- Word Count: 4091

Plagiarism Percentage

10%

sources:

1	6% match (Internet from 21-Jul-2017) http://repository.petra.ac.id/17473/1/Publikasi1_14001_3016.pdf
2	1% match (Internet from 26-Apr-2013) http://www.morphlabs.com/wp-content/uploads/Morphlabs_Powering_the_Cloud.pdf
3	1% match (publications) Henry Novianus Palit. "Deploying an Ad-Hoc computing cluster overlaid on top of public desktops", 2016 8th IEEE International Conference on Communication Software and Networks (ICCSN), 2016
4	< 1% match (Internet from 12-Nov-2017) http://engineering.utsa.edu/rkrishnan/wp-content/uploads/sites/27/2017/08/Dissertation_Navid_Pustchi.pdf
5	< 1% match (Internet from 31-Aug-2016) http://va.gov/TRM/ToolPage.asp?minYear=2016&tid=8832
6	< 1% match (Internet from 28-Mar-2016) http://www.cryptopros.com/2014/04/cryptocurrency-charitable-giving.html
7	< 1% match (Internet from 20-Apr-2010) http://www.cmg.org/proceedings/2004/4238.pdf
8	< 1% match (publications) Lecture Notes in Computer Science, 2007.
9	< 1% match (Internet from 12-Jul-2015) http://www.foibg.com/conf/ITA2007/iTech2007/iTech07-Book2.pdf
10	< 1% match (Internet from 24-Mar-2016) http://robotcomp.ru/product/planshet-digma-plane-ts804h-3g-black-80-ips2-1024768a912-1gb-8gb-3g-gps-glonass-wi-fi-bt-cam-mhdmiandroid-41/

11 < 1% match (Internet from 09-Sep-2017)
http://theses.ucalgary.ca/jspui/bitstream/11023/1371/2/ucalgary_2014_almami_amal.pdf

12 < 1% match (Internet from 20-Nov-2006)
http://personality-project.org/perproj/online.html

13 < 1% match (Internet from 14-Mar-2016)
https://dokumente.ub.tu-clausthal.de/servlets/MCRFileNodeServlet/Document_derivate_00000176/Db112213.pdf

14 < 1% match (publications)
Devi, D. Sudha, K. Thilagavathy, S. Vaghula Krishnan, S. Harish, and R. Srinivasan. "Integrating OTP Authentication Service in OpenStack", Advanced Materials Research, 2014.

paper text:

Private Cloud Deployment on Shared Computer Labs Henry Novianus Palit, Agustinus Noertjahyana, Christian Adi Widjaja, Albert Halim

Department of Informatics Petra Christian University Surabaya, Indonesia hnpalit@petra.ac.id, 1

agust@petra.ac.id, renmuachci@gmail.com, lim_bert_lie@yahoo.co

.id Abstract A computer laboratory in **a** 3

school or college is often shared for multiple class and lab sessions. However, often the computers in the lab are just left idling for an extended period of time. Those are potential resources to be harvested for cloud services. This manuscript details the deployment of a private cloud on the shared computer labs. Fundamental services like operation manager, configuration manager, cloud manager, and schedule manager were put up to power on/off computers remotely, specify each computer's OS configuration, manage cloud services (i.e., provision and retire virtual machines), and schedule OS switching tasks, respectively. OpenStack was employed to manage computer resources for cloud services. The deployment of private cloud can improve the computers' utilization on the shared computer labs. Keywords-private cloud; shared computer labs; cloud services; OpenStack; Wake-on-LAN (WOL); Preboot eXecution Environment (PXE) I. Introduction Past reports suggested that utilization of computer resources

in a data center is generally **low.** 3

Based on collected data from worldwide data centers in 2009-2011,

an IBM research report [1] found that **the CPU utilization of** a typical **data** 1

center ranged from 7% to 25%. An analysis by McKinsey & Company on 70 large data centers, as reported by the New York Times [2], also showed that

on average data centers were using **6% to 12% of the electricity** for **their servers' computations;** apparently **the** large share **of power** was **to keep servers** 2

doing nothing, just be

ready in case of a sudden rise **in activity. A** 1

comparable phenomenon can also be observed in a school's or college's computer laboratory. Computers in that particular lab are often turned on for long hours even if there is no class or practicum session taking place. Some students may have the habit of using the computers for a while and then leaving without turning them off. Evaluating resource idleness in a number of Windows computer laboratories,

