plan to have similar performance evaluations and comparisons for Open Stack, one of the emerging open source cloud middleware.

Acknowledgment

The authors would like to thank Ms. Farah Amjad, research assistant Information Technology University, Lahore, Pakistan for her helpful comments and suggestions.

References

[1] L. M. Vaquero, L. Rodero-Merino, J. Caceres, and M. Lindner, "A break in the clouds: towards a cloud definition," *ACM SIGCOMM Computer Communication Review*, vol. 39, no. 1, pp. 50–55, 2008.

[2] Z. He and G. Liang, "Research and evaluation of network virtualization in cloud computing environment," in *Networking and Distributed Computing (ICNDC), 2012 Third International Conference on*. IEEE, 2012, pp. 40–44.

[3] B. Pfaff, J. Pettit, K. Amidon, M. Casado, T. Koponen, and S. Shenker, "Extending networking into the virtualization layer." in *Hotnets*, 2009.

[4] J. Pettit, J. Gross, B. Pfaff, M. Casado, and S. Crosby, "Virtual switching in an era of advanced edges," in *2nd Workshop on Data Center–Converged and Virtual Ethernet Switching (DC-CAVES)*, 2010.

[5] N. Chowdhury and R. Boutaba, "A survey of network virtualization," *Computer Networks: The International Journal of Computer and Telecommunications Networking*, vol. 54, no. 5, pp. 862–876, 2010.

[6] M. Bari, R. Boutaba, R. Esteves, L. Granville, M. Podlesny, M. Rabbani, Q. Zhang, and M. Zhani, "Data center network virtualization: A survey," *Communications Surveys Tutorials, IEEE*, vol. 15, no. 2, pp. 909–928, Second 2013.

[7] H.-M. Tseng, H.-L. Lee, J.-W. Hu, T.-L. Liu, J.-G. Chang, and W.-C. Huang, "Network virtualization with cloud virtual switch," in *Parallel and Distributed Systems (ICPADS), 2011 IEEE 17th International Conference on*. IEEE, 2011, pp. 998–1003.

[8] 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.

[9] J. Lee, J. Tourrilhes, P. Sharma, and S. Banerjee, "No more middlebox: integrate processing into network," *ACM SIGCOMM Computer Communication Review*, vol. 40, no. 4, pp. 459–460, 2010.

[10] Q. Duan, "Modeling and performance analysis on network virtualization for composite network-cloud service provisioning," in *Services (SERVICES), 2011 IEEE World Congress on*. IEEE, 2011, pp. 548–555.

[11] Q. Duan, Y. Yan, and A. V. Vasilakos, "A survey on service-oriented network virtualization toward convergence of networking and cloud computing," *Network and Service Management, IEEE Transactions on*, vol. 9, no. 4, pp. 373–392, 2012.

[12] D. Kakadia and V. Varma, "Network virtualization platform for hybrid cloud," in *Cloud Computing Technology and Science (CloudCom), 2013 IEEE 5th International Conference on*, vol. 2. IEEE, 2013, pp. 69–74.

[13] J. Soares, J. Carapinha, M. Melo, R. Monteiro, and S. Sargento, "Resource allocation in the network operator's cloud: A virtualization approach," in *Computers and Communications (ISCC), 2012 IEEE Symposium on*. IEEE, 2012, pp. 000 800–000 805.

[14] G. Wang and T. E. Ng, "The impact of virtualization on network performance of amazon ec2 data center," in *INFOCOM, 2010 Proceedings IEEE*. IEEE, 2010, pp. 1–9.

[15] "Citrix XenServer 6.0 Virtual Machine Installation Guide," http://docs.vmd.citrix.com/XenServer/6.0.0/1.0/en_gb/guest.html, 2014, [Online; accessed 29-June-2014].