ICCSN
2016

Proceedings of
2016 8th IEEE International Conference on
Communication Software and Networks (ICCSN 2016)

IEEE Catalog Number: CFP 1669-CDR
ISBN: 978-1-5090-1779-9

Proceedings of
2016 8th IEEE International Conference on Communication Software and Networks (ICCSN 2016)

June 4-6, 2016
Beijing, China

IEEE

2016 8th IEEE International Conference on Communication Software and Networks

ICCSN 2016

Table of Contents

Session 1: Information Theory and Information Security

Session 2: Signal Processing and Application

Session 3: Communication and Information System

Session 4: Navigation System and Satellite Communication

Session 5: Antenna Technology and Microwave Engineering

Session 6: Electronic and Communication Engineering

Session 7: Network Resource Management

Session 8: Wireless Sensor Networks

Session 9: Mobile Communication and Wireless Technology

Session 10: Information Network and Multimedia Technology

Session 11: Image Processing and Application

Session 12: Computer Science and Applications

Author Profile

HENRY NOVIANUS PALIT

Universitas Kristen Petra

SINTA ID : 6024125

Subjects/Areas:

ID

Distributed Computing | Cloud Computing | Big Data Analysis

sinta

13.95
Overall Score

1.63
3 Years Score

1231
Overall Score V2

161.5
3 Years Score V2

0
Books

5813
Rank in National

27187
3 Years National Rank

1
IPR

Scoring

15
Rank in Affiliation

57
3 Years Affiliation Rank

Search..

Filter by type: Journal Proceeding Book Other All

Page 1 of 3 | Total Records : 22

Quartile	Publications	Citation
-	Hybrid heuristic for scheduling data analytics workflow applications in hybrid cloud environment IEEE International Symposium on Parallel and Distributed Processing Workshops and Phd Forum | vol: | issue : | 2011-12-20 | Conference Proceedin	59
-	Processing of wearable sensor data on the cloud - A step towards scaling of continuous monitoring of health and well-being 2010 Annual International Conference of the IEEE Engineering in Medicine and Biology Society, EMBC 1 | vol: | issue : | 2010-12-01 | Conference Proceedin	12
Q4	A dynamic hybrid resource provisioning approach for running large-scale computational applications on cloud spot and on-demand instances Proceedings of the International Conference on Parallel and Distributed Systems - ICPADS | vol: | issue : | 2013-01-01 | Conference Proceedin	12
-	Building an HPC-as-a-service toolkit for user-interactive HPC services in the cloud Proceedings - 25th IEEE International Conference on Advanced Information Networking and Applications | vol: | issue : | 2011-05-31 | Conference Proceedin	8
-	Evaluating hardware-assisted virtualization for deploying HPC-as-a-service VTDC 2013 - Proceedings of the 7th International Workshop on Virtualization Technologies in Distribu | vol: | issue : | 2013-07-17 | Conference Proceedin	7
Q4	Design and development of an adaptive workflow-enabled spatial-temporal analytics framework Proceedings of the International Conference on Parallel and Distributed Systems - ICPADS | vol: | issue : | 2012-12-01 | Conference Proceedin	5
-	Server Scalability Using Kubernetes TIMES-iCON 2019 - 2019 4th Technology Innovation Management and Engineering Science International Co | vol: | issue : | 2019-12-01 | Conference Proceedin	3
-	Linux PAM to LDAP authentication migration Proceedings - 2017 International Conference on Soft Computing, Intelligent System and Information Te | vol: 2018-January | issue : | 2017-07-02 | Conference Proceedin	2
Q3	Proxy-based pervasive multimedia content delivery Proceedings - International Computer Software and Applications Conference | vol: 1 | issue : | 2006-12-01 | Conference Proceedin	1
-	Content grid architecture for pervasive content delivery Proceedings - 2006 IEEE International Conference on Services Computing, SCC 2006 | vol: | issue : | 2006-12-01 | Conference Proceedin	1

