Laboratórios Virtuais para Ensino Remoto baseados em Encadeamento de Funções de Rede Virtualizadas

Renan C. Soprani[1], Harrison F. S. Machado[1], Mateus D. Schineider[1], Maxwell Monteiro[1], Alextian Liberato[1], Gilmar Vassoler[1] e Cristina Dominicini[1]

[1]Programa de Pós-graduação em Computação Aplicada (PPComp)
Campus Serra do Instituto Federal do Espírito Santo (IFES)

{renancs, cristina.dominicini, gilmarvassoler}@ifes.edu.br

{maxmonte, alextian.bartholomeu}@ifes.edu.br

{harrison.sanches, schineider.mateus}@gmail.com

Abstract. *The cloud paradigm has been used to provide innovative and scalable education environments. In this context, this work proposes a low-cost architecture with open technologies for on-demand provisioning of Virtual Laboratories, by exploring the concept of virtualized network function chaining. As a result, teachers can use high-level policies to configure remote student access to institution resources according to the specific needs of their classroom, without requiring specialized technical support. The experiments carried out demonstrate the feasibility and evaluate the performance of the proposal in a prototype with Openstack and OSM cloud technologies.*

Resumo. *O paradigma de nuvem vem sendo cada vez mais utilizado para prover ambientes inovadores de ensino com escalabilidade. Neste contexto, este trabalho propõe uma arquitetura de baixo custo com tecnologias abertas para alocação de Laboratórios Virtuais sob demanda, explorando o conceito de encadeamento de funções de rede virtualizadas. Como resultado, um professor pode usar políticas de alto nível para configurar o acesso remoto dos alunos aos recursos da instituição de acordo com as necessidades específicas da sua aula, sem necessitar de apoio técnico especializado. Os experimentos realizados demostram a viabilidade e avaliam o desempenho da proposta em um protótipo com as tecnologias de nuvem Openstack e OSM.*

1. Introdução

Com as novas dinâmicas de ensino, as ferramentas de suporte estão migrando cada vez mais para os meios digitais, mas ainda existem vários desafios que exigem inovação nos sistemas educacionais [Gómez-Zermeño 2020]. Neste contexto, as infraestruturas legadas baseadas nos métodos convencionais de acesso aos recursos de tecnologia da informação (TI) se tornam cada vez mais defasadas e não aderentes às novas necessidades.

Por exemplo, o tempo para disponibilização de um recurso pode ser relativamente alto, pois a infraestrutura de rede não suporta customização dinâmica e políticas flexíveis [Shirzad et al. 2012]. Além disso, os laboratórios convencionais de ensino não podem ser acessados em qualquer horário e nem sempre podem ficar disponíveis pelo tempo necessário para realização das atividades. Dessa forma, é importante encontrar soluções

que possibilitem a otimização do uso de recursos computacionais ao mesmo tempo que seja possível gerenciar de forma flexível a configuração da infraestrutura para atender aos requisitos específicos para realização das atividades remotas de ensino.

Nesta direção, o emprego do paradigma de computação em nuvem permite oferecer os recursos necessários (*software*, plataforma e infraestrutura) como serviço, para acesso remoto [Bhatia and Al Sulti 2019]. Como resultado, a computação em nuvem possibilita um melhor aproveitamento dos recursos disponíveis para o ensino em comparação com laboratórios físicos, devido à utilização de laboratórios virtuais [Heradio et al. 2016].

Entretanto, a maioria das soluções existentes para laboratórios virtuais apresenta os seguintes problemas: (i) dependem de infraestrutura especializada [Makoviy et al. 2017], (ii) são proprietárias e possuem alto custo de aquisição e operação [Xu et al. 2015], (iii) apresentam problemas de escalabilidade [Rahman et al. 2016], ou (iv) não conseguem representar a granularidade de diferentes políticas de configuração de recursos de acordo com as necessidades dos usuários [Owens et al. 2019].

Neste contexto, o problema abordado por este trabalho é como fornecer uma arquitetura de Laboratório, que permita que os usuários das instituições de ensino (i.e., alunos, professores e administradores) realizem suas atividades de forma remota por meio de uma solução escalável, flexível, de baixo custo e de código aberto. Para atingir este objetivo sem depender de aquisição de equipamentos especializados e caros, a proposta deste trabalho explora os conceitos de redes definidas por *software* (SDN, do inglês *Software-Defined Networking*) [Farhady et al. 2015, Chu et al. 2019], virtualização de funções de rede (NFV, do inglês *Network Function Virtualization*) [Nakhai and Anuar 2017, Mahmoodi et al. 2017] e encadeamento de funções de serviço (SFC, do inglês *Service Function Chaining*) [Bari et al. 2016].

Uma das principais contribuições deste trabalho é desenvolver mecanismos de orquestração, que permitam que políticas de alto nível sejam traduzidas em comandos de criação de cadeias de SFCs, instanciação de máquinas virtuais (VMs, do inglês *Virtual Machines*) e configuração dinâmica da infraestrutura. Dessa forma, um docente pode criar um laboratório virtual, informando apenas algumas características básicas e escolhendo entre as opções previamente configuradas pelos administradores do sistema.

Por exemplo, dependendo da necessidade de cada atividade letiva, poderiam ser escolhidas configurações como filtro de sites permitidos durante a aula e cache de conteúdos. De forma transparente ao usuário, essas configurações são implementados pelo orquestrador da nossa solução por meio do direcionamento dos fluxos de rede por funções de rede virtualizadas (VNFs, do inglês *Virtual Network Functions*) que implementam as funcionalidades necessárias. Assim, é possível direcionar dinamicamente fluxos de pacotes específicos para uma sequência ordenada de VNFs, possibilitando: (i) aplicação de políticas de rede flexíveis e sensíveis ao contexto, (ii) modulação do tráfego e (iii) priorização dos serviços de acordo com as demandas específicas da aplicação, do perfil de usuário e do estado da rede [Medhat et al. 2016].

