International Journal of Engineering & Technology, 7 (4) (2018) 2391-2396

International Journal of Engineering & Technology

Website: www.sciencepubco.com/index.php/IJET
doi: 10.14419/ijet.v7i4.13236
Research paper

A novel energy efficient virtual machine configuration and migration technique

L. Srinivasa Rao [1] *, I. Raviprakash Reddy [2]

[1] *Mother Teresa Institute of Science and Technology, Sathupally, Telangana State, India*
[2] *G Narayanamma Institute of Technology and Science, Hyderabad, Telangana State, India*
**Corresponding author E-mail: srinucsea4@gmail.com*

Abstract

The recent growth in the data centre usage and the higher cost of managing virtual machines clearly demands focused research in reducing the cost of managing and migrating virtual machines. The cost of virtual machine management majorly includes the energy cost, thus the best available virtual machine management and migration techniques must have the lowest energy consumption. The management of virtual machine is solely dependent on the number of applications running on that virtual machine, where there is a very little scope for researchers to improve the energy. The second parameter is migration in order to balance the load, where a number of researches are been carried out to reduce the energy consumption. This work addresses the issue of energy consumption during virtual machine migration and proposes a novel virtual machine migration technique with improvement of energy consumption. The novel algorithm is been proposed in two enhancements as VM selection and VM migration, which demonstrates over 47% reduction in energy consumption.

Keywords: *Energy Efficient VM Migration; Migration Technique; VM Components; VMM; VM Migration.*

1. Introduction

The core of cloud computing for providing the scalable infrastructure is virtualization. Virtualization is used to manipulate the underlying computing and communication infrastructure in order to create multiple instances of the same resources to be dedicatedly allocates to various customers or customer applications [1]. The virtualization as a technique allows the service providers to create different types of computing environments to match the customer requirements on a same server. Virtualization allows the service providers to meet the need for creating the scalable and application depended environments. At the same time, the customer can also reduce the cost for maintaining computing and storage related infrastructure on side. The majority of the service providers deploy the use of virtual machines in order to separate actual hardware from the computing environments.

To understand the applicability of virtualization through virtual machines, we consider the fact that in case of private and public cloud the requirements for multiple systems, maximization of resources and reducing the cost for implementation is the prime important factor. In case of a private cloud the customer tend to rent the infrastructure from a vendor or in some cases choose to host the infrastructure from own premises. In case of the on premises, hosting the same virtual machine can be used for variety of purposes and in case of off premises hosting the provider can use the same configuration for multiple cu0stomers. Hence, In both the cases, the implementation of virtualization through virtual machines reduces the cost [2].

In the other side for public cloud environment, the user can choose multiple configurations for virtual machines to suite the requirements. Thus the cost implementation for managing separate and custom configuration of hardware reduces for the providers.

The major implementation of virtualization is to manage the load balancing. The generic load balancing techniques enhance the response time for the applications running on the data centre. . Also it is to be understood that, the cloud based load balancing techniques allows the customers to use the global or geodetically distributed services based on geodetically distributed servers. Multiple parallel researches are been carried out to demonstrate the benefits of load balancing on cloud based data centres as handling the high unexpected traffic generally referred to Cyber Spikes. Making the application scalable based on demand without degrading the performance, increases the reliability at the cost of VM migration. The cost of the virtual machine migration is to be understood by the energy consumption as the major component. Thus this work presents a novel algorithm for virtual machine migration with reduced energy consumption.

The rest of the work is presented as in Section 2 the enhancements produced with VM migration is presented, in Section 3 the generic VM Image Storage architecture is presented, in Section 4 the work proposed the novel framework for virtualization, in Section 5 the work present the novel virtualization algorithm, the results are presented in Section 6 and in Section 7 the conclusion is presented.

2. VM migration benefits

The recent researches have demonstrated that the migration and adaptation of virtual Machine can enhance the performance of the client application and infrastructure cost can be reduced. Here in this work we consider the factors influencing the performance and productivity with the migration [4]:

Detailed Control:

The Virtual Machines come with a reduced abstraction is the system level and allows the provider, customer and researchers to access more properties of the system. The access to computing

Copyright © 2018 L. Srinivasa Rao, I. Raviprakash Reddy. This is an open access article distributed under the Creative Commons Attribution License, which permits unrestricted use, distribution, and reproduction in any medium, provided the original work is properly cited.

environment data, system level codes, hardware utilization statistics, traces of the active application, failing and down timing component configurations and the guest operating system configuration parameters and the ability to control them independently helps to understand the performance perimeters [Table -1].