Page 1 of 3 | Total Records : 22

Page 2 of 3 | Total Records : 22

Quartile	Publications	Citation
-	An efficient co-processing framework for large-scale scientific applications Proceedings of the International Conference on Cloud Computing Technology and Science, CloudCom | vol: 2015-February | issue : February | 2015-02-09 | Conference Proceedin	1
-	Exploratory research on developing hadoop-based data analytics tools Proceedings - 2017 International Conference on Soft Computing, Intelligent System and Information Te | vol: 2018-January | issue : | 2017-07-02 | Conference Proceedin	1
-	Near field communication technology in delivering information in museums Proceedings - 2017 International Conference on Soft Computing, Intelligent System and Information Te | vol: 2018-January | issue : | 2017-07-02 | Conference Proceedin	1
-	Streaming Media Implementation in Moodle-Based E-Leaming Application TIMES-iCON 2018 - 3rd Technology Innovation Management and Engineering Science International Confere | vol: | issue : | 2019-01-22 | Conference Proceedin	1
Q4	Optimization of computer resources using openstack private cloud ACM International Conference Proceeding Series | vol: | issue : | 2019-09-20 | Conference Proceedin	1
-	Using TCOZ for modeling web caching Proceedings of the Asia-Pacific Software Engineering Conference and International Computer Science C | vol: | issue : | 2001-12-01 | Conference Proceedin	0
-	Using TCOZ for modeling web caching Proceedings of the Asia-Pacific Software Engineering Conference and International Computer Science C | vol: | issue : | 2001-12-01 | Conference Proceedin	0
Q3	Layered video delivery from multiple servers Proceedings of SPIE - The International Society for Optical Engineering | vol: 6777 | issue : | 2007-12-01 | Conference Proceedin	0
-	Content grid: A novel approach to address pervasive internet access 2006 2nd International Conference on Semantics Knowledge and Grid, SKG | vol: | issue : | 2006-01-01 | Conference Proceedin	0
-	Two-layer host blocking: Caching the blocking decisions Proceedings - 5th International Conference on Algorithms and Architectures for Parallel Processing, | vol: | issue : | 2002-01-01 | Conference Proceedin	0

Page 2 of 3 | Total Records : 22

Page 3 of 3 | Total Records : 22

Quartile	Publications	Citation
-	Deploying an Ad-Hoc computing cluster overlaid on top of public desktops Proceedings of 2016 8th IEEE International Conference on Communication Software and Networks, ICCSN | vol: | issue : | 2016-10-07 | Conference Proceedin	0
-	Preface Proceedings - 2017 International Conference on Soft Computing, Intelligent System and Information Te | vol: 2018-January | issue : | 2018-01-16 | Conference Proceedin	0
Q2	Comparative Analysis of NFS and iSCSI Protocol Performance on OpenStack Cinder Technology Procedia Computer Science | vol: 171 | issue : | 2020-01-01 | Conference Proceedin	0

Page 3 of 3 | Total Records : 22

2016 International Conference on Communication Software and Networks

Deploying an Ad-Hoc Computing Cluster Overlaid on Top of Public Desktops

Henry Novianus Palit
Department of Informatics
Petra Christian University
Surabaya, Indonesia
e-mail: hnpalit@petra.ac.id

Abstract—**A computer laboratory is often a homogeneous environment, in which the computers have the same hardware and software settings. Conducting system tests in this laboratory environment is quite challenging, as the laboratory is supposed to be shared with regular classes. This manuscript details the use of desktop virtualization to deploy dynamically a virtual cluster for testing and ad-hoc purposes. The virtual cluster can support an environment completely different from the physical environment and provide application isolation essential for separating the testing environment from the regular class activities. Windows 7 OS was running in the host desktops, and VMware Workstation was employed as the desktop virtualization manager. The deployed virtual cluster comprised virtual desktops installed with Ubuntu Desktop Linux OS. Lightweight applications using VMware VIX library and shell scripts were developed and employed to manage job submission to the virtual cluster. Evaluations on the virtual cluster's deployment show that we can leverage on desktop virtualization to quickly and dynamically deploy a testing environment while exploiting the underutilized compute resources.**

