Rafael Turnes Silveira

OpenStack com Open vSwitch

São José – SC

Fevereiro / 2014

Rafael Turnes Silveira

OpenStack com Open vSwitch

Monografia apresentada a Coordenação do Curso Superior de Tecnologia em Sistemas de Telecomunicações do Instituto Federal de Santa Catarina para a obtenção do diploma de Tecnólogo em Sistemas de Telecomunicações.

Orientador:
Prof. Ederson Torresini, Msc.

CURSO SUPERIOR DE TECNOLOGIA EM SISTEMAS DE TELECOMUNICAÇÕES
INSTITUTO FEDERAL DE SANTA CATARINA

São José – SC

Fevereiro / 2014

Monografia sob o título “*OpenStack com Open vSwitch*”, defendida por Rafael Turnes Silveira e aprovada em 27 de Fevereiro de 2014, em São José, Santa Catarina, pela banca examinadora assim constituída:

_____

Prof. Ederson Torresini, Msc.

Orientador

_____

Fernando Schubert

Beck et al. Serviços de Informática Ltda.

_____

Prof. Marcelo Maia Sobral, Dr. Eng.

IFSC

“Liberdade não é poder escolher entre o preto e branco, mas sim abominar este tipo de propostas de escolha.”

Theodore Wiesengrund Adorno

Agradecimentos

Agradeço esta conquista à todos que sempre acreditaram no meu potencial e principalmente aos que nunca desistiram de me apoiar e sempre estiveram ao meu lado nos momentos mais difíceis.

Resumo

O ambiente em que encontram-se as redes de computadores está em frequente transformação, evolução e consequentemente a massificação dos seus recursos. Estes sistemas e recursos precisam ser dinâmicos, gerenciáveis e elásticos para poderem adequar-se às necessidades dos usuários. Diante deste cenário desenvolve-se um ambiente de computação em nuvem no IFSC, aplicando o modelo infraestrutura como serviço. Que possibilita o provisionamento dos recursos computacionais (processamento, armazenamento e rede de computadores) aos usuários de forma simples, automatizada e remotamente controlada. As pesquisas e atividades foram em cima da plataforma aberta OpenStack, demonstrando-se então uma ferramenta completa e simples do ponto de vista do usuário.

Palavras chave: OpenStack, Computação em Nuvem, IaaS.

Abstract

The current environment from the computer networks faces frequent changes, evolution and consequently the massification of its resources. These systems and resources need to be dynamic, manageable and elastic in order to adapt to the needs users. In this scenario has been developed an environment of cloud computing at IFSC, applying the model infrastructure as a service. It lets the provisioning of computers resources (compute, storage, networking) in a simple, automated, remotely controlled way to the users. The research and activities were on the open platform OpenStack, demonstrating a complete and simple tool from the point of view of the user.

Keywords: OpenStack, Cloud Computing, IaaS.

Sumário

1 Introdução 10

1.1 Motivação 11

1.2 Justificativa 11

1.3 Objetivos 12

1.3.1 Objetivo Principal 12

1.3.2 Objetivos Específicos 12

2 Fundamentação Teórica 10

2.1 SDN 10

2.2 OpenFlow 13

2.3 Virtualização 14

2.4 Open vSwitch 17

2.5 Computação em Nuvem 21

2.6 OpenStack 23

2.7.1 Interface Web de Administração (Dashboard) 25

2.7.2 Serviço de Identidade 26

2.7.3 Serviço de Imagem 26

2.7.4 Serviço de Computação 26

2.7.5 Serviço de Armazenamento 27

2.7.5.1 Armazenamento em Objeto 27

2.7.5.2 Armazenamento em bloco 28

3 Desenvolvimento 29

3.1 Arquitetura 30

3.2 Implementação 32

3.2.1 Recurso disponível 32

3.2.2 Infraestrutura física....32
3.2.3 Infraestrutura hospedeira....33
3.2.4 Instalação do OpenStack....34
3.2.5 Banco de dados e servidor de mensagens....35
3.2.6 Serviço de identidade e imagem....36
3.2.7 Serviço de Rede do OpenStack....39
3.2.8 Serviço de Computação do OpenStack....40
3.2.9 Serviço de armazenamento em bloco do OpenStack....42
3.2.10 Interface Web de Administração (Dashboard)....44
4 Testes e Resultados....46
4.1 Multi-inquilinos....46
4.1.1 Criar contas e usuários....47
4.1.2 Alocar recursos nas duas contas....48
4.2 Registro de imagens....49
4.3 Serviços de rede....50
4.3.1 Rede externa....51
4.3.2 Rede local....52
4.3.3 Interligar as redes no roteador.....55
4.4 Instanciando uma VM....56
4.5 Migração de máquinas virtuais entre VMMs....61
5 Conclusões....67
5.1 Trabalhos futuros....67
Lista de Abreviaturas....69
Referências Bibliográficas....71

Lista de Figuras

Figura 2.1 - Arquitetura das SDN – Portal Network Computing.....11

Figura 2.2 – Virtualização – Portal Oracle.....14

Figura 2.3 - Arquitetura Tradicional x Virtualização – Portal Network Computing 15

Figura 2.4 - Virtualização total – Oracle 16

Figura 2.5 – Virtualização auxiliada pelo hardware 16

Figura 2.6 – Paravirtualização.....17

Figura 2.7 - Infraestrutura de rede virtualizada - Portal IBM DeveloperWorks 18

Figura 2.8 - Switch virtual distribuído - Portal IBM DeveloperWorks 18

Figura 2.9 - Funcionalidades do Open vSwitch - Portal OpenvSwitch.org 19

Figura 2.10 – Arquitetura do Open vSwitch – Portal OpenStack.....20

Figura 2.11 - Visão Geral do OpenStack – Portal OpenStack.....24

Figura 3.1 - Visão geral do IaaS – Autor 29

Figura 3.2 - Responsabilidades no modelo IaaS – Google, Imagens.....30

Figura 3.3 - Arquitetura básica – Autor.....30

Figura 3.4 - Visão frontal e lateral do rack - Autor 33

Figura 3.5 – Infraestrutura hospedeira 34

Figura 3.6 - Instalação genérica.....35

Figura 3.7 - Visão geral do banco de dados e o servidor de mensagem 36

Figura 3.8 - Sequência de instalação do Keystone 37

Figura 3.9 - Sequência na criação dos elementos do Keystone.....38

Figura 3.10 - Componentes criados e suas relações. 38

Figura 3.11 - Sequência de instalação do Glance. 38
Figura 3.12 – Arquitetura Neutron – Portal OpenStack 40
Figura 3.13 - Arquitetura lógica do Nova – Portal OpenStack 42
Figura 3.14 – Gerenciamento das VMM pelo componente NOVA 42
Figura 3.15 - Arquitetura do Cinder 43
Figura 3.16 - Cinder e LVM 44
Figura 3.17 - Visão geral Dashboard – Google Imagens 45
Figura 4.1 – Sequência lógica Keystone 46
Figura 4.2 - Portal de autenticação – OpenStack 47
Figura 4.3 - Criar Projeto – OpenStack 47
Figura 4.4 - Criar usuário – OpenStack 48
Figura 4.5 - Lista de volumes – OpenStack 49
Figura 4.6 - Registro de imagens no Glance – OpenStack 50
Figura 4.7 - Lista de Imagens - OpenStack 50
Figura 4.8 - Criando rede externa, aba 1 - OpenStack 51
Figura 4.9 - Criando rede externa, aba 2 - OpenStack 51
Figura 4.10 - Criar rede externa, aba 3 - OpenStack 52
Figura 4.11 - Visão geral da rede – OpenStack 52
Figura 4.12 - Criando roteador - OpenStack 53
Figura 4.13 - Criando rede local, aba 1 - OpenStack 53
Figura 4.14 - Criando rede local, aba 2 - OpenStack 54
Figura 4.15 - Criando rede local, aba 3 - OpenStack 54
Figura 4.16 - Visão geral da rede – OpenStack 55
Figura 4.17 - Interligando o roteado à rede externa – OpenStack 55
Figura 4.18 - Interligando o roteador à rede local – OpenStack 56
Figura 4.19 - Visão Geral da rede – OpenStack 56

Figura 4.20 - Liberando o protocolo ICMP – OpenStack.....57
Figura 4.21 - Liberando o protocolo SSH - OpenStack.....58
Figura 4.22 - Requisitando uma VM, aba 1 - OpenStack.....58
Figura 4.23 - Requisitando uma VM, aba 2 - OpenStack.....59
Figura 4.24 - Requisitando uma VM, aba 3 - OpenStack.....59
Figura 4.25 VM integrada a rede.....60
Figura 4.26 - Associando um IP externo na VM - OpenStack.....60
Figura 4.27 - Testando conectividade de rede via ICMP – OpenStack.....61
Figura 4.28 - Lista de VM em execução e seus respectivos VMM. - OpenStack.....62
Figura 4.29 - Usuário requisitando a mudança do tamanho da instância - OpenStack 63
Figura 4.30 - Administrador requisitando a migração da VM – OpenStack.....63
Figura 4.31 - Execução da migração – OpenStack.....63
Figura 4.32 - Confirmação ou reversão da migração – OpenStack.....64
Figura 4.33 - Comparação da alocação do VMMs antes e epois da migração. – OpenStack.....64
Figura 4.44 - Testando a conectividade de uma VM durante a migração 65

1 Introdução

A notória evolução das redes de computadores e sistemas computacionais nas últimas décadas tem proporcionado um rápido avanço na massificação do uso de redes, especialmente a Internet e o desenvolvimento de novas técnicas e paradigmas de transmissão de dados.

Esta evolução é abordada em (SCHUBERT, 2011 apud GRIMES, JAEGER e LIN 2008), onde através da história da computação houveram várias mudanças de paradigma. Dos main-frames para mini-computadores, dos mini-computadores ao micro-processamento e do micro-processamento aos computadores em rede. Esta evoluão e mudanças proporcionaram diversos avanços tecnológicos como virtualização de sistemas, redes e protocolos definidos para comunicação e o desenvolvimento da Internet em nível de rede e aplicativo.

A massificação dos recursos computacionais é devido, em grande parte, ao crescimento e a popularização da Internet. Toda essa facilidade e disponibilidade demanda uma complexa infraestrutura que é oculta aos seus usuários. Equipamentos de redes, servidores, sistemas distribuídos, armazenamento de dados estão espalhados pelo mundo.

Este crescimento no número de equipamentos conectados e de infraestrutura começou a ocasionar problemas de gerenciamento e escalabilidade, além de gerar custos para as grandes empresas. Por exemplo, o acesso à websites durante o período de Natal é superior em 2 à 3 vezes o acesso durante o ano (GREENBERG et al. 2008), o que obrigava a grandes redes de varejo a manter infraestruturas gigantescas para atender apenas à estes feriados, ficando ociosos durante o restante do ano. Outro problema está relacionado com as redes tradicionais, o nível de gerenciamento manual e crescimento das redes torna a administração tradicional onerosa e complexa.

Visualizando estes dois cenários, da falta de escalabilidade das infraestruturas, ociosidade e limitações nas redes surgiu o paradigma da computação em nuvem, que, centraliza o gerenciamento e aumenta dramaticamente a utilização destas infraestruturas, reduzindo os custos de operação e a ociosidade. Mais recentemente

as redes de computadores estão passando por esta revolução com o surgimento das redes definidas por softwares e padrões, ainda estão em fase de desenvolvimento, que possibilitam o gerenciamento centralizado das mesmas.

É nesse contexto que plataforma de computação em nuvem OpenStack, do tipo infraestrutura como serviço e de código aberto, com auxílio do Open vSwitch proporciona uma alternativa para os problemas enfrentados hoje no que diz respeito a infraestrutura.

1.1 Motivação

Em um ambiente tradicional os recursos de infraestrutura normalmente são mensurados para suportar picos de demanda. Nessa situação os recursos podem ficar subutilizados na maior parte do tempo. Em casos em que infraestrutura é dimensionada para suportar uma demanda média, pode ocorrer instabilidades ou até mesmo a falta do serviço prestado em momentos de pico, ficando evidente manter a eficiência na relação recurso disponível X demanda. Atualmente a virtualização dos recursos de infraestrutura é praticada para otimizar o aproveitamento dos recursos de infraestrutura, assim diminuir a ociosidade.

Atualmente a virtualização está bastante difundida e amadurecida para processamento e armazenamento. O que está ocorrendo, agora, é o aprofundamento do estudo das redes virtuais: as unidades de processamento e armazenamento estão mais flexíveis e móveis, adaptáveis e em rede. Uma das formas em estudo para essa horizontalização das redes é o *Software-defined Networking* (SDN), assunto que será analisado adiante.

1.2 Justificativa

A possibilidade de construir um ambiente escalável, elástico, gerenciável e dinâmico a fim de otimizar o gerenciamento de recursos de infraestrutura. Proporcionando uma rápida readequação das necessidades do ambiente.

1.3 Objetivos

1.3.1 Objetivo Principal

Este trabalho tem como objetivo criar um ambiente de computação em nuvem com intuito de validar a proposta da mesma. Será usado, para tal, um software já concebido que suporte tais tecnologias.

1.3.2 Objetivos Específicos

- Disponibilizar um ambiente de computação em nuvem, do modelo infraestrutura como serviço, para o IFSC utilizando ferramentas desenvolvidas utilizando *software livre*;
- Documentar a instalação do ambiente, abordando as camadas de virtualização, *framework* de nuvem e redes definidas por *software*.

2 Fundamentação Teórica

Neste capítulo serão abordados os conceitos básicos de redes de computadores com ênfase às redes de computadores definidas por software (SDN – *Software Defined Networks*), visão geral da virtualização de hardware e computação em nuvem. As tecnologias utilizadas para o desenvolvimento desse trabalho também serão abordadas nesse capítulo, sendo elas os protocolos *OpenFlow* e OVSDB (*Open vSwitch Database Management Protocol*), o software *Open vSwitch* e a plataforma de computação em nuvem *OpenStack*.

2.1 SDN

As redes definidas por *software* (SDN) são um novo termo para o paradigma de programação de redes. SDN refere-se à habilidade de utilizar software para programar os equipamentos de rede de forma dinâmica e controlar o comportamento da rede como um todo (KIM e FEAMSTER, 2013). Um dos elementos chave das SDN é a introdução da abstração entre o encaminhamento tradicional e os planos de controle (*control planes*) de forma a separá-los e prover aplicações com suas respectivas APIs (*Application Programming Interface*) para controlar a rede de forma programável.

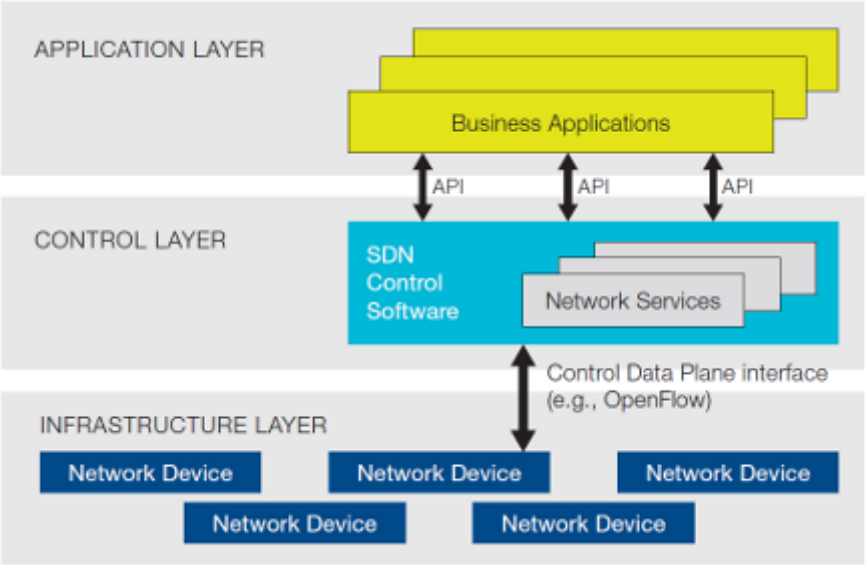

Figura 2.1 - Arquitetura das SDN – Portal Network Computing

O SDN surge para preencher lacunas e resolver as dificuldades já conhecidas de uma rede de computadores tradicional, como limitação de 4000+ VLANs. Os ambientes de rede cada vez mais requerem flexibilidade, dinamismo, fácil gerenciamento, baixo custo e adaptabilidade.

SDN é uma arquitetura emergente que desacopla as funções de controle de rede e a função de encaminhamento de pacotes. O que permite o controle da rede de forma altamente programável e abstraindo a infraestrutura para aplicações e serviços da mesma (ONF, 2013).

A figura 2.1 mostra que arquitetura SDN possui três camadas segundo Haleplidis (2013):

- **Camada de infraestrutura** - Composta por equipamentos de redes, que podem ser implementados em hardware ou em software, físicos ou virtualizados. É nessa camada que ocorre o tratamento e encaminhamento de pacotes. Também é responsável por representar o estado operacional do equipamento e suas respectivas portas e interfaces de rede.

- **Camada de controle** - essa camada é dividida em dois planos:

1) Plano de controle - é responsável por dar instruções à camada de infraestrutura de como pacotes devem ser tratados ou encaminhados.
2) Plano de gerenciamento - é responsável em gerenciar e monitorar os dispositivos de rede (camada infraestrutura), por exemplo o estado operacional e suas respectivas portas e interfaces de rede.
3) A comunicação e as instruções desses planos podem ser feitas através de protocolos, interface de programação de aplicativos (API) ou por comunicação entre processos. Vale a pena ressaltar o protocolo *OpenFlow* (plano de controle) e OVSDB (plano de gerenciamento), que serão vistos posteriormente.

