



Universidade Federal de Uberlândia
Faculdade de Engenharia Elétrica



**UTILIZAÇÃO DO *OpenStack* EM CONJUNTO
COM AS TECNOLOGIAS *SDN* -
Software-Defined Networking E *NFV* -
*Network Functions Virtualization***

LUCAS CARVALHO MIRANDA

Uberlândia
2017

LUCAS CARVALHO MIRANDA

**UTILIZAÇÃO DO *OpenStack* EM CONJUNTO
COM AS TECNOLOGIAS *SDN* -
Software-Defined Networking E *NFV* -
*Network Functions Virtualization***

Monografia apresentada como requisito de avaliação na disciplina Trabalho de Conclusão de Curso 2 do Curso de Engenharia Elétrica da Universidade Federal de Uberlândia.

Orientador: Prof. Me. Luiz Cláudio Theodoro

Uberlândia
2017

*Este trabalho é dedicado à minha família e amigos que,
quando eu mesmo duvidei, acreditaram em mim.*

Agradecimentos

Agradeço primeiramente a Deus que nos possibilita todas as realizações.

À minha família, por todos os valores, suporte e por sempre acreditarem na minha capacidade e por todo o esforço que fizeram para que pudesse chegar onde estou hoje.

Sou grato aos meus amigos, por todo o incentivo. Em especial ao Caio por todas as conversas acerca do tema, ao Marcelo Seigo por todo o apoio e por último, mas não menos importante, a minha amiga e companheira Rafaela, por todo o amor, compreensão e suporte.

Por fim, sou imensamente grato ao meu professor e orientador Luiz Cláudio Theodoro, não só pela paciência e orientação deste trabalho, mas também por todo o apoio durante o curso, pela orientação de estágio, por ter me auxiliado no caminho para o mercado de trabalho e por todo conselho e acompanhamento na minha formação profissional.

Meus sinceros agradecimentos a todos.

“Quem acredita sempre alcança.”
(Renato Russo)

Resumo

A computação em nuvem é uma tecnologia que tem tido uma excelente aceitação e sua adoção é cada vez maior, tudo isto, alavancado pela otimização dos recursos da computação e *hardware* sob demanda através da *internet*. O *OpenStack* surgiu como solução para a computação em nuvem de código aberto e infraestrutura como serviço, permitindo orquestração, gestão e automação dos recursos. Nessa linha, surgiram as tecnologias *Software-Defined Networking* (SDN) e *Networking Functions Virtualization* (NFV) que se baseiam na virtualização de *software*, onde SDN é uma arquitetura emergente com a função de separar o plano de controle e plano de dados em *hardware* de rede, e NFV transfere as funções de rede do *hardware* para dispositivos genéricos, concentrando as diferentes funções em um único *appliance* versátil e expansível. Sendo assim, o presente trabalho tem como objetivo apresentar a plataforma *OpenStack*, os conceitos de SDN e NFV e discutir sobre a utilização destas tecnologias de forma conjunta. O estudo tem como metodologia conceituar computação em nuvem, virtualização, *OpenStack*, SDN e NFV, por meio de uma pesquisa bibliográfica a fim de aprofundar e atualizar os conhecimentos sobre os temas para sequencial discussão e abstrair a experiência obtida por pesquisadores locais e distantes. As tecnologias SDN e NFV deverão se tornar tecnologias padrões para implementações futuras de rede. Como é necessário um ambiente de computação em nuvem completo para essas aplicações, o *OpenStack* se torna uma solução viável mesmo porque já é uma realidade nos ambientes TIC de grandes empresas.

Palavras-chave: Computação em nuvem. Virtualização. *OpenStack*. SDN. NFV

Abstract

Cloud computing is a technology that has had great acceptance and its adoption is increasing, all this, leveraged by the optimization of the resources of computing and hardware on demand through the internet. OpenStack has emerged as a solution for open source cloud computing and infrastructure as a service, enabling orchestration, management and automation of resources. In this line, the emergence of Software-Defined Networking (SDN) and Networking Functions Virtualization (NFV) technologies that are based on software virtualization, where SDN is an emerging architecture with the function of separating the control plane and data plane into network hardware, and NFV transfers the network functions from the hardware to generic devices, concentrating the different functions into a single, versatile and expandable appliance. Thus, the present work aims to present the OpenStack platform, the SDN and NFV concepts and discuss the use of these technologies together. The study has as a methodology to conceptualize cloud computing, virtualization, OpenStack, SDN and NFV, through a bibliographical research in order to deepen and update the knowledge on the themes for sequential discussion and to abstract the experience obtained by local and distant researchers. SDN and NFV technologies are expected to become standard technologies for future network deployments. As a complete cloud computing environment is required for these applications, OpenStack becomes a viable solution even because it is already a reality in large enterprise IT environments.

Keywords: Cloud computing. Virtualization. OpenStack. SDN. NFV.

Lista de figuras

Figura 1 – Computação em nuvem	14
Figura 2 – Camadas dos modelos de serviço	16
Figura 3 – Nuvem privada	17
Figura 4 – Nuvem pública	17
Figura 5 – Nuvem comunitária	17
Figura 6 – Nuvem híbrida	18
Figura 7 – <i>Openstack Cloud Operating System</i>	20
Figura 8 – Projetos <i>OpenStack</i>	21
Figura 9 – Arquitetura lógica de nuvem com <i>OpenStack</i>	22
Figura 10 – Arquitetura conceitual do <i>Neutron</i>	25
Figura 11 – Arquitetura conceitual do <i>SDN</i>	31
Figura 12 – Arquitetura de alto nível do <i>NFV</i>	32
Figura 13 – Exemplo de serviço com <i>VNF</i>	33

Lista de abreviaturas e siglas

API	Application Program Interface
AMQP	Advanced Message Queuing Protocol
BSS	Business Support System
CAPEX	Capital Expenditure
CPE	Customer-Premises Equipment
CPU	Central Processing Unit
DHCP	Dynamic Host Configuration Protocol
ETSI	European Telecommunications Standards Institute
FISL	Fórum Internacional de Software Livre
HP	Hewlett-Packard
I2RS	Interface to the Routing System
IaaS	Infrastructure as a Service
IoT	Internet of Things
IP	Internet Protocol
LAN	Local Area Network
LVM	Logical volume manager
M2M	Machine to Machine
MAC	Media Access Control
MMV	Monitor de Máquina Virtual
NFV	Network Functions Virtualization
NFVI	NFV Infrastructure
NTIC	Novas Tecnologias da Informação e Comunicação
NIST	National Institute of Standards and Technology
OPEX	Operational Expenditure

OSS	Operation Support System
PaaS	Plataform as a Service
PoPs	Points of Presence
QoS	Quality Of Service
SaaS	Software as a Service
SDK	Software Development Kit
SDN	Software-Defined Networking
SDNP	Software Driven Networking Protocol
SNMP	Simple Network Management Protocol
SSH	Secure Shell
TI	Tecnologia da Informação
TIC	Tecnologias da Informação e Comunicação
VFN	Virtual Network Functions
VLANs	Virtual Local Area Networks
VMM	Virtual Machine Monitor
VMs	Virtual Machines
VNF	Virtual Network Function
VPN	Virtual Private Network

Sumário

1	INTRODUÇÃO	11
2	METODOLOGIA	13
3	REVISÃO BIBLIOGRÁFICA	14
3.1	Computação em nuvem	14
3.2	Virtualização	18
3.3	<i>OpenStack</i>	20
3.4	<i>Software-Defined Networking - SDN</i>	28
3.5	Network Functions Virtualization - NFV	31
4	DISCUSSÃO	36
5	CONCLUSÃO	39
	Referências	40

1 Introdução

Nos últimos anos a computação em nuvem tem ganhado crescente espaço e vem se mostrando cada vez mais promissora, pois é uma ferramenta com a possibilidade de se pagar apenas pelos recursos utilizados pelo sistema (DAMASCENO et al., 2016).

A computação em nuvem atende a necessidade de construir complexas infraestruturas de tecnologia da informação, que tem o papel central e é objeto crítico para assegurar a continuidade de operação em várias organizações, oferecendo uma forma de utilização otimizada dos recursos de computação e *hardware* sob demanda através da *internet*, além das características essenciais para isso, como elasticidade, disponibilidade, compartilhamento de recursos, medição e monitoramento (ROSADO; BERNARDINO, 2014), apresentados para usuários finais sob a forma de serviço, sem a necessidade de ter conhecimento sobre a tecnologia utilizada e sem se preocupar com os problemas de alocação de espaço para *hardware* e até mesmo a manutenção de máquinas e equipamentos.

No campo da tecnologia da informação (TI), a computação em nuvem se tornou uma ferramenta essencial no desenvolvimento de tecnologias e aplicações permitindo a redução de custos, melhora no desempenho e uma melhor utilização de recursos. Como ferramenta para implementar melhorias na computação em nuvem foi se criado o *Openstack* (AMORIM et al., 2015).

OpenStack foi iniciado pela *National Aeronautics and Space Administration (NASA)* e *Rackspace Hosting* em julho de 2010, sendo um sistema operacional de nuvem de código aberto que proporciona a criação de nuvens privadas e públicas. Não se trata de um *software* monolítico, ou seja, você consegue utilizar apenas as ferramentas que precisar. Alguns dos módulos do *OpenStack* são: *Nova*, *Swift*, *Cinder*, *Glance*, *Keystone*, *Neutron*, *Horizon*, *Heat*, *Ceilometer*, *Trove*, *Sahara* (MARTINS; BERNARDES; BOAVIDA, 2014).

A utilização do termo “*Open*” se refere ao projeto não depender de fornecedores e de nenhuma grande empresa, sendo um programa de código aberto onde qualquer desenvolvedor possa utilizar e modificar a sua vontade, utilizável, escalável e sem limitações de recursos ou desempenho, distribuído sob a licença *Apache 2.0* (OPENSTACK, 2014). Existe uma comunidade independente que trabalha para otimizar o programa, e duas vezes ao ano uma versão atualizada é lançada. Pelo fato de possuir padrões abertos, essa tecnologia foi adotada por diversas empresas, sendo alguma delas, *Hewlett-Packard (HP)*, Mercado Livre, *Oracle*, *Dell HubSpot* e UOL (UOL, 2016).

Para continuar com os avanços em tecnologias de rede sem que ocorra interrupções na rede, a comunidade de pesquisa e desenvolvimento de redes de computadores continua investindo e desenvolvendo novas soluções que adicione mais recursos e interfaces de programação aos

componentes de rede. O *Software-Defined networking (SDN)* vem sendo uma dessas soluções. O *SDN* é um sistema de controle que permite ao *software* desenvolvido definir e alterar a comutação das redes por meio de uma interface de programação bem definida. Os controladores de *SDN* são considerados como sistemas operacionais de redes de computadores, sendo paralelos entre as aplicações que serão desenvolvidas para um controle de rede e o *hardware* (VERT, 2017).

Além disso, as empresas de telecomunicações sofrem com a sua dependência de *hardware* proprietário para o provisionamento de serviços, o custo para se ter um serviço deste porte é muito elevado, dificultando assim o lançamento de novos serviços. Para contornar essa adversidade surgiu a Virtualização de Funções de Rede (*NFV – Network Functions Virtualization*), viabilizando os serviços de telecomunicações de equipamentos proprietários em equipamentos genéricos que poderiam ser instalados em *datacenters* ou *Points of Presence (PoPs)*. O serviço *NFV* consiste em um conjunto de funções denominadas funções de Rede Virtuais (*VNF - Virtual Network Functions*), que implementam *software* em um ou mais servidores físicos genéricos (BEDOR et al., 2015).

O presente trabalho tem como objetivos apresentar a plataforma *OpenStack*, os conceitos de *Software-Defined Networking (SDN)*, *Networking Functions Virtualization (NFV)* e apresentar as possibilidades destas tecnologias interagirem.

