Virtualizing IMS Core and Its Performance Analysis

Lingxia Liao[1](✉), Victor C.M. Leung[1], and Min Chen[2]

[1] Department of Electrical and Computer Engineering, University of British Columbia, Vancouver, Canada
{liaolx,vleung}@ece.ubc.ca

[2] School of Computer Science and Technology, Huazhong University of Science and Technology, Wuhan, China
minchen@ieee.org

Abstract. Current IP Multimedia System (IMS) industry faces the issue that the complicated architecture of IMS and the huge early investment in its network construction has slowed down its deployment and service innovation. Furthermore, IMS network also causes more computing and network resource waste than current telecom network becuase no existed method can be used to predict the capacity of data service with guaranteed Quality of Service (QoS) in IMS network. Present research and practice consider that virtualizing IMS core and running it on cloud can be a way to solve these problems. However, current research shows the virtualization brings at least five times longer response delays to IMS and makes it unfeasible to be used. We argue that hardware-assisted virtualization technology can improve the virtual machine performance, and through carefully tuning the virtual machine parameters, the overhead caused by virtual machines can be minimized. We choose OpenIMSCore as an IMS core network, IMS Benchmark SIPp as a traffic generator, design and conduct a performance test. The results show that running IMS core network on virtual machines has comparable response delays with it running on bare boxes. It is feasible to virtualize the IMS core network and run it on private clouds.

Keywords: Cloud computing · Virtualization · IMS architecture · Performance testing

1 Introduction

IP Multimedia Subsystem (IMS) is an all-IP core network architecture designed by 3GPP to replace the current mobile circuit switch communication system and support wide range of multimedia applications [1]. It has been deployed in current Long-Term Evolution (LTE) systems to provide core network control for both voice and data services with Quality of Service (QoS) guaranteed. However, IMS network architecture is fully new and very complicated, which creates a big difficulty in network understanding and deployment, and causes huge early

V.C.M. Leung et al. (Eds.): CloudComp 2014, LNICST 142, pp. 53–65, 2015.
DOI: 10.1007/978-3-319-16050-4_5

investment and resource waste. These issues have slowed down the IMS network deployment and service innovation in the reality. Finding a way to reduce the deployment difficulty and the resource waste is significant for IMS industry.

Cloud Computing is a model for enabling convenient, on-demand network access to a shared pool of configurable computing resource that can be rapidly provisioned and released with minimal management effort or service provider interaction [2,3]. The Platform-as-a-Service (PaaS) provided by a cloud can dynamically scale up and down the resource without interrupting the applications running on top of it. Virtualizing IMS core network and running it on cloud PaaS environment can be a way to reduce IMS network deployment difficulty, early investment, and resource waste. Because IMS core is an application with rigid network requirement especially for response delay and applications running on public clouds may have undecidable network overhead caused by the uncontrollable network infrastructures for application service providers, virtualizing IMS core on a private cloud environment with fully controlled network infrastructure can greatly reduce the possible negative network performance influencing factors, which can be a way to solve the problems in current IMS industry without sacrificing the performance.

Currently, there are a couple of research work on IMS core virtualization. Reference [4] is focused on the scalability. It avoids the response delay comparability between running IMS core network on virtual machines and physical machines. Reference [5] targets to building a virtual IMS test bed with acceptable performance, but it doesn't provide the definition of an acceptable performance, and its way to measure delays lacks accuracy. Reference [6] presents a feasibility research on IMS core virtualization and gives five times longer response delay when running it on a virtual machine comparing to run it on a physical machine. However, to the best of our knowledge, all of them are based on software virtualization technology, and no effort has been made to investigate the feasibility to virtualize IMS core on hardware-assisted [7,8] virtual machines, which has shown the great performance improvement on I/O and network benchmark testing. This paper argues that virtualizing IMS core network on hardware-assisted virtual machines and running it on private clouds can be feasible. A performance test is designed and conducted to show the comparable network latencies when running IMS core network on a bare box and virtual machines.

