Journal of Engineering and Technology for Industrial Applications, 2018. Edition. 13.Vol: 04
https://www.itegam-jetia.org
ISSN ONLINE: 2447-0228
DOI: https://dx.doi.org/10.5935/2447-0228.20180022

Heterogeneous network management through the implementation of a topology using the python programming language and software-defined networking technology

Philip Floriano Rodrigues Ramkeerat[1], Kleber Bittencourt Oliveira [2], Vanise dos Santos Rodrigues[3], Charles de Freitas Guimarães[4], Greyce dos Santos Rodrigues[5]

[1,2,3]Programa de Pós-graduação em Engenharia de Processos - PPEP/ITEC da Universidade Federal do Pará (UFPA)-Belém-PA.
[1]Sistemas e Programador do Instituto de Desenvolvimento Tecnológico INDT.
[4]Programa de Graduação em Ciência da Computação do Centro Universitário do Norte-UNINORTE -Manaus-AM.
[5]Programa de Pós-graduação em Ciência e Meio Ambiente da Universidade Federal do Pará (UFPA). Belém-PA.

Email: philip_ramkeerat@hotmail.com

Received: January 11th, 2018.

Accepted: February 28th, 2018.

Published: March 30th, 2018.

ABSTRACT

According to the report of the United Nations Conference of October 2, 2017 on information economy digitalisation and development states that Brazil is the fourth country with the most internet users in the world. With the expansion of the use of the Internet, more and more devices with different characteristics are connected in different ways in the network, causing the network to become increasingly heterogeneous. This heterogeneity causes difficulties with the administration of these devices connected in large scale in the traditional network architecture, with efficiency. In the traditional network model there are a number of problems when working with different devices, and this factor impairs the creation of network management systems. This study aims to propose improvements in network management by implementing a topology using the Python programming language in conjunction with some Mininet libraries. As a result, the generation of a graphical interface of simple use following the premises of the Networks Defined by Software that is a new paradigm of network infrastructure is presented. This interface allows easy control and management of the network due to the programming that results in the capture of the hardware of any device connected to the network, thus simulating access to its features as if it had real access to the hardware of the equipment.

Keywords: Software Defined Networks, Virtualization, Programming.

Gerenciamento de redes heterogêneas através da implementação de uma topologia utilizando a linguagem de programação python e a tecnologia de redes definidas por software

RESUMO

Segundo o relatório da Conferência das Nações Unidas de 02 de outubro de 2017 sobre economia da informação digitalização e desenvolvimento afirma que o Brasil é o quarto país com mais usuários de internet do mundo. Com a expansão do uso da *internet*, cada vez mais dispositivos com características distintas estão conectados de diferentes formas na rede fazendo com que a rede se torne cada vez mais heterogênea. Essa Heterogeneidade ocasiona dificuldades quanto a eficiência. No modelo tradicional de redes existem uma série de problemas quando se deseja trabalhar com dispositivos diferentes e esse fator prejudica a criação sistemas administradores de redes. Esse estudo visa propor melhorias no gerenciamento das redes através da implementação de uma topologia utilizando a linguagem de programação Python em conjunto com algumas bibliotecas do Mininet. Como resultado apresenta-se a geração de uma interface gráfica de simples utilização seguindo as premissas das Redes Definidas por *Software* que são um novo paradigma de infraestrutura de redes. Como resultado apresenta-se a geração de uma interface gráfica de simples utilização seguindo as premissas das Redes Definidas por *Software* que são um novo paradigma de infraestrutura de redes. Essa interface permite o controle e gerenciamento fácil da rede por conta da programação que resulta na captura do *hardware* de qualquer dispositivo conectado na rede, simulando assim o acesso as suas funcionalidades como se tivesse tendo acesso real ao hardware do equipamento.

Palavras-chaves: Redes Definidas por *Software*. Virtualização. Programação

I. INTRODUÇÃO

Atualmente, novos dispositivos e produtos de comunicação em rede possuem, cada vez mais, características distintas e suas formas de comunicação também são divergentes resultando em um aglomerado de redes heterogêneas, proporcionados pela tecnologia "Internet" das Coisas.

