JULIANO ALVES DE LIMA PEREIRA

DIFERENCIAÇÃO DE SERVIÇOS USANDO O CONCEITO DE REDES DEFINIDAS POR SOFTWARE, COM FOCO NO PADRÃO OPENFLOW

LAVRAS – MG

2014

JULIANO ALVES DE LIMA PEREIRA

DIFERENCIAÇÃO DE SERVIÇOS USANDO O CONCEITO DE REDES DEFINIDAS POR SOFTWARE, COM FOCO NO PADRÃO OPENFLOW

Trabalho de Conclusão de Curso de Graduação apresentado ao Colegiado do Curso de Bacharelado em Ciência da Computação, para obtenção do título de Bacharel

Orientador
Prof. Dr. Neumar Costa Malheiros

Co-Orientador
Prof. Me. Hermes Pimenta de Moraes Júnior

LAVRAS – MG

2014

JULIANO ALVES DE LIMA PEREIRA

DIFERENCIAÇÃO DE SERVIÇOS USANDO O CONCEITO DE REDES DEFINIDAS POR SOFTWARE, COM FOCO NO PADRÃO OPENFLOW

Trabalho de Conclusão de Curso de Graduação apresentado ao Colegiado do Curso de Bacharelado em Ciência da Computação, para obtenção do título de Bacharel.

APROVADA em 28 de novembro de 2014.

Dr. Luiz Henrique Andrade Correia

Gláucio Ferreira Loureiro

Dr. Neumar Costa Malheiros (Orientador)
MSc. Hermes Pimenta de Moraes Júnior (Coorientador)

LAVRAS-MG
Novembro/2014

Dedico este trabalho à toda minha família e toda comunidade acadêmica!

AGRADECIMENTOS

Agradeço primeiramente à Deus por ter o privilégio de concluir o ensino superior, agradeço meus pais pelo amor e apoio incondicional e aos meus professores por todo conhecimento passado ao longo destes anos.

RESUMO

Redes Definidas por Software constituem hoje um novo paradigma em redes de computadores, que vem ganhando grande atenção da comunidade acadêmica e do mercado. O principal modelo deste tipo de rede é o padrão OpenFlow. A principal característica do padrão OpenFlow é a separação clara entre o plano de dados e o plano de controle em uma rede, tornando a rede programável e fácil de administrar. Este trabalho mostra como diferenciar dois tipos de serviços em uma rede local, mostrando as características de uma rede SDN e a vantagem de se utilizar o padrão OpenFlow.

Palavras-Chave: Redes Definidas por Software, diferenciação de serviços, Redes de computadores

SUMÁRIO

LISTA DE FIGURAS

1 INTRODUÇÃO

Hoje é impossível que a sociedade viva sem utilizar computadores e todos os benefícios que eles trazem. Difícil de imaginar também um mundo sem a conexão entre estes computadores, o usuário espera que não haja uma conexão lenta para que seu uso da rede seja eficiente. A atual forma como a rede de computadores é apresentada, apesar de versátil, mostra muitas limitações na forma como trata seus dados.

A arquitetura de rede mais utilizada para conectar computadores, servidores e afins é a arquitetura TCP/IP (KUROSE; ROSS, 2009), que foi difundida em 1983, uma arquitetura robusta para a época, visto que é usado até hoje, porém encontra-se defasada. Esta versão da arquitetura TCP/IP não prevê o crescimento das redes, um possível esgotamento de endereços IP, problemas relacionados à segurança e prioridade na entrega de determinados tipos de pacotes, o que é essencial para a eficiência da interação entre os computadores. Estas limitações foram remediadas com a criação de técnicas paliativas, como o DHCP (Dynamic Host Configuration Protocol) (KUROSE; ROSS, 2009) e o NAT(Network Address Translation)(KUROSE; ROSS, 2009). Técnicas que trazem melhorias momentâneas, mas não eliminam a causa real.

Com todos estas limitações houve a necessidade da criação de um novo protocolo de rede. O protocolo IPv6 (KUROSE; ROSS, 2009), sucessor do IPv4, foi criado com o intuito de resolver algumas questões pendentes da versão antiga: escalabilidade, segurança, configuração e administração de rede, suporte a QoS (qualidade de serviço), mobilidade, políticas de roteamento e a transição eficaz para a nova versão. Entre estas questões a principal é a escalabilidade, a versão 4 do protocolo IP identifica um host por um número de 32 bits, o IPv6 traz um aumento significativo deste endereço, 128 bits. Há outras características relevantes, que este protocolo traz, tais como um cabeçalho mais simples, com campos de extensão que podem ser utilizados e o suporte a QoS (qualidade de serviço).

Outra grande limitação do modelo atual do tratamento de dados na rede de computadores é a não diferenciação de serviços. A solução que está surgindo e ganhando grande atenção da comunidade acadêmica é o conceito de Redes Definidas por Software, do inglês Software Defined Networks(SDN). A ideia deste conceito é ter um software que permita identificar pacotes que pertençam a determinado fluxo, para os quais podem ser requeridos tratamentos especiais.

O principal destaque do conceito SDN é a implementação de um padrão aberto para Redes Definidas por Software, o OpenFLow (GUEDES *et al.*, 2012). Com ele, os elementos de encaminhamento oferecem uma interface de programação simples, que lhes permite estender o acesso e controle da tabela de rotas, utilizada pelo hardware para determinar o próximo passo de cada pacote recebido nos elementos da rede (GUEDES *et al.*, 2012).

Com o uso da diferenciação de serviços usando o padrão OpenFlow abre-se um leque de possibilidades para uso em redes comerciais. As técnicas que fazem este trabalho são todas escondidas em dispositivos "caixas-preta"e não ficam acessíveis para o estudo e aperfeiçoamento na comunidade acadêmica.

1.1 Objetivos

O objetivo geral do projeto é propor formas de diferenciação de serviços usando o conceito de redes definidas por software, utilizando o switch virtual Openvswitch e o controlador floodligth, em um ambiente simulado.

O ponto crucial do trabalho é fazer o OpenvSwitch trabalhar como um switch programável, podendo controlar os fluxo na rede de acordo com prioridades inseridas pelo administrador da rede. Os objetivos específicos deste processo são:

- Estudar os conceitos e aspectos relacionados a redes Definidas por Software.
- Estudar os conceitos e aspectos relacionados a redes OpenFlow.
- Estudar o funcionamento do OpenvSwitch.

- Levantar possibilidades de diferenciação de serviços usando o padrão OpenFlow.
- Elaborar e construir ambiente de testes para as soluções implementadas.
- Validar o projeto efetuando testes em redes simuladas.