Table 1: Parameters for Open Control

Type	Name	Access Permission	
		Traditional	Virtual Machine
Processing	CPU Type	Not Allowed	Allowed
	Allocation	Allowed	Allowed
	Priority	Allowed	Allowed
Memory	Size	Allowed	Allowed
	Buffer	Not Allowed	Allowed
Storage	Access IDE Bus	Not Allowed, Physical	Allowed, Logical
	Capture Mode	Not Allowed	Allowed
	Library Group	Allowed, Physical	Allowed, Logical
Network	IP Address	Allowed	Allowed
	MAC Address	Not Allowed	Allowed
	Internal Network	Partially Allowed	Allowed

Reduced Hardware Constraints:

The Virtual Machines comes with a flexibility to change or alter the operating system and hardware components seamlessly. After the initial cost for setting up a virtual environment, the provider, customer and researchers are free to modify the computing system including the operating system, libraries, tools and other supporting patches without investing the full time needed for computing system change or upgrade [Table -II].

Table 2: Up-Gradation Constraints

Type	Name	Accessibility	
		Traditional	Virtual Machine
Operating System	Version	Available	Available
	Interoperability	No Continuous Availability	Available
	Patch	Available	Available
IDE	Patch	Available	Available
	Device Driver	No Continuous Availability	Available
	Version	Available	Available
Configuration Time	-	Very High	Low

Replication Control:

The replication of the Virtual Machines using the snapshot feature allows the provider, customer and researchers to take timely and on demand backups of the virtual machine images. Thus the backups help to quickly reproduce the same computing environment without investing the complete setup time [Table -3].

Table 3: Replication Duration for Standard Configuration and Load

Type	Accessibility	
	Traditional	Virtual Machine
Windows Server	50 to 90 Mins	Just in Time
MAC Servers	40 to 60 Mins	Just in Time
Linux Servers	30 to 40 Mins	Just in Time

Availability:

The Virtual Machines are hosted by all service providers with similar configurations but with added advantages. Hence adopting to Virtual Machine computing is the best choice to avoid the lack of support and facility availability [Table – IV].

Table 4: VM Support for Cloud Service Providers

	1	2	3	4	5
Windows Server	YES	YES	YES	YES	YES
MAC Servers	YES	YES	YES	YES	YES
Linux Servers	YES	YES	YES	YES	YES

1) Amazon,
2) Microsoft
3) Google,
4) IBM,
5) Private Cloud

Regular Updates

The application on Virtual Machines hosted on cloud is always liable for automatic and regular updates from the service provider without any extra cost. However in the other side, hosting the traditional system demands the cost and time implications for updates.

Cost Control

Due to the tremendous competition in the cloud service provider space, the drop of price for each virtualization component used in the virtual machine configuration is dropping with an increasing speed. Hence rather than up-gradation cost for traditional systems, the cloud based virtual machines are very much cost effective [Table -5].

Table 5: Cost Drop History for Virtual Machines Hosting (Approximated Values)

	1	2	3	4	5
2012	$0.64	$0.70	$0.63	$0.61	$0.66
2013	$0.48	$0.45	$0.49	$0.47	$0.50
2014	$0.35	$0.39	$0.31	$0.30	$0.35
2015	$0.28	$0.26	$0.29	$0.26	$0.27

1) Amazon,
2) Microsoft
3) Google
4) IBM
5) Private Cloud

Collaborative Approach

The need for sharing the same testing and development environment is always a challenge in modern industrial requirements. Hence sharing the virtual machine image will certainly help in order to replicate the same development and testing environment timely and easily.

Manageable Data Loads

The recent researches have demonstrated that the closeness of the data and computational units can increase the speed of computation. However when the sting the virtual machines on the cloud with big data management features are quite efficient and directing towards the improvement in performance

3. Virtualization components and VMMS

After the detail understanding of the benefits and causes of the virtualization, here we understand the hardware and software components of virtualization using virtual machines. These two components will help us to understand and propose the novel framework.

a) Hardware Components for Virtualization:

Virtualization using Virtual Machines enables to isolate the physical hardware components from the software stack and increases the productivity discussed in early part on this work. In the process of virtualization, we understand the standard framework resides on the physical hardware and the virtual machine replicates the hardware logically. In the virtual machine the operating system gets install and runs the applications [Figure – 1].

