http://ansinet.com/itj

ITJ

ISSN 1812-5638

INFORMATION TECHNOLOGY JOURNAL

ANSInet
Asian Network for Scientific Information
308 Lasani Town, Sargodha Road, Faisalabad - Pakistan

OPEN ACCESS Information Technology Journal

ISSN 1812-5638
DOI: 10.3923/itj.2017.1.10

Research Article
Energy Management of the System: An Empirical Investigation of Virtualization Approaches in Static and Dynamic Modes

Basheer Riskhan, Ke Zhou and Raza Muhammad

School of Computer Science, Huazhong University of Science and Technology, 1037 Luoyu Road, 430074 Wuhan, China

Abstract

Background: Energy management has a vital impact on success of virtualization. Power consumption and generation of heat play critical roles in the core concept of energy management. Presently, energy consumption is continuously growing due to high usage of system resources and rising of demand in computer density. The high power usage and high heat generation lead to instabilities of their hardware systems and cause system reboot more often. **Materials and Methods:** To avoid above malfunctions and save from permanent damage to the system resources, this study are investigated and tested the power consumption and heat generation during the usage of low-level performance application, online education application and high-level performance application, respectively via the implementation of XEN, KVM and Docker virtualization technologies in between static and dynamic modes. **Results:** The result proved the power consumption and heat generation is higher for hypervisor-based virtualization and very low for container-based virtualization in all kind of applications. Even though the XEN and KVM are the same type of hypervisor is offering similar features but the output has deviated from each other. The limitation of VM/Docker on top of physical machine would control by mathematical formula, which created from our experiment. **Conclusion:** The overall analysis proved that XEN, KVM and Docker as the order of the power consumption rate and heat generation rate for all applications in high, medium and low, respectively.

Key words: Container, energy management, high performance application, hypervisor, low performance application, online education application, virtual machine, virtualization technology

Received: September 15, 2016 Accepted: November 04, 2016 Published: December 15, 2016

Citation: Basheer Riskhan, Ke Zhou and Raza Muhammad, 2017. Energy management of the system: An empirical investigation of virtualization approaches in static and dynamic modes. Inform. Technol. J., 16: 1-10.

Corresponding Author: Basheer Riskhan, School of Computer Science, Huazhong University of Science and Technology, 1037 Luoyu Road, 430074 Wuhan, China Tel: 0094714469676

Competing Interest: The authors have declared that no competing interest exists.

Data Availability: All relevant data are within the paper and its supporting information files.

INTRODUCTION

In recent trend the significant use of computers associated with the daily life. The development of computer technology has grown in such a way of cloud computing, machine learning and so on. Due to the said above reason and extended requirement of computer resources, the level of power consumption and heat generation rapidly growing as compare to the previous stages. In 2006, IT infrastructure in the USA consumption of the electricity was estimated at 4.5 billion dollars and in 2011 it tended to be double[1]. Above were the critical parameters during the usage of computer applications, which would impact on the performances, operational costs and deployment cost which created a negative impact on computer equipments[2]. Furthermore, the hardware devices consumed the power and converted into heat, which lead to increase the substrate temperature[3]. In addition, high heat creates the more degradation effects, which lead to change the time dependent during the operating characteristics of devices[4]. Considering the above overheads, there is a need to introduce and implement a new technology for energy management which is crucial. Also high performance, miniaturization and multi-functions were increased the generation of heat. So, control the heat generation effectively to the performance improvement and computer life extension[5]. Under this circumstance, came to know from the previous literature, developers considered vertical optimization or optimize the I/O or hypervisor or kernels would be the solution to improve the above overhead[6]. But according to the virtualization definition and understanding, hope that the Virtualization Based Approaches (VBA) would be better solution for energy management. Further this study discusses how PM power consumption and heat generation are influenced by the execution of different kind of application as well as different kind of virtualization technology.

Hypervisor Based Virtualization (HBV) and Container Based Virtualization (CBV) are the popular virtualization approaches were used in this study. Hypervisor can be implemented directly or separately on Physical Machine (PM), which creates the Virtual Machine (VM) and provides the virtualization hardware resources[6]. Single PM has the capacity to execute multiple VM but the PM must have advocate amount of resources for smooth operation[7]. Each VM runs on its own Operating System (OS). Currently users are rapidly increasing. This would result in an increasing need for VMs, which leads to increasing the demand and workload of PM. So, that resources should be shared more efficiently. Host OS can be used as the base in containers and it virtualizes the OS rather than the hardware[8].