- **Camada de aplicação** - são aplicativos ou serviços que utilizam a camada de controle. Por exemplo, aplicativos SDN, orquestradores de cloud (ex OpenStack, CloudStack[1]) e business application.

De acordo com Haleplidis (2013), as características de uma rede SDN são:

a) Ser altamente programável: o controle de tráfego da rede é programável.

b) De rápida adaptação: o administrador de rede consegue dinamicamente ajustar o tráfego de rede de acordo com a necessidade devido a separação da camada encaminhamento de pacote e a camada de controle.

c) Gerenciamento centralizado: permite uma visão global e centralizada da rede.

d) Configuração por programação: administradores de rede conseguem configurar, gerenciar, proteger e otimizar os recursos de rede de forma dinâmica e automatizada através de programas.

e) Baseado em padrão aberto: No atual cenário cada fabricante possui protocolos e firmwares proprietários, que impossibilitam a compatibilidade entre diferentes fabricantes. Uma rede SDN o controle do tráfego é feito por controladores

[1] http://cloudstack.apache.org/

SDN, que quando implementados por meio de padrões abertos permitem a compatibilidade entre fabricantes.

Segundo Helaplidis (2013), em uma rede convencional, equipamentos de rede possuem softwares proprietários, cuja a distribuição geralmente não é livre, chamados firmware. Regras de encaminhamento de pacote são criados por firmware que definem para onde o pacote deve ser encaminhado. Equipamentos da camada 2, como switch por exemplo, tratam pacotes do mesmo jeito, ou seja, pacotes para o mesmo destino são encaminhados pelo mesmo caminho. Equipamentos de redes que trabalharam na camada 3 ou em camada superiores, por exemplo os switches L3, que conseguem distinguir e tratar os diferentes tipos de pacotes. Porém esse tipo de equipamentos são caros, inviabilizando implementações em escala.

Uma rede definida por software permite que um administrador consiga programar e moldar o tráfego de uma rede através de uma console de controle centralizado, sem a necessidade de acessar diretamente equipamentos da rede. Através dessa console o administrador consegue monitorar, gerenciar, proteger, otimizar e alterar regras de tráfego da rede quando for necessário. Isso permite que um administrador de rede tenha um controle maior sobre o fluxo de tráfego da rede e adaptar rapidamente as mudanças (GIBILISCO, 2012).

2.2 OpenFlow

OpenFlow é um protocolo de comunicação aberto entre a camada de controle e a camada infraestrutura de uma arquitetura SDN. É através do OpenFlow que regras de encaminhamento de pacotes criadas na camada de controle são adicionadas, removidas ou modificadas na tabela de roteamento interna de um equipamento de rede (ONF, 2013).

A primeira versão do OpenFlow foi originalmente desenvolvido na universidade de Stanford. Atualmente OpenFlow está na versão 1.4 e está sendo desenvolvida e gerenciada pela Open Networking Foundantion (ONF). A ONF é uma organização sem fins lucrativos focada em promover e auxiliar a adoção de redes SDN.

2.3 Virtualização

Virtualização é a emulação ou a versão virtual de hardware ou de software, sistema operacional, dispositivo de armazenamento ou recursos de rede. No escopo do trabalho, a virtualização refere-se especificamente à virtualização de hardware e software em máquinas virtuais. A figura 2.2 apresenta no item (a) a virtualização de um software e no item (b) a virtualização de hardware.

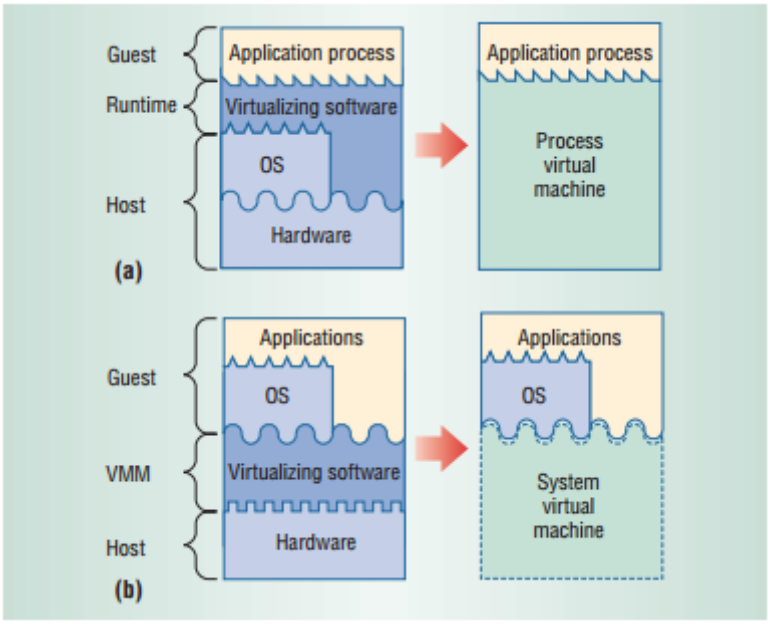

Figura 2.2 – Virtualização – Portal Oracle

A virtualização de sistema operacional, ou máquinas virtuais, se tornou uma ferramenta importante no desenvolvimento de sistemas, testes, simulação de ambientes e na consolidação de infraestruturas, reduzindo ociosidade de servidores físicos permitindo o compartilhamento de recursos entre diversas aplicações e serviços.

A virtualização de servidores apresenta-se como um paradigma de consolidação e otimização de recursos. A Figura 2.3 apresenta um paralelo entre estes dois modelos de implantação.

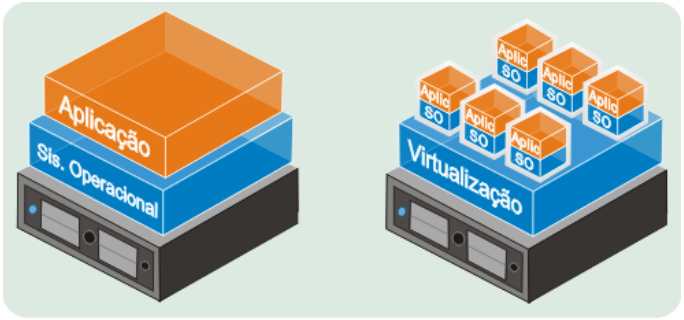

Figura 2.3 - Arquitetura Tradicional x Virtualização – Portal Network Computing

A máquina hospedeira, também chamada de monitor de máquinas virtuais ou *hypervisor*, é responsável pela camada de abstração entre o hardware e os sistemas operacionais convidados (máquina virtual). O trabalho clássico de (POPEK e GOLDBERG, 1974) estabelece três características para que um sistema seja considerado um monitor de máquinas virtuais (VMM - *virtual machine monitor*):

1. Fidelidade: o software no VMM executa identicamente à execução em hardware;
2. Performance: a maioria das instruções do hospedeiro são executadas pelo hardware sem intervenção do VMM;
3. Segurança: o VMM gerencia todos os recursos de hardware.

De acordo com os tipos de virtualização temos os seguintes modelos:

- Virtualização total: é o modelo onde a máquina virtual simula todo o hardware para permitir que um sistema operacional convidado seja executado de maneira isolada. É o caso conhecido também por emulação (ORACLE, 2009). A figura 2.4 apresenta um modelo da virtualização total.

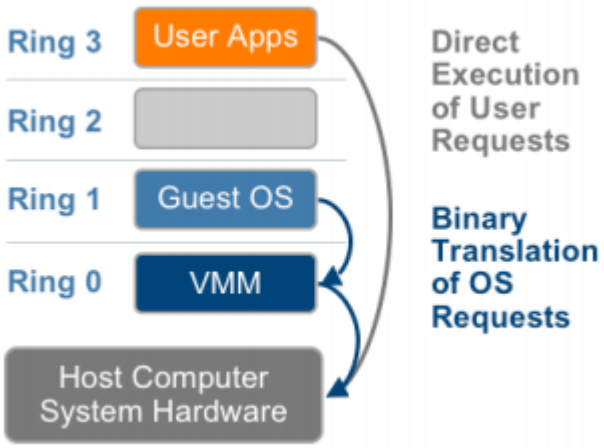

Figura 2.4 - Virtualização total – Oracle

- Virtualização auxiliada pelo hardware: quando o próprio hardware possui, em sua arquitetura, suporte a execução de máquinas virtuais de modo isolado. As arquiteturas atuais possuem suporte, como o Intel VT-x[1] e o AMD-V[2]. A Figura 2.5 apresenta um exemplo visual desta arquitetura.

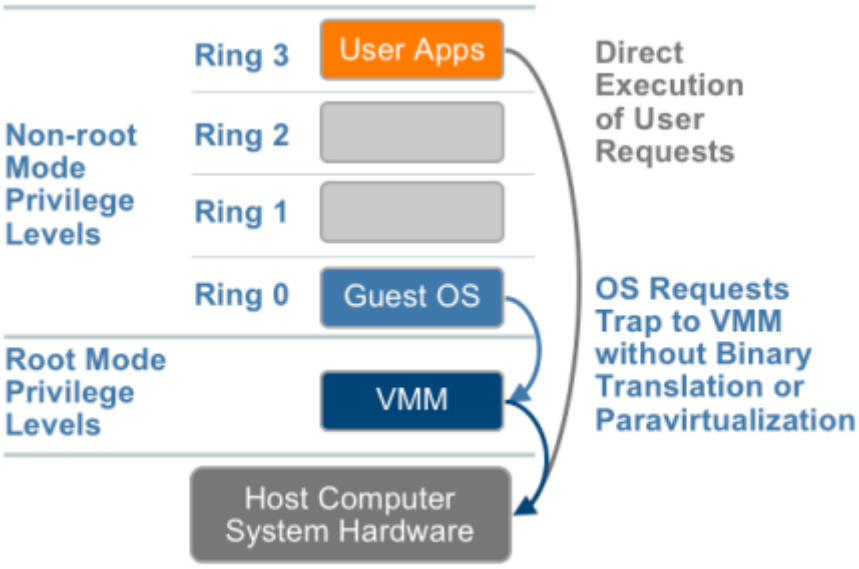

Figura 2.5 – Virtualização auxiliada pelo hardware

[1] http://www.intel.com/content/www/us/en/virtualization/virtualization-technology/hardware-assist-virtualization-technology.html
[2] http://www.amd.com/us/solutions/servers/virtualization/Pages/virtualization.aspx

- Paravirtualização: em linhas gerais, o sistema operacional da máquina virtual percebe que está rodando num ambiente virtualizado. Há um elemento, chamado Hypervisor, que constitui-se de uma API ou camada para que o sistema operacional convidado tenha acesso ao hardware. A figura 2.6 apresenta a representação deste modelo de virtualização.

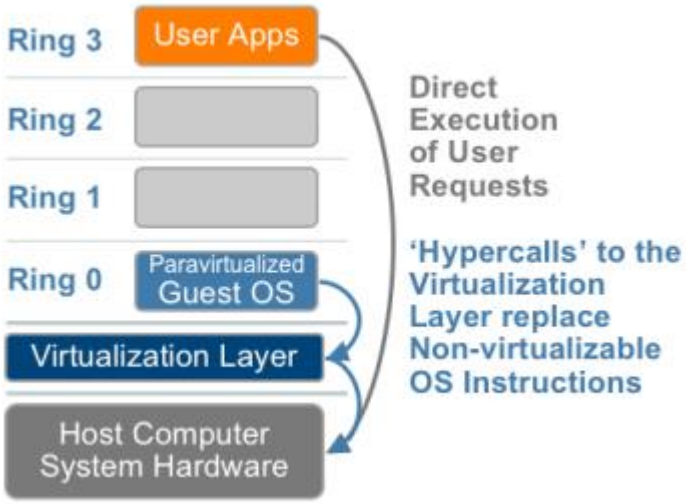

Figura 2.6 – Paravirtualização

2.4 Open vSwitch

Open Virtual Switch (Open vSwitch ou OVS) é um software multiplataforma de código aberto que emula switches virtuais permitindo que máquinas virtuais se comuniquem entre si ou encaminhar o tráfego para uma rede física

Switches virtuais permitem interligar interfaces de rede virtual (vNIC – *virtual network interface card*) e interfaces físicas ou interligar vNICs com vNICs que estão no mesmo mesmo servidor. A figura 2.7 apresenta uma infraestrutura de rede virtualizada.

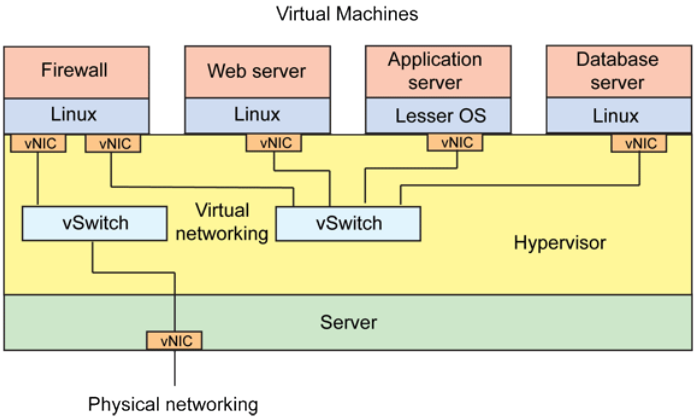

Figura 2.7 - Infraestrutura de rede virtualizada - Portal IBM DeveloperWorks

Por ser descentralizado, Open vSwitch possibilita que um switch virtual que está em um servidor possa facilmente e transparentemente ser juntar a um switch virtual que está em um outro servidor. O que facilita a migração de máquinas virtuais e suas respectivas vNICS entre servidores (ver figura 2.8). Esse é um cenário que será abordado no capítulo 4.

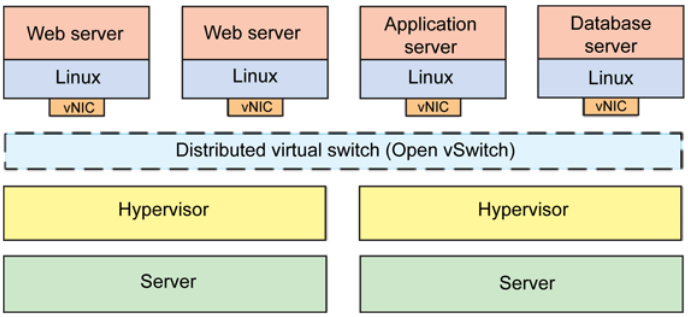

Figura 2.8 - Switch virtual distribuído - Portal IBM DeveloperWorks

Ser programável é uma característica fundamental para redes SDN. Os protocolos OpenFlow e OVSDB foram implementados no Open vSwitch para permitir a automatização de rede através de programas. Além do mais Open vSwitch traz funcionalidades comuns (LACP[1], VLAN[2], STP[3], QoS[4]) que são existentes em equipamentos de rede para dentro de ambientes virtualizados. Ver figura 2.9

[1] http://standards.ieee.org/getieee802/download/802.1AX-2008.pdf
[2] http://standards.ieee.org/getieee802/download/802.1Q-2011.pdf
[3] http://standards.ieee.org/getieee802/download/802.1D-2004.pdf
[4] http://tools.ietf.org/rfc/rfc2212.txt

OVS pode ser controlado e programável através do protocolo de gerenciamento OVSDB e da interface padrão de comunicação OpenFlow respectivamente. OVS foi desenvolvido para ser compatível com diversas plataformas e hardware, porém o principal desenvolvimento é feito em Linux. OVS provê suporte a interfaces padrão de gerenciamento, como sFlow[1], NetFlow[2] entre outros (Open vSwich, 2014a).

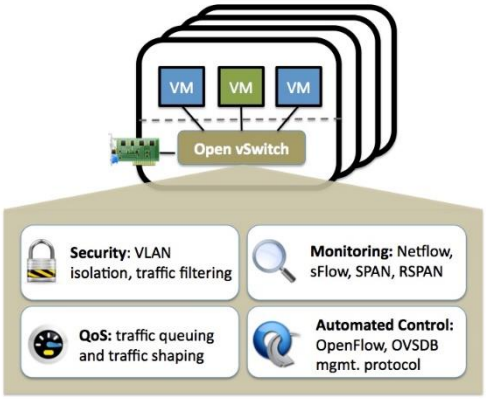

Figura 2.9 - Funcionalidades do Open vSwitch - Portal OpenvSwitch.org

Segundo Open vSwitch (2014b), os principais componentes da arquitetura do OVS são:

ovs-vswitchd – a execução de um programa em *background* (*daemon*) que implementa o switch, e em conjunto com o módulo do kernel Linux faz a comutação baseada em fluxo.

ovsdb-server - um servidor de banco de dados para armazenar informações de configuração do ovs-vswitchd.

ovs-dpctl – uma ferramenta para configurar o módulo de comutação do kernel.

ovs-vsctl – uma ferramenta para requisitar e atualizar a configuração do ovs-vswitchd.

ovs-appctl – uma aplicativo que envia comandos para *daemons* do Open vSwitch.

[1] http://www.sflow.org/
[2] http://www.ietf.org/rfc/rfc3954.txt

ovs-ofctl – uma ferramenta que consultar e controlar switches e controladores OpenFlow.

Na figura 2.10, mostra a arquitetura do OVS, o fluxo de gerenciamento e o processamento de pacotes.

