

# Redes Definidas por Software

*L. Calzolari      D. Ricci      B. Claus*

Relatório Técnico - IC-PFG-17-15

Projeto Final de Graduação

2017 - Dezembro

UNIVERSIDADE ESTADUAL DE CAMPINAS  
INSTITUTO DE COMPUTAÇÃO

The contents of this report are the sole responsibility of the authors.  
O conteúdo deste relatório é de única responsabilidade dos autores.

# Redes Definidas por Software

Lucas Calzolari      Daniel Ricci      Bleno Claus      Luiz Fernando Bittencourt

## Resumo

Este trabalho está relacionado à um sub-projeto do INCT da Internet do Futuro que visa o desenvolvimento de um protótipo de infraestrutura de rede unificada, aberta e gratuita, baseada na tecnologia de virtualização de funções de rede (Network Functions Virtualization - NFV), que permita oferecer, de maneira ágil e com baixo custo operacional, um serviço inovador de comunicação, processamento e armazenamento. O projeto adota a plataforma de código aberto OPNFV e espera-se como resultado um ambiente em nuvem, acessado e interconectado pela Internet, totalmente transparente da localização das máquinas físicas e a oferta de funções de rede virtuais de gerenciamento e segurança, tais como, firewall, caracterização de tráfego e sistema de detecção de intrusão. Neste trabalho montamos toda a infraestrutura e ambiente necessários para as próximas etapas do projeto.

## Sumário

<b>1</b>	<b>Introdução</b>	<b>2</b>
1.1	Objetivos . . . . .	2
<b>2</b>	<b>Metodologia</b>	<b>3</b>
2.1	Requisitos de Hardware . . . . .	3
2.2	Instalação Física . . . . .	3
2.3	Recursos utilizados . . . . .	3
2.4	Softwares utilizados para virtualização . . . . .	4
2.5	Infraestrutura virtualizada . . . . .	5
2.6	Planejamento das Redes e VLANS . . . . .	7
2.7	Instalação do servidor Fuel Master . . . . .	7
2.8	Ambientes OPNFV instanciados . . . . .	9
<b>3</b>	<b>Resultados</b>	<b>9</b>
<b>4</b>	<b>Divisão do Trabalho</b>	<b>10</b>
<b>5</b>	<b>Conclusão</b>	<b>11</b>

## 1 Introdução

Os estáticos e enrijecidos pilares da infraestrutura da internet são legados do paradigma simples e bem definido de sua origem. No início o uso da internet seguia o padrão cliente-servidor com diferenciação clara, onde surgiu uma hierarquia na estrutura muito direta e dependente do hardware. O tráfego de dados da internet mudou com o tempo, com a introdução de streamings, sistemas descentralizados e computação em nuvem. Essas tecnologias modificaram o sentido e a quantidade da informação a ser roteada, e como consequência dessas mudanças, é necessária uma nova aproximação em relação aos estudos de redes de computadores.

Existe, no entanto, um grande obstáculo à novos estudos nessa área: o acesso à internet hoje está presente em quase todas as atividades que permeiam todos os níveis da sociedade. Desde simples atividades em redes domésticas, como troca de mensagens eletrônicas ou estudos por parte de alunos de todos os níveis da educação, até reuniões feitas por vídeo conferências no meio empresarial, nossas atividades do dia-a-dia quase sempre envolvem o acesso à uma ou mais redes de computadores. Essa enorme dependência que temos da internet faz com que a estabilidade seja uma de suas características mais importantes - afinal, parcela significativa da população depende dela para alguma coisa, é quase impossível não realizar nenhuma atividade que envolve acesso à ela.

Essa necessidade tão grande por estabilidade impossibilita pesquisas na área, uma vez que testes com novas tecnologias e/ou protocolos colocariam em risco todas essas atividades que dependem da internet. Por isso, é comum dizer que a arquitetura de redes de computadores se tornou pouco flexível, ou, calcificada[1]. Nesse contexto, as redes definidas por software (SDN - Software Defined Network) surgem como possível solução para o problema. Trata-se de um modelo de implementação de redes em que elementos de rede como switches, roteadores, entre outros, são virtualizados, possibilitando assim que sejam administrados por softwares, o que, por sua vez, permite uma maior flexibilidade na rede, pois possibilita a programabilidade destes elementos, facilitando a pesquisa e inserção de novas tecnologias.