The main contributions of this paper are three folds: (1) the mainstream virtual machines and the virtualization technologies used are summarized, and the parameters affecting the performance of virtual machines are highlighted; (2) a performance testing is designed and conducted to quantitatively compare the performance that running IMS core on bare boxes and different virtual machines; and (3) running IMS core on hardware-assisted supported virtual machines have comparable network latency as running it on bare boxes.

The rest of this paper is organized as follows. The background of IMS and cloud computing is introduced in Sect. 2; the mainstream virtual machines and the virtualization technologies they used are summarized in Sect. 3. The performance testing is provided in Sect. 4 and the conclusions are drawn in Sect. 5.

2 Background

IMS and Cloud Computing are concepts belonging to the telecom and the Internet domains. In this section, the basic concepts in IMS and Cloud Computing domains are introduced.

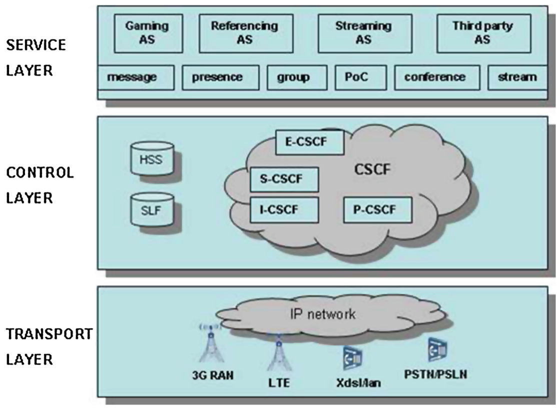

Fig. 1. IMS layered architecture.

2.1 IMS

IMS is a layered reference architecture. A simplified IMS architecture consists of three layers from the bottom up: the transport layer, the control layer, and the service layer as shown in Fig. 1 [9]. The transport layer enables the access from the different access networks and ensures the network inter-connectivity and bearer controlling with the collaboration of the IP core domains. The transport layer supports multiple access mechanisms and provides inter-connectivity to multiple networks [10]. The control layer is the IMS core network. It provides the call session controls, user managements, multimedia service resource controls and multimedia applications supporting, network inter-working. The service layer consists of various media capabilities servers and application servers, and has the ability to provide multiple multimedia services.

The main features of IMS can be summarized as the follows: (1) decoupling the access from applications to be the transport layer and the control from applications to be the control layer, such that the multimedia application services can be an independent layer on top of both; (2) the control is IP based through Session Initiation Protocol (SIP) to control call sessions, such as session creating, modifying and terminating. By working with Session Description Protocol (SDP), the media identification and negotiation is achieved; the QoS can be guaranteed for both voice and other multimedia services over IP. The major concerns of IMS include the complication of its system architecture and the possible performance bottleneck in its core and access components.

2.2 Cloud Computing

Cloud computing is defined by National Institute of Standards and Technology (NIST) as a model for service sharing. It is a layered architecture with three service delivery models and three service deployment models [11]. Hoff presents this layered architecture in [12] as shown in Fig. 2, in which the three service delivery models from the bottom up are Infrastructure-as-a-Service (IaaS), Platform-as-a-Service (PaaS), and Software-as-a-Service (SaaS). In such architecture, the lower level services form the base of the upper level services. IaaS provides the infrastructure, including network resources, servers, and storage space, in a way of on-demand usage and pay as you go hardware provisioning. PaaS facilitates the environment for developing, testing and implementing applications without having any control over the underlying operating system and hardware infrastructure. It is often termed as the development platform for SaaS. SaaS is the most commonly used service delivery model that provides software or application, on-demand, to the customer, using the Internet.

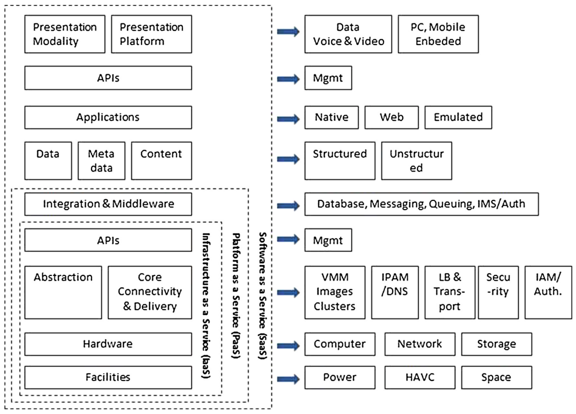

Fig. 2. Cloud layered architecture.