Com essa motivação, este trabalho propõe, implementa e avalia uma arquitetura escalável, flexível e de baixo custo para gerenciamento de laboratórios virtuais, denominada LABVER (Laboratório Virtual para Ensino Remoto), que se baseia em SDN, NFV e SFC. Como prova de conceito, foi implementado um protótipo com tecnologias utili-

zadas em ambientes de nuvem de produção, como OpenStack[1] e OSM[2]. Os resultados obtidos mostram que a solução é capaz de criar um Laboratório Virtual funcional e com bom desempenho a partir de políticas de alto nível e aplicar configurações, como bloqueio de fluxos de dados e controle parental.

Este artigo está organizado da seguinte forma. A Seção 2 discute os principais trabalhos relacionados. A Seção 3 apresenta os detalhes da arquitetura e a proposta de implementação. A Seção 4 demonstra os resultados da avaliação. Por fim, a Seção 5 discute as conclusões e trabalhos futuros.

2. Trabalhos Relacionados

Esta seção analisa trabalhos relevantes na literatura que tratam da implantação e otimização do fornecimento de recursos computacionais virtualizados para os usuários como ferramentas de ensino.

Considerando a habilitação de Laboratórios Virtuais, os seguintes requisitos são importantes: (i) suporte à definição de políticas de alto nível pelos professores e administradores, de modo a abstrair para o usuário as complexidades exigidas quanto às configurações necessárias para alocação dos recursos na nuvem; (ii) flexibilidade de definir diferentes funções de rede para aplicação de políticas de acordo com as configurações exigidas pelo ambiente educacional (e.g., restringir acesso a determinados sites durante provas e cache de conteúdo); e (iii) *software* livre e de código aberto, que possa funcionar na infraestrutura de nuvem, para redução dos custos operacionais (OPEX) e de capital (CAPEX). A Tabela 1 sumariza a comparação entre os trabalhos relacionados utilizando esses requisitos, além de incluir as tecnologias utilizadas.

O trabalho de [Xu et al. 2015] propõe o fornecimento de infraestrutura virtualizada usando o software Citrix Xen Desktop[3], a fim de reduzir os investimentos em infraestruturas pulverizadas pelas províncias chinesas, culminando numa redução global de investimentos. Os trabalhos de [Alagappan et al. 2016, Rot et al. 2018] elegeram o VMware Horizon[4] como ponto central do seu estudo, e demonstraram a redução de custos operacionais a longo prazo, mesmo considerando um elevado custo de capital. O *Microsoft VDI*[5] foi utilizado por [Nakhai and Anuar 2017] e seu estudo avalia o consumo de recursos dos sistemas operacionais virtualizados que serão fornecidos pela arquitetura. Todos esse trabalhos dependem de software proprietário, além de não explorar a utilização de políticas de alto nível, nem permitir implementar funções de rede flexíveis.

Apesar do paradigma de Laboratório Virtual possuir várias vantagens em relação aos laboratórios físicos, uma das maiores dificuldades para a migração do modelo tradicional é o CAPEX e OPEX. Porém a arquitetura LABVER, proposta neste trabalho, fornece uma solução baseada em software livre que atenda demandas dinâmicas exigidas nos ambientes educacionais, tanto de rede quanto de processamento, viabilizando a implementação de soluções inovadoras.

---

[1] https://www.openstack.org/

[2] https://https://osm.etsi.org/

[3] https://www.citrix.com/pt-br/downloads/xendesktop/

[4] https://www.vmware.com/br/products/horizon.html

[5] https://docs.microsoft.com/pt-br/windows-server/remote/remote-desktop-services/welcome-to-rds

Tabela 1. Comparação entre os trabalhos relacionados

Trabalhos	Políticas de alto nível	Funções de rede flexíveis	Software livre e código aberto	Tecnologia
[Nakhai and Anuar 2017]	-	-	-	Microsoft VDI
[Alagappan et al. 2016, Rot et al. 2018]	-	-	-	VMware Horizon
[Xu et al. 2015]	-	-	-	Citrix Xen Desktop
[Calle-Romero et al. 2019]	-	-	✓	OpenNebula
[Kabiri et al. 2017]	-	-	✓	OpenStack
LABVER	✓	✓	✓	OpenStack

A falta de aderência à infraestrutura legada e os altos custos de licenciamento dos softwares proprietários também foram considerados no trabalho de [Calle-Romero et al. 2019], que elegeu o OpenNebula[6] para fornecer uma ferramenta de apoio para atividades práticas para os alunos sem usar soluções proprietárias. Seguindo na mesma linha de software livre, [Kabiri et al. 2017] propôs o uso do Openstack como gerenciador de infraestrutura virtualizada, para facilitar o fornecimento de laboratórios experimentais e otimizar os recursos com baixa utilização.

Conforme ilustrado na Tabela 1, mesmo havendo vários estudos focados no fornecimento e otimização dos recursos virtualizados para uso educacional, esses trabalhos não fornecem uma interface de acesso e criação de Laboratórios Virtuais, que permita a configuração de rede de forma flexível por meio de políticas de alto nível, sem a exigência de conhecimentos técnicos para atender as necessidades dos ambientes educacionais. Na maioria dos casos, as configurações de políticas de rede são feitas apenas pelos administradores de rede, de forma desacoplada da solução de Laboratório Virtual com a utilização de equipamento de rede especializado.

Neste contexto, este trabalho inova ao propor uma solução aderente à infraestrutura de rede legada das instituições de ensino, para fornecer Laboratórios Virtuais usando VNFs com apoio de SFC para reforço de políticas. O uso desta abordagem traz vantagens como segurança e flexibilidade no acesso, uma vez que os reforços de políticas são configurados por meio do SFC e independem da infraestrutura de rede da instituição e do dispositivo remoto do aluno. Dessa forma, possuindo acesso à Internet e um dispositivo de baixa capacidade de processamento, o aluno tem condições de executar as mesmas tarefas que um aluno presente na sala de aula, de modo seguro e eficiente. Além disso, o professor consegue definir as configurações dos seus Laboratórios Virtuais de acordo com as suas necessidades sem conhecer os detalhes internos da infraestrutura ou depender de apoio dos administradores de rede.