Estes produtos são lançados, a todo momento, e, portanto, endereços virtuais serão destinados para abranger esses novos hosts, fazendo com que os "Data Centers" dos provedores de serviços processem cada vez mais informações, já que a procura por esses itens tem mais adeptos e, por conseguinte a demanda por serviços em redes cresce de maneira exponencial a fim de atender às necessidades comerciais. Esse aumento do volume de dados trafegados na rede é devido aos avanços de tecnologias que utilizam e disponibilizam serviços em rede, como por exemplo, Computação em Nuvem.

A arquitetura convencional de redes tem dificuldades para atender esse constante crescimento do volume de dados e a pergunta que surgiu deste contexto foi: Como resolver o problema de administração dos recursos em rede?

Neste contexto, surgiram as Redes Definidas por "Software", as quais trouxeram em sua estrutura, maior flexibilidade e versatilidade proporcionados por uma interface de programação e geração de recursos similares ao hardware real dos equipamentos, ou seja, as redes podem ser programadas a fim de atender às solicitações e suprir as necessidades das empresas, podendo inclusive resolver os problemas sem a necessidade de acesso ao hardware dos equipamentos conectados na rede.

De modo que resultem na diminuição de complexidade na criação de ferramentas voltadas para o gerenciamento da mesma, haja vista, esta tecnologia culminar em uma desvinculação acentuada do "hardware" da rede a qual pertença, isto é, independente do fabricante de um equipamento informático, a interoperabilidade e consistência da rede será mantida. Essa comunicação padrão foi possibilitada pelo protocolo "OpenFlow", o qual é a base para as Redes Definidas por "Software".

Esta pesquisa visa desenvolver uma ferramenta de fácil manuseabilidade para administração de redes com as características das Redes Definidas por *Software* e testá-la em uma topologia de redes SDN (Rede definidas por software) criada na linguagem de programação *Python* em conjunto com as bibliotecas do software *Mininet*.

II. REVISÃO BIBLIOGRÁFICA

Atualmente muitas são as pesquisas desenvolvidas sobre a implementação de redes. A criação de uma plataforma para gerenciamento de uma rede heterogênea foi a pesquisa desenvolvida por [1], onde foi apresentado os principais conceitos envolvidos para implementação de uma Rede Definida por *Software* e da importância da virtualização para essa tecnologia. Através da virtualização foi possível criar servidores virtuais dedicados e por causa disso aumentou-se a escalabilidade em rede.

Propuseram a implantação das redes definidas por *software* para o gerenciamento de redes móveis. A novidade nesta pesquisa foi o auxílio das metodologias ágeis para a implantação dessa rede, e essa novidade foi, segundo os autores, muito satisfatória em relação ao objetivo pretendido, o qual era inserir essa tecnologia de maneira facilitada e rápida [2].

Apresentaram-se as vantagens trazidas pelas Redes Definidas por *Software* para o gerenciamento de recursos quando em rede [3]. Na estrutura deste projeto, houve uma análise dos dados obtidos da tabela de fluxo dos switches OpenFlow. Com base nestes dados, os autores concluíram que essa tecnologia é importante para o gerenciamento, já que este se tornou mais fácil e automático e também observaram uma melhora no roteamento dos pacotes que trafegaram pelo switch. Portanto houve ganhos também na otimização de recursos.

Em [4], os autores apresentaram os principais conceitos e fundamentos do protocolo *OpenFlow* e fundamentaram a inserção das Redes Definidas por *Software* para o controle e administração dos dispositivos da rede. Suscitaram as mazelas das redes tradicionais, embasando a necessidade de uma modernização desta estrutura de rede e para isto indicaram as Redes Definidas por *Software* para tal feito. Implementaram testes práticos na ferramenta *Mininet*, a qual forma redes SDN instantemente, com propósito de testar, se realmente é fácil a configuração deste tipo de rede. Os testes indicaram que as configurações são facilitadas porque foram feitas de forma automática.

Outros também contribuíram com pesquisas com foco em redes definidas por software dentre os quais citam-se: [5-10].

III. MATERIAIS E MÉTODOS

III.1 REDES DEFINIDAS POR *SOFTWARE*

Esta inovação tecnológica está sendo considerada um novo paradigma de redes de computadores e afins e sua inserção no mercado acontece em exponencial crescimento [7]. Caracteriza-se pela separação do plano de dados do de controle e isso é viabilizado por causa da possibilidade da programação das redes [10].

Apresenta-se flexibilidade e versatilidade porque é permitido o crescimento e atualizações da rede de forma simples e fácil. Um ponto importante é que as Redes Definidas por *Software* resolvem problemas de configurações de rede não imaginados anteriormente.