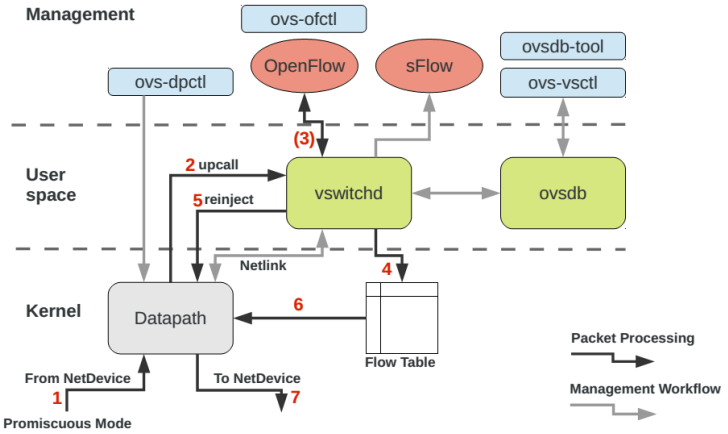

Figura 2.10 – Arquitetura do Open vSwitch – Portal OpenStack

Open vSwitch foi desenhado para suprir os requisitos de um ambiente virtualizado que é dinâmico. Características como ser de fácil adaptação, ser programável, remotamente controlado e monitorado fazem do Open vSwitch uma solução adequada para cenários de grande escala que requerem manutenção regular. Segundo Open vSwitch (2014c), as seguintes funcionalidades e características do OVS ajudam ambientes virtualizados:

Mobilidade de estado – a fácil identificação e a portabilidade de todos os estados da rede relacionados à uma máquina virtual entre diferentes VMMs.

Fácil adaptação às mudanças da rede – é normal em ambientes virtuais frequentes movimentações de máquinas virtuais entre VMMs e configurações lógicas de rede. O suporte a tecnologias, como NetFlow, sFlow, OVSDB e OpenFlow, auxiliam nessa tarefa.

2.5 Computação em Nuvem

De acordo com (Mell, 2011):

> "A computação em nuvem pode ser definida como um modelo para habilitar o acesso à rede de forma ubíqua, conveniente e sob demanda, a um conjunto de recursos computacionais compartilhados (por exemplo, redes, servidores, armazenamento, aplicações, serviços) que podem ser rapidamente provisionados e removidos com um esforço mínimo de gerenciamento e interação dos provedores de serviços."

Em essência, um sistema pode ser considerado uma nuvem computacional se apresentar as características (Mell, 2011):

- **Autoatendimento sob demanda:** um usuário deve possuir a facilidade de provisionar recursos computacionais de acordo com sua necessidade sem intervenção humana pelo lado do fornecedor;
- **Acesso amplo à rede:** os recursos providos pela computação em nuvem devem ser acessíveis através da rede por diferentes tipos de equipamentos, como tablets, celular, computadores, etc.;
- **Agrupamento de recurso (*resource pooling*):** os recursos são alocados dinamicamente, ou seja, o usuário não possui o controle ou não sabe exatamente onde recurso que está alocado. Porém os recursos computacionais podem ser divididos em grupos (por exemplo continente, país, estado ou *datacenter*) onde usuários possuem a possibilidade de escolher em qual grupo o recurso será alocado.
- **Elasticidade:** Do ponto de vista do usuário, a computação em nuvem é ambiente de recursos infinitos podendo requisitar e liberar recursos em qualquer momento.
- **Serviço mensurado:** um dos conceitos da computação em nuvem é a utilização da computação utilitária, proposta por um dos pioneiros da computação nos anos 60, John McCarthy, onde, previa que no futuro os serviços computacionais seriam comercializados da mesma forma que as

utilidades públicas como eletricidade, água, telefonia, ou seja, um modelo de serviço onde o usuário paga apenas pelo que efetivamente consome (Wei, 2010).

Segundo Mell (2011), os modelos de serviço são:

- Software como serviço (SaaS): apenas utiliza aplicativos que rodam na nuvem, sem se preocupar com a infraestrutura e a disponibilidade do software. Os aplicativos podem ter diversos clientes, como navegador web, dispositivos móveis, entre outros.
- Plataforma como serviço (PaaS): provê ao consumidor um conjunto de ferramentas que auxiliam na disponibilidade do fornecimento do software.
- Infraestrutura como serviço (IaaS): abstrai os recursos computacionais, de rede e armazenamento para os usuários da nuvem. O consumidor não tem controle sobre o hardware, porém tem o controle sobre o sistema operacional.

De acordo com Mell (2011), um ambiente de computação em nuvem pode ser implementado de 4 formas.

- Nuvem Privada – é provisionada exclusivamente para o uso de uma organização.
- Nuvem publica – normalmente disponibiliza a infraestrutura da nuvem pela internet, Não possui um público alvo.
- Nuvem comunitária – provisiona os recursos de forma compartilhada para organizações com objetivos em comum.
- Nuvem hibrida – composta por um ou mais tipos de implementação (público, privado e comunitária).

Nesse trabalho será abordado modelo de serviço IaaS, que pode ser encontrado em plataformas tanto de código aberto ou de código fechado. Recentemente a plataforma OpenStack, de código aberto, vem se destacando e ganhando a adoção de grandes empresas, por exemplo IBM, HP, Dell e RedHat.

2.6 OpenStack

OpenStack é um conjunto de software que possui a finalidade de controlar, gerenciar e automatizar, em larga escala, recursos computacionais, recursos de armazenamento e recursos de redes de computadores. O acesso é dado através de uma interface web ou através de interfaces de programação de aplicativos (APIs). Permitindo que os administradores tenham o máximo de controle sobre os recursos conectados ao OpenStack. E aos usuários a possibilidade de provisionar recursos de acordo com as suas necessidades (OpenStack,2014a).

OpenStack é um projeto que está em constante desenvolvimento e evolução. Tendo como média de duas versões por ano (ver tabela 2.1). Apesar de ser um projeto relativamente novo, OpenStack apresenta uma grande compatibilidade e integração com tecnologias consolidadas de virtualização, de armazenamento e de redes.

Nome	Estado	Versão	Data
Icehouse	Em desenvolvimento		17 de Abril, 2014
Havana	Versão estável, atual e suportada	2013.2	17 de Outubro, 2013
Grizzly	Versão suportada	2013.1	4 Abril, 2013
Folsom	Fim de vida	2012.2	27 de Setembro, 2012
Essex	Fim de vida	2012.1	5 de Abril, 2012
Diablo	Fim de vida	2011.3	22 de Setembro, 2011
Cactus	Obsoleto	2011.2	15 de Abril, 2011
Bexar	Obsoleto	2011.1	3 de Fevereiro, 2011
Austin	Obsoleto	2010.1	21 de Outrubro, 2010

Tabela 2.1 – Versões do OpenStack – Portal OpenStack

OpenStack pode ser instalado em sistemas operacionais GNU/LINUX e é suportado por distribuições populares, como Ubuntu[1], o que facilita a instalação da plataforma. Através do módulo Networking (Neutron), OpenStack se torna compatível

[1] http://www.ubuntu.com/

com equipamentos ou soluções de redes que permitem a implementação de redes SDN, atuando como controlador da mesma. Por essas características OpenStack é a plataforma escolhida para implementação nesse trabalho.

OpenStack na versão atual é constituído por 3 principais softwares: *OpenStack Compute*, *OpenStack Storage* e *OpenStack Networking*. Existe outros componentes que auxiliam na instalação e na operação da nuvem, como uma interface web de administração (Dashboard) e *OpenStack Shared Services* que é composto por, autenticação e autorização e catálogo de serviços (Identity) e registro de imagens de máquinas virtuais (Image Service). A figura 2.11, mostra uma visão geral do OpenStack. Outros projetos menores ainda estão em desenvolvimento.

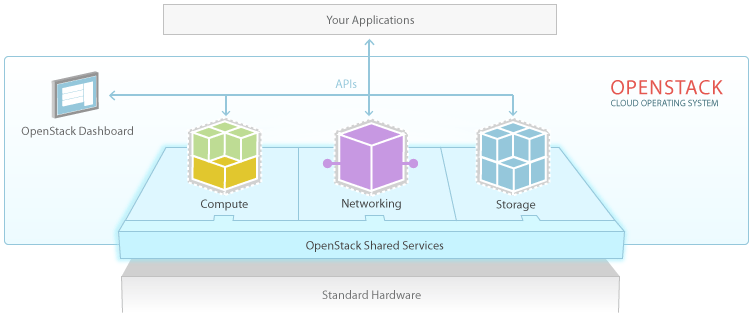

Figura 2.11 - Visão Geral do OpenStack – Portal OpenStack

OpenStack é divido em módulos que podem ser implementados ou não, ficando ao critério e a necessidade do provedor da nuvem. Os seguintes módulos são oficialmente integrados ao OpenStack Havana, que é a versão atual do software:

- **Keystone** – gerenciamento de usuários e suas permissões.
- **Glance** – gerenciamento de imagem de máquinas virtuais.
- **Cinder** – gerenciamento de discos virtuais persistentes, que são utilizados nas máquinas virtuais.
- **Nova** – gerencia o ciclo de vida das máquinas virtuais.
- **Neutron** – gerencia equipamentos de rede (hardware ou software) e prove rede como serviço para outros serviços OpenStack.

- **Horizon** – oferece uma interface gráfica baseado em web, facilitando o uso da nuvem.
- **Swift** – armazenamento de dados de forma não estruturado. É altamente escalável e redundante. Desenhado para ter alta durabilidade disponibilidade. Ideal para armazenamento de dados estático, como fotos, backup e imagem de máquinas virtuais.
- **Ceilometer** – coleta métricas dos componentes do OpenStack instalados, possibilitando que administradores da nuvem possam ter a visibilidade do uso e performance da nuvem.
- **Heat** – oferece aos desenvolvedores de aplicação, sendo esses usuários da nuvem, automatizar o processo de implementação de sua infraestrutura.

OpenStack suporta redes SDN através do módulo Networking (Neutron). Neutron possui plug-ins para diversas tecnologias de redes, inclusive redes SDN. Uma dessas tecnologias suportadas é o Open vSwitch.

Nesse capítulo será abordado brevemente cada um dos componentes. No capítulo 3, será abordado de uma forma mais detalhada o entendimento e o funcionamento de cada módulo.

2.7.1 Interface Web de Administração (Dashboard)

É um projeto que visa criar uma interface gráfica web para facilitar o acesso dos administradores e dos usuários aos componentes do OpenStack. Administradores conseguem gerenciar os serviços da nuvem e ter uma melhor visão da mesma. E os usuários uma interface amiga que facilita a execução das tarefas e o acesso aos recursos (OpenStack, 2014b)

Horizon, nome dado ao componente OpenStack Dashboard, foi desenvolvido para ser extensível, facilitando que produtos de terceiros possam se integrar à interface gráfica, como softwares de monitoramento, relatórios ou que for necessário.

2.7.2 Serviço de Identidade

Em ambientes onde os recursos são compartilhados ou acesso aos recursos necessitam ser restrito, que é o caso na computação em nuvem, se adequa um software para auxiliar no gerenciamento da identificação, na autenticação e nos privilégios de usuários que auxilia na segurança e confiabilidade do ambiente.

Com o objetivo de aumentar a segurança e prover um serviço centralizado de gerenciamento de identidade OpenStack desenvolveu um serviço de identidade, chamado de Keystone que disponibiliza uma ponto central de autenticação, autorização e um diretório de usuários. OpenStack Keystone pode se integrar a outros serviços de diretório, como LDAP (*Lightweight Directory Access Protocol*) definido na RFC 4510(Ed., 2006), e permite múltiplas formas de autenticação, como usuário e senha ou *token* (OpenStack, 2014c).

Segundo Rhoton(2013a), Keystone gerencia os catálogos de serviços do OpenStack e provê ao usuários uma lista de todos os serviços instalados no ambiente. É através do serviço de identidade Keystone, que administradores conseguem, de forma centralizada, criar projetos, usuários, políticas e permissões.

2.7.3 Serviço de Imagem

O software de serviço de Imagem, no OpenStack, é o software Glance que tem como as principais funções o serviço de descoberta, registro e entrega de imagens de máquinas virtuais. Glance apenas armazena metadados e informações da imagem e provê um repositório central de imagens para os usuários. Glance não responsável por armazenar a imagem, utiliza um serviço terceiro (Rhoton, 2013b).

2.7.4 Serviço de Computação

Nova, serviço de computação do OpenStack, é responsável por gerenciar e oferecer recursos computacionais por demanda. Recursos computacionais, entenda-se como CPU, memória, interface de rede e disco. Desenvolvedores de aplicações, administradores e usuários requisitam e liberam os recursos computacionais à através de APIs ou via interface web. Nova foi concebido para ter alta compatibilização e facilidade de adicionar novos equipamentos na infraestrutura existente, sendo que arquitetura foi desenvolvida para ser flexível e escalável horizontalmente. Capacidade de suportar diferentes tecnologias de VMM. OpenStack Nova não requer hardware ou software proprietário que credencia equipamentos comuns sejam utilizados (OpenStack, 2013).

2.7.5 Serviço de Armazenamento

OpenStack Storage provê armazenamento como serviço para os componentes do OpenStack, para aplicações, *backups* e máquinas virtuais através tecnologias já existentes de armazenamento. Na versão atual dois tipos de armazenamento são oferecidos: em objeto e em bloco.

2.7.5.1 Armazenamento em Objeto

Comumente utilizada para dados estáticos como fotos, imagens de máquina virtual, backup, arquivamento ou até mesmo por aplicações. Swift, serviço de armazenamento em objeto do OpenStack, é uma plataforma totalmente distribuída e acessível através de API.

Segundo OpenStack (2014d), Swift é um sistema de armazenamento descentralizado visando o escalonamento, redundância e durabilidade. Sua escabilidade é horizontal, simplificando que novos servidores sejam adicionados ao *cluster*. Um objeto ou arquivo uma vez armazenado é replicado em diferentes discos e servidores. Assim garantindo a integridade dos dados.

2.7.5.2 Armazenamento em bloco

Diferentemente do armazenamento em objetos, sua utilização é feita em ambientes que necessitam alto desempenho, como banco de dados, sistemas de arquivos expansíveis e até mesmo provendo blocos de armazenamento. Blocos de armazenamento são nada mais que discos virtuais, sem um formato de sistema de arquivos e que são conectados as máquinas virtuais criadas pelo OpenStack Nova. Cinder, serviço de armazenamento em bloco do OpenStack, integra desde a soluções simples de armazenamento até sistemas de armazenamento de grande porte (OpenStack, 2014d).

O sistema *Block Storage* gerencia a criação, a anexação (*attachment*) e desanexação (*detachment*) de discos virtuais nas máquinas virtuais. Sendo altamente integrado ao OpenStack Computacional e a interface web (*Dashboard*), onde usuários controlam e provisionam discos de acordo com a necessidade. Block Storage também possui um sistema de gerenciamento de *snapshots*. *Snapshot* é uma cópia fiel de bloco de armazenamento em um dado momento - pode ser utilizado como backup. *Snapshots* podem ser restaurados através de um novo bloco de armazenamento (OpenStack, 2014d).

3 Desenvolvimento

Nesse capítulo será abordado uma visão geral das tecnologias utilizadas, a definição da arquitetura da nuvem e seus componentes, a instalação da infraestrutura, a instalação do OpenStack e os requeridos softwares e as configurações necessárias para que a plataforma OpenStack esteja em pleno funcionamento.

Este trabalho consiste na implementação um ambiente de computação em nuvem no Instituto Federal Santa Catarina Campus São José (IFSC-SC). O modelo de computação em nuvem utilizado é o IaaS (Infraestrutura como serviço), que provê recursos computacionais, de redes e armazenamento de dados aos utilizadores da nuvem.

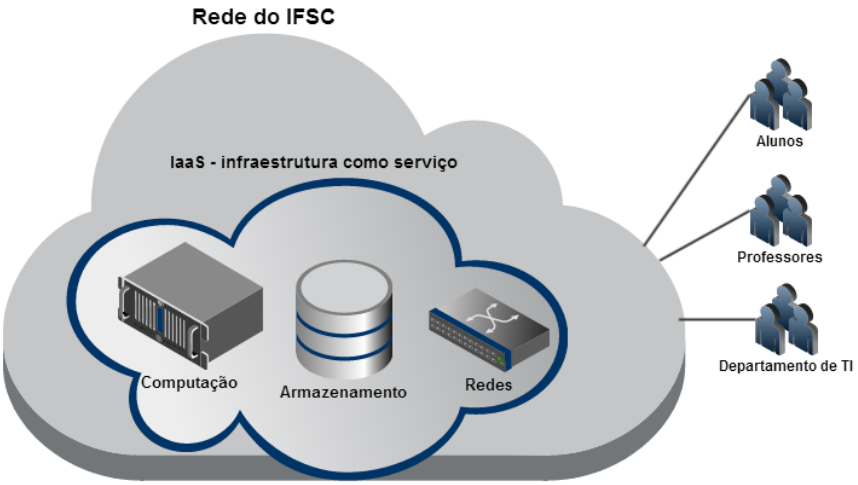

Figura 3.1 - Visão geral do IaaS – Autor

O modelo IaaS foi escolhido por se tratar da camada de abstração mais baixa da computação em nuvem e é a única que permite o acesso à redes virtualizadas, exatamente onde as redes SDN atuam. Na figura 3.2 é possível evidenciar os a responsabilidades do provedor e do usuário no modelo IaaS.

