Implementation of a Low Cost IaaS using Openstack

Tiago Rosado[1] and Jorge Bernardino[1,2]
[1]*ISEC – Superior Institute of Engineering of Coimbra, Polytechnic of Coimbra, 3030-190 Coimbra, Portugal*
[2]*CISUC – Centre of Informatics and Systems of the University of Coimbra,*
University of Coimbra, 3030-290 Coimbra, Portugal

Keywords: Openstack, IaaS, Cloud Computing, Open Source, Low Cost.

Abstract: Cloud computing has emerged as an important paradigm enabling software, infrastructure, and information to be used as services over the network in an on-demand manner. Cloud computing infrastructures can provide adaptive resource provisioning with very little initial investment while scaling to a large number of commodity computing nodes. In this paper, we present Openstack, a modular and highly adaptive open source architecture for both public and private cloud solutions. It is shown an experimental implementation of an IaaS, built on entry-level hardware, demonstrating the hardware, and most important, the basic software components needed to set the foundation for a solid entry-point to setup a low-cost Openstack cloud infrastructure.

1 INTRODUCTION

Cloud computing has emerged as a cost-effective and elastic computing paradigm for massively scalable, fault-tolerant, and adaptive computation. Cloud computing architectures scale to large numbers of commodity computers and adapt to changing hardware availability and requirements by dynamically allocating virtualized computing nodes. One of the main advantages of the cloud computing paradigm is that it simplifies the time-consuming processes of hardware provisioning, hardware purchasing and software deployment. For example, by decoupling the management of the infrastructure (cloud providers) from its use (cloud tenants), and by allowing the sharing of massive infrastructures, cloud computing delivers unprecedented economical and scalability benefits for existing applications and enables new scenarios. This comes at the cost of increased complexity in managing a highly multi-tenant infrastructure and posing new questions on attribution, pricing, isolation, scalability, fault-tolerance, load balancing, etc.

Cloud infrastructures can provide adaptive resource provisioning with very little initial investment while scaling to massive amounts of commodity computing nodes. Enterprise IT Organizations focused on the cutting edge of technology, developing new business solutions and platforms, have a constant need of resources for supporting their activity. Those resources could be storage, quality assurance servers or test beds and will imply different and specific configurations according to each project, development team or department. Rely on the IT department for delivery and the provisioning of such variety, changes in configurations for each server or infrastructure, consumes much time and human resources.

On the premise that improvement comes from inside, this work aims to deliver an open source private cloud computing implementation in order to respond IT Organizations demands regarding cost control issues, rapid and highly customizable infrastructure, maintaining high standards of quality, security and robustness. Implementing an open source private cloud computing solution avoids vendor lock-in and will provide resources on-demand to research and development departments well as independence to manage them without jeopardizes the enterprises critical IT infrastructure.

There are different offerings of cloud solutions like Infrastructure as a Service (IaaS), Platform as a Service (PaaS) and Software as a Service (SaaS). There are many studies about Infrastructure as a Service (IaaS) solutions focused on the study of middleware platforms and on comparative studies (Laszewski et al., 2012), (Bist et al., 2013) of the different open source solutions like Openstack;

Rosado, T. and Bernardino, J.
Implementation of a Low Cost IaaS using Openstack.
DOI: 10.5220/0006004402980303
In *Proceedings of the 11th International Joint Conference on Software Technologies (ICSOFT 2016) - Volume 1: ICSOFT-EA*, pages 298-303
ISBN: 978-989-758-194-6

(Sefraoui., 2012), Eucalyptus or OpenNebula, providing users a good basis in order to achieve freedom of choice. Studies on Openstack have shown major qualities focusing scalability and modularity (Castillo et al., 2013), security (Chen et al., 2010), (TaheriMonfared and Jaatun, 2011) and good performance in high demands (Steinmetz et al., 2012). However Openstack evolved since then having grown the partnership of big enterprises on the support and development, requiring an updated study. In this paper our focus is in the infrastructures, addressing a suitability of such implementation using the open source cloud provider solution from Openstack for maximizing existing in-house resources.

The remainder of this paper is organized as follows. Section 2 presents Openstack architecture and service followed by the object model. Section 3 presents a use case of how to design and implement an IaaS using Openstack. Finally, conclusions and future work are given in Section 4.