2 Metodologia

Para atingir o objetivo estabelecido para o trabalho foram seguidas etapas. Para conceituar computação em nuvem, virtualização, a plataforma Openstack, e as tecnologias *software-defined networking* e *network functions virtualization*, foi realizada uma revisão bibliográfica visando aprofundar e atualizar os conhecimentos descritos na literatura. Em seguida, foi proposto a utilização dessas tecnologias em conjunto, sabendo que se confirmados os benefícios e estabelecidos os riscos, essa prática pode ser uma nova tendência que irá agregar para a evolução da computação em nuvem.

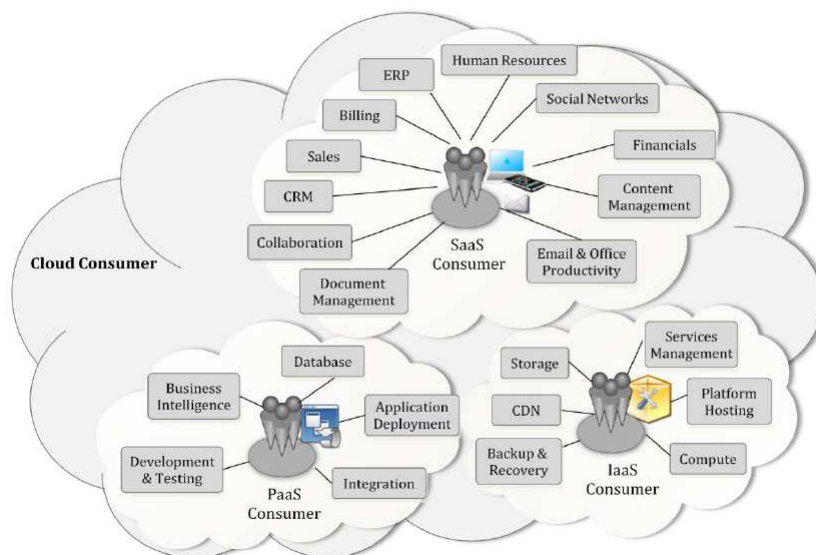
Para desenvolver a proposta o trabalho foi estruturado em três partes: a introdução, com objetivo de contextualizar o leitor e apresentar o objetivo do estudo; O referencial teórico, a fim de aprofundar nos conceitos necessários para compreender a problemática e abstrair a experiência de pesquisadores locais e distantes, durante a exposição também foram explicitados seus possíveis usos e, por fim a discussão, onde é apresentada e descrita a proposta de objetivo do estudo, destacando assim a relevância do mesmo.

3 Revisão Bibliográfica

3.1 Computação em nuvem

A computação em nuvem (*cloud computing*) é a entrega da computação em forma de serviço, onde ocorrem diversos compartilhamentos de recursos, *software*, rede, banco de dados, servidores, dentre outros. A computação em nuvem permite ao usuário acessar quaisquer desses recursos através de computadores, celulares ou *tablets*, conectados à *internet* (TECHTUDO, 2012).

Figura 1 – Computação em nuvem



Fonte: (LIU et al., 2011, p. 13).

Apesar da popularização relativamente recente, o conceito surgiu na década de 1960 a partir de idéias de pioneiros como John McCarthy, que definia a computação como utilidade pública, e Joseph Carl Robnett Licklider, com a idealização da computação na forma de uma rede global, porém o termo "computação em nuvem" foi utilizado pela primeira vez por Ramnath Chellappa, em 1997 (FORBES, 2011).

Porém a definição amplamente aceita de computação em nuvem foi estabelecida pelo *National Institute of Standards and Technology (NIST)*, em 2011.

A computação em nuvem é um modelo para habilitar o acesso por rede ubíquo, conveniente e sob demanda a um conjunto compartilhado de recursos de computação (como redes, servidores, armazenamento, aplicações e serviços) que possam ser rapidamente provisionados e liberados com o mínimo de esforço de gerenciamento ou interação com o provedor de serviços (MELL; GRANCE, 2011, p. 6).

De acordo com o *NIST* (MELL; GRANCE, 2011) as principais características da computação em nuvem são:

1. Autoatendimento sob demanda (*on-demand self-service*): são as funcionalidades computacionais, por exemplo, tempo de servidor, armazenamento, rede, fornecidas automaticamente conforme a demanda do cliente. O preço desse serviço é proporcional a quantidade de recursos solicitados;
2. Amplo acesso à rede (*broad network access*): esses recursos são disponíveis através da *internet*, sendo acessados por qualquer dispositivo conectado à rede (computadores, *tablets*, celulares ou estações de trabalho);
3. *Pool* de recursos (*resource pooling*): são recursos computacionais do provedor reunidos para atender inúmeros consumidores utilizando do modelo multilocatário, que apresenta diferentes recursos físicos e virtuais, armazenando e distribuindo dinamicamente de acordo com a vontade do usuário;
4. Elasticidade rápida (*rapid elasticity*): é a capacidade de uma computação em nuvem em disponibilizar mais recursos à medida que é solicitado pelo usuário;
5. Serviços mensuráveis (*measured services*): a capacidade de gerenciar a computação em nuvem automaticamente, controlando e otimizando os recursos utilizados pelo cliente, além de possibilitar a medição do serviço utilizado, sendo muito importante, pois o serviço é baseado no sistema *pay-per-usage* (SOARES; SILVA, 2015).

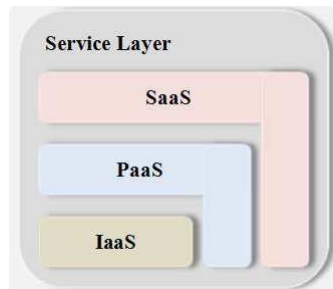
O *NIST* (MELL; GRANCE, 2011) demonstra três modelos de serviço de entrega da computação em nuvem, que constituem seu padrão arquitetural:

- *Software* como serviço (*Software as a service - SaaS*): Especificamente nesse modelo, o usuário tem a possibilidade de executar os aplicativos através de múltiplos dispositivos em uma infraestrutura de nuvem (MELL; GRANCE, 2011). A grande maioria dos recursos utilizados nesse modelo pelo usuário não estão sendo armazenadas ou processadas no terminal, mas sim em nuvem, isso faz com que gere uma facilidade na manutenção, gerenciamento e suporte do sistema e aplicativos (AMORIM et al., 2015). É o modelo mais difundido de entrega da computação em nuvem, e apesar de não ter conhecimento disso a maior parte da população utiliza serviços disponibilizados como *SaaS*, por exemplo, serviços de e-mail como *Gmail* e *Outlook*, redes sociais como *Facebook*, *Twitter* e *Instagram*, ou serviços de armazenamento e compartilhamento em nuvem como *Dropbox*, *Google Drive* ou *Onedrive* (CLOUDMARKET, 2014);
- Plataforma como serviço (*Platform as a service - PaaS*): esse modelo proporciona ao usuário implementar sobre a infraestrutura em nuvem aplicações desenvolvidas ou adquiridas

pelo usuário, utilizando linguagem de programação, bibliotecas, serviços e ferramentas suportados pelo provedor (MELL; GRANCE, 2011). Inicialmente a *PaaS* foi produzida com o objetivo de solucionar os problemas de desenvolver aplicativos para nuvem pública, que era em grande parte uma derivação do *SaaS*, em seguida foi se expandindo para atingir nuvens privadas e híbridas (AMORIM et al., 2015). Alguns exemplos conhecidos de *PaaS* são o *Microsoft Azure*, *OpenShift* e o *Google App Engine* (CLOUDMARKET, 2014);

- **Infraestrutura como serviço (*Infrastructure as a service - IaaS*):** É o modelo que mais se aproxima de um *hardware*/recursos físicos. esse serviço de maneira geral provém de uma API de gerenciamento para a aplicação de um conjunto de recursos físicos, de tal modo que o usuário tenha acesso às funcionalidades como escalabilidade sob demanda, armazenamento e configurações automatizadas do sistema operacional (DAMASCENO et al., 2016). O que difere esse modelo dos demais citados anteriormente, é que esse é responsável por gerenciar aplicações de dados, *middleware*, tempo de execução e armazenamento. Algumas empresas que fornecem esse tipo de serviço são a *Amazon*, *Google* e a *Microsoft* (AMORIM et al., 2015). Alguns provedores de *IaaS* conhecidos no mercado são a *Rackspace*, *Amazon Web Services*, *Google Cloud Platform* e *IBM Cloud* (CLOUDMARKET, 2014).

Figura 2 – Camadas dos modelos de serviço

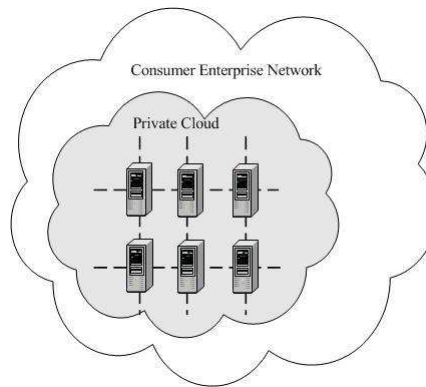


Fonte: (LIU et al., 2011, p. 20).

Outra maneira de classificar a computação em nuvem é baseada na forma com que os ambientes de nuvens são implantados (nuvem privada, nuvem pública, nuvem comunitária e nuvem híbrida) (MELL; GRANCE, 2011).

- **Nuvem Privada:** A nuvem privada oferece a infraestrutura de nuvem de uso exclusivo ao cliente que contrata o serviço, podendo ser administrada e operada pelo contratante ou por algum terceiro. Dentro da nuvem privada existem duas subdivisões: as nuvens privadas externas e as locais. O modelo de nuvem privada local, permite que o cliente tenha total controle sobre a infraestruturas e dados, sendo o modelo mais indicado para organizações que pretendem ter o total controle sobre a infraestrutura. O modelo externo tem como característica hospedar o servidor fora das instalações da organização, o provedor dessa nuvem é que gerencia a nuvem (AMORIM et al., 2015);

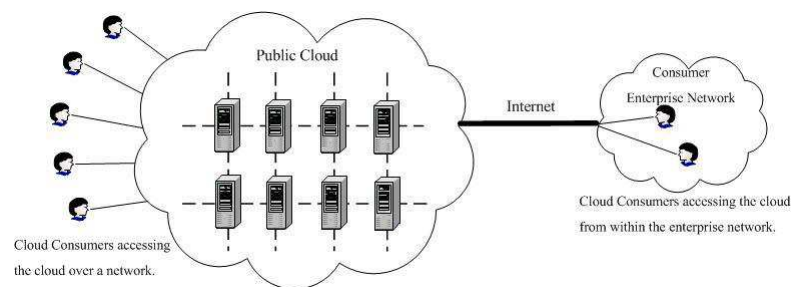
Figura 3 – Nuvem privada



Fonte: (LIU et al., 2011, p. 17).

- Nuvem Pública: Nesse modelo a infraestrutura da nuvem é disponibilizada ao público em geral, podendo ser acessado por todos e qualquer usuário que conheça a localização do serviço (SOARES; SILVA, 2015);

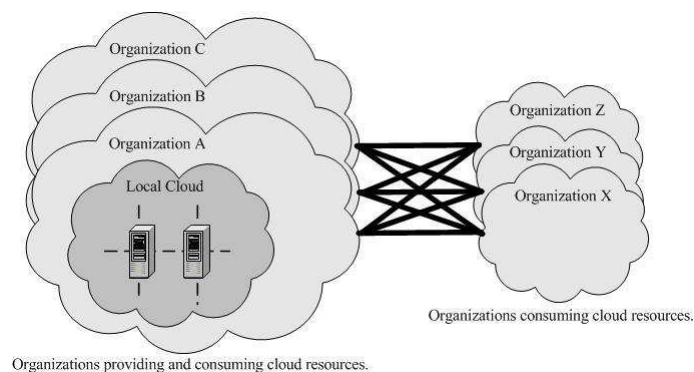
Figura 4 – Nuvem pública



Fonte: (LIU et al., 2011, p. 17).