Domingues et al. [3] found **that the average CPU idleness** 1

was almost 98% and the portion of unused memory was 42% on average. Using wireless power meters and simultaneously recording the user activities on a lab of 22 computers, Han and Gnawali [4] concluded that every day 60% of energy consumed by each computer was left unused as no user was logged in. Moreover, the study on the lab users' behavior revealed that only 5% of users employed the computers for long periods of time (taking

more than 3,000 KJ of energy), whereas the majority (75% 1

of users) just occupied them minimally and in turn

consumed less than or equal to 1,000 KJ of energy. 1

Evidently, computers in many school/college labs are not optimally utilized. They are often left idle or unused for long periods of time. There are two general approaches to tackle this inefficiency issue. The first approach is to turn the computers off when they are not being used. This can be performed either remotely or automatically to reduce human involvement. The second approach is to harness the idle CPU cycles for addressing other computational needs. The needs are most likely to come from other parties within or outside the school/college. Unquestionably, the idle resources may be offered voluntarily or on a pay-per-use basis. Following the second approach, the research work discussed in this manuscript tries to servitize the extra resources of some shared computer labs and deploy them as a private Cloud service (i.e., Infastructure-as-a-Service). OpenStack [5],

the open source software for creating **a cloud service,** 14

was employed to manage the idle computer resources. Since OpenStack requires Linux as the operating system (OS), while the desktops in the shared computer labs are running Windows as the default OS, a management system is needed to control the switching from Windows to Linux and vice versa. The

rest of the manuscript **is** arranged **as follows. Section 2** outlines **some related works,** 8

particularly on various ways the idle computer resources are being utilized. The proposed management system is described and detailed in Section 3. Some evaluations

are discussed in Section 4. Finally, **Section 5 concludes the** findings **and** 9

suggests some future works. II. Related Works Harnessing unused computer resources has been the focus of many previous research works. The works outlined here are by no means exhaustive, but they can represent some of the ideas of utilizing unused computer resources. The underlying concept of Grid computing [6] is essentially resource sharing in multi-institutional virtual organizations (VOs). Resource owners (providers) from different institutions may pool their resources in a VO to be used up by participants of the VO. The sharing is highly controlled. Each provider may share resources in multiple VOs and subject them to constraints on when, where, who is allowed, and what can be done. Sharing relationships are often peer-to-peer (i.e., providers can also be consumers), can exist among any subsets of participants, and can be coordinated across many resources belonging to disparate institutions. Through VOs, groups of institutions are enabled to collaborate by sharing resources to achieve a common goal. The Grid architecture encompasses many

protocols, services, Application Programming Interfaces (APIs), and Software Development Kits (SDKs) 7

so that applications can be developed to run in the complex and dynamic execution environments. While Grid computing concerns mainly with access to large-scale (i.e., clusters or supercomputers) and inter-institutional resources, there is another approach called Desktop Grid [7] that scavenges idle desktop computers. Desktop grid is often implemented within an institution, although a public desktop grid platform is still possible.

The desktops' participation is usually mandatory **and governed by the institution's policies.** 1

Many institutions – such as academics, enterprises, and government agencies – hold a large number of desktops for their employees. They may gain benefit from exploiting the idle cycles, without additional server investment, for executing some institutional-backing applications. The UC Berkeley Spaces Sciences Laboratory developed

a distributed computing platform called BOINC (Berkeley Open Infrastructure for Network Computing) 6

[8] comprising public resources. The platform was established on the success of the earlier SETI@home project, famously known for exploiting about 1 million voluntary computers worldwide in the quest for extraterrestrial intelligence. BOINC can run on various OSes

(e.g., Windows, UNIX/Linux, Mac OS/X, 12

etc.). It provides tools that allow contributors to remotely install the client software on a large number of computers, and then link the clients to selected projects. Started in 1984, the Condor project [9] also gives freedom for every participant

to contribute as much or as little as s/he wants. Basically, there **are** 1

two kinds of users: producers (who offer resources) and consumers (who consume resources). In the Condor's kernel, producers are represented by resources while consumers by agents. Resources and agents must advertise themselves to another component, matchmaker, which is responsible for matching compatible resources and agents.