2 OPENSTACK ARCHITECTURE

Openstack is a fully open sourced cloud solution, released under the terms of the Apache license, for delivering and managing Cloud IaaS originally developed by NASA and Rackspace (Castillo et al., 2013). At this time is governed by The Openstack Foundation having as supporter's companies like Canonical, Suse, Red Hat, Nebula, HP, IBM, Rackspace, Intel, AMD, Cisco, among others. By being open source and free to download gives to small players the possibility to deploying small cloud infrastructures. Openstack is a combination of software projects considered one of the most complete solutions with great potential due to its architecture, community and partner's support having gained so far such reliability robustness and availability as an IaaS solution for both public and private cloud infrastructures (Castillo et al., 2013).

2.1 Conceptual Architecture

The modular and highly configurable architecture of Openstack enables this solution to be conceived and tailored to fit available hardware resources. It can be used by professional or entry level users, with small or large number of computer nodes. This allows enterprises to choose from a variety of complementary services in order to meet different needs, regarding computing, networking and storage (Pepple, 2013). Figure 1 represents the Openstack conceptual architecture with all native software components, developed by companies and individual supporters, depicting how they interact with each other.

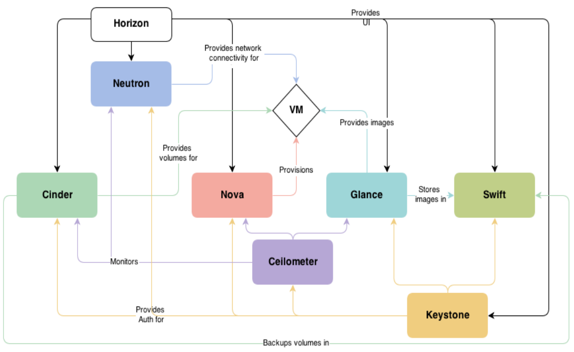

Figure 1: Openstack conceptual architecture.

Following, we describe those components dividing them into five groups due to their characteristics: Computing, Networking, Storing, Shared, and Supporting services. Although the supporting services are not native from Openstack's core, they are essential for its operation (Pepple, 2013).

2.1.1 Computing

The Nova Compute is the core software component of Openstack solution dictating services organization and interactions. Its own major components serve as basis for the cloud infrastructure, relying on Glance for Image services.

- *Nova*: Also known as Openstack Compute or Nova Compute, it provides virtual servers upon demand interacting with the hypervisors, supporting several kinds such as KVM, Xen, VMware or Hyper-V. It's an array of software that provides services for cloud resource management through its application programming interfaces (APIs), capable of orchestrating running instances, networks and access control. It's an essential service for a basic cloud architecture implementation.
- *Glance*: It's the Openstack Image service, a lookup and retrieval system for virtual machine (VM) images. It provides services for discovering, registering and retrieving virtual images through an API that allows querying of VM image metadata, catalog and manage large libraries of server images. It's an essential service for a basic cloud architecture implementation.

2.1.2 Networking

Openstack has recently gained a new component for networking control and management, the Neutron.

- *Neutron*: It is a project to provide network connectivity as a service between devices managed by other Openstack services (e.g. Nova). It provides capabilities for managing dynamic host configuration protocol (DHCP), static protocols (IP's), or virtual networks (VLAN's) along with other advanced policies and topologies. Due to its architecture, it allows users to take advantage of frameworks (e.g. intrusion detection systems, load balancing, virtual private networks) from supported vendors.

2.1.3 Storing

When moving into a larger infrastructure for a wider service it could be an option to have dedicated servers for storage management. In this case Openstack offers the Swift and Cinder.

- *Swift*: Also known as Openstack Object Storage, Swift is a highly available, distributed object/blob store. Allows users to store or retrieve files. It can be used by Cinder component to back up the VMs volumes.
- *Cinder*: It is known as Openstack Block Storage or Cinder Volumes and provides persistent block storage (or volumes) to guest virtual machines. By working with Swift, Cinder can use it to backup the VMs volumes. In earlier releases of Openstack this service was an integral part of Nova in the form of nova-volume. Cinder will mainly interact with Nova, providing volumes for its instances, allowing through its API the manipulation of volumes, volume types and volume snapshots.

2.1.4 Shared Services

These are services transversal to all infrastructures, needed in order to put the solution in functioning. The available services are: Keystone, Horizon, and Ceilometer.

- *Keystone*: The Openstack identity is a single point of integration for Openstack policy, catalog, token and authentication, applying them to users and services interactions. Without Keystone there is no compliance between services consequently the cloud solutions will not be available for end users. It is an essential service for a basic cloud architecture implementation.
- *Horizon*: The Openstack dashboard is a modular web application that provides a user interface for cloud infrastructure management by interacting with the public APIs off all other services.
- *Ceilometer*: A new component that provides a configurable collection of metering data in terms of CPU and network costs available from all other services in the platform, delivering a unique point of contact for billing systems.

