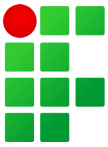


INSTITUTO FEDERAL DE EDUCAÇÃO, CIÊNCIA
E TECNOLOGIA DO AMAPÁ – IFAP
CAMPUS MACAPÁ
TECNOLOGIA EM REDES DE COMPUTADORES

IACY RIBAMAR GONÇALVES DE ALCÂNTARA JÚNIOR
RAILSON DOS PRAZERES PICANÇO

A UTILIZAÇÃO DE REDES DEFINIDA POR SOFTWARE: aplicações práticas do protocolo openflow

MACAPÁ – AP
2021

IACY RIBAMAR GONÇALVES DE ALCÂNTARA JÚNIOR
RAILSON DOS PRAZERES PICANÇO

A UTILIZAÇÃO DE REDES DEFINIDA POR SOFTWARE: aplicações práticas do protocolo openflow

Trabalho de Conclusão de Curso apresentado ao curso Superior de Tecnologia em Redes de Computadores, do Instituto Federal de Educação, Ciência e Tecnologia do Amapá – Ifap, como requisito avaliativo para obtenção de título de Tecnólogo em Redes de Computadores.
Orientador: Prof. Me. Célio do Nascimento Rodrigues.

MACAPÁ – AP
2021

Biblioteca Institucional - IFAP
Dados Internacionais de Catalogação na Publicação (CIP)

A347u Alcântara Júnior, Iacy Ribamar Gonçalves
A utilização de redes definida por software: aplicações práticas do protocolo openflow / Iacy Ribamar Gonçalves Alcântara Júnior, Railson dos Prazeres Picanço. - Macapá, 2021.
67 f.

Trabalho de Conclusão de Curso (Graduação) -- Instituto Federal de Educação, Ciência e Tecnologia do Amapá, Campus Macapá, Curso de Tecnologia em Redes de Computadores, 2021.

Orientador: Célio do Nascimento Rodrigues.

1. Rede definida por software. 2. Controlador ryu. 3. Openflow. I. Picanço, Railson dos Prazeres. I. Rodrigues, Célio do Nascimento, orient. II. Título.

Elaborada pelo Sistema de Geração Automática de Ficha Catalográfica do IFAP
com os dados fornecidos pelo(a) autor(a).

IACY RIBAMAR GONÇALVES DE ALCÂNTARA JÚNIOR
RAILSON DOS PRAZERES PICANÇO

A UTILIZAÇÃO DE REDES DEFINIDA POR SOFTWARE: aplicações práticas do protocolo openflow

Trabalho de Conclusão de Curso apresentado ao curso Superior de Tecnologia em Redes de Computadores, do Instituto Federal de Educação, Ciência e Tecnologia do Amapá – Ifap, como requisito avaliativo para obtenção de título de Tecnólogo em Redes de Computadores.
Orientador: Prof. Me. Célio do Nascimento Rodrigues.

BANCA EXAMINADORA

Prof. Me. Célio do Nascimento Rodrigues

Prof. Esp. Eonay Barbosa Gurjão

Prof. Doutor Klenilmar Lopes Dias

Aprovado(a) em: 09/06/2021

Nota: 9,0

RESUMO

Este Trabalho de Conclusão de Curso teve como objetivo emular uma rede definida por software e apresentar, por meio de referencial bibliográfico, abordar os conceitos, estrutura, composição e aplicações no ambiente local. Propôs-se utilizar emulador Mininet, OpenvSwitch, integrados ao controlador Ryu para implantação, de uma serie de ambiente de testes virtualizado pelo VirtualBox, com topologias diversificada, para demonstrar o gerenciamento do fluxo de dados por meio de um switch com protocolo OpenFlow.

Palavras chave: Rede definida por software. Controlador ryu. Openflow.

ABSTRACT

This Course Completion Work aimed to emulate a software-defined network and present, through a bibliographic reference, addressing the concepts, structure, composition and applications in the local environment. It was proposed to use a Mininet emulator, OpenvSwitch, integrated to the Ryu controller for deployment, of a series of test environment virtualized by VirtualBox, with diversified topologies, to demonstrate the management of the data flow through a switch with OpenFlow protocol.

Keywords: Software defined network. Ryu controller. Openflow.

LISTA DE FIGURAS

Figura 1 – Arquitetura - modelo atual e o modelo SDN 18
Figura 2 – Modelo de operação híbrido para tráfegos padrão e OpenFlow 28
Figura 3 – Executando o Mininet 35
Figura 4 – Topologia IP/SDN 36
Figura 5 – Exemplo de processo executando no namespace da rede 38
Figura 6 – Verificando interface com “ifconfig” 40
Figura 7 – Teste de conectividade 41
Figura 8 – Mininet - Teste de conectividade 42
Figura 9 – Carregando as configurações do controlador. 43
Figura 10 – Etapas de carregamento 44
Figura 11 – Visualização do OVS via terminal. 45
Figura 12 – Tabela de fluxo do OVS 48
Figura 13 – Host1 em teste de conectividade 49
Figura 14 – Captura de tráfego com SDN inativa 50
Figura 15 – Requisições TCP entre Ryu e Mininet 51
Figura 16 – Protocolo OpenFlow: Mensagem Hello 52
Figura 17 – Protocolo OpenFlow: Mensagem “Feature_Reply” 53
Figura 18 – Protocolo OpenFlow: Mensagem Flow_Mod 54
Figura 19 – Topologia SDN para testes de desempenho 55
Figura 20 – Teste de desempenho no modo usuário 56
Figura 21 – Teste de desempenho em espaço kernel 57

LISTA DE TABELAS

Tabela 1 – Cabeçalho padrão de mensagens OpenFlow 23
Tabela 2 – Campos de classificação dos comutadores OpenFlow Tipo 0 24
Tabela 3 – Campos de classificação de comutadores OpenFlow v1.1 26
Tabela 4 – Endereçamento IP da rede 37

LISTA DE ABREVIATURAS E SIGLAS

ACK	Acknowledgement (confirmação ou recebimento de mensagem)
API	Application Programming Interface (Interface de Programação de Aplicação)
ARP	Address Resolution Protocol (Protocolo de Resolução de Endereço)
DHCP	Datapath management utility (utilitário que envia mensagens OpenFlow ao switch)
ICMP	Internet Control Message Protocol (Protocolo de Mensagens de Controle de Internet)
IP	Internet Protocol address (Endereço de Protocolo da Internet)
IPv4	Internet Protocol version 4 (Protocolo de Internet versão 4)
MAC	Media Access Control (Controle de Acesso ao Meio)
MPLS	Multi Protocol Label Switching (mecanismo em redes de telecomunicações de alto desempenho)
OVS	Open vSwicth
PC	Plano de Controle
PD	Plano de dados
ROM	Read Only Memory (Memória Apenas para Leitura)
SD-WAN	Wide Area Network (Rede de Área Ampla)
SDN	Software Defined Networking (Rede Definida por Software)
SNMP	Simple Network Management Protocol (Protocolo Simples de Gerência de Rede)
SSL	Secure Socket Layer (Camada de Soquete Seguro)
SYN	Synchronize (Sincronizar)
TCAM	Ternary Content Addressable Memory (Memória de Endereçamento de Conteúdo Ternário)
TCP/IP	Transmission Control Protocol (protocolo de controle de transmissão)
VLAN	Virtual Local Area Network (Redes Locais Virtualizadas)

SUMÁRIO

1	**INTRODUÇÃO**	11
2	**PROBLEMA DA PESQUISA**	13
3	**JUSTIFICATIVA**	14
4	**OBJETIVOS**	15
4.1	**Geral**	15
4.2	**Específicos**	15
5	**REFERENCIAL TEÓRICO**	16
5.1	**Redes definidas por software**	16
5.1.1	Organização	16
5.1.2	Comutadores e as tabelas de fluxos	19
5.1.3	Conexão segura	19
5.1.4	Controlador	20
5.2	**O protocolo openflow**	22
5.2.1	Características	22
5.2.2	Funcionamento	26
5.3	**Métodos e aplicações**	28
6	**PROCEDIMENTOS METODOLÓGICOS**	32
6.1	**Controlador sdn**	33
6.2	**Emulador mininet**	33
6.3	**Topologia**	35
6.4	**Testes de conectividade**	37
6.5	**Ambiente de trabalho utilizado (ferramentas)**	39
6.5.1	Iniciando os testes	39
6.5.2	Emulando switch openflow	42
6.5.3	Analisando os fluxos com dpctl	47
6.6	**Analisando o protocolo openflow**	49
6.7	**Testes de desempenho**	54
7	**CONCLUSÃO**	58
	REFERENCIAS BIBLIOGRÁFICAS	60
	APÊNDICE A - PREPARANDO O SISTEMA OPERACIONAL	62
	APÊNDICE B - INSTALAÇÃO DO MININET	66

APÊNDICE C - INSTALAÇÃO DO CONTROLADOR RYU 67

1 INTRODUÇÃO

Desde a expansão da redes de computadores na década de 90, trouxe em conjunto os avanços tecnológicos da comunicação e internet. No entanto, toda criação necessita de melhoria e aperfeiçoamento para atender as necessidades da sociedade. As redes definidas por software é um dos pontos chave para melhoria e inovação na arquitetura de encaminhamento de pacotes, plano de dados e controle.

A escolha pelo tema está relacionado diretamente a linha de estudo no curso de Tecnologia em Redes de Computadores, unindo o conhecimento de infraestrutura de redes, aplicação lógica programável, com códigos aberto e inteligentes. Neste sentindo as redes definidas por software permite construir um ambiente de produção, fazendo uso da emulação como sendo a ferramenta essencial, engendrando o protocolo OpenFlow.

O objetivo deste Trabalho de Conclusão de Curso foi propor de forma clara e objetiva a aplicação do protocolo OpenFlow, através de um cenário cotidiano, as vantagens de gerenciar os encaminhamentos de pacotes na rede, por meio de software, permitindo a liberdade de configurar e definir a forma de trabalho. Desta forma, podendo trabalhar com diversos sistemas, com segurança da informação e obtendo melhor eficiência das ferramentas disponíveis.

Este trabalho apresenta no Capítulo 2 a problemática da pesquisa em se administrar o tráfego de dados por meio de software, centralizando o gerenciamento de controle de rotas de cada fluxo e melhorando a qualidade dos serviços.

No Capítulo 3 trata-se da justificativa sobre adoção de uma rede definidas por softwares, abordando através dos conceitos do tema sobre as preposições e vantagens de trabalhar com redes com protocolo OpenFlow.

No Capítulo 4 é apresentado os objetivos deste trabalho em que se propõe, analisar o funcionamento das redes definidas por software e do protocolo OpenFlow, posteriormente dissertar sobre os conceitos, emular o ambiente de teste e analisar o desempenho.

No Capítulo 5 abordamos o referencial teórico, trazendo os conceitos e premissas sobre o tema, com as abordagem dos autores, Amorim (2012), Bertholdo (2012), Guedes (2012), López (2014), Nobre (2014) e Rechia (2014).

No Capítulo 6 apresentamos os procedimentos metodológicos a serem adotados para o desenvolvimento do trabalho, realizando as aplicações através de um ambiente virtualizado com ferramentas emulando um cenário real.

No Capítulo 7 finalizamos o trabalho com as considerações finais, com as observações sobre toda a metodologia aplicada para a conclusão final. Por fim destacamos as referencias bibliográficas consultadas para fundamentar e desenvolver todo o trabalho.

2 PROBLEMA DA PESQUISA

As tecnologias de comunicação e tráfego de dados mudam de um paradigma do menor esforço, aonde os protocolos de roteamento localizam rotas mais curtos pelos quais todo o tráfego deve seguir, para que permita que o administrador controle a rota de cada fluxo, atingindo resultados que minimizam despesas e afetam positivamente na qualidade dos serviços.

Assim, é possível haver uma tecnologia definida em software que possibilite a produção por meio de um tráfego de dados centralizado? De que maneira o administrador da rede poderá obter uma visão global e flexível da rede, conjunta a elaboração de soluções redundantes e seguras?

3 JUSTIFICATIVA

As redes definidas por software (SDN – Software Defined Networking), está quebrando barreiras no que diz respeito a inovação e desenvolvimento em redes de computadores. O diferencial deste novo modelo está em sua arquitetura, onde o encaminhamento de pacotes no plano de dados é gerenciando remotamente pelo plano de controle, permitindo uma infraestrutura mais inteligente, aberta, programável e menos suscetível a erros (GUEDES, 2012).

Uma Rede Definida por Software (SDN) e caracterizada pela existência de um sistema de controle (software) que pode controlar o mecanismo de encaminhamento dos elementos de comutação da rede por uma interface de programação bem definida. De forma mais específica, os elementos de comutação exportam uma interface de programação que permite ao software inspecionar, definir e alterar entradas da tabela de roteamento do comutador como ocorre, por exemplo, com comutadores OpenFlow.

O software envolvido, apesar de potencialmente poder ser uma aplicação monolítica especialmente desenvolvida, na prática tende a ser organizado com base em um controlador de aplicação geral, em torno do qual se controle aplicações específicas para a fim de cada rede. E possível, ainda, utilizar-se um divisor de visões para permitir que as aplicações sejam divididas entre diferentes controladores.

Por fim, esta trabalho apresentou um estudo de caso, baseado na virtualização de switches, habilitados ao protocolo OpenFlow, e roteadores convencionais e na implementação de um controlador centralizado, a fim de demonstrar o funcionamento desta tecnologia. Porém, a emulação não alcançará o âmbito de testes de resiliência, utilizando uma arquitetura de alta disponibilidade.

4 OBJETIVOS

4.1 Geral

Analisar o funcionamento das redes definidas por Software e do protocolo OpenFlow.

4.2 Específicos

- Dissertar a respeito dos conceitos de redes definidas por software e do protocolo OpenFlow;
- Emular um ambiente de testes híbrido, baseado na virtualização de switches OpenFlow conectados a um controlador centralizado, por meio de uma rede local virtualizada;
- Permitir que o gerenciamento de tráfego (fluxos) seja implementado por meio de software;
- Analisar a comunicação entre controlador e comutador através da captura de pacotes.

5 REFERENCIAL TEÓRICO

Ao confrontar-se com problema do engendramento da internet, assinala Guedes (2012), o grupo de pesquisa tem debruçado esforços para implantação de redes com melhores recursos para programação. A ação de maior ocorrência, constituiu com a acepção do protocolo OpenFlow, um padrão de código aberto, que tem como fundamental escopo a separação dos planos de controle e de dados, consentindo a intervenção de tráfegos de dados em comparação com fluxos experimentais (AMORIM, 2012).

5.1 Redes definidas por software

As redes definidas por software originou-se através da arquitetura de redes Ethernet, que definia a forma de se construir políticas de controle de acesso, de forma difundida, a partir de uma estrutura de gerência concentrado, assinala Guedes (2012). Esse protótipo foi testado no Campus da Universidade de Stanford, através de um contíguo de equipamentos conectados com 300 hosts (HARANAS, 2016).

Dentro da arquitetura, cada membro da rede necessitaria sondar o gerenciador que deliberaria com base em um grupo de políticas integrais, como o elemento de encaminhamento precisaria tratar o novo fluxo de dados identificado. Essa deliberação estaria informando diretamente o comutador, programado com regra na tabela de encaminhamento. Esse molde foi posteriormente formalizado por alguns autores na configuração da arquitetura OpenFlow (ROTHERNBERG, 2010).

Essa arquitetura, segundo expõe Guedes (2012), consente que a rede seja controlada de forma extensível por meio de aplicações propagas em software. Ao novo protótipo foi chamado de redes definidas por software.

5.1.1 Organização

Avaliando a arquitetura dos roteadores, é presumível notar que se trata de um modelo constituído em seu interior por dois planos: um combinado pelo sistema

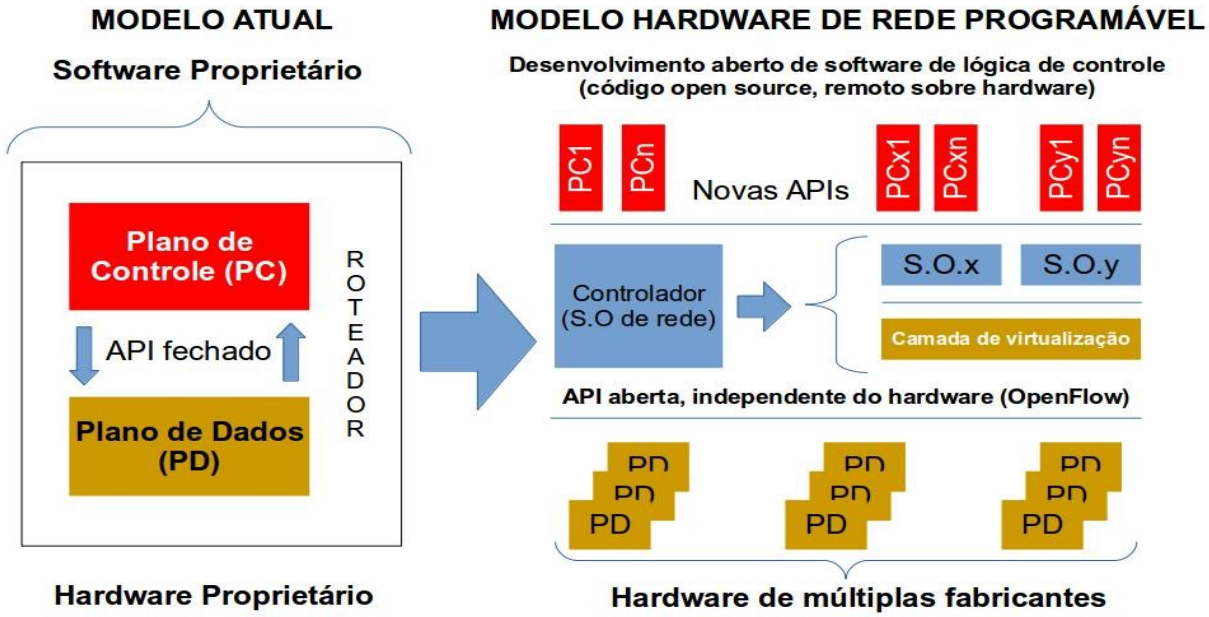
operacional, o software, responsável pelo controle, e outro, o hardware, responsável pela orientação do fluxo de dados (ROTHENBERG, 2010).