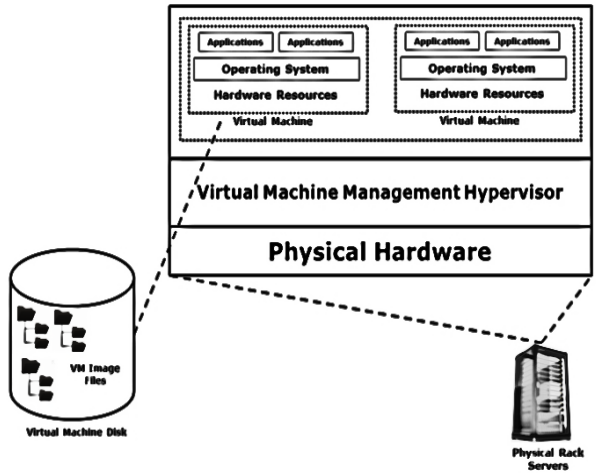


Fig. 1: Hardware Components for Virtualization.

Here we understand the solutions provided by multiple companies and by the independent searches, a very minimum scope is available to go for the enhancements. Thus we focus for the improvements possible in case Hypervisor architecture.

b) Software Components or Hypervisor for Virtualization

In order to provide the enhancements for Hypervisor technology, here we understand the architecture. The Hypervisor is mainly responsible for controlling the virtual machines along with managing the networking components, file systems, virtual machine images, software stack, the replication and other software controller. The Hypervisor is divided into three major layers as control layer, network layer and interfacing layer between physical and logical networking layers [Figure – 2].

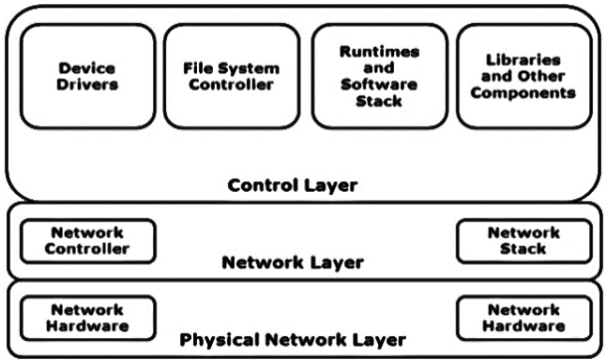


Fig. 2: Generic Architecture for Hypervisor.

This understanding helps us to realize that the aggregation of multiple physical servers is still not reached the highest pick of performance and research. The aggregation of the physical server running virtual machines are still not been realized due to the complex and diversified nature of the server vendors and configurations [5] [6].

Hence we propose the novel approach for storage and computing virtualization management. The detail of the proposed framework is discussed in the later part of this work [7].

4. Proposed computing virtualization management framework

After the detail understanding of the virtual machine framework and implementations, we realize the need for improvement in Hypervisor framework. Majorly we understand the following problems are generally encountered in most of the popular Hypervisor software.

The situation where multiple physical hardware servers are utilized for same or multiple clients, then the monitoring and managing individual hardware is nearly impossible from a single firmware or Hypervisors as the Hypervisors are restricted to specific hardware vendors. Hence we propose the Storage and Computing Virtualization Management Framework [Figure – 3].

The short comings are listed here:

- Lack of overall monitoring
- Lack of backup and restoration control
- Cross hardware and image format replication control
- Simplified monitoring and management of computational capacities
- Simplified monitoring and management of storage capabilities
- Hence we propose the novel framework to overcome all the listed shortcomings.

The proposed framework is a collection of software and monitoring applications stacks developed based on the open source Hypervisor implementation called Kernel Based Virtual Machine or KVM.