- Nuvem Comunitária: A nuvem comunitária possibilita o compartilhamento por diversas empresas que detenham o interesse comum (MELL; GRANCE, 2011);

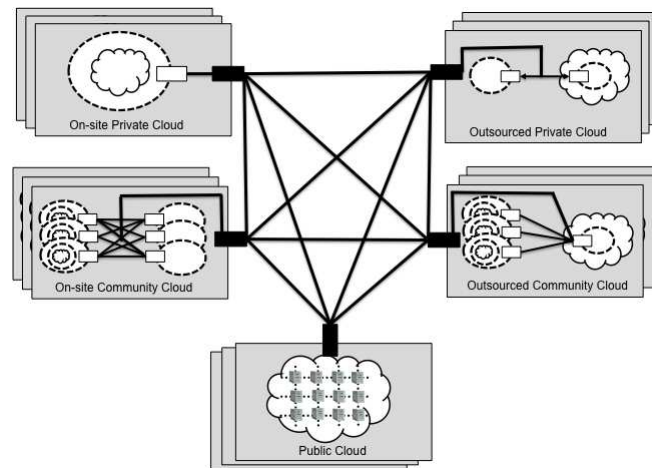
Figura 5 – Nuvem comunitária



Fonte: (LIU et al., 2011, p. 18).

- Nuvem Híbrida: Nesse modelo, a infraestrutura de nuvem é formada por uma composição de duas ou mais infraestruturas distintas (privada, comunitária ou pública) (MELL; GRANCE, 2011).

Figura 6 – Nuvem híbrida



Fonte: (LIU et al., 2011, p. 19).

3.2 Virtualização

A virtualização é um assunto recorrente na vida dos especialistas da computação, visto que a memória foi um dos primeiros componentes de computadores a ser virtualizado. Em 1970 o conceito de memória virtual já estava se desenvolvendo, nesta época existiam vários estudos sobre algoritmos de substituição de página. A virtualização de rede também é um assunto que foi muito estudado, existem *Virtual Local Area Networks (VLANs)*, que são redes virtuais de uma determinada área, e permite vários acessos de diferentes departamentos, dentro de uma mesma empresa, compartilhando uma *Local Area Network (LAN)* física com isolamento (PERLMAN et al., 2011).

Virtualização de computação consiste em uma técnica que permite ao usuário utilizar dois ou mais sistemas operacionais distintos e isolados em uma mesma máquina. Outra definição seria a criação de um ambiente virtual que simula um ambiente real acarretando a utilização de vários sistemas e aplicativos sem a real necessidade de um acesso físico à máquina no qual está hospedado (TECMUNDO, 2009).

Atualmente o interesse em se utilizar a virtualização não está somente no fato de permitir o uso de um mesmo sistema por vários usuários simultaneamente, mas pelas vantagens que o processo oferece aos usuários como segurança, custo, adaptabilidade, balanceamento de carga e suporte às aplicações legadas (DAMASCENO et al., 2016).

Para ter um melhor entendimento sobre a virtualização é necessário conhecer alguns conceitos: as instruções privilegiadas e não-privilegiadas que fazem parte de um grupo de

instruções em uma arquitetura. As instruções não-privilegiadas não modificam a alocação ou o estado de recursos compartilhados pelos processos simultâneos, por exemplo processadores, memória principal e registro especiais, já as instruções privilegiadas podem alterar o estado e a alocação desses recursos (DAMASCENO et al., 2016).

Um computador pode operar em dois modos, modo usuário (espaço de aplicação) ou modo de supervisor. Geralmente no modo usuário as aplicações são realizadas e não é possível executar as instruções privilegiadas, já que são restritas ao modo supervisor.

Em um ambiente virtualizado é necessário definir dois conceitos, os de sistema operacional hospedeiro e do sistema operacional visitante. O sistema operacional hospedeiro é de fato onde irá ocorrer a virtualização e o sistema operacional visitante refere-se ao sistema operacional onde é executado a máquina virtual.

E por fim o último conceito que iremos abordar é o conceito de *Virtual Machine Monitor* (*VMM*) também conhecido como *Hypervisor*. O *VMM* provê uma camada de abstração entre o sistema operacional e o *hardware*, é um constituinte de *software* que hospeda a máquina virtual. O *VMM* é encarregado pela virtualização e controle de recursos compartilhados entre as máquinas virtuais, sendo também responsável por ordenar qual máquina virtual vai ser executada em determinado momento (DAMASCENO et al., 2016).

A virtualização possui quatro tipos, são elas a virtualização de *hardware*, virtualização da apresentação, virtualização de aplicativos e virtualização de dispositivos de entrada e saída.

1. Virtualização de *Hardware*: A virtualização de *hardware* fundamenta-se em rodar diversos sistemas operacionais em uma mesma máquina. É realizada utilizando programas específicos que criam máquinas virtuais (*Virtual machines*, *VMs*), que emulam os componentes físicos de um computador (TECMUNDO, 2009);
2. Virtualização da apresentação: A virtualização da apresentação refere-se ao acesso a um determinado ambiente computacional sem a necessidade de estar em contato físico com ele, possibilitando a utilização de um sistema operacional completo de qualquer local (TECMUNDO, 2009);
3. Virtualização de aplicativos: A virtualização de aplicativos consiste em deter uma única cópia de um determinado aplicativo instalado em um servidor virtual, logo os usuários que desejam ter acesso ao aplicativo, conseguiram de forma direta, não havendo a necessidade de instalação do aplicativo em uma máquina física (TECMUNDO, 2009);
4. Virtualização de dispositivos de entrada e saída: Todo e qualquer dispositivo de entrada e saída é direcionada ao *VMM*, o monitor reconhece a destinação do acesso e o emula em uma cópia do *hardware* retornando-o ao sistema onde está hospedado (SOARES; SILVA, 2015).

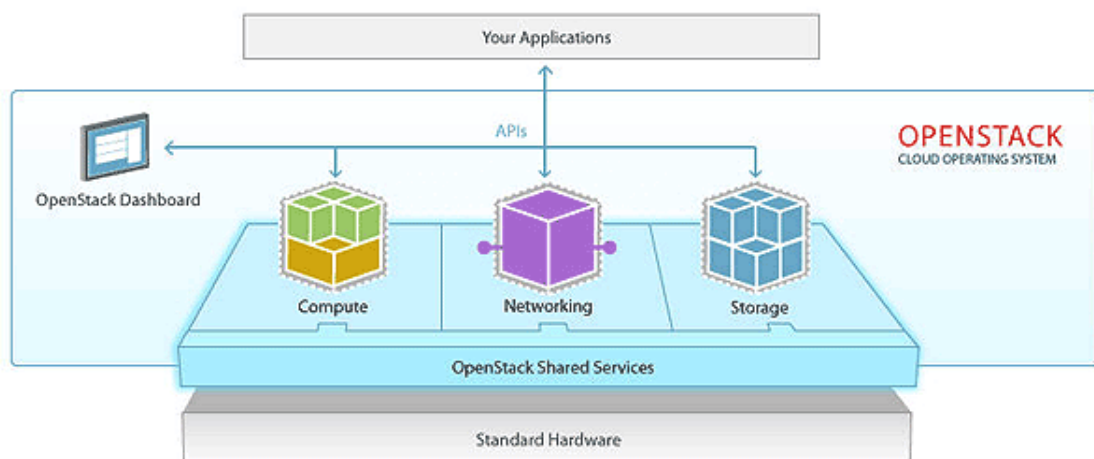
3.3 OpenStack

OpenStack é uma plataforma que surgiu como solução para computação em nuvem de código aberto a partir da iniciativa conjunta da *Rackspace Hosting* e da *NASA* em 2010, para possibilitar as organizações de oferecer ou consumir serviços de computação em nuvem compatíveis com elementos de *hardware* padrão. O código inicial era uma combinação da plataforma *Nebula*, da *NASA*, com a *Cloud Files*, da *Rackspace* (CIO, 2014). A *Rackspace* tinha o ideal de reescrever o código da infraestrutura que utilizava em seus servidores de computação em nuvem considerando tornar o código existente do *Cloud Files* aberto, enquanto a *Anso Labs*, contratada pela *NASA*, havia publicado o código beta *Nova*, um "controlador de estrutura de computação em nuvem", baseado em *Python* (OPENSTACK, 2013). Em 2012, após a difusão do projeto entre diversas empresas como *HP*, *Cisco*, *Dell* e *IBM*, foi divulgado a *OpenStack Foundation* como um órgão independente para proteger, melhorar e promover o conjunto de *software OpenStack* e criar uma comunidade *open-source* com pesquisadores, empresas e desenvolvedores (SOARES; SILVA, 2015).

A plataforma é uma alternativa *open-source*, juntamente com o *OpenNebula* e *CloudStack*, a grandes provedores de *IaaS* como Amazon Web Services, Microsoft Azure, Google Cloud Platform, IBM SoftLayer, VMware vCloud Air (GARTNER, 2015). Dentre as ferramentas de código aberto apontadas, o *OpenStack* é o mais resiliente e vantajoso para a implantação (VOGEL et al., 2016).

O *OpenStack* tem como missão criar uma plataforma onipresente de *cloud computing* de código aberto para nuvens privadas ou públicas, proporciona uma solução para infraestrutura como serviço (*IaaS*) que permite a gestão, orquestração e automação dos recursos de TI (OPENSTACK, 2013). A ferramenta se baseia nos fundamentos de entrega de serviço sob demanda em três pilares, que são eles: processamento, redes e armazenamento (RODRIGUES, 2016).

Figura 7 – *Openstack Cloud Operating System*



Fonte: *Openstack Cloud Operating System* (ORACLE, 2015).

A comunidade *open-source* é empenhada em desenvolver um *software* que seja utilizável, escalável e sem limitações de recursos ou desempenho, distribuído sob a licença *Apache 2.0*, com todos os processos documentados, abertos e transparentes e o repositório de código fonte é disponível publicamente por todo o processo de desenvolvimento (OPENSTACK, 2014). Por isso, várias empresas no mercado que buscam plataformas econômicas, ágeis e flexíveis estão se voltando ao *OpenStack* para suas soluções de *IaaS*, algumas delas são *Mirantis*, *Huawei Technologies*, *Canonical*, *Red Hat*, *SUSE*, *Rackspace* e *Platform9 Systems* (CIOREVIEW, 2017).

A plataforma nos dias atuais possui várias distribuições ou *releases*, sendo que a primeira foi divulgada em *Austin, TX* em julho de 2010 e a cada seis meses a *OpenStack foundation* lança uma nova *release* possuindo novos serviços, atualizações e correções de erros (SOARES; SILVA, 2015). No Brasil existe um movimento da comunidade de *OpenStack* onde todo mês ocorre um *Hangout* sobre assuntos técnicos relacionados ao programa e durante o ano acontece, pelo menos dois eventos presenciais chamados de *OpenStack Day* e o Encontro da comunidade no Fórum Internacional *Software Livre* (FISL) (RODRIGUES, 2016).

O *OpenStack* está sempre inovando e implementando novos projetos em cada *release*, mostrando a importância dessa tecnologia para o mercado. Na figura abaixo é possível notar a abundância de projetos que estão sendo trabalhados (OPENSTACK-A, 2017).

