International Journal of Grid and Distributed Computing
Vol. 9, No. 8, (2016), pp.249-260
http://dx.doi.org/10.14257/ijgdc.2016.9.8.22

CloudEVBench – Virtualization Technology Efficiency Testing Tool for the Distributed Infrastructures

Jan Fesl[1,2,*], Jiří Cehák[1], Marie Doležalová[1] and Jan Janeček[2]

[1]*University of South Bohemia, Faculty of Science, Institute of Applied Informatics, Branišovská 31, České Budějovice, 370 01,* Czech Republic
[2]*Czech Technical University in Prague, Faculty of Electrical Engeneering, Department of Computer Science, Karlovo Náměstí 13, Prague, 121 35,* Czech Republic
jfesl@prf.jcu.cz, cehakj00@prf.jcu.cz, dolezm01@prf.jcu.cz, janecek@fit.cvut.cz

Abstract

The virtualized systems are today a very popular topic and their using plays the great role in current datacenters. The virtualization efficiency is a very important aspect in the real system deployment. Some studies have been published about this topic [1], mainly are based on various benchmarking techniques and are integrated into the specialized testing tools. Such a benchmark tool, which is able to simulate the behavior of a real computing system under the stress of different virtualization configurations, can e.g. well answer the question how many virtual machines could be simultaneously executed on it and how big the virtualization overhead is [2]. We developed and applied a new benchmark tool, which is able to measure the virtualization efficiency and overhead in the virtualization environment.

Keywords*:* virtualization; benchmarking; cloud; efficiency; distributed; hardware assisted virtualization

1. Virtual Technology Efficiency Testing

The main aspect in using and deployment of the virtualization technology is its efficiency. The substance of this technology is as follows. The resources of one powerful physical machine can be shared between many simultaneously executed virtual computers. The efficiency of the virtualization technology is an important factor, which determines the possible count of the simultaneously executable machines.

Another aspect, which also fundamentally affects the efficiency, is the type of used virtualization technique. The main virtualization types are the full virtualization, hardware assisted virtualization, paravirtualization, partial virtualization and emulation [3]. Today is predominantly used the full virtualization technology. The term "virtualization" will always mean the hardware assisted virtualization technology in the following text.

1.1. Distributed System for Virtualization

The typical distributed system for virtualization consists of three main parts – the management node(s), computing node(s) and shared data storage node(s). This architecture is graphically represented by figure 1. The management node creates, executes, stops and migrates the virtual machines. The virtual disk drives are present on the shared network data storage. The count of physical computing nodes can also change during the system life time.