3. Proposta da solução LABVER

Esta seção apresenta a arquitetura LABVER e discute a sua implementação.

3.1. Visão geral da arquitetura

A Figura 1 detalha os componentes principais da arquitetura LABVER: Portal, Orquestrador e Infraestrutura. A ideia central da proposta é permitir que, ao requisitar alocação de recursos de nuvem, o professor seja capaz de definir políticas de alto nível, que serão

[6] `https://opennebula.io/`

traduzidas de forma transparente em uma cadeia de VNFs pelas quais o fluxo do usuário é forçado a passar para aplicação das políticas.

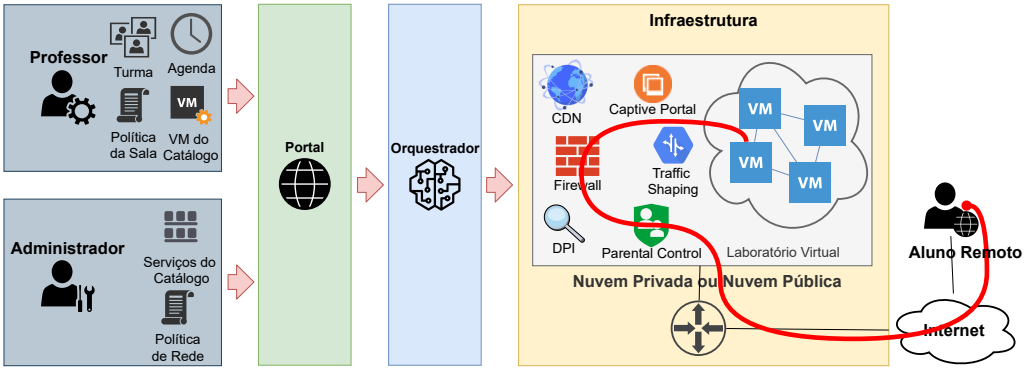


Figura 1. Arquitetura LABVER.

Os usuários interagem com o Portal, que é uma camada especializada na criação de laboratórios virtuais, que fornece uma linguagem de alto nível, específica para esse domínio de aplicação. Na literatura, essa solução é conhecida como DSL (Domain-Specific Language) e abstrai toda a complexidade necessária para criação de um ambiente composto por várias VMs e SFCs. Caso seja necessário adaptar a arquitetura LABVER para outros propósitos, uma nova DSL deve ser especificada, juntamente com o mapeamento dessa para as ações de mais baixo nível, proporcionando um certo grau de desacoplamento entre os requisitos de usuário e o núcleo da solução.

Esse Portal é especializado para cada tipo de usuário (professor, administrador e aluno). A interface para os administradores do sistema permite que os mesmos possam definir as políticas de rede e o catálogo de serviços. Já para o professor é disponibilizada uma interface intuitiva, onde o mesmo poderá criar laboratórios virtuais para seus alunos, sem que seja preciso realizar configurações mais avançadas, que comumente são necessárias para criação de ambientes complexos em nuvem. Por fim, os alunos contam com uma interface gráfica para acessar os recursos disponibilizados nos laboratórios virtuais. Assim, é possível executar todos os procedimentos previstos para a aula, com recursos de processamento, armazenamento e rede compatíveis, como se o aluno estivesse acessando um computador de dentro da rede da instituição.

O Portal envia os dados para o Orquestrador, que faz o tratamento das informações recebidas e as traduz em comandos de criação e configuração das cadeias de VNFs no ambiente de nuvem. Além disso, toda infraestrutura lógica da rede é configurada, incluindo as regras para acesso das máquinas virtuais à Internet ou para outras redes internas, de acordo com as políticas de alto nível definidas anteriormente. Assim, é possível traduzir políticas de alto nível automaticamente em rotinas de alocação de SFC e de recursos virtuais para criação de laboratórios virtuais. O módulo Orquestrador é composto por vários módulos que serão detalhados nas próximas seções.

A Figura 1 também ilustra com uma linha vermelha como o SFC é disposto e faz uso das VNFs previamente escolhidas para estabelecer o caminho ao qual o fluxo de dados entre a comunicação do aluno remoto e o recurso virtual dentro da nuvem irá seguir. A arquitetura LABVER foi projetada para suportar qualquer plataforma de nuvem, desde que seja possível desenvolver uma API (*Application Programming Interface)* para instan-

ciação, configuração e monitoramento de VNFs. Além disso, a plataforma de nuvem deve oferecer algum mecanismo que permita redirecionar um fluxo de rede por uma cadeia de funções que compõem o serviço. A possibilidade de gerenciar VNFs em qualquer nuvem pública ou privada torna a solução flexível para gerenciar recursos heterogêneos.

3.2. Implementação da arquitetura

Como prova de conceito da arquitetura apresentada na seção anterior LABVER, foi implementado um protótipo com tecnologias de código aberto. A Figura 2 mostra a proposta dessa implementação, composta por 3 camadas: usuário, orquestração e infraestrutura.

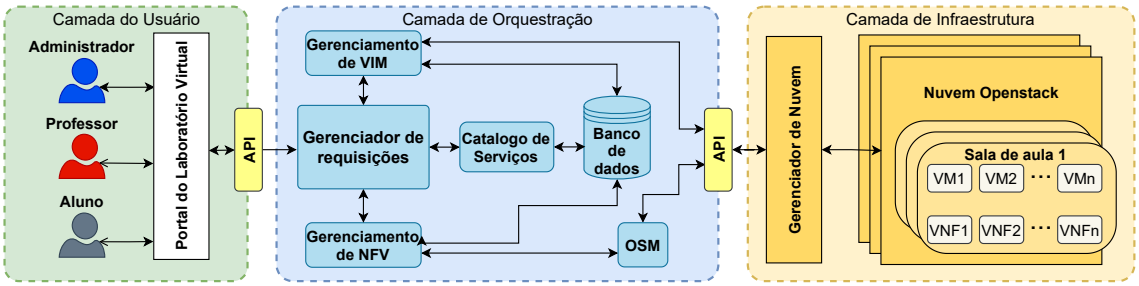


Figura 2. Implementação da arquitetura LABVER.

