

UNIVERSIDADE TECNOLÓGICA FEDERAL DO PARANÁ
DEPARTAMENTO ACADÊMICO DE ELETRÔNICA
CURSO SUPERIOR DE TECNOLOGIA EM SISTEMAS DE TELECOMUNICAÇÕES

JOSÉ BOGO JUNIOR

**USO DA FERRAMENTA MININET PARA ESTUDO DE
REDES DEFINIDAS POR SOFTWARE**

TRABALHO DE CONCLUSÃO DE CURSO

CURITIBA
2017

JOSÉ BOGO JUNIOR

USO DA FERRAMENTA MININET PARA ESTUDO DE REDES DEFINIDAS POR SOFTWARE

Trabalho de Conclusão de Curso de Graduação, apresentado ao Curso Superior de Tecnologia em Sistemas de Telecomunicações, do Departamento Acadêmico de Eletrônica, da Universidade Tecnológica Federal do Paraná – UTFPR, como requisito parcial para obtenção do título de Tecnólogo.

Orientador: Prof. Kleber Kendy Horikawa Nabas

TERMO DE APROVAÇÃO

JOSÉ BOGO JUNIOR

USO DA FERRAMENTA MININET PARA ESTUDO DE REDES DEFINIDAS POR SOFTWARE

Este trabalho de conclusão de curso foi apresentado no dia 14 de novembro de 2017, como requisito parcial para obtenção do título de Tecnólogo em Sistemas de Telecomunicações, outorgado pela Universidade Tecnológica Federal do Paraná. O aluno foi arguido pela Banca Examinadora composta pelos professores abaixo assinados. Após deliberação, a Banca Examinadora considerou o trabalho aprovado.

Prof. M.Sc. Danilo Leal Belmonte
Coordenador de Curso
Departamento Acadêmico de Eletrônica

Prof. M.Sc Sérgio Moribe
Responsável pela Atividade de Trabalho de Conclusão de Curso
Departamento Acadêmico de Eletrônica

BANCA EXAMINADORA

Prof. Dr. Edenilson José da Silva
UTFPR

Prof. M.Sc. Omero Francisco Bertol
UTFPR

Prof. Dr. Kleber Kendy Horikawa Nabas
Orientador – UTFPR

“A Folha de Aprovação assinada encontra-se na Coordenação do Curso”

RESUMO

JUNIOR, José Bogo. **USO DA FERRAMENTA MININET PARA ESTUDO DE REDES DEFINIDAS POR SOFTWARE**. 2017. 34 f. Trabalho de Conclusão de Curso (Curso Superior de Tecnologia em Sistemas de Telecomunicações), Departamento Acadêmico de Eletrônica, Universidade Tecnológica Federal do Paraná. Curitiba, 2016.

Este trabalho tem por objetivo caracterizar uma rede definida por software e busca, por meio da pesquisa bibliográfica, apresentar o emulador Mininet como ferramenta de estudos de tal tecnologia, implementando um ambiente de testes e demonstrando formas de aplicação prática.

Palavras chave: Redes Definidas por Software. OpenFlow. Mininet.

ABSTRACT

JUNIOR, José Bogo. **USE OF THE MININET TOOL FOR STUDY OF SOFTWARE DEFINED NETWORKS**. 2017. 34 f. Trabalho de Conclusão de Curso (Curso Superior de Tecnologia em Sistemas de Telecomunicações), Departamento Acadêmico de Eletrônica, Universidade Tecnológica Federal do Paraná. Curitiba, 2017.

This work aims to characterize a network defined by software and search, through the bibliographic research, to present the Mininet emulator as a tool for the study of such technology, implementing a testing environment and demonstrating practical applications.

Keywords: Software Defined Networks. OpenFlow. Mininet.

LISTA DE FIGURAS

Figura 1: Topologias de Redes de Computadores.....	13
Figura 2: Modelos OSI/ISO e TCP/IP.....	15
Figura 3: Exemplo de rede Lan (Local Area Network).....	18
Figura 4: Exemplo de WAN ou MAN.....	19
Figura 5: Exemplo de VLAN's.....	20
Figura 6: Estrutura básica do conceito SDN.....	22
Figura 7: Exemplo de tela do Mininet.....	25
Figura 8: Tela do MiniEdit.....	25
Figura 9: Conexão ao Mininet por SSH.....	26
Figura 10: Figura 10: Tela do WireShark capturando pacotes.....	29

LISTA DE SIGLAS

ARP	Address Resolution Protocol
DNS	Domain Name System
FTP	File Transfer Protocol
HTTP	Hypertext Transfer Protocol
ICMP	Internet Control Message Protocol
IP	Internet Protocol
ISO	International Organization for Standardization
LAN	Local Area Network
LTS	Long Time Support
OSI	Open System Interconnection
SDN	Software Defined Network
SMTP	Simple Mail Transfer Protocol
SSH	Secure Shell
TCP	Transmission Control Protocol
VLAN	Virtual Local Area Network
VM	Virtual Machine
WAN	Wide Area Network

SUMÁRIO

1 INTRODUÇÃO.....	8
1.1 OBJETIVOS.....	8
1.1.1 OBJETIVO GERAL.....	8
1.1.2 OBJETIVOS ESPECÍFICOS.....	8
1.3 JUSTIFICATIVA.....	9
1.4 PROCEDIMENTOS METODOLÓGICOS.....	9
2 REFERENCIAIS TEÓRICOS.....	10
3 REVISÃO TEÓRICA.....	12
3.1 AS PRIMEIRAS REDES.....	12
3.2 O PROTOCOLO TCP/IP.....	15
3.3 AS REDES CLÁSSICAS ATUAIS.....	17
4. Redes Definidas por Software.....	21
4.1 Definição de Redes Definidas por Software.....	21
4.2 O Protocolo OpenFlow.....	22
4.3. A Ferramenta Mininet.....	23
4.4 Elementos SDN no Mininet.....	26
4.5 Linhas de comando.....	27
4.6 API e exemplos prontos.....	28
4.7 Wireshark.....	29
5. Conclusão e Implementações Futuras.....	31

1 INTRODUÇÃO

O uso de redes definidas por software (*Software Defined Networks – SDN*) é cada vez mais difundido, principalmente dentro de grandes corporações como Google, Facebook, grandes empresas de telecomunicações, entre outras. SDN é uma evolução natural das grandes redes corporativas, trazendo benefícios como maior facilidade e precisão no controle das camadas de comunicação interna de suas redes.

Devido às características peculiares envolvidas, tais como ambientes de programação e protocolos específicos, é de vital importância que se conheçam quais são as opções disponíveis atualmente e iniciar uma base de conhecimento que servirá de plataforma para estudos futuros.

1.1 OBJETIVOS

1.1.1 OBJETIVO GERAL

Aumentar a base de conhecimento em redes computacionais, adicionando o conhecimento sobre SDN, comparando essa tecnologia com as usuais.

1.1.2 OBJETIVOS ESPECÍFICOS

Levantar conceitos e usos de redes definidas por software.

Identificar suas particularidades e funcionalidades.

Implementar um ambiente de estudos virtualizado de redes definidas por software.

Demonstrar uma rede SDN funcional para demonstração.

Compreender o funcionamento básico de redes definidas por software, suas aplicações e modos de operação e controle.

1.3 JUSTIFICATIVA

O sucesso das redes de computadores como base para a comunicação da sociedade moderna é evidente, porém a tecnologia acabou se tornando estagnada em vários aspectos, devido à dificuldade para a implantação de novos protocolos e tecnologias, pois quaisquer novas propostas, de forma geral, requerem mudanças profundas em todos os milhares de equipamentos de rede instalados.

Os administradores de rede acabam por descartar essas mudanças, pois as redes de computadores são parte da infraestrutura crítica, e portanto mudanças radicais tornam-se obstáculos. Todas essas barreiras dificultam a implantação de novas ideias que acabam não sendo testadas em ambientes reais.