* Corresponding Author

ISSN: 2005-4262 IJGDC

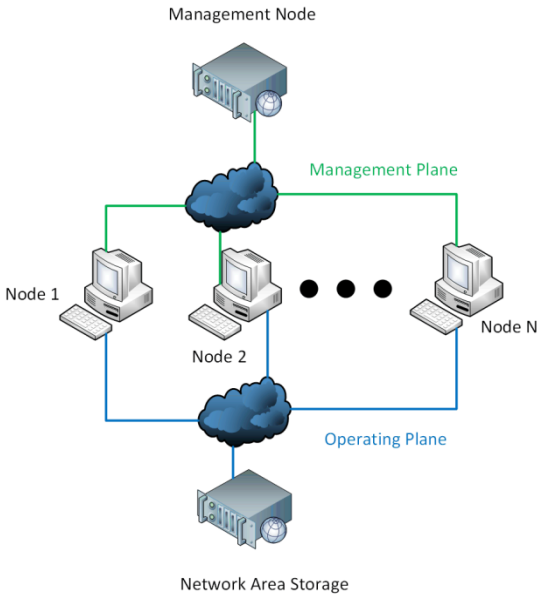

Figure 1. Distributed Virtualization Infrastructure

2. Related Work Overview

The common benchmarking techniques for standalone real computers have been developed many years ago and are commonly used in many benchmarking tools. The benchmarking tools, which are especially dedicated to the virtualization technology testing, are used much less. The most famous commercial current benchmarking tools are SPECvirt_sc®2013 [4] or VMmark [5]. For academic or scientific purposes have been developed tools such as vCosolidate [6], XenMon[7], XenoProf [8] and Perfctr-Xen [9] which are assigned only for the Xen hypervisor virtualization efficiency measurement [10]. Other tools such as GrenchMark [11] or C-Meter [12] have been introduced for the grid/cloud architectures, but none of these tools is able to quantify the virtualization efficiency. The new complex tool for virtualization technology testing is called Virt-B [13]. This tool measures the virtualization efficiency using a new approach – the three layer benchmarking methodology. The biggest problem of this solution lies in its implementation. The framework is written in the Java and for running uses the Java Virtual Machine (JVM) sandbox. The implementation of JVM is operating system dependent and significantly influences the measurements. Except the above mentioned tools have been published many methodologies [14] [15], strategies [16] [17] or reviews for better virtualization overhead measurement understanding [18] [19].

2.1. New ideas in Proposed Solution

Our solution has been developed for the maximal measurement precision. There are many factors, which can fundamentally affect the measurement accuracy. The proposed solution tries these factors to eliminate, all concrete negative aspects are presented in the section 3.4. The next advantage stands in the automated distributed system architecture, which allows finding out the virtualization efficiency through all virtualization system nodes. The solution has been optimized for the Linux and Windows operating system. The new command line style benchmarking tool Evbench [20] have been developed, all tests in this tool have been designed for the maximal measurement accuracy. This tool it is possible to download from its GitHub repository.

3. Virtualization Efficiency Modeling

The simple, but realistic model of a common computer, which is valid for the performance modeling, is as follows. The model consists of one or more single/multi core CPU units, operating memory ram, input/output devices (hard drive, CD/DVD drive, etc.) and the network card. The system performance (*P*) can be expressed as the 4 dimensional vector of values, which represent the performances for the specific operations (CPU, RAM, I/O and NET).

$$P = (P_{CPU}, P_{RAM}, P_{I/O}, P_{NET}) \quad (1)$$

3.1. Efficiency of the Virtualization Technology (E_{VT})

E_{VT} can be counted as the ratio of the performance of the virtual computer (P_{VPC}) and the real computer (P_{RPC}).

$$E_{VT} = P_{VPC}/P_{RPC} \quad (2)$$

3.2. Overhead of the Virtualization Technology (O_{VT})

O_{VT} is the part of the real computer performance, which is necessary for running of the virtual computer and can be counted according to this formula.

$$O_{VT} = 1 - E_{VT} = (P_{RPC} - P_{VPC}) / P_{RPC} \quad (3)$$

The performance of the real (P_{RPC}) and virtual computer (P_{VPC})

$$P_{RPC} = (P_{CPU(RPC)}, P_{RAM(RPC)}, P_{I/O(RPC)}, P_{NET(RPC)}) \quad (4)$$

$$P_{VPC} = (P_{CPU(VPC)}, P_{RAM(VPC)}, P_{I/O(VPC)}, P_{NET(VPC)}) \quad (5)$$

3.3. Performance of the Distributed Virtualization System

The architecture of the distributed system (figure 1) consists of *N* computing nodes. The aggregate performance of such distributed system (P_{DS}) can be counted according the following equation. P_i is the performance of the real single node.

$$P_{DS} = \sum_{i=1}^{N} P_i \quad (6)$$

Because all computing nodes of the distributed system are physical, the equation can be rewritten into the following form. $P(RPC)_i$ is the performance of the single physical computer. $P_{DS(RPC)}$ is the total performance of the distributed system, which consists from the physical computing nodes only.

$$P_{DS(RPC)} = \sum_{i=1}^{N} P(RPC)_i \quad (7)$$

The set of the virtual computers, which can be executed on the distributed virtualization system, contains maximally *K* virtual computers. All resources of the real computing nodes of the distributed system are assigned to the virtual computers. $P(VPC)_i$ is the performance of the single virtual computer. The performance of whole virtual computers $P_{DS(VPC)}$ can be counted as follows.