Figura 8 – Projetos *OpenStack*



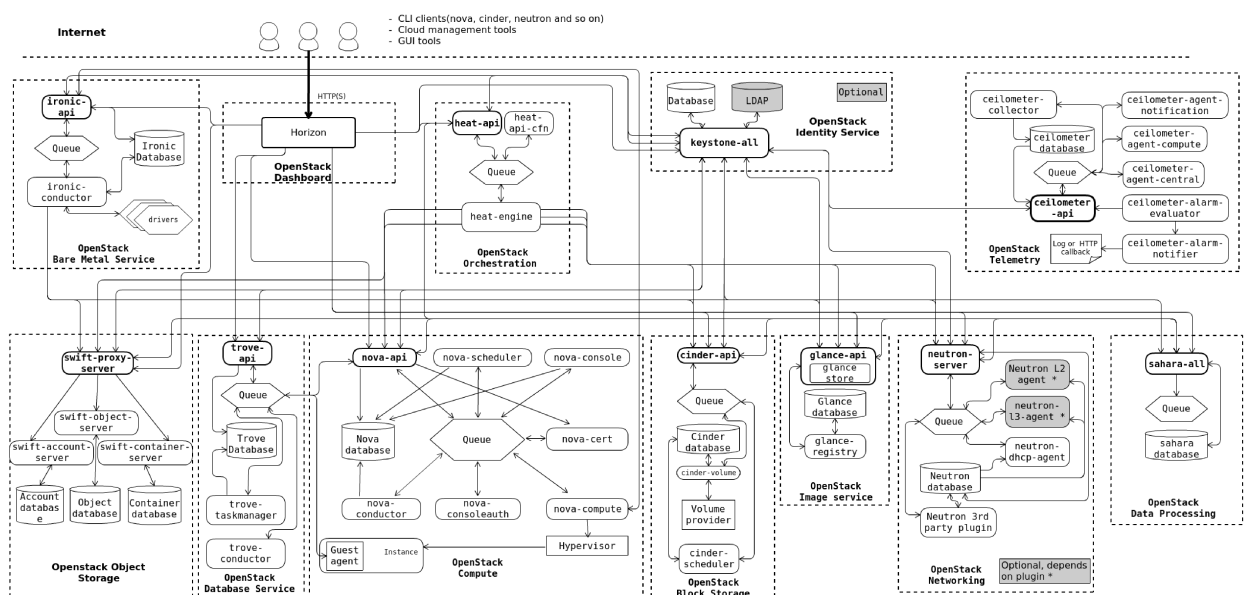
Fonte: *OpenStack projects and their mascots* (OPENSTACK-A, 2017).

A arquitetura principal do *OpenStack* pode ser dividida em três grupos: controle, rede e computação. A camada de controle engloba a *API* de serviços, interface *web*, banco de dados e serviços de mensagem. A camada de rede executa serviços de rede e agentes para gerenciamento e a camada de computação provê serviços e agentes para tratar máquinas virtuais. Os componentes do *OpenStack* operam utilizando um banco de dados e/ou serviços de mensagens. O banco de dados utilizado pode ser *MySQL*, *MariaDB* ou *PostgreSQL*, e quando utiliza-se o serviço de mensagens, *RabbitMQ*, *Qpid* ou *ActiveMQ* são os mais populares (DAMASCENO et al., 2016).

OpenStack pode ser instalado em um único ou diversos servidores, de modo *single node*, *two nodes* ou *multi-node*, mas a melhor prática é que cada um dos grupos da arquitetura principal sejam instalados em servidores separados, ou seja, cada grupo com seu próprio *node*. Isso porque uma das bases do *OpenStack* é que foi construído para funcionar com *hardware* genéricos e pode ser dimensionado horizontalmente com grande facilidade, o que significa adicionar mais servidores para agrupar recursos, independentemente da variedade desses servidores (RADEZ, 2016).

O *OpenStack* consiste em diversas partes independentes, denominados *OpenStack services* ou módulos (OPENSTACK-B, 2017). Por possuir uma arquitetura modular o *OpenStack* possibilita a escolha de combinações dos componentes que desejar, mas ainda assim existem alguns que estarão sempre presentes como o *Nova*, *Keystone*, *Neutron* e *Glance* (AMORIM et al., 2015). O diagrama abaixo mostra uma arquitetura comum, mas não a única possível, para uma nuvem *OpenStack* (OPENSTACK-B, 2017).

Figura 9 – Arquitetura lógica de nuvem com *OpenStack*



Fonte: Arquitetura lógica comum de uma nuvem *OpenStack* (OPENSTACK-B, 2017).

Os módulos se autenticam através de um serviço de identidade comum. Os serviços individuais interagem uns com os outros por meio de APIs públicas, exceto quando são necessários

comandos privilegiados de administrador. Os módulos do *OpenStack* são compostos por vários processos internamente, todos possuem pelo menos um processo de API, que recebe *API requests*, os executa e direciona para outros processos do módulo. Para a comunicação entre os processos de um serviço, um *AMQP message broker* é utilizado, e o estado do serviço é armazenado em banco de dados. Os usuários podem acessar ao *OpenStack* por meio da interface de usuário implementada pelo *Horizon Dashboard*, por meio de linhas de comandos e emissão de *API requests* através de ferramentas como *plugins* de navegadores ou *cURL*. Para as aplicações, vários SDKs são disponibilizados. Resumidamente, todos os métodos de acesso emitem chamadas *REST API* para os vários módulos do *OpenStack* (OPENSTACK-B, 2017).

A seguir serão apresentados alguns dos módulos do *OpenStack*.

Nova

O serviço *compute*, oferece servidores virtuais sob demanda, projetada sobre uma arquitetura baseada em mensagens. Com a função de hospedar e gerenciar sistemas de *cloud-computing*, e grande importância no fornecimento de infraestrutura como serviço (IaaS), sendo os principais módulos implementados em *Phyton* (SOARES; SILVA, 2015).

A *Nova* gerencia recursos de máquinas virtuais como CPU, memória, disco e interface de rede. Para se criar novas máquinas virtuais é necessário possuir um usuário autenticado com acesso ao *Glance* e uma rede configurada para instância de máquinas virtuais, os dois últimos recursos necessários são um par de chaves *SSH* e um grupo de segurança (DAMASCENO et al., 2016).

Os principais módulos do *Nova* são *Nova-API*, *Nova-Compute*, *Nova-Network*, *Nova-Volume*, *Nova-Scheduler* e *Rabbit MQ Server*.

Glance

O serviço de imagens, é um componente de gerenciamento de imagens que proporciona entrega de imagens para máquinas virtuais e serviços de *backup*. Para que as imagens possam ser criadas e replicadas várias vezes durante a criação de uma nova instância, sem que ocorra conflitos, é necessário que as imagens no *Glance* possuam algumas informações como chave *SSH* do *host* e o endereço do *MAC* removidos (DAMASCENO et al., 2016).

Horizon

O *dashboard*, fornece uma interface *web* com o objetivo de gerenciamento da plataforma *OpenStack*, instância, recursos e monitoramento de status. O *Horizon* teve início como o único aplicativo para controlar projetos de computação do *OpenStack* (DAMASCENO et al., 2016). A interface utilizada pelo *Horizon* é a *DashBoard* (SOARES; SILVA, 2015).

Keystone

O serviço de autenticação, encarregado pelo gerenciamento de usuários, possibilita o controle de usuários e o que eles estão autorizados a fazer. Para sua configuração utiliza-se o *python WSGI* e os aplicativos que estão presentes nele são configurados em conjunto através do *Paste* (OPENSTACK-C, 2017). O *Keystone* é um serviço indispensável para a elaboração de arquitetura básica de nuvem, é o único ponto de inserção entre o controle, catálogo e autenticação do *OpenStack* (ROSADO; BERNARDINO, 2014).

A arquitetura do *Keystone* divide-se em *Identity*, *Resource*, *Assignment*, *Token*, *Catalog* e *Policy* (OPENSTACK-C, 2017).

- *Identity*: proporciona a validação e informações para os usuários e grupos;
- *Resource*: esse serviço gera dados sobre projetos e domínios;
- *Assignment*: esse serviço gera dados sobre *roles*, e atribuição de *roles* que são geradas pelos serviços de *Identity* e *Resource*;
- *Token*: esse serviço tem como objetivo validar e gerenciar os *tokens* utilizados para autenticar as solicitações das credenciais do usuário;
- *Catalog*: esse serviço *catalog* gera um registro de terminal usado para descobrir o *endpoint*;
- *Policy*: esse serviço gera um meio de autorização fundamentado em regras e a interface de gerenciamento das regras associadas.

Neutron

O serviço de redes, tem como missão implementar serviços e bibliotecas associadas para fornecer abstração de rede sob demanda, escalável e independente a tecnologias. O *Neutron* é um projeto que tem como base a idéia de redes definidas por software (*Software-Defined Networking - SDN*) que fornece "*networking as a service* entre dispositivos de interface gerenciados por outros módulos do *OpenStack* (OPENSTACK-D, 2017), com objetivo de manter os serviços do *OpenStack* conectados entre si, por meio de conexões de rede. Através de uma API o *Neutron* fornece um gerenciamento de redes e endereço de *internet protocol* (IP), permitindo assim que os usuários configurem suas próprias redes dentro da nuvem *OpenStack* (SOARES; SILVA, 2015).

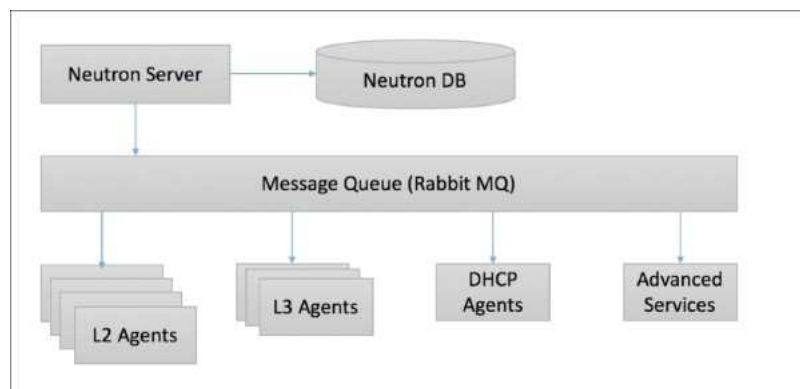
Possui uma arquitetura baseada em *plugins* permitindo que os usuários possam tirar vantagens dos recursos que estão disponíveis nos dispositivos de redes específicos. O *Neutron* possui extensões que permitem a integração com *software*, serviço de rede ou *hardware* (DAMASCENO et al., 2016).

Esse serviço substituiu uma versão antiga do módulo de redes denominada *Quantum* por um conflito de marca comercial. Antes do *Quantum* as redes do *OpenStack* eram controladas

pelo componente do *Nova*, o *Nova-Network*, que ainda pode ser utilizado para implementações simples na nuvem, mas utilizando o *Neutron* é possível oferecer diversos recursos (NETWORKCOMPUTING, 2017).

O *Neutron* possui um componente no nó do controlador denominado *Neutron Server*, juntamente com *agents*, que podem ser escolhidos entre diversos *agents* de acordo com o tipo da implementação, e também os *plugins*, que se comunicam por meio de uma fila de mensagens (NETWORKCOMPUTING, 2017).

Figura 10 – Arquitetura conceitual do *Neutron*



Fonte: Diagrama da arquitetura do *Neutron* (NETWORKCOMPUTING, 2017).

Pelo diagrama podemos identificar os principais componentes do *Neutron*.

- *Neutron Server*: sua função é ser a interface entre todo o ambiente do *Neutron* e o ambiente externo. É composto por três módulos (NETWORKCOMPUTING, 2017):
 1. *REST Service*: aceita os *requests API* de outros componentes e expõe os detalhes de trabalho interno em termos de redes, sub-redes, portas e assim por diante (NETWORKCOMPUTING, 2017);
 2. *RPC service*: se comunica com o barramento de mensagens com a função de estabelecer uma comunicação bidirecional (NETWORKCOMPUTING, 2017);
 3. *Plugin*: é uma coleção de módulos *Python* que implementam uma interface padrão que aceita e recebe chamadas API padrões e se conecta com o dispositivo final. Podem ser simples *plugins* ou implementar *drivers* para várias classes de dispositivos. Os *plugins* são divididos em *plugins* principais, que implementam o núcleo do *Neutron API* ou de gerenciamento de endereços de *IP* (NETWORKCOMPUTING, 2017).
- *L2 agent*: é executado no *hypervisor* com a função de fornecer conexões para novos servidores em segmentos de rede apropriados e notificar quando um dispositivo é conectado ou removido (NETWORKCOMPUTING, 2017);

- *L3 agent*: são executados no nó da rede e são responsáveis pelo roteamento estático, encaminhamento de *IP* e outros recursos de camada 3, como o *DHCP* (NETWORKCOMPUTING, 2017).