O paradigma de Redes Definidas por Software e o protocolo *OpenFlow* foram desenvolvidos com o propósito de viabilizar o teste de novas propostas em redes reais sem afetar as redes de produção. Dessa forma, além de atender os problemas da inflexibilidade das redes de computadores, SDN nos permite atender as necessidades das novas aplicações de rede que deverão surgir em um futuro próximo, resolvendo problemas associados a requisitos de escalabilidade, gerenciamento, mobilidade e segurança por meio de uma arquitetura de rede programável (COSTA, 2013).

1.4 PROCEDIMENTOS METODOLÓGICOS

A metodologia adotada para este projeto consiste em fazer uma revisão bibliográfica das diferentes tecnologias de rede e apresentar a estrutura da ferramenta Mininet. Em seguida serão demonstrados alguns exemplos de códigos e como estes afetam o ambiente de simulação e emulação de redes SDN.

2 REFERENCIAIS TEÓRICOS

A gestão de redes cada vez maiores, mais complexas e de larga escala podem exigir mudanças de regras de segurança e reconfigurações de tempos em tempos. SDN torna essa realidade possível por dissociar o plano de controle de plano de dados (FEI, QI e KE, 2014).

Segundo Rechia (RECHIA, 2014), “o termo SDN representa um conceito de arquitetura de rede na qual o controle dos fluxos de tráfego é feito por uma entidade independente dos equipamentos que encaminham os fluxos de tráfego. Ou seja, a camada de controle é separada da camada de tráfego de dados. O plano de controle, por sua vez, atua de acordo com aquilo que é definido pelas aplicações que utilizam a rede”.

O paradigma de Redes Definidas por Software e o protocolo *OpenFlow* foram desenvolvidos com esse propósito, viabilizar o teste de novas propostas em redes reais sem afetar as redes de produção. Dessa forma, além de atender os problemas da inflexibilidade das redes de computadores, o paradigma SDN nos permite atender as necessidades das novas aplicações de rede que deverão surgir em um futuro próximo, resolvendo problemas associados a requisitos de escalabilidade, gerenciamento, mobilidade e segurança por meio de uma arquitetura de rede programável (COSTA, 2013).

BENNESBY defende que “a arquitetura do roteamento interdomínio da Internet tem sofrido poucas mudanças desde sua criação. Ela apresenta problemas complexos de serem resolvidos, devido à dificuldade para a implantação de novas soluções, à dificuldade de entender seu comportamento e dinâmica, e à complexidade de identificar e corrigir falhas. A implantação de novas funcionalidades na arquitetura da Internet é uma tarefa difícil, devido à necessidade de manipular diretamente todos os roteadores. Além disso, é necessário aceitação global de novos protocolos e modificações nos existentes. Observa-se que a abordagem Redes Definidas por Software (SDN – *Software-Defined Networking*) poderia ser usada para prover inovação no roteamento interdomínio, mas que faltavam mecanismos para suportar esse tipo de roteamento” (BENNESBY, 2013).

A Internet levou à criação de uma sociedade digital, em que (quase) tudo está conectado e é acessível de qualquer lugar. No entanto, apesar da sua adoção generalizada, as redes IP tradicionais são complexas e muito difíceis de gerir. É ao mesmo tempo difícil de configurar a rede de acordo com políticas predefinidas, e

reconfigurá-lo para responder a falhas, carga e mudanças. Para tornar as coisas ainda mais difíceis, as redes atuais também estão integradas verticalmente: o controle e de dados planos são agrupados. Rede definida por software (SDN) é um paradigma emergente que promete mudar este estado de coisas, por quebrar a integração vertical, separando a lógica de controle da rede dos roteadores e switches subjacentes, promover a centralização (lógica) de controle de rede, e introduzindo a capacidade para programar a rede (KREUTZ et al. 2015).

Com o uso do paradigma SDN e do protocolo *OpenFlow*, a inovação e a evolução da rede são facilitadas. Dessa forma, muitos serviços típicos de rede podem ser repensados, de forma a torná-los mais flexíveis (CHINELATE, 2016).

3 REVISÃO TEÓRICA

As redes de computadores se tornaram parte da infraestrutura crítica das empresas, casas e escolas, sendo um sucesso do cotidiano de bilhões de pessoas. Desde sua origem, as redes de computadores tem crescido bastante e seu uso ficou cada vez mais diversificado. A massificação das tecnologias de rede foi obtida, em grande parte, pela sua boa adoção sistemática no meio comercial, ao mesmo tempo em que programas multiusuários começaram a ser desenvolvidos (COSTA, 2013).

Nesse contexto as redes de computadores cresceram agregando milhões de equipamentos. O sucesso das redes de computadores deve, em grande medida, a simplicidade de seu núcleo. Na arquitetura atual, a inteligência da rede está localizada nos sistemas de borda, enquanto o núcleo é simples e transparente. Embora essa simplicidade tenha tido sucesso, viabilizando a Internet, também é a razão para seu engessamento. Todas essas limitações apresentam problemas estruturais que são difíceis de serem resolvidos, tais como escalabilidade, mobilidades e gerenciamento de serviço (COSTA, 2013).

3.1 AS PRIMEIRAS REDES

As Redes de Computadores surgiram a partir de pesquisas militares no período da Guerra Fria na década de 1960 quando dois blocos ideológicos e politicamente antagônicos exerciam enorme poder sobre o mundo. O líder de um desses blocos, os Estados Unidos, temia um ataque do outro bloco, liderado pela União Soviética, a uma de suas bases militares. Um ataque poderia revelar todas as informações sigilosas dos Estados Unidos. Temendo tal ataque, o Departamento de Defesa dos Estados Unidos desenvolveu uma rede de comunicação que tinha o objetivo de descentralizar as informações até então concentrada em pontos específicos (COSTA, 2013).

Como exemplo, pode-se citar as redes telefônicas, que eram organizadas seguindo uma topologia em estrela que continha pontos centrais (Figura 1), dessa maneira, se algum desses pontos sofresse um ataque soviético a rede poderia se desestruturar, causando um grande prejuízo e impacto na guerra (COSTA, 2013).

Para descentralizar as informações foi criado um novo modelo de troca e compartilhamento das informações entre as bases militares, constituindo a primeira rede de computadores, a ARPANET (*Advanced Research Projects Agency Network*) (COSTA, 2013).

A ARPANET foi projetada e constituída para ser uma rede tolerante a falhas e altamente distribuída. Sua estrutura era organizada seguindo uma topologia em malha (Figura 1) que utilizava comutação por pacotes entre seus nós (COSTA, 2013).

Com o passar do tempo, e de acordo com as necessidades momentâneas, novas topologias foram surgindo, tais como Topologia em Barramento, em Anel, Estrela estendida e Hierárquica como mostrado na Figura 1.

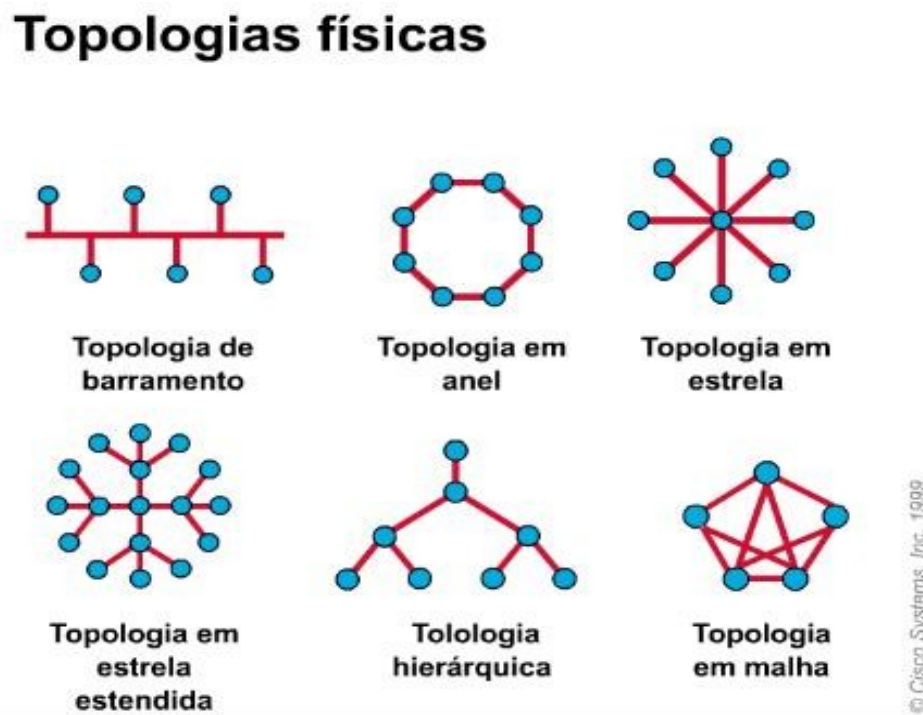


Figura 1: Topologias de Redes de Computadores
(Fonte: http://www.al.es.gov.br/appdata/imagens_news/rede.htm – acesso em 13/11/2017)