Este trabalho realizarou uma revisão bibliográfica sobre a estrutura atual de redes de computadores, de Redes Definidas por Software com foco no padrão OpenFlow. Foi feita uma especificação dos componentes de uma rede OpenFlow. Serão virtualizadas máquinas em um computador do tipo servidor para os testes de envios de pacotes usando o padrão OpenFlow e sem o padrão. A comparação entre os dois cenários de envio de pacotes foi feita através do tempo gasto pelos pacotes para chegar de uma máquina virtualizada a outra e de como os dados são tratados em uma rede utilizando os conceitos SDN.

1.2 Justificativa

Nas redes atuais, a diversidade de aplicações e serviços sendo oferecidos pelos servidores é muito grande, o que gera um grande volume de fluxos de tráfego de dados. Para os administradores de redes, é interessante priorizar alguns serviços que não podem parar, ou que a perda nestes serviços seja a menor possível. Uma rede programável, é atrativa neste caso, visto que, ora incluir determinadas regras e outras vezes exclui e incluiu outras, isso é um fator que provê dinamicidade a atual rede de computadores e facilita o trabalho dos administradores de rede. Os atuais roteadores fazem este serviço, porém seu custo é elevado. O padrão OpenFlow propõe uma solução a este problema: faz a separação clara entre os planos de dados e controle.

O presente trabalho avalia uma solução para este problema, aplicando regras OpenFlow aos fluxos de uma rede simulada usando o padrão OpenFlow, utili-

zando o switch Openvswitch e o controlador Floodligth como componentes desta rede.

1.3 Estrutura do Trabalho

O restante deste texto está organizado da seguinte forma. O Capítulo 2 descreve as bases teóricas de apoio para a escrita e desenvolvimento do trabalho. O Capítulo 3 descreve a metodologia da realização do trabalho e descreve os testes realizados. O Capítulo 4 apresenta os resultados e discussão sobre os mesmos. O Capítulo 5 apresenta as conclusões deste trabalho e trabalhos futuros que possam ser desenvolvidos. O Apêndice A trata dos detalhes técnicos para a implementação dos cenários de testes.

2 REFERENCIAL TEÓRICO

Redes Definidas por Software é uma nova arquitetura de redes de computadores que propõe uma forma alternativa para gerência da rede. Esta arquitetura surgiu devido às limitações encontradas na arquitetura atual.

2.1 Redes de Computadores

A Internet é uma rede de computadores que inter conecta milhares de dispositivos computacionais ao redor do mundo (KUROSE; ROSS, 2009). Esta é composta de camadas, vistas de cima para baixo são elas: aplicação, transporte, rede, enlace e física. A camada de aplicação é onde residem as aplicações e seus protocolos. A camada de transporte da Internet suporta mensagens da camada de aplicação entre os lados do cliente e servidor de uma aplicação. A camada de rede da Internet é responsável pela movimentação de pacotes de uma máquina para outra, conhecidos como datagramas. Para levar um pacote de um nó (sistema final ou comutador de pacotes) ao nó seguinte na rota, a camada de rede depende de serviços da camada de enlace. O trabalho da camada física é movimentar os bits individuais que estão dentro de um quadro de um nó para o seguinte (KUROSE; ROSS, 2009) .

Apesar do modelo atual da Internet funcionar bem e de ter evoluído muito, sua evolução não foi suficiente nos últimos vinte anos. O modelo atual tornou-se comercial e os equipamentos e softwares são fechados, não permitindo um grande avanço nesta área, causando assim o engessamento da Internet (KUROSE; ROSS, 2009).

A popularização da Internet trouxe consigo o problema de que a sociedade em sua grande parte depende dos serviços oferecidos. Por isso qualquer atraso ou contratempo neste seguimento pode se tornar um grande prejuízo, por exemplo, um banco ficar fora do ar durante muito tempo causando prejuízos. Por essa grande dependência uma pausa na Internet de modo geral pode gerar problemas, tornando

a rede pouco flexível. Pesquisas estão sendo feitas sobre novas arquiteturas para a Internet, como a "*new arch*"(GREENBERG *et al.*, 2005). Porém são soluções que a Internet deveria ficar inativa por determinado período para que essas arquiteturas fossem implantadas, o que seria inviável.

O conceito de Redes Definidas por Software não traz consigo este problema, trata-se de mudar apenas formas de tratamento dos dados em pontos estratégicos da rede, não interferindo assim nas aplicações do usuário final. Redes Definidas por Software abrem novas perspectivas em termos de abstrações, ambientes de controle e aplicações de rede que podem ser desenvolvidas de forma simples e livre dos modelos atuais. (GUEDES *et al.*, 2012)

2.2 Redes Definidas por Software

SDN constitue um novo paradigma para o desenvolvimento de pesquisas em redes de computadores (GUEDES *et al.*, 2012). Esta abordagem de rede surge da necessidade de evolução de tratamento dos dados da rede sem interromper o tráfego de dados. Este paradigma apresenta a diferenciação de serviços de dados, em que a rota que estes irão tomar é controlada por um software que busca o melhor caminho que este dado pode percorrer até chegar ao seu destino final.

O principal padrão desta adordagem de rede é o OpenFlow. Este padrão tem como principal objetivo permitir que se utilize equipamentos de rede comerciais para pesquisa e experimentação de novos protocolos de rede, em paralelo com a operação normal das redes. Isso é conseguido com a definição de uma interface de programação que permite ao desenvolvedor controlar diretamente os elementos de encaminhamento de pacotes presentes (GUEDES *et al.*, 2012). OpenFlow define um protocolo-padrão para determinar as ações de encaminhamento de pacotes em dispositivos de rede. As regras e ações instaladas no hardware de rede são de responsabilidade de um elemento externo denominado *controlador* (ROTHENBERG MARCELO RIBEIRO NASCIMENTO, 2010 - 2011).

