



Usando XOS para Gerenciamento de Infraestruturas Experimentais

RF – Relatório Final

Autor: Izabelly Marrianny Alves Peres

Data: 22/01/2016

1. Introdução

As redes definidas por software (SDN) tem criado um enorme interesse da indústria e academia devido as vantagens trazidas por esse modelo de arquitetura, tal como: divisão do plano de controle com o de dados, diminuição da complexidade, escalabilidade e flexibilidade.

Atualmente, o SDN é a base de sustentação para desenvolvimento e implementação das infraestruturas experimentais, tais como: GENI, FIRE ou FIBRE. Para gerenciá-las, frameworks de controle foram desenvolvidos, como é o caso do Ofelia e do OMF, disponíveis no FIBRE.

No entanto, umas das desvantagens desses softwares é que eles geralmente estão vinculados a projetos de pesquisa e desenvolvimento e, por conta disso, possuem tempo de vida vinculado aos respectivos projetos. O que atrapalha na evolução e soluções de problemas nesses ambientes. Portanto, a opção de soluções próprias, ou soluções de mercado é uma alternativa para lidar com esse problema, bem como permitir a inovação tecnologia no ambiente.

Nesse contexto, controladores têm evoluído bastante em suas ferramentas para servirem de suporte ou base ao desenvolvimento desse tipo de framework de controle. Dentre eles pode-se destacar o controlador ONOS (Open Network Operating System), disponibilizado pela ON.Lab. Ele é um sistema operacional de rede que permite o desenvolvimento de aplicações para redes SDN. O ONOS possui o suporte, a redes OpenFlow, Netconf e OVSDB e integrado com openVirtex (OVX) permite a construção e gerenciamento de fatias de rede.

Enquanto o XOS (XaaS Operating System) que é uma camada de abstração de serviços que orquestra a combinação de SDN, NFV e serviços em computação em nuvem, todos rodando em hardware commodities, aperfeiçoando o custo e agilidade das redes.

XOS é um sistema operacional em nuvem construído com a mesma estrutura e distribuição de um datacenter, ou seja, é um ambiente projetado para abrigar servidores e outros componentes como sistemas de armazenamentos de dados e ativos de redes, seu funcionamento é exemplificado como o núcleo do UNIX, fazendo uso de um shell de comando. É uma ferramenta que fornece suporte explícito para inúmeros serviços de múltiplas camadas, podendo gerenciar, criar e implementar novas tecnologias como plugins e sincronizadores.

O XOS tem como base dois projetos open sources, o primeiro é o OpenStack que gerencia recursos virtuais em cluster de servidores, ou seja, o OpenStack é responsável pela criação e provisionamento de máquinas virtuais e redes virtuais, enquanto o XOS define a camada de abstração para uso desses serviços. O Segundo é o ONOS que é responsável por coordenar e controlar recursos de redes, já o XOS é responsável por abstrair esses recursos em forma de serviço. Na Figura 01, tem-se a visão dessa arquitetura.

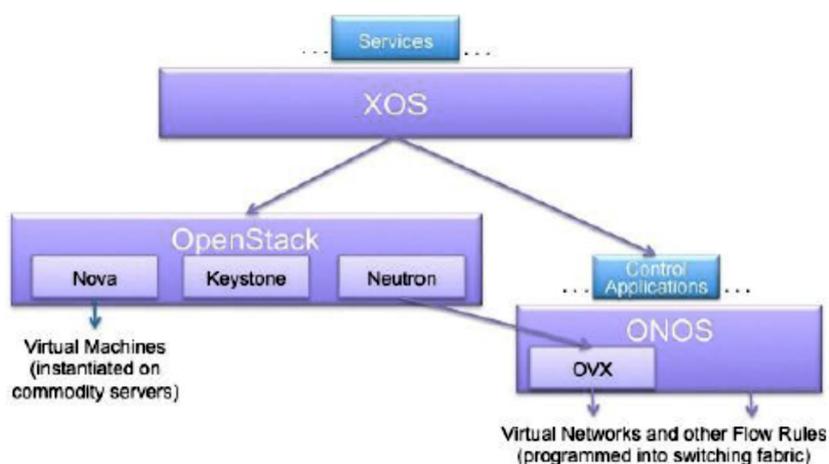


Figura 1: Arquitetura XOS

Atualmente o XOS esta sendo aplicado em dois casos de uso: Opencloud e o CORD. Opencloud é uma nuvem de produção sendo desenvolvida pela ON.LAB, PlanetLab e Internet 2. O