A camada do usuário contempla as interfaces de acesso, a gestão da arquitetura, e a gestão das requisições dos usuários por meio do Portal de acesso. Nessa implementação, há 3 níveis de permissão de acesso: professor, aluno e administrador. A interação da Camada do Usuário com a de Orquestração é feita através de uma API escrita em Python que envia ao gerenciador de requisições as configurações definidas pelo professor ou administrador em formato JSON[7].

A camada de Orquestração é composta por 6 módulos: Gerenciamento de Requisições, Banco de Dados, Catálogo de Serviços e VMs, Gerenciamento de VIM, Gerenciamento de NFV e OSM. No módulo de Gerenciamento de Requisições, é responsável pelo mapeamento entre a DSL oferecida pelo Portal e as ações do nível de infraestrutura virtualizada, processa as requisições e as direciona aos devidos módulos. No Banco de Dados são armazenados dados de autenticação de serviços e usuários, além de armazenar logs dos serviços, como horário de criação de usuário, exclusão e criação de projetos, entre outros. No Catálogo de Serviços e VMs, são disponibilizados os templates para as VNFs e VMs, que são compostos por imagens previamente preparadas.

Para implementação da arquitetura, utilizou-se o OpenStack (versão Victoria) como ferramenta de gerenciamento de infraestrutura virtual (VIM, do inglês *Virtualized Infrastructure Manager*), que é robusta e capaz de suportar diversos níveis de customização. Adotamos também o Open Source MANO (OSM), versão 8, que desenvolve uma pilha de gerenciamento e orquestração alinhada com os modelos de informação ETSI (*European Telecommunications Standards Institute*). Ambas as plataformas possuem código aberto e são amplamente utilizadas pelo mercado.

O módulo Gerenciamento de VIM garante que as configurações de usuário, projetos e *templates* de VMs estejam prontas previamente à utilização do OSM. Ele contém

[7]https://www.json.org/json-en.html

funções como criar, deletar e atualizar dados de usuário e de projetos na nuvem, além de funcionalidades de criação de redes e de grupos de segurança. Este módulo também é responsável por permitir que os administradores criem e gerenciem na nuvem os *templates* de VMs e VNFs por meio da interação como o módulo do Catálogo de Serviços e VMs. Na implementação, foi utilizado o `openstackSDK`[8], uma biblioteca para construir aplicativos para trabalhar com nuvens OpenStack.

O módulo de Gerenciamento de NFV tem a responsabilidade de receber as requisições traduzidas e transcrevê-las em descritores das funções de rede (VNFD, do inglês *Virtual Network Function Descriptor*) e serviços de rede (NSD, do inglês, *Network Service Descriptor*), que são baseados no modelo de informação ETSI NFV SOL006[9] e serão consumidos pelo OSM. Devido a sua interface unificada, baseada nos padrões ESTI NFV SOL005[10], o OSM garante uma robusta abstração que permite a manipulação do NS (*Network Service*) da VIM, sem que haja necessidade de exposição dos elementos de mais baixo nível.

É importante pontuar que as VNFs disponíveis no sistema são previamente cadastradas pelos administradores do sistema, com todas as informações necessárias para que as mesmas possam compor o encaminhamento de funções de rede, como: (i) recursos a serem instanciados, (ii) qual é a interface de rede que irá receber os dados, (iii) qual é a interface de rede que irá enviar os dados.

No escopo deste trabalho, o OSM usa um plugin especializado para interagir com a API `networking-sfc` do Openstack. Esta API se comunica com os agentes dos *switches* virtuais OVS (*Open Virtual Switch*) dos servidores dos nós de computação para gerar as entradas nas tabelas de fluxo de acordo com o protocolo *Openflow*, que irão forçar os fluxos a passarem pela cadeia ordenada de VNFs antes de atingir as VMs dos alunos. Assim, o SFC pode ser fornecido com escalabilidade em ambientes multi-nuvem em altas taxas de vazão, uma vez que o OVS é um *switch* virtual baseado em software de alto desempenho e amplamente utilizado em SDN.

3.2.1. Criação de um Laboratório Virtual

No protótipo, cada professor é um usuário do OpenStack e cada Laboratório Virtual é um projeto no OpenStack de propriedade do professor. Dessa forma, são fornecidos ambientes isolados e seguros para hospedagem das VNFs e VMs de cada laboratório virtual, que são logicamente isolados uns das outros, permitindo que seja associada uma quota de limite de recursos máximos que podem ser utilizados. Para cada Laboratório Virtual, é necessário instanciar na infraestrutura de nuvem: (i) um projeto, (ii) uma rede virtual, (iii) um conjunto de VMs para os alunos, (iv) um conjunto de VNFs e (v) um conjunto de regras de fluxo que serão utilizadas para montar os SFCs.

O Gerenciador de Requisições foi desenvolvido para tratar as informações provenientes do Portal, como realizar a autenticação dos usuários e compilar os dados da

[8] `https://pypi.org/project/openstacksdk`

[9] `https://www.etsi.org/deliver/etsi_gs/NFV-SOL/001_099/006/02.06.01_60/gs_NFV-SOL006v020601p.pdf`

[10] `https://www.etsi.org/deliver/etsi_gs/NFV-SOL/001_099/005/02.04.01_60/gs_NFV-SOL005v020401p.pdf`

criação do laboratório de modo a escalonar as etapas em consonância com os demais módulos. O diagrama de sequência definido na Figura 3 mostra a sequência de passos envolvida na criação de um Laboratório Virtual.