Depois que a tensão entre URSS e EUA diminuiu, a ARPANET cresceu, pois os EUA permitiram que pesquisadores desenvolvessem estudos para a ARPANET trazendo mais usuários a rede. Com isso a dificuldade em administrar o sistema cresceu e uma nova ARPANET surgiu com usuários que não tinham relações militares (COSTA, 2013).

Mas com o crescimento da ARPANET, ficou evidente que os protocolos utilizados em sua implementação já não eram suficientes para assegurar o crescimento da rede com confiabilidade e segurança. Dentro desse cenário, pesquisadores procuraram focar esforços na resolução desses problemas, e então foi criado o protocolo TCP/IP.

O TCP/IP foi criado com o objetivo de manipular a comunicação entre redes permitindo que o tráfego de informações fosse encaminhado de um lugar para outro com

confiabilidade, com pacotes de diferentes tamanhos, tratando e recuperando erros de transmissão e falhas de sequenciamento de pacotes, além de realizar controle de fluxo, congestionamento e verificação de erros fim-a-fim (COSTA, 2013).

Com o aumento da rede e sua conseqüente descentralização, surgiu o conceito de *backbone*, que basicamente consiste em computadores conectados entre si com a capacidade de dar vazão a grandes fluxos de dados. Ou seja, os *backbones* são ligações centrais de um sistema de rede mais amplo, servindo de intercomunicação entre todas as redes.

De acordo com COSTA (COSTA, 2013), a união de todas as redes define uma interligação global chamada *Internet*. É importante destacar que a *Internet* não possui um único dono. Isso acontece pelo fato de que as conexões entre essas redes é feita considerando as políticas locais de cada instituição. Nesse sentido não há uma administração centralizada, embora existam organizações que se dedicam a definir padrões, normas e regras para sua utilização e disponibilização, o que garante o seu funcionamento. Podemos exemplificar as seguintes organizações:

- O *World Wide Web Consortium (W3C)* que concentra-se na padronização de tecnologias da Web.
- A *OASIS (Organization for the Advancement of Structured Information Standards)* é um consórcio internacional especializado no desenvolvimento de padrões para segurança da Web.
- A *Internet Engineering Task Force (IETF)* que concentra-se no desenvolvimento da arquitetura das redes de computadores e a operação uniforme da mesma por meio de protocolos. Cada padrão da IETF é publicado como uma RFC (Request for Comments) e está disponível gratuitamente.
- O *IEEE (Institute of Electrical and Electronics Engineers)* é uma organização que cria padrões nas áreas de tecnologia da informação, telecomunicações, medicina e saúde, transporte e outros.
- O comitê *ISO (International Organization for Standards)* é o maior desenvolvedor de padrões do mundo e mantém uma rede de institutos nacionais de padronização em mais de 140 países.

3.2 O PROTOCOLO TCP/IP

Atualmente as redes de computadores utilizam, normalmente, uma arquitetura baseada em um modelo em camadas conhecido como modelo TCP/IP.

Baseado no modelo mais geral proposto pelo OSI/ISO, o modelo TCP/IP é utilizado atualmente na Internet devido a sua abrangência e abordagem geral de características descritas em cada camada (COSTA, 2013).

De acordo com COSTA (COSTA, 2013), o modelo TCP/IP, ou a pilha de protocolos TCP/IP, é um padrão industrial de protocolos destinados as redes de computadores sendo as principais peças da arquitetura das redes de computadores atuais. A arquitetura de redes tem como objetivo realizar a interligação de computadores e redes, não importando o tipo, entre si. A arquitetura TCP/IP está organizada em quatro camadas: a) a camada de aplicação; b) a camada de transporte; c) a camada de rede; d) a camada de enlace/físico. Na Figura 2 tem-se um comparativo entre o modelo OSI/ISO e o modelo TCP/IP.

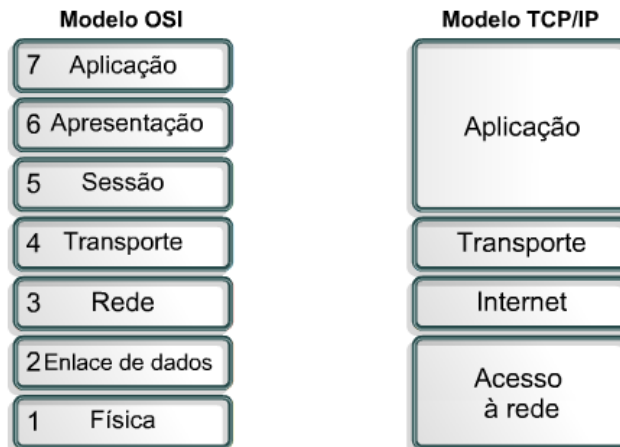


Figura 2: Modelos OSI/ISO e TCP/IP

(Fonte: <https://jbgsm.wordpress.com/2010/05/31/camadas-tcpip/> - Acesso em 13/11/2017)

Ainda segundo COSTA (COSTA, 2013), o modelo TCP/IP, ou a pilha de protocolos TCP/IP, é um padrão industrial de protocolos destinados as redes de computadores sendo as principais peças da arquitetura das redes de computadores atuais. A arquitetura de redes tem como objetivo realizar a interligação de computadores e redes, não importando o tipo, entre si. A arquitetura TCP/IP está organizada em quatro camadas: a

camada de aplicação; a camada de transporte; a camada de internet e a camada de acesso à rede.

a) **Camada de Aplicação** – A camada de aplicação localiza-se acima da camada de transporte, tem a função de prover serviços para as aplicações criando a comunicação, por meio de uma rede, com outras aplicações. A camada de aplicação é responsável por identificar e estabelecer a disponibilidade da aplicação nas máquinas disponibilizando os recursos para que tal comunicação aconteça. Ela contém todos os protocolos de níveis mais altos tais como o SMTP, o FTP, o HTTP, DNS e outros protocolos que foram incluídos no decorrer dos anos com novas aplicações e modificações na rede.

b) **Camada de Transporte** – A camada de transporte situada entre as camadas de aplicação e de rede, tem como função promover um canal de comunicação lógico fim-a-fim entre as camadas de aplicação rodando em diferentes computadores não se preocupando com os detalhes das camadas inferiores. Os protocolos de transporte são implementados, em geral, nos sistemas de borda da rede, já que por sua vez tem implementações mais complexas e oferecem um canal lógico fim-a-fim para a camada de aplicação.

c) **Camada de Internet** – A camada de internet é responsável pelo roteamento dos pacotes entre fonte e destino. A camada de internet permite que os *hosts* envie pacotes em qualquer rede garantindo que eles trafegarão independentemente até o destino, muitas vezes utilizando rotas diferentes chegando até mesmo em ordem diferente daquela em que foram enviados. A camada de internet é formada basicamente pelo protocolo IP que tem como característica o “melhor esforço” (*best effort*), isto é, faz o melhor esforço para envio de um pacote entre os *hosts* não dando nenhuma garantia de entrega, obrigando então as camadas superiores tratarem suas limitações.

d) **Camadas de Acesso à Rede** – A camada de Acesso à Rede é responsável pela transmissão e recepção de quadros da camada de rede de um dispositivo para a camada de rede de outro dispositivo fisicamente adjacente, estabelecendo um controle de fluxo por meio de um protocolo de comunicação entre sistemas.

