Towards a Private Cloud Infrastructure Implementation for a University delivering IT curriculum using OpenStack

Ray Carlo Añonuevo
Information Technology Department
College of Computer Studies
De La Salle University
Manila Philippines
ray_anonuevo@dlsu.edu.ph

Dea Ferrer
Information Technology Department
College of Computer Studies
De La Salle University
Manila Philippines
dea_ferrer@dlsu.edu.ph

Fernito Llanita
Information Technology Department
College of Computer Studies
De La Salle University
Manila Philippines
fernito_llanita@dlsu.edu.ph

Miguel Mercado
Information Technology Department
College of Computer Studies
De La Salle University
Manila Philippines
miguel_angelo_mercado@dlsu.edu.ph

Danny C. Cheng
Information Technology Department
College of Computer Studies
De La Salle University
Manila Philippines
danny.cheng@dlsu.edu.ph

ABSTRACT

The use of cloud computing has grown from personal day-to-day use to become an intrinsic part of operations of all sizes of organizations and enterprises. Whether it be infrastructure, platform, or software as a service, the use of cloud computing has become pervasive and preferred over traditional data centre operation models mainly due to its flexibility and scalability. In this paper, we discuss the considerations and some design recommendations in adopting a private cloud based infrastructure for an educational institution in support of its academic activities. Use cases such as laboratory work and discussion, to research and self-learning of new tools and technologies will be discussed as they require a higher level of flexibility to be afforded to the students that are not commonly practiced in a more stringent environment such as in a company. Results based on experiences on performance, compatibility of tools, as well as simulated work patterns will also be discussed showing the complexity as well as the applicability of cloud infrastructure to an academic institutions information technology program.

CCS CONCEPTS

• **Applied Computing** → **Enterprise Computing** → **Enterprise Information System** → *Data Centers*; • **Computer Systems Organization** → **Distributed Architectures** → Cloud Computing

KEYWORDS

Cloud Computing, OpenStack, Data Center Architecture

ACM Reference format:

R. C. Anonuevo, D. Ferrer, F. Llanita, M. Mercado, D. C. Cheng. 2017. PCSC Proceedings Paper in word Format. In *Proceedings of Philippine Computing Science Conference, Cebu City, Philippines, March 2017 (PCSC'2017)*, 9 pages.
DOI:

1 INTRODUCTION

In the last decade, there has been a high level of growth in the use of cloud computing in the day-to-day operations of various data centers in enterprises at various levels of acceptance. Looking at the history and the evolution of the data centers to a cloud architecture, data centers started out with large main frame centralized computers, then miniaturizing into client-server architectures, then adopting virtualization technology to improve utilization, and now cloud technology to allow for greater flexibility and scalability [1].

The concept of cloud computing is still evolving and should not be misunderstood as simply the use of virtualization technologies. In a cloud environment, the biggest benefit would refer to its ability to support user self-administration and self-service thus reducing the need for administrative intervention, as well as, its ability to scale as needed based on certain pre-configured parameters thus improving resource sharing and utilization overall [1]. Commonly available and used cloud computing approaches can be classified into three namely: Software-as-a-Service (SaaS), Platform-as-a-Service (PaaS), and Infrastructure-as-a-Service (IaaS) Fig. 1.

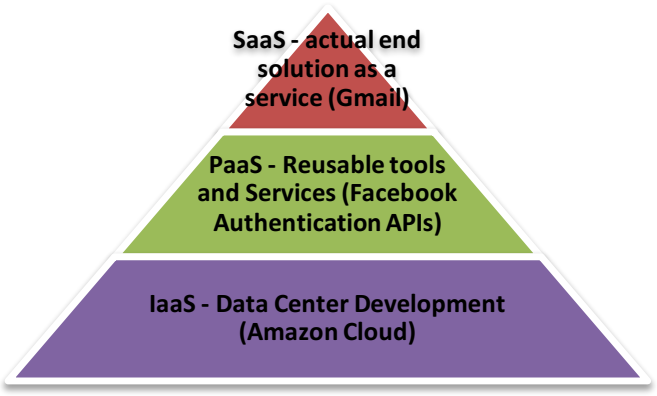

Figure 1: Cloud Computing Approaches, their relationships and uses.