Opencloud utiliza a api do XOS para avançar as fronteiras dos atuais pensamentos em *cloud*, fazendo o possível para inovar tanto acima como abaixo das interfaces oferecidas pela infraestrutura como serviço (IaaS). Opencloud serve como um showcase para inovação de serviços de rede.

No entanto, o XOS não foi preparado para trabalhar com ambientes experimentais e nem possui aplicações para o tipo, logo este projeto propõe avaliar a possibilidade de usar o XOS para orquestrar os recursos do FIBRE usando os elementos que o compõem (OpenStack e ONOS).

2. Ambiente de testes

CLOUDLAB (cloudlab.us)

Cloudlab é um ambiente de testes baseado no *ProtoGeni* e desenvolvida pela Universidade de Utah nos EUA, onde o objetivo geral é flexibilizar uma infraestrutura científica de pesquisa no futuro da computação em nuvem. Pesquisadores usam o cloudlab para construir suas próprias nuvens e experimentar novas arquiteturas que irão forma a base da próxima geração de plataforma de computação. Abaixo na Figura 2, tens a ilustração do site.

The screenshot shows the CloudLab website homepage. At the top, there is a navigation bar with links for Status, Team, Contact, Technology, Hardware, Press, Docs, and Log In. The main header features the CloudLab logo and a 'Request an Account' button. Below the header, there are several sections: 'Build Your Own Cloud ...' with a description of the service, 'Recent News' with a list of updates from July to October 2015, and 'Cluster Status' which includes a table of active experiments and activity for various federated facilities.

Cluster Status	Activity
Active Experiments: 102	Projects: 241
Utah: Up, 79% full	Users: 948
Clemson: Up, 94% full	Profiles: 1,595
Wisconsin: Up, 95% full	Experiments: 14,675

Figura 2. Site do CloudLab

Configurando o Ambiente

A primeira coisa a ser feita para ter acesso aos recursos do CloudLab é solicitar uma conta para acesso, através do botão “*Request Account*”, conforme pode ser visto na Figura 2. Após o preenchimento dos dados do usuário e a descrição do projeto que será criado na infraestrutura, sua conta será criada, no entanto, ela só será habilitada após a validação de um membro do *CloudLab*. Uma vez de posse da conta habilitada será possível criar o experimento seguindo os passos apresentados na Figura 3.