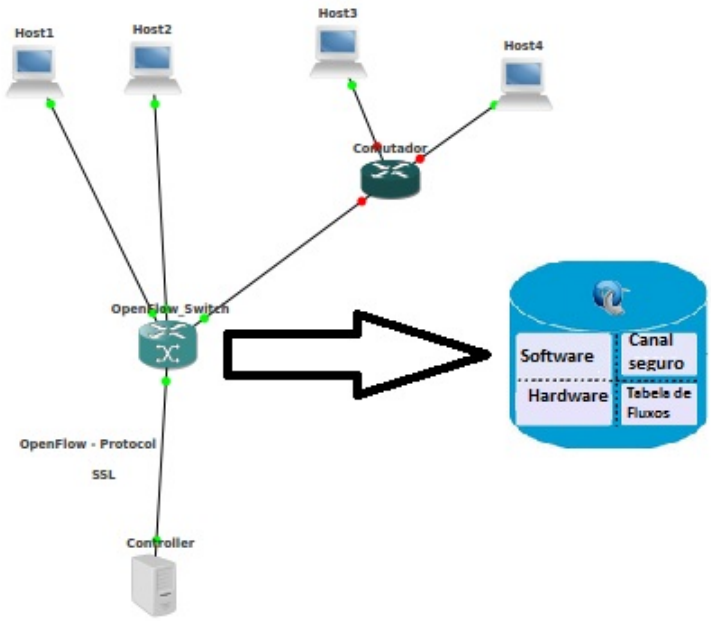

Figura 2.1: Topologia simples de uma rede SDN.

A Figura 2.1 mostra como funciona a estrutura básica de uma rede utilizando o padrão OpenFlow. Uma caracteristica importante do padrão OpenFlow é a separação clara entre o plano de dados e de controle. O plano de dados cuida do encaminhamento de pacotes com base em regras simples e o plano de controle é responsável por tomar decisões e adicionar ou remover as regras na tabela de controle de fluxos (*Flow table*) de acordo com o objetivo desejado.

2.2.1 Componentes de uma rede SDN

A principal característica de uma rede SDN é a presença de um sistema de controle para o gerenciamento da rede. Este sistema tem uma interface de programação bem definida e com suas características próprias, disponibiliza métodos para aplicativos de rede que realizam o controle do mecanismo de encaminhamento e comutação dos pacotes. Há também um ou vários controladores que ocultam os

detalhes internos da rede, centralizam a comunicação com os elementos programáveis e oferece uma visão unificada da rede. Com a visão geral e global da rede os controladores podem trabalhar de forma distribuída, por meio da divisão dos elementos que analisam a rede ou de algoritmos distribuídos, o que gera inúmeras novas possibilidades de controle na rede. São exemplos de controladores: NOX, POX, Beacon e Floodlight (usado no presente trabalho).

A figura 2.1 apresenta o exemplo de uma rede simples SDN, é mostrado os principais componentes. Na parte inferior da figura, encontra-se o controlador, ligado a ele, o switch (openvswitch), que é o software de gestão da rede, é ligado aos comutadores, servidores e usuários finais. O switch faz a função de *datapath*, ou seja, operações de encaminhamento de pacotes. A figura 2.2 mostra uma rede OpenFlow mais estruturada, com switch OpenFlow que é responsável pelo controle do gargalo na rede obtendo informações advindas do controlador. O controlador e o switch se comunicam através de um protocolo OpenFlow (GUEDES *et al.*, 2012). Esta é a grande diferença de uma rede SDN com uma rede comum, separação entre o datapath e as decisões de roteamento de alto nível, ou seja, separação clara entre o plano de controle e o plano de dados, o que traduz em um melhor gerenciamento da rede.

A grande abstração utilizada pelo OpenFlow que lhe confere uma vantagem na administração da rede é o conceito de fluxo. Cada fluxo de dados é identificado por um conjunto de campos do cabeçalho dos pacotes. Os fluxos são armazenados em uma tabela de fluxos. Ao chegar no switch, os campos do cabeçalho do pacote são comparados aos fluxos da tabela de encaminhamento e o switch dá ao pacote o tratamento especificado. Por exemplo, pacotes que pertencem a certo IP e determinada porta devem ser descartados. A figura 2.3 mostra os campos disponíveis no padrão OpenFlow para a criação de um fluxo.

Neste trabalho foi utilizado o OpenvSwitch, um switch virtual multi camadas que dá suporte ao protocolo OpenFlow.

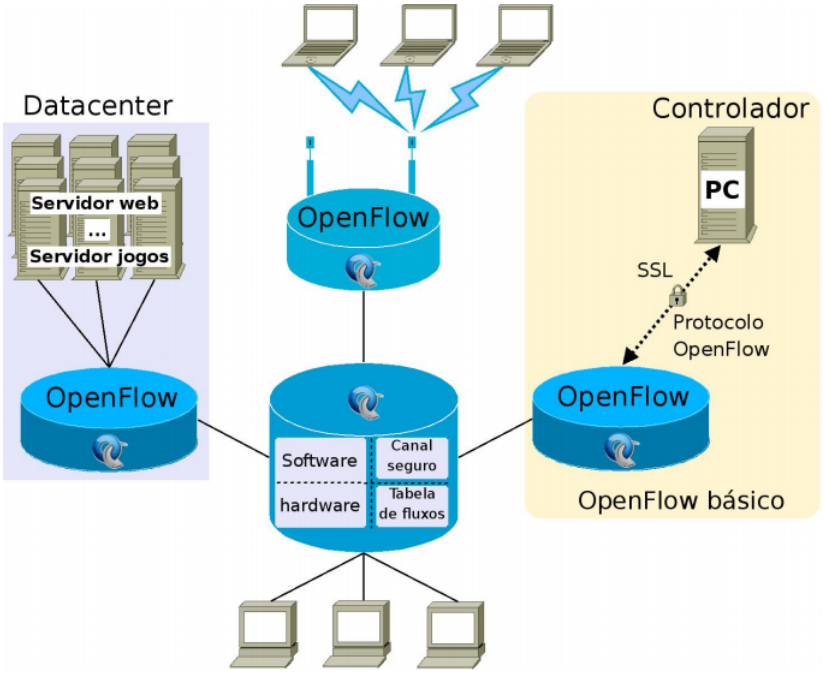

Figura 2.2: Exemplo de uma rede com OpenFlow habilitado (ROTHENBERG MARCELO RIBEIRO NASCIMENTO, 2010 - 2011).

		Ethernet			IP			TCP	
In_port	VLAN ID	SA	DA	Type	SA	DA	Proto	Source	destination

Figura 2.3: Campos do cabeçalho de um fluxo de um switch OpenFlow

O controlador utilizado no projeto foi o Floodlight. Este controlador trabalha com switchs tanto reais quanto virtuais que usam o protocolo OpenFlow e tem licença aberta, podendo ser utilizado para qualquer finalidade. É desenvolvido por uma comunidade aberta de desenvolvedores e é amplamente utilizado na comunidade acadêmica. (GUEDES *et al.*, 2012) A Figura 2.4 mostra o controlador Floodlight e como ele faz a conexão com os elementos de rede. O controlador tem um papel fundamental na decisão de como será tratada a rede.