Aside from the various approaches to cloud computing, there are also various models of deployment when working with the cloud. These models differ on the location of the actual components that make up the cloud deployment. The models are as follows [2]:

1. Public Cloud – outsourced to a third party and used strictly like a utility or pay-per-use.
2. Private Cloud – developed, deployed, and maintained within the organization. Usually the most difficult one to implement specially if IT is not the core business of the organization.
3. Community Cloud – shared cloud within a group of organizations.
4. Hybrid Cloud – a combination of public and private cloud sometimes due to legal or compatibility requirements

In this paper, we consider the deployment of a private IaaS for an academic institution where the courses to be supported are those that deal with Information Technology. This is a complex and unique domain as it does not deal with the traditional Learning Management System (LMS) deployment discussions for an academic institution nor does it strictly follow a development cloud environment that an enterprise would deploy due to the volume and paradigm of use. A private cloud deployment mode was selected due to its appropriateness towards the usage pattern and behavior of a university. Appropriateness was determined based on the following characteristics of a university [3,4]:

1. There is a large amount of computing machines in a university that is not used optimally.
2. The IT administrators of the university are having a lot of difficulties in the management of the requirements of the students and research projects.
3. The rate of change and the level of control needed by students, faculty, and administrators requires the cloud to be fully accessible.
4. The volume of use in a university makes a public cloud less attractive in terms of expenditures.

There are several open source IaaS solutions available however, OpenStack stands out in terms of service oriented architecture due to its design and its decoupling of components [5-8].

2 Use Cases

For this research the following use cases are considered in the design and evaluation of the cloud implementation.

2.1 Location and Time of Use Considerations

In a university setting, the location and time of use can be classified into 4 different location and time respectively:

1. Use in a class or laboratory – students and faculty would use the cloud simultaneously on a per class size level. This scenario means that heavy use would be encountered. A high utilization rate is to be expected as multiple concurrent sessions will be on-going where by each session would have a high requirement of resources due to the class activity being conducted.

2. Use within the campus – this would be a more moderate use of the cloud as it would assume that the number of sessions will not be the same as during peak class sessions and the load would also be moderate as well as the students would not all be performing heavy work as they are outside of class.

3. Use outside of campus – this is a low resource utilization scenario on its own but it allows for the usage to happen at any time of the day. Usage level is also limited by the speed of the connectivity that the students would have outside of campus. Although this is considered low in terms of resources requirements, it can be noted that this scenario can be observed in conjunction with the first 2 which would make it an added requirement to a potentially heavy utilization already.

4. Peak use during submission deadlines – a unique situation of a university as in this scenario, situations like the end of the academic calendar would trigger unusually heavy use as all of the students and faculty involved would have to submit their respective requirements at this time.

2.2 Tools to be supported based on roles

Given that the assumption of the cloud installation is for the support of Information Technology course requirements, a wide variety of support requirements become mandatory. In this research we look at the tool requirements based on the following roles or perspectives:

1. As a systems developer – this would require tools support for things like development environments, databases, desktop environments to test the system,

and internet connectivity. Traditionally this can be handled thru simply the use of virtualized desktop environments that will allow the students to install and deploy the tools that they would need. However, given recent changes in systems used in industry, development is no longer limited to traditional desktop and web applications but rather they now include emulated environments such as those used for developing mobile applications whereby the software can be installed and tested on an emulated mobile device which is in itself a virtualized environment.

2. A systems administrator – unlike a developer, an administrator would need full administrator privileges over the environment to allow the students to experience the entire systems lifecycle from installation and setup all the way to monitoring and maintenance. This environment requires proper isolation or encapsulation in order to prevent unintended actions from affecting other users of the system. Encapsulation not only happens at the hosts but also at the network level where by the student is given the flexibility of setting up its own virtualized private cloud data center without running the risk of conflicts with other environments on the same cloud.

2.3 Resource Management Considerations

In managing the resources of a university, efficiency is not as simple as just measuring the percentage of CPU utilization at a given time. Management of resources includes considerations for the following common requirements and scenarios in an academic institution:

1. There is a need to ensure that all the necessary hardware and software requirements are made available based on the need of the specific courses.

2. Troubleshooting of machines or installations should be made efficient and administrator support should be kept to a minimum even if there is potentially a high probability that such installations can be corrupted as part of the learning process of the students.

3. Flexibility in terms of resource allocation should be provided as students would require various different configurations of both hardware and software given the different set of courses that they take up within a term.

3 The OpenStack Private Cloud Design Implementation

The basic architecture of OpenStack is composed on a compute, networking, and storage node, a set of shared services, and an administrative dashboard Fig. 2.