O principal, seria incumbido de adotar as deliberações de roteamento que devem ser cumpridas pelo plano de encaminhamento através de uma API proprietária. Sendo que o único intercâmbio da administração com o dispositivo ocorre por meio de interfaces de configuração, restringindo o uso de suas funcionalidades primárias pelo fabricante (LOPEZ, 2014).

É coeso pensar numa arquitetura trajada por dois planos híbridos, não precisam essencialmente estar inseridos em um mesmo dispositivo. Para isso basta que haja uma forma padrão de programar o dispositivo de rede remotamente, transferindo o plano de controle para um dispositivo externo com elevada envergadura de processamento, mantendo a alta performance no encaminhamento desempenhado pelo hardware coligado a flexibilidade de se colocar, remover e particularizar o software por meio de um protocolo aberto. Esta solução alcançada vem através do Protocolo OpenFlow (RECHIA, 2014).

Na figura 1 expõe a arquitetura atual dos roteadores contestada com a arquitetura de equipamentos em uma rede definida por software:

Figura 1 – Arquitetura - modelo atual e o modelo sdn



Fonte: López, 2014.

Segundo López (2014) expõe de forma mais simples que a arquitetura SDN, há um pequeno módulo adicionado a um dispositivo que permite que o sistema externo configure o hardware mais próximo, esse sistema pode ser caracterizado como um controlador.

Sendo assim, é provável que uma rede definida por software é caracterizada através de um controlador externo que pode controlar a composição de orientação dos elementos de comutação da rede por meio de uma interface programável, com elementos de comutação permitindo que o dispositivo remotamente situado a verificação, defina e altere os parâmetros da tabela de encaminhamento. Assim, funciona um comutadores OpenFlow. (RECHIA, 2014).

Por conseguinte, uma rede programável através do OpenFlow incide, necessariamente, em equipamentos de rede certificados para que a conjuntura das tabelas de fluxos possa ser acomodado através de um canal seguro, professando as determinações de um controlador pelo software (GUEDES, 2012).

Na próxima seção abordaremos o princípio básico das redes definidas por software, possibilitando por meio da programação dos elementos de rede,

permitindo a manipulação de pacotes baseados no conceito de fluxos, dos pacotes que trafegam por rotas bem definidas.

5.1.2 Comutadores e as tabelas de fluxos

O fluxo de dados é formado pela combinação de campos do cabeçalho de um pacote a ser processado no dispositivo. Os “Scripts”, conjunto de dados sistemáticos e que não permite modificação após sua criação, são formadas por campos dentro das camadas de enlace, de rede e de transporte, conforme o modelo TCP/IP, conclui Rothenberg (2010).

Em súmula, cada entrada de dados na tabela de fluxos no hardware de rede consiste em haver uma regra, ações e contadores. Essa regra consistir em formar a base da definição do valor de um ou mais campos no cabeçalho do pacote, associado a um conjunto de atuações que decidirão o processamento e encaminhamento de cada pacote. A alimentação estatística de utilização, é registrada nos contadores, que utilizam para excluir fluxos inativos (AMORIM, 2012).

Nessa conjuntura, a menor unidade de informação, no plano de dados da rede, constituiriam as entradas da tabela de fluxos, decodificadas como decisões, pelo hardware, do plano de controle (software), expõe Rothenberg (2010).

É importante destacar que, a devaneio na tabela de fluxos está sujeita a aprimoramentos, para aproveitar o máximo do hardware, permitindo o parâmetro de várias tabelas disponíveis, como as tabelas IP, Ethernet e MPLS (Multiprotocol Label Switching) (NOBRE, 2014).

5.1.3 Conexão segura

Para não haver interferências externas na rede, tais como ataques mal-intencionados, o elemento de comunicação entre a rede e o controlador deve ser seguro, garantindo a credibilidade das informações.

Basicamente, a interface de acesso recomendada é o uso do protocolo SSL (Secure Socket Layer/Camada de Soquete Seguro). Entretanto em ambientes

virtuais e simuladores, as opções de interfaces (passivas ou ativas) podem variar, conforme sua simplicidade de utilização, pois não carecem de chaves criptográficas (GUEDES, 2012).

5.1.4 Controlador

O controlador é o software responsável pela tomada de decisões, em adicionar e remover as cabeçalhos na tabela de fluxo de acordo com o desígnio desejado. Desempenha essa função por meio de uma camada na infraestrutura física promovendo a criação de aplicações e serviços para o gerenciamento da rede (AMORIM, 2012). De acordo com Guedes (2012) a utilização do termo “Sistema Operacional” e definido como sendo um elemento importante na acepção de uma rede definida por software, de grande importância dentro do próprio protocolo OpenFlow.

Em tese, é possível instalar um novo sistema de controle em um dispositivo de rede, sem a obrigação de acrescentar mais hardware. Através da configuração de um novo software no equipamento, com uso de um processador embutido executa o programa de controle a partir do software instalado em uma ROM (Read Only Memory/Memória Apenas para Leitura). No entanto, seria inexecutável, uma vez que muitas implementações de software de controle são específicas para cada hardware, sendo assim, o plano de controle deve ser particularizado para cada dispositivo de hardware (ROTHENBERG, 2010).

O software em questão, segundo Guedes (2012) embora possa ser uma aplicação de maneira especial desenvolvida, de fato tende a ser constituído com base em um controlador para aplicação geral, no qual se arquitetam aplicações específicas para a finalidade na rede. Permitindo, uma divisão nas aplicações entre diferentes controladores.

O principal ponto de uma rede SDNs é a concepção de uma visão centralizada das classes de rede, sendo possível desenvolver análise detalhadas. Entretanto, é importante elucidar que a noção de visão centralizada não estabelece que o controlador opere fisicamente instalado em um único ponto da rede. Está

visão global do sistema, pode ser implementado de forma distribuída, o que permite o desenvolvimento de diversos sistemas interligados, como data center de grandes organizações, garantindo escalabilidade e disponibilidade das SDNs (HARANAS, 2016).

A concentração do controle em uma visão geral, permite a rede a implementação de inovação e serviços diretamente no centro da rede, no entanto a concentração física de controle da rede SDNs pode provocar desafios de segurança bem como prejudicar o desempenho. A escalabilidade poderá ser prejudicada tanto no distanciamento geográfico entre comutadores e controlador, quanto ao número de dispositivos de rede que requisita os serviços dos controladores, poderá não atender a todas as requisições dos comutadores (GUEDES, 2012).

Segundo López (2014) argumenta pontos referente a quantidade de controladores disponíveis e como deveriam ser distribuídos, levando em consideração os tipos de dispositivos de rede e do software SDN configurado. Para Guedes (2012) sobrepõe que a busca de soluções de controle otimizadas, devem considerar como uma função prática que pode elevar ao máximo a distribuição de controladores para permitir falhas, ao passo que pode também majorar o desempenho no grau em que a latência e o tempo de tramite de fluxos são reduzidos.

O conceito é que a rede seja dividida em três camadas lógicas. A camada de comutação representada pelos elementos de encaminhamento de pacotes (switches), os elementos que armazenam as tabelas de encaminhamento, igual como ocorre no protocolo OpenFlow. Dessa forma, seria mantido as estatísticas relativas aos fluxos utilizado nos contadores. A camada de distribuição “Sistema Operacional da Rede”, compõe a comunicação entre os controladores, anunciando os eventos e o estado de cada controlador. Nesta camada pode ser encarregada a um controlador, ao qual é conferida a responsabilidade de manter uma cópia atual da rede.

Na última camada lógica é a de aplicações de controle, que realiza comunicação utilizando a camada intermediária, e pode ser executada em um ou múltiplos nós (RECHIA, 2014).

5.2 O Protocolo openflow

É um protocolo de código aberto, arquitetado para comunicação e troca de mensagens entre controlador e os elementos de rede, segundo Amorim (2012) explica que permite que se empregue equipamento de redes comerciais para pesquisa e experimentação com novos protocolos de rede em paralelo com a operação de redes virtualizadas, conclui Rothenberg (2010).

Para isso ser alcançado uma interface de programação terá que permitir ao programador controlar diretamente os elementos de encaminhamento de pacotes de um determinado comutador, podendo ser um equipamento de rede convencional, que geralmente possui um poder de processamento maior do que equipamentos arquitetados virtualmente em ambientes de testes (LOPEZ, 2014).

5.2.1 Características

O protocolo OpenFlow exibe uma tecnologia de virtualização de rede, na qual comutadores podem ser configurados para trabalhar com o tráfego de rede especializado, protocolos experimentais diferente do padrão IPv4 (Internet Protocol version 4/ Protocolo de Internet versão 4), e tráfego de rede. Este protocolo é vital para o OpenFlow por ser responsável pela comunicação entre os comutadores, assim como uma rede de gerência SNMP (Simple Network Management Protocol/ Protocolo Simples de Gerência de Rede) carece que todos os elementos de rede estejam implantados em uma rede gerenciável, o OpenFlow precisa que todos os comutadores permaneçam conectados ao controlador (GUEDES, 2012).

Recentemente existem duas versões do protocolo OpenFlow que, em princípio serão denominadas de verões básica e avançada, apesar das especificações e funcionalidades diferirem, as características são comuns nas duas versões, a comunicação utilizada entre comutadores e controlador, tipos de comutadores e parâmetros que podem ser configurados em formato de mensagens trocadas durante a comunicação entre elementos da rede OpenFlow e o controlador (AMORIM, 2012).

A definição do controlador e os comutadores que utilizem TCP para se comunicar, não há necessidade de que haja uma conexão física entre eles, podendo utilizar uma rede de produção. Entretanto, seja utilizando um canal seguro para que a comunicação, possa garantir a confidencialidade de todas as informações (RECHIA, 2014).

Um comutador do Tipo 0, para comutadores OpenFlow de configuração básica, dispõe de uma tabela de fluxo que implementa classificação e encaminhamento. A parte de classificação da tabela de fluxo contém um conjunto padrão que são conferidos os pacotes, na maioria dos comutadores esta correspondência é implementada com hardware TCAM (Ternary Content Addressable Memory/Memória de Endereçamento de Conteúdo Ternário), que usa para executar a classificação, atingindo altas velocidades de processamento. O OpenFlow consente ao administrador a opção da implementação a ser utilizada para classificação de tráfego (GUEDES, 2012).

Tabela 1 – Cabeçalho padrão de mensagens openflow.

0	8	16	31
VERS	TIPO	TAMANHO TOTAL	
ID DA TRANSAÇÃO			

Fonte: Rothenberg, 2010.

A mensagem do OpenFlow inicia com um cabeçalho de tamanho fixo composto por 16 octetos conforme apresentado na figura 2. No campo VERS faz referência ao número da versão. O OpenFlow estabelece 24 tipos de mensagens que podem ser definidos no campo TIPO. O campo TAMANHO TOTAL, medido em octetos, poderá depender do tamanho total da mensagem, juntamente com o cabeçalho. O campo ID DA TRANSAÇÃO é um valor único que permite que o controlador imponha respostas a determinadas solicitações (GUEDES, 2012).

O OpenFlow é definido com formato exato das mensagens com representação para cada itens de dados. De acordo com suas especificações,

podem estabelecer um conjunto mínimo de requisitos e campos que o comutador na versão básica precisa ser capaz de corresponder (AMORIM, 2012).

Tabela 2 – Campos de classificação dos comutadores openflow tipo 0

CAMPO	SIGNIFICADO
Ethersrc	Endereço de origem thernet 48-bit
Etherdst	Endereço de destino ethernet 48-bit
VLAN ID	VLAN tag 12-bit no pacote
IPv4src	Endereço de origem IPv4 32 bits
IPv4dst	Endereço IPv4 de destino para IPv4 32 bits
Proto	Campo protocolo IPv4 8 bits
TCP/UDP/SCTP src	Porta origem TCP/UDP/SCTP 16 bits
TCP/UDP/SCTP dst	Porta destino TCP/UDP/SCTP 16 bits

Fonte: Amorim, 2012.

Vale ressaltar, que na tabela 2, muitos campos apresentam protocolos convencionais existentes, admitindo forma interação do tráfego SDN com redes IPv4 e Ethernet. Contudo, comutadores “Tipo 0” apresentam entraves, com impossibilidade de especificar o encaminhamento de tráfego ICMP (Internet Control Message Protocol/Protocolo de Mensagens de Controle de Internet) e distinguir entre solicitações ARP (Address Resolution Protocol/Protocolo de Resolução de Endereço). No entanto, permite que administradores instituam regras de encaminhamento para o tráfego em determinada porta do comutador, permitindo varios ensaios (GUEDES, 2012).

As limitações da versão básica de controladores OpenFlow são implementadas pela versão 1.3 ou conhecida como avançada, adicionando funcionalidades essenciais, entre elas, a inserção de campos no cabeçalho de pacote e inclusão de novos protocolos MPLS (NOBRE, 2014).

Na nova versão, os controladores assumem a possibilidade de múltiplas tabelas de fluxos dispostas em uma pipeline, na qual a primeira comparação ocorre sempre com a primeira tabela de fluxo, por onde são definidas ações específicas para cada entrada, podendo definir determinado pacote seja encaminhado para a

próxima tabela de fluxo. No entanto, o salto do pacote não poderá acessar a tabela anterior, evitando que ocorra loop (AMORIM, 2012).

Nesta versão apresenta novos campos, tanto para uso na classificação de pacotes, como dos campos de correspondência, apresentados na tabela 3. Destinados a serem utilizados dentro da pipeline, como o campo de Metadata. Em um determinado estágio da pipeline, com base no endereço IPv4, quando calculado o próximo estágio ao qual determinado pacote poderá ser encaminhado, o OpenFlow não especifica o conteúdo, pois a pipeline deve organizar de maneira que os estágios sequentes façam com que o conteúdo e o formato de dados sejam recebido pelo estágio anterior.

Tabela 3 – Campos de classificação de comutadores openflow versão 1.3

CAMPO	SIGNIFICADO
IngressPort	Porta do comutador pela qual o pacote chegou
Metadata	Campo de 64 bits de metadados usados na pipeline
Ether src	Endereço de origem ethernet 48-bit
Ether dst	Endereço de destino ethernet 48-bit
Ether type	Campo tipo ethernet 16-bit
Vlan ID	Vlan tag 12-tag no pacote
Vlan priority	Número MPLS prioritário vlan 3-bit
MPLS label	Rótulo MPLS 20-bit
MPLS class	Classe de tráfego MPLS 3-bit
IPv4 src	Endereço de origem Ipv4 32-bit
IPv4 dst	Endereço de destino Ipv4 32-bit
ARP proto	Campo protocolo IPv4 8-bit
ARP opcode	Opcode ARP 8-bit
IPv4 tos	Bits tipo de serviço Ipv4 8-bit
TCP/UDP/SCTP src	Porta origem TCP/UDP/SCTP 16-bit
TCP/UDP/SCTP dst	Porta destino TCP/UDP/SCTP 16-bit
ICMP type	Campo tipo ICMP 9-bit
ICMP code	Campo código ICMP 8-bit

Fonte: Amorim, 2012.

Com a definição do protocolo e apresentando a estrutura das mensagens é possível conhecer o seu funcionamento e a forma como os pacotes são analisados conforme as regras de encaminhamento instaladas.

5.2.2 Funcionamento

No recebimento de um pacote, o comutador do protocolo OpenFlow analisa o cabeçalho e checa as regras de encaminhamento definidas nas tabelas de fluxo. Os contadores atualizam-se e são executadas as ações adequadas. Caso não haja relação do pacote recebido com regras na tabela de fluxo o pacote é encaminhado,

por completo, ao controlador. Alternativamente, o pacote pode é armazenado em um buffer de memória do hardware (LOPEZ, 2014).

Necessariamente o OpenFlow define três ações básicas que são associadas a um padrão de classificação, apontadas por Rothenberg (2010) como:

I Ao encaminhar o pacote para uma ou várias portas do comutador, o comutador inicializa as regras configuradas pelo administrador. Desta forma, os pacotes serão encaminhados. Há possibilidade existir regras para um determinado conjunto de portas permitindo a implementação de broadcast e multicast.

II Ao encapsular o pacote e enviar para o controlador externo, permite o tratamento do pacote para os quais não foram definidas previamente as regras de encaminhamento, escolhendo um caminho para o fluxo TCP, configurado com regra de classificação e em seguida será encaminhado o pacote para o comutador que, irá continuar o processamento do pacote.

III Se o pacote for descartado sem o devido processamento, permitirá ao OpenFlow lidar com problemas, como ataque de negação de serviços ou broadcast excessivo de determinado host.