Swift

O serviço para *object storage*, é um componente de armazenamento baseado em objetos. No *Object Storage* os dados são armazenados em forma de objetos, sem metadados associados a ele. Com essa arquitetura o *Swift* foi criado para ser facilmente escalável e com alta disponibilidade, dessa forma é possível armazenar e recuperar grandes quantidades de dados e com uma API simples (OPENSTACK-E, 2017).

A arquitetura do *Swift* é composta por servidor *proxy*, servidor *ring*, servidores de objetivo, servidores de *contêiner*, servidores de contas e por fim um replicador.

- Servidor *Proxy*: Responsável por conectar as demais arquiteturas do *Swift*;
- Servidor *Ring*: É um mapeamento entre o nome das entidades que estão armazenadas em disco com a sua disposição física. Responsável por estabelecer quais dispositivos serão ativados ou replicados em casos de algum tipo de falha;
- Servidores de Objeto: Serve com um dispositivo de armazenamento de objetos, armazenados como arquivos binários;
- Servidores *Contêiner*: esse tem como objetivo catalogar os objetos que estão presentes dentro dos *contêineres*, embora ele não saiba onde cada objeto se localiza fisicamente, mas tem o conhecimento de onde ele se aloja em cada *contêineres*;
- Servidores de Conta: Encarregado de catalogar os *contêineres* que pertencem a uma determinada conta.
- Replicador: Responsável por conservar o sistema em um estado constante em caso de falha ou erro.

Cinder

Serviço de *block storage*, é um componente que gerencia o armazenamento em blocos para o *OpenStack*. esse tem como objetivo apresentar os recursos de armazenamento para os usuários finais, que são gerenciados pela Nova, o procedimento é realizados utilizando uma implementação de referência (LVM) ou *drivers* de *plug-in* para outro armazenamento.

É o Cinder que virtualiza e gerencia o dispositivo de armazenamento em blocos e permite aos usuários finais uma API de autoatendimento (OPENSTACK-F, 2017).

Ceilometer

Serviço de telemetria, tem como objetivo efetuar medições e monitoramento do consumo dos recursos do *OpenStack*. O gerenciamento da Telemetria é realizado por um conjunto de projetos que são eles (OPENSTACK-G, 2017):

1. AodH que é responsável por um serviço de alarmes, com base nas regras definidas em relação aos dados coletados pelo *Ceilometer*;
2. *Ceilometer* que é responsável por coletar e normalizar os dados coletados pelo *OpenStack*;
3. *Panko* foi criado com o objetivo de fornecer os índices de metadados, serviços de armazenamento de eventos que permite com que o usuário coleta as informações do estado dos recursos do *OpenStack*;
4. *Gnocchi* tem como objetivo fornecer uma saída escalável de armazenar dados a curto e longo prazo, fornecendo uma visão estatística com base nos dados de entrada.

Heat

É um serviço de orquestração, um programa de orquestramento do *OpenStack*, gerenciando as aplicações e infra-estruturas em nuvens do *OpenStack*. O *Heat* cria um mecanismo de orquestração para adicionar aplicativos de nuvem composta, baseados em arquivos em formato de texto que podem ser tratados como códigos. O *Heat* é compatível com o formato AWS *Cloudformation* ou o modelo nativo *HOT* (OPENSTACK-H, 2017).

Trove

Serviço *database*, é um banco de dados como serviço do *OpenStack*, com objetivo proporcionar aos usuários utilizar de forma rápida e fácil os recursos de banco de dados relacional ou não relacional. A princípio o serviço se concentra em fornecer isolamento em recursos em alto desempenho ao automatizar tarefas administrativas complexas, especificamente configuração, implementação, aplicação de *patches*, monitoramento e restauração através do fornecimento de banco de dados como um serviço (*Database-as-a-Service - DaaS*) e *backups* (OPENSTACK-I, 2017).

Sahara

Serviço para processamento de dados, o *Sahara* oferece uma forma simples de se gerenciar uma aplicação intensa em tratamento de dados clusterizados (*Hadoop* ou *Spark*) através da

arquitetura do *OpenStack* (OPENSTACK-J, 2017).

Os módulos descritos são os principais até o momento da realização desse estudo, porém a cada versão novos projetos são apresentados, outros são modificados e até descontinuados.

3.4 *Software-Defined Networking - SDN*

O protocolo *Openflow* foi uma iniciativa que obteve grande sucesso em seu desenvolvimento, oferecendo uma interface de programação simples que viabilizou essender o acesso e controle da tabela de consulta utilizada pelo *hardware* para determinar o próximo passo de cada pacote coletado. dessa forma, permite que o encaminhamento continuasse eficiente devido a tabela de encaminhamento continuar sendo tarefa de um *hardware*, porém a deliberação sobre cada pacote que deve ser processado pode ser transferida para um nível superior, onde diversas funcionalidades podem ser adicionadas. Posteriormente esse novo modelo foi nomeado como *Software-defined networking* (GUEDES et al., 2012).

O *SDN* é uma tecnologia em termos de rede em que grande parte dos componentes desse setor, incluindo fornecedores de equipamentos, provedores de serviços de *internet*, provedores de serviços em nuvem e usuários estão atentos, que consiste principalmente em quatro inovações: a separação dos planos de controle e de dados, a centralização do plano de controle, a programação do plano de controle e por fim a padronização de programação de aplicativos interfaces (APIs) (JAIN; PAUL, 2013).

A seguir cada uma dessas quatro vertentes será apresentada.

1. Separação de controle e plano de dados

O protocolo de rede normalmente é planejado em três planos, dados, controle e gerenciamento. Todas as mensagens criadas pelo usuário se tornam o plano de dados, e para que ocorra esse transporte de mensagens, a rede procura a rota que seja mais rápida utilizando protocolos de roteamento e rede de camada 3 (L3) ou protocolos de encaminhamento de redes de camada 2 (L2). O gerenciamento de rede é utilizado para se rastrear as estatísticas de tráfego e o estado de vários equipamentos de rede, não pode ser comparado com o controle de rede, pois são diferentes e o gerenciamento é um opcional. O plano de dados tem como função encaminhar os pacotes de dados por meio da tabela de encaminhamento, que são previamente preparadas pelo plano de controle, pelo fato de uma das principais inovações do *SDN* consistir em que o plano de controle e plano de dados devem ser separados. A lógica de controle então é implementada em um controlador que prepara a tabela para o encaminhamento e *switches* implementam a lógica no plano de dados de forma simplificada, o que reduz a complexidade e custo desses *switches* (JAIN; PAUL, 2013);

2. Centralização do plano de controle

A centralização do plano de controle, que até alguns anos atrás era algo considerado ruim, agora se torna uma boa prática pois faz com que a percepção do estado e ajuste de controle dinâmicos com base nas alterações desses estados de forma muito mais rápida do que em protocolos distribuídos. A rede é dividida em pequenos conjuntos ou áreas suficientemente pequenas para que possibilitem uma estratégia de controle comum para evitar problemas de escalonamento, assim como nos métodos distribuídos, mas a vantagem de se utilizar a centralização é de que mudanças de políticas se propagam mais rapidamente, além de que existem controladores de espera que podem ser usados em caso de falhas do controlador principal (JAIN; PAUL, 2013);

3. Plano de controle programável

Por possuir um sistema centralizado, possibilita que o gerenciador de rede implemente mudanças de controle dinamicamente a medida de que o sistema necessite, tornando o controle de programação bem mais eficaz devido a rapidez da propagação se comparado ao protocolo distribuído. O controle programável é o aspecto mais importante do *SDN* e permite que a rede possa ser dividida em várias redes virtuais que possuam políticas diferentes, mas que mesmo assim estejam em uma infraestrutura de *hardware* compartilhada e uma mudança dinâmica nessa política seria lenta e difícil com o plano de controle distribuído (JAIN; PAUL, 2013);

4. APIs padronizadas (APIs)

O *SDN* possui o plano de controle centralizado com *southbound application program interfaces (APIs)* (ou *SDN Southbound APIs*), utilizadas para a comunicação entre o controlador *SDN*, *switches* e os roteadores de rede, que é a comunicação entre a infraestrutura de *hardware* e *northbound application program interfaces (northbound APIs)* que é utilizada para comunicação entre o controlador *SDN* e os serviços e aplicações na rede (JAIN; PAUL, 2013).

Northbound e Southbound API

Como já introduzido, o *Northbound* é uma interface entre as aplicações e controladores, e oferece uma visão genérica da rede e recebe informações direta do componente e requisitos de rede (LINS, T. 2015). Um exemplo de *Northbound API* é o *Neutron*, o módulo de redes do *OpenStack*.

O *Southbound* é um dispositivo *SDN*, uma interface entre um controlador e a infraestrutura. É argumentado que alguns protocolos de controle e gerenciamento existentes anteriormente, como o *Simple Network Management Protocol (SNMP)*, *Interface to the Routing System (I2RS)*, *Software Driven Networking Protocol (SDNP)*, entre outros, também podem ser usados como *southbound APIs*, mas como cada um desses protocolos foi desenvolvido para outras aplicações

específicas eles teriam a aplicabilidade limitada se comparado com um *API* de controle nativo (JAIN; PAUL, 2013). O *Southbound API* mais conhecido é o *OpenFlow*.

Utilizando de um protocolo de rede como o *OpenFlow*, a *API* fornece o controle de todas as operações, capacidade de notificação e estatística de relatórios. Ambos são implementados de forma direta e aberta, não dependendo de fornecedores e evitando problemas de incompatibilidade (LINS, 2015).

OpenFlow

O *OpenFlow* é a principal *API southbound* e está sendo padronizado pela *Open Networking Foundation* (JAIN; PAUL, 2013). Permite que a rede seja configurada ou manipulada através de um *software*, criada com a proposta de promover a criação de redes *SDN*, utilizando elementos comuns de rede, por exemplo roteadores, ponto de acesso, *switches*, etc (LINS, 2015).

Controladores *SDN*

Conhecido também como sistema operacional de rede, é o principal componente da *SDN*, é ele quem define a natureza do paradigma *SDN*. É encarregado de concentrar a comunicação com todos os elementos programáveis da rede, ofertando uma visão unificada de rede. Existem diversas maneiras de se desenvolver aplicações *SDN*, dependendo do tipo de linguagem de programação em que o controlador foi desenvolvido. Podemos citar alguns exemplos de controladores disponíveis no mercado, como por exemplo o POX, NOX, Beacon, Floodlight (LINS, 2015).