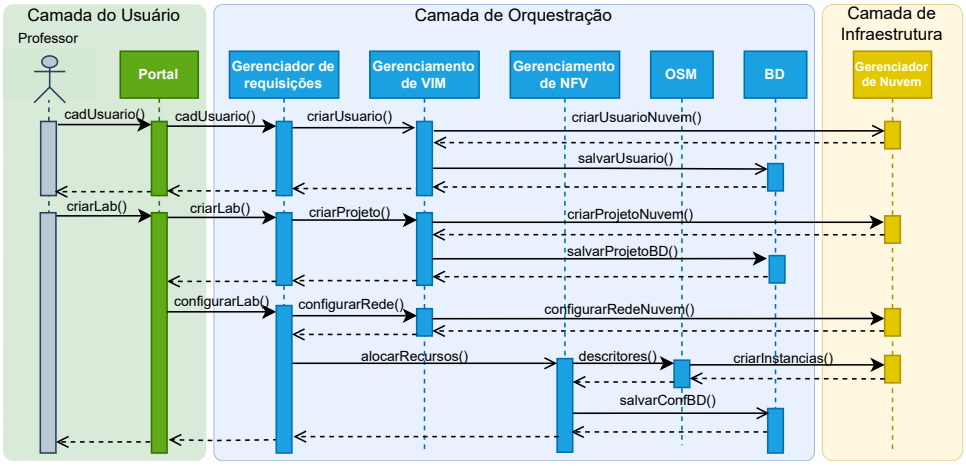


Figura 3. Diagrama de sequência para professor criar Laboratório Virtual.

No primeiro passo mostrado no diagrama, quando um professor se cadastra no Portal, o Gerenciador de Requisições aciona o Gerenciamento de VIM para enviar a requisição de criação do usuário para a nuvem. Este passo é finalizado com o salvamento das informações de acesso do usuário no Banco de Dados, incluindo o respectivo identificador de usuário na nuvem. Essas informações serão utilizadas nas próximas vezes que o professor efetuar o login no Portal.

Estando logado, o professor pode criar Laboratórios Virtuais para suas turmas de acordo com as suas necessidades (e.g., número de VMs, imagem do sistema operacional e configurações desejadas). Ao receber a requisição, o Gerenciador de Requisições aciona o módulo Gerenciamento de VIM para criação de um projeto exclusivo para o laboratório na VIM (Openstack), garantindo que os recursos disponibilizados não possam ser acessados por outros usuários que não tenham permissões. Depois outra requisição é enviada ao Gerenciamento de VIM para preparação da infraestrutura de rede virtual que será necessária para instanciar as VMs, garantindo que apenas os fluxos desejados possam ser transmitidos entre todos os elementos do Laboratório Virtual. Essas configurações são aplicadas diretamente nos grupos de segurança[11] fornecidos pelo *OpenStack*.

Depois, o Gerenciador de Requisições processa os dados da criação do Laboratório Virtual (e.g. número de VMs, imagens para as VMs dos alunos, VNFs utilizadas para aplicação das políticas e ordem do SFC) e os envia para o Gerenciamento de NFV.

Então, o módulo de Gerenciamento de NFV traduz as requisições recebidas em VNFD e dos NSD. Assim, é possível criar SFCs dinamicamente e tornar o transparente para o usuário todo detalhamento necessário para o encadeamento das VNFs. Após montagem dos descritores, os mesmos são enviados ao OSM, que por sua vez faz as requisições para instanciar as VMs, VNFs e SFCs que compõe os serviços de rede dos descritores.

[11] `https://docs.openstack.org/nova/victoria/admin/security-groups.html`

Durante os passos de criação do laboratório, é realizada a persistência das informações na base de dados. Ao final do processo de criação do Laboratório Virtual pelo professor, os alunos da turma poderão acessar as VMs do laboratório por meio de interface gráfica específica, pois os SFCs instaladas garantem que o respectivo fluxo será encaminhado pelas VNFs de aplicação de política antes de dar acesso a VM de destino.

4. Avaliação Experimental

Para validação funcional da arquitetura LABVER, consideramos como cenário para o ambiente de ensino remoto a criação de um Laboratório Virtual sob demanda.

4.1. Testbed e Metodologia

Na avaliação experimental foram utilizados dois servidores físicos PowerEdge T640 com 96 GB RAM e CPU Intel Xeon Silver 4114 de 2.2 GHz. Esses servidores foram utilizados para instalação de uma nuvem OpenStack, configurada em modo multi-nó, sendo um nó controlador (Controller) e um nó de computação (*Compute Node*). Neste último, foram instanciados: o Portal, o orquestrador LABVER, as VMs dos alunos (*VMa*), uma *VNF* de controle parental (*VNF1*) e um servidor web (*WebServer*). Este protótipo ainda não contempla os mecanismos de acesso remoto dos alunos às VMs, que serão abordados em trabalhos futuros. Na Tabela 2 são apresentadas as capacidades alocadas para cada tipo de instância.

Tabela 2. Configuração dos recursos virtuais utilizados no ambiente virtual.

Tipo de Nó	VirtualCPU	VirtualMemory (GB)	VirtualDisk (GB)
Orquestrador LABVER	5	8	100
Servidor Portal	1	2	50
VMa	1	1	4
VNF1	1	1	4
WebServer	1	1	5

O primeiro experimento tem por objetivo demonstrar o funcionamento e avaliar o tempo de execução das operações principais para criação de um Laboratório Virtual, de acordo com o diagrama da Figura 3. Além disso, demonstraremos os mecanismos de aplicação de política de controle de fluxo através de dois experimentos: i) via *VNF* de controle parental; e ii) via grupo de segurança. Ademais, analisaremos o impacto do comprimento do SFC na latência.

Para avaliar o gerenciamento e o controle de fluxo na arquitetura, foi utilizada a ferramenta `iperf3` com taxa de tráfego pré-definida em 500 Mbps durante 100 segundos (t_{100}) gerando um fluxo de dados UDP. Para eliminar o ruído gerado no estabelecimento de caminho fim-a-fim, apresentaremos os gráficos demonstrando as políticas de controle de fluxo desconsiderando os 10 segundos iniciais da transmissão entre origem e destino. Para cada experimento foi calculado o intervalo de confiança de 95% exibindo o valor médio das amostras. O bloqueio parental foi realizado a partir de t_{20}. Para medir a latência, utilizamos o RTT (*Round Trip Time*) dos pacotes ICMP (ping) com MTU de 1500 bytes. Para todos os cenários em que avaliamos a latência, a VMa (origem) enviou 200 pacotes (1 por segundo) para o WebServer (destino), variando apenas o tamanho do SFC.