Para López (2014) ressalta, que o encaminhamento do pacote para o processamento normal do equipamento nas camadas 2 ou 3, permite que o tráfego experimental não interfira no processamento padrão do tráfego de produção. Também existe outra forma de garantir a separação, através da configuração por meio de VLANs (Virtual Local Area Network/Redes Locais Virtualizadas) para fins experimentais.

A segmentação permitirá que a operação de equipamentos híbridos, processando o tráfego juntamente com os protocolos e as funcionalidades embarcadas do equipamento, funcione paralelamente nos moldes baseado no desenvolvimento OpenFlow. A exemplo de modelo híbrido de tráfego é apresentado na figura 2.

Figura 2 – Modelo de operação híbrido para tráfegos padrão e openflow

Fonte: Rothenberg, 2010.

As redes definidas em software nasceram da necessidade do desenvolvimento e implementação de tráfegos e protocolos experimentais sem conceber risco ao tráfego de dados reais, foi observado que o protocolo OpenFlow permite a conexão com redes IP, na próxima sessão apresentaremos as aplicações dessa nova tecnologia.

5.3 Métodos e aplicações

A estrutura de redes definidas por software permitem sistemas de rede flexível e centralizada, podendo ser amplamente utilizadas para o desenvolvimento de novas funcionalidades em um número variado de ambientes. Dentre estes, alguns domínios e aplicações identificados (LOPEZ, 2014).

De acordo com Nobre (2014) existe uma expectativa que as inovações tecnológicas devam apresentar taxas de crescimento de (54%) nos anos seguintes, abrangendo principalmente as categorias de software, nos segmentos de virtualização e controle representarão juntos, investimentos de aproximadamente 5,9 bilhões de dólares.

Esta evolução tecnológica e ampliação da quota de mercado, vem causando impacto diretamente nas compras, com clientes exigido novos equipamentos de redes que sejam habilitados ou compatíveis com SDN. Esta

tendência vai atingir cada segmento do mercado, de modo que em nenhum cliente ou fornecedor esteja longe às redes definidas por software.

No entanto, não significa o fim das redes tradicionais por hardware, entretanto, os recursos de redes precisarão dos roteadores e recursos de encaminhamento baseados na tecnologia TCAM. Portanto, essa nova abordagem para o hardware de rede ainda serão necessário.

Entre os diversos cenários de rede existente em que o uso da tecnologia OpenFlow proporciona resultados promissores, destacam-se as seguintes aplicações:

- Controle de Acesso: Possibilita a programação dos fluxos, permitindo que regras de acesso sejam configuradas com base nas informações macros e não apenas no uso de um firewall em um enlace de rede. As regras baseadas não apenas em protocolos e informações de destino e origem de determinado pacotes, mas no conjunto mais simplificado (LOPEZ, 2014).
- Gerência de Redes: Permitindo nas SDNs a simplificação de configuração e gerenciamento da rede, enquanto os contadores de fluxos permitem o monitoramento mais específicos de acordo a necessidade do administrador (BERTHOLDO, 2012).
- Redes Domésticas: A aplicação dos princípios de SDN em redes domésticas consiste na utilização de um roteador doméstico compatível com o protocolo OpenFlow, transferindo ao provedor de acesso o controle da rede através de um controlador SDN, sendo assim, permitindo programar cada roteador com as regras de acesso definidas para cada cliente e obter uma visão geral do tráfego de dados, através da enlace de acesso de cada cliente.

Portanto, identificando os padrões de tráfego de dados em encontro com padrões de tráfego que podem apresentar riscos na rede, utilizado por códigos maliciosos (LOPEZ, 2014).

- Data Center: Nos ambientes onde aplicações de usuários demandam de várias requisições, faz necessário o isolamento de tráfego entre diversos

usuários. Para obter esse isolamento por meio de dispositivos de rede, seria necessário a utilização de configuração de VLANs individuais para cada cliente. No entanto, como é um recurso limitado pelo tamanho do cabeçalho utilizado para identificação, permanece inviabilizável a medida que o número de clientes aumentam, deixando as tarefas de gerenciamento da rede bastante complexa (AMORIM, 2012).

A solução para o problema está diretamente ligado ao conceito de um comutador virtual distribuído, podendo ser implementado com o uso de um controlador de rede, inserindo automaticamente as regras de encaminhamento de tráfego através das portas dos switches virtualizados e que serão atribuídos a cada usuário específico (AMORIM, 2012).

Existem outra aplicação a redes SDNs em data center com soluções de conservação de energia, por meio da redução da taxa de transmissão e o desligamento de dispositivos de rede em sistemas redundantes com elementos ociosos, identificados pela visualização global da rede.

No entanto, o controle de rotas e decisões de encaminhamento permite que ocorra a implementação de pontos de controle eficazes na intercepção de pacotes de menor importância, que alcançarem os equipamentos em operação sob o regime de estado de hibernação do dispositivo, evitando que sejam ativados desnecessariamente (LOPEZ, 2014).

- SD-WAN (Wide Area Network/Rede de Área Ampla): Esta arquitetura é baseada em um sistema de variados pontos de conexão, sendo assim, as corporações desenvolveram as soluções de rede baseadas em conexões com o data center mais próximo, utilizando redes MPLS ou outra solução disponível no mercado (NOBRE, 2014).

Esse sistema é utilizado por muito tempo, atendendo as necessidades corporativas fornecendo conexões ponto a ponto. Entretanto, nem todos os empreendimentos estão conectados em um ambiente físico a um data center, havendo muitas empresas virtuais utilizando os recursos de nuvens (GUEDES, 2012).

A partir do ano 2000, grande parte do tráfego corporativo era voltado a data centers. Atualmente muitas corporações realizam o acesso por meio da rede mundial. Com o intuito de permitir a visualização de conteúdo organizacional como informações sobre os parceiros, porém o acesso fora de foco aos assuntos não profissionais ocorre, nos casos da necessidade de balanceamento do tráfego os dados serão encaminhados ao data center e a Internet (HARANAS, 2016).

Através da utilização da arquitetura tradicional, o tráfego de dados recorrente à Internet seria roteado do ambiente corporativo até o data center, por onde seria necessário um ou mais saltos entre roteadores até alcançar a Internet. Acrescendo latência e consumindo a largura de banda, além de aumentar as despesas da rede, pois quanto mais saltos, maior será o consumo de banda MPLS necessária. Contudo, se o tráfego fosse encaminhado diretamente a Internet, haveria poucos saltos até a rede WAN (BERTHOLDO, 2012).

A saída seria a implementação de uma rede híbrida, composta por duas conexões, uma convencional com MPLS e outra, conectando a filial ao data center. Destinando todo o tráfego corporativo, conectado diretamente a Internet, permitindo uma conexão direta com rede global, como uma rede definida em software na nuvem.

O problema do uso da banda MPLS para trafego Internet seria solucionado com benefício de uma segunda rede, sendo implementada a engenharia de tráfego de forma que o sistema pudesse redefinir um link com a menor latência (NOBRE, 2014).

Até o momento identificamos os componentes integrantes de uma rede definida por software, apresentado todos os padrões e protocolos necessários e suas aplicações. No próximo capítulo será realizada a implementação de uma rede SDN com a finalidade de avaliar o funcionamento da redes e com uso do protocolo OpenFlow.

6 PROCEDIMENTOS METODOLÓGICOS

Neste capítulo será detalhado os procedimentos adotados para elaboração do ambiente virtual composto por uma rede IP integrada a uma rede SDN em uma máquina virtual com a seguinte configuração:

Máquina Física:

- Processador Intel Core i7 2640M 2.80 GHz;
- 16GB de memória RAM DDR3 1333 MHz;
- Disco rígido SSD com 512GB de armazenamento;
- Sistema operacional GNU/Linux POP!OS 20.04 LTS.

Máquina Virtual:

- Processador Intel Core i7 2640M 2.80 GHz (Dois Núcleos);
- 4GB de memória RAM DDR3 1333 MHz;
- Disco rígido com 15GB de armazenamento;
- Sistema operacional GNU/Linux XUBUNTU 20.04 LTS.

No equipamento testado, construímos um ambiente, instalando os aplicativos na máquina virtual, para gerenciamento do Mininet, openvswitch e controlador Ryu, que permitirá a reprodução das características por meio de emulação de uma rede local similar no ambiente de T.I., possibilitando a criação de vários cenários.

Sendo o objetivo das redes definidas por software é a separação do plano de dados do plano de controle, esta emulação inicia com um ambiente de testes através da seleção do controlador compatível com o protocolo OpenFlow, apresentando os procedimentos para instalação e execução, nos tópicos a seguir.

6.1 Controlador sdn

Com bases em trabalhos desenvolvidos no segmento de SDN, aliado ao conteúdo ministrados nas disciplinas “Redes Convergentes e Redes de Longa Distância e Tecnologias de Acesso” utilizamos a plataforma Linux utilizando a configuração de em um host, atrelado a possibilidade de gerenciamento via console, optamos pelo controlador Ryu em sua versão 4.30.

Está versão escolhida pode ser executada diretamente no sistema operacional ou na própria máquina virtual configurada para este ambiente, que permite ao administrador de rede se familiarizar com o controle de fluxos e visualizar a topologia SDN empregada (LOPEZ, 2014).

A instalação do Ryu, segue o roteiro ministrado na disciplina de Redes de Longa Distância e Tecnologias de Acesso, exigindo atenção a certos pré-requisitos, na instalação de ferramentas e pacotes adicionais para construção do ambiente.

Após concluir esta etapa, é possível acompanhar o processo de download e a instalação de todos os pacotes a partir do terminal, na qual os arquivos estão detalhadamente exibidos.

6.2 Emulador mininet

Outro recurso importante para esta prática é o Mininet, um sistema que permite a rápida criação de grandes redes escaláveis utilizando um computador padrão, utilizando-se dos princípios de virtualização do Sistema Operacional. Os elementos de uma rede emulada é de executar os processos, permitindo o desenvolvimento e customização de hosts virtuais dentro de uma Redes Definidas por Software de maneira fácil (LOPEZ, 2014).

Possibilitando a emulação de uma rede SDN, composta por switches OpenFlow (Open vSwitches), hosts e controladores. Além de obter uma topologia, com aplicação definida através de uma rede customizada utilizando a linguagem de programação Python, utilizando o aplicativo (miniedit), necessário para o

direcionamento a um servidor. O Mininet pode ser carregado em modo texto, para reduzir o tamanho e uso de memória da máquina (RECHIA, 2014).

Dentre as opções de instalação, optamos por uma máquina virtual contendo o sistema de pacotes de dependências do Mininet devidamente instalada sob o sistema operacional Linux XUBUNTU 20.04. Os autores recomendam a utilização da versão i386 de 32bits para sistema operacional Windows e aplicativo de virtualização VirtualBox (GUEDES, 2012). Podendo ser ambos aplicáveis no ambiente de testes desta pesquisa. No entanto, o sistema utilizado não disponibiliza a versão i386, somente AMD64.

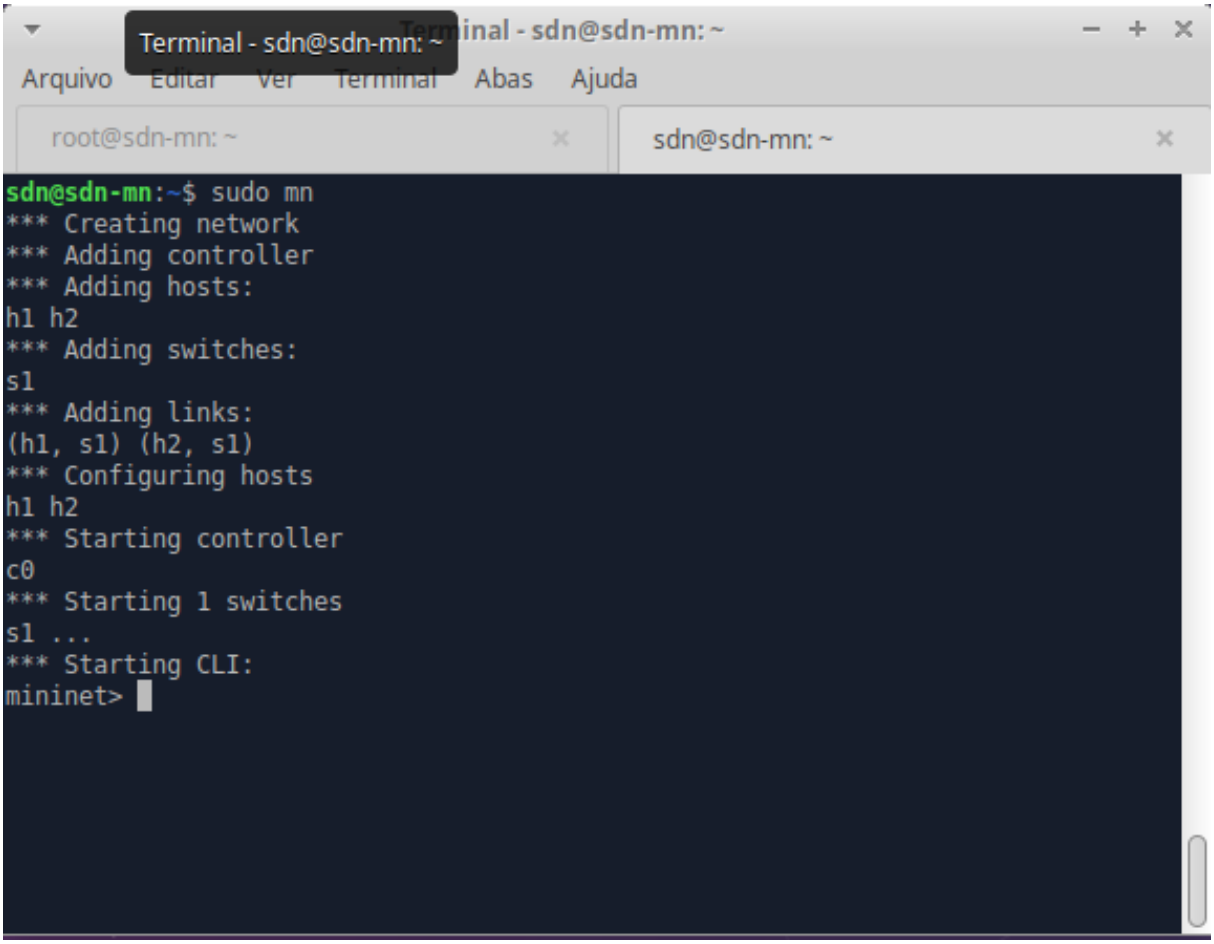
O arquivo baixado em uma máquina de teste preparada para a aplicação do Mininet, sendo necessário apenas descompactar o arquivo e importar as configurações utilizando o terminal Linux, conforme os passos a seguir.

A máquina de teste estando em execução, operando por meio do terminal Linux, deverá ser atualizado os repositórios e os pacotes, e em seguida escolher os pacotes que deverão ser essenciais para instalação dos programas de testes.

Para integrar o Mininet com a topologia foram configuradas quatro interfaces de console, a figura 3 mostra que cada terminal terá uma tarefa de executar uma aplicação para os testes.

Por fim, iniciando o “Mininet”, a máquina é iniciada e em seguida é possível acessá-la através do terminal executando o Mininet, conforme na figura 3.

Figura 3 – Executando o mininet



Fonte: Verificação do ambiente de teste no mininet (2021) e adaptado pelo autor.

Contudo, o acesso ao Mininet é feito por meio do termina Linux e, como são utilizados os adaptadores e conexões de rede, ainda não é possível emular uma SDN que possa se conectar ao controlador. Para tanto é necessário incluir a topologia, conforme será melhor apresentado na próxima sessão.

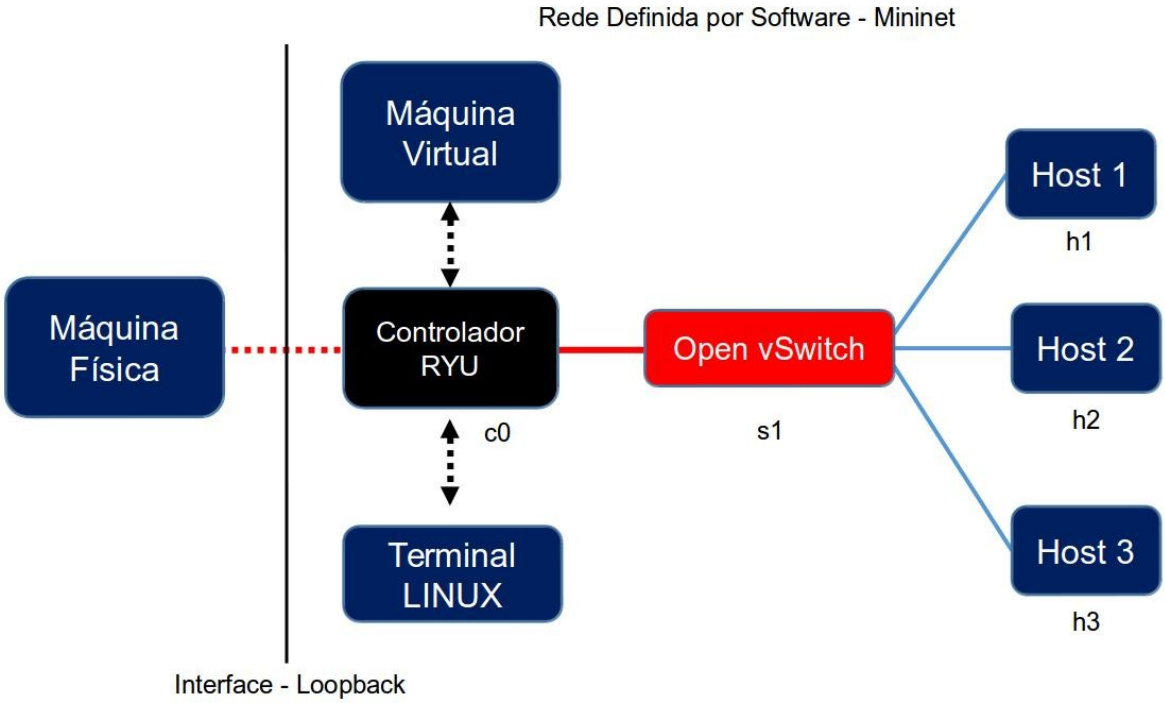
6.3 Topologia

A topologia híbrida de testes pode ser observada na figura 4, entretanto, em termos de extensão e volume de tráfego, a capacidade real de uma rede híbrida, é suficiente para comprovação das características e funcionalidades do Switches OpenFlow e do papel do controlador em redes SDN.