This study decided to implement and test the power consumption and heat generation using different kind of virtualization technologies during the execution of the various applications. For that, this study is mainly concerned about XEN and KVM in HBV, Docker in CBV. Application was divided as Low-level Performance Application (LPA), Online Education Application (OEA) and High-level Performance Application (HPA). The LPA includes small application such as office packages. The OEA is the powerful educational learning tool that includes the prompt technological resources such as learning management system, eBooks library, E-learning resources, learning record management. The OEA was developed using PHP, HTML 5 and CSS. MySql has been chosen as the back-end database. The HPA include on energy hungry game, "Flight gear". The energy consumption contain in two different modes as static and dynamic. Static energy consumption when idle and dynamic energy consumption due to different kind of workloads executed in PM. The measurement tools start to execute during the usage of VM and the usage of Docker during the execution of LPA, OEA and HPA. Overall power consumption and heat generation of system are the elements were used to measure the matrices during the 180 sec execution. Finally this study propose better platform with power efficiency and heat efficiency of virtualized application by quantitative analysis approach.

MATERIALS AND METHODOS

Hypervisor Based Virtualization (HBV): A hypervisor is called as Virtual Machine Monitor (VMM) that is executed on top of the real physical hardware with host OS. It is software that is offering the abstraction for VM and isolation[9]. Hypervisor allows to running multiple VMs with multiple OSs on top of single PM[6]. Hypervisor is controlling the entire processes and allocating the resources such as processor, memory, network, applications and I/O devices. Each VM are fully isolating from others and executing in its own OS through virtualization technology, while running on the same hardware. Hypervisor is categorizing as bare metal virtualization and hosted virtualization shown on Fig. 1. Both categories were work based on hardware level or vitalizing hardware resources. The VMM and VM are directly mapping with the hardware resources and the processing is comparable to the native execution in bare metal virtualization. It's providing better security and higher level of virtualization efficiency. Hosted virtualization is only appearing between a host OS and VM. This approach is the most suitable for the development purpose[10]. Due to that reason the implementation of bare metal virtualization is the appropriate method in this experiment.

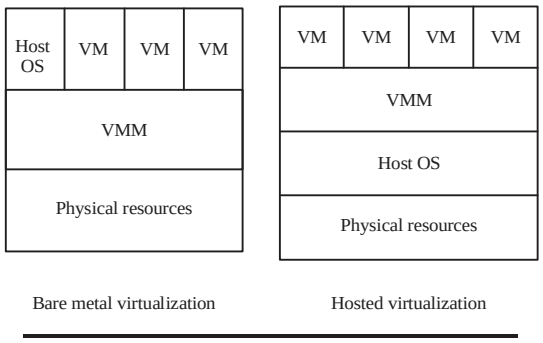

Fig. 1: Types of virtualization

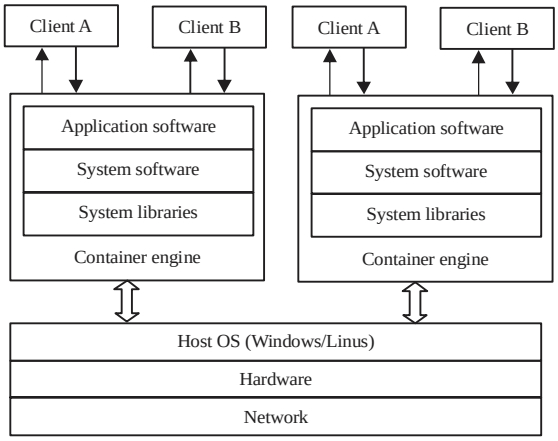

Fig. 2: Architecture of container based virtualization

Container Based Virtualization (CBV): The CBV is called as OS-level virtualization[9]. In this approach, container is using as an alternative for hypervisor[11]. Container works at OS level, which is fastest and lightweight process virtualization by the support of cloud computing[9]. Figure 2 exploring the architecture of CBV and container engine consisted in between host OS and binaries/libraries. Basically most container technology is performing and executing as a stand-alone server and it's including root access, memory, system libraries, IP addresses, files, users and application. Recently these technologies are getting a fair amount of attention as compare to other virtualization technologies in terms of its performance, scalability and security. Currently CBV has several implementation technologies such as LXC, OpenVZ, Linux-VServer, Dockers, Parallels Virtuozzo and so on. The organization and business firm is implementing these technologies depends on container configuration, number of OS and its environment, application capacity and demand of virtual resources. Docker is one of the approaches of application container, which is more supportive to package and compact an application into the virtual container[12]. Docker is enabling the portability, reusability, repeatability, continues integration, flexibility and fast development cycle.