Unlike BOINC, which is just one large pool of computer **resources, there are** many **Condor pools – which may or may not collaborate with each other – around the** 1

world. The Condor project has since been renamed to HTCondor (in 2012). Past research works also

tried to harness **idle** computer **resources from a** 3

network or cluster of workstations [10]–[12] for parallel computations. In that case, a number of idle workstations should be available throughout the parallel execution. They demonstrated

that the scheme can work subtly **with** negligible **disturbance to the** legit 1 jobs and /or users.

Nevertheless, less network-bound jobs are

preferred as they impose lower impacts. Recently virtualization has been employed to exploit unused computer resources. Compared to the physical counterpart, the virtual environment offers valuable features such as isolation, security, and fast deployment. 1

I-Cluster [13] conducts real-time analysis of the machines' workload and deploys a virtual cluster, utilizing the most suitable set of machines, in response to a user request. The platform can automatically switch workstations

between user-mode (on Windows OS) and cluster-mode (on Linux 1

OS). Each mode has a separate working space. The normal condition is the user-mode. A workstation enters the

cluster-mode when user idleness is detected. Similarly, it can switch back to the user-mode when user presence is detected (or anticipated). 1

Taking a different route, NDDE [14] deploys

virtual machines, in concurrence with the user's environment, to exploit the idle cycles. 1

A similar approach was employed in [15]–[16].

In that way, there is no need to switch between different modes. Both environments – physical and virtual – can coexist together 1

in the same machine

without interfering each other. In fact, the user may not even be aware of the presence of the virtual environment. 1

Cloud computing deployment has grown strongly lately in many enterprises. Cloud computing is believed to simplify the IT infrastructure and drastically cut IT investment costs, while simultaneously maintaining business agility and flexibility. As some enterprises are still concerned with public cloud's security, private cloud platforms are rather preferred by those enterprises. Under this scheme, underutilized servers can be consolidated and replaced by just a few servers with higher specifications (i.e., more CPU cores, memory, and disk space). Thereafter, cloud management software like OpenStack [5], CloudStack [17], or Eucalyptus [18] should be installed to deploy (virtual) servers in place of the underutilized (physical) servers. Server consolidation is a sure way to improve the overall computers' utilization as well as maintainability, as attested in [19]–[20]. In those works, the cloud management software was employed to manage the resource pool. Virtual desktops were elastically deployed from the resource pool to meet educational and experimental requirements. Distributed platforms such as the Grid, BOINC, and HTCondor can harness idle computer resources from widely spread locations over the globe. However, the platforms may not guarantee a fully isolated environment to protect the underlying resources from a mischievous job, even though a sandbox may be used to harmlessly run any foreign job. Cloud computing and virtualization, in general, can provide a better isolated environment, as the hypervisor will intercept, from the guest virtual machines, all instruction calls – including malicious ones directed toward the host OS. That

is the – reason for our **use of** cloud management software **in** our research **work.** 13