Nestes estão organizados um controlador, representando uma rede de padrão IP, com a máquina de teste com a aplicação do Mininet, responsável pela emulação da rede SDN e combinada com um switch OpenFlow e três hosts.

O controlador SDN Ryu instalado no sistema operacional na máquina de teste onde o ambiente está sendo emulado, e será acessado através de uma interface Loopback.

Figura 4 – Topologia ip/sdn



Fonte: Informações do layout da topologia de teste (2020) e adaptado pelo autor.

Executando via terminal Linux com a aplicação Mininet configurada na máquina de teste utilizando as interface loopback. As máquinas emuladas pelo Mininet estão conectada ao switch S1 está dedicada aos tráfegos de gerência e controle, todos conectados ao controlador C0, respectivamente.

O controlador C0 está conectado ao switch S1, bem como à interface loopback do host, responsável pela interligação das redes de gerenciamento, controle e produção.

Enquanto o h1, é configurado para obter o endereçamento IP através do servidor DHCP (Dynamic Host Configuration Protocol/Protocolo de Configuração Dinâmica de Host), utilizando endereços reservados em 3 faixas do endereço de rede 10.1.1.0/24. No switch desempenha a função de concentrar os hosts na rede, enquanto o controlador estará desempenhando a função comutação da rede.

Vale ressaltar que, por definição, o switch não propagará broadcast, utilizado pelos hosts para alimentar a tabela ARP, assim será necessário obrigar o controlador a encaminhar mensagens broadcast. Isto ocorrem principalmente, através do comando (ip helper), configurado na interface local (LOPEZ, 2014).

Tabela 4 – Endereçamento ip da rede

Elementos da Rede	Interface	Endereçamento de Rede (IP)
Host Físico	Loopback	IP classe b / 16
Mininet	h1, h2 ... hn	IP classe a / 24
Openswitch	S1	IP classe a / 24

Fonte: Informações do endereçamento de ip da rede deteste (2020) e adaptado pelo autor.

A escolha para o endereçamento automático ocorreu pela flexibilidade de implementação de mais hosts no ambiente de testes. O endereçamento dos demais dispositivos de rede pode ser verificado conforme a tabela 4. Uma vez arquitetada a topologia e os ativos da rede é possível iniciar a execução do ambiente de testes.

6.4 Testes de conectividade

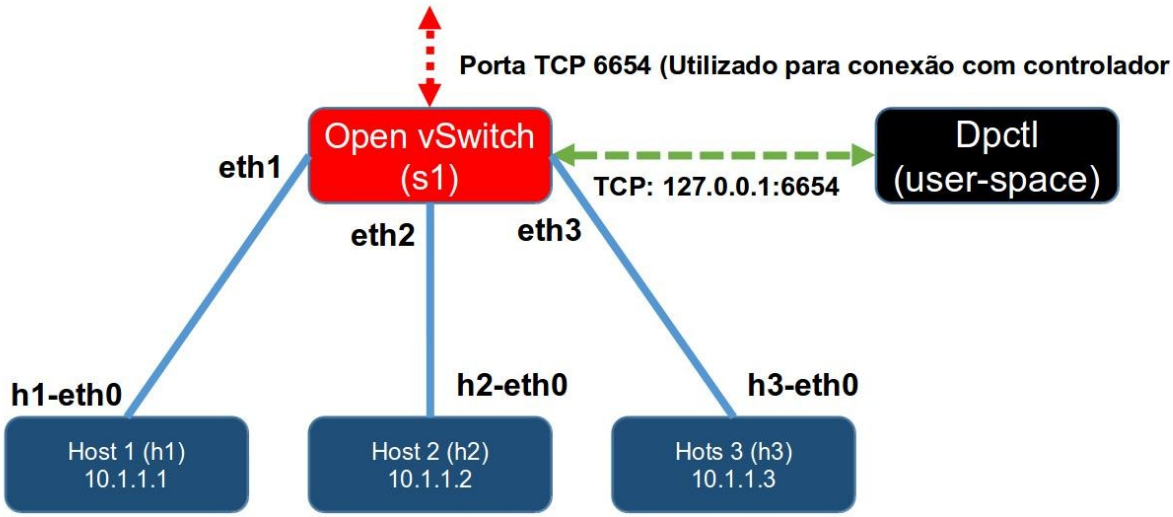
O Mininet teve seu início em 2009, quando Bob Lantz desenvolveu um script em Python para demonstrar o uso de Linux Containers (LXC) para a emulação de redes OpenFlow.

É uma ferramenta “leve” para a emulação de Redes SDN (Software-Defned Network) que permite: a rápida prototipação de grandes redes virtuais escaláveis; composta de hosts, switches, controladores OpenFlow e enlaces.

Para tanto, o Mininet é utilizado para configurações primitivas de virtualização no nível de Sistemas Operacionais. Utilizando de containers (LXC), que é uma virtualização leve em nível de kernel. No entanto, serve para virtualizar os

recursos da máquina através de uma abstração de processos e redes, permitindo criar, interagir e customizar protótipos de forma rápida, com gigabits de banda e centenas de nos. O Mininet cria uma rede virtual, alocando os processos dos hosts no namespace da rede, conectando-os com pares de placas Ethernet's virtuais.

Figura 5 – Exemplo de processo executando no namespace da rede



Fonte: Informações do processo de execução do namespace da rede (2020) e adaptado pelo autor.

Cada host (em azul) na figura é processo executando no namespace da rede, possui uma interface Ethernet virtual para se comunicar: h1-eth0.

A grande vantagem do Mininet é que podemos desenvolver funcionalidades ou até arquiteturas totalmente novas, sem depender da rede para tal (equipamentos e conexões físicas).

Dessa forma, podemos testar as novas implementações em topologias altamente diversas e, posteriormente, portar o mesmo código para um ambiente real. O Mininet permite desenvolver topologias personalizadas utilizando scripts em Python, com a possibilidade de interação com a rede física existente utilizando a linha de comando (CLI). É possível ainda a conexão com controladores externos, como o POX, Ryu, OPENDAYLIGHT e ONOS, para o gerenciamento das tabelas de encaminhamento dos switches.

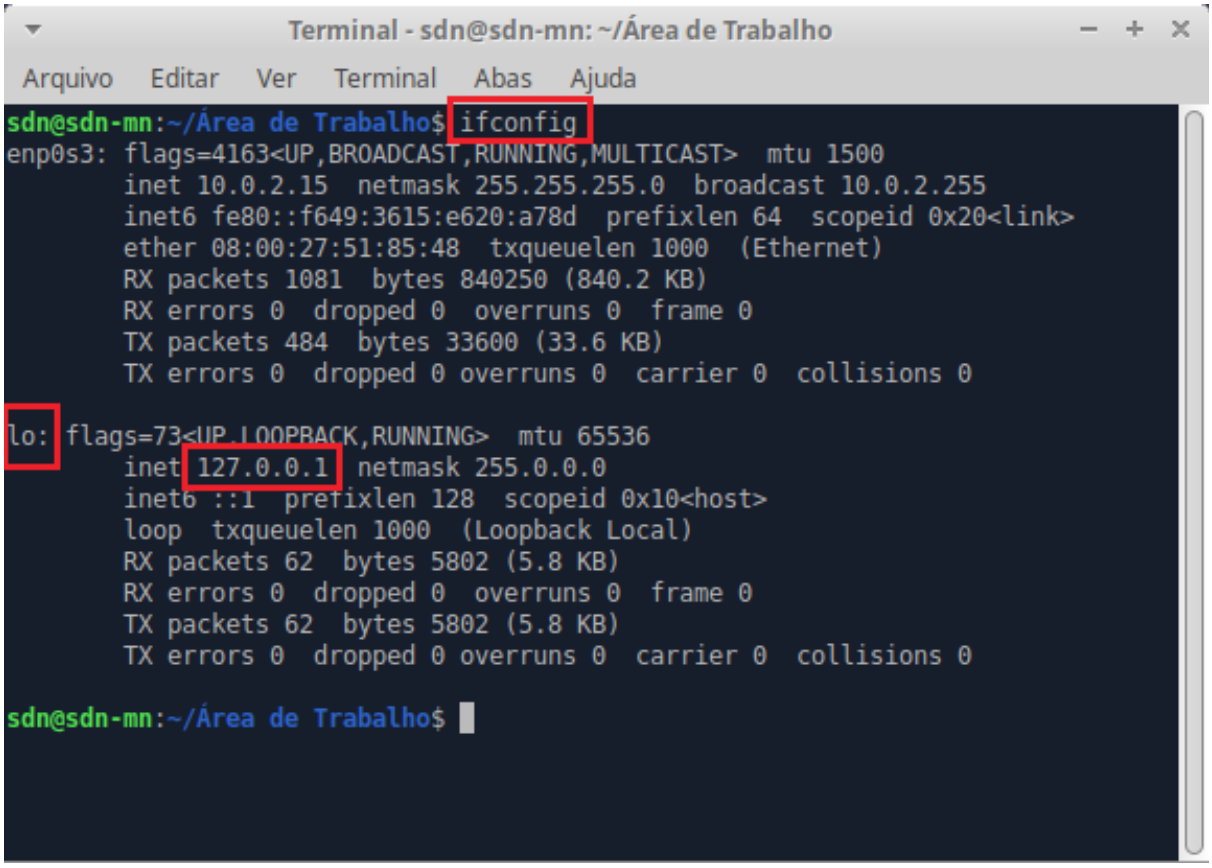
6.5 Ambiente de trabalho utilizado (ferramentas)

Sistema Operacional: GNU/Linux XUBUNTU 20.04 LTS.
Test Bed: Mininet.
Controlador: Ryu.
Switch: Open vSwitch.
Captura de Pacotes: Wireshark.
Gerador de Tráfego: IPerf.

6.5.1 Iniciando os testes

O primeiro passo recomendado é a verificação da conectividade com o emulador Mininet no sistema rodando o controlador. Através do terminal será possível executar comandos para verificação das configuração da interface de rede, através do comando “ifconfig”, conforme demonstrado na figura abaixo:

Figura 6 – Verificando interface com “ifconfig”

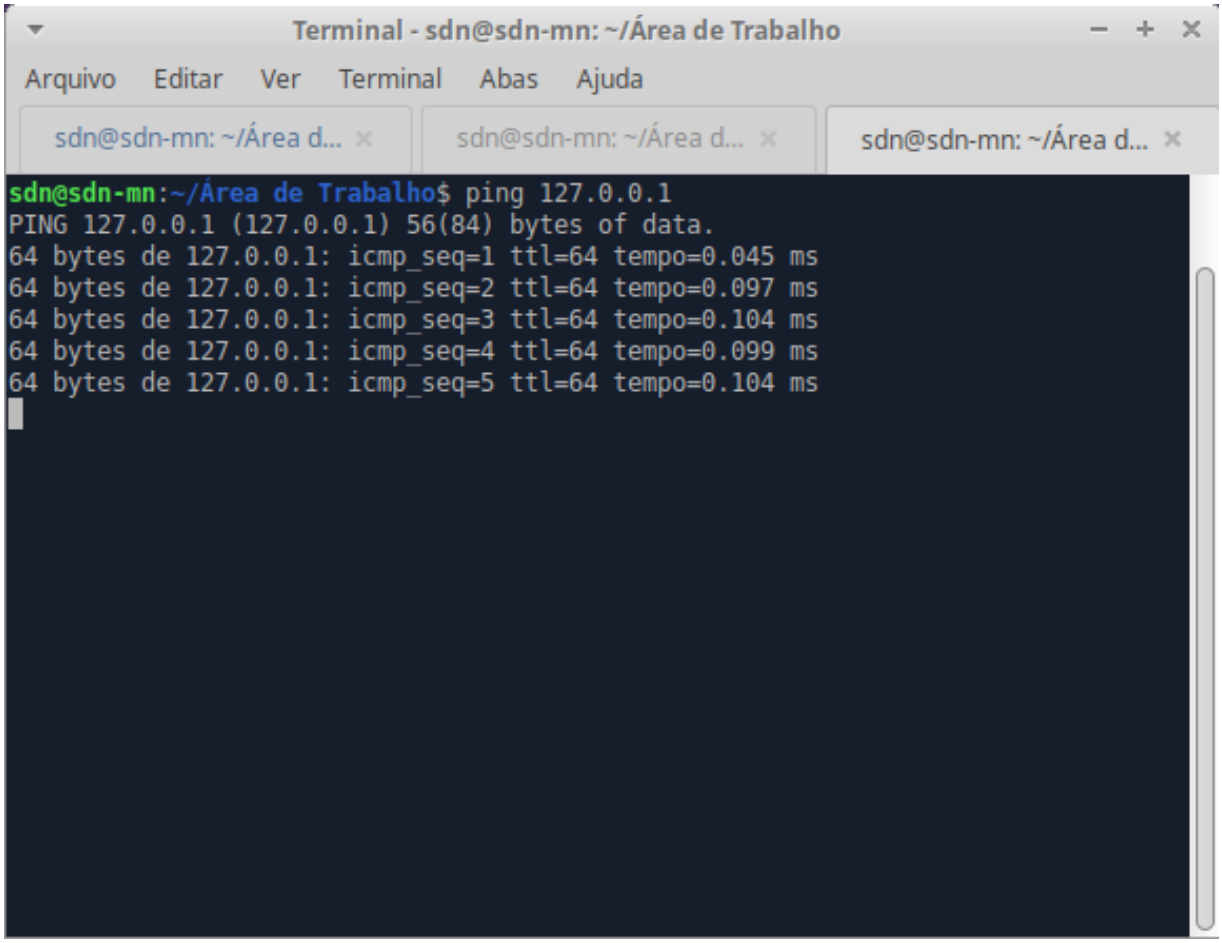


Fonte: Verificação do ambiente de teste no mininet (2021) e adaptado pelo autor.

Nota-se que as interface de loopback está listada, isso se deve ao fato que ela está pronta para conexão, logo pode-se visualizar mais especificamente executando o comando “ifconfig lo”.

Como demonstrado no figura 6, o endereço IP atribuído a interface Loopback do sistema e também ao controlador foi “127.0.0.1”. No entanto para validar a conectividade da máquina virtual, na qual está executado o Mininet, será necessário realizar um simples teste de conectividade pelo comando “ICMP” ou mais conhecido popularmente como “ping”, apresentado na figura a seguir:

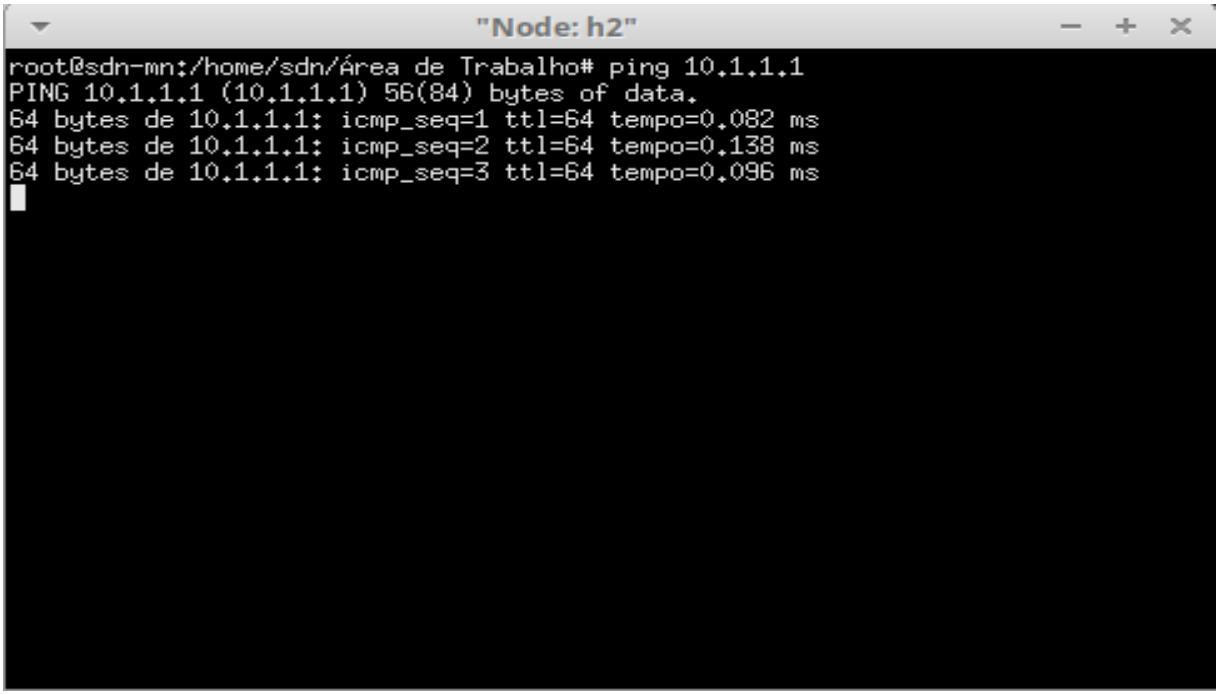
Figura 7 – Teste de conectividade



Fonte: Verificação do ambiente de teste no mininet (2021) e adaptado pelo autor.