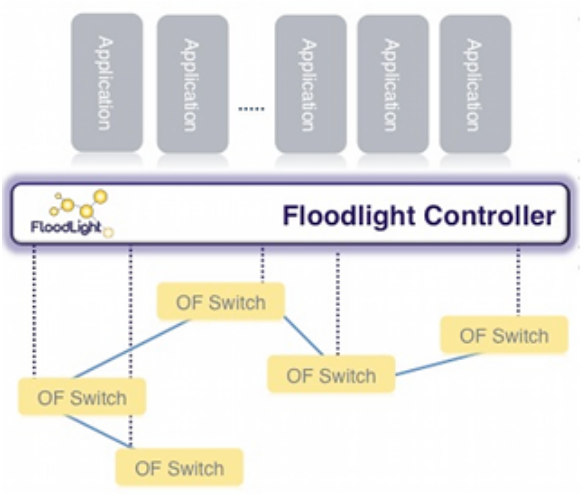

Figura 2.4: Controlador Floodlight(PARRAGA, 2013)

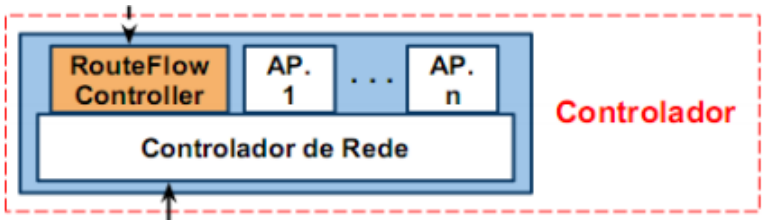

Figura 2.5: Exemplo básico de um controlador OpenFlow

2.3 Criação de Fluxos

A principal abstração do padrão OpenFlow é a criação de regras para controle dos fluxos de tráfego na rede. Estas regras definem como os fluxos serão tratados em relação à diversas variáveis. As regras são adicionadas a tabelas de fluxos (*Flow table*) que estão no controlador e são consultadas pelo switch OpenFlow ao chegar um pacote. Na criação das regras do fluxo há duas características imprescindíveis a definir:

- Bridge [1] em que a regra será aplicada;
- Ação de aplicação na regra.

E há caracteríticas incrementais na regra:

- Prioridade;
- Porta de entrada;
- Endereço MAC de origem;
- Endereço MAC de destino;
- Protocolo do serviço;
- Endereço IP de origem;
- Endereço IP de destino;
- Porta de origem;
- Porta de destino;

[1] Termo utilizado em informática para designar um dispositivo que liga duas ou mais redes informáticas que usam protocolos distintos ou iguais ou dois segmentos da mesma rede que usam o mesmo protocolo

Exemplo da criação de uma regra OpenFlow:

ovs-ofctl add-flow Bridge $priority = 10$,in_$port = 1$,dl_$type = 0x0800$,$actions =$ *normal*

Esta regra é criada e adicionada a tabela de encaminhamento do switch, ela define que todo pacote que pertencer a porta 1 do switch, for no tipo ethernet terá prioridade 10 e terá seu encaminhamento normal.

- ovs-ofctl - prepara o switch para alterações relacionadas à controle de regras OpenFlow
- add-flow - adição de regras no fluxo
- Bridge - bridge que será adicionada a regra
- $priority = 10$ - prioridade da regra criada
- in_$port = 1$ - porta do switch em que a regra será aplicada
- dl_$type = 0x0800$ - define o tipo do fluxo, neste caso é do tipo ethernet, código 0x0800 define o procoloto ethernet
- $actions =$ *normal* - ação de tratamento do fluxo criado

2.4 Diferenciação de serviços

Hoje, na Internet, temos diversos tipos de serviços que são oferecidos, como correio eletrônico, Word Wide Web, FTP (file transport protocol), serviços de telefonia, entre outros. Estes muitas vezes são utilizados simultaneamente, podendo sobrecarregar a rede. Há serviços que são de maior importância que outros, como por exemplo uma vídeo conferência que não pode travar, concorrendo simultaneamente com um acesso web que pode demorar para que o download da página seja feito. Neste contexto, diferenciar serviços, dando prioridade a uma classe é interessante.

Diferenciação de serviços está inserido em um contexto mais amplo de qualidade de serviço (QoS - quality of service) e tem como proposta privilegiar determinada classe de usuários e/ou aplicações.

Existem outros modelos que também implementam diferenciação de serviços estes são: IntServ e DiffServ.

O Differentiated Service (diferenciação de serviços - DiffServ ou DS), que especifica e controla o tráfego de rede de classe para que certos tipos de tráfego tenham precedência sobre outros. DiffServ pode, por exemplo, ser usado para fornecer baixa latência para o tráfego de rede crítico, como voz ou streaming de mídia, proporcionando para outros serviços uma melhor qualidade, outros considerados essenciais, tais como transferências de arquivos (FARIA; SANTOS, 2000).

O IntServ ou serviços integrados é um modelo de implementação de QoS que visa garantir a qualidade de certos serviços oferecidos. Este modelo utiliza o conceito de criação de fluxos, assim como a ideia que o OpenvSwitch implementa, reservando recursos nos roteadores para determinados tipos de fluxos. Cada roteador deve saber a quantidade de recursos que deve ser alocada para esta reserva, esta é feita através do RSVP – *Resource ReSerVation Protocol*, protocolo de sinalização executado por emissores, receptores e roteadores para reservar recursos na rede e para manter a informação do estado. Estes foram um dos motivos que este modelo não se tornou popular no mercado. (PEREIRA; MONTEIRO, 2004)

Qualidade de Serviço é definida como a capacidade de fornecer a um elemento da rede (aplicação, host ou roteador) algum nível de garantia de que seus requisitos de tráfego e serviço serão satisfeitos.

Este trabalho apresenta como o padrão OpenFlow lida com as questões de segurança e confiabilidade em uma Rede Definida por Software. Trata como os dados chegam, se eles chegam e no tempo que eles chegam, através de uma regra pré estabelecida.