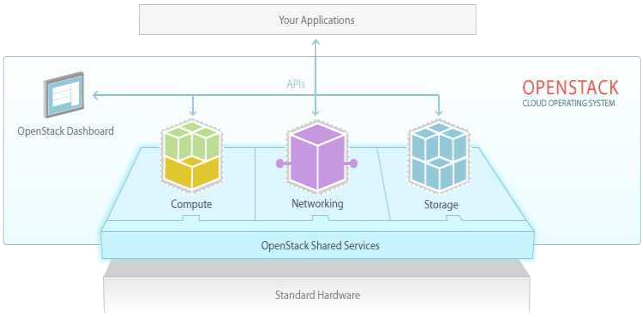

Figure 2: OpenStack basic architecture

The different nodes in OpenStack are given specific functions and tasks to perform and they are as follows:

1. Controller Node - The controller node is the center of the architecture. It handles all the requests from users and is responsible for resource allocation. The node is also responsible for user authentication and authorization as it provides and stores key pairs for users. The controller node caters to different services vital to the infrastructure; services such as Horizon (Dashboard), Keystone (Authentication service), Nova (Compute Service), Neutron (Networking service), and Glance (Image service). Aside from the mentioned services, the controller node is highly customizable giving the administrators the capability to include other services that might suit the organization's needs. This node houses a database (MySQL) that stores the metadata of each service and the Messaging Queue (RabbitMQ) which allows communication across the different OpenStack services.

2. Compute Node - The compute node handles the entire resource requirement for the whole architecture. The resource pool of the architecture is defined by how big the compute node would be. The architecture utilizes present resources (hardware specification) of the compute node such as random access memory, cores, and sometimes storage. The controller node uses the compute node API to communicate and utilize the node's resources.

3. Storage Node - The storage node supports the compute nodes' storage needs. By using logical volume management or other storage management methods, the storage node provides users with a self-service API for requesting persistent volumes. The storage node communicates with the instances via the ISCSI protocol.

4. Network Node - The network node provides the cloud with virtual networking and networking services to nova instances using the neutron services (L3 and DHCP Agents). In addition, to that the neutron API can also allow administrators to define security groups for

instances that enable or disable certain protocols/network traffic from reaching the instances.

3.1 OpenStack architecture customized

In order to address the uses cases as stated in Sec. 2 previously, the OpenStack standard architecture was customized and deployed based on the specific need. For the first use case, the concern is mostly on the amount of workload that the cloud can accommodate given that there will be regular repeated peak usage of the resources. In Fig. 3, the private cloud is structured to have multiple replicated nodes where the databases for the control nodes as well as the data in the storage nodes can be replicated using internal support from OpenStack such as swift.

The control nodes would handle the request of the students and provide the services that would be found on the OpenStack dashboard. The compute nodes would be the one providing the resources to the different request and would be connected to a corresponding block storage which would handle different storage needs of each user. The configured control nodes will be connected to a separate network node to handle networking services needed for OpenStack. Network services, as of the recent update of OpenStack (Liberty and above), has implemented it to be incorporated within each node, thus having different nodes to have a network service within them. The entire architecture will be connected to an external network which would be the internet service provider chosen by the organization. The cloud service would then be accessible through a dashboard that comes with the configured control node.

The current implementation has the cloud architecture deployed on top of a hypervisor environment to replicate actual resource usage for given use-cases of the project. Each of the nodes would be running under a hypervisor environment as the research wanted to lessen the utilization of resources by providing a bare-metal operating system that would have the least impact in resource needed for that node to function.

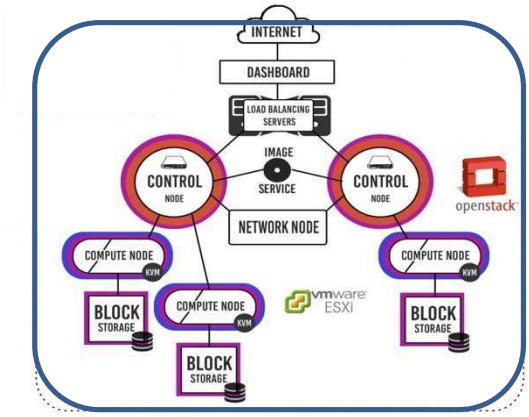

Figure 3: OpenStack customized general architecture

With regards to the various tools required by a system developer, the research implemented several templates or flavors of virtual machines within the cloud in order to test the behavior and performance of the developer tools within the virtualized cloud environment. The flavors implemented ranges from basic programming courses that would use simple tools such as a C compiler or Python interpreter, to complicated web and mobile applications that would be developed and used by students in their higher years within the curriculum. Tools such as programming environments (e.g. NetBeans, see Fig. 4) were also deployed to the cloud in order to test the usability of such tools in the cloud. In order to support the wide variety of tools, the research used KVM as its virtualization technology as when given a full stack for a virtual personal computer, it would minimize the possibility of incompatibilities or lack of support for needed developer tools.