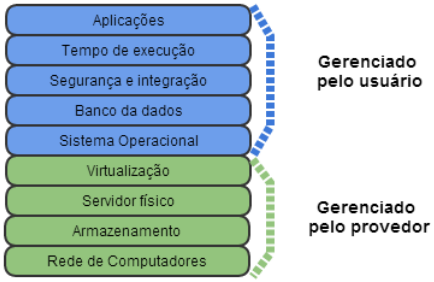

Figura 3.2 - Responsabilidades no modelo IaaS – Google, Imagens

3.1 Arquitetura

OpenStack possui uma gama considerável de possibilidades de instalação e na sua implementação. Nesse trabalho será implementado uma arquitetura básica do OpenStack. Sendo assim, visando apenas o simples funcionamento do ambiente e não alta performance ou a alta disponibilidade do sistema. Uma arquitetura simples é composta basicamente por 3 componentes, sendo eles o controlador da nuvem, o nó de rede e o nó de computação, ver figura (Figura 3.3).

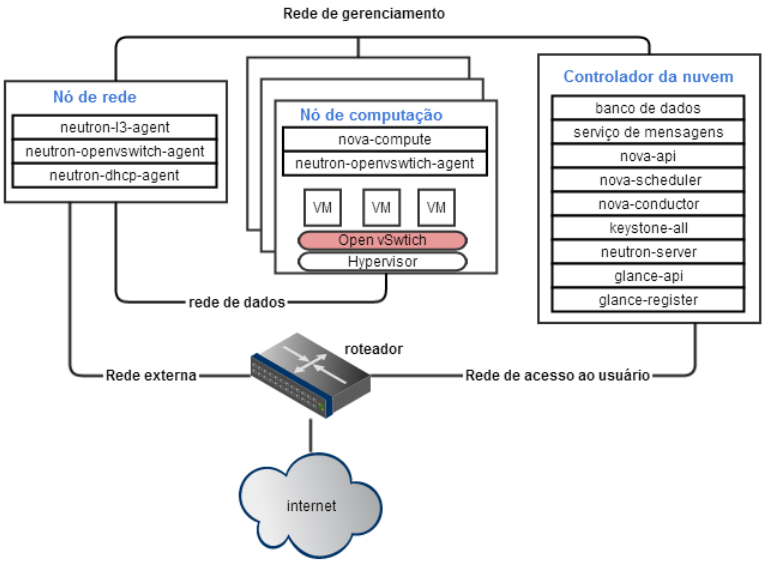

Figura 3.3 - Arquitetura básica – Autor

O nó controlador da nuvem é responsável por controlar os serviços como Nova, Neutron, Keystone, Glance, entre outros. É também no controlador que serviços secundários rodam, como o banco de dados relacional e o serviço de mensagens. Os serviços secundários são utilizados por quase todos os componentes do OpenStack.

O nó computacional é controlado pelo nó controlador da nuvem, através do modulo nova-compute. É nesse nó que o VMM (*hypervisor*) é instalado e consequentemente onde as máquinas virtuais são instanciadas. É possível ter mais de um nó computacional.

O nó de rede é responsável por fornecer a rede e serviços de rede para as máquinas virtuais, como roteamento, NAT[1] (*Network Address Translation*) e DHCP[2] (*Dynamic Host Configuration Protocol*). Esse nó também interligar a máquinas virtuais que estão em diferente VMM (*hypervisor).* O nó da rede também é responsável pela conectividade entre as máquinas virtuais e redes externas, por exemplo a internet.

Na figura 3.3 é possível visualizar 3 redes diferentes e isoladas. A rede de gerenciamento é utilizada apenas para comunicação entre componentes OpenStack. Por questões de segurança essa rede deve ser somente acessível para os componentes da nuvem. A rede de dados é utilizada para trafegar os dados entre as máquinas virtuais e é através dessa rede que serviços de rede (DHCP, NAT, roteamento) provido pelo nó de rede são entregues as máquinas virtuais. As redes rede externa e rede de acesso ao usuários podem ser ou não a mesma rede. A rede externa permite que máquinas virtuais acessam a internet ou sejam acessadas pela internet. A rede de acesso ao usuário é a rede que o usuário utiliza para acessar a nuvem, podendo ser através do OpenStack Horizon ou por APIs dos componentes OpenStack.

O cenário considerado ideal para fins de testes é composto por um nó controlador, 1 nó de rede e 2 nós de computação, assim a possibilidade simulações são maiores. Por exemplo, migrar uma máquina virtual de nó de computação para outro.

[1] https://www.ietf.org/rfc/rfc1631.txt
[2] https://www.ietf.org/rfc/rfc2131.txt

3.2 Implementação

3.2.1 Recurso disponível

Os seguintes recursos do IFSC-SJ foram alocados de acordo com as necessidades técnicas do projeto:

- 1 – Rack Piso 19” fechado
- 1 – Switch Dlink DGS-3100-24
- 3 – HP Workstation Z220
- Memória: 16GB RAM
- Processador: Intel® Xeon® Processor E3-1225v2
- Placa de Rede: Intel 82579LM Gigabit Network e RTL8169 Gigabit Network
- Armazenamento: 1TB 7200 RPM

3.2.2 Infraestrutura física

Entende-se como infraestrutura física as instalações dos equipamentos no rack, a conexão dos equipamentos à energia elétrica física e o cabeamento estruturado. Nessa etapa os computadores foram conectados ao switch, ressaltando que nenhuma configuração de rede foi executada.

Figura 3.4 - Visão frontal e lateral do rack - Autor

3.2.3 Infraestrutura hospedeira

Nesta etapa é consolidada a definição da função de cada computador, a instalação do sistema operacional, a definição de endereços de rede à ser utilizada e a configuração das interfaces de rede.

O sistema operacional utilizado foi GNU/Linux e a distribuição foi o Ubuntu Server 13.10, pois suporta oficialmente todos as tecnologias necessárias para o desenvolvimento do projeto.

Como descrito no item 3 desse documento, por questões de segurança as 3 redes utilizadas devem ser diferentes e isoladas, ou seja, não deve ter uma comunicação entre elas. Então tentou-se implementar o padrão 802.1Q(IEEE, 2011) que tem o objetivo principal de criar múltiplas redes locais virtuais (VLAN - *virtual local*

area network) em uma mesma rede local física, com a capacidade de restringir o tráfego entre as VLANs. O que resultaria na criação de uma VLAN para cada rede, que estariam isoladas entre si e aumentaria a segurança. Apesar de todos os equipamentos de rede (switch e interfaces de rede) suportarem a implementação de VLAN, a mesma não pode ser finalizada com sucesso. Por questão de tempo não foi possível uma investigação mais apurada para resolver o problema encontrado. Sabemos o quanto é importante a segurança na computação em nuvem, porém esse problema não é impeditivo para a continuação do projeto.

Na figura 3.5, ilustra a infraestrutura hospedeira consolidada e que o nó de controle e de rede ficam no mesmo hospedeiro. Devido ao número de três computadores disponíveis não foi possível criar a proporção de um computador para cada nó.

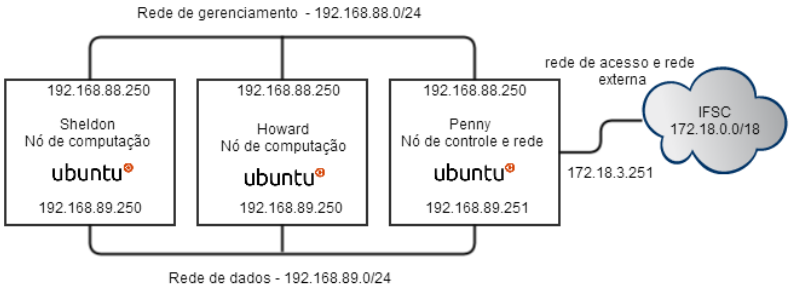

Figura 3.5 – Infraestrutura hospedeira

3.2.4 Instalação do OpenStack

Um dos objetivos do trabalho é disponibilizar uma documentação da instalação feita. Nesse item será abordada a sequência lógica dessa instalação. A documentação em si é encontrada em anexo deste trabalho ou a versão mais atualizada no endereço http://wiki.sj.ifsc.edu.br/wiki/index.php/OpenStack_com_Open_vSwitch_e_SDN_-_Guia_de_Instala%C3%A7%C3%A3o.

Os passos de instalação dos componentes do OpenStack possuem procedimentos semelhantes. A figura 3.6, mostra o procedimento genérico de instalação que pode ser dividido em instalação do software, configuração da conexão

com o banco de dados e criação de tabelas para os componentes que utilizam banco de dados, configuração específica do componente.

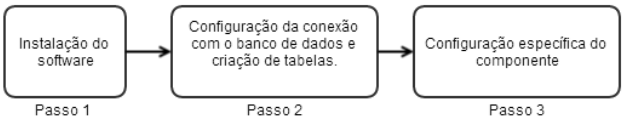

Figura 3.6 - Instalação genérica

A instalação dos softwares no sistema operacional hospedeiro se dá através da ferramenta APT[1] (*Advanced Packaging Tool*), que facilita a instalação e remoção de softwares no GNU/Linux. A configuração da conexão do banco de dados e a configuração específica do componente é dada através de arquivos de configuração.

3.2.5 Banco de dados e servidor de mensagens

A maioria dos módulos do OpenStack requer banco de dados para armazenar informações. Nova, Neutron, Cinder, Keystone e Glance são os componentes dessa arquitetura que necessitam de uma base de dados. É criada uma base de dados individual para cada um desses componentes. O banco de dados utilizado foi o recomendado pelo projeto, MySQL Community Server[2], atualmente na versão 5.5.34.

[1] https://wiki.debian.org/Apt
[2] http://www.mysql.com/

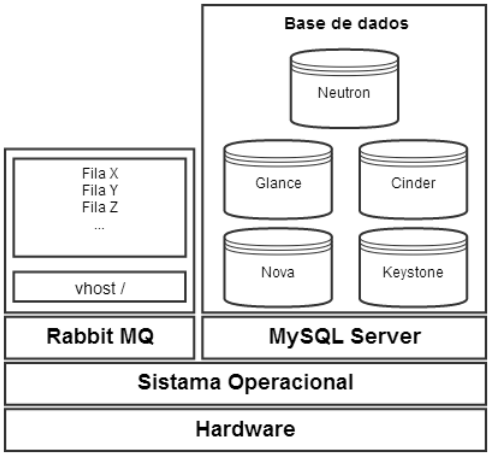

Figura 3.7 - Visão geral do banco de dados e o servidor de mensagem

Os módulos do OpenStack utilizam o Protocolo Avançado de Filas de Mensagem (AMQP[1] - *Advanced Message Queuing Protocol*) para fazer a comunicação entre si. AMQP é um protocolo para publicação, enfileiramento, armazenamento e consumo de mensagens. Um servidor AMQP é necessário para intermediar as mensagens. O servidor AMQP utilizado foi o RabbitMQ[2], atualmente na versão 3.1.3, pois é suportado pelo OpenStack e de código aberto.

Os dois serviços devem estar acessíveis apenas pela rede de gerenciamento.

3.2.6 Serviço de identidade e imagem

Os serviços de identidade e imagem, Keystone e Glance respectivamente, são considerados como serviços secundários que auxiliam os três principais módulos: OpenStack Computação, Redes e Armazenamento. Keystone e Glance facilitam a implementação e a operação da nuvem. Por isso a instalação desses serviços primeiro.

O procedimento de instalação do software Keystone pode ser baseado na Figura 3.6 e Figura 3.8. Keystone utiliza o banco de dados para armazenar informações. A configuração da conexão com o banco de dados é feito no arquivo

[1] http://www.amqp.org/
[2] https://www.rabbitmq.com/

"*/etc/keystone/keystone.conf"*. Uma vez a conexão estabelecida, utiliza-se o comando "*keystone-manage db_sync"* para criar as tabelas no banco.

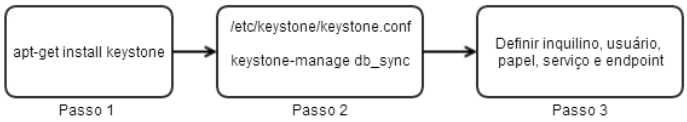

Figura 3.8 - Sequência de instalação do Keystone

Nesse momento o administrador pode acessar o serviço Keystone, que está em pleno funcionamento, e definir os elementos inquilino (*tenant)*, usuários, papel (*role*), serviços e *endpoints*. Baseado no OpenStack (2014e), será feito uma breve descrição sobre cada um dos elementos:

- ***Tenant*** – representa o contentor do recurso (usuário), podendo representar um grupo, uma organização, uma conta ou um projeto.
- **Usuário** – uma representação digital de uma pessoa, um sistema ou serviço que utiliza OpenStack. É associado as informações de usuário, senha e e-mail.
- **Papel (*role*)** – é a posição que o usuário assume ao executar uma operação. Uma role inclui um conjunto de direitos e privilégios sobre um tenent.
- ***Serviço*** – é um serviço do OpenStack, por exemplo Nova, Glance, Neutron.
- ***Endpoint*** – um endereço de rede que dá acesso a um serviço.

Cada serviço OpenStack necessita ser representado por um usuário e ser adicionado ao *tenent* "service" e ao *role* "admin". Par fins administrativos é criado um usuário mestre "admin" e adicionando a *role* "admin". Também é necessário definir os serviços que serão disponibilizados na nuvem e criar seus respectivos *endpoints.* No guia de instalação foi disponibilizado um programa que facilita e automatiza esse processo. Na Figura 3.9 é possível visualizar a sequência de criação dos elementos. A Figura 3.10 mostra os componentes e suas relações.

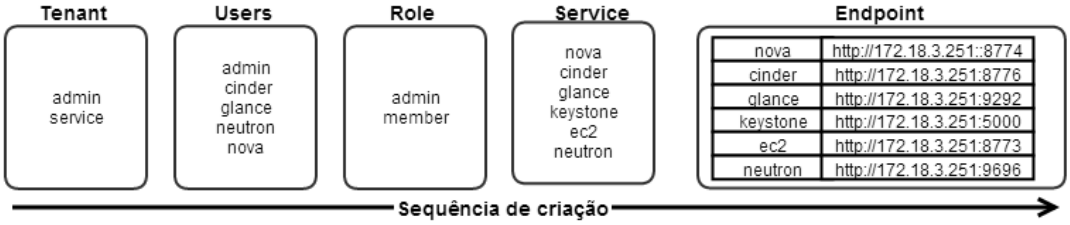

Figura 3.9 - Sequência na criação dos elementos do Keystone.

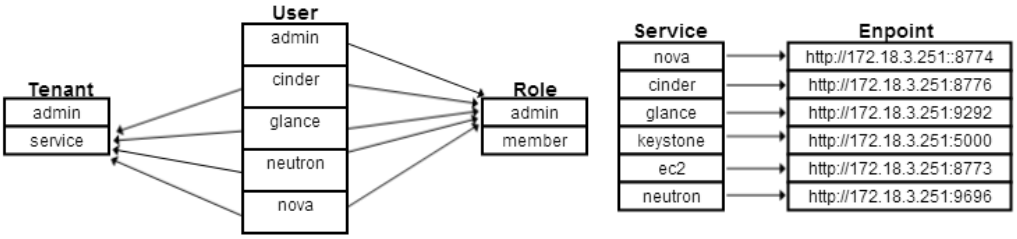

Figura 3.10 - Componentes criados e suas relações.

O processo de instalação e configuração do Glance segue a lógica da Figura 3.6 e Figura 3.11. O serviço Glance utiliza o Keystone para a autenticação. Para isso é preciso adicionalmente configurar os arquivos “/etc/glance/glance-api.conf” e “/etc/glance/glance-registry.conf” utilizando as informações inseridas durante o passo 3 da instalação do Keystone.

Figura 3.11 - Sequência de instalação do Glance.

O serviço de imagem do OpenStack está pronto para o uso. Nesse trabalho foi utilizado imagens prontas do Ubuntu Server Saucy 13.10 que podem ser encontradas na URL http://uec-images.ubuntu.com/saucy/current/.

3.2.7 Serviço de Rede do OpenStack

Atualmente Neutron é o software que provê serviços avançados de rede no OpenStack. Segundo OpenStack (2014e), Neutron é composto pelos componentes:

- **Servidor neutron** (*neutron-server* e neutron-*-plugin) – requer acesso ao banco de dados e acessa o servidor de mensagem para comunicação entre processos. É responsável por fornecer o serviço de API.
- **Agente plugin** (neutron-*agent) –é necessário independente do *plugin* utilizado e executado em cada nó de computação. Necessita do acesso ao servidor de mensagens.
- **Agente DHCP** (neutron-dhcp-agent) – requer acesso ao servidor de mensagens e é responsável por prover serviço de DHCP para as redes dos inquilinos (*tenant*).
- **Agente L3** (neutron-l3-agent) – prover serviço de NAT que dá acesso à rede externa para as máquinas virtuais.
- **Serviço provedor de rede** – entende-se como serviços e servidores SDN. Provê serviços adicionar de rede para os inquilinos (*tenant*) de rede

A Figura 3.12, visualiza a arquitetura lógica do Neutron e a comunicação entre os componentes.