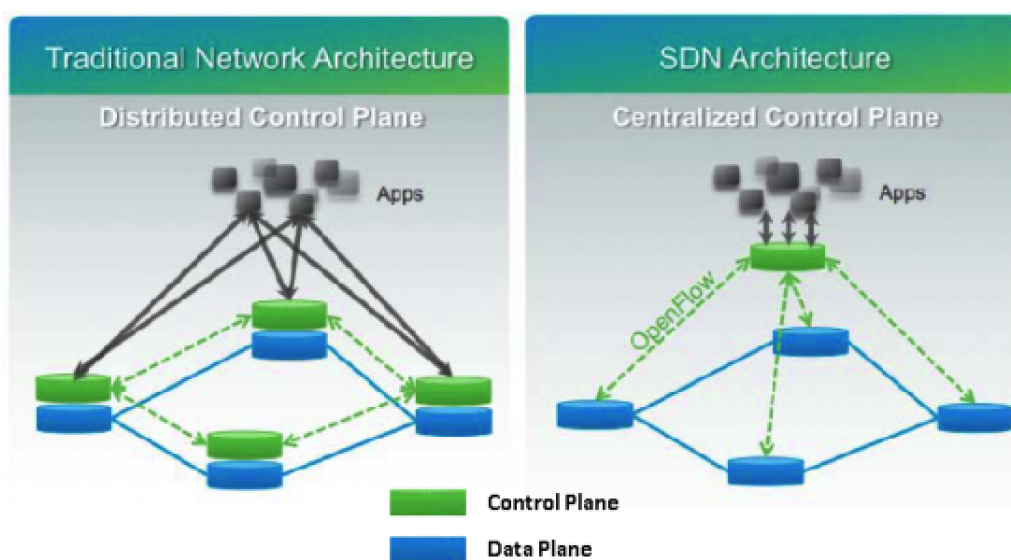
Aplicações

Ao contrário das redes convencionais, onde as aplicações devem indiretamente descrever os requisitos para um bom funcionamento envolvendo várias etapas e processamentos afim de negociar recursos e uma política de controle que a suportem, com *SDN* as aplicações passam a ser ativas na rede, pois a rede tem conhecimento da aplicação que está sendo executada. Utilizando *SDN*, as aplicações tem a possibilidade de monitorar o estado da rede e adaptá-la conforme for preciso, é o controlador de *SDN* que resume o estado de rede para aplicativos e traduz os requisitos das aplicações (LINS, 2015).

A tecnologia *SDN* vem se consolidando no mercado, para substituir a arquitetura de rede convencional devido às tendências de mercado, que são elas, as nuvens híbridas, consumerização de TI que requer redes que sejam mais flexíveis e seguras. Podemos mensurar o quão importante vem sendo a tecnologia *SDN*, analisando a quantia de dinheiro que gera em torno desse modelo, segundo o instituto de pesquisa IDC o mercado *SDN* moveu cerca de US\$ 3,7 bilhões no ano de 2016, no qual 58% desse investimento foram relacionados a infraestrutura e controle de redes de dados (VERT-A, 2017).

Juntamente com a tecnologia *NFV*, onde ambas trabalham com a ideia de virtualização de *software*, o *SDN* é uma arquitetura emergente, tendo como principal função separar o plano de controle, responsável pelas funções de controle, e o plano de dados, responsável pelo encaminhamento na rede, viabilizando com que o controle de rede se torne diretamente programável e a infraestrutura subjacente seja ligada aos aplicativos e serviços de rede (ONF, 2017).

Figura 11 – Arquitetura conceitual do *SDN*



Fonte: Plano de dados e plano de controle (LINS, 2015).

Permitindo que a rede seja mais flexível e facilitando a administração, o *SDN* pode ser aplicado em casos de uso de engenharia de tráfego, mobilidade e redes sem fio, monitoramento e medição, segurança e redes de centro de dados. (GOMES et al., 2015). Como exemplo, pesquisadores da UFAM (Universidade Federal do Amazonas) trabalham no desenvolvimento do uso de *SDN* em conjunto com técnicas de inteligência artificial para criar um sistema que ofereça um ponto focal sobre a rede com as técnicas de detecção de anomalias ou padrões comuns podem ser utilizados para observar tipos de comportamentos diferentes, como diversos tipos de ataques, e tomar medidas necessárias para prevenção (GUEDES et al., 2012). Para provedores de serviço de *internet*, é possível utilizar roteadores com recursos *SDN* e transferir a responsabilidade de controle de rede para um controlador *SDN*, possibilitando a programação de cada roteador doméstico de acordo com as regras de acesso contratadas por usuário, diminuindo o custo e aumentando a gestão e segurança no serviço disponibilizado (GOMES et al., 2015).

3.5 Network Functions Virtualization - NFV

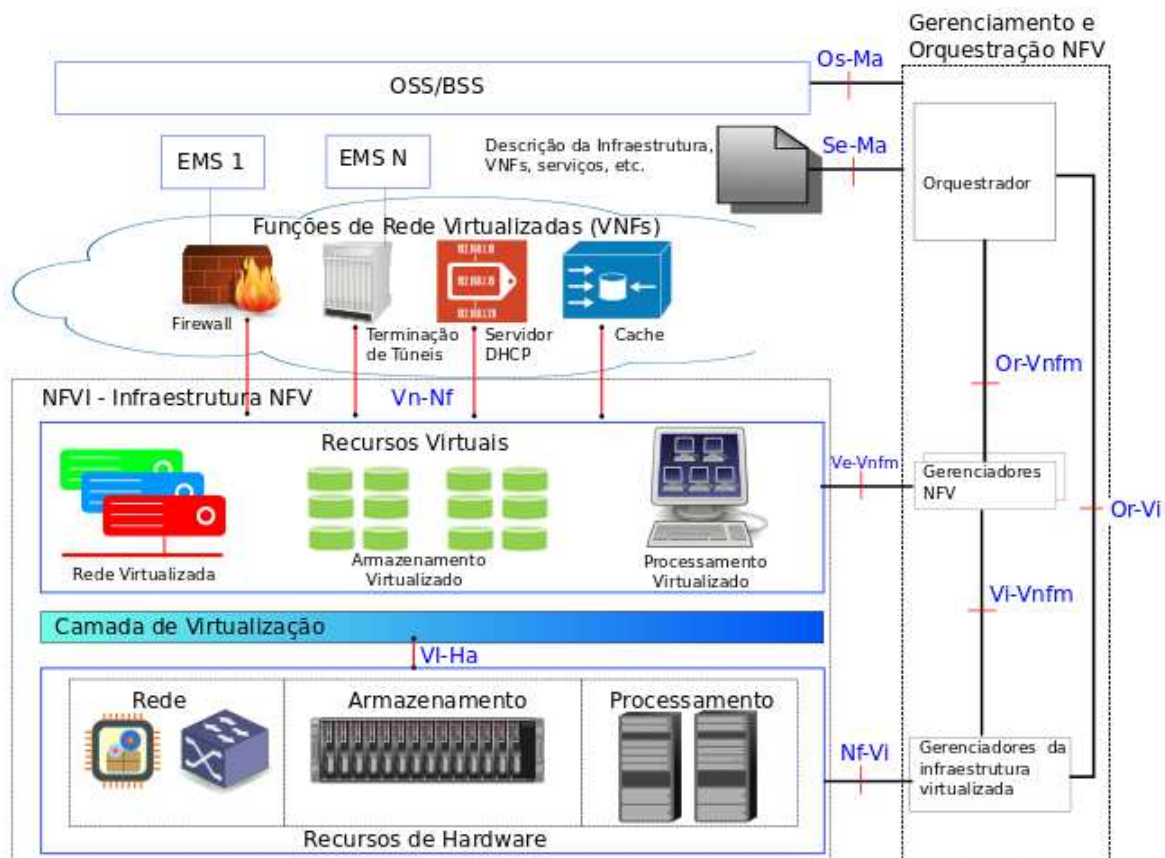
O princípio básico da virtualização de funções de rede equivale a migração dessas funções de *hardware* dedicados para os dispositivos genéricos e a concentração de diferentes funções em um único *appliance* versátil. A *NFV* torna possível a instalação de diferentes pontos de rede, indo

de acordo com os requisitos de desempenhos exigidos pela aplicação do cliente, assim como a adequação do provedor de serviço (HAN et al., 2015).

O *NFV* adiciona funções de rede por meio de técnicas de virtualização de software e desempenha em *hardware* de *commodities*, a vantagem é que esses aparelhos virtuais podem ser criados sob demanda sem que haja necessidade de instalação de novos equipamentos (CHIOSI et al., 2012).

No modelo *NFV* as funções de rede não baseiam-se diretamente na infraestrutura física de comunicações e são implementadas no domínio de software, sendo aplicações de alto nível que dependem do sistema operacional e das camadas de virtualização presentes na infraestrutura computacional (ROSA et al., 2014). A camada de orquestração é responsável pelo monitoramento, gerenciamento e pelo provisionamento das *NFV*, na infraestrutura da rede é o orquestrador quem coordena o destino dos recursos necessários para a realização das funções da virtualização de rede.

Figura 12 – Arquitetura de alto nível do *NFV*



Fonte: Arquitetura detalhada proposta pelo ETSI (ROSA et al., 2014).

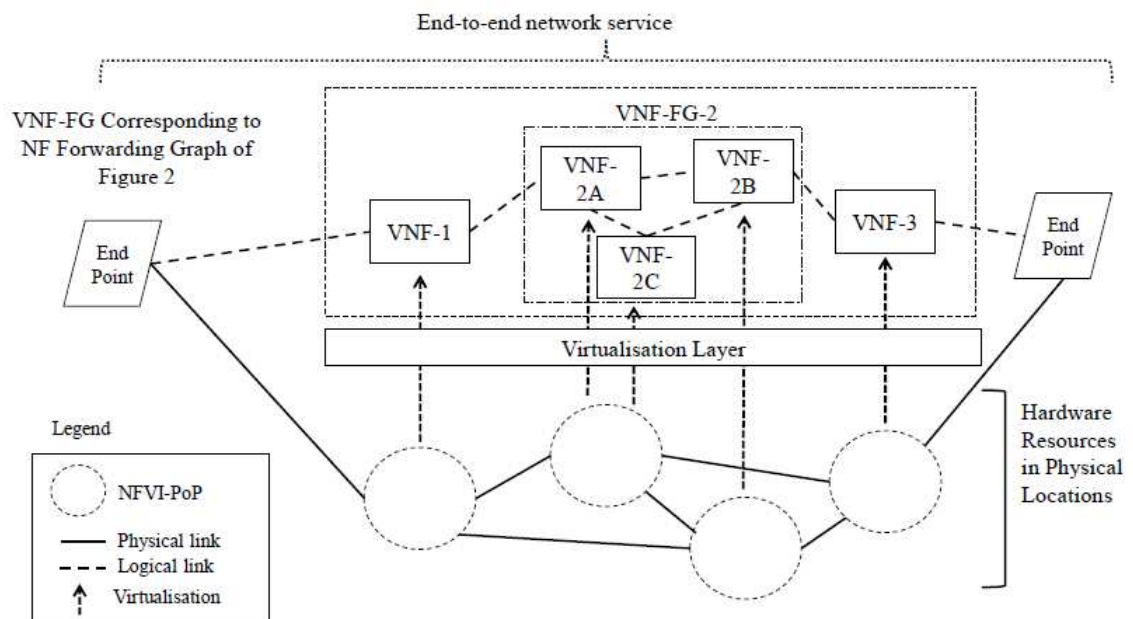
A arquitetura de referência para a implementação da *NFV* foi definida pela *European Telecommunications Standards Institute (ETSI)*. As três camadas de arquitetura da *NFV* são:

funções de rede virtualizada ou *Virtual Network Functions*, *NFV Infrastructure (NFVI)*, que inclui a diversidade de recursos físicos e como podem ser virtualizados, suportando a execução das funções de rede virtualizadas e *NFV Management and Orchestration*, responsável orquestração e gerenciamento dos recursos físicos e de *software* que suportam a infraestrutura virtualizada e o ciclo de vida de VNFs (ETSI, 2013).

As funções de rede virtualizadas são constituídas por uma camada de *hardware* e *software* que serve de base para as *virtual network function (VNF)*, esta plataforma possui três componentes, os recursos físicos, que são os *hardware* acessíveis para a virtualização das *network function* (CPU, memória e rede), os recursos virtuais e por último o *hypervisor* que delibera a alocação dos recursos físicos que serão implementados nas máquinas virtuais (BEDOR et al., 2015).

Um *Virtual network function (VNF)* consiste na virtualização das funções de rede definidas por uma ou mais máquinas virtuais, a função de rede pode ser efetuada em diversos componentes externos, nos quais podem ser implementadas em diferentes máquinas virtuais, todavia o conteúdo funcional da função de rede deve ser independente (ROSA et al., 2014).