$$P_{DS(VPC)} = \sum_{i=1}^{K} P(VPC)_i \quad (8)$$

The efficiency of the distributed virtualization system can be counted as the ratio of the performance of its real and virtualized infrastructure.

$$E_{DS} = P_{DS(VPC)}/P_{DS(RPC)} = \sum_{j=1}^{K} P(VPC)_j / \sum_{i=1}^{N} P(RPC)_i \quad (9)$$

3.4. Accurate Virtualization Measurement Aspects

The accomplishment of the following aspects is necessary for the credible measurement results. These aspects were observed within the development.

The running benchmark program sameness (on the instruction level) in the native and virtualized environment

This requirement seems to be a bit strange, but it is very important. The power efficiency of the compiled program can differ by using of the various compilers. If the application running in the native environment has been compiled by the other compiler than the application in the virtualized environment, there can arise the power inaccuracy about 10-40 % depending on the used compiler, which is caused by the compilers code generator settings or quality. The difference was observed between compilers from different producers and between different versions of the same compiler as well. The using of the same binary code of the benchmark further means that the results (their absolute values) obtained from the Windows can't be without normalization directly compared to the results obtained from the Linux.

The next important factor is the same version of all additional system libraries, which are dynamically loaded by the benchmark software, because their implementation can be different too. This fact leads to using of the exactly same system for the virtualized and native environment (with the same version of the operating system kernel and libraries). This can cause a performance difference between 5-10 % depending on the operating system.

The last most important factor stands in the implementation of the specific functions and libraries, which are used by the benchmarking software compilation. E.g. the implementation of the common used function memcpy() strongly differs in Linux glibc library in comparison to the implementation in MS Visual Studios. This factor may cause the differences between measurements in hundreds percent.

Hardware computing power sameness in native and virtualized environment

This requirement is logical, but it is not every time met as well. The problem stands in the architecture of the modern multi-core CPUs. The main measurement inaccuracy can be caused by the "Turbo boost", which allows short-term power increase of some CPU cores. If the measurement, which is performed in the virtual environment, runs in fact on the CPU core(s) with the increased computing power, the power of the virtual computer may seem greater than the physical one. This factor causes the measurement inaccuracy above 50 % (depends on the CPU type). The solution is to switch-off the turbo boost feature during the measurement phase.

System resources utilization

If the measured system doesn't have enough free computing resources (free CPU cores, memory or not saturated I/O devices) the other background processes can affect the benchmark performance. The inaccuracy depends on the concrete state.

Virtualizer support by the virtualized operating system kernel

The performance of the virtualized operating system is affected by its virtualizer support. This aspect doesn't directly cause the measurement inaccuracy, but the virtualization efficiency, yes.

Native execution of the compiled benchmark

This fact means that the binary form of the executed benchmark must be directly executed by the operating system without compatibility emulation. This problem can arise for example when the Win32 program is emulated in the Win64 environment, because its run is emulated.

3.5. System Utilization Modeling

The average system utilization is also the important factor in the virtualized system deployment, because the partial system utilization allows the multiple

resources sharing among virtual computers. The lower system utilization means the higher available amount of the deployable virtual computers. The whole system utilization is given by the partial utilization of the CPU, RAM, I/O bandwidth and network bandwidth.

System Snapshot (SS) Parameters

SS = {CPU utilization [%], Memory usage [%], Hard drive usage, Network usage}

CPU utilization [%] - means the time of the total time in which is the CPU working; *Memory usage [%]* - means the amount of the operating memory, which is used as the data storage. The relative usage is recalculated on maximal system memory size; *Hard drive usage* - means the utilization of the specific disk drive bandwidth, the metric is the amount of the read/write data. The maximal value of the data throughput depends on the concrete drive; *Network usage* - means the utilization of the specific network bandwidth. The relative usage is recalculated on maximal link speed of the network card.