Experimental background: This experiment was conducted in real time environment and it has mainly involved with PM, VM and Docker. The LPA, OEA and HPA were the target application. All the matrices were measured in the same physical host using java and shell programming with the support of Linux benchmark tools that were executed with VM and Docker. This experiment could get error free correlation during the comparison between VM and PM with Docker due to use of same physical resources and could calculate the mean value throughout the whole experiment. Power showed the average power consumption during the certain interval of time by the unit of watts. Temperature is recorded by sensor on the hardware resources by the unit of celsius. The experiment executed in the following steps:

- Setup the PM with high configure resources
- Evaluation tools were executed in the PM in order to get static state native execution result
- Reporting the evaluations matrix of power and temperature after completing the 180 sec execution
- Running the LPA, OEA, HPA and evaluation tools were executed in the PM in order to get dynamic state execution result
- Reporting the evaluations matrix of power and temperature after completing the 180 sec execution
- XEN and VM were installed on the top of PM, after VM is being activated
- Evaluation tools were executed in the PM in order to get static state single VM execution result
- Reporting the evaluations matrix of power and temperature after completing the 180 sec execution
- Running the LPA, OEA, HPA on single VM and evaluation tools were executed in the PM in order to get dynamic state single VM execution result
- Reporting the evaluations matrix of power and temperature after completing the 180 sec execution
- Next VM was increased one by one on top of PM and evaluation tools were executed in the PM in order to get the X number of VM execution result in static state.
- Then LPA, OEA, HPA running on top of each VM, evaluation tools were executed in the PM in order to get the X number of VM execution result in dynamic state
- All the same above steps were continued to KVM and Docker
- The reported matrices of native execution, XEN, KVM, Docker of power consumption and temperature generation had been taken for further analysis

This experiment faced technical issues at the beginning of the experiment such as continues execution of evaluation tools, reading the matrices during the selected interval time, stopping the execution in the particular time and storing the execution output. To overcome the above mentioned issues shell programming and java programming were used as evaluation tools.

Experiment resources: The matrixes are measuring in PM that are hosting and executing the several VM and Dockers. This experiment did not measuring in VM/Docker because it is not purely hardware[2]. The HDD, processor, memory and several internal components having the power consumption and heat generation, that can be measured using build-in meter tools and build-in sensors. Control unit, ALU, IO control, interrupt handling, instruction pipeline, fetch and decode hardware were the important computational task performed by system resources which too leads to consume power and generate heat. Further cache memory is also having the power and heat, which were added to the processor to improve execution. Considering the above fact about the energy management and indented to conduct this experiment. Table 1 expresses the system configuration, which used in this experiment.

Physical machine: The PM is the crucial part of our experiment, because VM/Docker is not physically bounded in hardware and one or more VM/Docker can host in a single PM. Further determining the number of VM/Docker and its workload are depends on hosted PM configuration. In order to that this experiments were conducted on large-scale multi-core PM with high processing power, huge memory and multi-core accelerate processor.

Virtual machine: The first approach of this experiment was conducted using the VM. The VM was installed on top of the PM. The same PM and same configuration were used in second experiment approach too. So, it could be avoided the conflict of the machine configuration. This experiment conducted in order to measure the power and heat during the execution of VM. It was using bare metal virtualization, which brought to near native execution comparing to hosted virtualization.

Dynamic resource allocation method used to this experiment. That trims the issue of insufficient resources of VM, because PM offers the resources to the VM if needed. The XEN 4.0, KVM+QUMU were installed as a hypervisor that was controlling the resources, host process and allocating the needed resources to the correct VM in the correct time. Ubuntu 14.04 LTS was installed as their guest OS, that handling the aggregation of I/O request, internal memory and run the OES. Table 1 shows the specification of VM.

Experimental variables: Power and heat of the system were used as variables in this experiment, that were the most appropriate element for estimating the number of VM/Docker hosting on PM. Power consumption indicates that the sufficient amount of power or energy was needed to execute the workload of running application[2]. Heat is another variable depends on the characteristics of system and its component. Overheating system resources is the main issue of high heat that was lead to instability in the hardware of the system and its reboots. The measuring unit of the power is watts and heat is celsius.

Experiment tools: Java and shell programming were used as experiment tools. POWERSTAT and SENSOR are the Linux Kernel modules that used to measuring the power consumption of mobile PC and measuring the current reading of the heat in all sensor chips, respectively. But default powerstat (powerstat <interval >< No. of sample >) has given some issues such as starting time of execution and limiting the number of sample. The default sensor command has given some issues such as interval time, limitation of execution. So, that, Linux kernel module tools were inserted inside java code and Linux shell code to the experiment requirement and runs in the PM.

Table 1: Resources configurations

Physical machine configuration	Virtual machine configuration	Container configuration
Processor: Intel core 4200 series 2.4 GHz	Processor: Intel core 4200 series 2.4 GHz	Container: Docker engine
Host OS: Ubuntu server 12.04.3 LTS	Guest OS: Ubuntu 14.04 LTS	OS image: Ubuntu 14.04 LTS
Web server: Nginx	Web server: Nginx	Web server: Nginx
Application: LPA/OEA/HPA	Application: LPA/OEA/HPA	Application: LPA/OEA/HPA
Memory: 8 GB	Memory: 1 GB	
Hard disk: 1024 GB SATA	Hard disk: 20 GB	
Cache memory: 1642 MB	Resource allocation method: Dynamic	
File system: Ext 3	Hypervisor XEN 4.0	