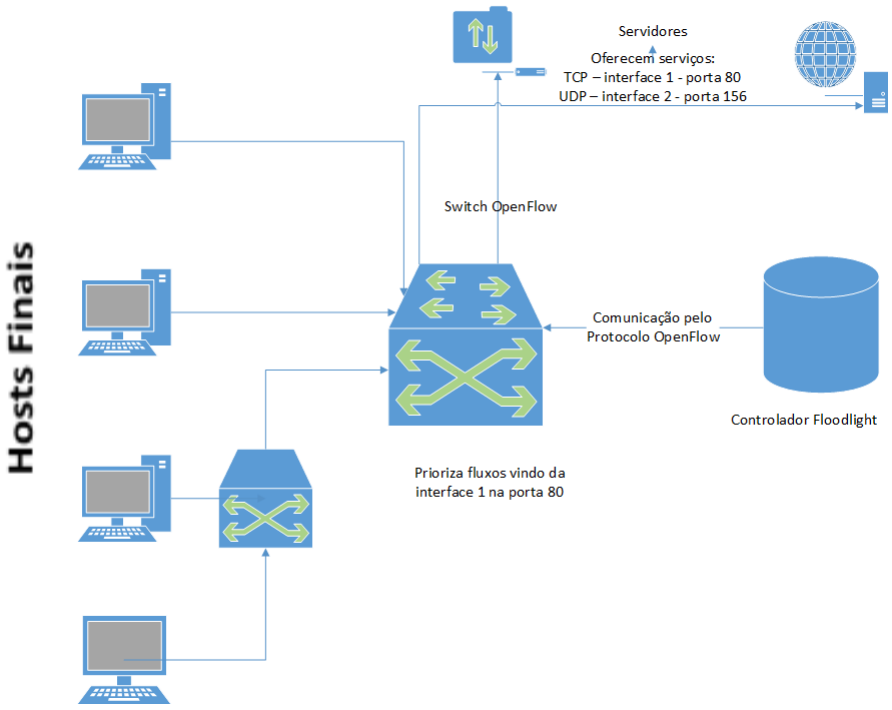

Figura 2.6: Exemplo de uma rede SDN com diferenciação de serviços.

No trabalho, avalia-se o uso do padrão OpenFlow para priorizar os fluxos de uma determinada aplicação. Esta aplicação vai ser vinculada a um determinado fluxo, este que será priorizado e assim, a aplicação será privilegiada. O método usado para conferir se o fluxo/aplicação foi privilegiado é a vazão obtida pelo fluxo.

A Figura 2.6 mostra um exemplo simples de uma rede com diferenciação de serviços que o protocolo OpenFlow oferece. Os dois servidores na parte de cima da figura mostram serviços oferecidos, a interface 1 oferece o serviço no protocolo TCP na porta 80 e a interface 2 oferece o serviço UDP na porta 156. O switch OpenFlow cria um fluxo priorizando a interface 1 na porta 80, assim será reservado uma parte maior da vazão para este fluxo, por dar uma prioridade maior a ele.

3 METODOLOGIA

A realização dos testes foi feita para a comprovação de como o padrão OpenFLow trata os dados em uma rede, como são criadas as regras para gerência da rede e como o conceito de Redes Definidas por Software muda a arquitetura atual de redes computadores.

3.1 Identificação do Tipo da Pesquisa

Este trabalho é uma pesquisa aplicada. Quanto à sua abordagem, é quantitativa, visto que irá utilizar a ideia de pesquisa aplicada, na qual será modificada a forma como pacotes são manipulados na rede, testar esta nova forma e apresentar resultados.

3.2 Introdução

Neste trabalho, os testes são realizados em ambiente simulado. Um ambiente simulado apresenta vantagens e desvantagens em relação ao ambiente real. Pelo ambiente ser totalmente virtualizado em uma máquina, os testes não possuem todas as variáveis que estão presente em um ambiente real, podendo não corresponder na sua totalidade à um cenário real. As vantagens são que por ser simulado, o custo do teste é mais baixo, visto que não será necessário trabalhar com equipamentos reais, as variáveis que interferem no ambiente são mais fáceis de ser controladas, como a não interferência de outras pessoas na rede. E há a facilidade na portabilidade do cenário. Por estes motivos foi escolhido um ambiente simulado.

Foi utilizado o simulador de redes *GNS3*, que permite a simulação de redes com vários dispositivos de rede. Este simulador faz integração com o software Virtualbox, que faz virtualização de máquinas reais, com isso a simulação torna o ambiente bem próximo da realidade.

O trabalho conta com um ambiente de teste em que tem-se uma máquina que faz o papel de um switch OpenFlow, um servidor de serviços, que fornecerá serviços TCP e UDP para dois hospedeiros.

Na avaliação de desempenho realizada, foram realizados oito testes para medição da vazão. A vazão foi medida através do software *iperf* - software para medir o desempenho da rede, mede a largura de banda máxima TCP, permite ajustes em parâmetros para medições UDP. Este é multi plataforma, bem simples e dá um relatório sobre as principais características da rede. Umas das principais características para a escolha deste software, é o amplo uso pela comunidade acadêmica e a eficácia comprovada. As medições são feitas cinco vezes por teste, calcula-se uma média das vazões e o intervalo de confiança com os dados obtidos por este experimento. Foram feitos cinco experimentos pois o *iperf* faz uma média de dez testes feitos por ele e por gastar um bom tempo para montar o teste.

3.3 Descrição dos testes

Com o objetivo de mostrar como o padrão OpenFlow pode ser usado para priorizar um fluxo e que este fluxo, ao ser priorizado leva vantagem sobre outro fluxo não priorizado, foram feitos os seguintes experimentos comparando entre os fluxos a vazão (*Bandwidth*) - largura de banda - medida da capacidade de transmissão de um determinado meio, conexão ou rede, determinando a velocidade que os dados passam através desta rede específica (KUROSE; ROSS, 2009).

Foram criados dois fluxos: UDP e TCP. Nos dois foram realizados os seguintes experimentos:

- Nenhum fluxo é priorizado e são medidas as vazões nos dois fluxos;
- O fluxo TCP é priorizado e o fluxo UDP não recebe nenhum tratamento, são feitas as medidas das vazões nos dois fluxos e comparadas ao primeiro experimento;

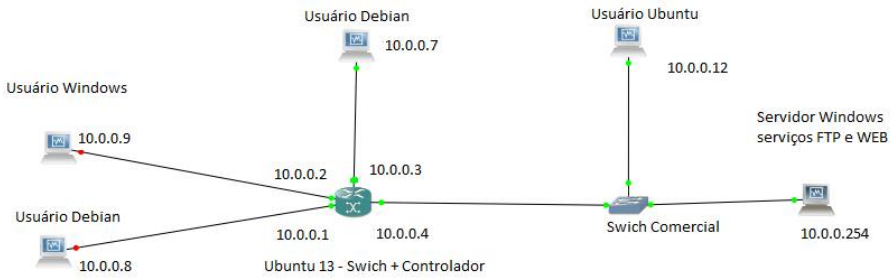

Figura 3.1: Topologia criada para realização dos experimentos

- O fluxo UDP é priorizado e o fluxo TCP não recebe nenhum tratamento, são feitas as medidas das vazões nos dois fluxos e comparadas ao primeiro experimento;

A Figura 3.1 apresenta o cenário criado para execução dos testes e apresenta a seguinte configuração: um máquina virtualizando o sistema operacional Ubuntu 13.10, esta sendo utilizada como um switch programável utilizando o conceito de redes definidas por software. Existe um switch comercial com funções básicas que liga uma máquina que virtualiza o sistema operacional Windows 7, que serve como servidor de dois serviços que serão consumidos: serviço web e FTP. Há também duas máquinas, que possuem apenas o software *iperf* instalado no sistema operacional, que serão os consumidores dos serviços que passarão pela máquina que virtualiza o openvswitch.