The above described parameters are measured periodically and further statistically interpreted by various statistical methods.

3.6. CloudEVBench System

The benchmark system is determined for the performance testing of the distributed virtualization system, which architecture was introduced by figure 1. The testing tool divides the testing process into two parts. The first part is the performance testing of the real infrastructure and the second part is the performance testing of the virtualized infrastructure. All tests and evaluations are executed automatically. The main parts of the system are:

1. *benchmarkServer* – the tool, which is installed on the management node. It has the capability of executing scripts remotely on all nodes. It enables complete central control of the tests preparation, operation and of the measured data collection for different tests configurations. The implementation of this tool is primary for the Windows platform only, but it is fully functional on Linux using the Wine tool [20] as well.
2. *benchmarkClient* – this tool is installed on every physical node of the distributed virtualization system and on every virtualized computer. The implementation of this tool has been realized for the Linux and Windows and this tool communicates with the benchmarkServer by using the standard BSD sockets.
3. *resultsAnalyzer* – the statistical tool, which evaluates the data collected by the benchmarkServer.

Used Tool for Benchmarking

The in-house developed tool Evbench was used for the results measurement. Evbench has the code optimization for the Linux (g++) and Windows compilers (MS Visual Studio). The development of these tools was necessary for the exact knowledge of the tests principles and their evaluation. The next requirement was that the benchmarking tool must be executable from the command line and its results must be machine workable.

CloudEVBench Network Topology

The benchmarking process is carried out as follows. The first tool, which is started, is the benchmarkServer. This tool runs on the management node and sends

the multicast datagrams with settings for benchmarkClient connected to the network. The IP address of the management node can be assigned dynamically and all benchmarkClient must be able to obtain the correct settings. The situation is depicted below in figure 2. The communication between the client and the management node is of two types. The first communication type provides the client parameters receiving from the management node. The second provides the standard communication from the client to the management node. The first type is realized by the multicast packet sending and receiving, the second type is the simple unicast TCP connection between the client and the management node.

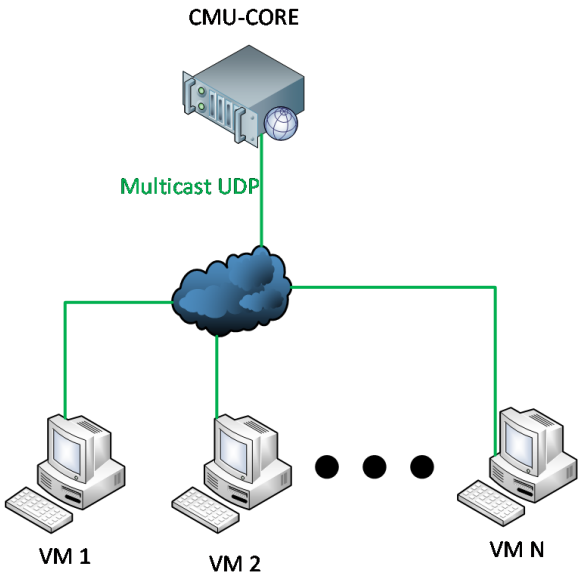

Figure 2. Tested System Scheme (CMU)

The testing process starts after the connection of all clients to the management node is established. Then the management node sends the testing commands for the given measurement case simultaneously to all connected clients. The received commands are immediately executed by the client operating system.

Testing Scenarios

The CloudEVBench tests the virtualization infrastructure according to the following scheme. The physical infrastructure is tested as first – for obtaining the reference values, the CloudEVBenchClient must be installed on all physical nodes of the infrastructure. The configuration of the whole system must be stable and mustn't be modified during the test.

The virtual infrastructure is tested in various system configurations (VPCC), which are defined as follows.