Based on the network infrastructure, physical location of the computing resources, the cloud deployment model can be classified to private cloud, public cloud, hybrid cloud. Private cloud is the infrastructure fully operated and used by a single enterprise with full control over the underlying hardware and software environment. Public cloud is owned by the cloud service provider, and the services are offered to and shared by public users based on the resource's usage through the Internet. Hybrid cloud is a mix of private and public cloud model. In this model, companies connect their private cloud to public clouds. It is used in the scenario where companies want to store and process the critical data in their private cloud and take the advantage of highly available and scalable resource in the public cloud as well.

Cloud computing has known the features as (1) On-demand self-service; (2) Broad network access; (3) Resource pooling; (4) Rapid elasticity; (5) Measured service. The major concerns in cloud computing consist of accurately billing and auditing, QoS monitoring, network troubleshooting and inter- operating, performance isolation, and security.

2.3 Virtualizing IMS Core

As we have mentioned in last subsection, IMS is a layered architecture, which decouples the access and control from applications and provides independent interface for the service on top of them. It is designed and supposed to greatly reduce the difficulty in new service innovation and integration. However, this layered structure itself is complicated, which causes the difficulty in system understanding and implementation and further slows down the third party multimedia service innovations and deployments in the reality. And furthermore, such layered structure in theory avoids the central bottleneck but splits the bottleneck into many points, especially for the control functions and database in its core network. In order to have guaranteed QoS for each service, the core has to have the capability to handle a big number of requests in a short time from both voice and other multimedia users. However, as a fully new IP core to support both voice and data services with guaranteed QoS, there is no existing ways to predict the usage of IP users, which can cause huge resource waste or service degrading. Even the IMS network provider can bear this huge resource waste, no matter how much resource has been reserved, there is always a day when the resource is used up with the users growing, and a system capacity expansion is unavoidable, which means the serve interruption and all the invest in the old system is wasted.

Since cloud can provide elastic environment to support resource scaling up and down without service interruption, virtualizing IMS core and running it on cloud can be a way to achieve the goal of reducing deployment difficulty and resource waste, and provide system capacity expansion without service interruption. As public cloud is traditionally considered as a platform for CPU consumed applications rather than network consumed applications, an application such as IMS core with rigid network requirement may not be fitted into its environment. However, private cloud model gives the applications on top of it the fully computing and network resource controlling, which can greatly reduce the possible factors to influence the application performance. On the other hand, telecom service providers have the experiences and requirements to fully control the network and services, and constructing a private cloud in their data center is workable for telecom service providers.

The big difference on running IMS core on bare boxes and private cloud is the computing resources on private cloud are virtualized; and virtual machines contributes most part of the performance difference. Virtual machines with different virtualization technology employed perform varied. Choosing the right virtual machines can reduce the performance influence to minimum and it is the key for virtualizing IMS core.

3 Virtualization technologies

Virtualization for x86 architecture (such as IA32 and IA64) was happening in recent a couple of years [13]. The main obstacles to virtualize x86 systems are the visibility of privileged state and lack of traps when privileged instructions run at user-level. The main approaches are used to conquer these two obstacles are Binary Translation (BT) and special privileged partition. Regarding to the various approach, current virtualization technology, which creates a virtual machine working like a real computer with an operating system to provide an isolated running environment to the applications on top of it, can be categorized into the following three types:

- Full virtualization: It almost completely simulate of the actual hardware to allow an unmodified guest OS and other applications to be run in isolation. The key technology used in full virtualization is BT, which translate the guest binary code (including all privileged instructions) to mostly user-mode instructions, such that they can be safely used on host OS with high performance. VirtualBox fills in this category.
- Para virtualization: It has not to totally simulate a hardware environment, but it offers a special API to modify the guest OS such that it can be run in this environment. Instead of using BT to handle the privileged instructions, para virtualization addresses it by creating a special privileged partition, whose privilege is lower than host OS privilege instructions but higher than the host OS user-mode instructions. The guest OSes run in this special partition and all the privilege system calls made by guest applications are also directed to run in this partition, for example, the Dom0 of Xen.
- Hardware assisted virtualization technology is a way of improving the efficiency of hardware virtualization. It involves employing specially designed CPUs and hardware components that help improve the performance of a guest environment, for example the Intel VT and AMD-V.

Table 1 lists some popular open source or free virtual machines. Kernel-based Virtual Machine (KVM) is a full virtualization solution for Linux on x86 hardware [14]. It consists of a loadable kernel module to provide the core virtualization infrastructure and a processor specific module. It has shown great I/O and network performance improving in many performance testings. VirtualBox is a virtualization product of Oracle using full virtualization technology [15]. It supports x86 and AMD64/Intel64, and easy to use with rich features. It is freely available for enterprise as well as home users. Xen is a virtualization solution using a Para virtualization technology [13]. It is first developed by the University of Cambridge Computer Laboratory and now it is open-source and maintained by Xen community.

There is no simple way to compare the performance of different virtual machines. Different test scenario and configuration makes big difference in performance testings. However, Hardware-assisted method shows a big performance improving [16–20], some best practice in tuning the virtual machine performance are widely used [21,22]:

Table 1. Virtual machine and the virtualization technoligies used

Virtual machines	Virtualization type	Hardware-assisted supported
KVM	Full	Yes
Virtualbox	Full	Yes
Xen	Para and Full	Yes

- Image type: each type of virtual machine defines its own image format to support some advanced functionality, but raw images always get the better performance and compatibility.
- Cache mode: the write cache mode affects the performance of disk I/O. Virtual machines normally support three write cache modes: none, write through, and write back. None is the mode without cache; write through is the mode that the host page cache is enabled while the virtual machine disk cache is disabled; write back is the mode that both the host page cache and virtual machine disk cache are enabled. Cache back mode has the best disk performance because the virtual machine disk cache improves the disk performance greatly.
- Device driver type: Virtual machines normally support para virtualized device drivers, these device drivers have been performance optimized. Using virtio to enable para-virtualized device drivers gets better performance.
- Network type: virtual machines normally supported two networking types: Network Address Translator (NAT) and Bridged. Bridged mode has the better performance.

4 Performance Benchmark

In order to demonstrate that hardware-assisted virtual machine can reduce the performance overhead, we design and conduct a performance test. We introduce the test bed and test scenarios used and discuss the test result in the rest of this section.

4.1 OpenIMSCore and IMS Benchmark SIPp

OpenIMSCore [23] is an open-source implementation of IMS core network. It realizes the fundamental functions of IMS core network, which includes the Call-Session Control Functions (P-CSCF, I-CSCF, S-CSCF), and a lightweight Home Subscriber Server (HSS) developed in compliance to the IMS architecture standards given by the 3GPP. P-CSCF is the first point connecting to users. It receives the service requests and transfers them to the next point. I-CSCF is the gateway point of IMS, it allocates the service requests to a particula S-CSCF, answers the routing requests, and hides the topology of the IMS domain. S-CSCF is the key control element in IMS. It controls the call sessions, user data, and user registration authentication. HSS stores all the information about the subscribers. It is a database including all the data about the basic identifications, routing information, and the QoS levels. OpenIMSCore implements

CSCFs in C and HSS on MySQL database. The IMS Bench SIPp [24] is a free open source traffic generator for the SIP protocol. It is a test tool that meets the criterion of IMS Performance Benchmark specification, ETSI TS 186.008 [25]. The whole test system consists of three modules:

- one manager instance that controls the whole benchmark running,
- a fixed number of SIPp load generators,
- a monitoring tool for the System Under Tests (SUT) to collect information about CPU load and memory consumption.