3.3 AS REDES CLÁSSICAS ATUAIS

Para um entendimento de redes SDN, é importante compreendermos o funcionamento, de forma geral, de uma rede de computadores atual.

Ao passar dos anos as redes de computadores foram crescendo tornando-se um aglomerado de redes em escala mundial. A interconexão de milhares de computadores por meio de protocolos de comunicação padronizados que permite o acesso a informações e a todo tipo de serviço de dados define a Internet, que é proveniente da expressão *internetwork* (comunicação entre redes) (COSTA, 2013).

Uma maneira simples de visualizar a Internet é considerar uma nuvem com computadores conectados a ela. Essa nuvem é considerada o núcleo das redes de computadores. É uma nuvem dinâmica e cresce a medida que crescem e nascem novas redes (COSTA, 2013).

Segundo COSTA (COSTA, 2013), a arquitetura atual das redes de computadores atende praticamente os mesmos requisitos especificados na arquitetura da ARPANET.

Esses requisitos consistem em:

- conectividade – permite que qualquer estação de qualquer rede possa enviar dados para qualquer outra estação de qualquer outra rede;
- generalidade – dar suporte a diferentes tipos de serviços e aplicações;
- heterogeneidade – permitir a interconexão de diferentes dispositivos e tecnologias de rede;
- robustez – efetuar a comunicação desde que exista algum caminho entre origem e destino;
- acessibilidade – facilitar a conexão de novas estações ou redes.

As redes de computadores atuais podem ser classificadas de diversas formas, e comumente são divididas de forma hierárquica entre redes locais, ou LAN – *Local Area Networks*, redes metropolitanas, ou MAN – *Metropolitan Area Networks* e redes de longa distância, chamadas de WAN – *Wide Area Networks*, cada uma com tamanho maior. A interconexão de uma ou mais redes é chamada de *Internetwork*, cujo principal exemplo é a *Internet* (RECHIA, 2014).

Redes LAN são redes que se limitam a uma área bem definida como um escritório ou um prédio de escritórios, ou até mesmo um condomínio residencial ou empresarial.

São baseadas em computadores (geralmente PC's, ou computadores pessoais, sejam eles Desktops ou Notebooks), servidores, *switches* e roteadores.

É de comum conhecimento que a maior parte das redes locais são baseadas em *switches* Ethernet, os quais são ligados a computadores e servidores através de cabos e portas com padrão Ethernet, através de cabos óticos ou de cobre. Os *switches* são responsáveis por realizar então o encaminhamento de quadros (*frames*) de dados entre os diversos dispositivos conectados a ele, utilizando o campo de endereço de destino do quadro para encaminhá-lo corretamente (RECHIA, 2014).

A Figura 3 mostra um exemplo de rede LAN:

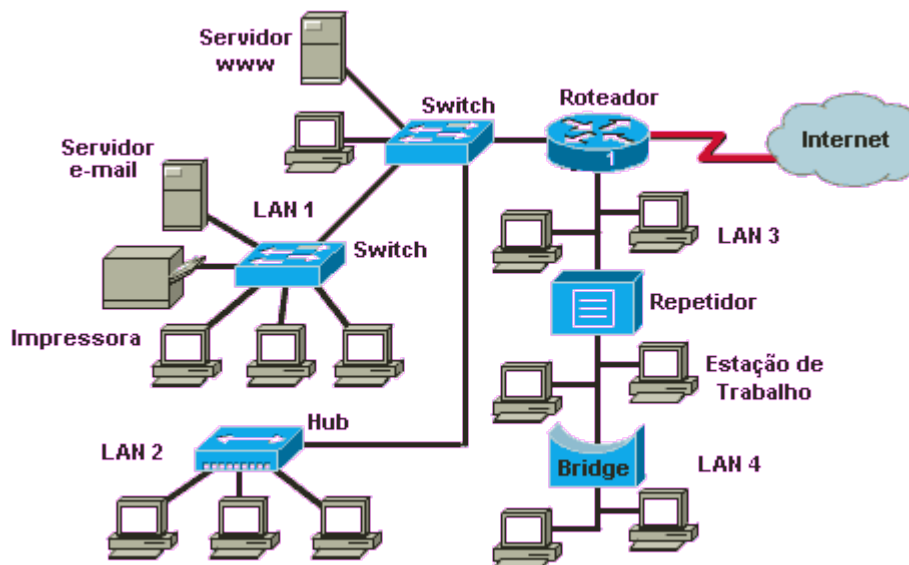


Figura 3: Exemplo de rede Lan (Local Area Network)

(Fonte: <http://www.proietoderedes.com.br> – Acesso em 13/11/2017)

Redes WAN são as redes que interconectam as LAN's entre si e essas com a Internet. São provedores de acesso que possuem equipamentos (roteadores e *switches*) de maior porte e grande volume de dados trafegando em suas redes.

Quando uma rede WAN se limita a uma cidade ou região metropolitana, geralmente é denominada MAN, como mostra a Figura 4:

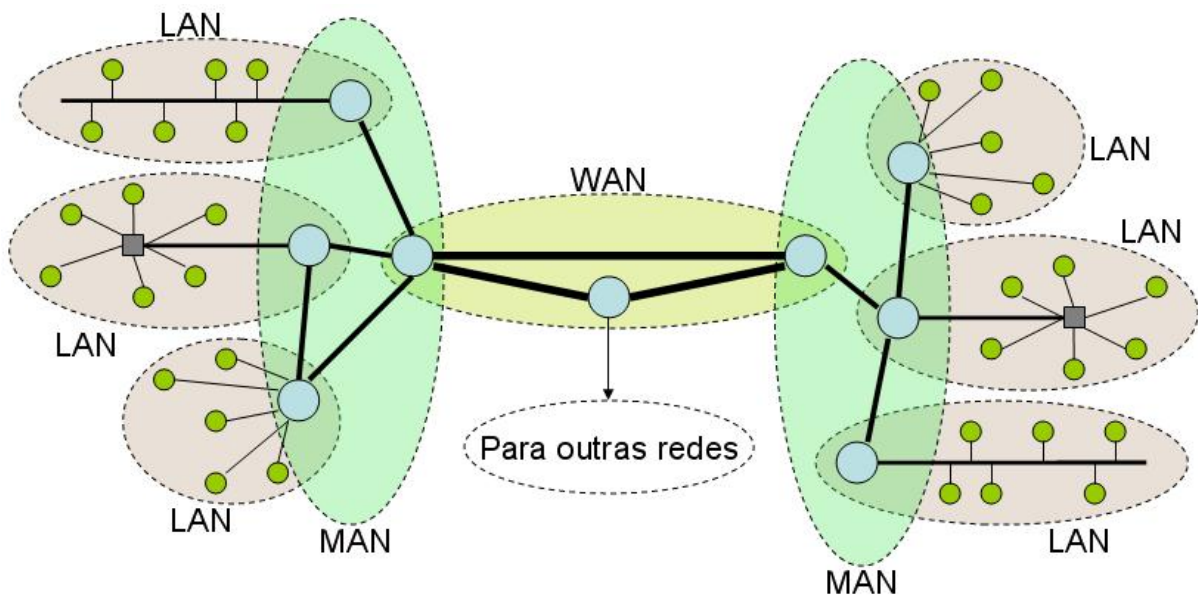


Figura 4: Exemplo de WAN ou MAN

(Fonte: <http://www.ata.ufri.br> – Acesso em 13/11/2017)

Outro modelo de redes muito comum utilizado atualmente para facilitar o gerenciamento e proporcionar mais desempenho e segurança às redes é o conceito de VLAN (Figura 5), também conhecido como Redes Virtuais.

Redes corporativas de grandes empresas podem ter centenas ou até milhares de usuários, e isso torna praticamente obrigatória a segregação da rede toda em diversas LAN's, o que é feito através de separações lógicas de LAN's, realizadas com VLAN's (virtual LAN's). Segmentar a rede pode trazer diversos benefícios, como facilitar o controle de acesso a recursos e também evitar degradação de performance (RECHIA, 2014).