VPCC = {cpuCounts, ramSize, diskDriveSize, netCardEnabled, Operating System}

cpuCounts - the count of the physical CPU cores, which are assigned to the specific virtual machine; *ramSize* - the size of the operating memory, which is assigned to the specific virtual computer; *diskDriveSize* – the hard drive capacity on the shared network area storage, which is assigned to the specific virtual computer; *Operating system (OS)*

Operating system, which is used for the testing, can be determined by the following attributes.

OS = {Platform, architecture, used file system}

Platform – means the used operating system – Windows, Linux, BSD, IOS or Android; *Architecture* – only 64 bit system architecture; *Used File System* – is platform dependent, can influence the efficiency of the I/O disk operations.

Real Testing Parameters Used for the Measurements

The CloudEVBench tool was deployed on the CMU computing system. This system consists of four nodes now. The computing nodes contains 2x Intel Xeon 2660v2 CPUs, 128 GBs operating memory, 10 Gb Ethernet card and 2x raid mirror connected 1TB hard disks. These nodes physically execute and emulate the virtual computers. The network area storage node is based on the Intel i5-4440 CPU, has installed 32 GBs operating memory, 18TBs hardware raid 5 hard drive (6x 3 GBs SATA III Seagate Constellation, NAS type drives). This node stores all hard drives of the virtualized computers. The management node is an older computer, it contains only 1x Intel Xeon e5405 CPU and 6GBs operating memory, the operating system is stored on the raid mirror hard drive. The internal naming of this node is the CMU-Core, The network architecture of this system is divided into the management plane and the operating plane. The management plane is used for the infrastructure administration, the operating plane is strictly used only for mapping of the virtual computers hard drives mapping and migration of the virtual machines. The situation is depicted in the figure 1. The used virtualization technology was the Microsoft Hyper-V running on the Microsoft Windows Server 2012 R2. The real configurations of the virtualized computers were as follows.

The real efficiency testing was targeted to the CPU, RAM, hard drive throughput and the network interface throughput.

4. Measurements and Results

4.1. CPU Virtualization Efficiency and Overhead Measurement

The basic idea of this test is the CPU utilization. The problem, which the CPU solves, is the prime searching in the given interval. This task can be divided into many computing threads. The metric for evaluation of this test was the time, which was necessary for this task completing. The testing process was provided by the Sysbench tool. The overhead value was calculated according to the equation (3), the reference time value was the time of the task, which was executed with the same parameters on the physical system. The results are graphically depicted in figure 3. The average measured overhead value was about 1.1 %, which means, that the virtualization efficiency was 98.9 %. This result can be achieved on the Intel platform only by using the VT-x technology, which rapidly increases the virtualization efficiency.