Here we describe the components of the proposed framework:

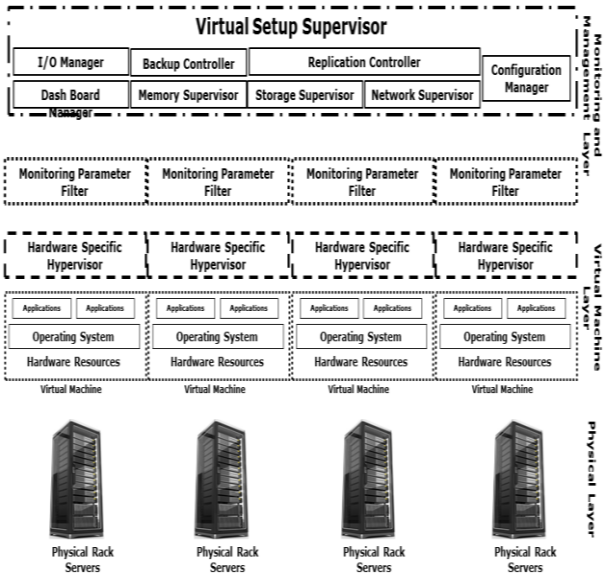


Fig. 3: Storage, Computing Virtualization Monitoring, and Management Framework.

a) Physical Layer

The physical layer is consisting of multiple physical hardware servers from different vendors. The servers are generic configuration and can be from any vendor specific configurations.

The servers are equipped with a small software package to read the server configuration and performance parameters mentioned in the previous part of this work to be supplied to the next layer.

b) Virtual Machine Layer

The virtual machine layer is generic and standard virtual machine implementation on top of the physical hardware layers.

The virtual machine layers are equipped with another small software agent to collect the virtual machine performance parameters from the specific Hypervisor tool and collect the parameters from the physical layer of the proposed implementation. The software agent will forward the same to the next layer.

c) Monitoring and Management Layer

The top layer of implementation is the Monitoring and Management Layer and works as the name suggests.

The management and monitoring layer is consisting of multiple software agents and described in detail here:

Dash Board: The Dash Board is the overall system report for the complete framework. The portal will be giving the information of the system based on the parameters like Unique Name of the VM, Host Architecture Type, Running or Stopped or Critical status of the system, Last Backup Date and Time, Time of Total Availability, Total Memory Utilization time, Total Disk Utilization in Gigabytes and Total Network Utilization in time.

Configuration Manager: The configuration manager is deployed to detect the change in physical or virtual level configuration and inform the respective software manager available in the framework in monitoring and management layer.
Memory Supervisor: The memory supervisor is responsible for keeping the memory monitoring system updated. The memory supervisor will monitor the system based on the parameters like Amount of Active memory in Gigabytes, Amount of Over heading Memory in Gigabytes, Amount of Swappable Memory in Gigabytes or Megabytes, Amount of Total Shared Memory and Temperature of the Memory Units.
Storage Supervisor: The storage supervisor is responsible for keeping the storage monitoring system updated. The storage supervisor will monitor the system based on the parameters like unique name of the Storage Container, Container Size in Gigabytes and Container Utilization in Gigabytes.
Network Supervisor: The network supervisor is responsible for keeping the network monitoring system updated. The storage supervisor will monitor the system based on the parameters like unique id for the Network Interface Card, Total up Time, Total down Time, Unique assigned IP Address, unique assigned MAC Address and Data Transfer Rate in Megabytes per second.
I/O Manager: The input/output or the peripheral supervisor is responsible for keeping the input/output or the peripheral monitoring system updated. The input/output or the peripheral supervisor will monitor the system based on the parameters like Unique Device ID, Read or Write type, Number of Read Operations and Number of Write Operations.
Backup Controller: The backup controller is responsible for taking the backups and restoring the same image into different formats on the underlying physical servers.
Replication Controller: The replication controller is responsible for replicating the delta change or the complete replications over multiple underlying physical servers in the physical layer.
The majority of the problems identified in the parallel research outcomes are been solved in the implementation. Mostly the proposed cross image conversion automation, with the use of backup and replication controller this work demonstrates satisfactory outcomes.