Figura 13 – Exemplo de serviço com VNF



Fonte: Example of an end-to-end network service with VNFs and nested forwarding graphs (ETSI, 2013).

A Orquestração fica encarregada de gerenciar as redes virtuais de forma semelhante ao *IaaS* e dessa forma determinar a vida útil das *VNF*, e como elas serão alojadas pela rede, construindo um elo entre o ambiente físico e virtual, sendo também responsável pela integração entre *Operation Support System (OSS)* e *Business Support System (BSS)* (BEDOR et al., 2015).

Requisitos técnicos

Os requisitos técnicos necessários para a instalação de uma *VNF* são: capacidade de gerenciamento, desempenho de rede, confiabilidade, segurança e coexistência com as plataformas existentes.

- **Performance:** Se compararmos a capacidade de instância de uma *VFN* com uma versão física de um *hardware* dedicado, percebe-se que a instância da *VFN* será inferior. Para se introduzir instâncias de *VFN*, é necessário a criação de algoritmos eficientes, que consiga suportar a divisão da carga de rede em várias VMs distribuídas e agrupadas, além de que a infraestrutura da *NFV* deve ser capaz de englobar informações de desempenho da rede em diferentes níveis (HAN et al., 2015);
- **Capacidade de gerenciamento:** A infraestrutura da *NFV* deve estar apta à instanciar os *VNFs* nos locais adequados e no momento correto, destinar e dimensionar recursos de *hardware* e conectá-los dinamicamente para obter uma construção de serviço. Os *NFV* podem melhorar o desempenho e a utilização de recursos por meio das características de elasticidade da computação em nuvem. Os recursos de nuvem ao serem compartilhados tem seus vários serviços e seus modos de falhas independentes, podendo impulsionar o conjunto de recursos em excesso, para dividir entre as demais *VFNs*, dessa forma lidando com o aumento ou falha no tráfego (GREENBERG et al., 2009);
- **Confiabilidade e estabilidade:** Para conseguir manter um padrão de qualidade onde os clientes estarão satisfeitos com o serviço, os operadores devem mover os constituintes da *VNF* de uma plataforma de hardware para outra plataforma diferente, e ao mesmo tempo tem que atender aos requisitos de continuidade do serviço, além de precisarem especificar valores dos diversos indicadores de desempenho para alcançar a estabilidade e dar continuidade ao serviço (HAN et al., 2015);
- **Segurança:** Ao se utilizar as funções de rede virtualizadas, os operadores devem garantir a segurança dos recursos de sua rede, no entanto alguns *appliances* virtuais podem ser executados em centro de dados que não são de domínio dos operadores de rede diretamente (HAN et al., 2015). A rede e o armazenamento compartilhados podem gerar novas ameaças de segurança, juntamente com a introdução de novos elementos, por exemplo orquestradores e *hypervisor*, também podem gerar uma certa fragilidade no sistema de segurança e acarretando em uma maior detecção de invasores no sistema. Fora esses problemas, os componentes baseados em *software* podem ser de fornecedores diferentes, ou seja aumenta a chance de furos de segurança (HAN et al., 2015). Para que se tenha uma melhora no sistema de segurança, é necessário que os operadores de rede possam automatizar a colocação de *firewall* virtualizados, criando *firewall* de *software* dedicados sob demanda, com o objetivo de proteger o domínio de redes específicos e atualizar as regras de segurança de *firewalls* implementados (HAN et al., 2015);

- Interoperabilidade e Compatibilidade:

Outro ponto que está sendo bastante explorado é o fato de a NFV projetar interfaces padrões entre dispositivos virtuais, equipamentos legados e a implementação virtualizada. Em um único ambiente os operadores de rede precisam integrar e operar servidores, *hypervisor*, *appliances* virtuais de diferentes fornecedores, essa integração necessita de uma interface unificada para a interoperabilidade entre os componentes. As soluções desenvolvidas pela NFV são direcionadas para o trabalho em um ambiente híbrido com funções de rede física e virtual, o OSS e BSS existentes devem ser compatíveis com o sistema de gerenciamento de elementos de rede (HAN et al., 2015).

Uma aplicação comum para NFV, e até uma oportunidade para operadoras prestadoras de serviços de *internet*, é a possibilidade de virtualizar o roteador de borda em alguns casos, apesar de que é mais vantajoso a virtualização do CPE por conta de ser um dispositivo final e de atuação local. Um vCPE pode substituir um conjunto de equipamentos existentes do lado do cliente por soluções NFV localizadas nos provedores de nuvem ou nos centros de dados de provedores de serviços, sendo capaz de prover serviços de roteamento com *QoS*, *firewall*, monitoramento e diversos outras soluções sem a necessidade de alteração de *hardware* (ROSA et al., 2014).

4 Discussão

A partir dos estudos encontrados por meio da revisão bibliográfica, nota-se que atualmente a virtualização está presente em vários aspectos de nossa vida, existem locais de trabalhos virtuais, meios de educação virtual, as compras feitas virtualmente são cada vez mais a opção dos consumidores, o entretenimento acima de todos é totalmente virtual e grande parte da computação também (JAIN; PAUL, 2013), e até mesmo em ambientes domésticos nos dias de hoje, por tudo que suas características proporcionam ao usuário, fornecendo recursos computacionais sob demanda através da *internet*, amplo acesso de rede, elasticidade, proporcionando medição e monitoramento (ROSADO; BERNARDINO, 2014).

Além desses fatores, a *internet* das coisas (IoT), termo que se refere a uma *internet* de próxima geração onde não são conectados somente os dispositivos das pessoas, mas também permite que objetos e máquinas se conectem, troquem informações ou executem ações sem intervenção humana. É um mercado promissor que deve ser a maior fonte de crescimento de novas tecnologias de informação e comunicação (NTIC) no futuro próximo. O número estimado inicialmente de dispositivos conectados, que era de 1.5 bilhões em 2014, pode alcançar 70 bilhões ainda em 2020. Atualmente o tráfego de dados é gerado principalmente por seres humanos interagindo com seus computadores, servidores e objetos, a partir da interação M2M (*machine-to-machine*) os objetos irão gerar tráfego progressivamente maior, e a consequência disso para as infraestruturas de rede e TI ainda não é totalmente compreendida (OMNES et al., 2015).

Nesse cenário, o desafio será garantir o controle de congestionamento e evitar os efeitos colaterais causados pelo aumento do processamento de informações, quantidade do controle de serviço e controle de rede, visto que a segurança, privacidade, escalabilidade, *QoS* (*Quality of Service*) e robustez são características essenciais para o funcionamento de várias aplicações. Dessa forma, com a infraestrutura de rede e TI sendo testada em seu limite, é necessário que ela seja ágil, escalável, flexível e confiável (OMNES et al., 2015).

Com isso se fez necessário o desenvolvimento de ferramentas e tecnologias capazes de garantir esse objetivo, e o *SDN* e *NFV* são oportunidades para se construir uma nova infraestrutura. A virtualização possibilita desacoplar as funções de rede dos *hardware* de rede, o que reduzirá o custo da rede e permitirá a implantação dinâmica de serviços de infraestrutura, e a capacidade de programar os serviços de infraestrutura no lugar de reestruturar essa infraestrutura para cada uso é o que torna o *NFV* tão interessante, com as vantagens de automatizar a exploração e manutenção, permitindo reduzir o tempo de mercado, o custo operacional e aumentando a flexibilidade. Já o *SDN* surgiu como uma saída para a integração vertical e específica dos fornecedores de rede, a partir da ideia de tornar o comportamento dos dispositivos de rede programáveis, com a principal vantagem de simplificar e reduzir o custo da operação e manutenção da rede. Com *SDN*,

as políticas de rede são definidas no plano de gerenciamento, aplicadas no plano de controle executadas no plano de dados (OMNES et al., 2015).

O *NFV* e *SDN* possuem objetivos comuns que permitem que os serviços de rede sejam implantados e programados automaticamente. O *SDN* é uma ferramenta comumente utilizada para dinamicamente estabelecer conexões entre *VNFs*. Os serviços de infraestrutura abordados pelo *SDN* são de conectividade básica enquanto os do *NFV* possuem maior escopo e fornecem uma estrutura para virtualização e orquestração, e pela sua arquitetura e processos, a camada de serviços de infraestrutura é desacoplada da camada de recursos, permitindo com que os recursos de rede e TI sejam em breve, alocados e gerenciados por serviços de infraestrutura. Dessa forma, é possível que os casos de uso de IoT sejam isolados, prevenindo com que os impactos citados afetem toda a infraestrutura drasticamente (OMNES et al., 2015).

Os serviços de infraestrutura podem ser instanciados dinamicamente graças a combinação de *NFV* e *SDN*. Essas tecnologias são mutuamente benéficas, complementares entre si e promovem inovação, abertura e competitividade. Apesar de serem independentes, combinadas possuem grande valor, o *SDN* pode suportar o *NFV* melhorando seu desempenho, facilitando seu funcionamento e simplificando a compatibilidade com as implementações já existentes (HAN et al., 2015).

Com a evolução dessas tecnologias os servidores de *hardware* serão compartilhados entre muitos serviços de infraestrutura, e a implantação de um desses serviços só exigirá a implantação automática de componentes de *software* nesses servidores, aprimorando a agilidade da infraestrutura e abrindo caminho para a uma inovação orientada a *software* dentro dos serviços de infraestrutura (OMNES et al., 2015).

A implantação dessas tecnologias torna-se possível na realidade com os desenvolvimentos no meio da computação em nuvem, como a plataforma *Openstack*, que com seu esquema de gerenciamento de nuvem e orquestração permite a instanciação e migração automática de máquinas virtuais que executam serviços de rede específicos (HAN et al., 2015). Nota-se que utilizando o *Openstack*, é mais fácil de se monitorar a rede e identificar pontos de falha, além de que como sua arquitetura pode ser distribuída em vários servidores é possível evitar a indisponibilidade dos serviços de rede com as máquinas redundantes ainda funcionando.

Outro ponto é que o *Neutron*, o componente de rede do *Openstack*, pode ser um obstáculo para implantações, por não possuir capacidade de roteamento em camada 3, e utilizar o *kernel* e roteamento do *Linux*. Em um ambiente em nuvem com um grande número de redes virtuais e aplicações, todo tráfego requer roteamento e serviços de IP flutuantes tratados pelo mesmo *Neutron* L3 e o *SDN* pode distribuir os próprios operadores L2/L3 nos nós do *OpenStack* para ajudar a eliminar esse problema (REDHATOPENSTACK, 2016), por meio das APIs pode fornecer dinamicamente os recursos necessários para as aplicações e integrar efetivamente os recursos de computação e armazenamento, além de que o gerenciamento centralizado também beneficia o suporte em uma infraestrutura de nuvem baseada de vários fornecedores (SUBRAMANIAN;

VORUGANTI, 2016).

Dessa forma, a implementação de infraestruturas em nuvem utilizando o *OpenStack* possibilita o uso do *SDN* com maior garantia de qualidade e ainda sendo auxiliado com recursos que facilitam na implementação de redes nessa infraestrutura. Aliado aos princípios *NFV* podem ser criadas diversas novas possibilidades de funcionalidades, proporcionando inovação e reduções de custos como CAPEX, as despesas de capital ou investimento em bens de capital na implantação de infraestruturas, e OPEX, o capital utilizado na melhoria e manutenção dessa infraestrutura, ou seja, os custos de operação.

5 Conclusão

Diante da discussão apresentada conclui-se que o *SDN* e *NFV* devem se tornar tecnologias padrões para as implementações futuras de redes, dado que a capacidade de programação deve ser uma característica que tende a se tornar comum em todo *hardware* de rede e que permite a redução dos custos com infraestrutura, sendo assim as empresas irão buscar cada dia mais essas soluções. Para suportar essas aplicações deve existir um ambiente de computação em nuvem completo, que garanta a disponibilidade e fácil manutenção desses recursos, o que o *OpenStack* oferece e já é uma realidade em ambientes corporativos de grandes empresas.