O software *iperf* foi instalado em todas as máquinas e ele possui as configurações para se comportar como servidor ou como cliente, para medir desempenho TCP e desempenho UDP. As máquinas clientes foram configuradas como cliente na recepção de pacotes na configuração do iperf. O resultado é apresentado pelo software em forma de relatório mostrando a vazão ponto a ponto no fluxo determinado. Em todos os testes a medição da vazão foi feita em Mbps. A descrição técnica de como foram feitos os testes encontra-se apêndice A.

3.3.1 Primeiro Cenário de Testes

No primeiro teste foram criados dois fluxos que oferecem serviços TCP. Foram medidas as vazões nos fluxos sem a inserção de regras OpenFlow e com a inserção de regras OpenFlow priorizando um dos fluxos TCP. A criação das regras TCP está presente no Apêndice A. Cada fluxo oferece um serviço em uma porta de saída, utilizada para definir a prioridade da regra do fluxo OpenFlow.

3.3.2 Segundo Cenário de Testes

No segundo teste será feito o teste de dois fluxos UDP e medido a vazão nos dois fluxos. Em um segundo teste mede-se a vazão nos dois fluxos, porém priorizando apenas um dos fluxos e o outro sem regra alguma.

3.3.3 Terceiro Cenário de Testes

No terceiro teste a comparação foi feita como no primeiro teste, medindo a vazão da rede apenas utilizando o openvswitch como um switch básico e depois aplicando regras OpenFlow no switch e medidndo sua vazão. Neste experimento foram usados serviços TCP e UDP, sem nenhuma priorização e priorizando depois o serviço TCP e um segundo experimento dentro deste cenário priorizando o serviço UDP. É importante frisar que cada serviço é oferecido em uma porta de saída e esta porta é a referência para a criação da regra do fluxo.

4 RESULTADOS E DISCUSSÃO

Os resultados foram divididos em etapas. São estas: comparação entre o cenário sem a utilização do openvswitch apenas como um switch simples e o modelo usando o padrão OpenFlow, ora priorizando pacotes usando serviço TCP, ora priorizando pacotes usando serviço UDP. Em todos os testes foram gerados dados da vazão na rede em Mbps, os testes foram repetidos cinco vezes, calculando assim a média e o intervalo de confiança dos dados. Os resultados são apresentados em gráficos comparativos, mostrando-se que o padrão OpenFlow é um protocolo flexível e prático para administradores de redes, que podem se livrar de equipamentos de caixa fechada e engessados.

4.1 Avaliação com fluxos TCP

No primeiro experimento são apresentados dois gráficos que mostram a vazão e o intervalo de confiança somado as médias em Mbps de dois fluxos, ambos usando o protocolo TCP. No cenário utilizando regras do padrão OpenFlow um fluxo é priorizado e nota-se uma vazão maior que o fluxo em que não está presente regras do padrão OpenFlow com a priorização do serviço. Visto pelas comparações no gráfico da Figura 4.1. São apresentados os fluxos com e sem a utilização de regras OpenFlow. O fluxo priorizado tem maior vazão ao serem adicionadas regras para priorização, demonstrando que fica fácil para um administrador de redes priorizar um fluxo em relação a outro. Neste gráfico, percebe-se que o teto do intervalo de confiança da vazão do cenário sem a utilização do protocolo OpenFlow é menor que piso do intervalo de confiança da vazão do cenário utilizando a difenciação de serviços com o protocolo OpenFlow.

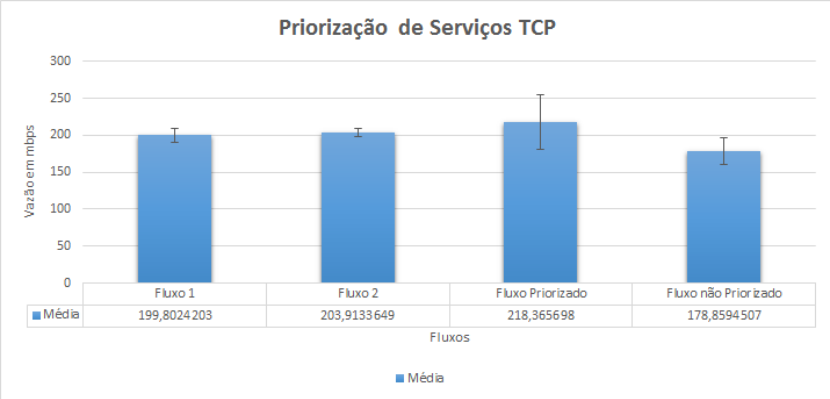

Figura 4.1: Gráfico que mostra um serviço TCP sendo priorizado em relação à outro sem aplicação de regra OpenFLow

4.2 Avaliação com fluxos UDP

No segundo experimento é feita a comparação apenas de fluxos UDP. São criados dois fluxos UDP, sendo medida a vazão nestes fluxos sem a utilização de regras OpenFlow.

São aplicadas regras priorizando um dos fluxos UDP e o outro não se aplica regra nenhuma. São feitas as medições da vazão nos fluxos. O gráfico da Figura 4.2 apresenta o resultado das médias e o intervalo de confiança somado as médias . Com a aplicação das regras OpenFlow o fluxo priorizado tem uma significativa melhora de 0,013988 mbps, em relação ao outro fluxo. Mostrando a facilidade e os ganhos ao se utilizar o padrão OpenFlow na gerência de redes.

4.3 Avaliação de fluxos TCP e UDP

No terceiro experimento a comparação é feita utilizando serviços TCP e UDP. É criado um fluxo para cada serviço. Os fluxos passam pelo openvswitch até chegar nos hospedeiros finais. É medida a vazão nos dois fluxo sem a imposição de regras OpenFlow. Depois são adicionadas regras aos fluxo priorizando, ora um serviço e ora outro. É feita comparação entre o cenário sem regras e o cenário com regras.