Keywords-desktop virtualization; overlay virtual cluster; exploiting idle resources

I. INTRODUCTION

It is well known that the resource utilization in a data center is considerably low. According to an IBM Research Report [1], the CPU utilization of various data centers across continents ranges from 7% to 25%. Previous study [2] also found that the average daytime utilization of Windows servers is around 5%, whereas it is 15-20% among UNIX servers. Inquired by the New York Times [3], the consulting firm McKinsey & Company estimated that only 6% to 12% of the electricity powering the data centers is really used to perform computations. The majority of the electricity is used to keep servers idling and ready in case of a surge in activity.

Resource utilization in a university's computer laboratory can only be worse than that in a data center since the computers in the former are often turned on from morning till evening even though there is no class or practicum session going on. A computer laboratory is often a homogeneous environment, in which the computers have the same hardware and software settings. It would be an excellent environment for conducting system tests, if we could configure the computers freely. However, since it is supposed to be shared with regular classes, computer configurations in the laboratory often cannot be disturbed, lest changes to those configurations would bring some disruption to the regular classes. This condition further causes underutilization of computers in the laboratory. Domingues *et al.* [4] concluded that the average CPU idleness of desktop computers in computer laboratories was near 98%. Among workstation clusters, a study by Acharya *et al.* [5] found that their idle time (i.e., when a cluster of a particular number of workstations is free) was up to 80%. Han and Gnawali [6] asserted that 60% of energy consumed in a computer laboratory was wasted, because the computers were on and no one was logged in. Further, they studied the laboratory users' behavior and concluded that only 5% of users used the computers extensively (consuming more than 3,000 KJ of energy), whereas the majority (75%) consumed less than or equal to 1,000 KJ of energy. Hence comes the idea to utilize the laboratory's computers for other purposes. These idle resources have been targeted for harvest and used for analyzing data, rendering movies, running some simulation models, and so forth.

Computer laboratories in our department suffer the same inefficiency in resource utilization. The potential of this multitude of idle computers is huge as each machine has a four-core processor and at least 8 GB of memory. Separately, there is a growing demand for computer resources to analyze a large amount of data (often termed as big data). Although the computation demand is high, the resource requirements of big data analytics often can be fulfilled by off-the-shelf computers. Obviously we should be able to match these unfulfilled demands with those excess supplies.

Consequently, we devise a simple, lightweight, and non-invasive mechanism to deploy a virtual cluster on the idly running computers in our laboratory. The overlaid, virtual cluster does not disturb the existing computer configurations and in fact, if necessary, both schemes (i.e., the virtual cluster and the native physical computers) can run concurrently in the laboratory. The virtual cluster can be exploited to address the big data issues, perform a distributed computing batch job, or even conduct an extensive system test. Although there are available solutions for automated deployment of virtual machines, we opt for creating our own solution due to the following reasons:

- The existing solutions (e.g., cloud management tools) entail major configuration changes to the computers, such as replacing the existing OS with a bare-metal hypervisor.

- The existing solutions are commonly heavyweight requiring many services or libraries to be in place to support them.
- Most of the existing solutions do not support Windows as the host OS, while the use of Windows OS is essential in our computer laboratories to hold regular classes.

Our proposed solution can overcome the issues that come with the existing solutions. It leaves the existing Windows OS and software configurations on each computer intact, needs no extra services or libraries, and supports deployment and termination of the virtual cluster on demand. We employ VMware and our developed VIX-based [7] applications running on Windows 7 to deploy the virtual cluster.

The rest of the paper is organized as follows. Section 2 presents some related work. Section 3 describes our proposed solution. Evaluation on the solution is discussed in Section 4. We conclude our findings and suggest some future work in Section 5.