Entretanto se faz necessária a validação a partir do host virtualizado, e o processo prossegui o mesmo princípio. Vale ressaltar, que o sistema operacional por padrão define como rota default o gateway da interface de rede externa LAN (Local Area Network / Rede Local de Trabalho), quando ativa. Nesta situação todas as solicitações direcionadas ao IP 10.1.1.0 (Mininet) são direcionadas ao switch e encaminhados a controlador, como pode ser observado na figura a seguir:

Figura 8 – Mininet - Teste de conectividade



Fonte: Verificação do ambiente de teste no mininet (2021) e adaptado pelo autor.

6.5.2 Emulando switch openflow

Para demonstrar a finalidade do funcionamento de uma rede SDN, é preciso iniciar o controlador Ryu, emulado na rede composta por um switch OpenFlow e quatro hosts. Esta topologia será carregada, no terminal da máquina virtual, através do comando:

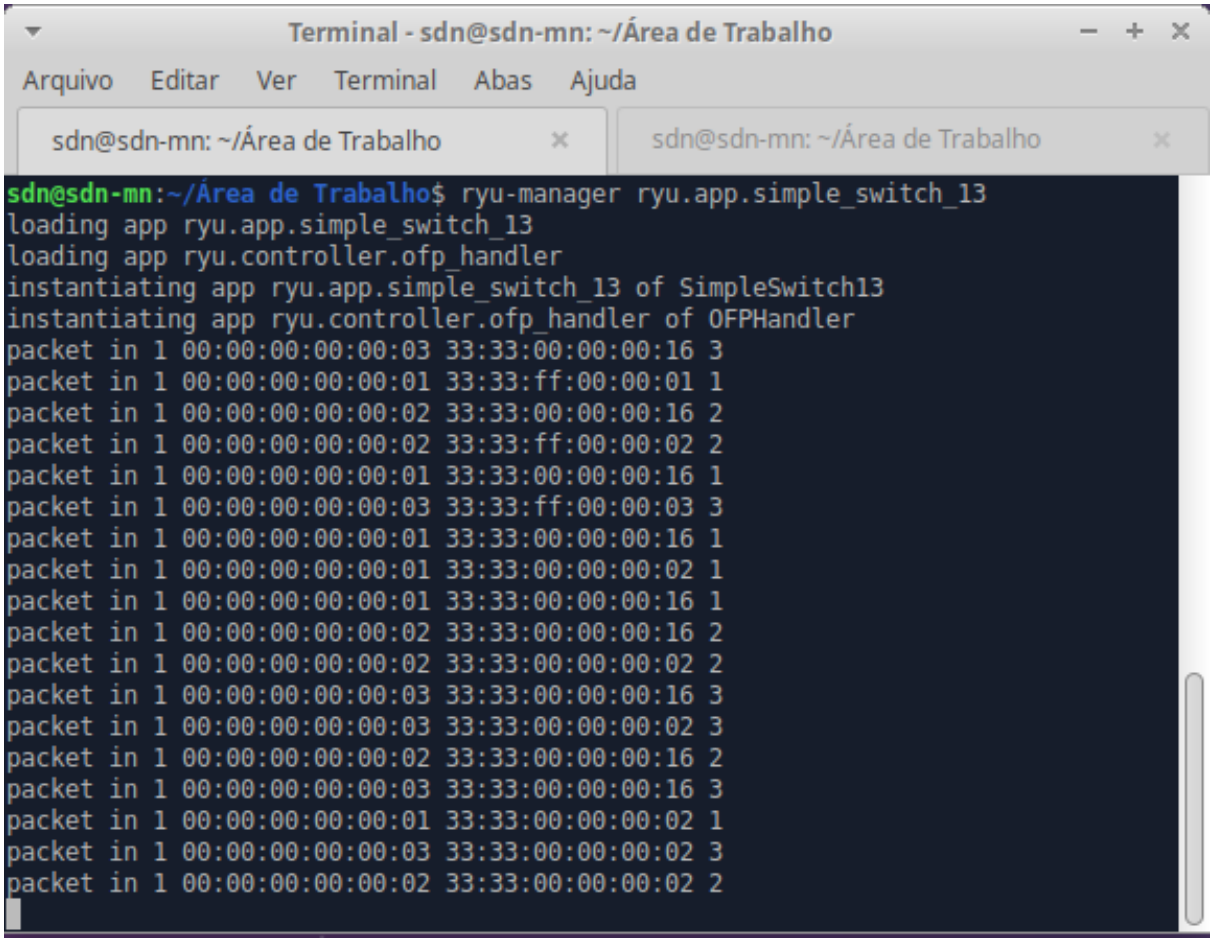
Abrindo o terminal primeira aba:

```
~$ ryu-manager ryu.app.simple_switch_13
```

Abrindo segunda aba no terminal:

```
~$ sudo mn --controller=remote,ip=127.0.0.1 --switch=ovsk,protocols=OpenFlow13 --mac --ipbase=10.1.1.0/24 --topo=single,4
```

Figura 9 – Carregando as configurações do controlador.

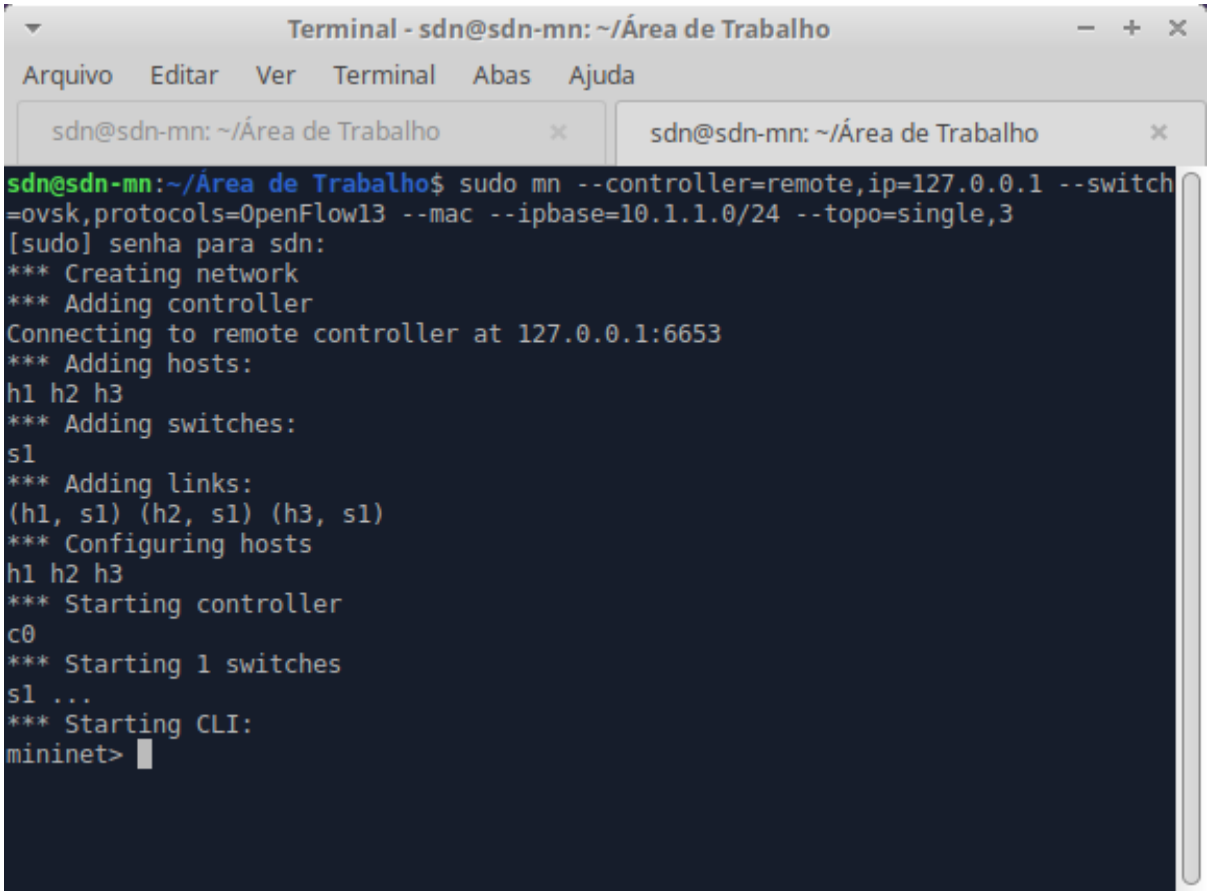


Fonte: Verificação do ambiente de teste no mininet (2021) e adaptado pelo autor.

Vale ressaltar, que existem alguns comandos primários ou básico no Mininet, neste caso se utilizar somente o comando “sudo mn” a aplicação é inicializada, com base nos parâmetros prescritos na linguagem do programa. A rede SDN será gerenciada por um controlador com endereçamento de rede 127.0.0.1:6653, sendo assim, não será definido o parâmetro de referência da topologia, portanto, o sistema definirá por uma topologia padrão, formada por um switch e três hosts.

Durante o processo de inicialização, o Mininet apresenta na tela as etapas da criação da topologia, podendo ser observado na figura 10.

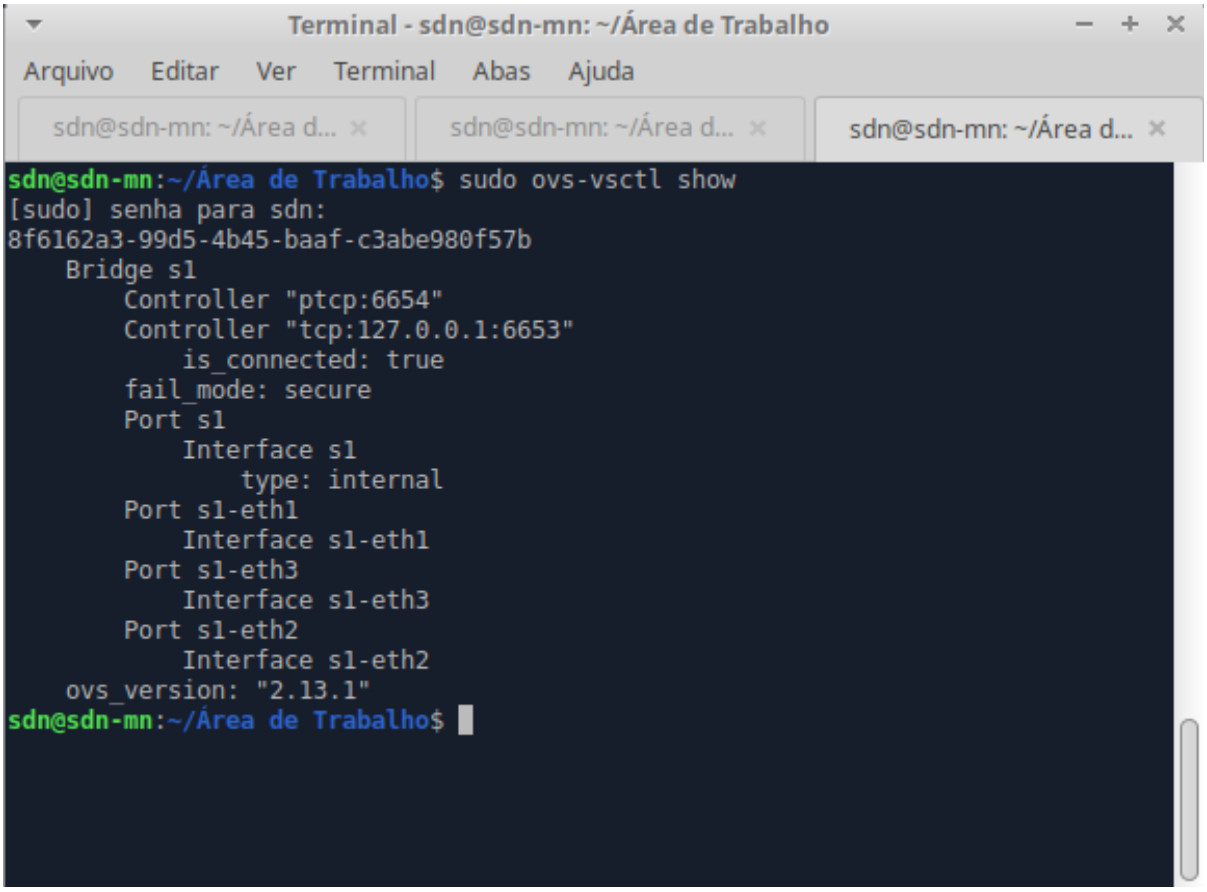
Figura 10 – Etapas de carregamento



Fonte: Verificação do ambiente de teste no mininet (2021) e adaptado pelo autor.

Portanto, é possível observar, por meio do terminal que no controlador, a descrição de todos os elementos da rede, neste caso o Switch OpenFlow.

Figura 11 – Visualização do ovs via terminal.



Fonte: Verificação do ambiente de teste no mininet (2021) e adaptado pelo autor.

No entanto, ainda não há configurações de fluxo atribuídas ao elemento no ambiente do switch emulado “s1”, identificado no controlador pelo ID 00:00:00:00:00:00:00:01, representado pelo o endereço MAC (Media Access Control).

Portanto, será necessário aos demais elementos da rede, para que sejam interpretados é indispensável que haja conectividade entre eles e, por tanto, as interfaces do switch possam ser configuradas.

Vale ressaltar, que na máquina virtual os emulados no ambiente sem interface gráfica, as configurações dos elementos da rede SDN poderá ser realizados a partir de acesso pelo próprio terminal ou redirecionamento a outro console de configuração “konsole e/ou x-term” (GUEDES, 2012). Portanto, o emulador utilizará da instância do OVS, configurado no sistema operacional e

executando dentro do Kernel do host hospedeiro, realizando todas as configurações necessárias, sem a necessidade de instalação e configurações de servidores em máquinas virtuais.

É plausível convalidar as conexões e obter informações específicas através do comando (sudo ovs-vsctl show), estabelecendo comunicação com os switches OpenFlow (LOPEZ, 2014).

Apresentando um único switch denominado de “s1”, seguido do controlador e das interfaces internas e externas definidas. Ao final é plausível notar a versão do OVS utilizada (LOPEZ, 2014).

Segundo Guedes (2012), contribui que a maioria dos switches OpenFlow começa com uma porta de escuta passiva na qual é admissível interagir com o mesmo sem a necessidade de ação por meio do controlador, na configuração atual é denominada a porta 6653.

De tal modo, é admissível conectar-se ao switch e checar o estado e capacidade das portas, bem como os fluxos ativos, através do utilitário DPCTL[1], que pode ser iniciado diretamente no console do Mininet, conforme o comando a seguir:

```
$ dpctl dump-flows tcp:127.0.0.1:6653
```

Mensagens de resposta:

```
stats_reply (xid=0x158a3b4a): flags=none type=1(flow)
```

Esta função irá apresentar a seguinte resposta: “descarregar a tabela de fluxo do switch OVS em execução no momento”. A aplicação Mininet deverá indicar que o dispositivo ao qual o comando está sendo aplicado irá retornar o fluxo dos OVS carregados.

1 Datapath management utility – utilitário que envia mensagens OpenFlow ao switch. É útil tanto para visualizar o status do switch quanto para inserir manualmente entradas da tabela de fluxo. Ele é especialmente útil para identificar erros, pois permite exibir o estado e os contadores dos fluxos. A maioria dos switches OpenFlow inicia com uma porta de escuta (por default a porta 6634), na qual é possível interagir com o switch – como obter suas regras de fluxo – sem ter que passar pelo controlador.

Logo que analisado, pode ser observado que não há carregamento na tabela de fluxos do switch SDN. Também não será possível conectar-se ao host1, desse modo deverá atribuir um fluxo a tabela do switch.