A segmentação em redes virtuais pode ser realizada por diversos protocolos, sendo o IEEE 802.1Q o mais amplamente utilizado, que permite a subdivisão de uma LAN em até 4094 VLAN's através da inserção de um tag de 4 bytes no quadro Ethernet, sendo 12 bits utilizados para o VID – VLAN ID, responsável pela identificação da VLAN (RECHIA, 2014).

O conceito pode ser melhor exemplificado pela figura abaixo:

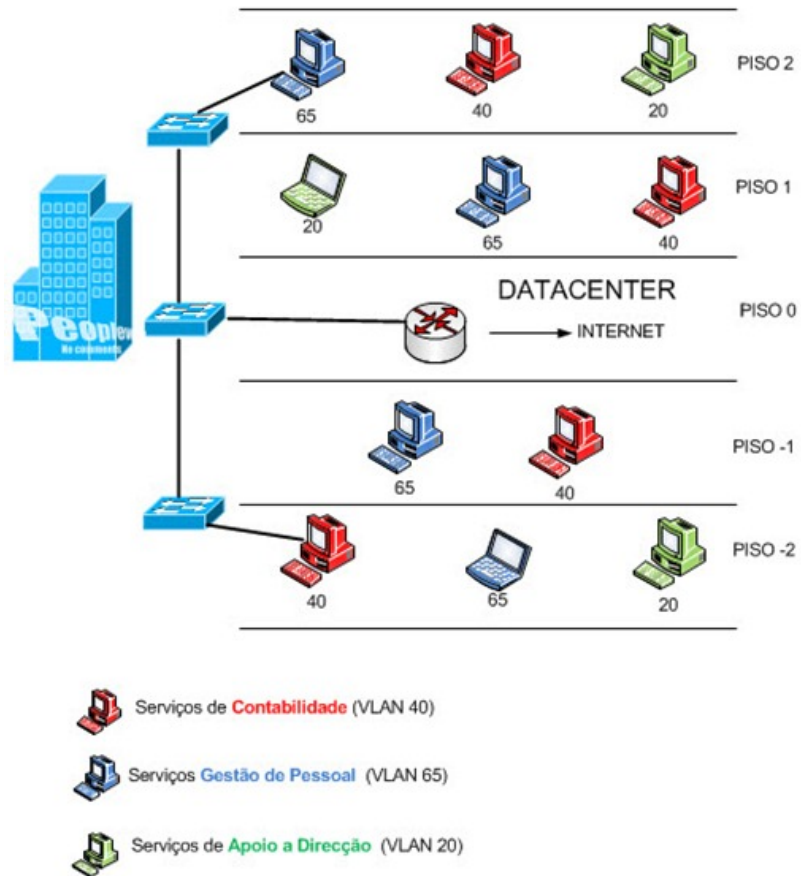


Figura 5: Exemplo de VLAN's

(Fonte: <https://pplware.sapo.pt> – Acesso em 13/11/2017)

Esses são os exemplos mais comuns de redes atuais. Outros modelos e plataformas são variações e implementações a partir desses modelos básicos.

4. Redes Definidas por Software

4.1 Definição de Redes Definidas por Software

O grande problema enfrentado pelos administradores de rede atuais é o grande volume de equipamentos a serem administrados quando a rede cresce em larga escala, exigindo equipes maiores e, conseqüentemente, aumentando os custos de manutenção das redes, além do maior tempo a ser gasto na análise e solução de problemas.

Lembramos também que com toda essa evolução das redes de computadores, a mesma tornou-se comercial e nesse sentido os equipamentos de rede tornaram-se “caixas-pretas”, ou seja implementações integradas baseadas em software e hardware proprietário. O resultado de toda essa evolução causou o já comentado engessamento das redes de computadores. Em contraste a essa realidade os pesquisadores de redes e a comunidade científica começaram a desenvolver novas propostas para a criação das redes de computadores do futuro, ou seja, novas arquiteturas de implementação do núcleo da rede (COSTA, 2013)

A principal proposta para essa nova arquitetura se baseia em redes capazes de serem programadas sob demanda, ou seja, redes que sejam programáveis ou que sejam flexíveis o suficiente para lidarem com requisitos e problemas que ocorram. Uma das formas de se transformar redes “estáticas” em redes “programáveis” é por meio da implementação de Redes Definidas por Software.

Redes Definidas por Software, ou redes programáveis, são redes são compostas por equipamentos de propósito geral, mas a função de cada equipamento é realizada por um software especializado.

Nas redes definidas por software, separam-se a camada de controle da camada de infraestrutura (outras publicações referem-se a essa camada como camada de dados). A camada de controle fica responsável pelos protocolos e pelas tomadas de decisão que resultam nas tabelas de encaminhamento ou roteamento. A camada de infraestrutura gerencia a comutação e repasse dos pacotes de rede.

Nos equipamentos atuais os planos de controle e de dados são executados e implementados no próprio equipamento, impedindo quaisquer tomadas de decisão que não tenha sido prevista nos protocolos implementados e configurados nesses equipamentos.

A Figura 6 a seguir exemplifica o conceito de SDN:

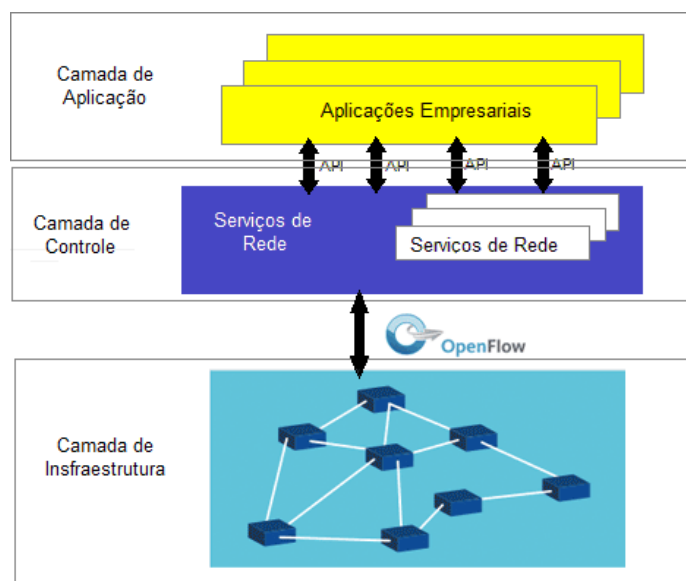


Figura 6: Estrutura básica do conceito SDN

(Fonte: <https://www.qta.ufri.br> – Acesso em 13/11/2017)

Para que haja uma tomada de decisão, é necessário quebrar essa restrição permitindo que o equipamento de rede encaminhe os pacotes por meio de protocolos e implementações externas sendo abrigadas em uma máquina física ou virtual. Nesse sentido o plano de controle será independente daquele pré-configurado e não se limitará aos protocolos implementados pelo proprietário ou fabricante (COSTA, 2013).

O *OpenFlow* é uma proposta que permite a implementação das Redes Definidas por Software, uma vez que estabelece uma interface de comunicação entre o plano de encaminhamento e o plano de controle com um padrão de interface bastante flexível para que seja possível a instalação de regras de encaminhamento baseadas em diversos parâmetros de protocolos de camadas distintas (COSTA, 2013).

4.2 O Protocolo OpenFlow

A premissa mais importante por trás do paradigma de Redes Definidas por Software é a capacidade de controlar, à distância, a tarefa de encaminhamento de pacotes, através de uma interface de programação bem definida. Foram desenvolvidos,

ao longo dos anos, alguns protocolos capazes de implementar essa comunicação entre dispositivos inseridos em uma SDN. Contudo, um desses protocolos está associado ao paradigma desde a sua concepção, e, de fato, foi o que tornou possível a implementação prática da SDN que é conhecida atualmente: o protocolo *OpenFlow* (CHINELATE, 2016).