II. RELATED WORK

Harvesting idle compute resources has been the subject of many studies. After the success of SETI@home project [8], which has employed millions of voluntary computers worldwide to process radio signals in the search for extraterrestrial intelligence, the U.C. Berkeley Spaces Sciences Laboratory has developed a platform for public-resource distributed computing called BOINC (Berkeley Open Infrastructure for Network Computing) [9]. One of its goals is to encourage the world's computer owners to participate in one or more scientific projects by contributing their unused resources (e.g., CPU, memory, disk space, and network bandwidth) and specifying how the resources are allocated among the projects.

The Condor project (renamed to HTCondor in 2012) [10] was started in 1984 at the University of Wisconsin. Similar to BOINC, every Condor's participant has the freedom to contribute as much or as little as s/he wants. Three parties are involved in the system: agents, resources, and matchmakers. The agent enforces the submitting user's policies on what resources are trusted and suitable for the user's jobs. The resource enforces the machine owner's policies on what users are to be trusted and served. The matchmaker enforces communal policies like limiting a particular user to consume too many machines at a time. Unlike BOINC, which is just one large pool of compute resources, there are multiple Condor pools – which may or may not collaborate with each other – around the globe.

Some institution may have spare processing capacity on its desktop PCs and it can pool these resources to execute the institution-level applications. Such practice is often termed as desktop grid [11]. In this case, the desktops' participation is usually obligatory and governed by the institution's policies.

BOINC, Condor, and desktop grid require some services – at least the CPU-load and job monitoring agents – to be installed on the participating computers. The services are imperative to the decision-making and execution flow of a distributed computing system. However, these background running services would nip some resources from the participating computers, as they need to capture the current status and update the central server on a regular basis.

Other studies tried to exploit idle compute resources, either from a computer laboratory [5], [12], [13] or a data center [14], for parallel computation. In that case, there should be multiple computers available for a certain period of time to accomplish the parallel execution. Monitoring agents and adaptive scheduler are the key components in the scheme. The studies demonstrate that the scheme can work smoothly with minimal disturbance to the rightful jobs or users. However, sequential or embarrassingly parallel applications, as attested in Conillon [13], are still preferred as they impose the least impact.

Recently virtualization has been employed as the means to utilize idle compute resources. Compared to the physical environment, the virtual counterpart has added features like isolation, security, and fast deployment. HP Labs' researchers and their collaborators developed vCluster [15] and I-Cluster [16] that can switch workstation nodes between user-mode (Windows) and cluster-mode (Linux). A node in user-mode can automatically switch to cluster-mode when user idleness is detected, and likewise, it can switch back to user-mode when user presence is detected (or anticipated). The virtual cluster – comprising idle workstation nodes – can be used for various computation purposes, in isolation from the user's local data and applications. A different approach leveraged by NDDE [17] is to run virtual machines, in concurrence with the user's environment, to utilize the idle cycles. In that way, there is no need to switch between different environments. Both environments can run together independently, without interfering each other; the user may not even be aware of the presence of the virtual environment.

Meanwhile, the proliferation of cloud computing may drive server consolidation in many enterprises. Employing open-source cloud management software like Eucalyptus [18] and OpenStack [19], an enterprise can consolidate underutilized servers, and instead, deploy (virtual) servers on demand to improve the overall utilization of its compute resources. Subsequently, idle computer cycles can be used for accomplishing other computation tasks. While this approach is very effective in harvesting idle computer cycles, its adoption in our computer laboratory is impractical since the physical computers in the laboratory cannot be consolidated and should be left intact for the regular class activities.

Considering and evaluating all the above alternatives, we opted to employ desktop virtualization to utilize the idle computer cycles. The solution does not disturb much the existing computer configurations and can run in subtlety without interrupting normal laboratory usage.