Em uma arquitetura SDN, há um dispositivo especial chamado de controlador SDN, que é responsável pelo encaminhamento de pacotes, tirando essa decisão dos dispositivos de rede. O protocolo mais utilizado para a comunicação com o controlador é o OpenFlow[2]. O OpenFlow possui uma interface simples de programação, permitindo o controle da tabela de encaminhamento que será utilizada pelos dispositivos da rede.

### 1.1 Objetivos

Os objetivos deste trabalho são:

- O estudo e entendimento de conceitos relacionados à Redes Definidas por Software e à Virtualização das Funções de Rede.
- Montagem de um ambiente e instalação da plataforma OPNFV[3] em uma máquina servidora.
- Implantação de toda a arquitetura e redes virtualizadas, entendendo seu funcionamento e configurações.

## 2 Metodologia

Esta seção descreve os passos seguidos para instalação do OPNFV, o planejamento e montagens da infraestrutura de redes virtualizadas, a montagem do ambiente, os problemas encontrados durante esses passos e as soluções encontradas para contornar esses problemas.

### 2.1 Requisitos de Hardware

Os requisitos de hardware descritos na documentação são:

- No mínimo 3 nós, sendo um para o servidor master, um para o controlador e um para computação
- Mínimo de 16GB de RAM
- Mínimo de 256GB de espaço em disco
- 4 redes virtuais (Public, Private, Management e Storage), e uma outra para bootar por PXE (Admin)

### 2.2 Instalação Física

A primeira tentativa foi de montar uma rede física (uma máquina física para cada um desses nós, ligados por cabo de rede). Baixamos a imagem ISO disponível no site do OPNFV, versão Danube 3.0, criamos um DVD bootável com essa imagem e instalamos em um servidor. Este seria o servidor Fuel Master. Para os nós, o plano era usar notebooks para criar uma rede física, ligando-os com cabos de rede através de um switch.

Para a instalação do servidor Master, seguimos as instruções presentes na própria documentação. Após a instalação, conectamos um notebook no servidor e bootamos por PXE (Preboot Execution Environment). A ideia era que o servidor reconhecesse o notebook, e automaticamente configurasse o nó. Porém isso não ocorreu. Levantamos alguma hipóteses de qual poderia ser o motivo. O primeiro era de que deveríamos ter um cabo crossover para conectar o notebook diretamente ao servidor. Fizemos essa tentativa porém também não funcionou. A outra hipótese era de que alguma configuração de rede estivesse errada, porém não buscamos uma solução pois tínhamos outro problema, que era a falta de recursos. Apenas dispúnhamos de um notebook, além do servidor master, sendo que são necessários no mínimo 3 nós, logo precisaríamos de pelo menos 2 notebooks. Resolvemos, então, partir para uma nova abordagem, virtualizando toda a infraestrutura.

### 2.3 Recursos utilizados

Devido à complexidade do software que roda no servidor Master ou Fuel Deployment Master, se faz necessário um hardware robusto. Pela documentação, é exigido uma máquina com 16GB de memória RAM. Porém, realizamos alguns testes antes com uma máquina de menor capacidade de processamento a fim de configurar um ambiente minimamente funcional e, em seguida, ir acrescentando novos nós de controle ou processamento de forma

incremental. Assim, foi possível montar um ambiente que exigisse o menor recurso computacional possível, e verificamos seu funcionamento, mesmo com requisitos menores que os especificados.

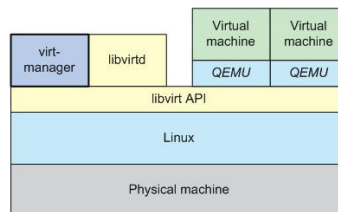
Como optamos pelo ambiente virtualizado, não se fez necessária a utilização de equipamentos como Hub ou Switches, foram utilizados apenas dois computadores, sendo o primeiro com capacidades menores que os requisitos mínimos e outro que atende os requisitos.

- Máquina 1:
  - 8GB de memória RAM
  - 1TB de memória em disco
  - Processador intel i7 com suporte a virtualização
- Máquina 2:
  - 16GB de memória RAM
  - 1TB de memória em disco
  - Processador Intel Xeon de 16 núcleos com suporte a virtualização

## 2.4 Softwares utilizados para virtualização

Para a virtualização utilizamos o Virtual Machine Manager[4], ou virt-manager, que é um conjunto de aplicativos leves, que apresenta interface gráfica com o usuário (GUI) para gerenciar máquinas virtuais (Virtual Machines - VMs). Além dos recursos de gerenciamento para VMs, o virt-manager também possibilitou que criássemos redes virtuais, necessárias para criar a infraestrutura virtualizada. Uma representação do virt-manager pode ser vista na Figura 1.