O protocolo *OpenFlow*, desenvolvido na Universidade de Stanford, e proposto no trabalho de McKEOWN et al. (2008) em 2008, é um padrão aberto para Redes Definidas por Software que tem o objetivo de permitir que aplicações em software possam programar simplificadaamente as tabelas de fluxo dos dispositivos de comutação de pacotes presentes em uma rede, através de uma interface aberta de programação. Essa programação implica no controle desses dispositivos por uma entidade centralizada, o controlador, o que acaba ocasionando, na prática, a separação dos planos de dados e de controle da rede. Nesse ponto, vale ressaltar a diferença entre o paradigma SDN e o protocolo *OpenFlow*, que muitas vezes são considerados sinônimos. Enquanto SDN é um paradigma que consiste na ideia de separação dos planos, o *OpenFlow* descreve o modo como o software controlador e os *switches* devem se comunicar em uma arquitetura SDN (CHINELATE, 2016).

4.3. A Ferramenta Mininet

Quando pesquisadores querem testar os novos recursos SDN nos controladores, *switches* ou até mesmo no próprio protocolo *OpenFlow*, existem algumas dificuldades. Essas dificuldades ocorrem especialmente porque há poucos dispositivos baratos disponíveis que são capazes de implementar no padrão SDN (OLIVEIRA et al, 2014).

Além disso, em mais casos mais específicos, quando é necessário simular grandes redes com grande número de *hosts*, *switches* e controladores SDN, usar a própria Internet ou redes de produção pode não ser uma boa idéia, pois configurações errôneas podem causar problemas indesejados, tais como desconexões, *loops* de roteamento, inconsistências e indisponibilidade de serviços (OLIVEIRA et al, 2014).

Uma das soluções para este problema é fazer protótipos e simulá-los no modo virtual, e dessa forma pode-se criar situações onde eventuais falhas podem ser simuladas sem afetar sistemas em produção. Para fazer isso, algumas ferramentas foram criadas e o mais conhecido deles é o software Mininet. O Mininet é um sistema que permite a prototipagem rápida de grandes redes em um único computador, criando redes escaláveis

definidas por software usando virtualização leve, com pouco uso dos recursos do computador hospedeiro (OLIVEIRA et al, 2014).

Oliveira et al (2014) nos mostra que esses recursos permitem que o Mininet crie, interaja com, personalize e compartilhe os protótipos rapidamente.

Algumas características guiadas pela criação do Mininet são:

- 1) Flexibilidade: Novas topologias e novos recursos podem ser configurados em software, usando linguagens de programação e sistemas operacionais;
- 2) Aplicabilidade: Implementações corretas feita em protótipos também podem ser usado em redes reais baseadas em hardware sem alterações nos códigos-fonte;
- 3) Interatividade: Gerenciamento e execução da rede simulada deve ocorrer em tempo real como se acontecesse em redes reais;
- 4) Escalabilidade: O ambiente de prototipagem pode ser dimensionado para grandes redes com centenas ou milhares de switches em apenas um computador;
- 5) Realista: O comportamento do protótipo deve representar um comportamento em tempo real com um alto grau de confiança, então os aplicativos e protocolos devem ser utilizáveis em quaisquer modificações de código;
- 6) Compartilhado: Os protótipos devem ser facilmente compartilhados com outros colaboradores, que pode então executar e modificar as experiências.

A aparência do Mininet é bastante simplória, sendo tão somente uma interface texto, ou console, por onde é possível interagir com o ambiente por meio de comandos e scripts, como mostra a Figura 7.

```

u99319@ws116615-ubuntu: ~/mininet
File Edit View Search Terminal Help
u99319@ws116615-ubuntu:~/mininet$ sudo python ./test6.py
*** Configuring hosts
h1 h2 h3 h4
*** Starting CLI:
mininet> h1 ping -c2 h2
PING 10.0.0.2 (10.0.0.2) 56(84) bytes of data.
64 bytes from 10.0.0.2: icmp_req=1 ttl=64 time=11.7 ms
64 bytes from 10.0.0.2: icmp_req=2 ttl=64 time=0.168 ms

--- 10.0.0.2 ping statistics ---
2 packets transmitted, 2 received, 0% packet loss, time 1001ms
rtt min/avg/max/mdev = 0.168/5.962/11.757/5.795 ms
mininet> h1 ping -c2 h4
connect: Network is unreachable
mininet> h1 route add -net default h1-eth0
mininet> h2 route add -net default h2-eth0
mininet> h3 route add -net default h3-eth0
mininet> h4 route add -net default h4-eth0
mininet> h1 ping -c2 h4
PING 11.0.0.2 (11.0.0.2) 56(84) bytes of data.
64 bytes from 11.0.0.2: icmp_req=1 ttl=64 time=38.6 ms
64 bytes from 11.0.0.2: icmp_req=2 ttl=64 time=0.306 ms

--- 11.0.0.2 ping statistics ---
2 packets transmitted, 2 received, 0% packet loss, time 1001ms
rtt min/avg/max/mdev = 0.306/19.477/38.648/19.171 ms
mininet>

```

Figura 7: Exemplo de tela do Mininet

(Fonte: <http://windysdn.blogspot.com.br/2013/10/mininet-script-with-two-switches.html> – Acesso em 13/11/2017)

Atualmente existe uma implementação de interface gráfica denominada MiniEdit (Figura 8), ainda em fase experimental.

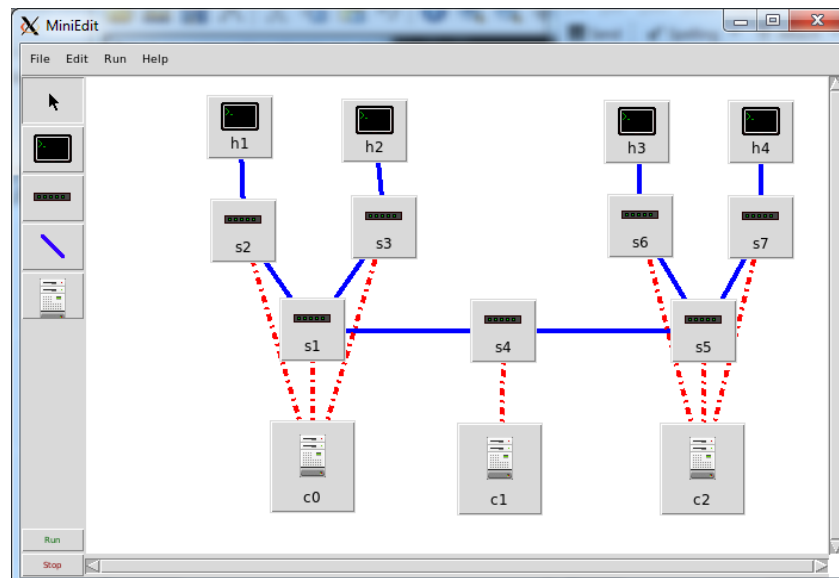
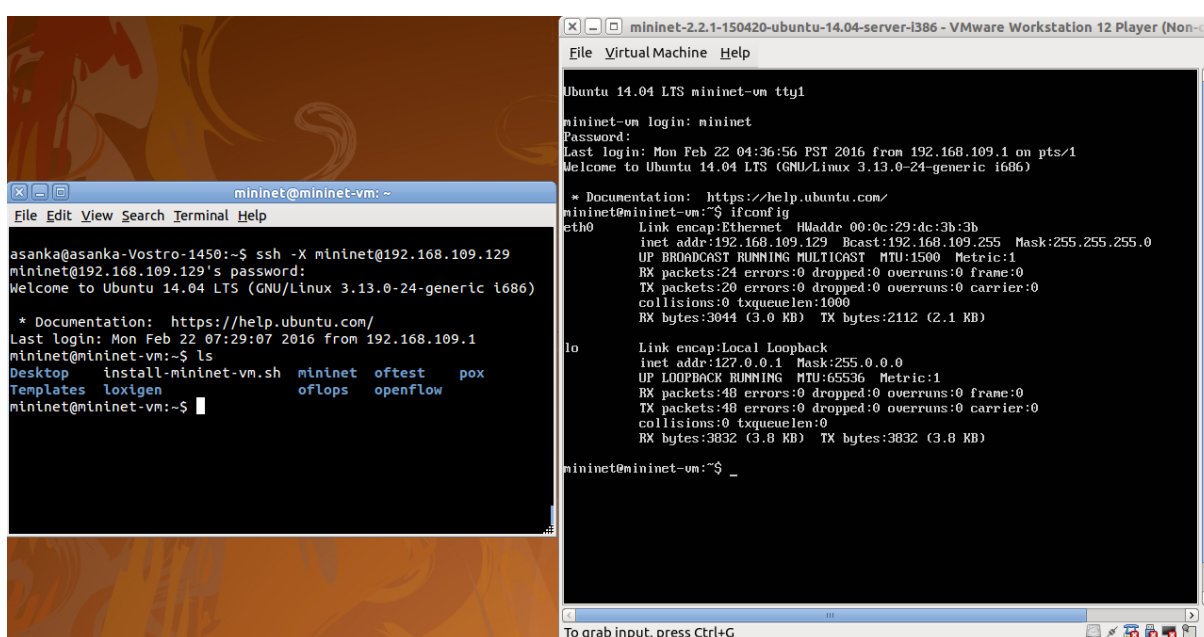