Shell coding which includes Linux Kernel module: Measure the power consumption in 180 sec execution, interval 3 sec

```
#!/bin/bash
powerstat 3 180-d 0>tmp0 &
sleep 180
kill-9 $!
echo ' '
cat tmp0 | grep '0.0' | awk '{print $10}'>mp1
sum = 0
echo $sum
for line in `cat tmp1`
do
echo $line
sum = $(echo "scale = 2;${sum}+${line}"|bc)
done
avg = $(echo "scale = 3;${sum}/60" | bc)
printf "Average watts: %s\n" ${avg}
rm-f tmp0 tmp1
```

Java coding which includes Linux Kernel module: Measure the Heat in 180 sec execution, interval 3 sec

```
public class cputempsecond {
public static void main(String[ ] args){
try {for (int i=1; i<=60; i++)
{ Thread.sleep(3×1000);
System.out.println ("NO" + i);
DateFormat df = new SimpleDateFormat ("HH:mm:ss");
Date date = new Date();
System.out.println ("current time is" + df.format(date));
String shellCommand = "sensors";
String[] cmd = { "/bin/sh", "-c", shellCommand };
Process ps = Runtime.getRuntime().exec(cmd); ps.wait For ();
BufferedReader br = new BufferedReader(new
InputStreamReader (ps.getInputStream()));
StringBuffer sb = new StringBuffer(); String line;
while ((line = br.readLine()) != null)
{ sb.append(line).append("\n"); }
String result = sb.toString();  String filePath = "tmp.txt";
File file = new File(filePath);
FileWriter fw = new FileWriter(file, true);
BufferedWriter bw = new BufferedWriter(fw);
bw.write(result);                bw.close(); System.out.println(result);}}
catch (Exception e) {e.printStackTrace(); }}}
```

RESULTS

This experiment was conducted first for static state and second for dynamic state successfully. The matrices were started to read in normal mode for static state and in the beginning of the application execution for dynamic state. The experiment was started in 3 states such as native execution, VM/Docker install on PM and level of VM/Docker extended one by one. Further mathematical equations were formed by us for power consumption and heat generation using least square method, that's helps to limit the extended resources accepted by PM and got to the know the workload of application before the execution. Each test was contained the different workload of application with different parameters. Power consumption is the hardware measurable parameters and it's showing the energy units in per seconds or current power consumption rate. Table 2 showed the noted power consumption during the extended level of VM/Docker in static and dynamic mode.

Heat is another parameter of this experiment that generates the heat from all the hardware devices include hard drives, graphics cards, processors connected with the system. It varies from one device to another depending upon its component and running applications. Table 3 shows the generated heat during the extended level of VM/Docker in static and dynamic mode.

The analysis is discussed in following different ways based on collected power and heat measurement. The native machine reading is the base value and it is considered to be constant during the discussion:

- Is virtualization environment has given the near native or how the native execution varies from the virtualization environment

Table 2: Power consumption

Native												
				Dynamic state								
Static state				LPA			OEA				HPA	
10.33				11.37			12.93				14.47	
	XEN				KVM				Docker			
	Dynamic state				Dynamic state				Dynamic state			
No. of VM/Doc	Static state	LPA	OEA	HPA	Static state	LPA	OEA	HPA	Static state	LPA	OEA	HPA
1 resource	44.01	44.10	46.14	48.72	43.93	45.05	45.92	47.28	41.70	42.62	43.89	45.61
2 resource	45.17	46.25	46.65	50.02	45.02	46.11	46.38	49.15	41.91	42.93	44.57	46.26
3 resource	45.82	47.13	48.01	50.97	45.37	46.97	47.11	50.34	42.27	43.22	45.33	46.99
4 resource	46.14	47.69	48.89	52.01	45.96	47.62	48.17	51.19	43.32	43.67	45.92	47.75
5 resource	47.09	48.88	49.12	53.36	46.69	48.01	48.94	52.82	43.71	44.19	46.62	48.87

Table 3: Heat generation

Native												
				Dynamic state								
Static state				LPA				OEA				HPA
41.04				42.11				43.24				44.97
	XEN				KVM				Docker			
	Dynamic state				Dynamic state				Dynamic state			
No. of VM/Doc	Static state	LPA	OEA	HPA	Static state	LPA	OEA	HPA	Static state	LPA	OEA	HPA
1 resource	44.01	44.1	46.14	48.72	43.93	45.05	45.92	47.28	41.7	42.62	43.89	45.61
2 resource	45.17	46.25	46.65	50.02	45.02	46.11	46.38	49.15	41.91	42.93	44.57	46.26
3 resource	45.82	47.13	48.01	50.97	45.37	46.97	47.11	50.34	42.27	43.22	45.33	46.99
4 resource	46.14	47.69	48.89	52.01	45.96	47.62	48.17	51.19	43.32	43.67	45.92	47.75
5 resource	47.09	48.88	49.12	53.36	46.69	48.01	48.94	52.82	43.71	44.19	46.62	48.87

Table 4: System power consumption (x: No. of resources)