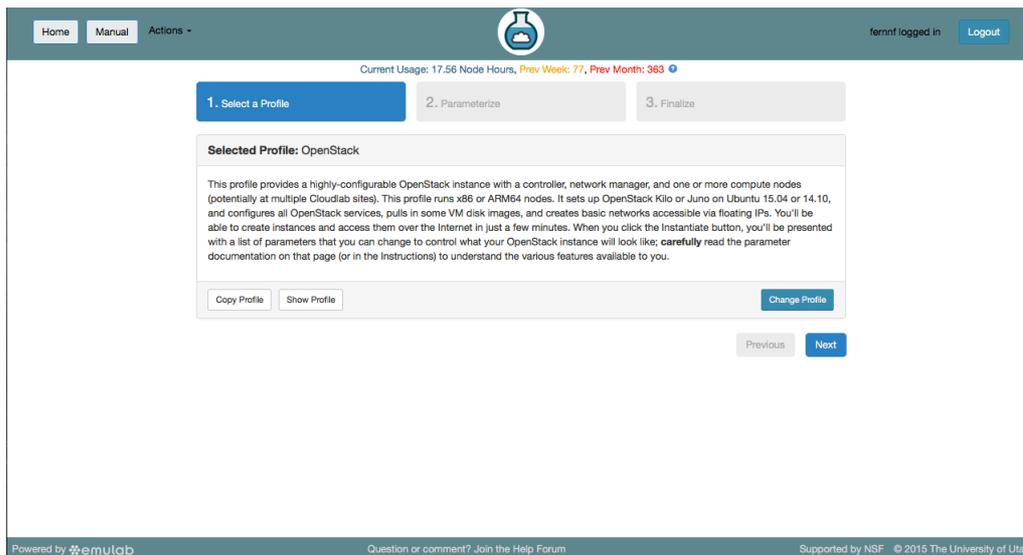


Figura 3. Seleção do Profile

No primeiro item, tem-se a seleção do profile que será inicializado, neste caso utiliza-se o profile “OpenStack”, que irá instalar automaticamente os serviços Nova, Keystone, Horizon e Neutron. Após clicar em “Next”, a próxima tela será de configuração do ambiente, conforme apresentado na Figura 4.

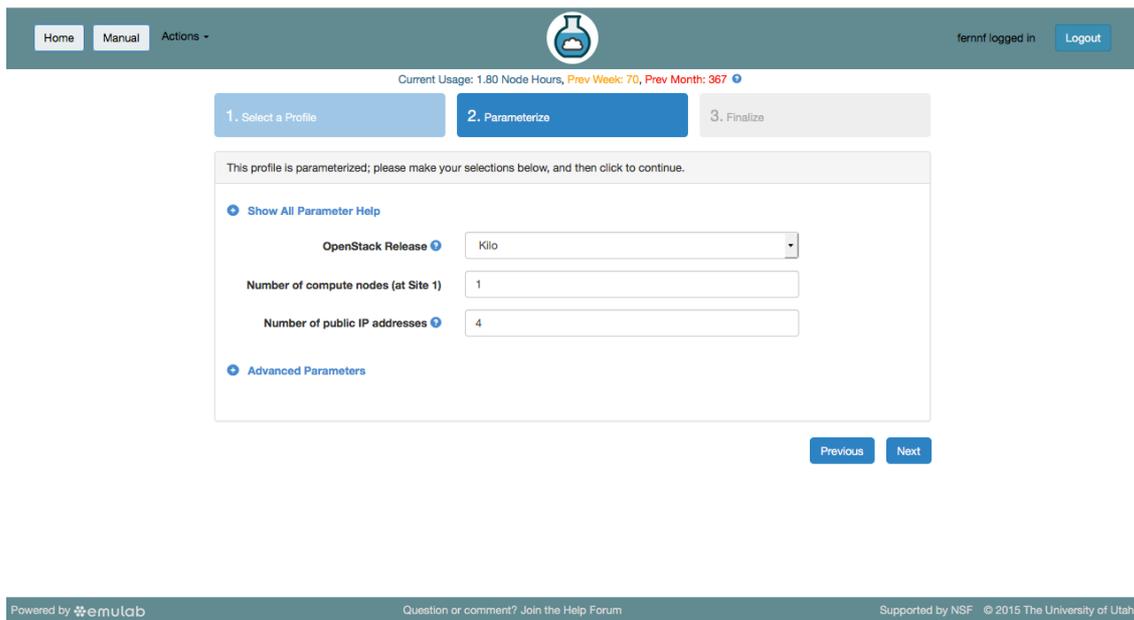


Figura 4. Configuração do Ambiente.

Nesta tela pode-se configurar a versão do OpenStack, a quantidade de nós de computação neste caso colocamos só um e parâmetros avançados nesta aba deve-se marcar a opção “Disable Security Group Enforcement” para que o XOS consiga fazer as consultas. Após isso clica-se em “Next” novamente. Na Figura 5, tem-se a configuração dos recursos e em que cluster será utilizados para instalar o XOS.