Figura 8: Tela do MiniEdit

(Fonte: <https://techandtrains.com/> - Acesso em 13/11/2017)

A forma usual de se obter o Mininet é como uma imagem de uma máquina virtual (VM), podendo ser baixado diretamente do site dos desenvolvedores (<http://www.mininet.org/download>). A máquina virtual é um Sistema Operacional Ubuntu 16.04.2 LTS com o Mininet instalado e já configurado. Basta efetuar o download da imagem e importá-lo para um software de execução de máquinas virtuais, tal como VirtualBox ou VMWare.

O acesso ao Mininet pode ser feita de forma direta pelo próprio terminal, ou então através de conexão SSH, como demonstrado na Figura 9.



```

mininet-2.2.1-150420-ubuntu-14.04-server-i386 - VMware Workstation 12 Player (Non-
File VirtualMachine Help

Ubuntu 14.04 LTS mininet-vm tty1

mininet-vm login: mininet
Password:
Last login: Mon Feb 22 04:36:56 PST 2016 from 192.168.109.1 on pts/1
Welcome to Ubuntu 14.04 LTS (GNU/Linux 3.13.0-24-generic i686)

* Documentation:  https://help.ubuntu.com/
mininet@mininet-vm:~$ ifconfig
eth0    Link encap:Ethernet  HWaddr 00:0c:29:dc:3b:3b
        inet addr:192.168.109.129  Bcast:192.168.109.255  Mask:255.255.255.0
        UP BROADCAST RUNNING MULTICAST  MTU:1500  Metric:1
        RX packets:24  errors:0  dropped:0  overruns:0  frame:0
        TX packets:20  errors:0  dropped:0  overruns:0  carrier:0
        collisions:0  txqueuelen:1000
        RX bytes:3044 (3.0 KB)  TX bytes:2112 (2.1 KB)

lo      Link encap:Local Loopback
        inet addr:127.0.0.1  Mask:255.0.0.0
        UP LOOPBACK RUNNING  MTU:65536  Metric:1
        RX packets:48  errors:0  dropped:0  overruns:0  frame:0
        TX packets:48  errors:0  dropped:0  overruns:0  carrier:0
        collisions:0  txqueuelen:0
        RX bytes:3832 (3.8 KB)  TX bytes:3832 (3.8 KB)

mininet@mininet-vm:~$ _

To grab input, press Ctrl+G

```

Figura 9: Conexão ao Mininet por SSH

(Fonte: <http://recolog.blogspot.com.br/> - Acesso em 05/11/2017)

4.4 Elementos SDN no Mininet

A ferramenta Mininet pode criar elementos SDN, customizá-los, compartilhá-los com outras redes e realizar interações. Esses elementos incluem hosts, switches, roteadores e links (conexões) (OLIVEIRA et al, 2014).

Um host no Mininet é um processo simples com sua própria rede em execução no sistema operacional. Cada um fornece processos com propriedade exclusiva da rede virtual tais como interface, portas, endereços e tabelas de roteamento (como ARP e IP) (OLIVEIRA et al, 2014).

Os *switches OpenFlow* criados pelo Mininet fornecem a mesma semântica de entrega de pacotes que seria fornecida por um hardware real. Na simulação Mininet, os controladores podem ser executados na rede real ou na rede simulada, desde que a máquina no qual os controladores estejam sendo executados tenham conectividade com os elementos controlados (OLIVEIRA et al, 2014).

Por padrão, o Mininet cria um controlador dentro do ambiente de simulação local e conexões virtuais também podem ser criadas entre os elementos através de suas interfaces virtuais.

4.5 Linhas de comando

Segundo a demonstração de Oliveira et al (2014), por padrão, a interação com o ambiente Mininet se dá por linhas de comando convencionais ou *scripts*.

Um exemplo básico seria o seguinte:

```
mininet> mn -topo single,3 -mac -switch ovsk -controller remote
```

O comando cria três *hosts* virtuais, cria um *switch OpenFlow* com 3 portas, conecta os *hosts* ao *switch* virtual usando links virtuais, seta os endereços MAC e endereços IP para cada *host* e finalmente configura o *switch* para conectá-lo ao controlador remoto. Nesse caso, o controlador está executando localmente no mesmo hardware onde o Mininet está em execução.

Depois de criados os elementos e todas as conexões entre eles, é crucial podermos executar comandos nos *hosts* para testar a conectividade da rede virtual e verificar a operação do *switch*.

Para esse propósito, o Mininet inclui comandos que permitem aos desenvolvedores, pesquisadores e estudantes controlar e gerenciar toda a rede.

Os comandos digitados são interpretados pelo emulador e executados no ambiente de rede simulado.

Eis alguns exemplos:

```
mininet > nodes - Mostra a lista de nós disponíveis;  
mininet > help - Mostra a lista de comandos de rede disponíveis;  
mininet > h2 ifconfig - Mostra o endereço IP do host, h2 no caso;
```

`mininet > h2 ping h1` - Envia um ping (ICMP REQUEST) do host2 para o host1;

Muitos exemplos e ferramentas, tais como *scripts* e aplicações gráficas, estão inseridos na máquina virtual que contém o Mininet. Duas ferramentas muito usadas para monitorar as redes Mininet são o *dpctl* e o *wireshark* (OLIVEIRA et al, 2014).

O *dpctl* é uma ferramenta que mostra e controla a tabela de fluxo dos switches. É especialmente usada para depuração de dados, por permitir visualizar e medir os fluxos de dados dos switches. Muitos switches que implementam o *OpenFlow* “escutam” esses comandos pela porta 6634 (por padrão), o que possibilita interagir com o *switch* sem a necessidade de depurar código no controlador (OLIVEIRA et al, 2014).

Por exemplo, o comando:

```
mininet> dpctl dump-flows tcp:127.0.0.1:6634
```

Conecta-se ao *switch* local e mostra a tabela de fluxo instalada. Também é possível adicionar regras à tabela de fluxo dos *switches* manualmente, como no exemplo abaixo:

```
mininet> dpctl add-flow actions=output:2 tcp:127.0.0.1:6634 in_port=1
```

Esse comando cria uma regra a qual todos os pacotes que chegam à porta 1 do switch são redirecionados à porta 2 (OLIVEIRA et al, 2014).

4.6 API e exemplos prontos

No site do projeto Mininet (<http://mininet.org>) podemos encontrar um completo manual de referência de sua API (*Application Programming Interface* ou, em português, Interface de Programação de Aplicativos), assim como scripts prontos com rotinas básicas que auxiliam nos primeiros passos nos estudos sobre SDN utilizando-se do Mininet.

No manual de referências encontramos a documentação completa do código que compõe a ferramenta Mininet, dividido em classes e sub-classes (*List*, *Hierarchy* e *Members*).

Já a página com exemplos provê vários *scripts* python já prontos para várias tarefas. Esses *scripts* têm o propósito de auxiliar o estudante nos primeiros passos nos estudos com o Mininet (<https://github.com/mininet/mininet/tree/master/examples>) .