Figura 1: Uma representação simples da pilha do virt-manager com QEMU



Uma vantagem de utilizar o virt-manager é que ele faz uso do KVM[5], (Kernel-based Virtual Machine), um software de virtualização de máquinas cujo processador possua suporte à virtualização, facilitando a criação e instalação de VMs, oferecendo melhor gerenciamento e desempenho[6], o que foi muito útil devido às limitações de recursos. O KVM trabalha junto com o QEMU, um software de emulação, que em conjunto com o KVM, oferece suporte à virtualização diretamente no kernel.

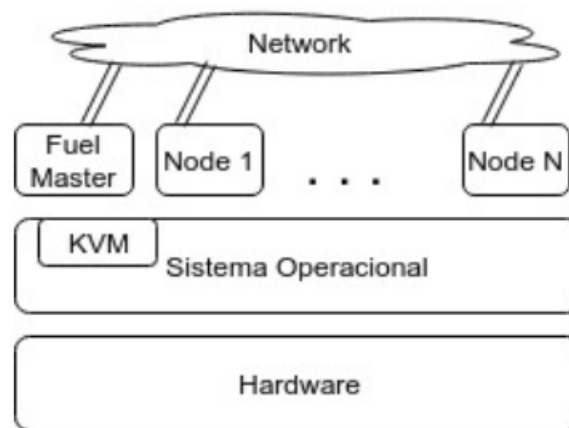
Outro software utilizado, para o armazenamento nos nós virtualizados, é o Ceph[7], que permite criar um ambiente com espaço de armazenamento escalável com replicação, tolerância à falhas e alto desempenho com cargas de trabalho variáveis.

Além disso, vale mencionar dois outros softwares que foram muito importantes: o Cinder, que é um serviço para armazenamento que pode ser consumido pelo OpenStack, virtualizando a gestão do armazenamento e fornecendo uma API para consumir recursos. E também o neutron-server, que é o processo principal do servidor OpenStack Networking.

## 2.5 Infraestrutura virtualizada

O ambiente virtualizado necessário para rodar o framework OPNFV versão Denube 3.0 pode ser visto na Figura 2. O software exige a utilização do virtualizador KVM (Kernel-based Virtual Machine), sendo que após a correta virtualização do servidor Fuel Master, os outros nós devem ser conectados entre si e com o Fuel Master por uma complexa infraestrutura de redes, representada na figura de forma simplificada por Network. No item 2.6 esta rede será explicada com mais detalhes, mas por enquanto vale destacar que uma dessas redes deve ser isolada e interligar diretamente os nós e o Fuel Master com a finalidade de reconhecimento.

Figura 2: Esquema representando o ambiente virtualizado criado



Os nós devem ser configurados para realizar boot no modo PXE, pois desta forma a placa de rede configurada para a rede interna citada acima, começará a se comunicar com o Fuel Master e uma versão simplificada de um sistema operacional será transferida para o nó apenas com a rede interna configurada. Neste ponto o Fuel Master apenas reconhecerá os nós e todas as configurações das outras redes e funções de processamento e controle deverão ser configuradas.

A infraestrutura pode ser montada em uma única máquina, porém durante nosso projeto tínhamos apenas a máquina 1 (que não possui os requisitos mínimos recomendados) sempre disponível, já que a máquina 2 é um servidor instalado em uma rede privada do instituto de computação da Unicamp. Para o desenvolvimento e acesso remoto sem causar problemas com a rede dos servidores do instituto, as redes foram virtualizadas com cuidado pois o servidor Fuel Master possui um DHCP próprio para distribuir IPs aos nós conectados nas diferentes redes, então para a máquina 2 existia a possibilidade de ser configurada uma

dessas redes, conflitando o IP com as máquinas já em funcionamento no laboratório.

Para realizar os acesso remoto foi utilizado um terceiro host apenas como proxy de acesso ao servidor. Foi criado um túnel SSH para facilitar o acesso e poder executar as interfaces gráficas (como o virt-manager) das aplicações remotas do servidor em nossas máquinas e também acessar a interface Fuel IU web disponibilizada pela aplicação Fuel Master que rodava na porta 8443 do nó virtualizado correspondente ao server. A complexidade do acesso remoto ficou alta devido à necessidade de exportar a interface gráfica do server para nossas máquinas, pois se fez necessário acessar de nossa máquina uma máquina de proxy, e depois a máquina servidora, que por sua vez estava com todos os nós virtualizados rodando, então a partir da máquina servidora se acessava o nó Master e só então era possível acessar os outros nós.