5. Proposed migration techniques

In this part of the work, the novel algorithm for virtual machine migration is been presented. The objective of the algorithm is facilitating the migration of the virtual machine from one server to another. The virtual machine on the physical server is identified based on the job load can noted as source and to the server virtual machine is to be migrated is noted as destination. The algorithm is explained here in multiple steps:
Step-1. In the first step, the calculation of the load on each virtual machine is calculated. This step is proposed to be executed on each physical server for each virtual machine.

$$Phy_{CPU_Capacity} = \sum_{i=1}^{n} VM(i)_{CPU_Capacity} \tag{1}$$

$$Phy_{Memory_Capacity} = \sum_{i=1}^{n} VM(i)_{Memory_Capacity} \tag{2}$$

$$Phy_{IO_Capacity} = \sum_{i=1}^{n} VM(i)_{IO_Capacity} \tag{3}$$

$$Phy_{Network_Capacity} = \sum_{i=1}^{n} VM(i)_{Network_Capacity} \tag{4}$$

$$\Pi = (Phy_{CPU_Capacity} + Phy_{Memory_Capacity} + Phy_{IO_Capacity} + Phy_{Network_Capacity}) \tag{5}$$

Now in the second step the highest and lowest loaded virtual machine is to be identified:

$$\Pi_{MAX} = \begin{cases} If\ \Pi_i > \Pi_j, then\ \Pi_{MAX} = \Pi_i \\ Else\ \Pi_j > \Pi_i, then\ \Pi_{MAX} = \Pi_j \end{cases} \tag{6}$$

$$\Pi_{MIN} = \begin{cases} If\ \Pi_i < \Pi_j, then\ \Pi_{MIN} = \Pi_i \\ Else\ \Pi_j < \Pi_i, then\ \Pi_{MIN} = \Pi_j \end{cases} \tag{7}$$

Step-2. Once the source and destination is identified as MAX and MIN respectively, the identification of job to be migrated is carried out, thus the identification of the virtual machine holding the job.

$$Migratable_Job_{Priority} = \begin{cases} \exists Job, Priority[Job(i)] > Priority[Job(j)], \\ then\ Migratable_Job \leftarrow Job(i) \\ Else\ Migratable_Job \leftarrow Job(j) \end{cases} \tag{8}$$

$$Migratable_Job_{Capacity} = \begin{cases} \exists Job, Capacity_Required[Job(i)] > Capacity_Required[Job(j)], \\ then\ Migratable_Job \leftarrow Job(i) \\ Else\ Migratable_Job \leftarrow Job(j) \end{cases} \tag{1}$$

$$Migratable_Job = \begin{cases} If\ Migratable_Job_{Priority} = Migratable_Job_{Capacity}, \\ then\ Migratable_Job \leftarrow Migratable_Job_{Capacity} \\ Else\ Migratable_Job \leftarrow Migratable_Job_{Priority} \end{cases} \tag{2}$$

After the successful identification of the job to be migrated, the correct virtual machine is to be identified.

$$VM(i) \leftarrow Identified_VM \tag{11}$$

After the identification of the virtual machine, the migration process will take place and after the migration new load of the source and destination is to be identified.

$$VM(i) = VM(i)_{CPUCapacity} + VM(i)_{MemoryCapacity} + VM(i)_{IOCapacity} + VM(i)_{NetworkCapacity} \tag{12}$$

$$\Pi_{MAX} - VM(i) = \Delta_{Source} \tag{13}$$

$$\Pi_{MIN} + VM(i) = \Delta_{Destination} \tag{14}$$

After the calculation of the new load, the source and destination nodes must obtain the optimal load condition, where the loads are nearly equally balanced.

$$\begin{cases} If\ \Delta_{Source} \approx \Delta_{Destination}, Then\ Migrate\ VM(i) \\ Else\ i = \in (n) \end{cases} \tag{15}$$

Where n is total number of virtual machines in Source node.
Results
This work has performed extensive testing to demonstrate the improvement over the existing migration techniques [13 - 17]. The various considered migration techniques are listed with the used acronyms here [Table – 6]:

Table 6: List of Techniques used for Performance Comparison

Used Name in this Work	Selection Policy	Allocation Policy
IQR MC	Maximum Correlation	Inter Quartile Range
IQRMMT	Minimum Migration Time	Inter Quartile Range
LR MC	Random Selection	Local Regression
LRMMT	Minimum Migration	Local Regression