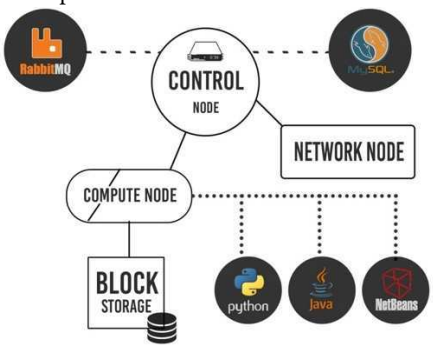

Figure 4: Basic programming development tools as deployed in the cloud

A similar approach was done to deploy advanced development tools into the cloud as seen in Fig. 5. However, it was realized that tools such as the emulator tools for mobile application development testing were not compatible with the virtualization technology. Running a virtualized environment inside a virtualized desktop is not allowed. As a workaround, the mobile device emulator was installed as a separate virtual machine with its own IP address that would allow the developer tool to deploy the program to the emulator.

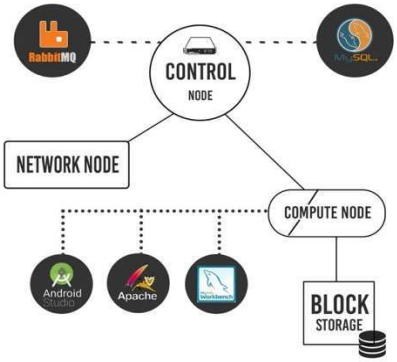

Figure 5: Advance programming development tools as deployed in the cloud

On the role of system administrator, the initial deployment done was that of a database administrator. The configuration in Fig. 6 is used both for database programming and administration courses. The main difference lies in a more extensive use of the block storage nodes for this requirement. The amount of workload differs from just compiling or executing programs. Database queries tend to deal with large amounts of data that directly affects processing time due to input and output speeds. This results to the block storage nodes being utilized more wherein instances for this class shall be booted in volumes so that the read and write functions would be handled more efficiently due to storage nodes have fast storage backend that deal with high I/O loads. Not to mention the possibility of extending the storage capacities as needed due to the nature of the activities involved in database courses. Aside from the block storage node, isolation for administration courses is also key, as such, the setup for such courses would have an isolated network to prevent problems and conflicts from spilling over to other environments (see Fig. 7). Such isolation is needed for cases like Active Directory administration or Domain and Route configuration topics.

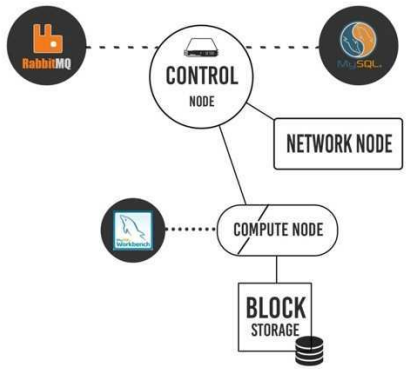

Figure 6: Database programming and administration tools as deployed in the cloud

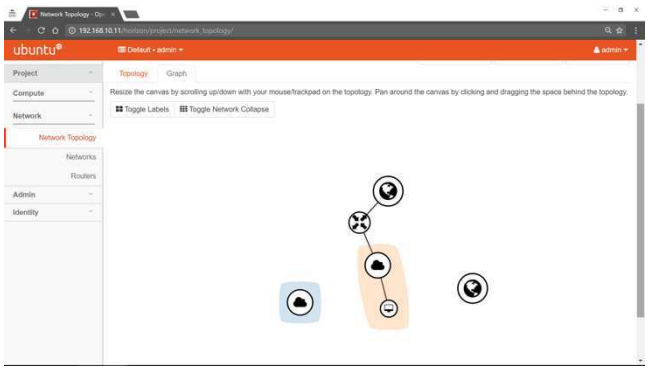

Figure 7: Isolated private networks for each individual in order to prevent conflicts

Lastly, on the use case or concern on resource management, to solve the availability of resources whenever and wherever is needed based on specific course requirements, OpenStack allows for the creation of images that serves as templates for creating instances. These images can and is to be pre-configured for the requirements on a per course basis (see Fig. 8). In doing so, students and faculty for a course would always have access to the needed software requirements without the need to install everything themselves. In the event the updates are needed, the use of images also allows the quick deployment of changes as if the images are updated, instances created after would be automatically updated as well.