Em estudos futuros essas tecnologias devem ser implementadas e testadas como prova de conceito e determinar suas possíveis aplicações, afim de enriquecer a bibliografia que ainda não foi muito explorada.

Referências

AMORIM, B. et al. **Trabalho teórico para o curso EEL879 - Redes de Computadores IIC**. 2015.

BEDOR, A. A. B. et al. **NFV - Network Function Virtualization**. 2015. Disponível em: <https://www.gta.ufrj.br/ensino/eel879/trabalhos_vf_2015_2/NFV/index.html>. Acesso em: 26.11.2017.

Network Function Virtualisation: An Introduction Benefits, Enablers, Challenges and Call for Action., (SDN and OpenFlow World Congress). [S.l.]: ETSI White Paper, 2012.

CIO. **Pequena cartilha sobre OpenStack para executivos de TI**. 2014. Disponível em: <<http://cio.com.br/tecnologia/2014/01/22/pequena-cartilha-sobre-openstack-para-executivos-de-ti/>>. Acesso em: 18.12.2017.

CIOREVIEW. **20 Most Promising OpenStack Solution Providers - 2017**. 2017. Disponível em: <<https://openstack.cioreview.com/vendors/most-promising-openstack-solution-providers-2017.html>>. Acesso em: 18.12.2017.

CLOUDMARKET. **IaaS, PaaS e SaaS, descubra agora tudo sobre os 3 modelos de Nuvem**. 2014. Disponível em: <<http://www.cloudmarket.com.br/blog/cloud-computing/iaas-paas-e-saas-descubra-agora-tudo-sobre-os-3-modelos-de-nuvem/>>. Acesso em: 17.12.2017.

DAMASCENO, J. C. et al. Construindo uma nuvem privada com openstack. In: PIAUÍ, F. U. E. do (Ed.). **ANAIS ELETRÔNICOS ENUCOMP 2016**. Recurso eletrônico, 2016. p. 9–37. Disponível em: <https://www.enucomp.com.br/2016/anais/anais_enucomp2016.pdf>. Acesso em: 10.11.2017.

ETSI. **Network Functions Virtualisation (NFV); Architectural Framework**. [S.l.], 2013. Disponível em: <http://www.etsi.org/deliver/etsi_gs/nfv/001_099/002/01.01.01_60/gs_nfv002v010101p.pdf>. Acesso em: 20.11.2017.

FORBES. **The History and Future of Cloud Computing**. 2011. Disponível em: <<https://www.forbes.com/sites/dell/2011/12/20/the-history-and-future-of-cloud-computing/#2af851bd79d9>>. Acesso em: 16.12.2017.

GARTNER. **Solution Comparison for Public IaaS Cloud Providers**. 2015. Disponível em: <<https://www.gartner.com/doc/3134221/solution-comparison-public-iaas-cloud>>. Acesso em: 18.12.2017.

GOMES, L. de C. et al. **Redes Definidas por Software (SDN)**. 2015. Disponível em: <https://www.gta.ufrj.br/ensino/eel879/trabalhos_vf_2015_2/SDN/index.html>. Acesso em: 20.12.2017.

GREENBERG, A. et al. The cost of a cloud: Research problems in data center networks. **Computer Communications Review**, v. 39, n. 1, p. 68–73, 2009.

Redes Definidas por Software: uma abordagem sistêmica para o desenvolvimento das pesquisas em Redes de Computadores, (Minicursos do Simpósio Brasileiro de Redes de Computadores - SBRC 2012). [S.l.]: SBRC, 2012. 160–210 p.

HAN, B. et al. Network functions virtualization: Challenges and opportunities for innovations. **IEEE Communications Magazine**, v. 53, n. 2, p. 90–97, 2015.

JAIN, R.; PAUL, S. Network virtualization and software defined networking for cloud computing: a survey. **IEEE Communications Magazine**, v. 51, n. 11, p. 24–31, 2013.

LINS, T. **Redes Definidas Por Software (Software Defined Networks) SDN**. 2015. Disponível em: <<http://www.decom.ufop.br/imobilis/redes-definidas-por-software-software-defined-networks-sdn/>>. Acesso em: 5.11.2017.

LIU, F. et al. **NIST Cloud Computing Reference Architecture - Recommendations of the National Institute of Standards and Technology**. 2011. Disponível em: <http://ws680.nist.gov/publication/get_pdf.cfm?pub_id=909505>. Acesso em: 17.12.2017.

MARTINS, D. L. S.; BERNARDES, M.; BOAVIDA, F. **IMPLEMENTAÇÃO DE CLOUD PRIVADA**. 2014. Disponível em: <<https://estudogeral.sib.uc.pt/bitstream/10316/35601/1/Implementacao%20de%20cloud%20privada.pdf>>. Acesso em: 12.8.2017.

MELL, P.; GRANCE, T. **The NIST Definition of Cloud Computing - Recommendations of the National Institute of Standards and Technology**. 2011. Disponível em: <<http://nvlpubs.nist.gov/nistpubs/Legacy/SP/nistspecialpublication800-145.pdf>>. Acesso em: 24.11.2017.

NETWORKCOMPUTING. **OpenStack Neutron Components and Concepts**. 2017. Disponível em: <<https://www.networkcomputing.com/data-centers/openstack-neutron-components-and-concepts/1828660212>>. Acesso em: 20.12.2017.

IEEE (Ed.). **A programmable and virtualized network and IT infrastructure for the internet of things**, (2015 18th International Conference on Intelligence in Next Generation Networks). [S.l.]: IEEE, 2015. 64-69 p.

OPENSTACK. **Introduction: A Bit of OpenStack History**. 2013. Disponível em: <<https://docs.openstack.org/project-team-guide/introduction.html>>. Acesso em: 18.12.2017.

OPENSTACK. **Open**. 2014. Disponível em: <<https://wiki.openstack.org/wiki/Open>>. Acesso em: 16.12.2017.

OPENSTACK-A. **Project Mascots**. 2017. Disponível em: <<https://www.openstack.org/assets/software/mascots/OS-Mascot-Key.pdf>>. Acesso em: 22.11.2017.

OPENSTACK-B. **Logical architecture**. 2017. Disponível em: <<https://docs.openstack.org/install-guide/get-started-logical-architecture.html>>. Acesso em: 19.12.2017.

OPENSTACK-C. **Keystone Architecture**. 2017. Disponível em: <<https://docs.openstack.org/keystone/latest/getting-started/architecture.html>>. Acesso em: 8.11.2017.

OPENSTACK-D. **Neutron**. 2017. Disponível em: <<https://wiki.openstack.org/wiki/Neutron>>. Acesso em: 20.12.2017.

OPENSTACK-E. **Swift**. 2017. Disponível em: <<https://wiki.openstack.org/wiki/Swift>>. Acesso em: 8.11.2017.

OPENSTACK-F. **Cinder**. 2017. Disponível em: <<https://wiki.openstack.org/wiki/Cinder>>. Acesso em: 8.11.2017.

OPENSTACK-G. **Telemetry**. 2017. Disponível em: <<https://wiki.openstack.org/wiki/Telemetry>>. Acesso em: 8.11.2017.

OPENSTACK-H. **Heat**. 2017. Disponível em: <<https://wiki.openstack.org/wiki/Heat>>. Acesso em: 8.11.2017.

OPENSTACK-I. **Trove**. 2017. Disponível em: <<https://wiki.openstack.org/wiki/Trove>>. Acesso em: 8.11.2017.

OPENSTACK-J. **Sahara**. 2017. Disponível em: <<https://wiki.openstack.org/wiki/Sahara>>. Acesso em: 8.11.2017.

ORACLE. **Deploying an Enterprise Private Cloud with OpenStack in 20 Minutes**. 2015. Disponível em: <<http://www.oracle.com/technetwork/systems/hands-on-labs/deploying-cloud-openstack-2403118.html>>. Acesso em: 19.12.2017.

PERLMAN, R. et al. **Routing Bridges (RBriges): Base Protocol Specification**. 2011. Disponível em: <<https://tools.ietf.org/pdf/rfc6325.pdf>>. Acesso em: 25.8.2017.

RADEZ, D. **OpenStack Essentials**. Birmingham: Packt Publishing Ltd, 2016.

REDHATOPENSTACK. **Red Hat Enterprise Linux OpenStack Platform 7 Networking Guide - An Advanced Guide to OpenStack Networking**. [S.l.], 2016. Disponível em: <https://access.redhat.com/documentation/en-US/Red_Hat_Enterprise_Linux_OpenStack_Platform/7/pdf/Networking_Guide/Red_Hat_Enterprise_Linux_OpenStack_Platform-7-Networking_Guide-en-US.pdf>. Acesso em: 16.11.2017.

RODRIGUES, S. **Infraestrutura como Serviço (IaaS): Iniciando com OpenStack**. 2016. Disponível em: <<https://www.profissionaisti.com.br/2016/11/infraestrutura-como-servico-iaas-iniciando-com-openstack/>>. Acesso em: 29.8.2017.

Network Function Virtualization: Perspectivas, Realidades e Desafios., (XXXII Simpósio Brasileiro de Redes de Computadores e Sistemas Distribuídos). [S.l.]: SBRC, 2014.

ROSADO, T.; BERNARDINO, J. An overview of openstack architecture. **Proceedings of the 18th International Database Engineering and Applications Symposium**, v. 14, n. 1, p. 366–367, 2014.

SOARES, J.; SILVA, G. G. da. **Implementação de um serviço de telemetria do OpenStack em um ambiente Cloud Computing**. 2015. Disponível em: <<https://monografias.ufrn.br/jspui/bitstream/123456789/1408/1/Implementa%C3%A7%C3%A3o-de-um-servi%C3%A7o-de-Telemetria-do-OpenStack-em-um-ambiente-Cloud-Computing-Jackson-Soares.pdf>>. Acesso em: 3.8.2017.

SUBRAMANIAN, S.; VORUGANTI, S. **Software-Defined Networking (SDN) with OpenStack**. Birmingham, UK: Packt Publishing Limited, 2016.

TECHTUDO. **O que é cloud computing?** 2012. Disponível em: <<http://www.techtudo.com.br/artigos/noticia/2012/03/o-que-e-cloud-computing.html>>. Acesso em: 23.11.2017.

TECMUNDO. **O que é Virtualização?** 2009. Disponível em: <<https://www.tecmundo.com.br/web/1624-o-que-e-virtualizacao-.htm>>. Acesso em: 18.9.2017.

UOL. **CONHEÇA O OPENSTACK, TECNOLOGIA QUE NASCEU DE UMA NECESSIDADE DA NASA.** 2016. Disponível em: <<http://www.uolhost.uol.com.br/academia/noticias/tecnologia/2016/05/11/conheca-o-openstack-tecnologia-que-nasceu-de-uma-necessidade-da-nasa.html>>. Acesso em: 3.9.2017.

VERT. **Software Defined Networking: saiba o que é SDN e como se beneficiar dessa tendência.** 2017. Disponível em: <<http://www.vert.com.br/blog-vert/software-defined-networking-saiba-o-que-e-sdn-e-como-se-beneficiar-dessa-tendencia/>>. Acesso em: 26.11.2017.

VERT-A. **SDN (Software Defined Network) realidade do mundo virtual?** 2017. Disponível em: <<http://www.vert.com.br/blog-vert/sdn-software-defined-network-realidade-do-mundo-virtual/>>. Acesso em: 26.11.2017.

IEEE (Ed.). **Private IaaS Clouds: A Comparative Analysis of OpenNebula, CloudStack and OpenStack**, (2016 24th Euromicro International Conference on Parallel, Distributed, and Network-Based Processing). [S.l.]: IEEE, 2016. 672 - 679 p.