2.1.5 Supporting Services

The supporting services are crucial for Openstack to run, but are not part of the core components or developed for the solution, instead they are software from external vendors to provide internal support. Then, we briefly describe the Database and Advanced Message Queue Protocol (AMQP).

- *Database*: By default Openstack uses MySQL as its RDBMS for core services to store configurations and management information. Typically MySQL database is installed on the main node of the infrastructure, the controller node. It's an essential service for a basic cloud architecture implementation.
- *Advanced Message Queue Protocol*: AMQP is the messaging technology used in Openstack for inter-process communication. It uses by default RabbitMQ and supports others such as Qpid or ZeroMQ. The broker gives a common platform for processes to send and receive messages between them making use of little or no knowledge of each other's component definitions. It's an essential service for a basic cloud architecture implementation.

2.2 Service Architecture

The Nova Compute is the core software component of Openstack solution dictating services organization and interactions, as shown in Figure 2.

This component has as major components the API Server, Message Queue, Compute Worker or Compute Node, Network Controller, Volume Controller, Scheduler and Image Store.
This API Server makes available to users the command and control of the hypervisor, storage and networking configurations through the API dashboard via basic http web services. A typical

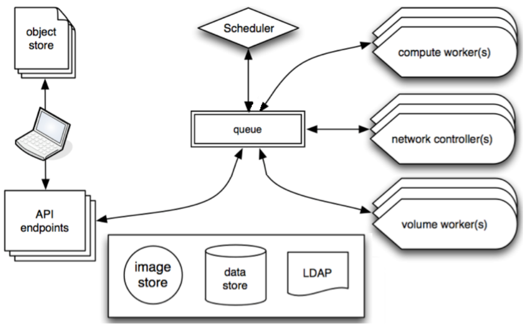

Figure 2: Openstack service architecture.

exchange of messages for an event request begins with the API server receiving user request, which authenticates and warrants that the user is permitted to do so. Is then evaluated the availability of resources requested by the user and, if available, the request is delivered to the queuing engine where services are listening to take action. When the work is done, a message is dispatched again into the queue for API server respond to the client.

2.3 Object Model

The object model represents the Openstack objects and how they stand before each other, showing how they relate, as depicted in Figure 3.

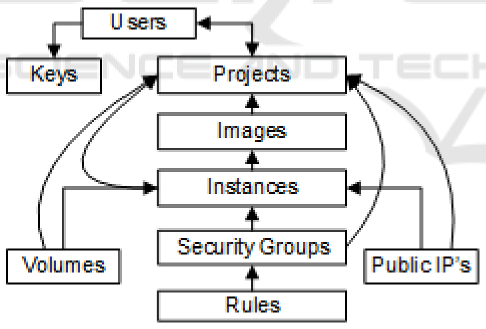

Figure 3: Openstack Object model.

We have Users to access the cloud environment, Keys for authentication and security purposes and then Projects. Projects are private areas created inside cloud computing infrastructure by the administrator. We can have one single project with the total amount of resources of the infrastructure or create more than one configured individually for different purposes with different rules, policies, and virtual machines. Projects can be seen as smaller private clouds within enterprise's private cloud, a unique space tailored on demand for different departments with different needs giving them privacy and freedom without jeopardize other's work.

3 IMPLEMENTATION OF AN IAAS IN OPENSTACK

In this use case implementation we used Ubuntu Server 12.04 Long Term Support (LTS) for its reliability regarding longer support and maintenance and Havana has the last stable release of Openstack. In order to provide a clear insight that is feasible implement a robust, modular, scalable and yet still performant low cost cloud solution we used commercial range hardware.

3.1 Hardware Configuration

For easier understanding, Openstack uses the term Cloud Controller to identify the main node in the infrastructure and Compute Nodes for all the others. The technical main characteristics are the following:

- *Cloud controller*: Intel® Core™ i3-3220 CPU @ 3.30GHz, 8 GB DDR3 / 1067 MHz, 250Gb Hard Disc Drive. Added one extra Gigabit Ethernet controller.
- *Compute Node*: Intel® Core™ i3-3220 CPU @ 3.30GHz, 16 GB DDR3 / 1067 MHz, 250Gb Hard Disc Drive. Added one extra Gigabit Ethernet controller
- *Networking*: Two eight port Gigabit Ethernet switches

The network layout and the physical connections for the cloud implementation are described in Figure 4.