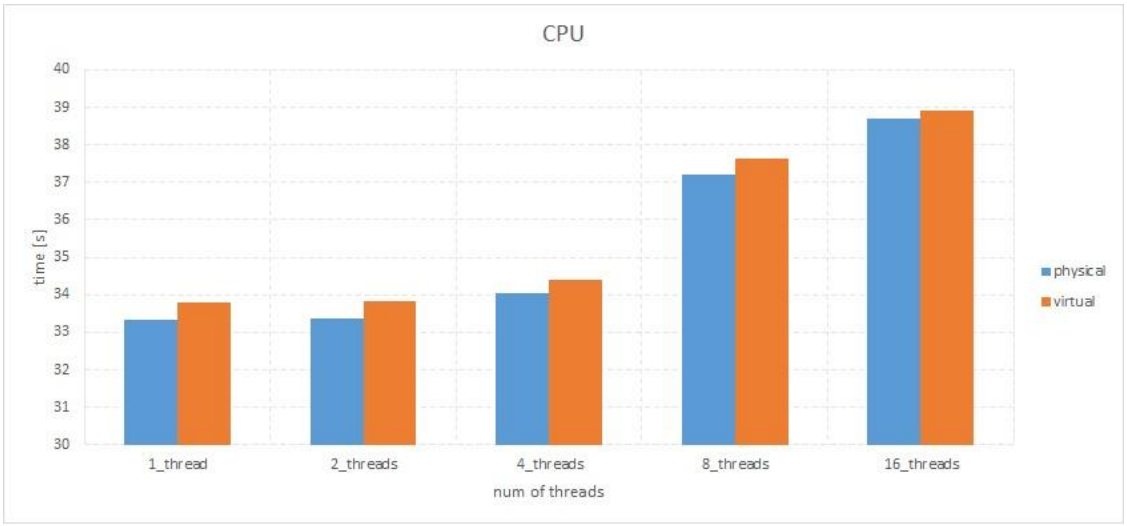

Figure 3. CPU Virtualization Efficiency Testing (VT-x enabled)

4.2. Operating Memory Virtualization Efficiency Measurement

This test was performed by reading/writing the various large data blocks from/into the operating memory. The testing proved, that the maximal throughput of the data is possible to achieve by increasing the higher count of parallel working threads. The evaluation metric is the memory bandwidth utilization in MB/s. The average measured overhead value was about 2.04 %, which means the 97.96 % virtualization efficiency. This result is achievable by using the Intel VT-d technology, which allows the direct access of the virtual machine to the physical operating memory. The results are graphically depicted in figure 4.

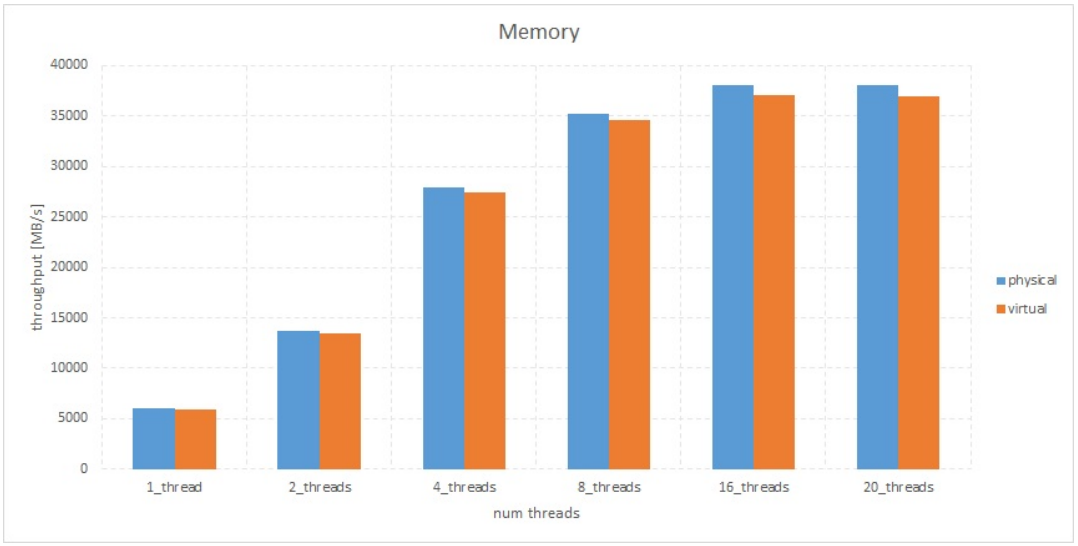

Figure 4. Operating Memory Virtualization Efficiency Testing (VT-d enabled)

4.3. I/O virtualization Efficiency and Overhead Measurement

The I/O testing rests in the measurement of the data reading/writing from/to the hard drive. The metric is the reading/writing throughput. The measurement was targeted on the single thread sequential reading/ writing from/to the physical hard disk with the SATA III connection. The measured overhead value was about 4.42 %, which means the virtualization efficiency better than 95 %. The results are graphically depicted in figure 5.

Figure 5. I/O Devices (HDD) Virtualization Efficiency Testing

4.5. Network Virtualization Efficiency and Overhead Measurement

The network testing was realized as the bandwidth measurement between the local and remote network side, which were connected via the 10Gbit/s Ethernet switch. The remote side was the hardware router without virtualization, the local side were first the native computer a next the virtual computer. The maximal bandwidth saturation is possible by using the simultaneous TCP connections only. The measured average overhead was only 1.29 %, which means the 98.71 % virtualization efficiency. The results are graphically depicted in figure 6.