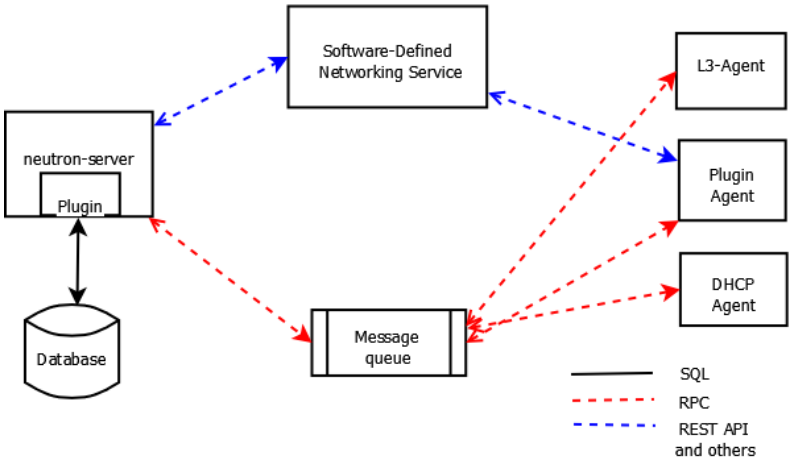

Figura 3.12 – Arquitetura Neutron – Portal OpenStack

3.2.8 Serviço de Computação do OpenStack

Nova se comunica com grande parte dos serviços do OpenStack requisitando serviços de rede, armazenamento, identidade e imagem. É considerada a principal parte da infraestrutura como serviço. Segundo OpenStack (2014e), os componentes do Nova são:

> **Serviço nova-api** – suporta as APIs do tipo OpenStack Compute, Amazon EC2[1] e uma especial para administradores Admin. Responsável por aceitar e responder requisições de API.
> **Serviço nova-api-metadata** – recebe requisições das máquinas virtuais do tipo metadado.
> **Processo nova-compute** – é um *daemon* responsável por receber ações (e. g.criar, pausar, terminar uma VM), através da fila de mensagens. Interage com APIs dos VMMs.

[1] http://aws.amazon.com/ec2/

Processo nova-scheduler – identifica requisições de VM através da fila de mensagens e determina em qual nó de computação (VMM) a VM será alocada.

Módulo nova-conductor – intermedia a comunicação entre nova-compute e o banco de dados, o que elimina o acesso direto.

Daemon nova-network – através da fila de mensagens recebe requisições de rede e executa-as para manipular a rede. Esse daemon está desuso pois Neutron assumiu.

Script nova-dhcpbridge – identifica endereços de IP alocados por VM e registra no banco de dados. Esse script está desuso pois Neutron assumiu.

Daemon nova-consoleauth – autoriza *tokens* para usuários que utilizam proxies de console.

Daemon nova-novncproxy – intermedia conexões VNC[1] para acessar VMs em execução. Suporta clientes baseados em navegadores web.

Daemon nova-xvpnvncproxy - intermedia conexões VNC[2] para acessar VMs em execução. Suporta um cliente Java desenvolvido pelo OpenStack.

Daemon nova-cert – gerencia certificados x509[3].

Cliente nova – usuário utiliza para interagir com Nova.

Client nova-manage – utilitário para administradores nuvem

A Figura 3.13 mostra seus componentes e suas interligações.

[1] https://tools.ietf.org/rfc/rfc6143.txt
[2] https://tools.ietf.org/rfc/rfc6143.txt
[3] http://www.ietf.org/rfc/rfc2459.txt

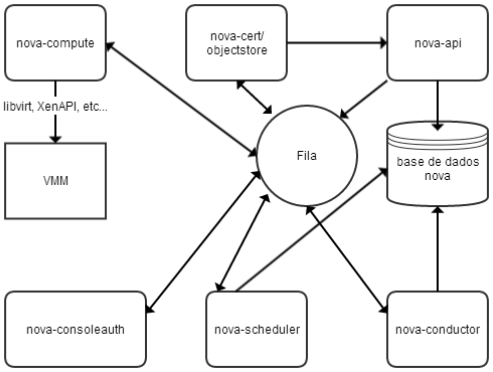

Figura 3.13 - Arquitetura lógica do Nova – Portal OpenStack

Os passos de instalação seguem a lógica da figura 3.9. No nó de controle são instalados os componentes do Nova que possuem a finalidade de controlar o serviço. Uma configuração adicional foi feita para forçar a mudança do VMM quando uma requisição de mudança do tamanho de uma máquina virtual, o que vai facilitará nos testes abordado futuramente. Nos dois nós de computação é instalado o módulo do nova com o *driver* do VMM para criar a interface de comunicação com o VMM. Nesse trabalho foi utilizado KVM[1] (Kernel-based Virtual Machine) como VMM e o *driver* libvirt[2]. A Figura 3.14 mostra como o VMM é gerenciado pelo componente Nova.

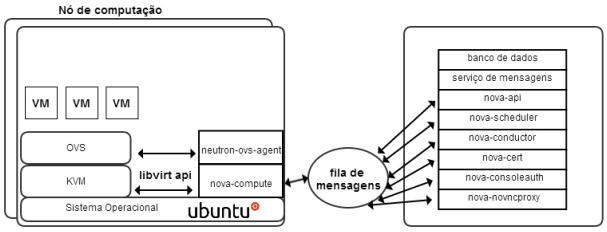

Figura 3.14 – Gerenciamento das VMM pelo componente NOVA

3.2.9 Serviço de armazenamento em bloco do OpenStack

[1] http://www.linux-kvm.org/page/Main_Page
[2] http://libvirt.org/

Cinder é o serviço de armazenamento em bloco do OpenStack que gerencia volumes e *snapshots*. Cinder possui três componentes:

- **cinder-api** – responsável por aceitar requisições de API e rotear para o cinder-volume.
- **cinder-volume** – responde as requisições, acessa o banco de dados, interage com outros processos através do protocolo AMQP e através de *drivers* acessa tecnologias de armazenamento, como Network Attached Storage (NAS) e Logical Volume Manager (LVM).
- **cinder-scheduler** – identifica o(s) nó(s) provedor(es) de armazenamento em bloco e determina em que nó será criado o volume virtual.

A Figura 3.15 mostra seus componentes e suas interligações.

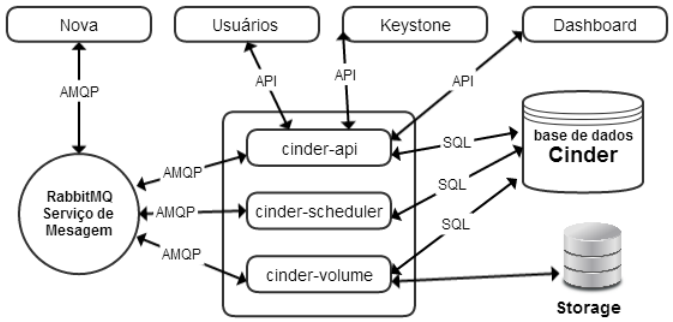

Figura 3.15 - Arquitetura do Cinder

Cinder possui drivers que controlam diferentes soluções de armazenamento. O LVM, uma das soluções suportadas, é um sistema de gerenciamento de volumes lógicos, que possibilita uma maior flexibilidade em particionar um disco do que o método tradicional (Ubuntu, 2005). Os volumes lógicos criados no LVM podem ser acessados pela rede de computadores através do protocolo *Internet Small Computer System Interface* (iSCSI). Definido na RFC 3720 (Satran et al., 2004) tem como principal função disponibilizar a comunicação entre sistemas e dispositivos de entrada e saída, especialmente dispositivos de armazenamento. As

tecnologias LVM e iSCSI são suportadas pelo OpenStack e pelo GNU/Linux. Nesse trabalho foi utilizado LVM para armazenar os blocos de armazenamento criado pelo Cinder, iSCSI para tornar esses blocos de armazenamento acessíveis pela rede e conectar(*attach*) e para desconectar(*detach*) os volumes lógicos nas máquinas virtuais, como mostra a Figura 3.16

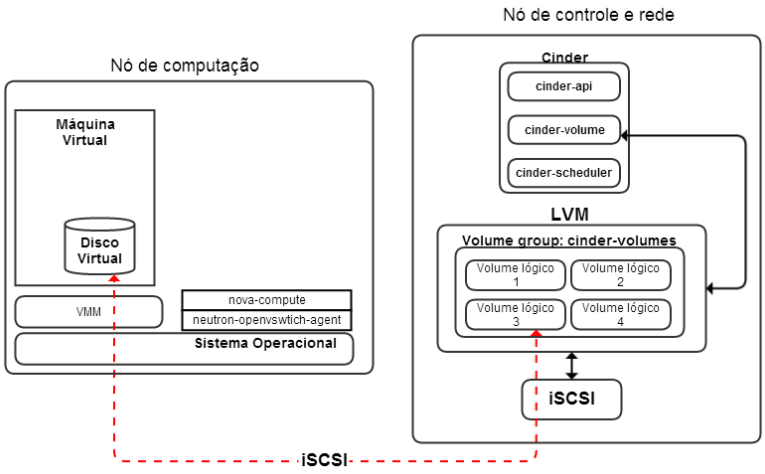

Figura 3.16 - Cinder e LVM

O processo de instalação e configuração do Cinder segue a lógica da Figura 3.6. O serviço Cinder utiliza o Keystone para a autenticação. A configuração se dá nos arquivos de configuração do Cinder “/etc/cinder/api-paste.ini” e “/etc/cinder/cinder.conf”, utilizando as informações inseridas durante o passo 3 da instalação do Keystone, ver figura Sequência de instalação do Keystone. A configuração adicionais foram feitas no arquivo “/etc/cinder/cinder.conf”, por exemplo o *driver* LVM e o nome do *volume group* que o Cinder controlará.

3.2.10 Interface Web de Administração (Dashboard)

A instalação do Horizon requer algumas condições: OpenStack Compute dever estar funcionando, Keystone deve estar funcionando e acessível para Dashboard, O

usuário do serviço Keystone deve rodar o Apache[1] com privilégios de *sudo*[2] e Python[3] 2.6 ou 2.7 com suporte a Djngo[4]. O serviço Horizon está disponível na url http://tele.sj.ifsc.edu.br/horizon

A mostra os componentes do OpenStack que a Dashboard se comunica.

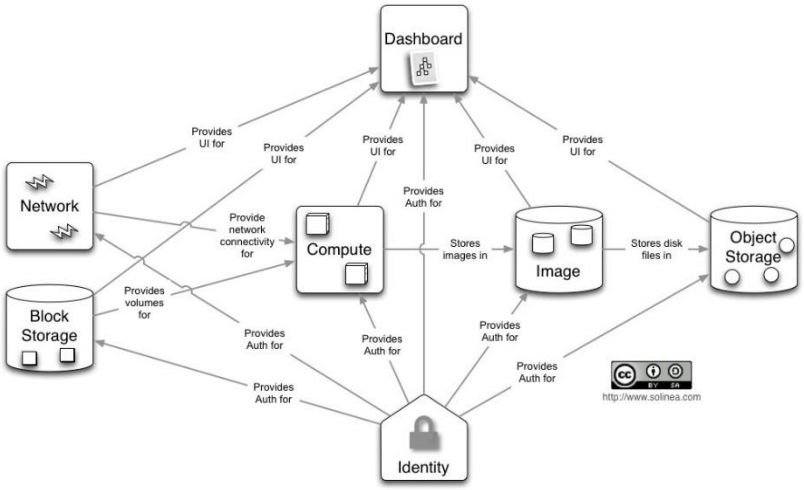

Figura 3.17 - Visão geral Dashboard – Google Imagens

[1] http://httpd.apache.org/
[2] http://www.sudo.ws/sudo/
[3] http://www.python.org/
[4] https://www.djangoproject.com/

4 Testes e Resultados

Nesse capítulo são apresentados alguns cenários de testes visando situações reais de uso do ambiente OpenStack.

4.1 Multi-inquilinos

Uma conta (*tenant*) no OpenStack dá a possibilidade de um usuário provisionar recursos computacionais, de rede e armazenamento da nuvem. Uma conta pode conter mais de um usuário e um usuário pode estar em mais de uma conta. Por padrão, não existe um portal de autoatendimento para que futuros usuários possam criar contas sem a intervenção do administrador da nuvem. O administrador da nuvem precisa criar a conta manualmente através da Dashboard ou do cliente Keystone. Nesse cenário será criado duas contas “IFSC” e “Alunos”, onde os usuários de uma conta não deve visualizar e utilizar os recursos provisionados da outra. Foi criado os usuários “Usuário1” e “Usuário2”, onde serão associados à conta “IFSC” e “Alunos” respectivamente. Na Figura 4.1 é possível visualizar a sequência lógica da criação e como é estruturado no Keystone.

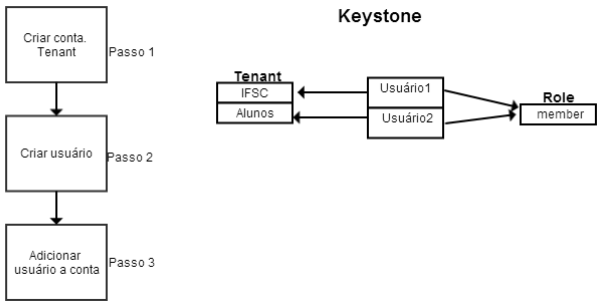

Figura 4.1 – Sequência lógica Keystone

4.1.1 Criar contas e usuários

Passo 1) Será utilizado a Dashboard e a conta de administrador para a criação. Ver Figura 4.2

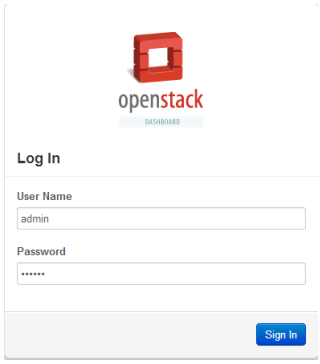

Figura 4.2 - Portal de autenticação – OpenStack

Passo 2) Criar contas. No menu do lado esquerdo da tela, clica em “Projects”. No lado direto superior da tela aparecerá um botão “Create Project”, clica nesse botão. Ver Figura 4.3

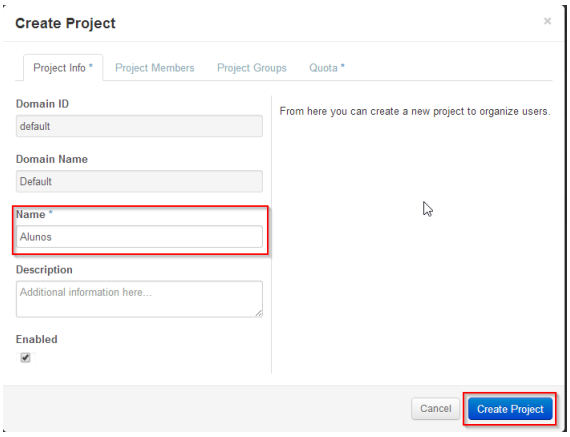

Figura 4.3 - Criar Projeto – OpenStack

Passo 3) Criar usuários. No menu do lado esquerdo da tela, clica em “Users”. No lado direto superior da tela aparecerá um botão “Create User”, clica nesse botão. Ver figura 4.4

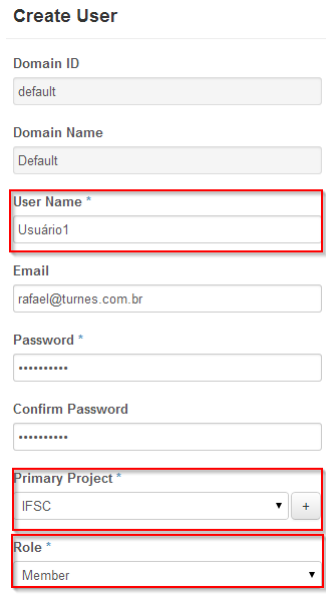

Figura 4.4 - Criar usuário – OpenStack

4.1.2 Alocar recursos nas duas contas

Primeiramente foi criado um volume (discos virtuais) em cada uma das contas. Os volumes da conta IFSC não devem ser vistos pela conta Alunos e vice e versa. Na figura 4.5, mostra a lista de volumes das contas via dashboard.

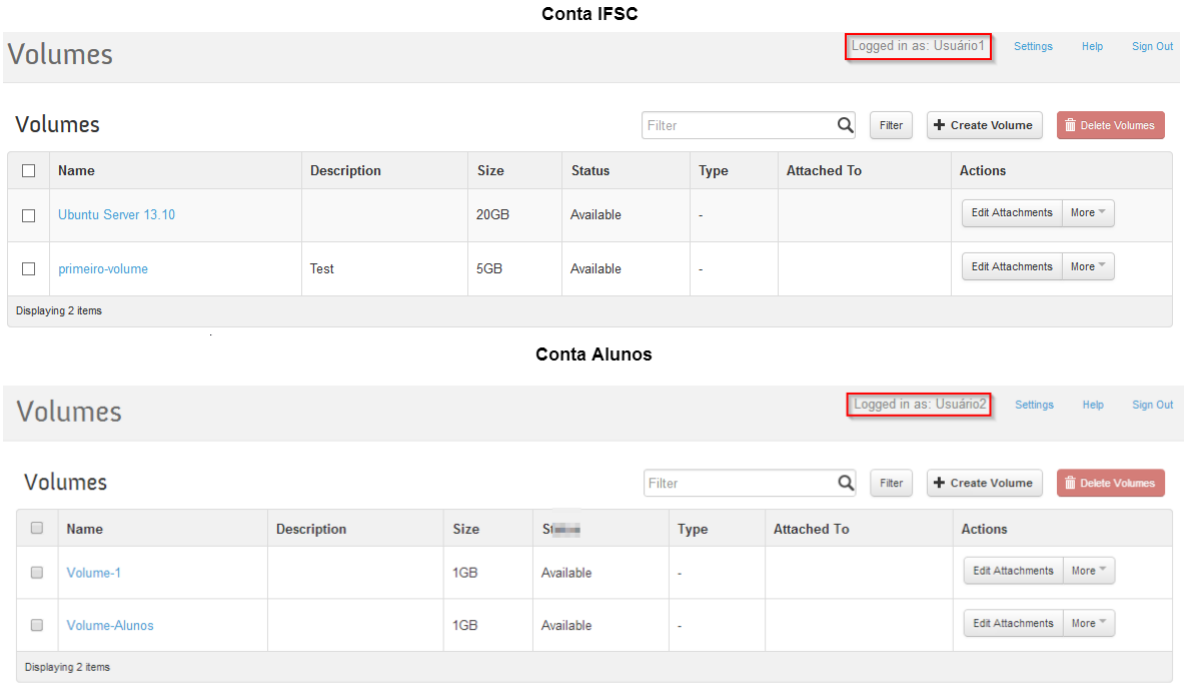

Figura 4.5 - Lista de volumes – OpenStack

4.2 Registro de imagens

Nesse item foi utilizado uma imagem pronta do Ubuntu, que pode ser encontrada na URL “http://uec-images.ubuntu.com”. Esse teste consiste em testar o serviço de imagens do OpenStack através da dashboard.