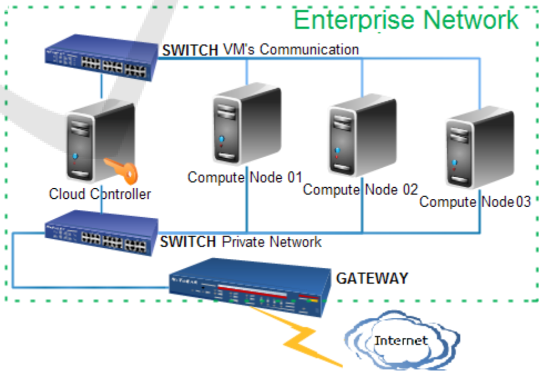

Figure 4: Cloud network topology.

All the nodes in the infrastructure communicate via two Gigabit private switches, one for VM's internal communications and another to ensure a connection to the enterprise internal network. The controller node is configured to do all the backend processes via Nova, image management via Glance, hosting the website via Horizon and dealing with user identification via Keystone. All the Compute

Nodes are configured with the Nova compute services and the KVM hypervisor.

Cloud architectures exploit virtualization techniques to provision multiple VM's on the same physical host so as to more efficiently use available resources.. With this basic configuration, built on entry level hardware, we aim to provide IT departments of companies' the guidelines for future implementations and exploitation of cloud computing technology based on Openstack modular architecture.

3.2 Software Configuration

Having chosen and implemented this basic architecture with legacy networking, in order to deliver an infrastructure as a service based on Openstack, there is no need to install all software components described for the architecture in order the cloud-computing environment to work.

Following we describe the software components to be installed on each machine, the main one who detains all the management activities for the environment, known as Cloud Controller, and remaining nodes or satellites named Compute Nodes, the machines that support all the available resources to be delivered.

3.2.1 Cloud Controller

The Cloud Controller is the core node in this environment. Following, we describe the services to be installed in order to provide such management capabilities that makes this the principal machine.

Nova is the core software component and is intended to be modular and easy to extend and adapt having the nova-api as a broker to its services in addition to native API Horizon. The nova-cert does all the certificate management for the platform for users and service's request flow. The nova-novncproxy is the software component that allows the access to instances through VNC clients bridging between public network and private network mediating token authentication and hypervisor-specific connections details. The connections are authorized by the validation of user's token provided by nova-consoleauth. The nova-scheduler manages users' requests for resources selecting the most suitable compute node for instances to run. The host selection criteria for dispatching requests considers factors like available physical memory, storage, processor or network load in a two-step process, which can be managed by the system administrator. The nodes in the system are filtered then weighted and sorted according to match those configured criteria. Glance is the image service for Openstack; it provides discovery, registration and delivery services for disk and server images, capable to copy or snapshot a server image to store it promptly or use them later as a template for new servers' creation. These capabilities are provided by glance-api, the focal point for image requests and glance-registry for processing images metadata like size, or type.

In this configuration, as in all others, shared services provided by horizon and keystone are needed in order to ease users' requests by being a single point of interaction between those requests and all the services responsible for delivering them and ensure identity among services, authentication and system policy, respectively.

Similar to Horizon and Keystone every Openstack implementation needs a database and a message queue. By default and here, we use MySQL for database management and RabbitMQ as the message broker for services. Figure 5 below, illustrates those services.

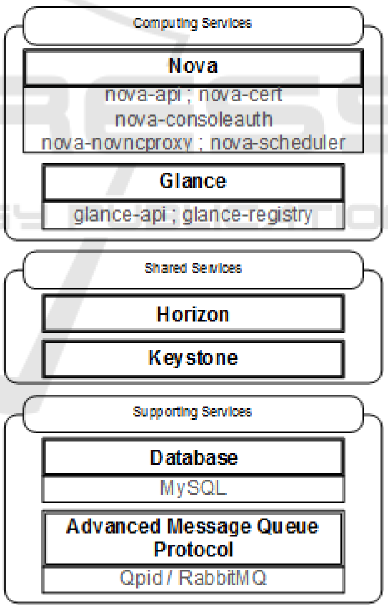

Figure 5: Cloud Controller software modules.

3.2.2 Compute Nodes

The compute nodes are the machines responsible for delivering to end users the resources available in the cloud platform under the form of virtual machines or storage. It runs the hypervisor portion of Compute, using KVM by default as a virtualization infrastructure, which operates tenant virtual machines, as well as provisioning network services and implementing security groups using nova-compute and nova-network for that.