Static stage power consumption in XEN hypervisor	$y = 0.446x+12.30$ ($R^2 = 0.919$)
Dynamic stage power consumption for HPA in XEN hypervisor	$y = 0.617x+17.95$ ($R^2 = 0.974$)
Dynamic stage power consumption for OEA in XEN hypervisor	$y = 0.481x+14.33$ ($R^2 = 0.962$)
Dynamic stage power consumption for LPA in XEN hypervisor	$y = 0.398x+12.97$ ($R^2 = 0.919$)
Static stage power consumption in Docker container	$y = 0.371x+11.54$ ($R^2 = 0.975$)
Dynamic stage power consumption for HPA in Docker container	$y = 0.545x+17.16$ ($R^2 = 0.994$)
Dynamic stage power consumption for OEA in Docker container	$y = 0.382x+13.64$ ($R^2 = 0.982$)
Dynamic stage power consumption for LPA in Docker container	$y = 0.313x+12.58$ ($R^2 = 0.983$)

- How the hypervisor differs from container based virtualization
- Get the idea of power intensive and heat management for the various kind of application

The vast deviation of power consumption was noticed in between native execution and virtualization execution in both static and dynamic stage. The matrixes were gradually increased while extending the level of resources. Scheduling the multiple processing tasks during the extended level of VM/Docker might be the reason of increasing the power consumption. Same number of core usage or more core usage in the virtual environment has led to busier. When the processor gets busier, it has drawn more power. The XEN, KVM in HBV and Docker in CBV also have different power consumption but not the same as native execution. This might be the reason of VM loaded their own OS, load process has taken more time and more power when compared to Docker. Further needed some storage to the hypervisor to keep the track of number of parameters over the small period and increased I/O process has a chance to consume more power too. Figure 3 describes clearly about the flow of all power consumption. The XEN was consumed more power when compared to KVM. But Docker was consumed less power as compared with XEN and KVM. The created mathematical equation of power consumption shown in Table 4, which was helps to limit the number of VM/Docker.

In heat generation, compared to the native execution, the Docker shows less different because, light weight OS executed separately inside the PM. But hypervisor generated more heat than native execution, because VM installed and executed its full OS inside the PM. Further VMs/Dockers were generated more heat in terms of usage of more CPU, high CPU intensive process and higher resource utilization. Figure 4 explores the heat generation that generated the heat in descending order to XEN, KVM and Docker, respectively. The XEN, KVM processes their own memory and it can be run in many instances as compared to Docker that leads to generated more heat. The created mathematical equation of heat generation shown in Table 5, which helps to limit the number of VM/Docker.

DISCUSSION

This study is mainly discussing about the system energy consumption deviation between HBV and CBV in static and dynamic modes. Currently power consumption and heat generation rate are increasing due to high usage of system resources and energy consuming application. Mobile phone and tablet PC are enhancing their architectures but still remain limitation of processing memory, power constrains and heat management, application portability due to different device platform. These overheads have created the issues to use all applications properly and the performances of the devices not