III. OVERLAY VIRTUAL CLUSTER

In this section, we explain how the virtual environment is set up within a physical computer and how the virtual machines are assigned to form the virtual cluster. The virtual cluster is overlaid on top of the physical computers and network connectivity. Some illustrations are provided to explain our concept.

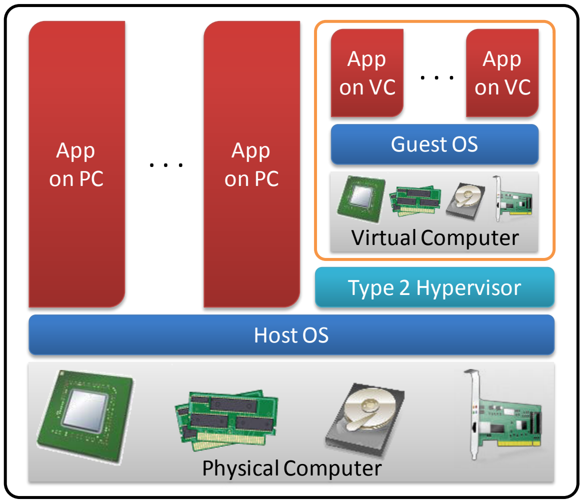

Figure 1. Physical and virtual environments within a computer

A. *Deploying the Virtual Environment*

For regular class and practicum activities, Windows 7 is the default OS running in our laboratory's computers. To facilitate a virtual environment, a desktop virtualization manager needs to be installed on each physical computer (PC). It employs a Type 2 hypervisor, which runs on top of the host OS. In our case, VMware Workstation is running on top of Windows 7 OS. A virtual computer (VC), a.k.a. virtual machine, can in turn be deployed with the hypervisor's help.

As seen in Figure 1, the physical computer can still be used by any user; existing applications (on PC) can run as per normal. The hypervisor can be considered as another application running on the physical computer. Once a virtual computer is deployed on top of the hypervisor, we can also run many applications (on VC) in the virtual domain. Hence, the physical and virtual environments can coexist together within a computer.

B. *Building the Virtual Cluster*

In a common computing cluster, one computer should act as the master whereas the rest are workers. Jobs (i.e., any computation tasks) are sent to the master node, which in turn distributes the jobs among the worker nodes and execute them remotely. By executing a job in parallel or concurrently, the job can be completed quickly. The master node also monitors the job's execution progress. Once the job is completed, results that are stored in the worker nodes are then collected by (or, sent to) the master node to be returned to the job's owner.

Similarly, in our virtual platform, as shown in Figure 2, one master node and multiple worker nodes can be dynamically deployed for executing computation jobs. The master node needs to be deployed first, and then followed by deploying the worker nodes. The master node has a list of active worker nodes. Each worker needs to register and deregister itself to that master's list when it is turned on and turned off, respectively. Therefore, the master node knows where to distribute and monitor the jobs.

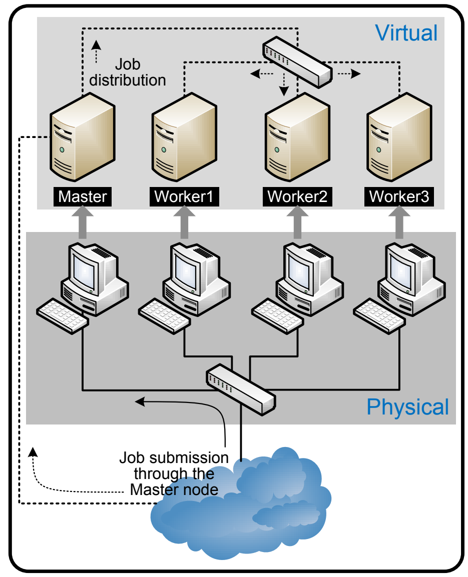

Figure 2. Physical and virtual clusters

The master and worker nodes in the virtual platform were running Linux OS. The operations to register and deregister a worker node were implemented in shell script, and so were those to distribute and execute jobs remotely. To automate the virtual cluster's (i.e., master and worker nodes) deployment, we developed lightweight applications using VIX (Virtual Infrastructure eXtension) API [7], which is a VMware library for writing scripts and programs to manipulate virtual machines. Multiple virtual machines may be hosted within a physical desktop. Thus, we have two deployment alternatives: one virtual machine per host and multiple virtual machines per host. We will evaluate both alternatives in the next section.