	Time	
LR MU	Minimum Utilization	Local Regression
LR RS	Rom Selection	Local Regression
LRR MC	Maximum Correlation	Robust Local Regression
LRRMMT	Minimum Migration Time	Robust Local Regression
LRR MU	Minimum Utilization	Robust Local Regression
LRR RS	Rom Selection	Robust Local Regression
MAD MC	Maximum Correlation	Median Absolute Deviation
MAD MMT	Minimum Migration Time	Median Absolute Deviation
MAD MU	Minimum Utilization	Median Absolute Deviation
MAD RS	Rom Selection	Median Absolute Deviation
THR MC	Maximum Correlation	Static Threshold
THRMMT	Minimum Migration Time	Static Threshold
THR MU	Minimum Utilization	Static Threshold
THR RS	Rom Selection	Static Threshold
OPT ALGO	Proposed Algorithm Part – 1	Proposed Algorithm Part – 2

The simulation of the algorithm is based on CloudSim, which is a framework for modeling and simulation of cloud computing infrastructures and services. The experimental setup used for this work is been explained here [Table – 7]:

Table 7: Experimental Setup

Setup Parameters	Number of Physical Hosts	Number of Virtual Machines	Total Simulation Time (In Sec)
Values	800	1052	86400.00

Firstly the work analyses the energy consumption by the existing algorithms and the proposed algorithm. During the testing of the proposed algorithm, the result demonstrates an average of 19% reduction in energy consumption [Table – 8].

Table 8: Experimental Setup

Algorithm	Every Consumption (kWh)	Change (%)	Variation Comparison
LRR RS	34.1	-1.729106628	Deteriorate
LR MC	34.35	-1.008645533	Deteriorate
LRR MC	34.35	-1.008645533	Deteriorate
LRMMT	35.37	1.930835735	Enhanced
LRRMMT	35.37	1.930835735	Enhanced
LR MU	35.38	1.959654179	Enhanced
LRR MU	35.38	1.959654179	Enhanced
THR MC	40.85	17.72334294	Enhanced
THR RS	41.08	18.38616715	Enhanced
THRMMT	41.81	20.48991354	Enhanced
THR MU	44.08	27.03170029	Enhanced
MAD RS	44.6	28.53025937	Enhanced
MAD MC	44.99	29.65417867	Enhanced
MAD MMT	45.61	31.44092219	Enhanced
IQR MC	46.86	35.04322767	Enhanced
MAD MU	47.36	36.48414986	Enhanced
IQR RS	47.41	36.62824207	Enhanced
IQRMMT	47.85	37.8962536	Enhanced
IQR MU	49.32	42.13256484	Enhanced
Proposed Optimal Algorithm	34.7	-	

The result is also been analysed graphically [Fig-4].

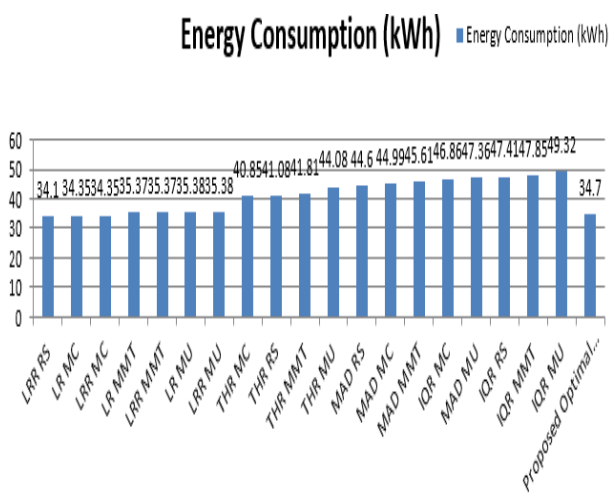


Fig. 4: Hardware Components for Virtualization.

6. Conclusion

This work have considered the benefits of virtual machine migrations from the on-side implementation of the systems and the benefits like Detailed Control, Reduced Hardware Constraints, Replication Control, Availability, Regular Updates, Cost Control, Collaborative Approach and Manageable Data Loads can be achieved. This work majorly demonstrates the optimal virtual machine migration technique with up to 13% reduction of energy consumption compared to the other virtual machine migration technique. The comparative analysis is been done with the proposed technique with the existing techniques like IQR MC, IQRMMT, LR MC, LRMMT, LR MU, LRR MC, LRRMMT, LRR MU, LRR RS, LR RS, MAD MC, MAD MMT, MAD MU, MAD RS, THR MC, THRMMT, THR MU and THR RS

References

[1] J. Rao, Y. Wei, J. Gong and C.-Z. Xu “Qos guarantees and service differentiation for dynamic cloud applications", IEEE Trans. Netw. Serv. Manag., vol. 10, no. 1, pp.43 -55 2013 https://doi.org/10.1109/TNSM.2012.091012.120238.