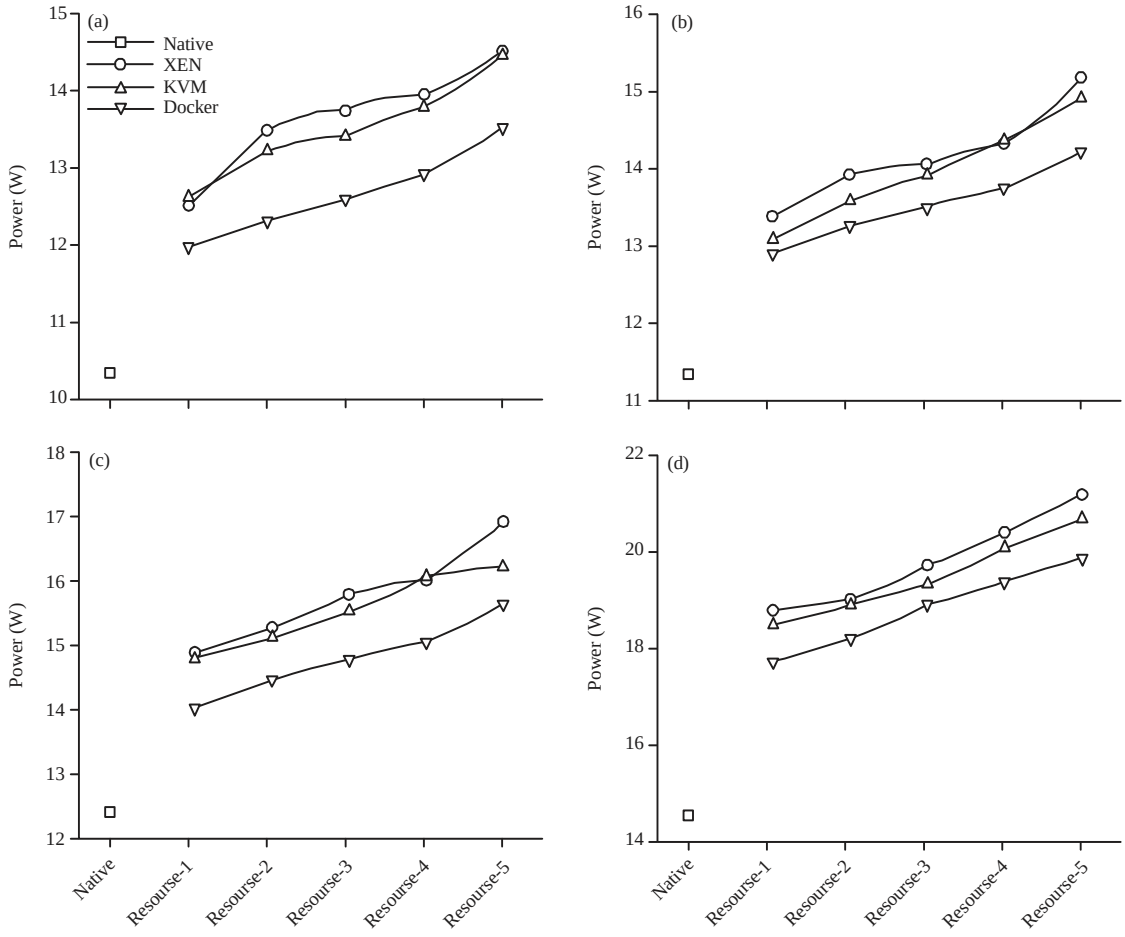

Fig. 3(a-d): Consumption of all applications, (a) Power consumption in static mode, (b) Power consumption for LPA, (c) Power consumption for OEA and (d) Power consumption for HPA

Table 5: System heat generation (x: No. of resources)

Static stage heat generation in XEN hypervisor	$y = 0.713x+43.50$ ($R^2 = 0.966$)
Dynamic stage heat generation for HPA in XEN hypervisor	$y = 1.127x+47.63$ ($R^2 = 0.996$)
Dynamic stage heat generation for OEA in XEN hypervisor	$y = 0.820x+45.30$ ($R^2 = 0.954$)
Dynamic stage heat generation for LPA in XEN hypervisor	$y = 1.100x+43.51$ ($R^2 = 0.943$)
Static stage heat generation in Docker container	$y = 0.543x+40.95$ ($R^2 = 0.937$)
Dynamic stage heat generation for HPA in Docker container	$y = 0.801x+44.69$ ($R^2 = 0.988$)
Dynamic stage heat generation for OEA in Docker container	$y = 0.681x+43.22$ ($R^2 = 0.998$)
Dynamic stage heat generation for LPA in Docker container	$y = 0.388x+42.16$ ($R^2 = 0.983$)

equally supported to all application. The implementation of proper virtualization technology is the significant solution to above mentioned overheads. Based on the above output, the researchers came to know how the different workload of application has impacted by the different kind of virtualization technology.

Patil and Bhavani[13] has explored, VM must have heterogeneous resources due to run different kind of application and PM satisfied the needed resources of all VMs running on it. Otherwise PM lead performance degradation of its VMs while application runs on the VMs. The resource utilization and memory utilization were improved by minimizing the skewness. Due to save the power wastage of PM in idle mode researcher has introduced the green computing, which save energy by finding the rate of average resource utilization. The cost of the running application and cloud computing infrastructure are minimizing by cloud computing. Xavier *et al.*[11] has expressed, the Container Based Technology (CBT) such as Linux-VServer, OpenVZ and Linux container (LXC) has low overhead as compared to other virtualization technology during the usage of HPC application, because CBT is offering lightweight virtualization, better