Figure 2 clearly illustrates the deployment of a virtual computing cluster on top of a physical cluster of desktops. Both platforms, physical and virtual, can run concurrently as far as the resources permit. In that way, the overall computers' utilization can be improved, and the laboratory's main responsibility – i.e., to support class and practicum sessions – is not disturbed. Moreover, the virtual computing cluster can be deployed on demand, at any convenient time, and perhaps remotely.

IV. SYSTEM EVALUATION

As a proof of concept, we had developed the system's prototype to demonstrate the benefits of the proposed scheme. Four desktops in our laboratory were used for the experiment. The specifications of each desktop are as follows:

- Processor: Intel Core i5-3340, quad-core, 3.1GHz
- RAM: 16GB
- Host OS: Windows 7 Professional SP1
- Virtualization manager: VMware Workstation 9.0.0 build-812388
- Guest OS: Linux Ubuntu Desktop 12.04

The above-mentioned VIX-based applications were developed and can be used by any user to direct and interface with the VMware Workstation Managers (residing in the physical desktops). The virtual cluster can be deployed dynamically, meaning that the user can specify which physical desktops to employ and how many worker nodes to be deployed on each machine. At the moment, the virtual cluster can be used to execute embarrassingly parallel jobs, which do not require much communication between processing components. The jobs can either be direct or indirect tasks. A direct task means the program (i.e., can also be a command or a script) is available and can be executed directly in the worker nodes; e.g., executing Unix commands `ls`, `ps`, `date`, `curl`, etc. If the program is currently available in the user's computer, and not available in the worker nodes, then it is considered an indirect task. When executing an indirect task, the would-be-executed program in the user's computer is firstly transferred to the master node, and then it is distributed to the worker nodes, and finally it is executed concurrently on each worker node. In either kind of task, all screen outputs are collected and returned to the user's computer requesting the job execution.

We do not evaluate the different execution timings between physical and virtual clusters, as it is common knowledge that execution in the virtual environment incurs some penalty (variably between 1% and 15%) compared to that in the native environment. Instead, we evaluate the time needed to deploy different configurations of virtual clusters. The purpose of this evaluation is to determine which deployment is the fastest to deploy and to execute.

Referring to Table 1, the different configurations of the virtual clusters are 1 master + 2 workers on 1 host (2-on-1), 1 master + 2 workers on 2 hosts (2-on-2), 1 master + 3 workers on 1 host (3-on-1), and 1 master + 3 workers on 3 hosts (3-on-3). The master node is always deployed first, exclusively in a physical desktop. Once it is ready, the worker nodes are started one after another. Two schemes are evaluated: sequential and concurrent strategies. In the sequential strategy, the next worker node is started only after the currently started worker node is in a ready state (i.e., the VMware Tools daemon is successfully started). By contrast, in the concurrent strategy, the next worker node is immediately started without waiting for the currently started worker node's readiness.

TABLE I. AVERAGE DELIVERY TIMES (IN SECONDS) OF DIFFERENT CLUSTER CONFIGURATIONS

Cluster Configuration	Sequential-Start		Concurrent-Start		Stop
	Master	Workers	Master	Workers	
2-on-1 (2 workers on 1 host)	15.57	47.12	15.20	27.63	14.02
2-on-2 (2 workers on 2 hosts)	15.36	46.78	15.91	23.80	12.90
3-on-1 (3 workers on 1 host)	15.88	72.60	15.73	36.16	14.81
3-on-3 (3 workers on 3 hosts)	15.28	67.21	15.57	27.61	13.12