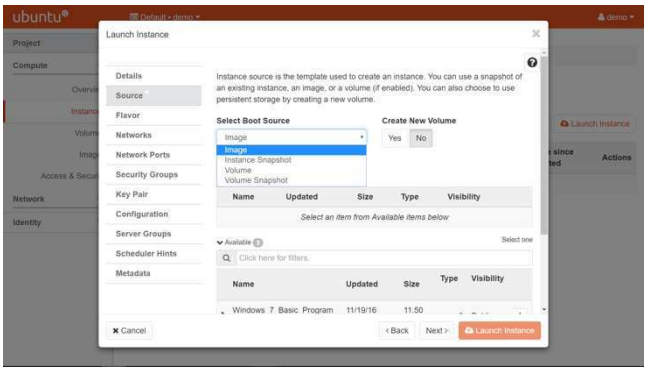

Figure 8: Instantiating course specific instances thru the use of pre-defined images.

On the aspect of resource allocation flexibility, each individual can be given resource quotas that allow for flexibility as well as better utilization and control of resources. Pre-defined virtual machine specifications can also be defined thru the use of flavors within OpenStack to control and prevent unnecessary or over provisioning for the instances that will be created (see Fig. 9 and 10).

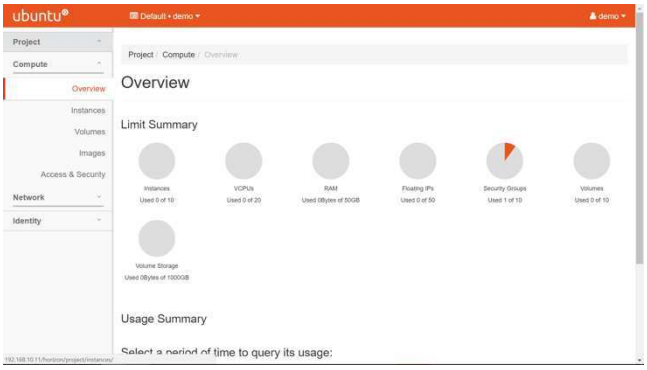

Figure 9: Monitoring resource utilization thru the individual dashboards.

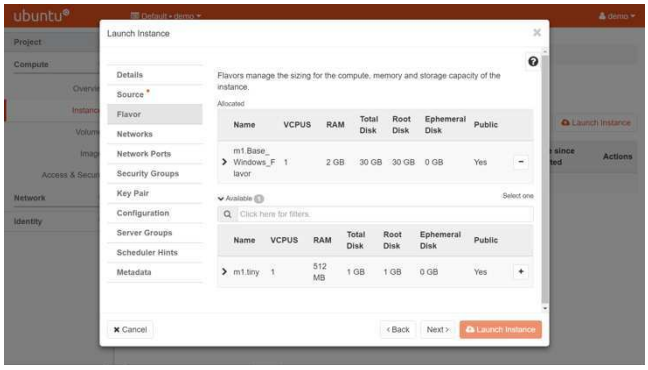

Figure 10: Instantiating new virtual machines with pre-defined flavors containing resource use specifications.

Aside from monitoring and control, OpenStack also allows for elasticity as well as support for over provisioning. The concept lies with the assumption that not every use will fully utilize the resources allocated to the instance while others might temporarily require more resources for a certain period of time. Lastly, on the need for troubleshooting support, there is an internal support for the creation of snapshots for each instance that would allow each individual to recover or prevent problems on their own without the need to seek support from helpdesk or administrators.

3.2 Support structure in the customized architecture

One of the support mechanisms that is provided by OpenStack in order to allow the creation of instances from images is the implementation of Glance Image Stores. In this research, the Glance Image Stores were utilized to store all the default or pre-configured images for each of the course requirements. During the instance creation, the Glance Image Store is accessed and a copy of the image is copied over to serve as the new instance on existing cinder volumes (see Fig. 11).