Algumas ferramentas já vêm inclusas na máquina virtual Mininet.

A seguir temos o nome e descrição de alguns desses *scripts*:

- **bind.py**: Código que possibilita criar diretórios privados para cada topologia criada no Mininet.
- **clusterSanity.py**: Este exemplo executa a edição de cluster localmente como uma verificação de integridade para testar a funcionalidade básica.
- **cpu.py**: Este exemplo analisa a largura de banda iperf para variar os limites da CPU.
- **linearbandwidth.py**: Este exemplo mostra como criar uma topologia personalizada e como executar uma série de testes nela.
- **miniedit.py**: Este exemplo demonstra como criar uma topologia de rede via um editor gráfico.
- **nat.py**: Este exemplo mostra como conectar uma rede Mininet à Internet.
- **simpleperf.py**: Um exemplo simples de configuração de limites de largura de banda da rede e da CPU.

4.7 Wireshark

O Wireshark é uma ferramenta que captura e analisa todos os dados transmitidos pela interface de rede. Exclusivamente para o Mininet, o Wireshark possui uma biblioteca chamada *OpenFlow*, que filtra todos os pacotes *OpenFlow*, o que então permite que sejam analisados pela ferramenta. (OLIVEIRA et al, 2014).

A máquina virtual Mininet obtida no site do desenvolvedor já possui o Wireshark instalado e configurado para capturar pacotes *OpenFlow*, o que se mostra uma vantagem da máquina virtual sobre uma instalação do pacote Mininet em uma distribuição Linux, que demandaria instalações e configurações adicionais. A Figura 10 mostra o WireShark capturando pacotes icmp em um ambiente Mininet.

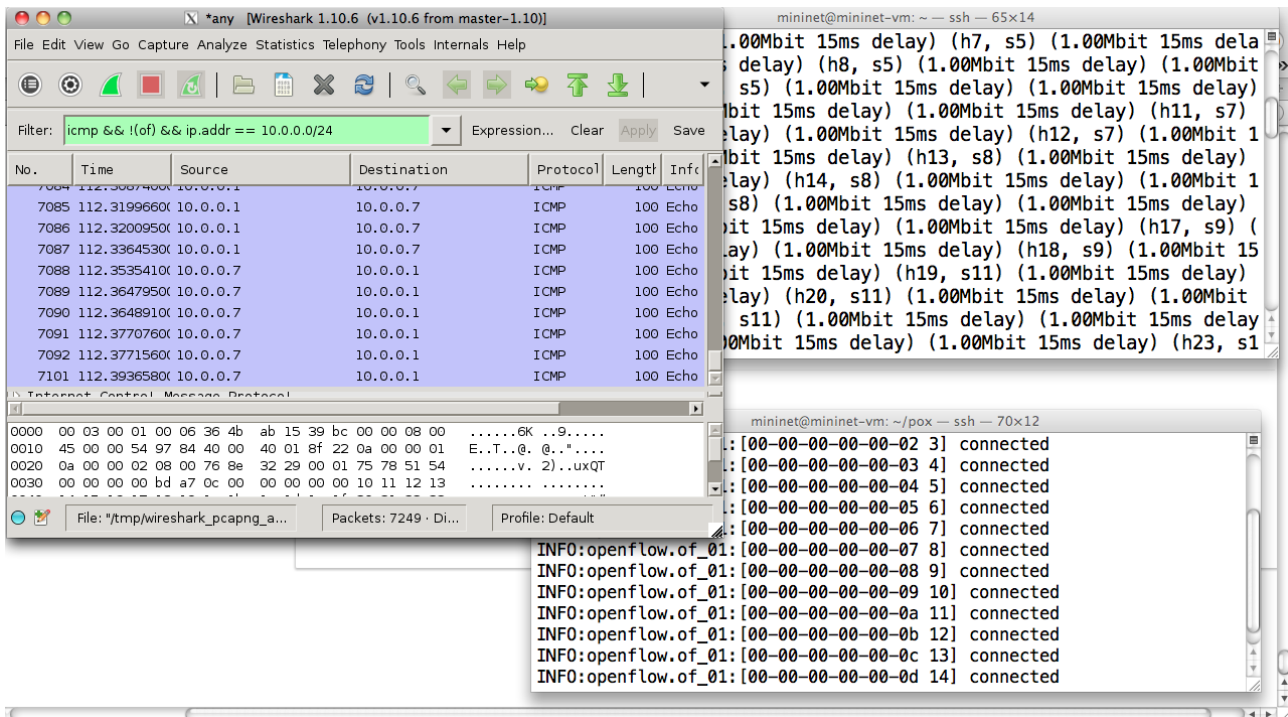


Figura 10: Tela do WireShark capturando pacotes

(Fonte: <https://datablast.wordpress.com/2014/10/31/discovering-rhipe-with-sdn-mininet/> - Acesso em 13/11/2017)

5. Conclusão e Implementações Futuras

Por ser de código aberto, possuir vasta documentação e uma comunidade ativa no desenvolvimento e suporte, o uso da ferramenta Mininet possibilita o desenvolvimento de estudos focados no âmbito da programação e controle de redes definidas por software, sem preocupações com elementos de rede físicos ou virtualizados. O protocolo OpenFlow, que é a base do Mininet, possibilita a programação da rede com um baixo nível de abstração, e torna possível controlar cada fluxo de tráfego da rede separadamente, porém essa tarefa exige conhecimentos sobre redes e programação mais aprofundados do que os vistos e sala atualmente, o que levaria à criação de novas disciplinas, ou talvez a fusão de algumas.

Como proposta para trabalhos futuros, propõe-se a aplicação da ferramenta Mininet na construção e desenvolvimento de ambientes de rede virtuais em sala de aula, implementando e aprimorando códigos que controlem e modifiquem esses ambientes, de forma a possibilitar um melhor entendimento do que são e como são controladas as redes definidas por software.

REFERÊNCIAS

BENNESBY, Ricardo da Silva. **Dissertação de Mestrado em Informática**. Instituto de Computação da Universidade Federal do Amazonas, 2013

CHINELATE, Leonardo Costa. **Balanceamento de Carga Utilizando Planos de Dados OpenFlow Comerciais** – Dissertação apresentada ao Programa de Pós-Graduação em Ciência da Computação da Universidade Federal de Juiz de Fora, 2016

COSTA, Lucas Rodrigues. **OpenFlow e o Paradigma de Redes Definidas por Software**. Monografia apresentada como requisito parcial para conclusão do Curso de Computação – UNB – Universidade de Brasília, 2013

FEI, QI, KE, Hu, Hao and Bao. **A Survey on Software-Defined Network and OpenFlow: From Concept to Implementation** - IEEE COMMUNICATION SURVEYS & TUTORIALS, VOL. 16, NO. 4, FOURTH QUARTER, 2014

KREUTZ, RAMOS, VERÍSSIMO, ROTHENBERG, AZODOLMOLKY e UHLIG, Diego, Fernando, Paulo Esteves, Christian Esteve, Siamak, and Steve. **Software-Defined Networking: A Comprehensive Survey**. Proceedings of the IEEE | Vol. 103, No. 1, January 2015

OLIVEIRA, Rogério Leão Santos et al. **Using Mininet for Emulation and Prototyping Software-Defined Networks**. IEEE *Xplore* Digital Library, 2014.

RECHIA, Felipe. **Estudo de Redes Definidas por Software e a sua Implantação em Redes Corporativas**. 110 f. Monografia (Especialização em Teleinformática e Redes de Computadores) – Programa de Especialização em Teleinformática e Redes de Computadores (Área de Concentração: Telemática), Universidade Tecnológica Federal do Paraná. Curitiba, 2014.

R. SHERWOOD, G. GIBB, K.-K. YAP, G. APPENZELLER, M. CASADO, N. McKEOWN, and G. PARULKAR. **Flowvisor: A network virtualization layer**. In Technical Report Openflow-tr-2009-1, OpenFlow. Stanford University, 2009.