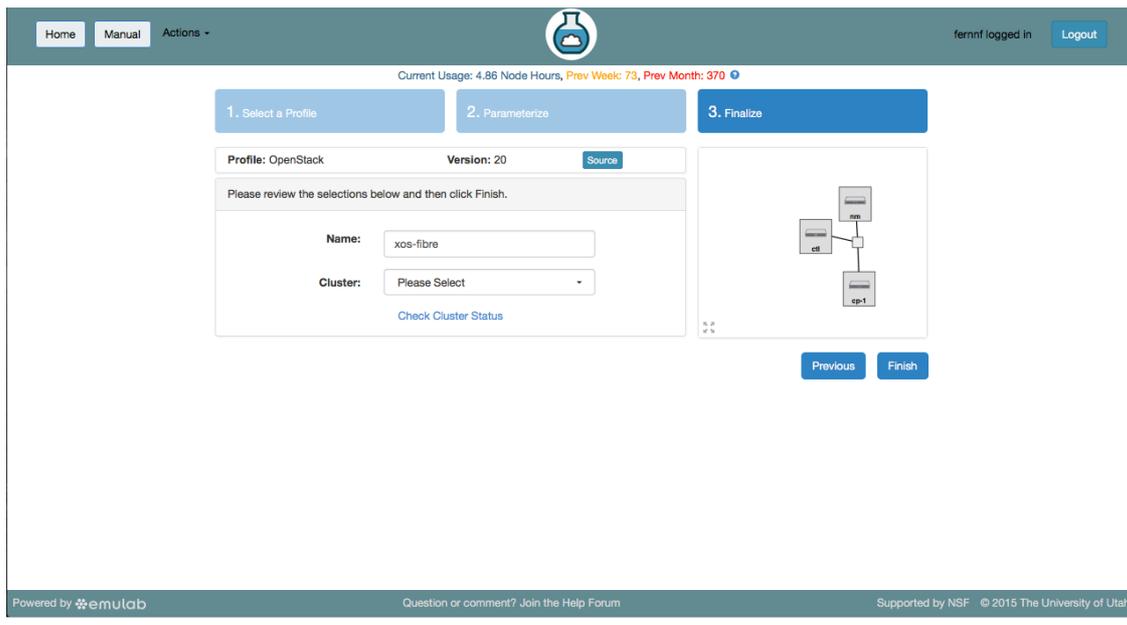


Figura 5. Alocação do Cluster

Por fim, deve-se alocar o cluster que será instalado o ambiente. E após a inicialização do experimento o Openstack estará instalado e pronto para ser usado pelo XOS e o CORD.

Instalando XOS

Para a instalação do XOS requer alguns pré-requisitos básicos como:

- Ubuntu versão a partir de 14.04 LTS 64-bit
- 2 GB ou mais de RAM
- 2 ou mais Processadores

Na instalação do XOS é necessário configurar o Git e em seguida instalar o Git do XOS:

```
$ git clone https://github.com/open-cloud/xos.git
$ cd xos/xos/configuration/devel
$ make
```

Após a execução do comando *make*, o xos será instalado e os seus componentes estarão em execução. O xos é composto por três principais serviços:

- Um banco de dados *backend* (postgres)
- Um *frontend* e API servidor web (Django)
- Um sincronizador que interage com o back-end do OpenStack.
- Para criar e executar banco de dados:

```
postgres $ cd; make build && make run
```

- para criar e executar o módulo XOS webserver:

```
xos $ cd; make build && make run
```

A partir de agora pode-se acessar a página visitando <http://localhost:80> e fazer o login usando o padrão `padmin@vicci.org` e a senha `letmein`. Pode se configurar a implantação de serviços, controladores, sites, slices. Pode-se também usar o `tosca` que é um conjunto de dados do OpenCloud.

```
Xos $ cd; make runtosca
```

Podemos criar um arquivo próprio de configuração tosca e modificar todo um conjunto de dados. É só redefinir as variáveis de ambiente TOSCA_CONFIG_PATH e executar o comando make:

```
xos $ cd; TOSCA_CONFIG_PATH = /path/to/tosca/config.yaml make
runtosca
```