The manager instance uses a xml file to predefine the test configuration, which consists of the IP of each load generator, the IP and Port number of the SUT, the distribution of scenarios, and the number of Inadequately handled Scenario(HIS). The test system also provides some Perl scripts to add new user to HSS, to report the test data, and to analysis the test data.

4.2 Test Bed

We choose OpenIMSCore as IMS core network to test the performance, especially the delay when it runs on bare mental and multiple virtual machines. IMS benchmark SIPp is used to generate the test traffic. Our test bed consists of two PCs, one is the SUT, which runs the OpenIMSCore and the SIPp monitoring module; the other is the test system, which runs the SIPp with one manager instance and one load generator. The SUT and the test system is connected through a Syslink compact wireless broadband router with 4 Ethernet ports. Table 2 lists the test bed hardware configurations.

We run P-CSCF, I-CSCF, S-CSCF, and HSS modules of OpenIMSCore in a bare mental as the base line. Two test cases are considered: running P-CSCF, I-CSCF, S-CSCF, and HSS modules of OpenIMSCore in one virtual machine instance and in 4 virtual machines. Each virtual machine in our test has 1 CPU, 1 G memory, 10 G raw image file, the write back cache, the bridged network, the virtio device driver, and hardware-assisted enabled. We use this configuration for each virtual machine instance under test.

4.3 Test Result and Discussion

In this sub section, we analyze three typical scenarios: calling, messaging, and registering. We present the test results and discuss the possible reasons to cause them.

Calling Scenario: A full IMS calling scenario consists of three sub-scenarios: session setup, invite arrive, and session release. The session setup is the period from a caller sending a SIP INVITE request to a callee receiving the corresponding ACK message; the invite arrive is the period from a caller sending a SIP INVITE request to a callee receiving a SIP INVITE request, and the session release is the period from the first BYE to the corresponding 200 OK. We test the delays under different test load for each sub-scenario on each type of the virtual machines. The test

Table 2. Performance benchmark test bed

Machine	CPU	Memory	Hard disk	Network	OS
SUT	Intel Core i7-2600, 3.4 GHz*8	8 G	400 G	1000 Mb/s	Ubunbu 12.04 LTS
Test machine	Intel Core i3-M370, 2.4 GHz	4 G	200G	1000 Mb/s	CentOS 6

load is expressed in Scenarios Attempts Per Second (SAPS), which are 10, 15, 20 in our testing. SIPp calculates the number of delay in mean, minimum, maximum, percentile 50, percentile 90, percentile 95, and percentile 99. We choose the data in percentile 90 and draw them in Figs. 3, 4, and 5 for the delays in each sub-scenario. It can be observed that the 90 percentile of delay is in the range of 4 to 9 ms during the session setup sub scenario, 1 to 3.5 ms during the invite arrive sub scenario, and 2 to 5 ms during the session release sub scenario, and they does not increase with the increase of the number of SAPS during the testing. The delay in 4 VMs is longer than the one in 1 VM, because the Round Traffic Time(RTT) between virtual machines is longer than the communication time between the applications inside a virtual machine instance. Among the three types of virtual machines, Xen with hardware-assisted supported has the shortest delay, which is even shorter than the delay on physical machines. The most possible reason for it is the Xen virtual machine creates a performance isolation environment for the OpenIMSCore running on top it, whereas the performance of OpenIMSCore running on the physical machine can be affected by the other applications sharing the same running environment (Fig. 7).

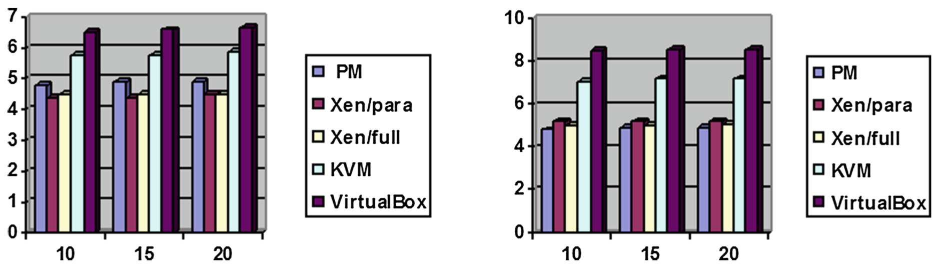

Fig. 3. Delay in call session setup (left: running on 1 VM, right: running on 4 VMs).