The previous works [19]–[20] required two pools of computer resources: one was for the virtualization hosts and the other for the thin clients to access the deployed virtual machines. Different from them, we just use one pool of computer resources in the labs. In fact, class/lab activities will still use the physical computers, whereas (virtualized) cloud services will be provided for remote users only when the computers are not in use. Details of the mechanism are given in the next section. Operation Configuration Manager Manager Cloud Schedule Manager Manager ... ... Network of Computers Network of Virtual Machines Fig. 1 System architecture of a shared computer lab, providing physical and virtual computers. III. Shared Labs Architecture Two different OSes are needed to concurrently utilize the computer labs for daily class/lab activities and private cloud services. The default OS of the computers in the shared labs is Windows. By contrast, the OS required to deploy cloud services is Linux. Therefore, both OSes, Windows and Linux, were installed on each computer. To manage the computers in the shared labs, the following fundamental services – as shown in Fig. 1 – were set up: operation manager, configuration manager, cloud manager, and schedule manager. They are functional services. Implementation wise, each service may be put up in a single server, or alternatively, multiple services may be realized in a single server. A. Operation Manager This service is to control all computers in the shared labs. Using this service, every computer can be turned on, turned off, or rebooted remotely. Different tools were employed to construct this service. Wake-on-LAN (WOL) – a.k.a. Magic Packet Technology [21] – is an Ethernet standard that allows a computer to be turned on, or awakened, by a network message. This feature is supported by most Ethernet cards and motherboards. The operation manager leverages this WOL feature to turn on any computer in the labs. The most important argument required by the WOL command is the Media Access Control (MAC) address of the Ethernet card attached to a computer. Thus, the paired list of MAC addresses and computerIDs needs to be maintained by the operation manager. Depending on the currently active OS on a remote computer, the operation manager has different ways to turn off or reboot the machine. On a Windows machine, the operation manager employs Samba net (rpc) utility to turn it off or reboot it, whereas the shutdown command – executed remotely through ssh (secure shell) – is used to achieve the same goal on a Linux machine. In both ways, the machine's Internet Protocol (IP) address or domain name is required. The operation manager can also report the current status (ON or OFF) and guesstimate the active OS of each computer in the shared labs. To accomplish this, the operation manager will do these steps: 1. Try to connect to port 22 (ssh) of the machine's IP address. If it is successful, then the machine is ON and its active OS is Linux. 2. Try to connect to port 3389 (rdp) of the machine's IP address. If it is successful, then the machine is ON and its active OS is Windows. 3. Otherwise, it is inferred that the machine is OFF. B. Configuration Manager This service is to record the currently assigned OS (i.e., Windows or Linux) for each computer in the shared labs. When a computer is turned on or rebooted, it must consult with this service and boot the assigned OS accordingly. Different tools and protocols were employed to construct this service. Preboot eXecution Environment (PXE) [22] is a standardized client-server mechanism to boot a software assembly, retrieved from the network, on a client machine. It requires a PXE-capable Network Interface Card (NIC) on the client side and standard network

services such as Dynamic Host Configuration Protocol (DHCP) and **Trivial File Transfer Protocol (TFTP).** 5

Details of the mechanism is beyond this manuscript's scope; interested readers are referred to [22] for further information. The PXE feature can be enabled on the labs' computers. Thus, when started off, each computer will retrieve its associated PXE configuration file (i.e., assigned based on the computer's MAC address), run the script and, consequently, select one of the local OSes to boot. Some scripts were created for the configuration manager to switch the assigned OS (from Windows to Linux and vice versa) for particular computers by changing their respective PXE configuration files. The OS switching action may be asked by the system administrator or by a schedule task; the schedule manager, which executes various schedule tasks, will be discussed in the later subsection. C. Cloud Manager This service is to manage the pool of unused computers for provisioning private cloud services. The computers in the pool must run Linux OS, as required by the cloud management software. The cloud management software being employed is OpenStack [23]. OpenStack consists of many modules, and the basic ones forming the cloud manager

are: • Keystone – identity service, • Nova – compute service, • Neutron – networking service, • Glance – image service, and 4

• Horizon – dashboard. Details of the implementation and the evaluation of OpenStack deployment in our shared computer labs can be found in [24]. Through this cloud manager, a user may request a number of (virtual) computers for computations or experiments. The cloud manager also controls user access (i.e., when and who is allowed) and handles reservation requests. Once the jobs are done, the user may release the (virtual) computers back to the pool. D. Schedule Manager This service is to create and execute the schedule tasks of switching the OS of particular computers. The schedule tasks are usually created by the system administrator to automatically switch the computers' OS to Linux (i.e., when there is no class/lab activity) or to switch them back to Windows (i.e., when a class/lab activity is slated to start soon). A schedule task comprises of the scheduled date and time, selected OS, description, status, and targeted computerID. All schedule tasks are stored in database. A cron job regularly checks the active schedule tasks and, when the time comes, executes them by sending commands to the other services. An OS switching command is sent to the configuration manager, followed by a reboot command sent to the operation manager. Afterwards, the status is updated accordingly. IV. Implementation and Evaluation A. Implementation of Shared Labs Management System As per designed, the fundamental services were realized for constructing the proposed management system. The required tools, as discussed in the previous section, were installed and configured. Scripts were devised to bundle the execution of a sequence of commands and to interface between services. In addition, a Web application was developed for the system administrator to easily and centrally manage the computers in the shared labs. All fundamental services can be configured and executed from this Web interface. A screenshot of the Web interface, through which the OS can be assigned and the power on/off command can be sent to selected computers, is presented in Fig. 2. The fundamental services were installed in a single server. Windows and Linux OSes were installed in all computers being managed. The Kernel-based Virtual Machine (KVM) Fig. 2 Web interface to manage the labs' computers and the OpenStack's Nova client were also installed in the managed computers, so virtual machines can be created and deployed through the OpenStack cloud service. All computers used for this research work have the same specification, i.e.: Processor :