A figura 1 apresenta um exemplo de rede definidas por software.

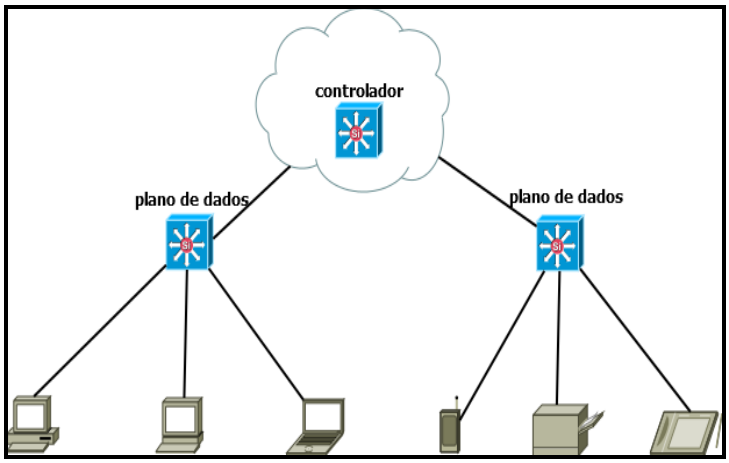

Figura 1: Exemplo de uma Rede Definida por *Software*
Fonte: Autores, (2018).

III.2 VIRTUALIZAÇÃO

Virtualização das funções de redes é uma técnica que possibilita o compartilhamento dos componentes da estrutura

física de uma rede por arquiteturas de rede heterogêneas a fim de usufruir dos mesmos elementos daquela rede, como por exemplo roteadores, comutadores[1], *switches* e etc.

A ligação desta tecnologia com as Redes Definidas por *Software* consiste no fato que os roteadores e *switches* podem ser virtualizados por meio da programação a fim de atender à procura de aplicações em rede.

A figura 2 apresenta um exemplo de servidores virtualizados.

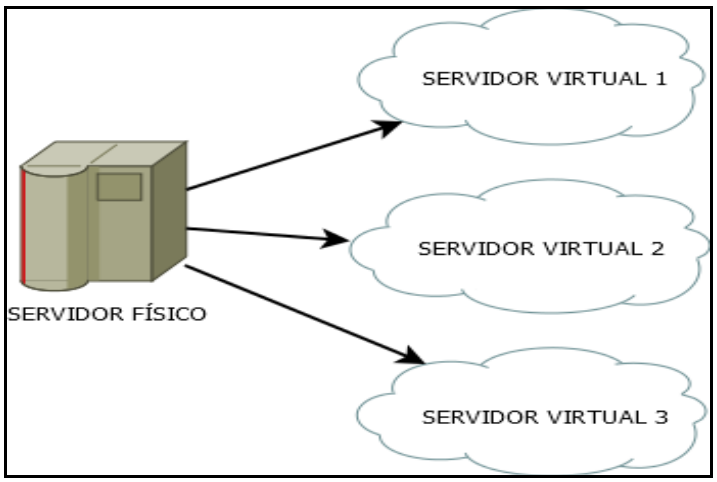

Figura 2: Exemplo de servidores virtualizados.
Fonte: Autores, (2018).

III.3 IMPLEMENTAÇÃO DE UMA REDE SDN EM *PYTHON* COM O *MININET*

Para executar as teorias propostas nessa pesquisa, o software *Mininet* se fez vantajoso, por conta das suas ferramentas disponibilizadas para o desenvolvimento de Redes Definidas por *Software*. *Além de apresentar uma l*inguagem de programação que possui uma ótima documentação de utilização, pois há uma grande comunidade empenhada para desenvolver projetos nessa linguagem denominada *Phython.*

Para criar a rede SDN proposta nesta pesquisa foram necessários utilizar os *softwares* discriminados a seguir.

✓ Sistema Operacional Linux/Ubuntu 16.4 LTS na Arquitetura de 64 *bits*

O sistema operacional Linux é o mais indicado para se trabalhar com o Mininet, pelo fato de que utilizando o mesmo, foi possível ter acesso a todas as funcionalidades do *software*, ou seja, ter liberdade total de criação de códigos de Redes Definidas por *Software* na linguagem de programação *Python*.