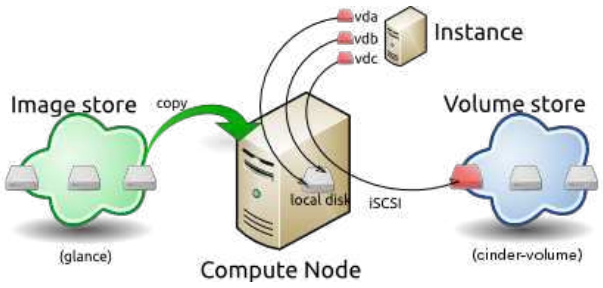

Figure 11: Glance Image Store in the process of creating a new Instance from an Image. (http://docs.openstack.org/admin-guide/_images/instance-life-2.png)

4 ISSUES AND OBSERVATIONS

In order to validate the assumptions and designs of the OpenStack deployment, an prototype test environment was setup. The setup is composed of 2 servers running IBM X3100 M4 Server with an Intel Xeon E3 3.1Ghz Quad-Core CPU, 16GB DDR3 1600MHz RAM and 1 TB HDD. The 4 OpenStack nodes namely Controller, Compute, Storage, and Network are virtualized running on top of a VMWare ESXi Hypervisor. Physically, the Compute node was given its own physical server node while the Storage, and Controller node are merged into the 2nd physical node. The Network node was deployed as part of the Compute and Storage nodes. Test was done in a laboratory environment thus the speed of connectivity is not currently considered as a factor.

Upon testing the customized architecture, the different flavors to support different types of classes previously identified were determined (see Table 1). A flavor in OpenStack is defined as a resource allocation preset which is assigned for each instance. A flavor is configured by the administrator which consists of basic values such as number of cores, amount of virtual RAM and amount of storage. The use of different flavors allows administrators to efficiently allocate the organization's resources. The basis in determining the amount of vCPU, vRAM, and storage size are the recommended system requirements of the tools used in class. An example would be the specifications for the *Advance Development Environment Class* which uses the *prod-flav* flavor. It assumes the use of tools like IDEs (e.g NetBeans or Android Studio) where the recommended system requirements states Intel Core i5 or equivalent, 4GB RAM, 1.5GB free space. This translated to 1 vCPU, 4GB vRAM, and 13GB storage as we consider the space requirements of the virtual machine while following the recommended RAM requirements. As for the CPU, since OpenStack defines it based on number of vCPUs, 1 was allocated as it is enough for development purposes.

Table 1: OpenStack flavors to support the customized architecture

Type of Class	Flavor Name	vCPU	vRam (MB)	Storage Size (GB)
Basic Programming Class	basic-prog-flav	1	1024	11
Database Admin and Prog Class	db-flav	1	2048	13
Advance Dev't Environment Class	prod-flav	1	4068	13
Android Image Kitkat 4.4	android-img-kitkat	1	1024	2

Running the instances under these flavors provides the minimal and most optimal resource allocation for each running instance.

In the context of designing a private cloud for an academic institution a number of issues were identified. Given that each course has a different set of tools that each individual use, the amount of resources utilized and the workload would vary to a degree and one of the issues present were to accurately identify how much computing resources can be allocated on a per individual basis without directly affecting the performance of the whole cloud. In addition to that the issue of scalability would also need to be addressed by looking at the usage pattern of the different courses that an IT curriculum contains and decide when to deploy more resources onto the private cloud. Usability is also a prominent concern since transitioning from standard desktops to a full cloud solution would require a paradigm shift on how individuals conduct their activities. In a class setting, it is important to take note of certain usage patterns like how individuals can download files to bring home. Should the creation of download facilities within the OpenStack network be provided or should students be able to create their own networks setup within OpenStack. These kinds of design decisions would help ease the adoption of the new technology during the transition period so that the new infrastructure would be fully utilized.

5 CONCLUSIONS AND FUTURE WORK

In summary, the research was able to show how a customized private cloud infrastructure can be implemented for an academic institution. However, performance of the current configuration can still be improved as scalability of the cloud would be a key concern. One possible way to improve performance may be thru the use of a different virtualization technology such as Docker. Docker is an open source container virtualization technology which enables the automation of application deployment inside software containers. A container is a virtualization method at the operating system level that allows running of multiple instances of an operating system on a single kernel. Using containers are much more efficient than using hypervisors because instead of virtualizing hardware, containers reside on a single kernel instance. Docker provides this ability where it packages and runs an application inside a container which is a loosely isolated environment. It also lets its users manage their infrastructure in the same manner as their applications are being handled.