O sincronizador é construído em cima da imagem do XOS, ou seja, tem que construir a imagem do XOS antes de construir a imagem do sincronizador. Os componentes XOS e sincronizador pode ser executado em hosts separados, mas deve-se construir a imagem do XOS no host que pretendesse executar o sincronizador, para criar e executar o sincronizador segue abaixo:

```
$ cd sincronizador; make build && make run
```

Via Docker Compose:

Esta ferramenta já vem atualiza no Git, porém em algumas instalações verificou-se problemas de configuração, então caso exista esse problema segue abaixo a solução:

```
curl -L https://github.com/docker/compose/releases/download/1.5.2/docker-
compose-`uname -s`-`uname -m` > /usr/local/bin/docker-compose
$ chmod +x /usr/local/bin/docker-compose
$ docker-compose --version docker-compose version: 1.5.2
$ docker-compose up -d
```

Os comandos atualizam todo o conjunto de ferramenta que compõem o Docker compose, verificam erros e os corrigem.

Configuração do CORD no CloudLab:

Esta configuração pode ser usada para alocar um ambiente de desenvolvimento CORD. Ele faz o seguinte:

- a) Configura um dataplane básico para testar fluxo de pacotes end-to-end entre um cliente e a Internet.
- b) Traz aplicativos Onos para controlar o dataplane: virtualbng, olt.
- c) Configura XOS com os serviços do CORD: vCPE, vBNG, Volt.

End-to-end dataplane

O XOS usa a configuração em um dataplane end-to-end, para executar um ambiente de desenvolvimento de serviços XOS e Onos gerenciados pelo CORD. Ele extrai as informações do hardware CORD e usa redes virtuais usando ovswitch (OVS) switches, a Figura 6 abaixo mostra a estrutura do dataplane:

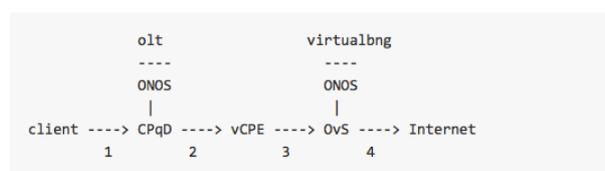


Figura 6. Dataplane do experiment

No caminho de dados são duas chaves OVS, controlada pelo olt e virtualbng aplicações do Onos. Uma vez que todos os componentes estão configurados, o cliente à esquerda deve ser capaz de obter um endereço IP via DHCP do vCPE e enviar pacotes para a Internet.

O modo de execução no CloudLab:

Após essa instalação e configuração serão criadas as máquinas virtuais. A partir da criação de todas as Vms e da configuração das aplicações do Onos, o XOS faz um mapeamento de endereços, para verificar o funcionamento da rede. Após configurar as aplicações no Onos é necessário configurar os switches de software através do dataplane e essa configuração é feita a partir do cord no diretório de configuração.

Observação: certifique-se que esteja em modo Root

Verifique se executou o componente vCPE, indo para o XOS UI, a seleção de 'Serviços', 'service_vcpe', 'Administração', 'Vcpe Tenants', e certifique-se há um ícone verde ao lado do vCPE. Se o vCPE inquilino ainda está vermelho, então a instância ainda está off devido a erros durante a tentativa de sincronizar antes o dataplane.yaml foi executado. Você pode redefinir o corrigir esse erro rastreando a vCPE Instância (Slices-> mysite_vcpe-> Instâncias, e encontrar a Instância associado ao vCPE Tenant) e apertar o botão salvar.

Após feito esses passos é necessário configurar as rotas de Ips para o acesso via ssh no Onos, através da aplicação OLT, porém é necessário ativar uma espécie de assinatura.

```
Onos> add-subscriber-access of:00000000000000001 1 432
```

Neste ponto, o cliente deve ser capaz de obter um endereço IP a partir do vCPE via DHCP. Para configurar o endereço IP e a rota padrão no cliente:

```
cliente: $ sudo rota del default gw 10.11.10.5
cliente: $ sudo dhclient br-sub
```

Após esse procedimento ser efetuado cliente deve agora ser capaz de navegar na Internet através do dataplane.