6.5.3 Analisando os fluxos com dpctl

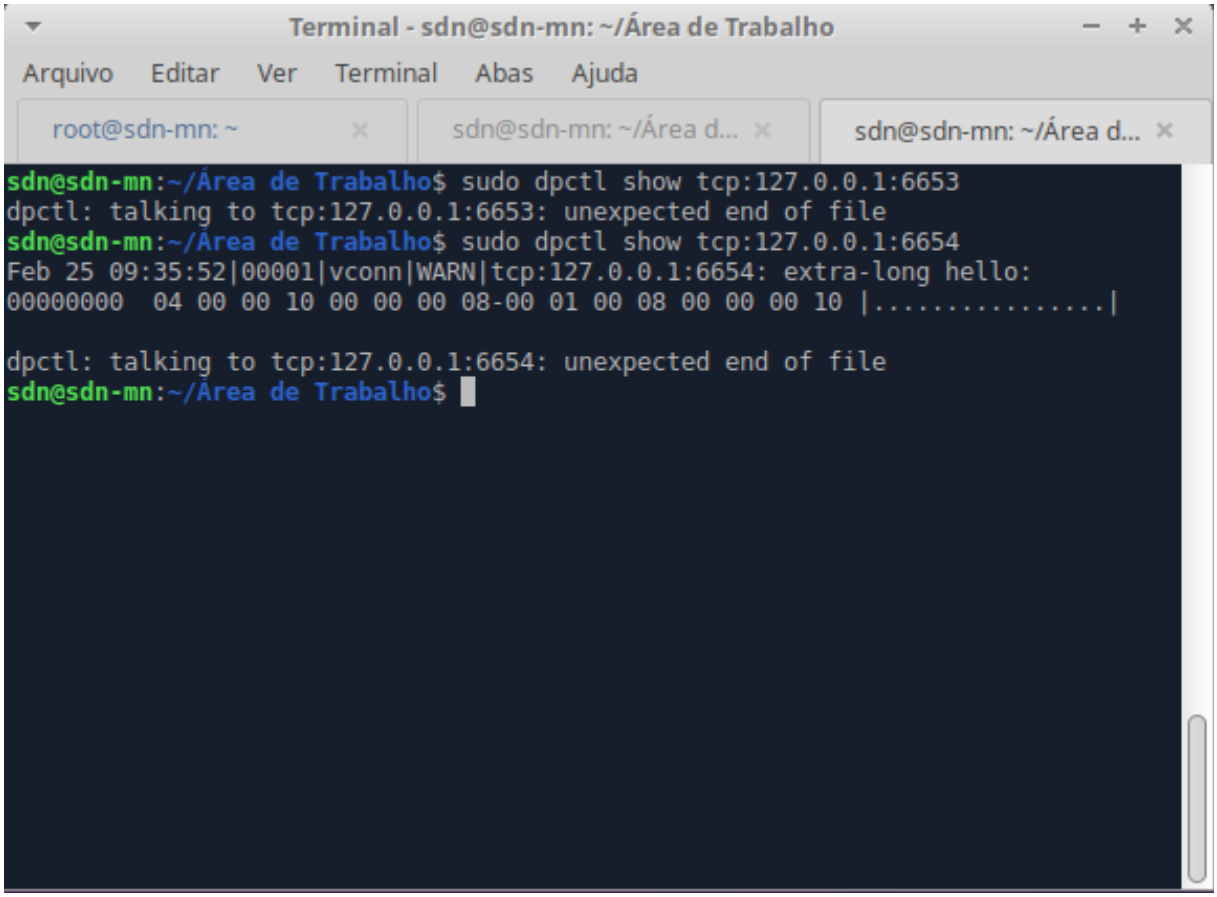
Através do uso do utilitário DPCTL, configurado diretamente no switch permite mostrar o encaminhamento de fluxos indicando a porta de entrada, a ação e a porta de saída.

Utilizando o comando a seguir, será possível identificar os pontos observados, com o uso do “dpctl” através da instrução, com o encaminhamento da porta de escuta do switch com a regra, todo o tráfego recebido na porta 2, definida a ação, precisará ser encaminhado para a porta de saída 1, e o inverso é aplicado.

Contudo, o OVS por definição conta com várias interfaces, que são utilizadas para a conexão com os hosts criados, respectivamente, na topologia SDN, com a acrescentamento de portas ao OVS, será necessário atentar-se a configuração de fluxo nas portas.

Imediatamente, executando o comando de consulta a tabela de fluxos é possível notar, conforme demostra a figura 12.

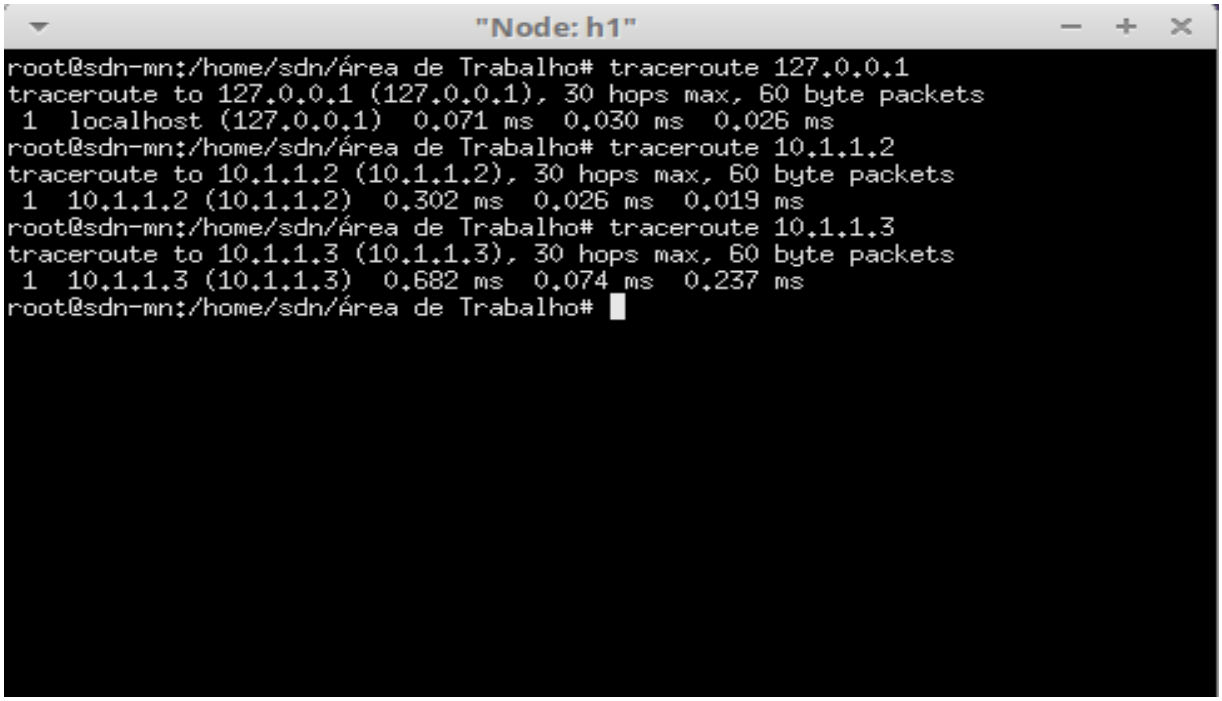
Figura 12 – Tabela de fluxo do ovs



Fonte: Verificação do ambiente de teste no mininet (2021) e adaptado pelo autor.

Vale ressaltar, que também será possível averiguar se o host1 já está acessível por meio dos outros elementos da rede, bem como a partir do mesmo poderá ser realizados testes de conectividade. Conforme a figura 13 é possível verificar o endereço de IP atribuído pelo Mininet na (interface loopback).

Figura 13 – Host1 em teste de conectividade



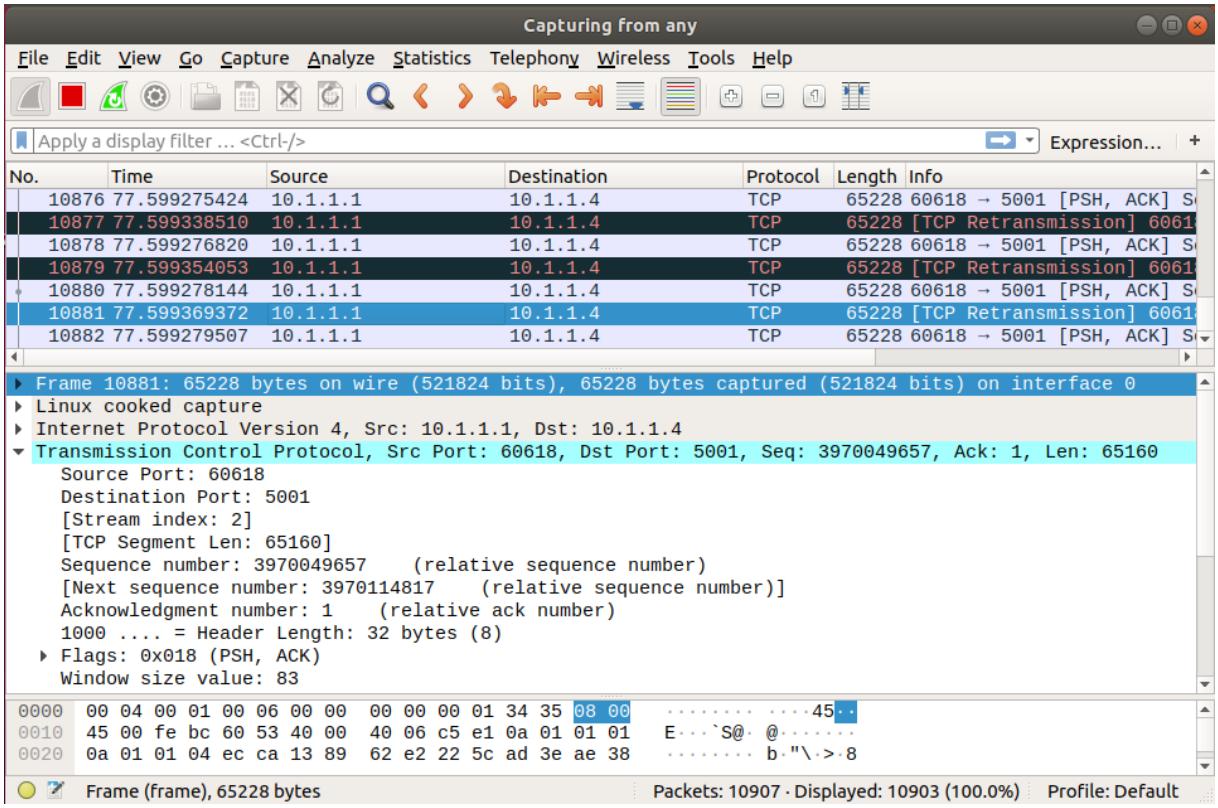
Fonte: Verificação do ambiente de teste no mininet (2021) e adaptado pelo autor.

6.6 Analisando o protocolo openflow

O Wireshark é um ferramenta para análise de redes, onde apresenta o teor de pacotes capturados em um determinado link (GUEDES, 2012).

Quando selecionado o tipo de protocolo, com o Mininet com a topologia montada, basta clicar na opção “start capture” que o programa será iniciado automaticamente. Para visualizar as requisições do protocolo OpenFlow, a rede SDN é descontínuada e todas as configurações atribuídas ao programa OVS serão removidas utilizando o comando (sudo mn –c) no console da máquina virtual. Executando o teste “ping” na máquina virtual na interface Loopback, o tráfego capturado não insinua em nenhuma solicitação do protocolo OpenFlow, conforme mostra a figura 14.

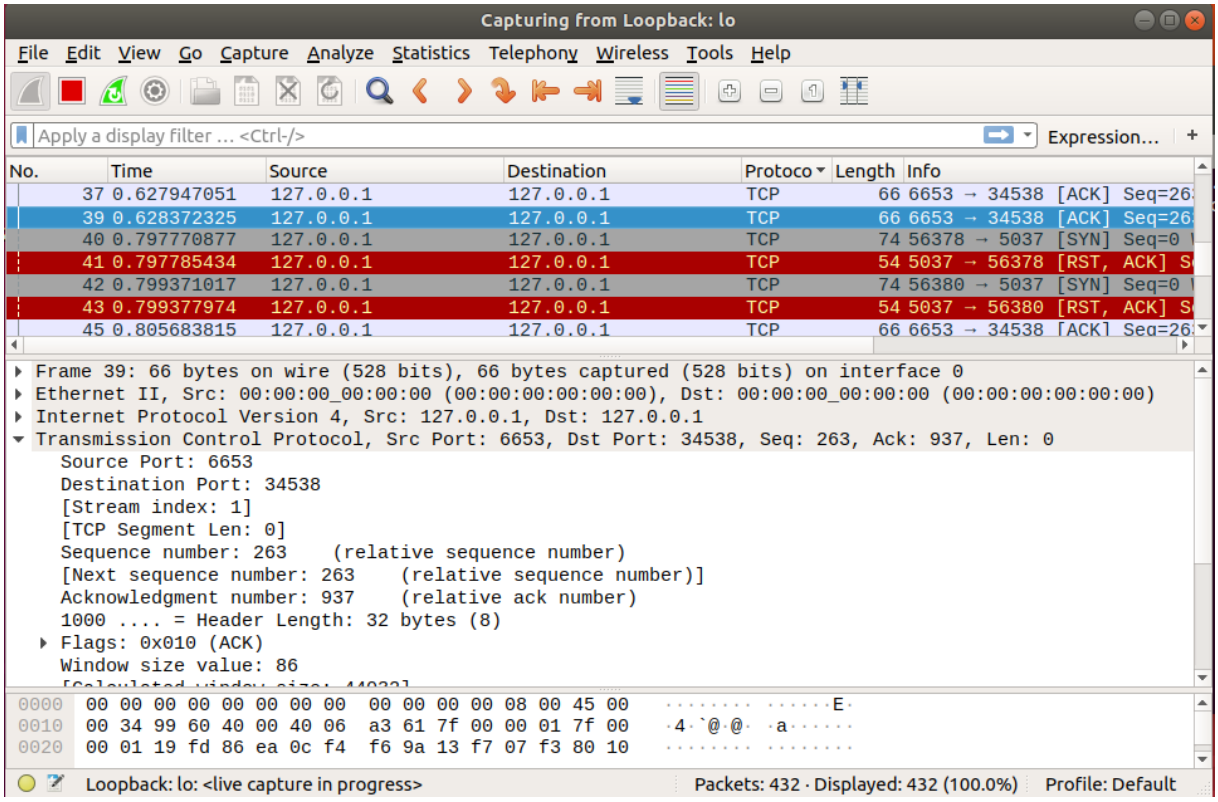
Figura 14 – Captura de tráfego com sdn inativa



Fonte: Utilizando o software de captura de pacotes wireshark (2021) e adaptado pelo autor.

Inicializado novamente a rede SDN Mininet, será possível notar na figura 15, a conexão do circuito emulado sendo constituída pela análise das requisições do protocolo TCP. Primeiramente será enviado o segmento de sincronismo SYN (Synchronizing), para sincronização dos lados do circuito TCP, professado do segmento de reconhecimento ACK (Acknowledgement) do sincronismo SYN, ou somente SYN, ACK (RECHIA, 2014).

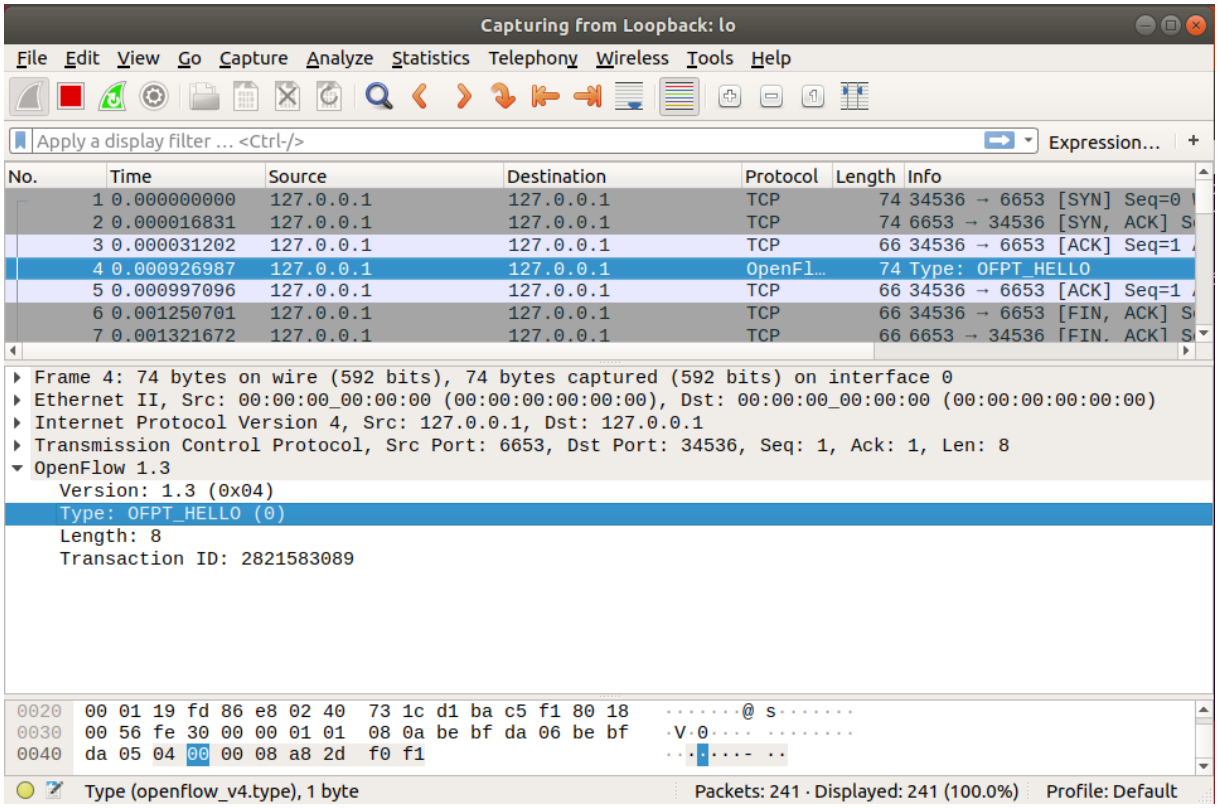
Figura 15 – Requisições tcp entre ryu e mininet



Fonte: Utilizando o software de captura de pacotes wireshark (2021) e adaptado pelo autor.