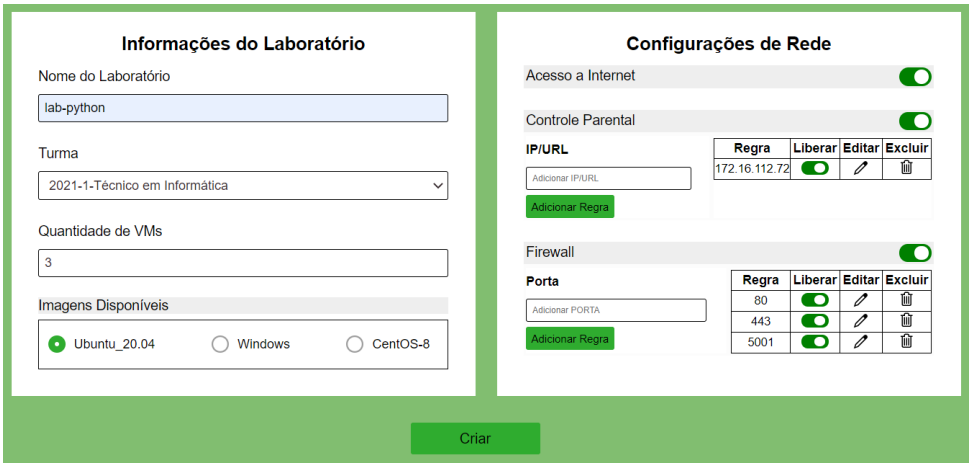


Figura 4. Protótipo da interface de criação do laboratório virtual.

4.2. Demonstração da criação de um Laboratório Virtual sob demanda

Para ilustrar o funcionamento, considerou-se a criação de um Laboratório Virtual de programação, com características especificadas pelo docente. Neste laboratório os participantes utilizarão a linguagem Python e terão acesso ao repositório dos códigos usados na aula. Para configurar o ambiente, o professor deve acessar o portal, escolher a funcionalidade "criar laboratório virtual" e interagir com a interface ilustrada na Figura 4.

Ainda na Figura 4, é possível observar que foram especificados os seguintes parâmetros, nome do laboratório: *lab-python*, imagem escolhida: $Ubuntu20.04$, quantidade de instâncias: 3, acesso à internet: Sim, função de rede: $controle\ parental$ e $firewall$, IPs liberados: $172.16.112.72$ e Portas liberadas: $80, 443, 5001$.

Após confirmar os parâmetros para criação do Laboratório Virtual, o Portal irá submeter a requisição ao Orquestrador LABVER. Conforme descrito na seção 3.2.1, o Gerenciador NFV irá gerar o descritor em formato YAML, apresentado na Figura 5. Este arquivo posteriormente será encaminhado para o OSM, que vai interagir com o OpenStack para criação da infraestrutura virtual que será utilizada no laboratório.

Para o protótipo, implementamos duas funções de rede: controle parental e firewall. O controle parental foi implementado usando uma VNF com a distribuição Ubuntu-cloud-server-20.04, que possui ferramentas nativas para encaminhamento e bloqueio de fluxo. Para o Firewall, fizemos o uso dos grupos de segurança, que é um recurso nativo do VIM OpenStack. Neste método, é possível aplicar em todas as instâncias do projeto um conjunto de regras de filtro de IP e porta que determinam os acessos das instâncias a rede a partir do *switch* virtual da rede do projeto. Como trabalho futuros, planejamos implementar outras opções de VNFs, que poderão ser configuradas no Portal.

Na Figura 6 é ilustrada a infraestrutura resultante da solicitação realizada no Portal pelo professor. A Figura 6a apresenta a rede do $LabPython$ (cor laranja) com 3 máquinas virtuais (*VMa*, *VMa2* e *VMa3*), a função de rede (*VNF1*), referente ao controle parental, bem como a rede para acesso externo da infraestrutura em azul. A Figura 6b demonstra a composição do SFC, o elemento identificado como *router-labsfc* se refere ao roteador que permite o acesso para uma rede externa. Previamente à criação do laboratório, foi criada uma VM de um servidor web, identificado como *WebServer*, que será usado para

```
lab-name : lab-python
vm-image : ubuntu_20.04
vm-qnt   : 3
internet : True
vnf :
    - vnf-name : controle-parental
      ip :
        - allow : 172.16.112.72
          deny  : all
vnf :
    - vnf-name : firewall
      port :
        - allow : 80, 443, 5001
          deny  : all
```

Figura 5. Arquivo de configuração em formato YAML.

os experimentos para controle de fluxo de dados.

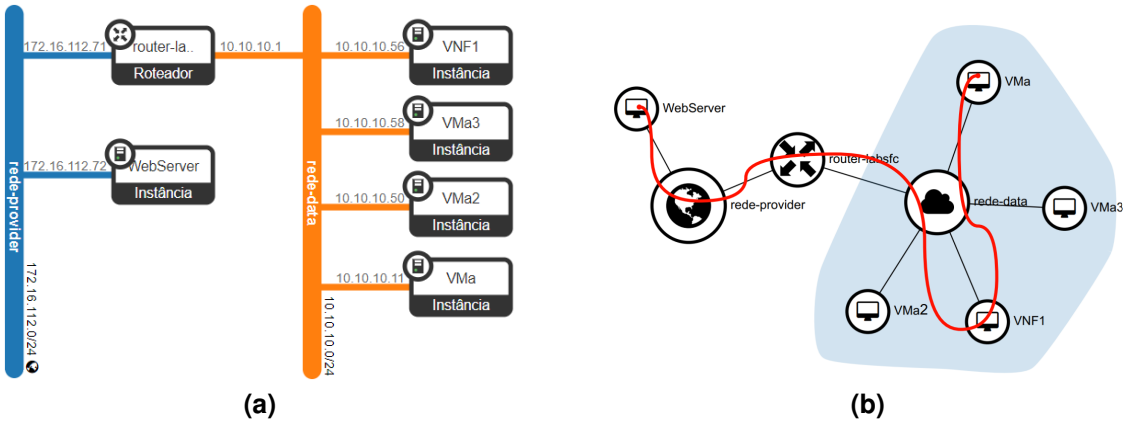


Figura 6. Visão da topologia após criação do laboratório virtual.