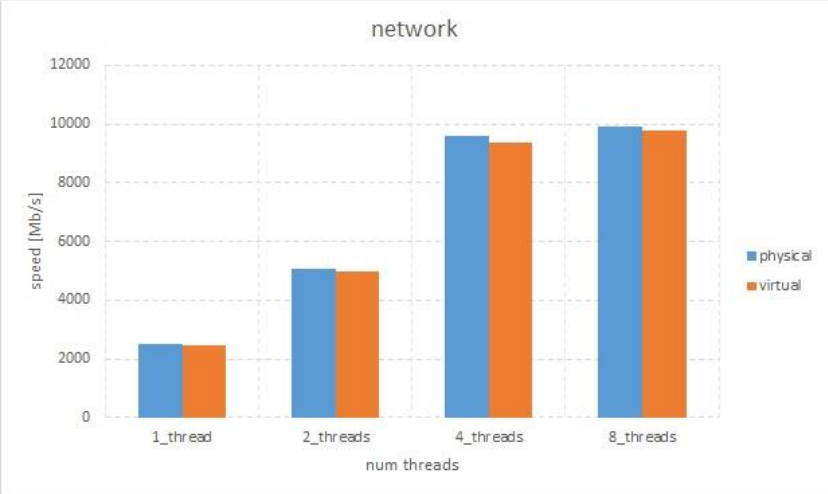

Figure 6. Virtualized Network Efficiency Testing (10Gbit/s Ethernet)

5. Conclusions

This work contains the description, development and deployment of the CloudEVBench tool, including the new command line benchmarking tool Evbench, which is dedicated to the accurate virtualization efficiency measurement. The aspects for the accurate virtualization overhead measurements have been defined within the development. The non-fulfillment of these aspects causes the measurement inaccuracy, which is higher than the virtualization efficiency. The testing was performed for the Linux and Windows operating system and the relative overhead results are very similar for both environments.

The obtained results proved, that the performance of the virtualized system can be similar (maximal measured overhead was 4.4 % only) to the physical system, which manages the virtualization process. The average system utilization was further measured for the various system deployments. The connection of the system utilization and the virtualization efficiency has the fatal impact to the real count of the simultaneously executable virtual computers. All results contained in this work were measured by the real deployment and proved the high abilities of the advanced virtualization technologies.

Acknowledgement

The authors would like to acknowledge to the University of South Bohemia, Faculty of Science for providing the technical background, which was necessary for this paper creation. All authors were financed by the internal grants of the University of South Bohemia.

References

[1] R. Goldberg, "Survey of virtual machine research", Computer, 1970, vol. 7, no. 6, pp. 34-45.
[2] K. Ye, Performance Combinative Evaluation from Single Virtual Machine to Multiple Virtual Machine Systems, International Journal Numerical Analysis & Modeling, Vol. 9, No. 2, pp. 351-370, 2012.
[3] Everything You Need to Know About the Intel Virtualization Technology, hardware secrets, 2012, available online, http://www.hardwaresecrets.com/printpage/Everything-You-Need-to-Know-About-the-Intel-Virtualization-Technology/263.
[4] Standard Performance Evaluation Corporation, PECvirt_sc®2013, available online, https://www.spec.org/virt_sc2013/, 2015.
[5] VMWare Inc.,VMmark, available online, http://www.vmware.com/a/vmmark/, 2015.