Nova has no specific hypervisor implemented. Instead, it has an abstraction layer for compute drivers that allow users to choose one from some other known vendors. Figure 6 shows the Compute Nodes software modules.

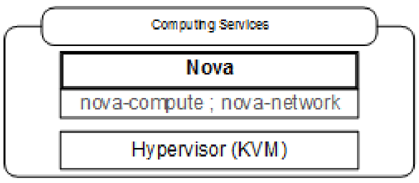

Figure 6: Compute Nodes software modules.

4 CONCLUSIONS

Cloud architectures exploit virtualization techniques to provision multiple Virtual Machines (VMs) on the same physical host, so as to efficiently use available resources. The cloud computing paradigm and available solutions in the market, regarding to Openstack, have been implemented and explored in various case scenarios, with major tests in academy and industry, presenting good results for performance, reliability and final user acceptance. Openstack leverages due to its modular architecture and simplicity to implement. It can be adopted into major datacentres but also to entry-level hardware providing enterprises a massively scalable cloud infrastructure. Allied to the fact of being open source, it helps on cost control, enables to deliver rapidly on-demand resources to final users.

This work has presented an overview of Openstack architecture, services available and how they work and adapt into any hardware infrastructure providing a sustainable and robust private cloud solution.

Based on this study, future work will involve an investigation on how to improve final user experience with the platform and administration activities regarding to measures and monitoring the cloud environment.

REFERENCES

Bist, M. Wariya, M. Agarwal, A. 2013. Comparing Delta, Open Stack and Xen Cloud Platforms: A Survey on Open Source IaaS. In *2013 3rd IEEE International Advance Computing Conference*, 2013.

Castillo, J. A. L. Mallichan, K. Al-Hazmi, Y. 2013. Openstack federation in experimentation multi-cloud testbeds. In 2013 IEEE International Conference on Cloud Computing Technology and Science. 2013.

Chen, Y. Paxson, V. Katz, R. H. 2010. What's New About Cloud Computing Security? Electrical Engineering Computer Sciences, University of California at Berkeley, Technical Report No. UCB/EECS-2010-5. Jan. 2010.

Dargha, R. 2012. Cloud computing: from hype to reality. Fast tracking cloud adoption in International Conference on Advances. In *computing, Communications and Informatics* 2012, Aug 2012.

Eucalyptus – [online] https://www.eucalyptus.com (Accessed March 2015).

Infrastructure as a Service – [online] http://www.gartner.com/it-glossary/infrastructure-as-a-service-iaas (Accessed February 2015).

Kostanos, K. Kapsalis, A. Kyriazis, D. Themistocleous, M. Cunha, P. 2013. Open-source IaaS Fit for Purpose: A Comparison between Opennebula and Openstack. *Int. Journ.. of Electronic Business Manag.,* Vol.11, Nº3, pp. 191-201, 2013.

Laszewski, G. Diaz, J. Wang, F. Fox, G. C. 2012. Comparison of multiple cloud frameworks. Pervasive Technology Institute, Indiana University, 2012.

OpenNebula – [online] http://opennebula.org/ (Accessed March 2015).

Openstack – [online] http://www.openstack.org/ (Accessed March 2015).

Pepple, K. 2013. Deploying Openstack, Creating opensource clouds, 2nd Edition, O'Reilly Media.

Platform as a Service – [online] http://www.gartner.com/it-glossary/platform-as-a-service-paas/ (Accessed February 2015).

Sefraoui, O. Aissaoui, M. Eleuldj, M. 2012. OpenStack: Toward an Open-Source Solution for Cloud Computing. In *International Journal of Computer Applications*, Vol. 55 Nº3, October 2012.

Software as a Service – [online] http://www.techterms.com/definition/saas (Accessed February 2015).

Steinmetz, D. Perrault, B. Nordeen, R. Wilson, J. Wang, X. 2012. Cloud Computing Performance Benchmarking and Virtual Machine Launch Time. In *SIGITE'12* Calgary, Alberta, Canada. 2012.

TaheriMonfared, A. Jaatun, M. G. 2011. As strong as the weakest link: Handling compromised components in OpenStack. In *3rd IEEE International Conference on Cloud Computing Technology and Science.* 2011.

Openstack object model – [online] http://docs.openstack.org/developer/nova/object.model.html

Openstack conceptual architecture – [online] http://docs.openstack.org/juno/install-guide/install/apt/content/ch_overview.html

Openstack service architecture – [online] http://docs.openstack.org/developer/nova/service.architecture.html