✓ GitHub

Para utilizar a instalação nativa do *Mininet*, foi preciso fazer o *download* do projeto que está hospedado no endereço

oficial[2] da comunidade de desenvolvimento do *Mininet*. Como o *Mininet* é uma ferramenta de código aberto, várias pessoas têm acesso ao projeto, com o intuito e contribuir com o desenvolvimento da plataforma.

[1]Comutadores são os agentes da rede que processam os pacotes e os encaminham para os diversos enlaces formados pelos hosts dessa rede.
[2] https://github.com/mininet/mininet

✓ Visual Studio Code

Este programa foi necessário porque permite suporte à linguagem *Python*, criações de funções, variáveis, além de sua extensa quantidade de *plug-ins* desenvolvidas.

✓ Linguagem de Programação Python

Para o desenvolvimento e configuração da topologia proposta com o esquema de 16 *hosts*, 5 *switches* e 1 controlador, foi necessário utilizar essa linguagem de programação Python que é a arquitetura do *Mininet* .

✓ Mininet

Esta ferramenta fornece um ambiente para implementações de Redes Definidas por *Software* e é uma das grandes referências nesse cenário atualmente, possuindo uma grande comunidade de desenvolvedores que sempre estão corrigindo e auxiliando na criação, aperfeiçoamento de novas funcionalidades nas tecnologias SDN. Resultando assim em uma extensa quantidade de métodos para se trabalhar com a criação de *scripts* de Redes Definidas por *Software*.

III.4 CRIAÇÃO DA TOPOLOGIA EM *PYTHON*

Para criar a topologia proposta de gerenciamento de uma Rede Definida por Software foi necessário o desenvolvimento de um script em Python que realizou a importação de bibliotecas do software Mininet.

Código: topologia.py

```
def myNetwork(): net = Mininet(topo = None, build = False, ipBase = '10.0.0.0/8')
info('*** Adicionando controller\n')
c0 = net.addController(name = 'c0', controller = Controller, protocol = 'tcp', port = 6633)info('*** Adicionando switches\n')
s1 = net.addSwitch('s1', cls = OVSKernelSwitch)
s2 = net.addSwitch('s2', cls = OVSKernelSwitch)
s3 = net.addSwitch('s3', cls = OVSKernelSwitch)
s4 = net.addSwitch('s4', cls = OVSKernelSwitch)
s5 = net.addSwitch('s5', cls = OVSKernelSwitch)
info('*** Adicionando hosts\n')
h1 = net.addHost('h1', cls = Host, ip = '10.0.0.1', defaultRoute = None)
h2 = net.addHost('h2', cls = Host, ip = '10.0.0.2', defaultRoute = None)
h3 = net.addHost('h3', cls = Host, ip = '10.0.0.3', defaultRoute = None)
h4 = net.addHost('h4', cls = Host, ip = '10.0.0.4', defaultRoute = None)
h5 = net.addHost('h5', cls = Host, ip = '10.0.0.5', defaultRoute = None)
h6 = net.addHost('h6', cls = Host, ip = '10.0.0.6', defaultRoute = None)
h7 = net.addHost('h7', cls = Host, ip = '10.0.0.7', defaultRoute = None)
h8 = net.addHost('h8', cls = Host, ip = '10.0.0.8', defaultRoute = None)
h9 = net.addHost('h9', cls = Host, ip = '10.0.0.9', defaultRoute = None)
h10 = net.addHost('h10', cls = Host, ip = '10.0.0.10', defaultRoute = None)
```

h11 = net.addHost('h11', cls = Host, ip = '10.0.0.11', defaultRoute = None)
h12 = net.addHost('h12', cls = Host, ip = '10.0.0.12', defaultRoute = None)
h13 = net.addHost('h13', cls = Host, ip = '10.0.0.13', defaultRoute = None)
h14 = net.addHost('h14', cls = Host, ip = '10.0.0.14', defaultRoute = None)
h15 = net.addHost('h15', cls = Host, ip = '10.0.0.15', defaultRoute = None)
h16 = net.addHost('h16', cls = Host, ip = '10.0.0.16', defaultRoute = None)

Essas bibliotecas possuem métodos pré-definidos facilitando assim a criação da topologia. Este código resultou em uma topologia com 16 hosts, 5 switches OpenFlow e um controlador.

Para executar o código na plataforma Mininet, dentro da pasta onde o código foi criado, foi necessário abrir um terminal no linux e digitar o seguinte comando: sudo ./topologia.py. O comando utilizado resultou na criação da topologia, conforme figura 3.