[6] Apparao, Padma; Iyer, Ravi; Newell, Don; Towards Modeling & Analysis of Consolidated CMP Servers, ACM SIGARCH Computer Architecture News, Volume 36 Issue 2, pp. 38-45, 2008.
[7] Gupta, Diwaker; Gardner, Rob; Cherkasova, Ludmila, XenMon: QoS Monitoring and Performance Profiling Tool, available online, the technical report of the Hewlet Packard company, available online, http://www.hpl.hp.com/techreports/2005/HPL-2005-187.html, 2005.
[8] Aravind Menon, Jose Renato Santos, Yoshio Turner,Xenoprof - System-wide profiler for Xen VM, developed by the Hewlet Packard company, available online, http://xenoprof.sourceforge.net/, 2005.
[9] Nikolaev, R. Back, G.Perfctr-Xen: A Framework for Performance Counter Virtualization, Acm Sigplan Notices, Vol. 46, Issue 7, pp. 15-25, 2010.
[10] Xen Project, available online, http://www.xenproject.org/developers/teams/hypervisor.html, 2015.
[11] A. Iosup, R. Prodan, and D. Epema, IaaS Cloud Benchmarking: Approaches, Challenges, and Experience, Proc. Int'l Workshop on Hot Topics in Cloud Services, pp. 1-2, 2012.
[12] N. Yigitbasi et al., C-Meter: A Framework for Performance Analysis of Computing Clouds, Proc. 9th IEEE/ACM Int'l Symp. Cluster Computing and the Grid, pp. 472–477 2009.
[13] Ye, K. J., Wu, Z. H., Zhou, B. B., Jiang, X. H., Wang, C. Zomaya, A. Y., Virt-B: Toward Performance Benchmarking of Virtual Machine Systems, IEEE Internet Computing, Vol. 18, Issue 3, pp.64-72, 2014.
[14] Xu, X. G., Zhou, F., Wan, J., Jiang, Y. C., Quantifying Performance Properties of Virtual Machine, Issue 2008: International Symposium on Information Science and Engineering, Vol 1, 2008.
[15] Xiao, P., Hu, Z. G., Liu, D. B., Yan, G. F., Qu, X. L., Virtual machine power measuring technique with bounded error in cloud environments, Journal of Network and Computer Application, Vol. 36, Issue 2, pp. 18-28, 2013.
[16] Wen, C. J., Xiang, L., Yang, Y., Ni, F., Mu, Y. F., System Power Model and Virtual Machine Power, Metering for Cloud Computing Pricing, 2013 Third International Conference on Intelligent System Design and Engineering Applications (Isdea), pp. 1379-1382, 2013.
[17] Shao, Z. Y.Jin, H. Li, Y.Huang, J. XenMVM: Exploring Potential Performance of Virtualized Multicore Systems, Information an International Interdisciplinary Journal, Vol. 14. Issue 7, p. 2315-2326, 2011.
[18] Du, J. Q., Sehrawat, N., Zwaenepoel, W,Performance Profiling of Virtual Machines, Acm Sigplan Notices, Vol. 46, Issue 7, pp. 3-14, 2011.
[19] O. Tickoo, Modeling Virtual Machine Performance: Challenges and Approaches, AC Sigmetrics Performance Evaluation Rev., Vol. 37, no. 3, pp. 55-60, 2010.
[20] Fesl J., Exact Vitualization BenchMark (evbench), available online, http://https://github.com/UniThinkTeam/EVBench

Authors

Jan Fesl, He was born in Ceske Budejovice, Czech Republic in 1982. His M.Sc. Diploma received in Computer Science in 2007 at the Czech Technical University of Prague, Czech Republic. Currently, he is an assistant professor at the University of South Bohemia and he is a Ph.D. student at the Czech Technical University of Prague. His current research is focused on computer networks and distributed computing systems.

Jiří Cehák, He was born in Ceske Budejovice, Czech Republic in 1990. He is the M.Sc. student at the University of South Bohemia. Main areas of his research are bioinformatics and computer graphics.

Marie Doležalová, He was born in Hradec Králové, Czech Republic in 1981. She is the M.Sc. student at the University of South Bohemia. Main areas of her research are bioinformatics and computer graphics.

Jan Janeček, He is an associate professor at the Czech Technical University in Prague, head of the network research group. His current research is focused on computer networks and distributed computing.