Os comandos para o acesso remoto são:

```
ssh -L [PortBind]:[IpServer]:[PortSsh] -p [PortSshProxy] [UserHostProxy]@[IpProxy]
```

```
ssh -XY -p [PortBind] [UserServer]@localhost
```

onde:

- PortBind: Uma porta local que não esteja em uso, para realizar o túnel ssh.
- IpServer: IP da máquina servidora.
- PortSsh: Porta ssh configurada na máquina servidora.
- PortSshProxy: Porta ssh configuração da máquina intermediária que faz o papel de proxy.
- UserHostProxy: Usuário da máquina intermediária.
- IpProxy: Endereço IP da máquina intermediária.
- UserServer: Usuário da máquina servidora.

Para acessar em um navegador local a interface web rodando no nó Fuel Master, virtualizado na máquina servidora, com acesso somente por uma máquina intermediária, o comando é:

```
ssh -f -N -D [LocalPort] -oProxyCommand="ssh -W %h:%p -p [PortSshProxy] [UserServer]@[IpProxy]" [UserServer]@[IpServer]
```

onde LocalPort é uma porta local que não esteja em uso.

Depois basta alterar a seguinte configuração no navegador: Proxy Socket Host = localhost, Port = [LocalPort] e acessar pelo navegador a URL: https://[Ip]:8443, substituindo [Ip] pelo IP configurado para o nó Fuel Master.

## 2.6 Planejamento das Redes e VLANs

O OPNFV exige, para seu funcionamento, uma infraestrutura de redes com diferentes funções entre o nó Fuel Master e os outros nós. Essas redes podem ser implementadas através de VLANs (Virtual LAN) utilizando assim um número menor de interfaces de rede dos nós, uma vez que mais de uma rede pode ser colocada em uma mesma interface, diferenciadas por uma tag atribuída à rede, ou mesmo implementar cada rede necessária em uma interface, sem a utilização de VLAN, e fazendo um relacionamento *rede x interface* um para um.

As redes criadas foram:

- Rede de Administração: Rede responsável por reconhecer os nós através do boot PXE. Na fase de deploy, que é a fase pós-validação das redes configuradas, são instalados os sistemas operacionais completos nos nós através desta rede de administração. Esta rede é isolada de comunicação externa e deve ter uma interface de rede no Fuel Master configurada com um IP dela, para que ele possa reconhecer os novos nós à medida que são inicializados e suas placas de redes conectadas nesta rede. Não é necessário DHCP pois o Fuel Master já possui um servidor DHCP que se configura na instalação.
- Rede Pública: É a rede que possui comunicação com a internet, e que permite acesso aos controladores e nós de computação. No processo de instalação dos sistemas operacionais, é feito o download de pacotes específicos de repositórios que se configuram na instalação do Fuel Master, então o acesso à internet se faz necessário, e é feito por essa rede.
- Rede de Armazenamento: Esta rede é usada para fornecer serviços de armazenamento, tal como o tráfego de replicação do Ceph.
- Rede de Gerenciamento: É usada principalmente para o gerenciamento do OpenStack Cloud e para acessar os serviços OpenStack.
- Rede Privada: Esta rede facilita a comunicação entre as máquinas virtuais de cada inquilino. Os espaços de endereço da rede privada não fazem parte do espaço de endereço da rede pública.

Estas três últimas redes são isoladas, sem NAT, e não é necessário DHCP, pois os IPs serão atribuídos no processo de deploy pelo nó Fuel Master. Um esquema mostrando essa estrutura de redes pode ser visto na Figura 3.