Passo 1) Acessar a Dashboard.

Passo 2) Registrar imagem com o usuário “Usuário1” da conta “IFSC”. No menu do lado esquerdo da tela clicar no botão “Images & Snapshot”. Clique no botão “Create image” que irá aparecer. Nesse momento você pode escolher uma imagem local ou a URL da imagem, foi utilizado o método via URL, ver figura 4.6. Uma imagem pode ser pública ou privada, foi registrado uma pública e uma privada.

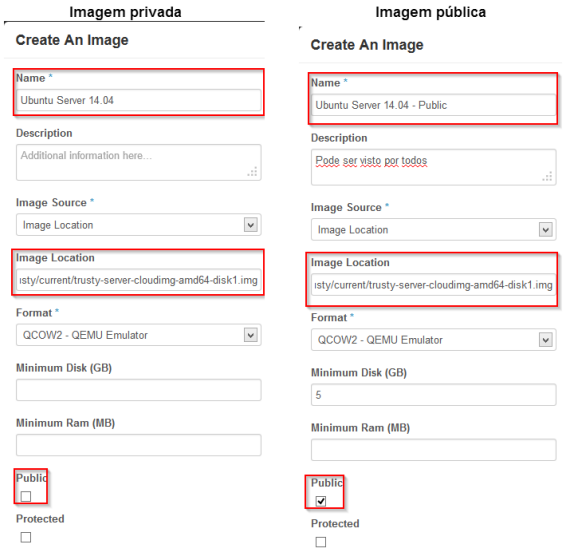

Figura 4.6 - Registro de imagens no Glance – OpenStack

Passo 3) Listar imagens com os usuários das conta “IFSC” e “Alunos” e comparar o resultado. Os usuários da conta “Alunos” devem ver apenas a imagem pública e os usuários da conta “IFSC” devem ver as duas criadas no passo 2. O resultado pode ser visto na figura 4.7.

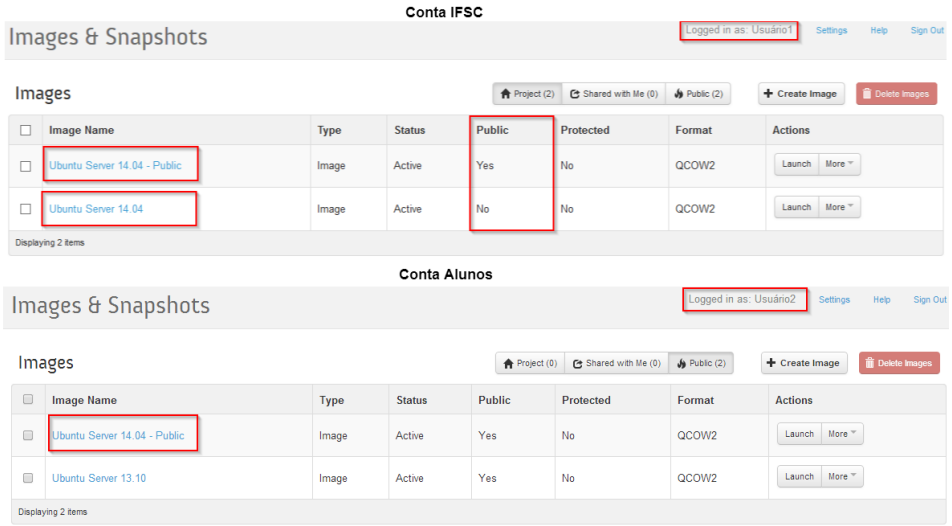

Figura 4.7 - Lista de Imagens - OpenStack

4.3 Serviços de rede

4.3.1 Rede externa

A rede externa é uma rede compartilhada e é utilizada pelas VMs para acessar a internet. Apenas o administrador tem a permissão para fazer essa tarefa. Também será alocado a faixa de IPs 172.18.3.180 até 172.18.3.190, que previamente foram disponibilizados pelo IFSC para atribuir as VMs, o que tornam acessíveis pela rede externa.

Passo 1) Acessar a dashboard

Passo 2) Criar a rede externa. Menu do lado esquerdo da tela clicar no botão “Networks”. Clique no botão “Create network” que irá aparecer. ConFigura de acordo com as figuras 1, 2 e 3

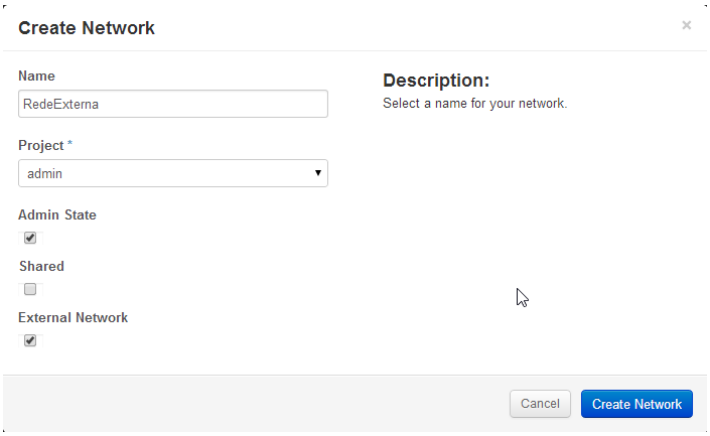

Figura 4.8 - Criando rede externa, aba 1 - OpenStack

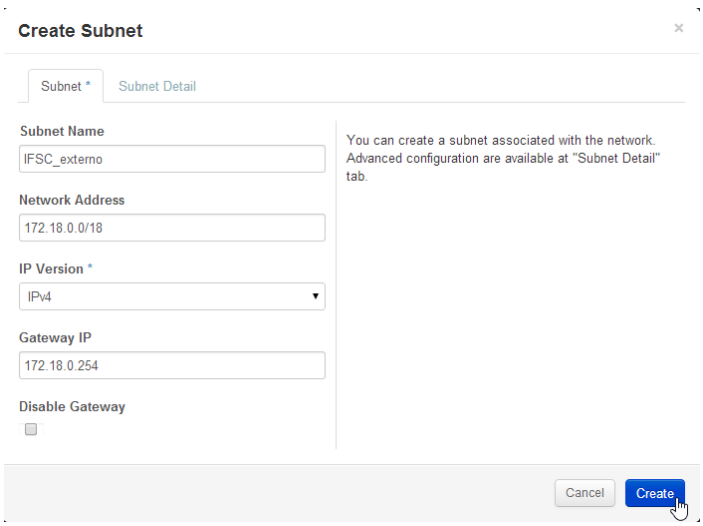

Figura 4.9 - Criando rede externa, aba 2 - OpenStack

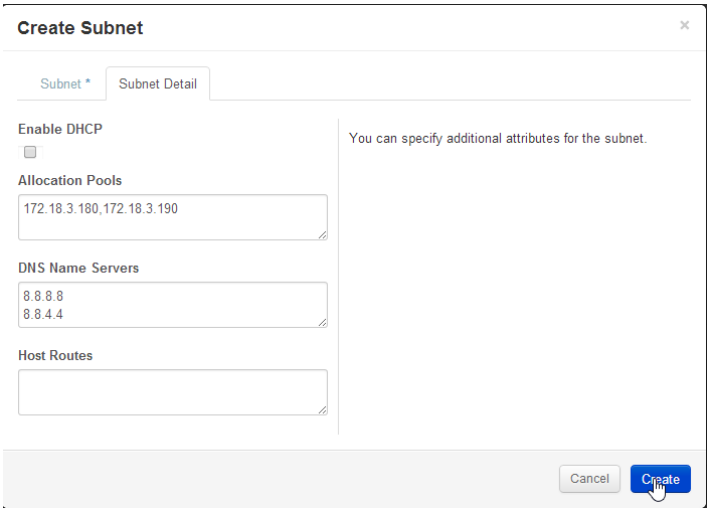

Figura 4.10 - Criar rede externa, aba 3 - OpenStack

Passo 3) Visualizar a rede externa. Menu do lado esquerdo da tela clicar no botão “Network Topology”. Na figura 4.11, representa a visão de qualquer conta da nuvem, pois essa rede é compartilhada e utilizada por todas as contas.

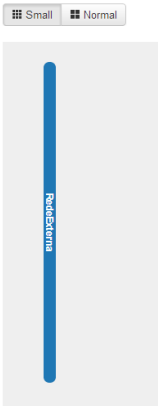

Figura 4.11 - Visão geral da rede – OpenStack

4.3.2 Rede local

Neste contexto, rede local refere-se a uma rede lógica que interliga VMs de uma mesma conta. Essas redes são facilmente criadas pelo usuário através da Dashboard. Nesse item será criado uma rede e um roteador para cada conta e a

conexão entre a rede local e rede externa. Serviços DHCP e DNS também serão configurados para a rede local.

Passo 1) Acessar a dashboard.

Passo 2) Criar um roteador. Menu do lado esquerdo da tela clicar no botão “Routers”. Clique no botão “Create Router”, que irá aparecer. ConFigura de acordo com a figura 4.12.

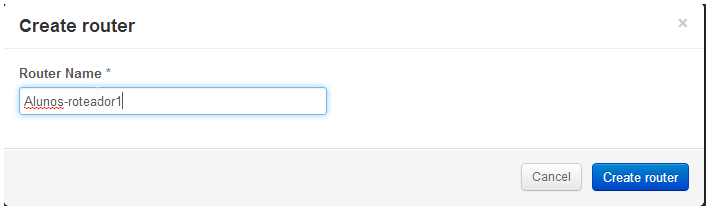

Figura 4.12 - Criando roteador - OpenStack

Passo 3) Criar a rede local para cada conta. Menu do lado esquerdo da tela clicar no botão “Networks”. Clique no botão “Create network”, que irá aparecer. ConFigura de acordo com as figuras 1, 2 e 3.

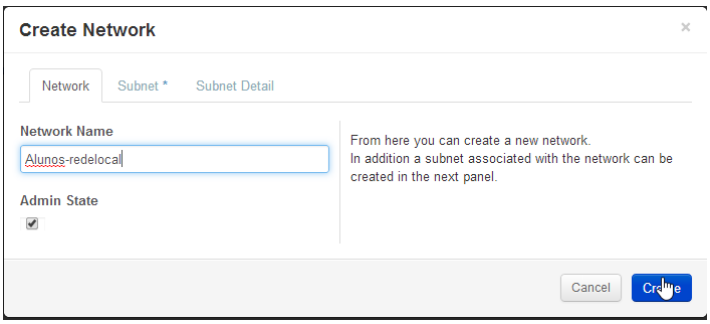

Figura 4.13 - Criando rede local, aba 1 - OpenStack

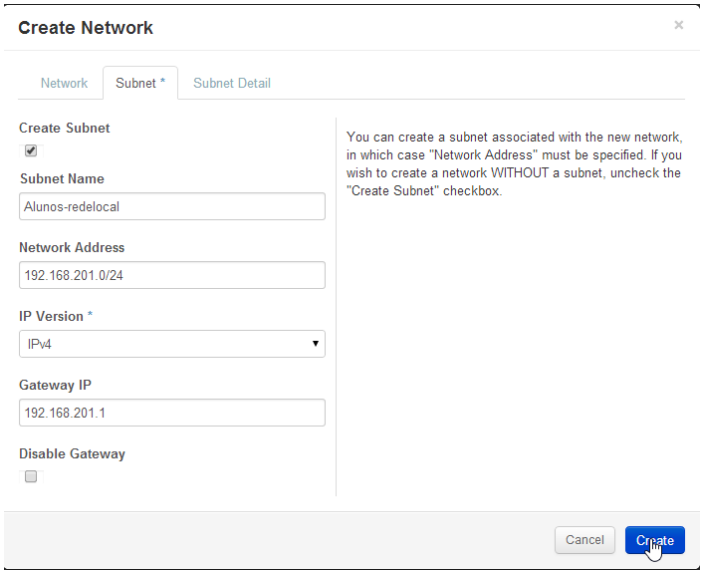

Figura 4.14 - Criando rede local, aba 2 - OpenStack

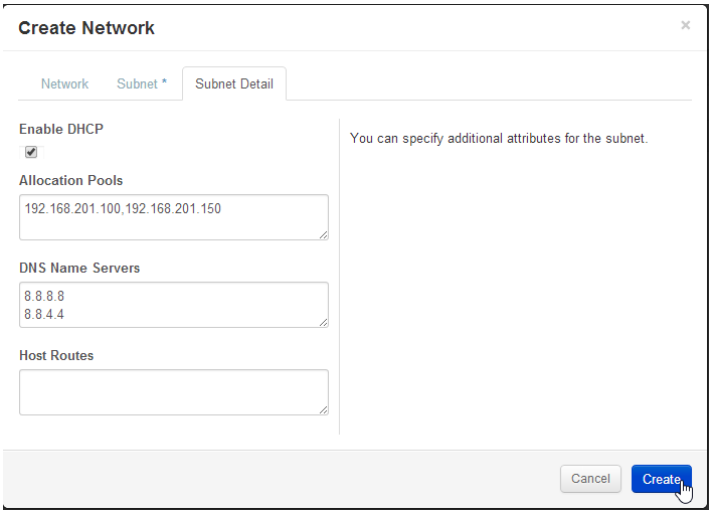

Figura 4.15 - Criando rede local, aba 3 - OpenStack

Passo 4) Visualizar a rede externa. Menu do lado esquerdo da tela clicar no botão “Network Topology”. A rede local criada por uma conta não pode ser visualizada pela outra. Na figura 4.16, é possível ver que os componentes criados 2 e 3 que ainda não estão conectados.

Figura 4.16 - Visão geral da rede – OpenStack

4.3.3 Interligar as redes no roteador.

Passo 1) Para a rede externa, menu do lado esquerdo da tela clicar no botão “Routers”. Selecione o roteador criado no item 2 e clica no botão “Set Gateway”. ConFigura de acordo com a figura 4.17

.

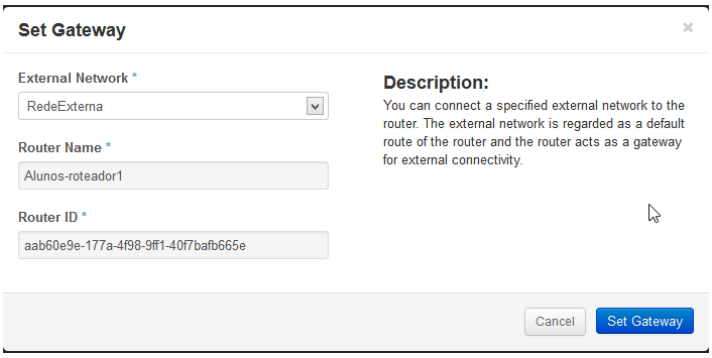

Figura 4.17 - Interligando o roteado à rede externa – OpenStack

Passo 2) Para a rede local, menu do lado esquerdo da tela clicar no botão “Routers”. Selecione o em cima do nome do roteador criado no item 2, clique no botão “Add Interface” que irá aparecer. Configura de acordo com a figura 4.18.

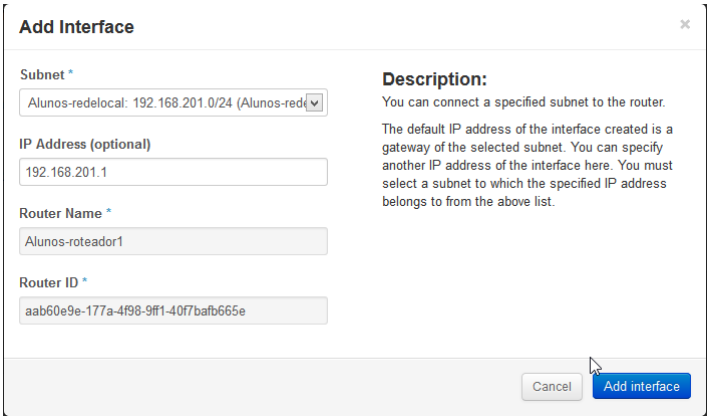

Figura 4.18 - Interligando o roteador à rede local – OpenStack

Passo 3) Visualizar a rede externa novamente. Menu do lado esquerdo da tela clicar no botão “Network Topology”. A figura 4.19, mostra que as redes estão interligadas pelo roteador.