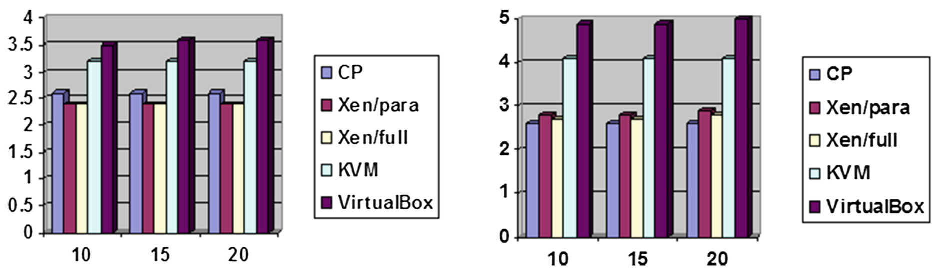

Fig. 4. Delay in call invite arrivel (left: running on 1 VM, right: running on 4 VMs)

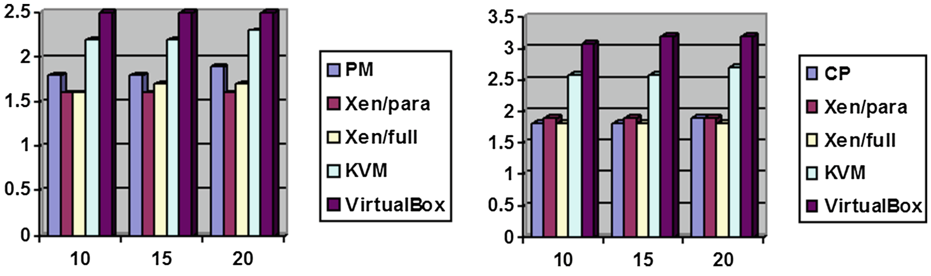

Fig. 5. Delay in call session release (left: running on 1 VM, right: running on 4 VMs)

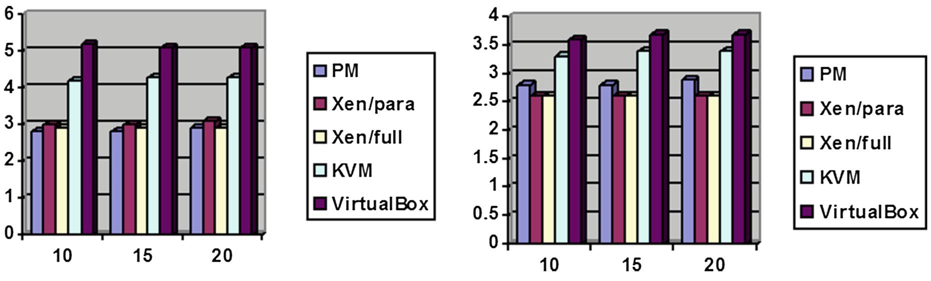

Fig. 6. Delay in message scenario (left: running on 1 VM, right: running on 4 VMs)

Messaging Scenario: IMS benchmark specification includes a message scenario. The delay is the time in milliseconds between the sending of a message and the corresponding 200 OK. We test this scenario under different load of 10, 15, 20 SAPS for each type of virtual machines. Figure 6 shows the test results in 90 percentile. It can be observed that the 90 percentile of delay is in the range of 2 to 6 ms in this scenario, and it does not increase with the increase of the number of SAPS. The delay in 4 VMs is longer than the one in 1 VM in terms of longer RTT between virtual machines than application in the same virtual machine. Xen has the shortest delay among the three types of virtual machines. The delay for all the OpenIMSCore modules running in a Xen virtual machine instance has even shorter delay than the one running in the physical machine in terms of the possible reason mentioned in last subsection.