## 2.7 Instalação do servidor Fuel Master

O servidor Fuel Master é a figura central no processo de montagem e configuração do ambiente. Para instalá-lo baixamos uma imagem ISO disponível no site, que é referente à instalação virtualizada. Utilizamos o virt-manager e instanciamos o nó com 3GB de memória RAM e 80GB de disco, além de criarmos todas as interfaces de rede conectadas



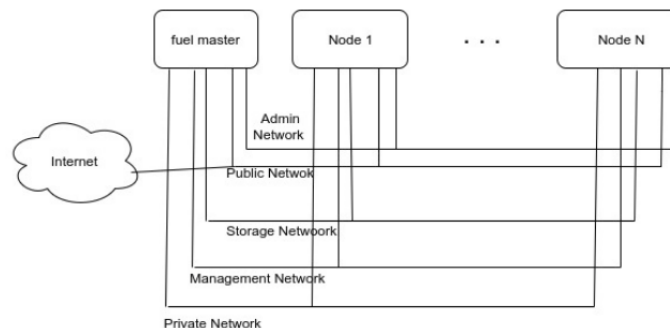
às redes, conforme descrito no item anterior. Optamos por criar uma interface para cada rede para podemos testar o ambiente com e sem VLANs.

Durante a instalação é necessário configurar vários parâmetros:

1. IPs das interfaces de redes adicionadas: Estes IPs devem estar contidos dentro do domínio de cada rede criada e configurada acima. O gateway padrão deve ser o gateway da rede pública pois é a única com acesso a internet, a máscara de rede pode ser configurada de acordo com a necessidade.
2. Restrições de acesso ssh deve ser colocado uma rede que o usuário esteja conectado ou remover a restrição, pois o padrão é deixar apenas a rede de administração acessar.
3. DHCP da rede de administração e PXE: É preciso colocar um intervalo de IPs para a rede de administração, já que o servidor Fuel Master usará estas definições para atribuir IPs às máquinas que tentarem se comunicar pela rede de administração ao realizarem o boot no modo PXE.
4. É necessário adicionar um DNS externo para o correto acesso à internet.
5. Configuração dos repositórios onde ficam as imagens dos sistemas operacionais que o servidor Fuel Master instalará nos nós na fase de deploy.
6. O servidor Fuel Master possui servidores NTPs para poder realizar a correta sincronização dos relógios dos nós. Pode-se utilizar os fornecidos (0.fuel.pool.ntp.org, 1.fuel.pool.ntp.org e 2.fuel.pool.ntp.org), ou utilizar algum outro disponível.
7. Também é possível definir os usuários e senhas dos nós configurados.

Após a correta configuração, uma aplicação subirá na porta 8443, com a qual é possível criar ambientes e validar as configurações de redes. Também é possível utilizar comandos CLI. Muitas vezes esses comandos foram necessários devido à bugs na interface, como por exemplo, uma tarefa de deploy entrar em loop e não ser possível pará-la. Pelos comandos de CLI é possível listar todos os nós encontrados e o estados atuais deles, verificar todos os ambientes criados e listar as tarefas realizadas, podendo então ver quais estão em loop e removê-las da lista de execução.

Figura 3: Esquema simplificado das redes virtualizadas



## 2.8 Ambientes OPNFV instanciados

Criamos vários ambientes, desde robustos até o mais simples possível, a fim de verificar o quão simples deveria ser o ambiente para rodar na máquina 1, e posteriormente fazer a validação deles na máquina 2.

- Ambiente 1: Criamos um ambiente com OpenDaylight e segmentação por tunelamento (assim evitando o uso de VLANs) utilizando Ceph para armazenamento. Foram instanciados 4 nós.

Nome	Papel	RAM	Disco
Node 1	Controller - Ceph	1,0GB	150GB
Node 2	Controller - Ceph	1,0GB	150GB
Node 3	Controller - OpenDayLight	1,0GB	150GB
Node 4	Controller - Ceph	1,0GB	150GB

- Ambiente 2: Criamos um ambiente com Neutron e plugin ML2 também com segmentação por tunelamento. Foram instanciados 3 nós.

Nome	Papel	RAM	Disco
Node 1	Controller - Ceph	1,5GB	150GB
Node 2	Controller - Ceph	1,5GB	150GB
Node 3	Controller - Ceph	1,5GB	150GB

- Ambiente 3: Novamente um ambiente com Neutron e plugin ML2 também com segmentação por tunelamento. Foram instanciados 2 nós.