Figura 3:Comando usado no terminal para inicializar a topologia.
Fonte: Autores, (2018).

III.5 APRESENTAÇÃO DA TOPOLOGIA PROPOSTA

A representação visual da topologia criada no código **topologia.py**, é demonstrada na figura 4.

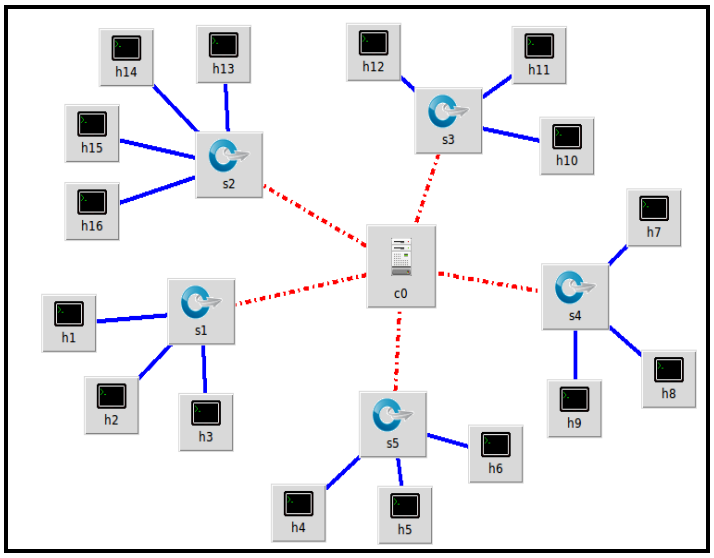

Figura 4: Representação visual da topologia.
Fonte: Autores, (2018).

A figura mostra um controlador que contém toda a inteligência da rede, 16 *hosts* com características distintas e 5 *switches OpenFlow*, pois estes equipamentos são os mais indicados para se trabalhar com Redes Definidas por *Software*.

III.6 TESTE DOS ENLACES DA REDE

Para confirmar o pleno funcionamento dos enlaces da rede foi realizado um teste de *ping*, conforme a figura 5, a qual mostrou o correto funcionamento da rede. O comando utilizado foi: **mininet> pingall.**

```
mininet> pingall
*** Ping: testing ping reachability
h5 -> h13 h14 h4 h15 h6 h16 h7 h8 h9 h10 h1 h2 h3 h11 h12
h13 -> h5 h14 h4 h15 h6 h16 h7 h8 h9 h10 h1 h2 h3 h11 h12
h14 -> h5 h13 h4 h15 h6 h16 h7 h8 h9 h10 h1 h2 h3 h11 h12
h4 -> h5 h13 h14 h15 h6 h16 h7 h8 h9 h10 h1 h2 h3 h11 h12
h15 -> h5 h13 h14 h4 h6 h16 h7 h8 h9 h10 h1 h2 h3 h11 h12
h6 -> h5 h13 h14 h4 h15 h16 h7 h8 h9 h10 h1 h2 h3 h11 h12
h16 -> h5 h13 h14 h4 h15 h6 h7 h8 h9 h10 h1 h2 h3 h11 h12
h7 -> h5 h13 h14 h4 h15 h6 h16 h8 h9 h10 h1 h2 h3 h11 h12
h8 -> h5 h13 h14 h4 h15 h6 h16 h7 h9 h10 h1 h2 h3 h11 h12
h9 -> h5 h13 h14 h4 h15 h6 h16 h7 h8 h10 h1 h2 h3 h11 h12
h10 -> h5 h13 h14 h4 h15 h6 h16 h7 h8 h9 h1 h2 h3 h11 h12
h1 -> h5 h13 h14 h4 h15 h6 h16 h7 h8 h9 h10 h2 h3 h11 h12
h2 -> h5 h13 h14 h4 h15 h6 h16 h7 h8 h9 h10 h1 h3 h11 h12
h3 -> h5 h13 h14 h4 h15 h6 h16 h7 h8 h9 h10 h1 h2 h11 h12
h11 -> h5 h13 h14 h4 h15 h6 h16 h7 h8 h9 h10 h1 h2 h3 h12
h12 -> h5 h13 h14 h4 h15 h6 h16 h7 h8 h9 h10 h1 h2 h3 h11
*** Results: 0% dropped (240/240 received)
```

Figura 5: Teste de ping.
Fonte: Autores, (2018).