The delivery times – measuring the time taken until the execution is completed (i.e., when all nodes are ready or when they all are down) – of different cluster configurations are presented in Table 1. Deploying the master node takes between 15 and 16 seconds. Deploying the worker nodes, on the other hand, takes a longer time. In the 2-on-1 sequential strategy, it takes about 47.12 seconds (or, roughly 23.56 seconds per node). The 3-on-1 sequential strategy yields a similar result, about 24.20 seconds per node. Deploying the worker nodes in different hosts improves the results slightly (0.7% and 7% improvements in the 2-on-2 and 3-on-3 sequential strategies, respectively). The concurrent strategies produce much better results than the sequential counterparts. The performance gains are quite significant (i.e., 1.7 – 2.0 times and 2.0 – 2.4 times faster for 2 and 3 worker nodes, respectively).

Terminating (stopping) the virtual cluster takes about 13-15 seconds. Generally, the terminations in multiple hosts yield slightly better delivery times than those in a single host (about 8% and 11% improvements in the 2-on-2 and 3-on-3 cases, respectively).

Lastly, we evaluated the job execution – both, direct and indirect tasks – in different cluster deployments. We executed various jobs, from shell scripts to simple programs. Executing jobs in different cluster deployments produces pretty consistent delivery times, no matter whether the jobs are direct or indirect. Thus, the job execution is not affected by the different cluster deployments.

V. CONCLUSION AND FUTURE WORK

Most computer laboratories have very low resource utilization. Various schemes had been proposed to improve the resource utilization. However, majority of the schemes require some changes to the computer installation, and some may nip a portion of the resources for running essential background processes. In a computer laboratory where the hardware and software settings shall not be disturbed, lest they cause disturbance to regular classes or practicum sessions, such schemes cannot be employed.

Leveraging on desktop virtualization, our proposed scheme is lightweight, can be deployed on demand and at any time, and does not disturb much the computer installation in the laboratory. Evaluation on different cluster deployments shows that deploying the virtual nodes in multiple hosts, in contrast to deploying them in a single host, can slightly reduce the delivery times. However, deploying the virtual nodes concurrently, in contrast to deploying them sequentially, can yield performance gain by a factor of around 2 (for up to 3 worker nodes). The delivery times of executing direct or indirect jobs are not affected by the different cluster deployments.

Presently, the user has to give commands to the VMware Workstation Managers (i.e., executing the VIX-based applications) through the Windows (DOS) command prompt window. We intend to improve this in future work by developing a Web-based user interface. Additional VIX-based applications are also in our plan to provide features like data transfers and scheduled job execution.

REFERENCES

[1] R. Birke, L. Y. Chen, and E. Smirni, Data Centers in the Wild: A Large Performance Study, IBM Research Report RZ3820, 2012, URI=http://domino.research.ibm.com/library/cyberdig.nsf/papers/0C306B31CF0D3861852579E40045F17F.

[2] D. G. Heap, "Taurus – A taxonomy of actual utilization of real UNIX and windows servers," Technical Report GM12-0191, IBM White Paper, 2003.

[3] J. Glanz, "The cloud factories: Power, pollution and the Internet," The New York Times, Sep 2012, URI=http://www.nytimes.com/2012/09/23/technology/data-centers-waste-vast-amounts-of-energy-belying-industry-image.html.

[4] P. Domingues, P. Marques, and L. Silva, "Resource usage of Windows computer laboratories," Proc. 34th Int. Conf. on Parallel Processing (ICPP) Workshops, Oslo (Norway), pp. 469–476, Jun. 2005.

[5] A. Acharya, G. Edjlali, and J. Saltz, "The utility of exploiting idle workstations for parallel computation," ACM SIGMETRICS Performance Evaluation Review, vol. 25, no. 1, pp. 225–234, Jun .1997.

[6] D. Han and O. Gnawali, "Understanding desktop energy footprint in an academic computer lab," Proc. 3rd IEEE Int. Conf. on Green Computing and Communications (GreenCom), Besancon (France), pp. 541–548, Nov. 2012.