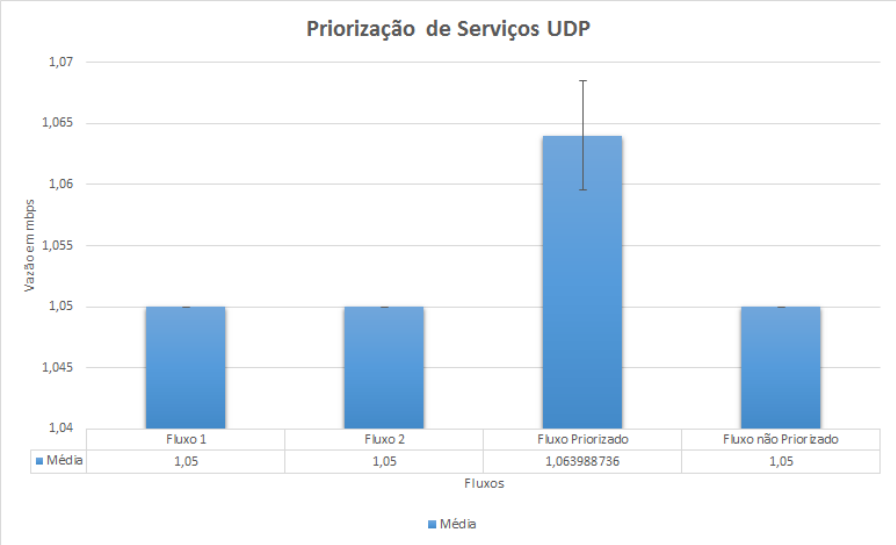

Figura 4.2: Priorização de um fluxo UDP em relação à outro fluxo UDP

No gráfico da Figura4.3 percebe-se que a vazão aumenta quando os serviços são priorizados. Esta flexibilidade é desejada por um administrador de redes, visto que privilegiar ora um serviço, ora outro é interessante em uma rede comercial. O gráfico da Figura4.3 apresenta o resultado com a priorização e o intervalo de confiança somado as médias do serviço TCP em relação ao fluxo sem priorização.

O gráfico da Figura 4.4 mostra a comparação entre o cenário com a utilização de regras OpenFlow e sem regras OpenFlow. Na comparação do fluxo UDP priorizado em relação ao não priorizado o aumento médio é de 0,01 Mbps por segundos, em um serviço UDP, como por exemplo uma vídeo-conferência, a qualidade pode ser melhorada reservando uma maior parte da vazão na rede para o fluxo em questão. Mas há casos em que este aumento na vazão do fluxo se torna um detalhe e o tempo para a criação de fluxo acaba se tornando grande em relação a melhora na vazão, visto no ganho que a criação da regra traz para o administrador, se tornando inviável este procedimento.

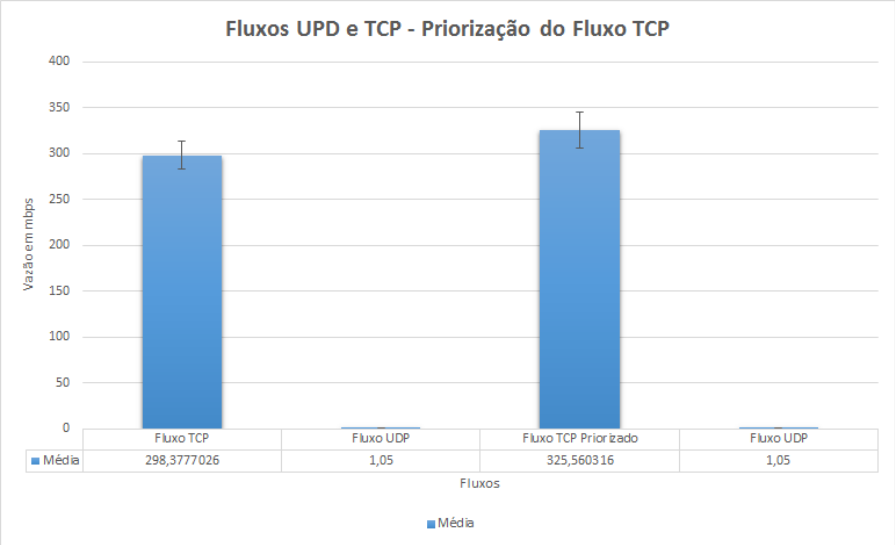

Figura 4.3: Priorização de um fluxo TCP em relação à um fluxo UDP

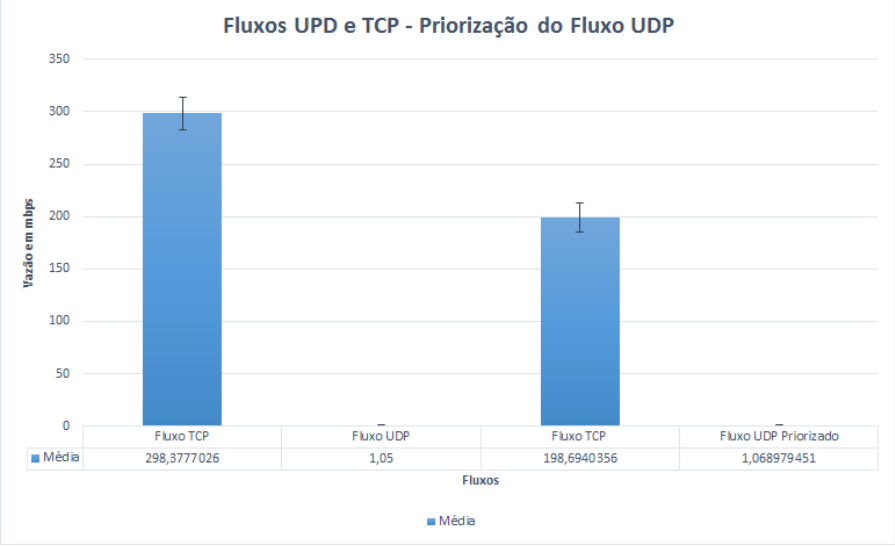

Figura 4.4: Priorização de um fluxo UDP em relação à um fluxo TCP

5 CONCLUSÃO E TRABALHOS FUTUROS

Com o grande crescimento de hospedeiros no atual cenário da rede mundial de computadores é importante uma administração eficiente dos pacotes que trafegam na rede. Os atuais padrões de tratamento de dados na rede são em sua grande maioria detidos por softwares proprietários de código fechado e que possuem uma abstração alta. O administrador de redes fica preso às opções de controle da rede presentes nos equipamentos proprietários e não pode dar características especiais a sua rede. Redes Definidas por Software vêm para acabar com este engessamento da gerência de redes.