Nome	Papel	RAM	Disco
Node 1	Controller - Cinder	2,5GB	150GB
Node 2	Controller - Base OS	2,0GB	150GB

## 3 Resultados

Os ambientes descritos foram testados nas máquinas 1 e 2. Os resultados obtidos em cada uma delas são mostrados nas tabelas a seguir:

- Máquina 1:

	SOs instalados	OpenStack instalado	Deploy
Ambiente 1	Falhou	Falhou	Falhou
Ambiente 2	Falhou	Falhou	Falhou
Ambiente 3	Instalou	Falhou	Falhou

Apesar de toda a infraestrutura de rede estar montada corretamente, a fase de deploy, que é a fase de instalação dos sistemas operacionais e OpenStack, não foi bem sucedida nem com um cenário trivial, como o ambiente 3. Ocorreram muitas falhas durante essa fase para todos os ambientes, com muitos scripts reportando timeout. Chegamos à conclusão que com os recursos disponíveis na máquina 1 não seria possível completar a fase de deploy, mesmo com um cenário simples. Foi necessário, portanto, um hardware mais robusto, como o máquina 2.

- Máquina 2:

	SOs instalados	OpenStack instalado	Deploy
Ambiente 1	-	-	-
Ambiente 2	-	-	-
Ambiente 3	Instalou	Instalou	Sucesso

Como a máquina 2 possui mais recursos, o ambiente 3 foi instalado e configurado com sucesso. Infelizmente, por problemas de rede do instituto, onde está instalada a máquina 2, não foi possível verificar o funcionamento dos ambientes 1 e 2. Porém, mesmo tendo sucesso no deploy, muitos problemas foram encontrados, como a perda das referências dos servidores NTP por parte dos nós, o que fez com que fosse necessário sincronizações frequentes para ajustar as referências, pois isso gerava falhas nos scripts, que precisavam dessas sincronizações. Outro problema foi que, sempre que o servidor DNS fosse alterado no servidor Fuel Master, os nós deveriam ser reconfigurados manualmente, pois eles não atualizavam automaticamente, gerando falhas nos scripts.

Após a fase de deploy bem sucedida do ambiente 3, na máquina 2, foi possível acessar o plataforma OpenStack do OPNFV, visualizar a topologia da rede e verificar a correta instalação do ambiente.

## 4 Divisão do Trabalho

Todos os membros participaram de todas as etapas do projeto, porém em algumas partes específicas houve maior foco por parte de alguns integrantes:

- Estudo e entendimento dos conceitos de redes definidas por software e virtualização de funções de rede: todos.
- Instalação Física: Bleno e Lucas.
- Criação da infraestrutura virtualizada: Bleno e Daniel.
- Planejamento das redes e VLANs: Bleno e Daniel.
- Instalação do servidor Fuel Master: Bleno e Lucas.
- Criação dos ambientes e testes finais: todos.

## 5 Conclusão

A realização desse projeto nos proporcionou um grande aprendizado sobre o funcionamento de um ambiente SDN e de funções de rede virtualizadas, bem como sobre a plataforma OPNFV. Foi possível entender melhor o processo de instalação e de configuração do ambiente e das redes virtualizadas para o seu correto funcionamento, e também aprender mais sobre os recursos necessários, informações que devem ser de grande utilidade para o prosseguimento do projeto de desenvolvimento de uma infraestrutura de rede unificada.

Pensando na continuidade do projeto, é possível, a partir dos resultados obtidos e da instalação bem sucedida da plataforma, explorar mais funcionalidades e fazer o deploy com ambientes mais complexos. Dessa forma, seria possível a realização de mais testes e medidas, a fim de avançar para os próximos estágios do projeto, que podem incluir interconexão com outras universidades, novos algoritmos e até novas funções virtualizadas de rede.

## Referências

- [1] Dorgival Guedes, Luiz Felipe Menezes Vieira, Marcos Menezes Vieira, Henrique Rodrigues, Rogério Vinhal Nunes: *Redes Definidas por Software: uma abordagem sistêmica para o desenvolvimento das pesquisas em Redes de Computadores*
- [2] Anderson T. Balakrishnan H. Parulkar G. Peterson L. Rexford J. Shenker S. McKeown, N. and J. Turner. Openflow: enabling innovation in campus networks. Em *SIGCOMM Comput. Commun.*, volume 38, páginas 69–74, 2008.
- [3] Página da plataforma OPNFV: <https://www.opnfv.org/>
- [4] Página do Virtual Machine Manager: <https://virt-manager.org/>
- [5] Página do KVM: [https://www.linux-kvm.org/page/Main\\_Page](https://www.linux-kvm.org/page/Main_Page)
- [6] Charles David Graziano: *A performance analysis of Xen and KVM hypervisors for hosting the Xen Worlds Project*, 2011.
- [7] Página do Ceph: <https://ceph.com/>