Na Figura 7 ilustramos o período de tempo em segundos, necessário para execução das principais operações usadas no LABVER, de acordo com o diagrama da Figura 3. Este teste compreende os métodos iniciais, do ponto de vista do Gerenciador de Requisições, para criar um usuário ($\pm$ 238 ms), criar um projeto ($\pm$ 1710 ms), configurar a rede ($\pm$ 430 ms), até alocar os recursos e tornar o ambiente acessível aos usuários, com todas as instâncias em execução (28 segundos com variação aproximada de $\pm$ 2 segundos). Esta variação no tempo para alocação ocorre devido à comunicação com a nuvem e completa instanciação das máquinas virtuais. Para simplificar a visualização no gráfico os valores do eixo Y entre 3 e 25 foram suprimidos, considerando que o valor absoluto da operação de alocação é bem maior que das outras operações.

4.3. Demonstração do Controle de fluxo e análise de latência

Para demonstrar o controle de fluxo, vamos considerar o primeiro cenário (vide Figura 6b), denominado: Controle de fluxo de dados via VNF. O fluxo de dados foi gerado entre a *VMa* (origem) e o *WebServer* (destino), passando pela função *VNF1* de controle parental. Neste cenário, a *VNF1* possui apenas uma interface de rede para entrada e saída e, portanto, contabiliza a vazão recebida de forma duplicada quando o fluxo passa por ela em direção ao destino.

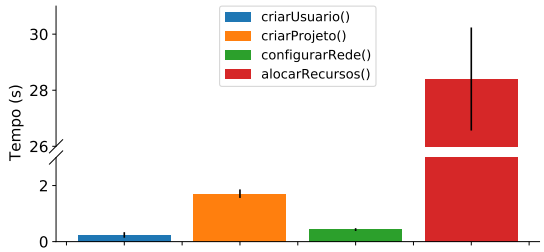
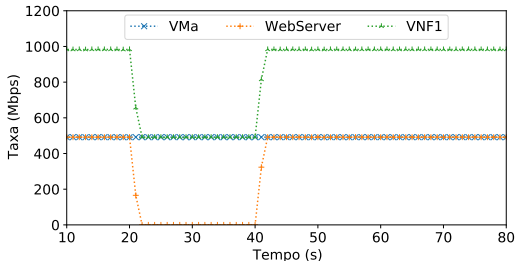
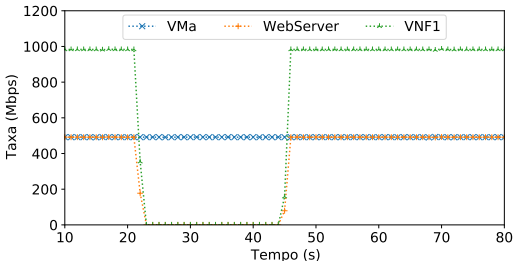


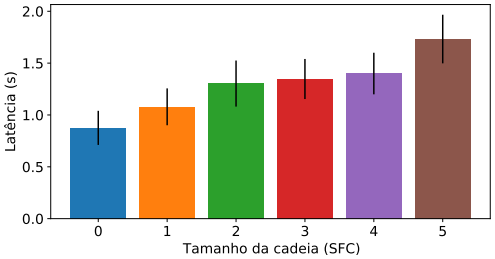
Figura 7. Tempo para execução das principais operações da LABVER.



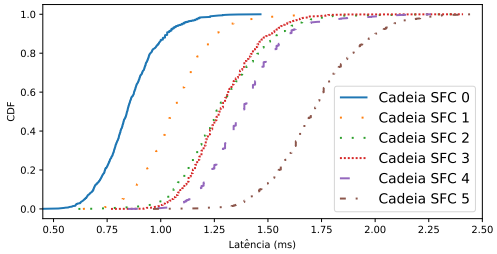
(a) Bloqueio por VNF de controle parental.



(b) Bloqueio por grupos de segurança.



(c) Latência por tamanho do SFC.



(d) CDF da latência por tamanho do SFC.

Figura 8. Demonstração do controle de fluxo e análise de latência na arquitetura.

Utilizando nossa arquitetura, 20 segundos (t_{20}) após o início do tráfego a função de bloqueio de fluxo de dados foi ativada, entre a *VNF1* e o *WebServer* com duração de 20 segundos. Após o instante t_{40}, o bloqueio foi desativado. O resultado foi ilustrado na Figura 8a. Note que o fluxo de dados UDP (porta 5001) continua sendo enviado a partir da interface da *VMa* (origem), mantendo-se constante após t_{10}. Como o controle foi aplicado na interface *VNF1*, todo fluxo entre a *VNF1* e o *WebServer* foi bloqueado, logo o fluxo na interface de rede da *VNF1* é reduzido pela metade. O tempo médio para aplicar a regra de bloqueio observada nos experimentos com o ambiente em execução foi de cerca de 2 segundos. Isso ocorre devido a latência de comunicação entre os recursos computacionais utilizados na infraestrutura virtual.

No segundo cenário utilizamos a mesma metodologia, mas usando os grupos de segurança da VIM (firewall nativo) para controle de fluxo de dados. Os resultados são apresentados na Figura 8b. Observe que, utilizando este método, todo tráfego entre *WebServer* $\rightarrow$ *VNF1* e *VNF1* $\rightarrow$ *VMa* é bloqueado. Apenas a *VMa* (origem) continua gerando o tráfego durante o período do experimento. Os grupos de segurança permitem executar o bloqueio diretamente nas interfaces do *switch virtual rede-data*, mas o tempo para aplicar

a regra sofre um atraso médio de 3 segundos se o ambiente estiver em execução. Considerando que o laboratório virtual é criado com as políticas já definidas, este atraso não será notado pelos usuários.