3. Visão Geral

3.1. Lições aprendidas

Xos é desenvolvido para proporcionar quase tudo como serviço (*Everything as a Service*) e pretende utilizar para isso a unificação de SDN, NFV e Cloud Computing, sobre uma única, simples e coerente modelo de programação. Como dito anteriormente o xos apresenta dois estudos de casos: *OpenCloud* e *Cord*, desenvolvidos a partir de sua API aberta e extensível. Logo a criação de uma adaptação do XOS para ambientes experimentais, é totalmente viável permitindo a interação de múltiplos recursos através de uma única interface.

Nos experimentos criados, o XOS, através do OpenCloud, foi capaz de orquestrar funcionalidades da nuvem OpenStack, como a criação de máquinas virtuais, gerenciamento de identidade de usuários, criações de redes privadas, portas virtuais e roteadores internos e externos.

Por outro lado, através do CORD foi possível orquestrar tanto a parte da nuvem com OpenStack como também a rede através do controlador ONOS que contém o plug-in VTN (*Virtual Tenant Network*) capaz de criar caminhos virtuais via equipamentos SDN. No entanto, há limitações na questão de redes virtuais SDN. Essas limitações seguem na linha que a rede sobreposta criada não pode ser controlada externamente devido ao fato de que o XOS não dá suporte há um serviço de rede virtualizada, tais como: Flowvisor ou OpenVirtex. Isso limita a criação de redes virtuais sobrepostas, essências para as redes experimentais.

A partir das experiências aprendidas pela utilização do XOS foi possível fazer uma comparação com o atual framework de controle do FIBRE (OFELIA), com itens definido pela própria equipe, no entanto não foi feito nenhum tipo de enquete, de tal forma que este resultado retrata a opinião pessoal da equipe. Na Tabela I, tem-se o quadro comparativo de funcionalidades e características entre o XOS e o *Control Framework Ofelia*

Tabela I. Comparação entre XOS e OFELIA

Características e Funcionalidades	XOS	OFELIA
Linguagem de Desenvolvimento	Django + Python	Django +Python
Suporte a Virtualização	OpenStack + Nova (Xen, KVM, Contieneer, etc.)	Xen
Suporte a Autenticação	OpenStack + Keystone	OpenLDAP
Suporte a Virtualização de Rede	VTN (sem controlador associado)	FlowVisor
Suporte a Documentação	Ativo (guide.xosproject.org/)	Não Existe
Roadmap de Desenvolvimento	Ativo e constantemente atualizado	Não Existe
Suporte a extensão de novos serviços	Possui (XOS + XOS API)	Não existe informações
Capacidade na manutenção	Fácil	Difícil
Capacidade na instalação	Fácil	Difícil

3.2. Limitações

Apesar de toda a flexibilidade do XOS, com uma API de desenvolvimento e sua característica de sistema operacional. Há algumas limitações para que ele se encaixe no mundo de frameworks de controle para ambientes experimentais.

A primeira limitação se diz respeito ao a dependência do OpenStack. A arquitetura atual do XOS trabalha com o gerenciamento de nuvens computacionais baseadas no OpenStack apenas esta tecnologia de orquestração de nuvens. Isso é um pouco preocupante devido ao fato de que a arquitetura atual do FIBRE não é baseada em OpenStack e apenas no hypervisor XEN. Portanto para a adoção do XOS ao FIBRE é necessário que a ilha do FIBRE seja alterada para o suporte ao OpenStack.

A outra limitação se diz a um requisito das redes de experimentação que é a virtualização de equipamento OpenFlow. Neste caso, o XOS não da suporte a esse tipo de virtualização. Logo isto seria um trabalho futuro caso o uso XOS venha a se tornar o padrão dentro do FIBRE.

Para suprir essa limitação propõe-se a criação de um serviço de topologias virtuais em um ambiente SDN. Neste caso, seria desenvolvido um serviço ao XOS capaz de interagir com OpenVirtex e criar essas topologias virtuais. Na Figura X, há uma ilustração da proposta.