[2] Li, A. Raghunathan and N.K. Jha, &ldquo,Secure Virtual Machine Execution under an Untrusted Management OS,&rdquo, Proc. Int',l Conf. Cloud Computing, pp. 172-180, July 2010.

[3] Intel VT-D, http://www.intel.com/technology/virtualization/technology.htm, 2012.

[4] BYTEmark, http://www.tux.org/mayer/linux/byte/bdoc.pdf, 2012.

[5] M. Armbrust, A. Fox, R. Griffith, A.D. Joseph, R.H. Katz, A. Konwinski, G. Lee, D.A. Patterson, A. Rabkin, I. Stoica and M. Zaharia, &ldquo,Above the Clouds: A Berkeley View of Cloud Computing,&rdquo, Technical Report UCB/EECS-2009-28, http://www.eecs. berkeley.edu/Pubs/TechRpts/2009/EECS-2009-28.html, 2010.

[6] T. Deshane, Z. Shepherd, J.N. Matthews, M. Ben-Yehuda, A. Shah and B. Rao, &ldquo,Quantitative Comparison of Xen and KVM,&rdquo, Xen Summit Boston 2008, http://xen.org/xensummit/xensummit_ summer_2008.html, 2010.

[7] Y. Dong, X. Yang, X. Li, J. Li, K. Tian and H. Guan, &ldquo,High Performance Network Virtualization with SR-IOV,&rdquo, Proc. IEEE 16th Int',l Symp. High Performance Computer Architecture (HPCA), pp. 1-10, 2010.

[8] P. Padala Automated management of virtualized data centers, 2010:Univ. of Michigan.

[9] T. Patikirikorala, A. Colman, J. Han and L. Wang “A systematic survey on the design of self-adaptive software systems using control engineering approaches", Proc. Symp. Softw. Eng. Adaptive Self-Manag. Syst., pp.33 -42 2012.

[10] X. Wang and Y. Wang “Coordinating power control and performance management for virtualized server clusters", IEEE Trans. Parallel Distrib. Syst., vol. 22, no. 2, pp.245 -259 2011 https://doi.org/10.1109/TPDS.2010.91.

[11] X. Wang, M. Chen and X. Fu “MIMO power control for high-density servers in an enclosure", IEEE Trans. Parallel Distrib. Syst., vol. 21, no. 10, pp.1412 -1426 2010 https://doi.org/10.1109/TPDS.2010.31.

[12] T. Patikirikorala, A. Colman, J. Han and L. Wang “A multi-model framework to implement self-managing control systems for QoS management", Proc. Int. Symp. Softw. Eng. Adaptive Self-Manag. Syst., pp.218 -227 2011.

[13] X. Liu, C. Wang, B. Zhou, J. Chen, T. Yang and A. Zomaya, "Priority-based consolidation of parallel workloads in the cloud", IEEE Trans. Parallel Distrib. Syst., vol. 24, no. 9, pp. 1874-1883, 2013 https://doi.org/10.1109/TPDS.2012.262.

[14] Carrera, M. Steindler, I. Whalley, J. Torres and E. Ayguad, "Autonomic placement of mixed batch and transactional workloads", IEEE Trans. Parallel Distrib. Syst., vol. 23, no. 2, pp. 219-231, 2012 https://doi.org/10.1109/TPDS.2011.129.

[15] Y. Lee and A. Zomaya, "Energy efficient utilization of resources in cloud computing systems", The J. Supercomput., vol. 60, no. 2, pp. 268-280, 2012 https://doi.org/10.1007/s11227-010-0421-3.

[16] T. Ferret, M. Netto R. Calheiros and C. De Rose, "Server consolidation with migration control for virtualized data centers", Future Generation Comput. Syst., vol. 27, no. 8, pp. 1027-1034, 2011 https://doi.org/10.1016/j.future.2011.04.016.

[17] K. Mills, J. Filliben and C. Dabrowski, "Comparing vm-placement algorithms for on-demand clouds", Proc. IEEE 3rd Int. Conf. Cloud Comput. Tech. Sci., pp. 91-98, 2011. https://doi.org/10.1109/CloudCom.2011.22.