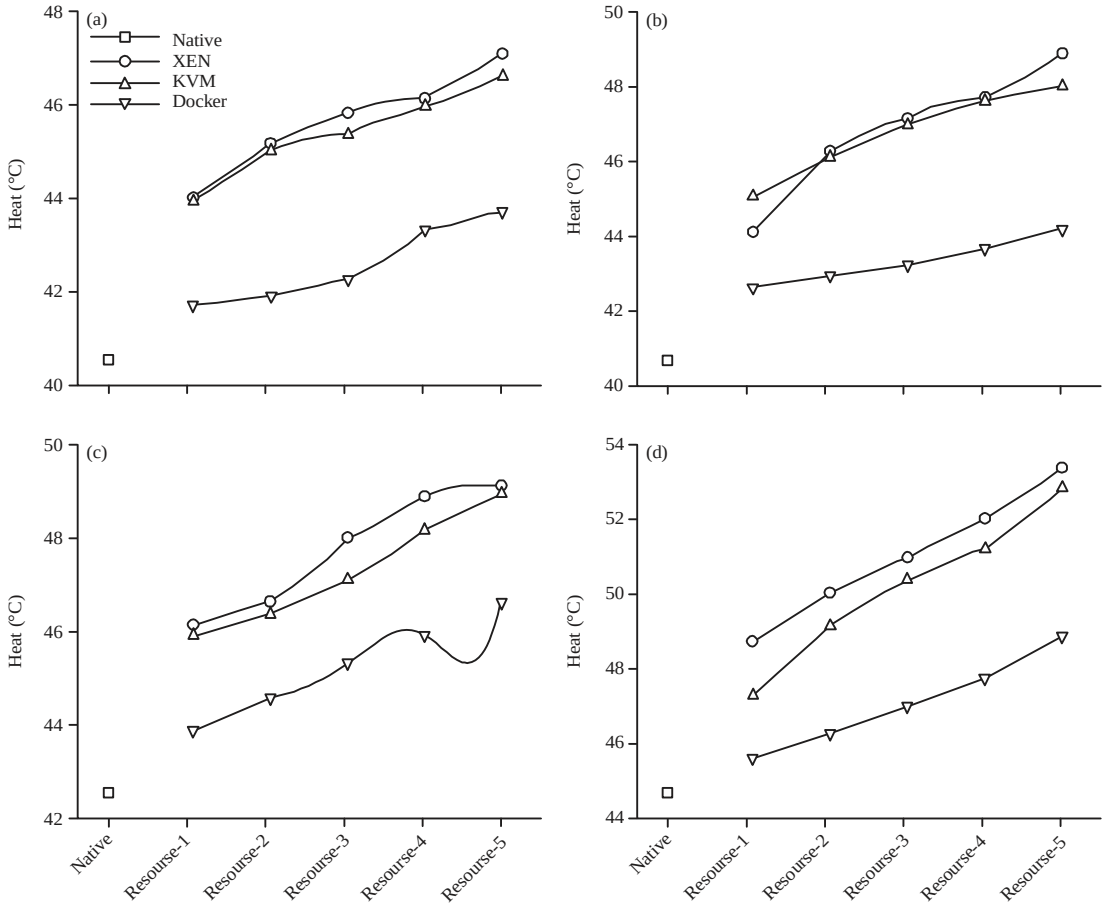

Fig. 4(a-d): Heat generation of all applications, (a) Heat production in static mode, (b) Heat production for LPA, (c) Heat production for OEA and (d) Heat production for HPA

resource sharing, custom environment and etc. In order to that, researcher is expecting the near native performance from the PM resources. Researcher said, all virtualization systems obtained near-native performances. The XEN does not perform natively for CPU intensive benchmarks and the Memory. In disk performance, LXC and Linux-VServer were obtained near native and OpenVZ was obtained similar native performance. In network, native performances were obtained in Linux-VServer, LXC and OpenVZ as followed. But overall, XEN was obtained bad performance in all virtualization. In container, poor isolation and less security were observed in the resource management implementation. In isolation, XEN has showed better performance due to non-shared OS.

Dhiman *et al.*[14] proposed the concept of vGreen that is the energy efficient software system which managed the VM scheduling with implementation of different PMs. In vGreen architecture, researcher represent vgnode as the cluster of each PM, vgserv as the central server mange the VM scheduling and vgpolicy as the policies running on vgserv. Each vgnode consists of XEN (vgxen), Dom0 (vgdom) and matrices evaluate using vgreen modules. The memory access per cycle, instruction per cycle and different VMs utilization were the matrices captured during the VM execution on vgnodes. Finally researcher observed the 20% Improvement of performance and 15% Improvement of system level energy saving during the implementation of vGreen in real time.

Tikotekar *et al.*[15] has raised the question, which was related to performance impact such as. (1) How to quantify the difference between the flexibility offered by two VM configurations?. (2) Is the performance impact between two VM configurations? to finding the answer, he was created two type of VM with different configuration and executed the LAMMPS application. He measured wall clock time for overall performance. He conclude that there is a 3% performance impact between different configurations VMs. Linux VM is the better control then XEN in handling the application, resources.

According to the above literatures, understand that the researches are ongoing regarding virtualization and its

applications. So, as per my review of literature, less amount of researchers has concerned about the issues of power consumption and heat generation while using different workloads. So, this study has enforced and investigated the issues of power and heat.

CONCLUSION

In order to reap the benefits of virtualization technology, reducing the current overhead of power consumption and heat generation, recommendation is to provide better platform for using high performance application, online education application and low performance application. These investigations were done based on the quantitative analysis approach. Mainly this experiment was concerned with XEN, KVM in hypervisor based virtualization and Docker in container based virtualization through the static and dynamic mode. The static is the native value, which is the basic and constant value to the particular machine configuration. Online education application is a web application, which was developed and deployed on the cloud. Linux benchmark office applications were used for low performance application and "Flight gear" game was used for high performance application. Power and heat was used as variables that were measured by watts and celsius, respectively. Java and Linux shell coding were used as experiment tools to execute on the PM. Several kinds of tests were conducted for 60 times consecutively with the 3 sec interval. Detail comparison was done with the implementation of VM/Docker. The innovation after our investigation resulted as follow:

- Each application and each virtualization technology consumed different level of power and generated different level of temperature
- In all 3 types of applications ascending order power consumption and temperature generation rate was observed in Docker, XVM and XEN, respectively
- The XEN and KVM is the similar type of hypervisor based virtualization, even though it has consumed relatively at the same amount of power and small deviation of heat generation. But vast differences were noticed in Docker container, which consumed very less power and less heat generation than hypervisor based virtualization
- Limit the number of VM/Docker on top of physical machine through the mathematical expression which created from least square method with the use of measured matrices

As a conclusion, this study recommended that the Docker virtualization technology is providing better energy management in static and dynamic mode to all 3 kinds of applications.