The implementation of Docker within OpenStack's cloud infrastructure will enable infrastructure optimization as it will allow the organization to reduce the amount of storage that will be used and eliminate hypervisor licensing costs. Therefore, the organization can run and handle more workload given the same amount of resources.

Finally, in order to measure the feasibility and scalability of the cloud deployment, future work can include the definition of Service Level Agreements (SLAs) to formalize the expectations and requirements between the service provider and its customers or end-users. In defining and formalizing SLAs, the success of the OpenStack deployment can then be more quantitatively measured.

REFERENCES

[1] Rajleen Kaur et al, / A Review Paper on Evolution of Cloud Computing, its Approaches and Comparison with Grid Computing (IJCSIT) International Journal of Computer Science and Information Technologies, Vol. 5 (5), 2014, 6060-6063 ISSN: 0975-9646 https://pdfs.semanticscholar.org/7d94/150feaeeddd1cba7f1bd8ca92a50fc0efb3c.pdf

[2] C.Lakshmi Devasena. IMPACT STUDY OF CLOUD COMPUTING ON BUSINESS DEVELOPMENT. Operations Research and Applications: An International Journal (ORAJ), Vol. 1, No.1, August 2014

[3] S. Kumar, O. Murthy. Cloud Computing for Universities: A Prototype Suggestion and use of Cloud Computing in Academic Institutions. International Journal of Computer Applications (0975 – 8887) Volume 70– No.14, May 2013

[4] S. Bansal, S. Singh, A. Kumar. Use of Cloud Computing in Academic Institutions. IJCST Vol. 3, Iss ue 1, Jan. - March 2012. ISSN : 0976-8491 (Online) | ISSN : 2229-4333 (Print)

[5] J. Yunxia, Z. Bowen, W. Shuqi, S. Dongan. Research of Enterprise Private Cloud Computing Platform Based on OpenStack. International Journal of Grid Distribution Computing Vol.7, No.5 (2014), pp.171-180 http://dx.doi.org/10.14257/ijgdc.2014.7.5.16

[6] L.Chao. Application of Infrastructure as a Service in IT Education. American Journal of Information Systems, 2014, Vol. 2, No. 2, 42-48 Available online at http://pubs.sciepub.com/ajis/2/2/3

[7] Shamsul Anuar Mokhtar, Siti Haryani Shaikh Ali, Abdulkarem Al-Sharafi, and Abdulaziz Aborujilah. 2013. Cloud computing in academic institutions. In Proceedings of the 7th International Conference on Ubiquitous Information Management and Communication (ICUIMC '13). ACM, New York, NY, USA, , Article 2 , 7 pages. DOI=http://dx.doi.org/10.1145/2448556.2448558

[8] Mary-Jane Sule. 2011. A Conceptual Framework of Deploying Cloud IaaS in Higher Educational Institutions. In Proceedings of the 2011 IEEE Third International Conference on Cloud Computing Technology and Science (CLOUDCOM '11). IEEE Computer Society, Washington, DC, USA, 489-493. DOI: https://doi.org/10.1109/CloudCom.2011.72

[9] NetBeans System Requirements: https://netbeans.org/community/releases/81/relnotes.html#system_requirements

[10] OpenStack Platform Delivers for Private Cloud Users https://www.redhat.com/cms/managed-files/451-research-openstack-preferred-private-cloud-analyst-paper-f6228-201701-en.pdf

[11] Ibad Kureshi, Carl Pulley, John Brennan, Violeta Holmes, Stephen Bonner and Yvonne James, "Advancing Research Infrastructure Using OpenStack" International Journal of Advanced Computer Science and Applications(IJACSA), Special Issue on Extended Papers from Science and Information Conference 2013, 2013. http://dx.doi.org/10.14569/SpecialIssue.2013.030408 - See more at: http://thesai.org/Publications/ViewPaper?Volume=3&Issue=4&Code=SpecialIssue&SerialNo=8#sthash.bhnZLuLP.dpuf

[12] M. Dreyer1, J. Döbler, D. Rohde. June 2015. Building Service Platforms using OpenStack and CEPH: A University Cloud at Humboldt University. EUNIS 2015, At Dundee