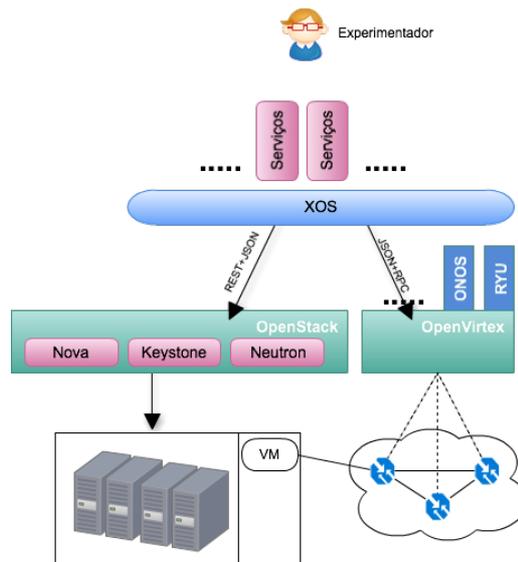


Figura x. Proposta de integração do OpenVirtex com XOS

Esse serviço serviria de suporte aos experimentadores para criarem suas topologias virtuais a partir do XOS e interligarem as suas máquinas virtuais, criadas no ambiente OpenStack a esta rede de experimentação, tanto na ilha local como no *backbone*. Além disso, haveria a necessidade de fazer uma ferramenta de interação com o experimentador para prover a manipulação dessas topologias virtuais pelo mesmo.

3.3. Problemas Encontrados

Os problemas encontrados durante essa fase de execução do projeto foi necessidade de um ambiente robusto para a interação com OpenStack, pois o mesmo necessita de uma grande quantidade de recursos computacionais, o que inviabilizava a utilização de máquinas virtuais como VirtualBox ou VMware Play.

Porém, iríamos utilizar a infraestrutura de testbed do GERCOM, como foi liberado o acesso ao CloudLab (Infraestrutura de experimentação em Cloud) a viabilidade do estudo do XOS se tornou ainda mais rápida.

Por fim, uns dos principais problemas encontrado é que houve uma grande alteração no diretório de desenvolvimento do XOS o que a equipe teve perde-se um tempo para se adaptar as novas alterações do software.

4. Conclusão e avaliação

XOS é um sistema operacional em nuvem construído com a mesma estrutura e distribuição de um datacenter, ou seja, é um ambiente projetado para abrigar servidores e outros componentes como sistemas de armazenamentos de dados e ativos de redes. O seu controle se da principalmente em Cloud, SDN e NFV.

Neste período de estudo do XOS, observou-se que a sua maturidade se desenvolveu bastante o software foi extremamente atualizado e novas características foram adicionadas ao mesmo. O processo de aprendizagem foi meio lento enquanto dependíamos de máquinas virtuais, depois que se utilizou-se ambientes mais robustos, a execução e instalação se tornou mais fácil e rápida. A nossa avaliação em relação ao XOS e a melhor possível pois atendeu a todas as nossas expectativas de funcionamento dentro de ambiente de Cloud e as experiências aprendidas poderá ser migrado para o ambiente de experimentação facilmente.

Quanto aos resultados esperados, dentro da linha de entregáveis, que foi prometido no projeto atual, conseguiu-se alcançar quase tudo o que foi prometido. Ficando faltando apenas a questão

de criar finalizar a instalação do XOS com o CORD e apresentar um exemplo de novo serviço integrado ao XOS.

Já em relação a adoção do XOS ao FIBRE, jugou-se que extremamente favorável a adoção do XOS ao FIBRE. Para prover a evolução do mesmo tanto na questão de serviços quanto em relação a tecnologias utilizadas. Sua API extensível mostra-se mais flexível que o framework OFELIA. Que atualmente não dispõe de documentação e suporte ao seu desenvolvimento.

Por fim, o esforço de implantação do XOS atualmente no FIBRE seria atualizá-lo para trabalhar com *hypervisor* de redes SDN (*OpenVirtex* ou *Flowvisor*), com objetivo de oferecer ao experimentador uma forma mais dinâmica de criar topologias virtuais aos seus experimentos. E os serviços caso seja necessário ao FIBRE.