Intel Core i5-3340, 3.1 GHz (4 cores) 10

Memory : 16 GB Harddisk : 500 GB NIC : Broadcom NetXtreme Gigabit Ethernet (WOL and PXE are supported) Although the NICs are 1 Gbps Ethernet, our network switch can only support 100 Mbps interconnection network. It is beyond our power to upgrade the network infrastructure. B. System Evaluation The developed management system has been tested in our shared computer labs. It can greatly reduce the system administrator's workloads as the computers can be controlled (i.e., powered on, powered off, rebooted, or OS switched) from anywhere within our campus, without the need to be present in front of the computers. Table I shows the time taken for a computer to respond to power-on and power-off commands. As seen in the table, the Linux computer can respond to the power-off command slightly faster than the Windows computer. The power-on command using WOL is responded almost immediately by the computer. TABLE I RESPONSE TIME OF POWERING A COMPUTER OFF/ON OS Powered OFF Powered ON Windows Linux 0.60–0.80 s 0.30–0.50 s 0.020–0.025 s The configuration manager and the schedule manager also have been tested rigorously. The computer can boot up (or reboot) correctly the assigned OS, whether it is Windows or Linux, on the scheduled time and without human intervention. Through the cloud manager, user requests for (virtual) computers can be fulfilled without any issue. The (virtual) computers can be accessed remotely within our campus for executing scientific computation, simulation, or any other experimental work. Later on, the (virtual) computers can be released back when the user is done with them. Since the computer resources in the labs are still prioritized for the class/lab activities, the execution of a large job on the (virtual) computers are not recommended, lest it is aborted early due to the incoming class/lab activity. V. Conclusion The low utilization of a computer lab is a common phenomenon in most schools and colleges. Meanwhile, the demand for computation keeps expanding and escalating, especially in this era of big data. On the one hand, we have a supply of unused or idle computers. On the other hand, we have a high demand for computation. The private cloud deployment on the shared labs, as proposed in this manuscript, can help in meeting the demand for computation, without the need to invest in new servers. At the same time, the private cloud deployment also improves the utilization of computers in the shared labs. The research work presented in this manuscript is a working in progress. For the future works, we intend to explore the incorporation of OpenStack's Swift and Cinder (i.e., the cloud storage services) into our shared labs platform. Implementation-wise, we want to put up some frequently requested service applications on the platform. An example that crosses our mind is some big data analytics framework. Acknowledgment This research project was funded by an Applied Product Research Grant,

received in 2017,

Indonesia. We thank the Center of Research, Petra Christian University, for the support and guidance.