III.7 DESENVOLVIMENTO DA PLATAFORMA DE GERENCIAMENTO SDN

Para facilitar o gerenciamento de uma rede SDN foi desenvolvida uma ferramenta na linguagem de programação Python, que realiza importações de métodos da biblioteca da plataforma Mininet que são declaradas no código fonte, a mesma com uma interface gráfica simples e intuitiva. Nesta estrutura existem diversas ferramentas para uma administração completa da rede.

A parte do código que apresenta a interface gráfica para o monitoramento de uma topologia SDN é o que se apresenta abaixo. O referido código irá gerar uma interface com botões de diferentes funcionalidades. Ao ser acionado, será ativada a sua função respectiva.

Código: interface.py

```
def createMenuBar( self ):
    "Create and return a menu (really button) bar."
    f = Frame( self )
    buttons = [
        ( 'Monitorar Hosts', lambda: self.select( 'hosts' ) ),
        ( 'Monitorar Switches', lambda: self.select( 'switches' ) ),
        ( 'Monitorar Controlador', lambda: self.select( 'controllers' ) ),
        ( 'Grafico Iperf', lambda: self.select( 'graph' ) ),
        ( 'Teste de Ping', self.ping ),
        ( 'Teste de Iperf', self.iperf ),
        ( 'Interromper', self.stop ),
        ( 'Limpar', self.clear ),
        ( 'Finalizar', self.quit )
```

IV. RESULTADOS E DISCUSSÕES

IV.1 APLICAÇÃO DA PLATAFORMA NA TOPOLOGIA PROPOSTA

Nesta seção serão explicados os procedimentos para a utilização prática da plataforma de Redes Definidas por Software que atende à premissa proposta no trabalho, de acordo com a topologia que foi criada no arquivo topologia.py e sua aplicação de interface no arquivo monitoramento.

Para executar a ferramenta de gerenciamento proposta neste trabalho, utilizou- se o seguinte comando: sudo ./monitoramento.py. Após a execução apareceram as mensagens indicando que os hosts, switches OpenFlow e o controlador foram inicializados, de acordo com a figura 6.

```
philip@philip-Lenovo-IdeaPad-Z400: ~/Documentos/mininet/philip
philip@philip-Lenovo-IdeaPad-Z400:~/Documentos/mininet/philip$ sudo ./monitoramento.py
[sudo] senha para philip:
*** Creating network
*** Adding controller
*** Adding hosts:
h1 h2 h3 h4 h5 h6 h7 h8 h9 h10 h11 h12 h13 h14 h15 h16
*** Adding switches:
s1 s2 s3 s4 s5
*** Adding links:
(s1, s2) (s1, s3) (s1, s4) (s1, s5) (s2, h1) (s2, h2) (s2, h3) (s2, h4) (s3, h5) (s3, h
6) (s3, h7) (s3, h8) (s4, h9) (s4, h10) (s4, h11) (s4, h12) (s5, h13) (s5, h14) (s5, h1
5) (s5, h16)
*** Configuring hosts
h1 h2 h3 h4 h5 h6 h7 h8 h9 h10 h11 h12 h13 h14 h15 h16
*** Starting controller
c0
*** Starting 5 switches
s1 s2 s3 s4 s5 ...
```

Figura 6: Inicialização dos dispositivos de rede.
Fonte: Autores, (2018).

Ao término desse procedimento é carregada uma janela contendo todos os botões que oferecem as ferramentas necessárias ao gerenciamento da Rede Definida por Software proposta, a qual reconheceu todos os dispositivos da topologia proposta e permitiu um controle central da rede. Ao clicar em um dos botões são ativadas funções respectivas ao que está proposto no seu contexto.

A figura 7 apresenta esta interface gráfica intuitiva.