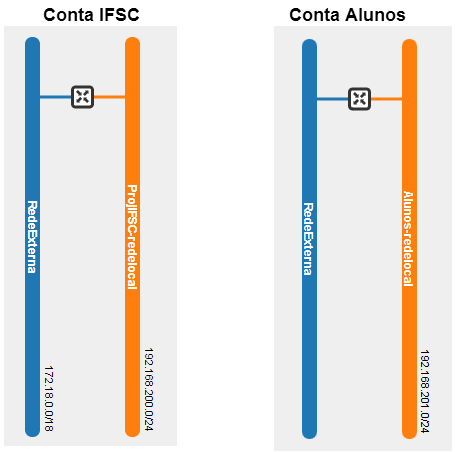

Figura 4.19 - Visão Geral da rede – OpenStack

4.4 Instanciando uma VM

A maioria dos componentes do OpenStack provêm serviços para componente Nova. Esses serviços foram testados nos itens anteriores desse capítulo. Para instanciar uma VM e deixa-la acessível para o usuário é preciso fazer alguns passos

antes, como criar uma rede local (ver item 4.3), configurar o *firewall*[1], criar uma *keypair*[2], utilizar uma imagem de uma máquina virtual (ver item 4.2). Nesse cenário será instanciada uma máquina na rede local e deixar acessível para a rede física do IFSC através dos serviços de SSH[3] (*Secure Shell*) e ICMP (Internet Control Message Protocol)[4].

Passo 1) Criar uma *keypair*. Menu do lado esquerdo da tela clicar no botão “Access & Security”. Selecione a aba “Keypairs” e clique no botão “Create Keypair”. Dê um nome e, por exemplo alunos-servidor, e armazene o arquivo (extensão) gerado pelo OpenStack.

Passo 2) Configurar Security Groups (*firewall*). Menu do lado esquerdo da tela clicar no botão “Access & Security”. Selecione a aba “Security Groups” e clique no botão “Create Security Group”. Dê um nome se sua preferência, por exemplo “SSH-ICMP”. Selecione o security group criado e clique no botão “Edit Rules”. Configura de acordo com as figuras 4.20 e 4.21.

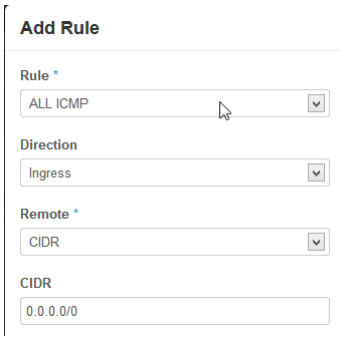

Figura 4.20 - Liberando o protocolo ICMP – OpenStack

[1] http://www.microsoft.com/security/pc-security/firewalls-whatis.aspx
[2] https://help.ubuntu.com/community/SSH/OpenSSH/Keys
[3] http://tools.ietf.org/html/rfc4252
[4] http://tools.ietf.org/html/rfc792

Figura 4.21 - Liberando o protocolo SSH - OpenStack

Passo 3 Requisitar um endereço de IP da rede externa. Menu do lado esquerdo da tela clicar no botão “Access & Security”. Selecione a aba “Floating IPs”, clique no botão “Allocate IP to Project” e selecione a rede externa criada no item 4.3.1. Um endereço IP será alocado a conta em questão. Esse IP pode ser associado a qualquer VM da conta e através dele que conectamos à máquina.

Passo 4 Instanciar a máquina. Menu do lado esquerdo da tela clicar no botão “Instances” e clique no botão “Launch Instance” que fica do lado superior direito da tela. É nesse momento que o usuário pode escolher os recursos computacionais (CPU, memória e armazenamento). As figuras 4.22, 4.23 e 4.24, mostram a requisição de uma máquina virtual.

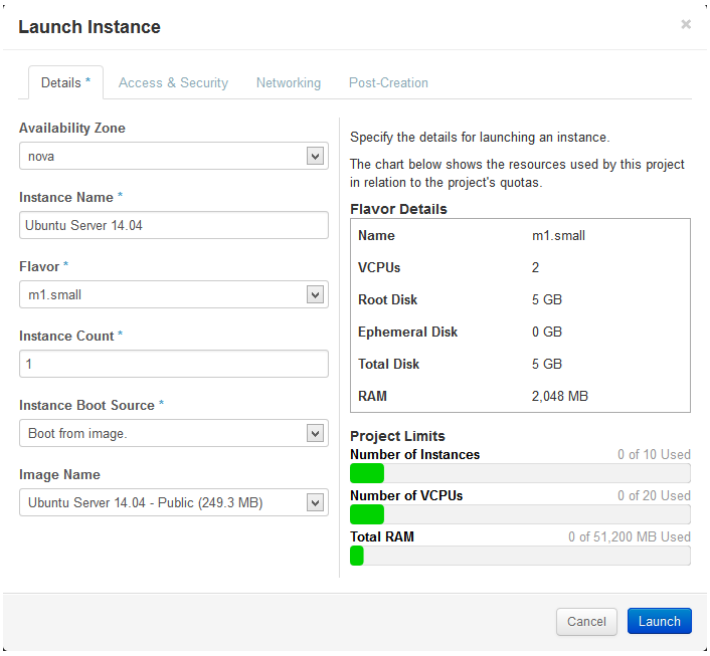

Figura 4.22 - Requisitando uma VM, aba 1 - OpenStack

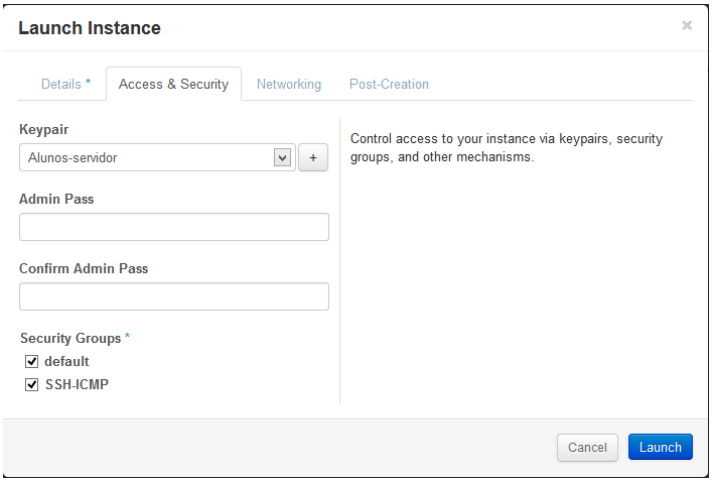

Figura 4.23 - Requisitando uma VM, aba 2 - OpenStack

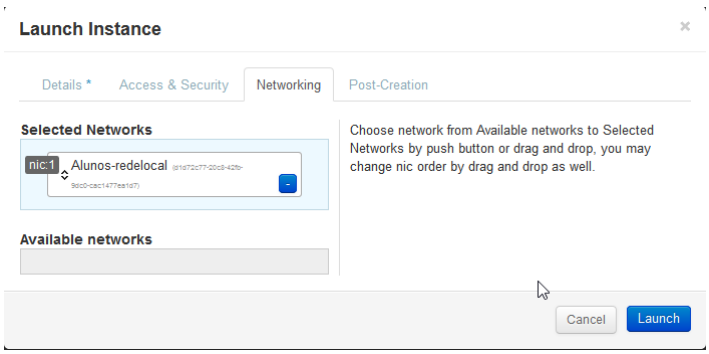

Figura 4.24 - Requisitando uma VM, aba 3 - OpenStack

A figura “Requisitando

uma VM, aba 3”, o usuário associa a VM à uma rede. Na figura 4.25 é possível visualizar graficamente a associação da VM à rede.

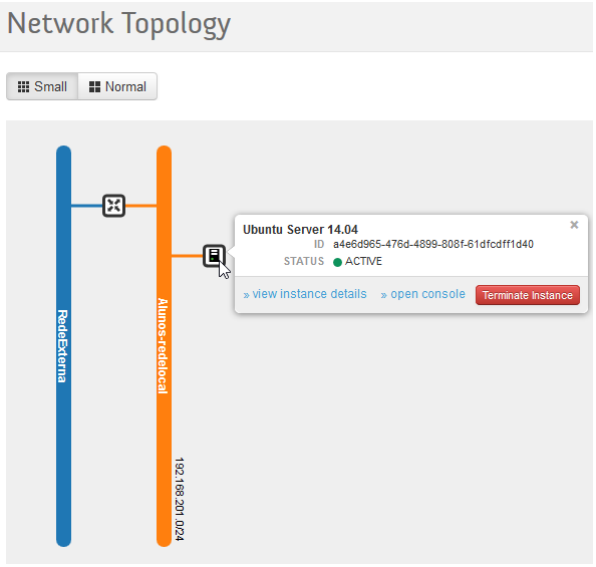

Figura 4.25 VM integrada a rede

Passo 5) Associar o endereço de IP da rede externa na instância. A VM está em execução e não está acessível para a rede externa do IFSC. Selecione a VM, clique no botão “More” e escolha a opção “Associate Floating IP”. ConFigura de acordo com a figura 4,26

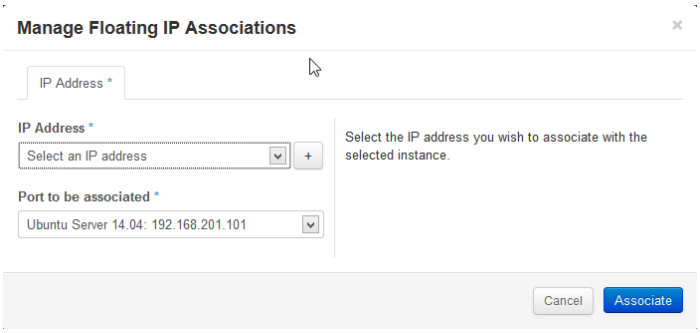

Figura 4.26 - Associando um IP externo na VM - OpenStack

Passo 6) Testar conectividade via ICMP e acessar a máquina virtual via *SSH*. A figura 4.27 mostra que a máquina está acessível.

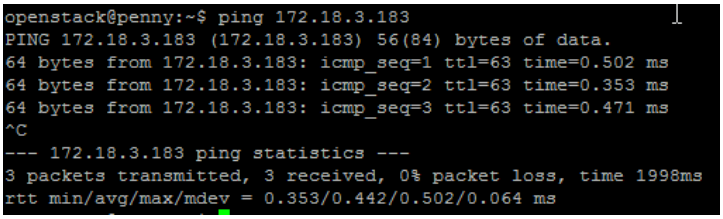

Figura 4.27 - Testando conectividade de rede via ICMP – OpenStack

Foi possível constatar no atual cenário que o tempo médio que o OpenStack levou para o processo de criação de uma máquina virtual (implica configuração de processamento, memória, rede e armazenamento) e disponibiliza-la ao usuário foi de 10 segundos.

4.5 Migração de máquinas virtuais entre VMMs

> A migração de máquinas virtuais (VMs) através de servidores físicos distintos traz diversas vantagens, tais como: (i) balanceamento de carga - as VMs podem ser rearranjadas em nós físicos do datacenter para aliviar a carga sobre hospedeiros saturados, (ii) manutenção online e tolerância a falhas - uma máquina física pode precisar de upgrade ou manutenção do sistema devido a falhas, de forma que um Administrador pode migrar as VMs em execução para uma máquina alternativa, liberando a máquina original para manutenção e, (iii) gerenciamento de energia - quando as VMs podem ser consolidadas em um número reduzido de servidores, reduzindo assim o custo de energia. Tais vantagens permitem aumentar a flexibilidade e reduzir a complexidade de gerenciamento dos recursos físicos em ambientes virtualizados. Entretanto, a migração provoca uma degradação no desempenho do sistema, uma vez que para realizá-la, é necessário parar a execução da VM, provocando sua indisponibilidade e, por conseguinte, das aplicações que nela executam. (GOMES, MAGALHÃES & SOARES, 2012, p. 236)

Segundo RedHat (2014), existe dois tipos de migração:

Migração em tempo real (*live migration* ou a quente) - a VM continua sendo executada no VMM de origem, enquanto as suas páginas de memória são transferidas ao VMM de destino. Durante a migração a VM é suspendida, a diferença das páginas de memória entre as VMM é calculada e transferida para a VMM de destino e a VM é

iniciada novamente. Esse tipo de migração requer armazenamento compartilhado, como iSCSI, NFS[1], FCoE[2].

Migração a frio – a VM é suspendida, uma imagem da memória da VM é transferida para o VMM de destino e então a VM é iniciada no novo VMM. Nesse caso o tempo de migração depende do o tamanho do disco da VM e da performance da rede para transferir o disco entre VMMs.

De acordo com a atual configuração do Nova deste trabalho, esperasse que a migração de máquinas virtuais ocorra através do método migração em tempo real. A migração ocorre quando o administrador da nuvem, por algum motivo, precisa migrar VMs que estão em um determinado VMM ou o usuário solicita a mudança do tamanho da VM. As duas situações podem ser executadas através da Dashboard.

Passo 1 – Instanciar uma VM, ver item 4.4.

Passo 2 – Através da Dashboard, o administrator consegue visualizar em qual VMM a VM está sendo executada, como mostra a figura 4.28.

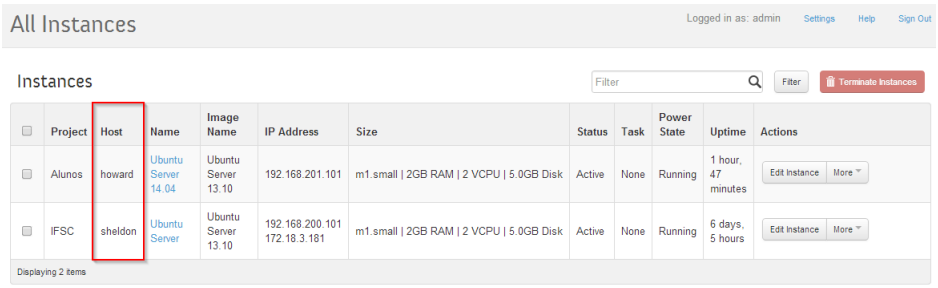

Figura 4.28 - Lista de VM em execução e seus respectivos VMM. - OpenStack

Passo 3 – Usuário muda o tamanho da instância ou o administrador executa a migração. Nas figuras 4.29 e 4.30, mostra como é feito a requisição respectivamente as situações.

[1] http://tools.ietf.org/html/rfc3530
[2] http://www.t11.org/ftp/t11/pub/fc/bb-5/09-056v5.pdf

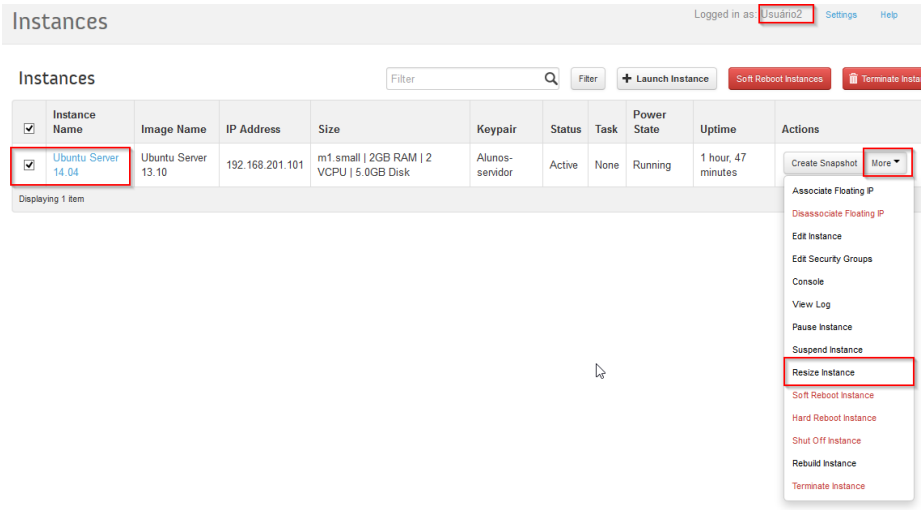

Figura 4.29 - Usuário requisitando a mudança do tamanho da instância - OpenStack

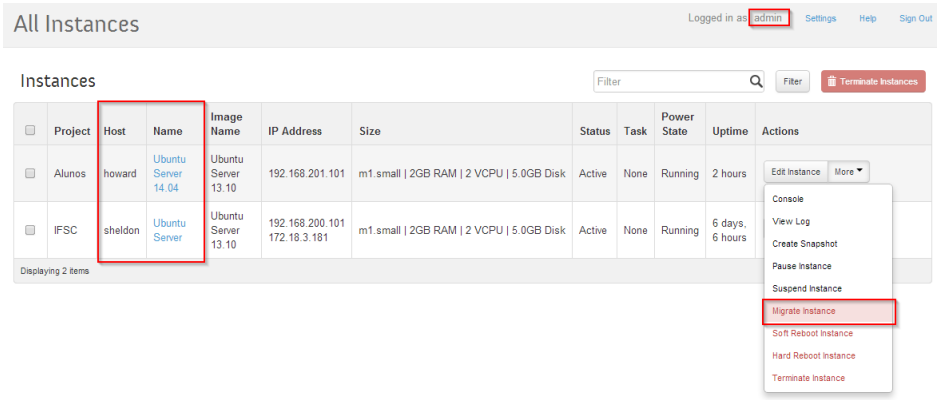

Figura 4.30 - Administrador requisitando a migração da VM – OpenStack

Passo 4 – A figura Execução da migração, ilustra OpenStack executando a migração. Após a migração existe as opções de reverter ou confirmar a migração, conforme mostra a figura 4.31.