SIGNIFICANT STATEMENT

The number of extending resources and extending applications on top the physical host is consuming more power and generating more heat as compare to native execution that leads to instabilities of their hardware resources. It has been identify through the implementation of virtual machine using virtual box in my previous investigation. This study demonstrated to execute the applications with the implementation of Docker virtualization technology were consumed less power and generated less heat among the other virtualization technologies of XEN and KVM.

REFERENCES

1. Kinger, S. and K. Goyal, 2013. Energy-efficient CPU utilization based virtual machine scheduling in Green clouds. Proceedings of the 5th International Conference on Advances in Recent Technologies in Communication and Computing, September 20-21, 2013, Bangalore, India, pp: 28-34.
2. Marcu, M., D. Tudor and S. Fuicu, 2011. Power consumption and temperature measurement of virtualization solutions. Proceedings of the 17th International Workshop on Thermal Investigations of ICs and Systems, September 27-29, 2011, Paris, pp: 1-6.
3. Calimera, A., E. Macii, D. Ravotto, E. Sanchez and M.S. Reorda, 2010. Generating power-hungry test programs for power-aware validation of pipelined processors. Proceedings of the 23rd Symposium on Integrated Circuits and System Design, Sao Paulo, Brazil, September 6-9, 2010, ACM, New York, USA., pp: 61-66.
4. Alam, M., 2008. Reliability-and process-variation aware design of integrated circuits. Microelectron. Reliabil., 48: 1114-1122.
5. Oh, Y.K. and H.D. Yang, 2015. A study on control of heat generation in computer using thermoelectric cooling system. J. Korea Academia-Industrial Cooperat. Soc., 16: 43-49.
6. Riskhan, B. and M. Raza, 2016. Virtual machine performance approaches in the online education system. Proceedings of the International Multi Conference of Engineers and Computer Scientists, March 16-18, 2016, Hong Kong, pp: 94-99.
7. Selokar, A., S.D. Zade and C.U. Chavan, 2014. Survey on dynamic resource allocation using virtual machines for cloud computing environment. Int. J. Adv. Res. Comput. Commun. Eng., 3: 6449-6452.

8. Badola, V., 2015. Container virtualization: What makes it work so well? Cloud Academy, October 27, 2015. http://cloudacademy.com/blog/container-virtualization/.
9. Joy, A.M., 2015. Performance comparison between linux containers and virtual machines. Proceedings of the International Conference on Advances in Computer Engineering and Applications, March 19-20, 2015, India, pp: 342-346.
10. Scott, S.L., G. Vallee, T. Naughton, A. Tikotekar, C. Engelmann and H. Ong, 2008. System-level virtualization research at oak ridge national laboratory. Oak Ridge National Laboratory, Oak Ridge, January 2008.
11. Xavier, M.G., M.V. Neves, F.D. Rossi, T.C. Ferreto, T. Lange and C.A.F. de Rose, 2013. Performance evaluation of container-based virtualization for high performance computing environments. Proceedings of the 21st Euromicro International Conference on Parallel, Distributed and Network-Based Processing, February 27-March 1, 2013, Belfast, pp: 233-240.
12. Adufu, T., J. Choi and Y. Kim, 2015. Is container-based technology a winner for high performance scientific applications? Proceedings of the 17th Asia-Pacific Network Operations and Management Symposium, August 19-21, 2015, Busan, pp: 507-510.
13. Patil, S.S. and K. Bhavani, 2014. Dynamic resource allocation using virtual machines for cloud computing environment. Int. J. Eng. Adv. Technol., 3: 218-221.
14. Dhiman, G., G. Marchetti and T. Rosing, 2009. vGreen: A system for energy efficient computing in virtualized environments. Proceedings of the International Symposium on Low Power Electronics and Design, August 19-21, 2009, San Francisco, California, USA., pp: 243-248.
15. Tikotekar, A., H. Ong, S. Alam, G. Vallee, T. Naughton, C. Engelmann and S.L. Scott, 2009. Performance comparison of two virtual machine scenarios using an HPC application: A case study using molecular dynamics simulations. Proceedings of the 3rd ACM Workshop on System-Level Virtualization for High Performance Computing, March 31, 2009, Germany, pp: 33-40.