Portanto, serão iniciadas as “mensagens” do protocolo OpenFlow, a primeira será de ”Hello”, que pode ser observada na figura 16, onde ocorre a troca entre switch e controlador durante o estabelecimento da conexão, para produzir a versão do protocolo OpenFlow suportada, caso está transação falhe, será enviada uma mensagem do tipo “HelloFailed” (GUEDES, 2012).

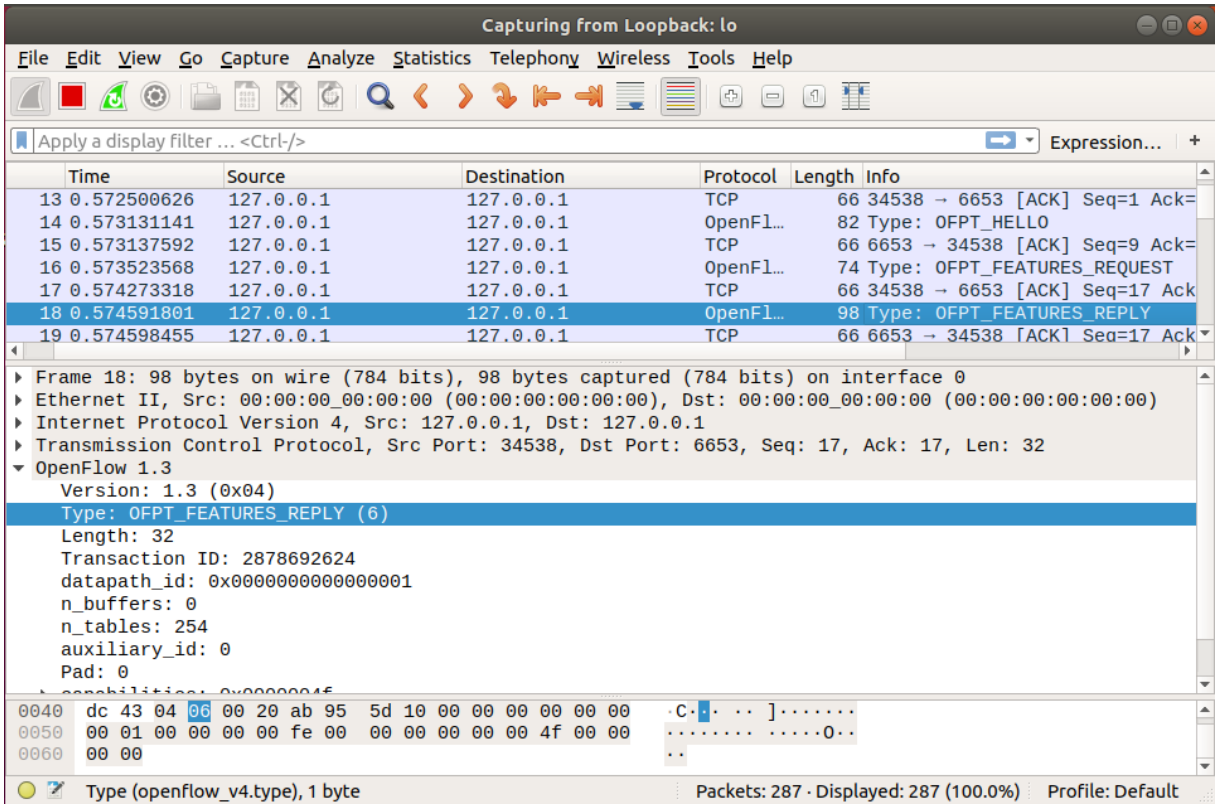
Figura 16 – Protocolo openflow: mensagem hello



Fonte: Utilizando o software de captura de pacotes wireshark (2021) e adaptado pelo autor.

Assim que o canal for estabelecido entre o switch e o controlador, a principal atividade é a definição das características do equipamento. O controlador enviará uma requisição (Feature_Request). Portanto, o switch reportará as informações através de uma mensagem (Feature_Reply).

Figura 17 – Protocolo openFfow: mensagem “feature_reply”



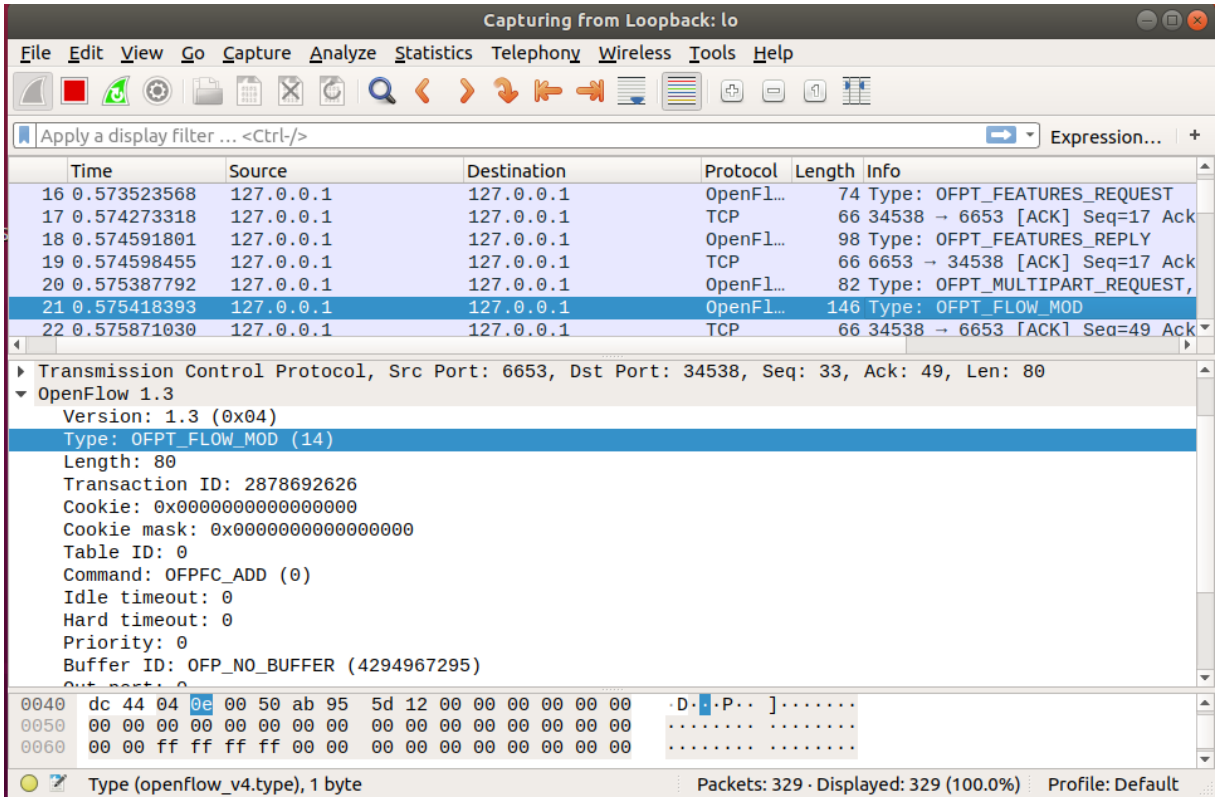
Fonte: Utilizando o software de captura de pacotes wireshark (2021) e adaptado pelo autor.

Conforme demonstrado na figura 17, é possível visualizar o exemplo da estrutura da mensagem, com os campos concernentes a versão, tipo, tamanho total e ID da requisição, além das informações específicas, sendo estas as características do equipamento.

Enquanto o controle dos fluxos ativos instalados faz a requisição “Stats_Request” e encaminha ao switch que retorna com a mensagem “Stats_Reply” em conjunto com as informações dos fluxos ativos, com os pacotes, contendo todas as mensagens, não serão “checados” pelo Wireshark. Um outro ponto importante que pode ser avaliado é o conteúdo da mensagem “Flow_Mod”, permiterá ao controlador alterar e criar um novo estado de fluxo (GUEDES, 2012).

Conforme a figura 18 será possível observar o conteúdo do pacote com a mensagem de modificação, com informações da configuração do fluxo.

Figura 18 – Protocolo openflow: mensagem flow_mod



Fonte: Utilizando o software de captura de pacotes wireshark (2021) e adaptado pelo autor.

Portanto, com a observação dos pacotes do protocolo OpenFlow, poderá ser iniciado uma avaliação desempenho do circuito híbrido SDN/IP para que haja um melhor compreensão, será apresentado na próxima seção a seguir.

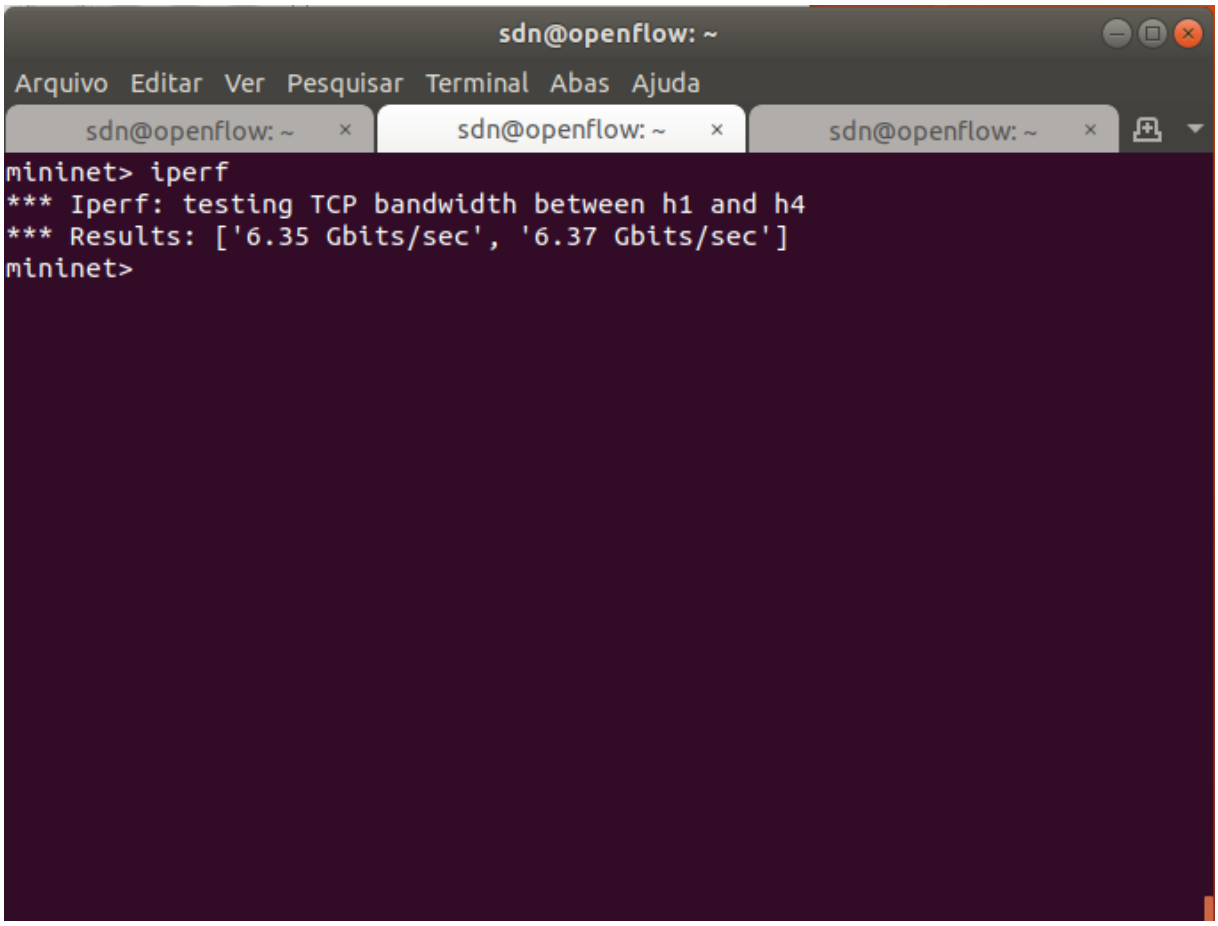
6.7 Testes de desempenho

De acordo com Rechia (2014), a máquina configurada com o sistema Mininet dispõe, na instalação, duas opções de switches - um executada em espaço usuário e outro, OVS, que é executa diretamente no espaço do kernel do sistema. Em testes realizados, comprovam que o switch executado em modo kernel é 13,3 vezes mais veloz, por causa dos pacotes permanecerem alocado na memória do sistema, enquanto o fluxo trafega pelo switch. Enquanto, no modo usuário, cada pacote necessitará trafegar pelo espaço de memória do usuário e ir em direção o espaço kernel e retornar a cada salto “hop”.

Em todas as emulações realizadas até o momento utilizaram um switch OVS e hosts virtuais para explicar a interoperabilidade de redes IP/SDN. Todavia, os testes utilizados com os hosts inicializados pela topologia do Mininet, bem como os fluxos criados foram removidos para evitar interferências da rede de produção nos resultados. Contudo, essencialmente avaliamos o funcionamento do comutador em modo usuário, com uma topologia composta por um switch e três hosts que será montado através do comando:

```
~$ # sudo mn
```

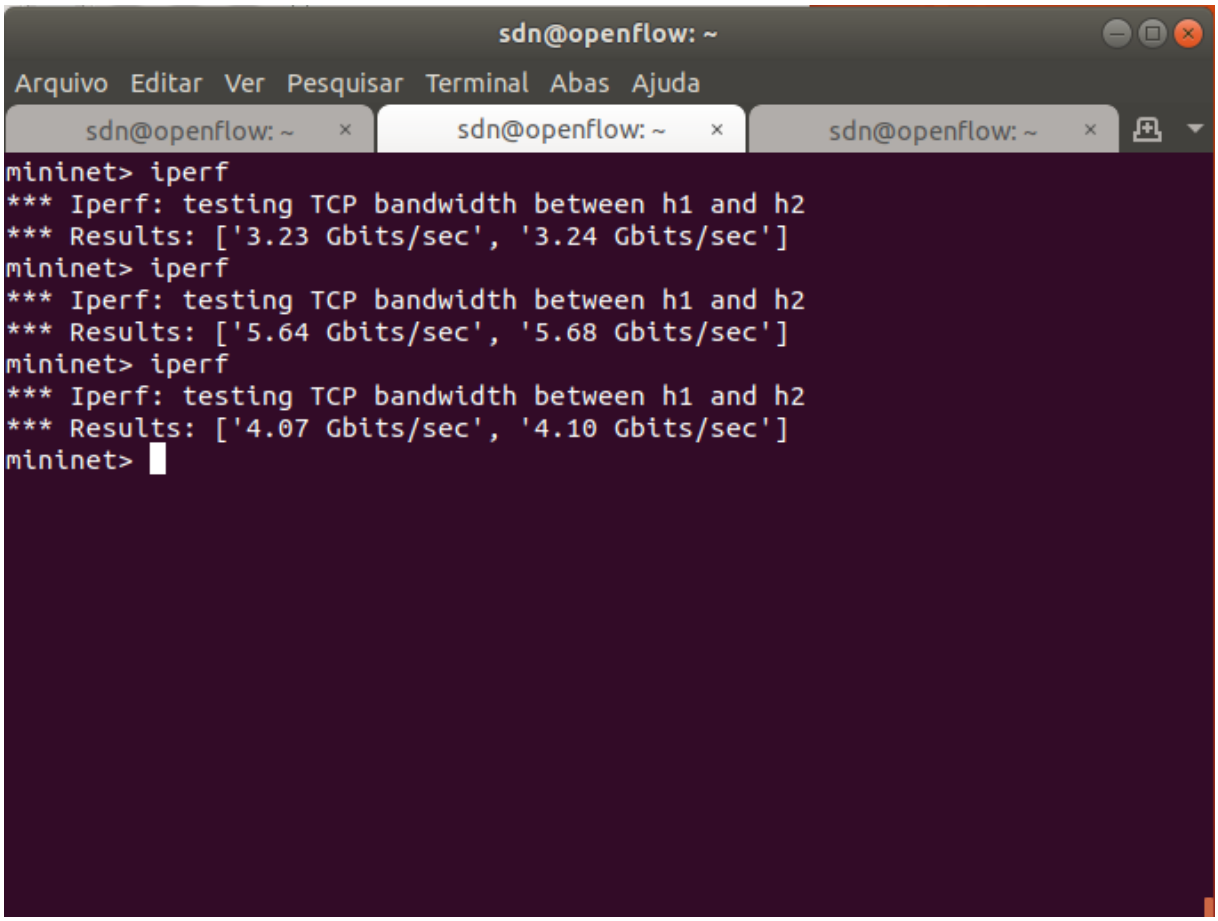
Figura 19 – Topologia sdn para testes de desempenho



Fonte: Utilizando o iperf para teste de desempenho de rede (2021) e adaptado pelo autor.

Utilizando o terminal do Mininet podemos utilizar a ferramenta “Iperf”, que promoverá a um dos hosts a geração de tráfego, enquanto outro host assume o papel de cliente. Contudo, já estabelecido o canal, serão geradas imersão de pacotes entre os hosts. O comando para iniciar a aplicação e o obter o resultado em um switch emulado no modo usuário pode ser observado na figura 20.