Por último, demonstramos o impacto na latência dos fluxos de dados em função do comprimento do SFC, considerando que o fluxo de dados seja encadeado por várias VNFs entre origem e destino. Os resultados mostram que o tamanho deste SFC pode aumentar sem comprometer o sistema. Os resultados são apresentados na Figuras 8d e 8c. Conforme ilustrado na Figura 8c a latência cresce linearmente de cerca 0.20 segundos, com variação de $\pm$ 0.15s, até cerca de 1.80 segundos, com variação de $\pm$ 0.21s quando o tamanho da cadeia varia de 0 (nenhuma VNF) a 5 (5 VNFs). Na Figura 8d, considerando 90% de percentil é possível observar que SFC 0 leva $\pm$ 0.87 ms, SFC 1 $\pm$ 1.10 ms, SFC 2 $\pm$ 1.25 ms, SFC 3 $\pm$ 1.35 ms, SFC 4 $\pm$ 1.40 ms e SFC 5 $\pm$ 1.75 ms. Portanto, considerando o limite inferior 0.87 ms, sem nenhum SFC, pode-se concluir que cada SFC incluída na infraestrutura vai representar um acréscimo em termos de latência média de $\pm$ 0.22 ms em nossa arquitetura.

5. Conclusão

Este trabalho apresentou a proposta, implementação e avaliação de uma arquitetura de baixo custo baseada em NFV e SDN para o fornecimento de Laboratórios Virtuais. O LABVER foi projetado para permitir a implementação de políticas de controle em alto nível, que são traduzidas em comandos de criação de cadeias SFCs, instanciação de VMs e configuração dinâmica de toda infraestrutura virtual de rede. Além disso, é totalmente baseada em *software* livre e aderente a infraestrutura legada das instituições de ensino.

A partir dos experimentos realizados foi possível visualizar o funcionamento da arquitetura, com a criação de uma infraestrutura complexa de nuvem sem exigir do professor conhecimentos técnicos especializados. Para demonstrar a flexibilidade ofertada na arquitetura, políticas de controle de fluxo via *VNF* de controle parental e via grupo de segurança foram exibidas. Com o objetivo de avaliar cenários mais realistas, medimos o impacto na latência dos fluxos de dados em função do comprimento do SFC, considerando que o fluxo pode passar por várias VNFs entre origem e destino.

Como trabalhos futuros, será explorado a utilização de outras VIMs, a extensão dos testes de desempenho e escalabilidade, a utilização da arquitetura em ambientes multi-nuvem. Também consideramos a implementação da interface de aluno com VDI e da interface de administrador. A ampliação do catálogo de serviço com novas funções de rede, também é objeto dos trabalhos futuros que serão desenvolvidos neste projeto.

Agradecimentos

Agradecemos ao IFES pelo apoio financeiro via Programa Institucional de Apoio à Pós-graduação Stricto Sensu (Propós) e Programa Pibic, e à FAPES pelo suporte ao laboratório CIDIG do Centro de Pesquisa Inovação e Desenvolvimento do Espírito Santo.

Referências

Alagappan, A. et al. (2016). Virtual desktop infrastructure for rendering education technology in multifaceted learning platforms — a case study at botho university. In *2016 SCOPES*, pages 1717–1720.

Bari, F. et al. (2016). Orchestrating virtualized network functions. *IEEE Transactions on Network and Service Management*, 13(4):725–739.

Bhatia, G. and Al Sulti, I. H. (2019). Cascloud: An open source private cloud for higher education. In *2019 International Arab Conference on Information Technology (ACIT)*, pages 14–20. IEEE.

Calle-Romero et al. (2019). Virtual desktop infrastructure (vdi) deployment using opennebula as a private cloud. In *International Conference on Applied Technologies*, pages 440–450. Springer.

Chu et al. (2019). Nextlab: A new hybrid testbed and development platform for software-defined networking. In *Proceedings of the Tenth ISICT*, pages 186–190.

Farhady, H., Lee, H., and Nakao, A. (2015). Software-defined networking: A survey. *Computer Networks*, 81:79–95.

Gómez-Zermeño, M. G. (2020). Massive open online courses as a digital learning strategy of education for sustainable development. *Journal of Sustainable Development of Energy, Water and Environment Systems*, 8(3):577–589.

Heradio et al. (2016). Virtual and remote labs in education: A bibliometric analysis. *Computers & Education*, 98:14–38.

Kabiri et al. (2017). An experimental evaluation of a cloud-based virtual computer laboratory using openstack. In *2017 6th IIAI-AAI*, pages 667–672. IEEE.

Mahmoodi, T., Van Helvoort, H., and Mansfield, S. (2017). Management and orchestration. *IEEE Communications Standards Magazine*, 1(1):90–91.

Makoviy et al. (2017). Server hardware resources optimization for virtual desktop infrastructure implementation. In *CEUR Workshop Proceedings*, volume 1904, page 178.

Medhat et al. (2016). Service function chaining in next generation networks: State of the art and research challenges. *IEEE Communications Magazine*, 55(2):216–223.

Nakhai, P. H. and Anuar, N. B. (2017). Performance evaluation of virtual desktop operating systems in virtual desktop infrastructure. In *2017 IEEE Conference on Application, Information and Network Security (AINS)*, pages 105–110. IEEE.

Owens, K., Suskin, M., and Wiesemann, C. (2019). Bringing campus experience to online classes. In *Proceedings of the 2019 ACM SIGUCCS Annual Conference*, pages 61–63.

Rahman, U. et al. (2016). Performance evaluation of vdi environment. In *2016 Sixth International Conference on Innovative Computing Technology (INTECH)*, pages 104–109. IEEE.

Rot et al. (2018). Benefits, limitations and costs of it infrastructure virtualization in the academic environment. case study using vdi technology. In *ICSOFT*, pages 738–745.

Shirzad, M. et al. (2012). E-learning based on cloud computing. In *2012 International Conference on Cloud Computing Technologies, Applications and Management (ICCCTAM)*, pages 214–218.

Xu, Z., Yang, L., and Lei, J. (2015). Conception and design of desktop virtualization cloud platform for primary education: Based on the citrix technology. In *2015 International Conference of Educational Innovation through Technology (EITT)*, pages 226–230.