References [1] R. Birke, L.Y. Chen, and E. Smirni, “Data centers in the wild: a large performance study,” IBM Research Report RZ3820, 2012. [Online] Available: http://domino.research.ibm.com/library/ cyberdig.nsf/papers/0C306B31CF0D3861852579E40045F17F. [2] J. Glanz, “The cloud factories: power, pollution and the internet,” The New York Times, Sep. 2012. [Online] Available: http://www.nytimes.com/2012/09/23/technology/data-centers- waste-vast-amounts-of-energy-belying-industry-image.html. [3] P. Domingues, P. Marques, and L. Silva, “Resource usage of windows computer laboratories,” Proc. 34th Int. Conf. on Parallel Processing (ICPP) Workshops, Oslo (Norway), Jun. 2005, pp. 469–476, doi: 10.1109/ICPPW.2005.77. [4] D. Han and O. Gnawali, “Understanding desktop energy footprint in an academic computer lab,” Proc. 8th IEEE Int. Conf. on Green Computing and Communications (GreenCom), Besancon (France), Nov. 2012, pp. 541–548, doi: 10.1109/ GreenCom.2012.77. [5] OpenStack, “Open source software for creating private and public clouds.” [Online] Available: https://www.openstack.org. [6] I. Foster, C. Kesselman, and S. Tuecke, “The anatomy of the Grid: enabling scalable virtual organizations,” Int. J. High Performance Computing Applications, vol. 15 (3), pp. 200–222, Aug. 2001, doi: 10.1177/109434200101500302. [7] N.Z. Constantinescu-Fülöp, A Desktop Grid Computing Approach for Scientific Computing and Visualization, Doctoral thesis, Norwegian University of Science and Technology, Trondheim (Norway), 2008. [8] D.P. Anderson, “BOINC: a system for public-resource computing and storage,” Proc. 5th IEEE/ACM Int. Wksh. on Grid Computing (GRID), Pittsburgh (PA, USA), Nov. 2004, pp. 4–10, doi: 10.1109/GRID.2004.14. [9] D. Thain, T. Tannenbaum, and M. Livny, “Distributed computing in practice: the Condor experience,” Concurrency and Computation: Practice and Experience, vol. 17 (2–4), pp. 323–356, Feb–Apr 2005, doi: 10.1002/cpe.938. [10] 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., Paderborn (Germany), Aug. 2002, Springer LNCS, vol. 2400, B. Monien and R. Feldmann, Eds., pp. 863–872, doi: 10.1007/3-540-45706-2 _122. [11] 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), Jul. 2011, pp. 13–18, doi: 10.1145/ 2016716.2016720. [12] D. Vyas and J. Subhlok, “Volunteer computing on clusters,” Proc. 12th Int. Wksh. on Job Scheduling Strategies for Parallel Processing (JSSPP), Saint-Malo (France), Jun. 2006, Springer LNCS, vol. 4376, E. Frachtenberg and U. Schwiegelshohn, Eds., pp. 161–175, doi: 10.1007/978-3-540-71035-6_8. [13] 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 (8–9), pp. 813–838, Aug–Sep 2005, doi: 10.1016/j.parco.2005.06.001. [14] 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 (3), pp. 107–115, 2005. [15] H. Severini, H. Neeman, C. Franklin, J. Alexander, and J.V. Sumanth, “Implementing Linux-enabled Condor in Windows computer labs,” Proc. IEEE Nuclear Science Symposium and Medical Imaging Conf. (NSS/MIC), Dresden (Germany), Oct. 2008, pp. 873–874, doi: 10.1109/NSSMIC.2008.4774533. [16] H.N. Palit, “Deploying an ad-hoc computing cluster overlaid on top of public desktops,” Proc. 8th IEEE Int. Conf. on Communication Software and Networks (ICCSN), Beijing (China), Jun. 2016, pp. 693–697, doi: 10.1109/ICCSN.2016. 7586613. [17] CloudStack, “Apache

CloudStack: open source cloud computing.” [Online] Available: https://cloudstack.apache.org. [18] Eucalyptus, “Official documentation for Eucalyptus cloud.” [Online] Available: https://docs.eucalyptus.com/eucalyptus/ latest. [19] W. Gao, Y. Dong, and A. Li, “Utilization of cloud computing and virtualization technology in university public computer laboratory,” Proc. 5th Int. Conf. on Computational and Information Sciences (ICCIS), Shiyang (China), Jun. 2013, pp. 1513–1516, doi: 10.1109/ICCIS.2013.398. [20] F. Luo, C. Gu, and X. Li, “Constructing a virtual computer laboratory based on OpenStack,” Proc. 10th Int. Conf. on Computer Science & Education (ICCSE), Cambridge (UK), Jul. 2015, pp. 794–799, doi: 10.1109/ICCSE.2015.7250353. [21] AMD, “Magic Packet Technology,” AMD White Paper #20213, Nov. 1995. [Online] Available: https://support.amd.com/ TechDocs/20213.pdf. [22] Intel Corporation, “Preboot Execution Environment (PXE) specification – version 2.1,” Sep. 1999. [Online] Available: http://www.pix.net/software/pxeboot/archive/pxespec.pdf. [23] K. Jackson, C. Bunch, and E. Sigler, OpenStack Cloud Computing Cookbook, 3rd ed., Packt Publishing, Aug. 2015. [24] A. Noertjahyana, H.N. Palit, “Implementation of Private Cloud for Optimization of Computer Resources at University,” Proc. Int. Conf. on Research and Innovation in Computer Engineering 2017, JTEC, vol. 10, to be published. and Computer Sciences (RICCES), Langkawi (Malaysia), Aug.