[7] VMware, VIX API v1.12 Reference, URI=https://www.vmware.com/support/developer/vix-api/vix112_reference/

[8] D. P. Anderson, J. Cobb, E. Korpela, M. Lebofsky, and D. Werthimer, "SETI@home: An experiment in public-resource computing," Comm. ACM, vol. 45, no. 11, pp. 56–61, Nov. 2002.

[9] D. P. Anderson, "A system for public-resource computing and storage," Proc. 5th IEEE/ACM Int. Wksh. on Grid Computing (GRID), Pittsburgh (PA, USA), pp. 4–10, Nov. 2004.

[10] D. Thain, T. Tannenbaum, and M. Livny, "Distributed computing in practice: the Condor experience," Concurrency and Computation: Practice and Experience, vol. 17, no. 2–4, pp. 323–356, Feb.–Apr. 2005.

[11] N. Z. Constantinescu, "A desktop grid computing approach for scientific computing and visualization," Doctoral thesis, Norwegian University of Science and Technology, Trondheim (Norway), 2008.

[12] P. H. J. Kelly, S. Pelagatti, and M. Rossiter, "Instant-access cycle-stealing for parallel applications requiring interactive response," Proc. 8th Int. Euro-Par Conf., Padenborn (Germany), LNCS, vol. 2400 (eds.: B. Monien and R. Feldmann), pp. 863–872, 2002.

[13] H. Silva, A. L. Christensen, and S. Oliveira, "Performance study of Conillon – a platform for distributed computing," Proc. 2011 Wksh. on Open Source and Design of Communication (OSDOC), Lisboa (Portugal), pp. 13–18, Jul. 2011.

[14] D. Vyas and J. Subhlok, "Volunteer computing on clusters," Proc. 12th Int. Wksh. on Job Scheduling Strategies for Parallel Processing (JSSPP), Saint-Malo (France), LNCS, vol. 4376 (eds.: E. Frachtenberg and U. Schwiegelshohn), pp. 161–175, 2007.

[15] C. De Rose, F. Blanco, N. Maillard, K. Saikoski, R. Novaes, O. Richard, and B. Richard, "The virtual cluster: a dynamic environment for exploitation of idle network resources," Proc. 14th Symp. on Computer Architecture and High Performance Computing (SBAC-PAD), Vitoria (Brazil), pp. 141–148, Oct. 2002.

[16] B. Richard, N. Maillard, C. A. F. De Rose, and R. Novaes, "The I-Cluster cloud: Distributed management of idle resources for intense computing," Parallel Computing, vol. 31, no. 8–9, pp. 813–838, Aug.–Sep. 2005.

[17] R. C. Novaes, P. Roisenberg, R. Scheer, C. Northfleet, J. H. Jornada, and W. Cirne, "Non-dedicated distributed environment: a solution for safe and continuous exploitation of idle cycles," Scalable Computing: Practice and Experience (SCPE), vol. 6, no. 3, pp. 107–115, 2005.

[18] Eucalyptus, Eucalyptus Documentation v4.2.1, URI=http://docs.hpcloud.com/eucalyptus/4.2.1/

[19] OpenStack, OpenStack Configuration Reference (Liberty), URI=http://docs.openstack.org/liberty/config-reference/config-reference-liberty.pdf.

CERTIFICATE OF PARTICIPATION

Paper Title: Deploying an Ad-Hoc Computing Cluster Overlaid on Top of Public Desktops （S64)

This is to certify *Henry Novianus Palit* from *Petra Christian University, Indonesia* has attended and delivered an oral presentation on 2016 8th IEEE International Conference on Communication Software and Networks (ICCSN 2016) held in Beijing, China during June 4-6, 2016.

Conference Committee of ICCSN 2016

SIE IEEE CETC 54 ITD CN NATIONAL KEY LABORATORY