Registration Scenario: A full IMS registration scenario consists of 2 sub-scenarios: between the first SIP Register request and the 401 Unauthorized response and between the second Register message and the corresponding 200 OK message. We test the delay under different test load of 10, 15, 20 SAPs for each sub-scenario on each type of virtual machine. Figures 5 and 6 show the test results in 90 percentile. It can be observed that the 90 percentile of delay does not increase with the increase of the number of SAPS during the testing. For each type of virtual machine, the delay in 4 VMs is longer than in 1 VM in terms of the longer RTT between virtual machines than applications within the same

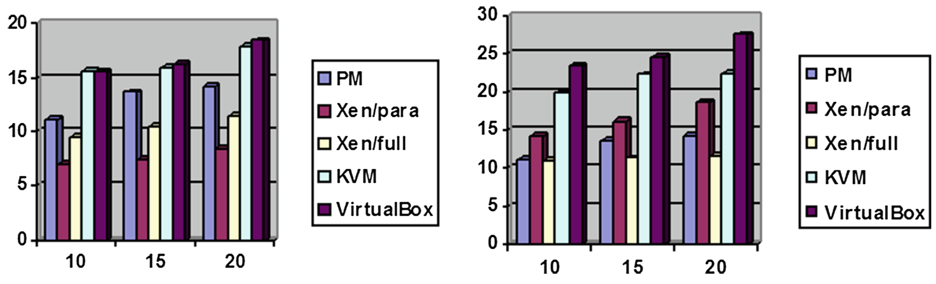

Fig. 7. Delay in first registration (left: running on 1 VM; right: running on 4 VMS)

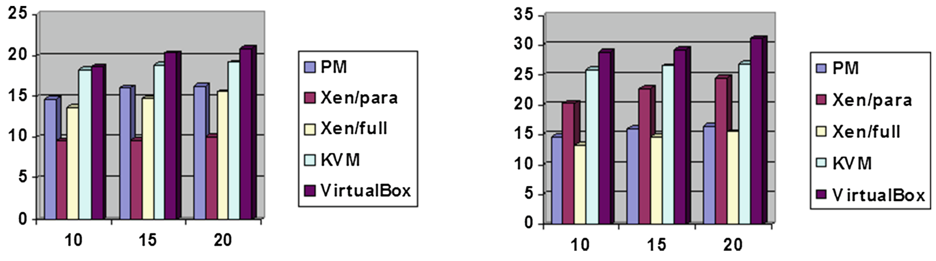

Fig. 8. Delay in second registration (left: running on 1 VM; right: running on 4 VMS).

virtual machine. The delay on Xen is the shortest among the three types of virtual machines. The same, the delay of OpenIMSCore running in one Xen virtual machine instance is shorter than the one running in the physical machine, the possible reason has been mentioned in last subsection.

Discussions: We conduct an IMS core network performance test by virtualizing OpenIMSCore on KVM, Xen, and VirtualBox virtual machines with Hardware-assisted virtualization technology supported. We use IMS Benchmark SIPp as test tool to compare the performance of running OpenIMSCore on a bare box and the virtual machines. The results show that the response delay of running on virtual machines is comparable to that running on bare boxes, which greatly improves the delay (at least 10 times longer) provided by current researches. Hardware-assisted virtualization technology and current best practice in virtual machine performance tuning can really improve the network performance. Under our test scenarios, the Xen presents the shortest delay among all three types of virtual machines tested in either para or full virtualization mode Fig. 8.

5 Conclusions

In order to solve the problem that the complicated architecture, the huge early invest, and the big possible resource waste in IMS network deployment has slowed down the IMS network deployment and its service innovation, this paper firstly introduces the basic concepts in IMS and cloud computing, then designs and conducts a performance testing to test the response delays of OpenIMSCore under different loads and configurations. The test bed, the test tool, and test scenarios are provided in detail, and the results are advanced discussed. The test results show that running OpenIMSCore on virtual machines with Hardware-assisted virtualization technology enabled can greatly reduce the response delays, and all the response delays running on all the virtual machine types used are comparable to those on bare boxes. Virtualize IMS core network and running it on a private cloud is feasible.