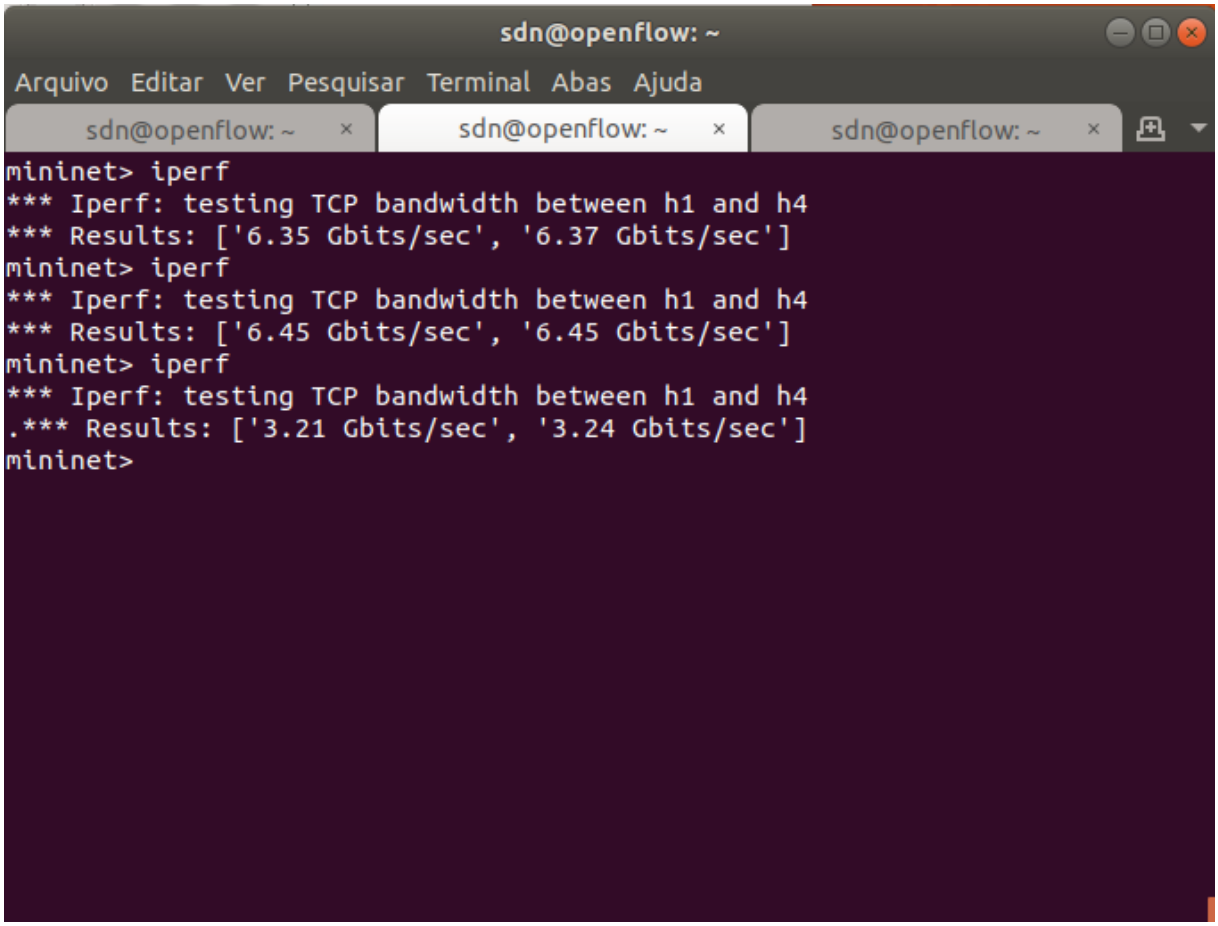
Figura 20 – Teste de desempenho no modo usuário



Fonte: Utilizando o iperf para teste de desempenho de rede (2021) e adaptado pelo autor.

Em seguida, após a limpeza do cache de configurações do teste anterior, foi novamente inicializada a topologia composta por três hosts, desta vez, utilizando o OVS (carregado diretamente no espaço Kernel). Conforme a figura 21 é apresentado o resultado do teste com “Iperf”, que comprova os resultados apontados por Rechia (2014).

Figura 21 – Teste de desempenho em espaço kernel



Fonte: Utilizando o iperf para teste de desempenho de rede (2021) e adaptado pelo autor.

Portanto, os resultados do switch executado em modo Kernel foram, em média, 5.330 vezes mais rápidos que o executado em modo usuário. Para determinar se a execução da máquina virtual sofre interferência do ambiente emulado, os testes foram novamente realizados diretamente no terminal, e os resultados não tiveram modificações.

Portanto, podemos confirmar que a perda de desempenho quando escolhido o “switch default” ocorre em consequência da imperatividade de interação entre o modo usuário e o modo kernel. Todos os testes realizados ocorreram em um laptop, utilizando o Oracle VM VirtualBOX, alocando as configurações, com processador Intel Core I7 2640M 2,8ghz (com um núcleo de processamento) e 4GB de RAM usando o controlador Ryu.

7 CONCLUSÃO

Durante o processo de pesquisa para o desenvolvimento deste trabalho nos possibilitou conhecer na prática os conceitos das redes definidas por software e o protocolo OpenFlow. Principalmente, no decorrer dos testes realizados e a coleta de informações no ambiente virtual. Por meio deste, foi possível entender de forma clara e objetiva o funcionamento da separação dos planos de controle e de dados dentro dos elementos de rede.

Vale ressaltar, que este novo modo de gerenciamento da rede possibilita um maior controle sobre o fluxo na tabela de encaminhamento no controlador e possibilita uma gama de alternativas para garantir maior segurança e redundância dos serviços de rede.

O controlador Ryu trouxe a possibilidade de nos permitir operar o funcionamento de uma rede, na qual nossa proposta era mostrar a integração do software como um sistema operacional dedicado para o funcionamento da rede. Como a prática foi toda implementada através de um ambiente virtualizado, foi necessário a utilização do kernel do sistema hospedeiro e alocação de memória para o funcionamento do dispositivo, para que possibilitasse a emulação da rede. Muitos controladores dispõem de uma interface gráfica através de um acesso via browser, no entanto, a nossa intenção não foi de trazer especificamente a visualização da topologia nesse sentido, mas de explorar suas funcionalidades. Portanto, isso garantiu que pudesse obter maior entendimento sobre o controle de fluxo e o funcionamento do protocolo OpenFlow.

A composição da rede emulada, integrando o emulador Mininet, Open vSwitch e o controlador Ryu, proporcionou um aprendizado no entendimento sobre os protocolos TCP/IP e OpenFlow. Entretanto, enfrentamos algumas dificuldades na implementação, principalmente na instalação dos pacotes e dependências requerida para o funcionamento da ferramenta. No entanto, as dificuldades podem ser um ganho significativo, pelo fato de buscar soluções frente aos problemas enfrentados, como experiencia acadêmica e no desenvolvimento da pesquisa. Observamos que este protocolo OpenFlow vem sendo constantemente atualizado, principalmente em

relação as distribuições Linux disponíveis no mercado. Isso nos forçou a buscar repositórios mais recentes e implementá-lo dentro do sistema operacional em teste.

Nosso trabalho, trouxe essa proposta, para que novas pesquisas possam ser estimuladas por outros acadêmicos, aprofundando sua aplicação e os conceitos ligados a redes definidas por software, podendo até abordar novas implementações em um ambiente real de testes. Possibilitando uma experiencia maior relacionada ao tráfego de rede e suas resiliências.

REFERENCIAS

AMORIM, Roberto José de. **O Uso do Protocolo OpenFlow em Redes Definidas por Software**. 2012. Monografia (Especialização em Configuração e Gerenciamento de Servidores e Equipamentos de Redes) – Departamento Acadêmico de Eletrônica, Universidade Tecnológica Federal do Paraná, Curitiba, 2012. Disponível em: <http://repositorio.roca.utfpr.edu.br/jspui/bitstream/1/1831/1/CT_GESER_II_2012_09.pdf>. Acesso em: 23 fev. 2020.

BERTHOLDO, Leandro. **Tecnologias, conceitos e serviços emergentes: Openflow.** 13º WRNP Workshop RNP. Universidade Federal do Rio Grande do Sul – UFGS – PoP – RS, 2012.

GUEDES, Dorgival O. **Redes Definidas por Software: uma abordagem sistêmica para o desenvolvimento de pesquisas em Redes de Computadores.** In: MINICURSOS DO XXX SIMPÓSIO BRASILEIRO DE REDES DE COMPUTADORES E SISTEMAS DISTRIBUÍDOS, 30, 2012, Ouro Preto, Livro Texto, Ouro Preto: Editora Sociedade Brasileira de Computação, 2012. P. 161-207. Disponível em: <http://homepages.dcc.ufmg.br/~mmvieira/cc/papers/minicurso-sdn.pdf>. Acesso em: 23 fev. 2020.

HARANAS, MARK. **Os 10 Principais Líderes de Mercado de SDN em Data Centers e em Empresas.** Disponível em: <http://www.crn.com/slide-shows/networking/300079644/top-10-sdn-market-leaders-in-the-data-center-and-enterprise-in-2016.htm>. Acesso em 23 fev. 2020.

LOPEZ., F. **Arquitetura e Protótipo de uma Rede SDN-OpenFlow para Provedor de Serviço.** Dissertação de Mestrado em Engenharia Elétrica, Publicação PPGEE.DM-562/14, Departamento de Engenharia Elétrica, Universidade de Brasília, Brasília, DF, 59p, 2014.

NOBRE, Tito Sérgio Martins Pereira. **SDN em Rede de Transporte Ópticas.** 2014. 107 f. Dissertação (Mestrado em Engenharia Informática) - Faculdade de Ciências, Universidade de Lisboa, 2014. Disponível em: <http://docs.di.fc.ul.pt/bitstream/10451/15936/1/ulfc112524_tm_Tito_Nobre.pdf>. Acesso em: 23 fev. 2020.

OpenFlow v. 1.3. **OpenFlow king Foundation v. 1.3**. OpenFlow switch Open specification Networ-version. 1.3.0. Disponível em: <https://www.opennetworking.org/images/stories/downloads/specfication/openflow-spec-v1.3.0.pdf>. Acesso em: 05 fev. 2021.

RECHIA, Felipe Stall. **Estudo de Redes Definidas por Software e sua Implantação em Redes Corporativas.** 2014. 110 f. Monografia (Especialização em Teleinformática e Redes de Computadores) – Pró Reitoria de Pesquisa de Pós-

Graduação, Universidade Tecnológica Federal do Paraná, Curitiba, 2014. Disponível em: <http://repositorio.roca.utfpr.edu.br/jspui/bitstream/1/3864/1/CT_TELEINFO_2013_1_04.pdf>. Acesso em: 23 fev. 2020.

ROTHENBERG, Christian Esteves. **OpenFlow e redes definidas por software: um novo paradigma de controle e inovação em redes de pacotes.** Cad. CpqD Tecnologia, Campinas, jul. 2010. Disponível em: <http://www.cpqd.com.br/cadernosdetecnologia/Vol7_N1_jul2010_jun2011/pdf/artigo6.pdf>. Acesso em: 23 fev. 2020.

APÊNDICE A - PREPARANDO O SISTEMA OPERACIONAL

- Linguagem: Português
- Instalar o Xubuntu
- Layout: Português (Brasil)
- Variante: Português (Brasil)
- Atualizações e outros softwares: Baixar atualizações enquanto instala o Xubuntu
- Atualizações e outros softwares: instalar softwares de terceiros
- Configuração de armazenamento guiada: Apagar o disco e instalar o Xubuntu
- Configuração de Região: América/São Paulo
- Seu do host: sdn
- Nome do computador: sdn-mn
- Nome de usuário: sdn
- Senha: sdn
- Confirmação da senha: sdn
- Configurado para entrar automaticamente: selecionado

Após definir as configurações para instalação do sistema operacional, será necessário remover alguns softwares desnecessários. Remoção dos softwares ocorrerá via terminal com os seguintes comandos:

```
~$ sudo apt-get purge -y libreoffice-*
~$ sudo apt-get purge -y parole xfburn pulseaudio
~$ sudo apt-get purge -y gimp ristretto
~$ sudo apt-get purge -y gnome-mines gnome-sudoku sgt-puzzles
~$ sudo apt-get purge -y thunderbird
~$ sudo apt-get purge -y transmission-* pidgin
~$ sudo apt-get -y autoremove
```

Para explorar os recursos de instalação no VirtualBox, será necessário insira o “CD do Guest Additions” para instalar as adições de convidados na máquina virtual no host do sistema.

Realizando o seguinte passo, no menu dispositivos > “Inserir imagem de CD para adicionais de convidados…”, em seguida será feito o downloads dos arquivos, para posteriormente executar com os seguintes comandos:

```
~$ cd /media/ubuntu/Vbox _Gas_6.1.16
~$ sh ./autorun.sh
```

Ao concluir o procedimento, será necessário reiniciar o sistema.

```
~$ reboot
```

Como medida de segurança, é recomendado realizar a exportação da appliance. Em seguida, é necessário criar uma pasta compartilhada no sistema operacional do host.

```
~$ mkdir ~/SHARED_DIR
```

Na máquina virtual, foi adicionado o usuário ‘sdn’ no grupo vboxsf para permitir o acesso ao diretório.

```
~$ sudo usermod -a -G vboxsf sdn
```

Em seguida, utilizando o menu do VirtualBox selecionamos a seguinte opção para o diretório criado. Em Dispositivos > Pastas compartilhadas > Configurações da pasta compartilhada. Será necessário selecionar as opções: auto-montagem e tornar permanente.

Em seguida, será necessário selecionar a opção: Dispositivos > Área de transferência compartilhada > Bidirecional; Dispositivos > Arrastar e Soltar > Bidirecional.

Em seguida será necessário instalar o openssh (acesso remoto via ssh):

```
~$ sudo apt-get install -y openssh-server git libncurses-dev libssl-dev
```

Somente após concluir as configurações anteriores, que será preciso realizar a atualização do sistema. Ao concluir será necessário reiniciar o sistema:

```
~$ sudo apt-get -y update && sudo apt-get -y upgrade
~$ sudo shutdown -r now
```

Nesta etapa, será importante instalar os pacotes[2] a seguir:

```
~$ sudo apt-get install -y build-essential mtr tcpdump lynx iperf tshark fping
wireshark net-tools curl openvswitch-switch git python3-pip python3-scapy
```

Uma observação: No momento da instalação e configuração do wireshark-common, é recomendado escolher a opção “yes” ou “sim” permitindo que qualquer usuário tenha permissão para executar a captura pacotes. Em seguida, será necessário adicionar o usuário ‘sdn’ ao grupo threadshark, conforme os comandos a seguir:

```
~$ sudo usermod -a -G wireshark sdn
```

2 O Open vSwitch é um switch virtual que implementa o protocolo OpenFlow. Algumas de suas principais características são: Implementado em software; Plano de dados executa dentro do kernel do Linux; Plano de controle executa em nível de usuário; e Pode operar como um switch Ethernet padrão.
Wireshark é uma ferramenta que analisa o tráfego de rede, e o organiza por protocolo, com funcionalidades parecidas com tcpdump, e com mais possibilidades de utilização de filtros.
IPERF é útil para realizar teste de largura de banda, fazendo injeção de pacotes (TCP quanto UDP) na rede.
cURL é um projeto que fornece uma biblioteca e uma ferramenta de linha de comando para transferir dados usando vários protocolos.

Serão necessários realizar algumas configurações no wireshark, seguindo os seguintes comandos:

```
~$ sudo chgrp wireshark /usr/bin/dumpcap
~$ sudo chmod 4711 /usr/bin/dumpcap
~$ sudo setcap cap_net_admin,cap_net_raw=eip /usr/bin/dumpcap
```

APÊNDICE B - INSTALAÇÃO DO MININET

A instalação do Mininet necessitará que todas as dependências e arquivos estejam instalados para a execução do Mininet. O procedimento mais indicado para a instalação é utilizar os pacotes direto do “github”, realizando as instalações conforme a seguir:

```
~$ git -C /home/sdn/ clone git://github.com/mininet/mininet
~$ sudo /home/sdn/mininet/util/install.sh -a
~$ sudo mn --version
2.3.0b2
```

APÊNDICE C - INSTALAÇÃO DO CONTROLADOR RYU

Será instalada a compilação mais recente do Ryu. Para realizar a instalação, será necessário realizar os seguintes passos no terminal:

```
~$ pip3 install netaddr six pbr
~$ pip3 install ryu
```

Existe uma outra alternativa de instalação, utilizando a última versão estável no repositório do sistema operacional, realizando o procedimento conforme a seguir (vale ressaltar, que não é indicado realizar os dois procedimentos de instalação em conjunto, recomendando a escolha de somente um procedimento):

```
~$ sudo apt-get install -y python3-ryu
```

Por padrão o diretório onde será depositado os arquivos do aplicativo Ryu, estará localizado no seguinte caminho:

```
~/.local/lib/python3.8/site-packages/ryu
```

Vale ressaltar, que o caminho apontado não será regra geral, dependendo a distribuição Linux e sua respectiva versão, este caminho poderá ser alterado.

Neste caso, os arquivos para execução do controlador Ryu podem ser localizados conforme a seguir:

```
~/.local/bin
~$ ls ~/.local/bin | grep ryu
ryu
ryu-manager
```