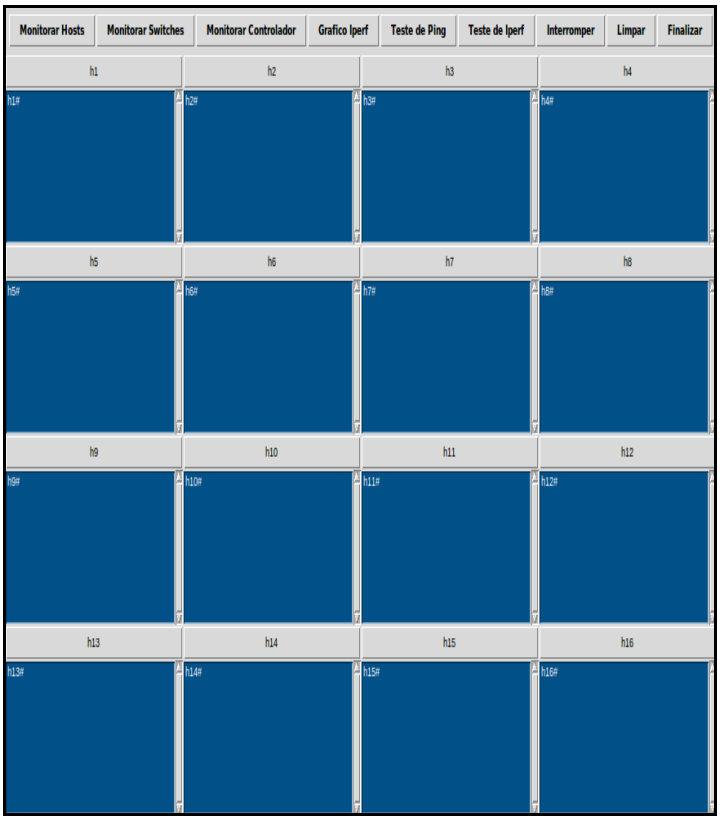

Figura 7: Interface gráfica que apresenta todas as funcionalidades para administração da rede.
Fonte: Autores, (2018).

IV.2 DESENVOLVIMENTO DA PLATAFORMA DE GERENCIAMENTO SDN

Para facilitar o gerenciamento de uma rede SDN foi desenvolvida uma ferramenta na linguagem de programação Python, que realiza importações de métodos da biblioteca da plataforma Mininet que são declaradas no código fonte, a mesma com uma interface gráfica simples e intuitiva. Nesta estrutura existem diversas ferramentas para uma administração completa da rede.

IV.3 SCRIPT EM PYTHON REFERENTE A INTERFACE GRÁFICA DO SOFTWARE PROPOSTO PARA O GERENCIAMENTO DE REDES SDN

A parte do código que apresenta a interface gráfica para o monitoramento de uma topologia SDN é o que se apresenta abaixo. O referido código irá gerar uma interface com botões de diferentes funcionalidades. Ao ser acionado, será ativada a sua função respectiva.

Código: interface.py

```
    def createMenuBar( self ):
        "Create and return a menu (really button) bar."
        f = Frame( self )
        buttons = [
            ( 'Monitorar Hosts', lambda: self.select( 'hosts' ) ),
            ( 'Monitorar Switches', lambda: self.select( 'switches' ) ),
            ( 'Monitorar Controlador', lambda: self.select( 'controllers' ) ),
            ( 'Grafico Iperf', lambda: self.select( 'graph' ) ),
            ( 'Teste de Ping', self.ping ),
            ( 'Teste de Iperf', self.iperf ),
            ( 'Interromper', self.stop ),
            ( 'Limpar', self.clear ),
            ( 'Finalizar', self.quit )
```

IV.4 INICIALIZANDO A INTERFACE DE MONITORAMENTO

Para executar a ferramenta de gerenciamento proposta neste trabalho, utilizou- se o seguinte comando: sudo ./monitoramento.py. Após a execução apareceram as mensagens indicando que os hosts, switches OpenFlow e o controlador foram inicializados, de acordo com a figura 8.