Os resultados apresentados mostram que para diferenciação de serviços, o uso do padrão OpenFlow é viável para a maioria dos casos. Ao se diferenciar dois serviços TCP, nota-se uma diferença média de 20 mbps na vazão quando se impõe regras no switch OpenFlow.

Com os serviços UDP, a diferença entre as vazões de um cenário que se tem serviços priorizados e outro com serviços não priorizados é pequena, de 0,016988 mbps. A questão fica para o administrador da rede, caso esta diferença seja significativa, valerá a pena impor as regras.

E utilizando os dois serviços em conjunto, notou-se também que o serviço TCP tem uma maior vazão ao ser priorizado, diferença de 27 mbps. E o serviço UDP, quando priorizado, a diferença na vazão é pequena, 0,01687 mbps.

Este trabalho mostra que o conceito de Redes Definidas por Sotware, focado no padrão OpenFlow, possibilita uma grande flexibilidade no gerenciamento de redes facilitando o trabalho dos administradores de redes na alteração, criação e remoção das regras dos fluxos na rede. A gerência da rede fica centralizada, o switch OpenFlow ligado ao controlador mantêm conhecimento de toda topologia, levando a tomar decisões mais eficientes por considerar toda a rede.

Os elementos usados no projeto são todos de código aberto e de acesso irrestrito e por se destacar cada vez mais no cenário atual da comunidade acadêmica e comercial a tendência é melhorar.

O principal trabalho futuro a ser desenvolvido é desenvolver um projeto em uma rede real, podendo assim abranger todas as variáveis, implantando este sistema em uma rede comercial e comparar o resultado com o modelo tradicional.

Outro trabalho futuro a ser desenvolvido é usar Redes Definidas com Software em conjunto com o protocolo IPv6, visto que estes dois padrões tendem a ser o futuro do mundo das redes de computadores.

REFERÊNCIAS BIBLIOGRÁFICAS

FARIA, J.; SANTOS, A. Avaliação de qos numa arquitectura diffserv. *FCCN*, v. 1, n. 1, p. 1–15, 2000.

GREENBERG, A.; HJALMTYSSON, G.; MALTZ, D. A.; MYERS, A.; REXFORD, J.; XIE, G.; YAN, H.; ZHAN, J.; ZHANG, H. A clean slate 4d approach to network control and management. *SIGCOMM Comput. Commun. Rev.*, ACM, New York, NY, USA, v. 35, n. 5, p. 41–54, out. 2005. ISSN 0146-4833. Disponível em: <http://doi.acm.org/10.1145/1096536.1096541>.

GUEDES, D.; VIEIRA, L. F. M.; VIEIRA, M. M.; RODRIGUES, H.; NUNES, R. V. Redes definidas por software: uma abordagem sistêmica para o desenvolvimento de pesquisas em redes de computadores. *XXX Simpósio Brasileiro de Redes de Computadores e Sistemas Distribuídos - SBRC*, v. 1, n. 1, p. 160–210, 2012.

KUROSE, J. F.; ROSS, K. W. *Computer Networking: A Top-Down Approach*. 5th. ed. USA: Addison-Wesley Publishing Company, 2009. ISBN 0136079679, 9780136079675.

PARRAGA, K.-C. K. W. J. *Floodlight Documentation*. [S.l.], 2013. Disponível em: <http://docs.projectfloodlight.org/display/floodlightcontroller/Floodlight+Documentation>.

PEREIRA, A.; MONTEIRO, E. Avaliação de mecanismos de gestão de recursos no mapeamento entre os modelos intserv e diffserv. *Proc. of CRC2004, Leiria*, p. 179–191, 2004.

ROTHENBERG MARCELO RIBEIRO NASCIMENTO, M. R. S. M. F. M. a. C. E. Openflow e redes definidas por software: um novo paradigma de controle e inovação em redes. *Cad. CPqD Tecnologia, Campinas*, v. 7, n. 1, p. 65–76, julho 2010 - 2011.

A APÊNDICES

A.1 Configuração Iperf

A.1.1 Configuração Cliente

Para configurar uma máquina para ser cliente no iperf é usado o seguinte comando no shell do sistema operacional :

iperf -c "Ip do servidor"

Comandos usados no presente trabalho :

- iperf -c 10.0.0.254 - comando para serviços TCP
- iperf -c -u 10.0.0.254 - comando para serviços UDP

A.1.2 Configuração Servidor

Para configurar uma máquina como servidor no software iperf o seguinte comando é usado:

iperf -s

Comandos usados no preente trabalho:

- iperf -s - comando para serviços TCP
- iperf -s -u -p 21 - comando para serviços UDP na porta 21
- iperf -s -u -p 22 - comando para serviços UDP na porta 22

A.2 Configuração das interfaces de rede no OpenvSwitch

O Openvswitch usado no trabalho proporciona uma interface de programação bem simples, nesta são inseridos os comandos para a criação de interfaces, adição de portas nas interfaces criadas e manutenção das regras.

A.2.1 Configuração de Bridge OpenFlow

Comandos para adição de bridge OVS no switch OpenFlow :

- ovs-vsctl add-br teste

A.2.1.1 Controle da Bridge OpenFlow

Comandos para adiçao de portas na bridge criada:

- ovs-vsctl add-port teste eth1
- ovs-vsctl add-port teste eth2
- ovs-vsctl add-port teste eth3
- ovs-vsctl add-port teste eth4

A.2.1.2 Ligação do controlador ao switch OpenFLow

Comando para ligação do controlador ao switch OpenFlow:

- ovs-vsctl set-controller teste tcp 0.0.0.0/0

A.2.1.3 Criação das regras OpenFlow

Comandos para criação das regras OpenFlow :

- ovs-ofctl add-flow teste dl_$type = 0x0800$,$priority = 99$,in_$port = 4$,tp_$src =$ 80,$actions = normal$
- ovs-ofctl add-flow teste $priority = 9999$,in_$port = 4$,tp_$src = 21$,dl_$type =$ $0x0800$,nw_$proto = 17$,$actions = normal$
- ovs-ofctl add-flow teste $priority = 9999$,in_$port = 4$,tp_$src = 21$,dl_$type =$ $0x0800$,nw_$proto = 17$,$actions = normal$

- ovs-ofctl add-flow teste dl_$type = 0x0800$,nw_$proto = 6$,$priority = 9999$,in_$port =$ 4,tp_$src = 5001$,$actions = normal$