References

1. Poikselk, M., Mayer, G.: The IMS: IP Multimedia Concepts and Services. John Wiley and Sons, New York (2009)
2. Rings, T., Caryer, G., Gallop, J., et al.: Grid and cloud computing: opportunities for integration with the next generation network. J. Grid Comput. **7**(3), 375–393 (2009)
3. Nokia Siemens Network. Cloud computing-business boost for communications industry (2011). http://bit.ly/nVlxRI/
4. Umair, M.: Performance Evaluation and Elastic Scaling of an IP Multimedia Subsystem Implemented in a Cloud (2013)
5. Corte, G.D., et al.: An IMS-based virtualized architecture: performance analysis. In: Proceedings of the 11th WSEAS International Conference on Mathematical Methods and Computational Techniques in Electrical Engineering. World Scientific and Engineering Academy and Society (WSEAS) (2009)
6. Chuan, S., Xiaoyong, H., Xiaodong, D.: Feasibility of the virtualization based on OpenIMSCore. In: Yang, G. (ed.) Proceedings of the ICCEAE2012. CCIS, vol. 181, pp. 539–544. Springer, Heidelberg (2013)
7. INTEL CORPORATION. Intel virtualization technology specification for the IA-32 Intel architecture, April 2005
8. AMD. AMD64 Virtualization Codenamed 'Pacifica' Technology: Secure Virtual Machine Architecture Reference Manual, May 2005
9. 3GPP, IP Multimedia Subsystem (IMS), TS 23.228, Release 6 (2004)
10. 3GPP, IP Multimedia Subsystem (IMS), TS 23.228, Release 7 (2007)
11. Mell, P., Grance, T.: The NIST definition of Cloud Computing. Special Publication Draft-800-145 (2011)
12. Hoff: The Frogs Who Desired a King: A Virtualization and Cloud Computing Security Fable Set To Interpretive Dance (2009). http://www.rationalsurvivability.com/presentations/Frogs.pdf
13. Barham, P., et al.: Xen and the art of virtualization. ACM SIGOPS Oper. Syst. Rev. **37**(5), 164–177 (2003)
14. Linux: 2.6.20 Kernel release notes. Virtualization support through KVM (2007). http://kernelnewbies.org
15. Oracle Corporation. Oracle and Virtualization (2010). http://www.oracle.com/us/technologies/virtualization/index.html
16. Deshane, T., et al.: Quantitative comparison of Xen and KVM. Xen Summit, pp. 1–2, Boston, MA, USA (2008)

17. Danti, G.: Vmware vs Virtualbox vs KVM vs XEN: virtual machines performance comparison (2010). http://www.ilsistemista.net/index.php/virtualization/1-virtual-machines-performance-comparison.html
18. Chen, W., et al.: A novel hardware assisted full virtualization technique. In: The 9th International Conference for. IEEE (2008)
19. Adams, K., Agesen, O.: A comparison of software and hardware techniques for x86 virtualization. ACM SIGOPS Operating Systems Review **40**(5), 2–13 (2006). ACM
20. Menon, A., et al.: Diagnosing performance overheads in the xen virtual machine environment. In: Proceedings of the 1st ACM/USENIX International Conference on Virtual Execution Environments. ACM (2005)
21. IBM. Kernel Virtual Machine (KVM): Tuning KVM for performance. http://www-01.ibm.com/support/knowledgecenter/linuxonibm/liaat/liaattuning.pdf
22. VMware. Best Practices for performance Tuning of Latency Sensitive. http://www.vmware.com/files/pdf/techpaper/VMW-Tuning-Latency-Sensitive-Workloads.pdf
23. OpenIMSCore Project official website. http://www.openimscore.org
24. IMS bench SIPp official website. http://sipp.sourceforge.net/imsbench/intro.html
25. Institute, E.T.S.: Telecommunications and Internet converged Services and Protocols for Advanced Networking (TISPAN); IMS/NGN Performance Benchmark. ETSI TS **186**, 1–8 (2007)