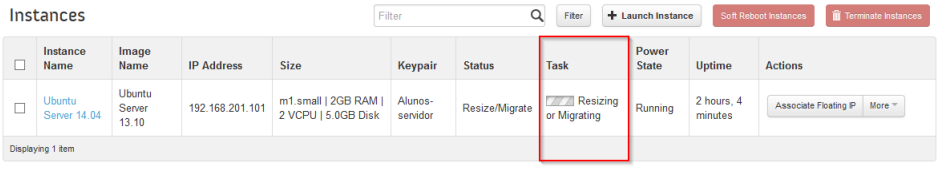

Figura 4.31 - Execução da migração – OpenStack

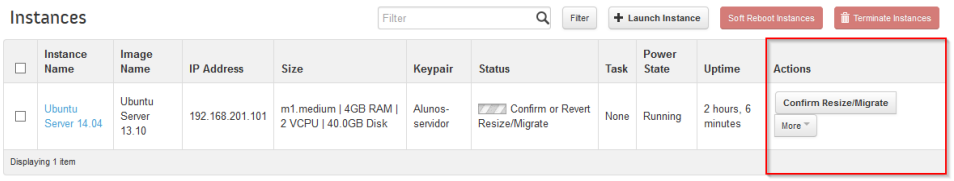

Figura 4.32 - Confirmação ou reversão da migração – OpenStack

Passo 5 – Verificar a mudança de VMM. A figura 4.33, mostra o antes e depois da migração.

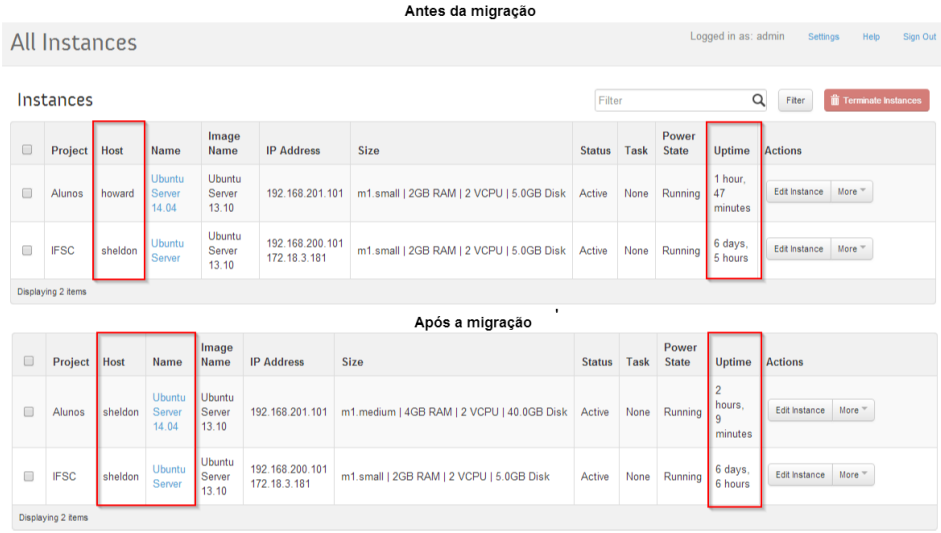

Figura 4.33 - Comparação da alocação do VMMs antes e epois da migração. - OpenStack

No atual cenário, foi possível constatar que ocorreu a perda de conectividade da máquina virtual durante o processo de migração entre VMMs. Entretanto a VM manteve seu estado inalterado após a migração, ou seja, os processos que estavam rodando e suas respectivas alocação de memória no sistema, as suas configurações de rede e de armazenamento continuaram exatamente igual. Na figura 4.34 é possível verificar através do comando ping essa perda de conectividade. Foi possível verificar que o tempo médio de migração foi de 8 segundos.

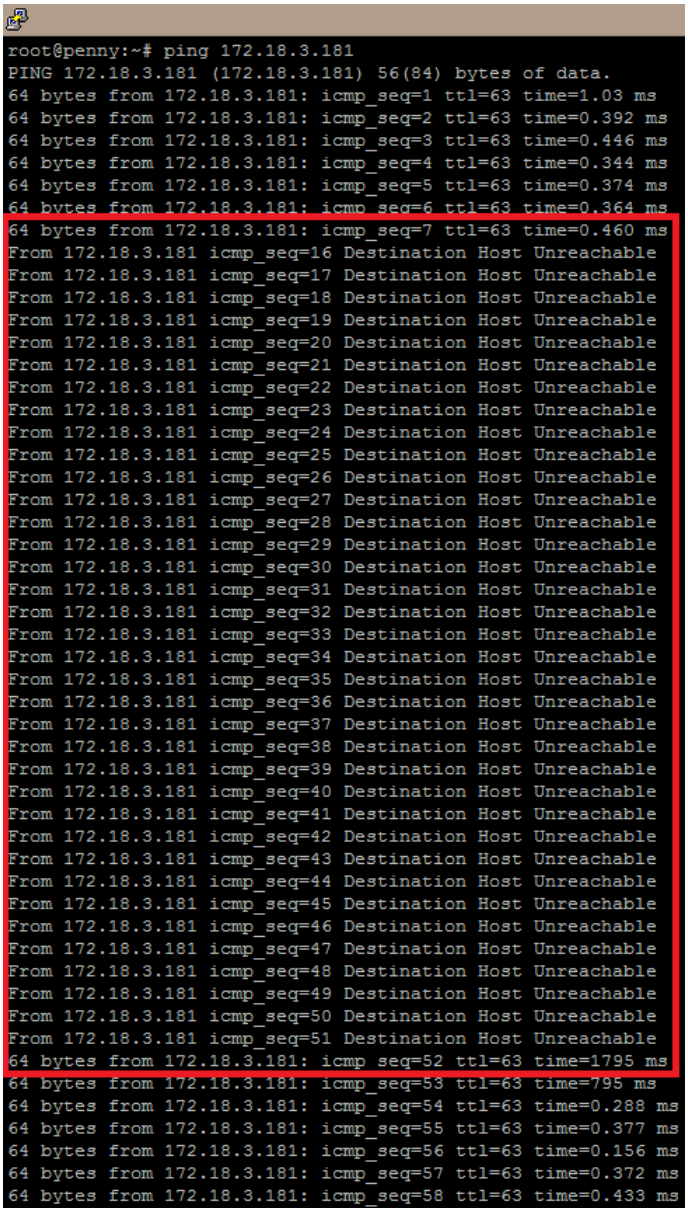

Figura 4.44 - Testando a conectividade de uma VM durante a migração

O resultado alcançado sobre migração de máquinas virtuais no presente momento possui as características de uma migração a frio. Ou seja, as VMs não estavam utilizando os discos virtuais oferecido pelo módulo Cinder do OpenStack através do protocolo iSCSI. O que é característica essencial para a utilização do método *live migration*.

Uma averiguação foi efetuada para descobrir o motivo das migrações não estarem sendo feitas pelo método migração em tempo real. Foi identificado que o módulo iSCSI no GNU/LINUX não foi configurado corretamente, impossibilitando a

utilização dos discos virtuais pelas VMs através da rede. Por questão de tempo não foi possível solucionar o problema e consequentemente não foi possível obter resultados com migração do tipo tempo real. Onde a expectativa do tempo de indisponibilidade da VM durante a migração é menor do que a migração a frio.

5 Conclusões

OpenStack mostrou-se ser uma plataforma completa, por conseguir entregar de uma forma simples e abstraída, do ponto de vista do usuário, os serviços de armazenamento, computação e redes. A excelência em gerenciar de forma centralizada os recursos computacionais, ainda mais de uma forma automatizada.

Por envolver diversos tipos de conceitos e tecnologias, entender o OpenStack por completo foi uma tarefa árdua. O entendimento sobre os módulos do OpenStack, as interações entre si e suas funções foi um processo longo. Conseguir pela primeira vez um ambiente com o mínimo de funcionalidades foi oneroso. Entretanto, uma vez que arquivos de configuração estão devidamente configurados, a implementação do ambiente se torna uma tarefa simples apesar da complexidade envolvida. O resultado dessa curva de aprendizado possibilitou a entrega do ambiente de computação em nuvem para o IFSC e a documentação do ambiente proposto, abordando virtualização, computação em nuvem e redes SDN.

O módulo de serviço de rede do OpenStack contribui positivamente para as direções almejadas neste contexto de redes de computadores. A integração do OpenStack com Open vSwitch possibilitou a abstração da rede de computadores e a disponibilizou como serviço (*network as a service*) as VMs, e aos usuários um maior controle e dinamismo sobre a mesma. Almejando com sucesso uma nova proposta de arquitetura de redes de computares se comparado as redes tradicionais.

Apesar dos resultados alcançados, devido ao tempo não foi possível obter um cenário funcional com migração de máquinas virtuais em tempo real. Ficando como uma proposta os próximos trabalhos.

5.1 Trabalhos futuros

Proponho para possíveis trabalhos futuros:

- Aprofundamento na flexibilização na rede, que possibilite que máquinas virtuais possam ser migradas através de redes diferentes. Por exemplo, máquinas virtuais que estão IFSC do continente possam ser migradas em

tempo real para o IFSC de São José. Tentando ao máximo diminuir o tempo de indisponibilidade da máquina virtual.

- O melhor aproveitamento dos recursos computacionais do IFSC através de uma plataforma de computação em nuvem do tipo IaaS. Por exemplo, disponibilizar os recursos computacionais dos laboratórios quando ociosos aos professores e alunos.

Lista de Abreviaturas

AMPQ *Advanced Message Queuing Protocol*

API Application Programming Interface

APT Advanced Packaging Tool

DHCP *Dynamic Host Configuration Protocol*

GNU *General Public License*

IaaS *Infrastructure as a Service*

IFSC Instituto Federal de Santa Catarina

IFSC-SJ Instituto Federal de Santa Catarina de Sao José

IP *Internet Protocol*

IPFIX *Internet Protocol Flow Information Export*

LACP *Link Aggregation Control Protocol*

LDAP *Lightweight Directory Access Protocol*

LVM *Logical Volume Manager*

NAT *Network Address Translation*

NAS *Network Attached Storage*

NIST *National Institute of Standards and Technology*

NTP *Network Time Protocol*

ONF *Open Networking Foundation*

OVSDB *Open vSwitch Database Management Protocol*

PaaS *Platform as a Service*

QoS *Quality of service*

RSPAN *Remote Switch Port Analyzer*

SaaS *Software as a Service*

SDN *Software defined network*

SSH *Secure Shell*

STP *Spanning Tree Protocol*

URL *Uniform Resource Locators*

VLAN *Virtual Local Area Network*

vNIC *Virtual Network Interface Card*

VMM *Virtual Machine Monitor*

Referências Bibliográficas

IEEE STANDARDS ASSOCIATION. *Get IEEE 802*. 2011. http://standards.ieee.org/getieee802/download/802.1Q-2011.pdf. Último acesso em 15 de Fevereiro de 2014.

GIBILISCO, Stan. *Software-defined networking (SDN)*. 2012. http://searchsdn.techtarget.com/definition/software-defined-networking-SDN. Último acesso em 17 de Fevereiro de 2014.

GOMES, Danielo G. MAGALHÃES, Deborah V. SOARES, José Marques. Análise do Impacto de Migração de Máquinas Virtuais em

Ambiente Computacional Virtualizado. In: XXIX Simpósio Brasileiro de Redes de Computadores e Sistemas Distribuídos. Anais. 235-248. 2012.

GREENBERG et al. 2008. *The Cost of a Cloud: Research Problems in Data Center Networks.* ACM SIGCOMM, 2008. Disponível em http://www.cs.rutgers.edu/~badri/552dir/papers/intro/p68-v39n1o-greenberg.pdf.

GROSS, Jesse. *Programmable Networking with Open vSwitch*. In: LINUXCON, 2013. http://events.linuxfoundation.org/sites/events/files/slides/OVS-LinuxCon%202013.pdf. Último acesso em 16 de Fevereiro de 2014.

HALEPLIDIS, E. et al. *SDN Layers and Architecture Terminology*. 2013. http://tools.ietf.org/html/draft-haleplidis-sdnrg-layer-terminology-02. Último acesso em 15 de Janeiro de 2014.

JONES, M. *Virtual networking in Linux*. 2010. http://www.ibm.com/developerworks/library/l-virtual-networking/. Último acesso em 14 de Fevereiro de 2014.

KIM e FEAMSTER, 2013. *Improving Network Management with Software Defined Networking*. IEEE Magazine, 2013. Disponível em: http://www.cc.gatech.edu/grads/h/hkim368/publication/SDN_ieeemagazine_Kim.pdf

NIST, National Institute of Standards and Technology. *Special Publication 500-291*. 2011. http://www.nist.gov/itl/cloud/upload/NIST_SP-500-291_Jul5A.pdf. Último acesso em 19 de Fevereiro de 2014.

ONF, Open Networking Foundation. *Software-Defined Networking (SDN) Definition*. 2013. https://www.opennetworking.org/index.php?option=com_content&view=article&id=686&Itemid=272&lang=en. Último acesso em 14 de janeiro de 2014.

ONF, Open Networking Foundation. *White Papers*. 2013. https://www.opennetworking.org/index.php?option=com_content&view=category&layout=blog&id=46&Itemid=158&lang=en. Último acesso em 16 de janeiro de 2014.

ONF, Open Networking Foundation. *FAQ's*. 2013. https://www.opennetworking.org/index.php?option=com_content&view=category&layout=blog&id=12&Itemid=126&lang=en. Último acesso em 14 de janeiro de 2014.

ORACLE, 2009. Paravirtualização e outros tipos de Virtualização. Disponível em: https://blogs.oracle.com/galdino/entry/paravirtualizacao_e_outros_tip

OPENSTACK. *OpenStack Installation Guide for Ubuntu 12.04 (LTS)*. 2012. http://docs.openstack.org/havana/install-guide/install/apt/content/index.html. Último acesso em 15 de Fevereiro de 2014.

OPENSTACK. *Chapter 24. Networking Architecture*. 2012. http://docs.openstack.org/security-guide/content/ch031_neutron-architecture.html. Último acesso em 15 de Fevereiro de 2014.

OPENSTACK. *OpenStack Compute*. 2013. http://www.openstack.org/software/openstack-compute/. Último acesso em 15 de Fevereiro de 2014.

OPENSTACK. *The Open Source Cloud Operating System*. 2014. http://www.openstack.org/software/. Último acesso em 15 de Fevereiro de 2014.

OPENSTACK. *OpenStack Dashboard*. 2014. http://www.openstack.org/software/openstack-dashboard/. Último acesso em 15 de Fevereiro de 2014.

OPENSTACK. *OpenStack Shared Services*. 2014. http://www.openstack.org/software/openstack-shared-services/. Último acesso em 15 de Fevereiro de 2014.

OPENSTACK. *OpenStack Storage*. 2014. http://www.openstack.org/software/openstack-storage/. Último acesso em 15 de Fevereiro de 2014.

OPENVSWITCH. *FAQ's*. 2014. http://git.openvswitch.org/cgi-bin/gitweb.cgi?p=openvswitch;a=blob_plain;f=FAQ;hb=HEAD. Último acesso em 14 de Fevereiro de 2014.

OPENVSWITCH. *What is Open vSwitch*. 2014. http://git.openvswitch.org/cgi-bin/gitweb.cgi?p=openvswitch;a=blob_plain;f=README;hb=HEAD. Último acesso em 14 de Fevereiro de 2014.

OPENVSWITCH. *Why Open vSwitch?.* 2014. http://git.openvswitch.org/cgi-bin/gitweb.cgi?p=openvswitch;a=blob_plain;f=WHY-OVS;hb=HEAD. Último acesso em 14 de Fevereiro de 2014.

POPEK e GOLDBERG, 1974. *Formal Requirements for Virtualizable Third Generation Architectures*, 1974. Disponível em: http://www.logos.ic.i.u-tokyo.ac.jp/~tau/lecture/os/gen/papers/p412-popek.pdf

REDHAT. *KVM live migration*. 2014. https://access.redhat.com/site/documentation/en-US/Red_Hat_Enterprise_Linux/6/html/Virtualization_Administration_Guide/chap-Virtualization_Administration_Guide-KVM_live_migration.html. Último acesso em 10 de Março de 2014.

RHOTON, John. *Discover OpenStack: Architectures, functions, and interactions*. 2013. http://www.ibm.com/developerworks/cloud/library/cl-openstack-overview/index.html. Último acesso em 16 de Fevereiro de 2014.

RHOTON, John. *Discover OpenStack: The Compute components Glance and Nova*. 2013. http://www.ibm.com/developerworks/cloud/library/cl-openstack-nova-glance/index.html. Último acesso em 16 de Fevereiro de 2014.

SDN Central. *What is OpenFlow*?. 2013. http://www.sdncentral.com/what-is-openflow/. Último acesso em 12 de Fevereiro de 2014.

SCHUBERT, 2011 apud GRIMES, JAEGER e LIN 2008. *Aplicação de Algoritmos de Provisionamento Baseados em Contratos de Nível de Serviço para Computação em Nuvem.* SBRC 2011 – WCGA 2011. Campo Grande, 2011.

VMWARE White Paper, 2010. Understanding Full Virtualization, Paravirtualization, andHardware Assist. VMWARE Corporation. 2010. Disponível em: http://www.vmware.com/files/pdf/VMware_paravirtualization.pdf

ZEILENGA,K. *Lightweight Directory Access Protocol* (*LDAP*). 2006. https://tools.ietf.org/html/rfc4510. Último acesso em 15 de Fevereiro de 2014.