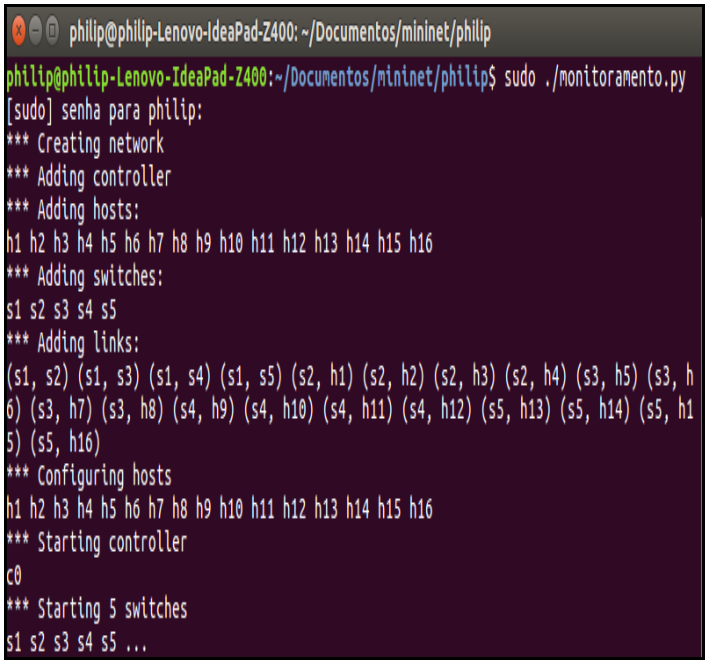

Figura 8: Inicialização dos dispositivos de rede.
Fonte: Autores, (2018).

Ao término desse procedimento é carregada uma janela contendo todos os botões que oferecem as ferramentas necessárias ao gerenciamento da Rede Definida por Software proposta, a qual reconheceu todos os dispositivos da topologia proposta e permitiu um controle central da rede. Ao clicar em um dos botões são ativadas funções respectivas ao que está proposto no seu contexto.

A figura 9 apresenta esta interface gráfica intuitiva.

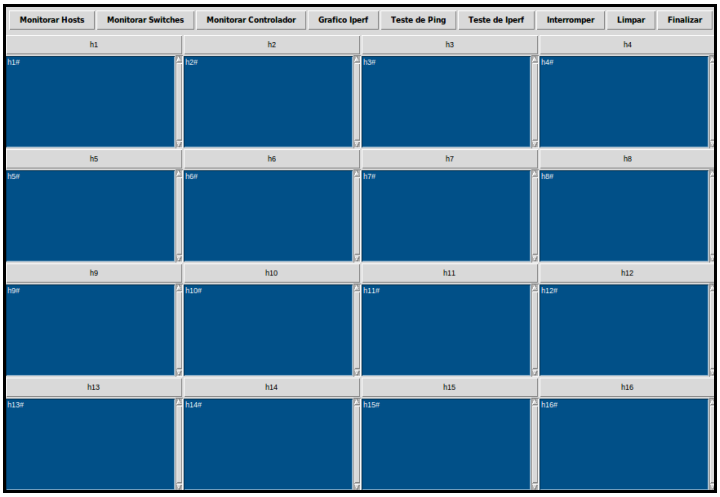

Figura 9: Interface gráfica que apresenta todas as funcionalidades para administração da rede.
Fonte: Autores, (2018).

IV.5 UTILIZANDO AS FUNCIONALIDADES DA INTERFACE

As funcionalidades foram desenvolvidas para facilitar o cotidiano de quem trabalha com o gerenciamento de redes SDN. Sendo assim, uma abstração das opções mais importantes envolvidas no processo de administração de redes foi implementada.

- ✓ Monitorar Hosts: Através da ativação do botão de monitoramento de hosts foi exibida uma interface gráfica que reconheceu todos os 16 hosts, 5 switches e 1 controlador que foram definidos no código topologia.py, e todas as operações que aconteceram por segundo nestes dispositivos na rede.
- ✓ Monitorar Switches: Ao ativar o botão de monitoramento dos switches foi exibida uma interface, que reconheceu a quantidade de switches que estavam presentes no código topologia.py, que nesse caso foram 5 switches OpenFlow. Apresentou também todos os processos e eventos ocorridos com os hosts que estão interligados com os mesmos e este monitoramento aconteceu por um intervalo de segundo a segundo.
- ✓ Monitorar Controlador: Ao ser acionado este botão, foi exibida uma interface com a quantidade de controladores presentes na rede, neste caso 1 controlador de acordo com o código desenvolvido na topologia.py.
- ✓ Gráfico de Iperf: O gráfico de iperf apresentou uma imagem da velocidade média do fluxo total de hosts definidos na topologia estudada, ou seja, uma forma mais intuitiva de visualizar o desempenho da rede, mais agradável ao usuário.
- ✓ Teste de ping: O teste de ping verificou o funcionamento da rede através da percepção dos envios e recebimentos de pacotes, o tempo que estão sendo necessários para realizar cada processo e a quantidade que está sendo enviada a cada host em um intervalo de segundo a segundo. Este teste tem relevância porque mostrou se também ocorreram falhas e perdas de pacotes, o que não foi o caso.
- ✓ Teste de Iperf: Este botão implementou uma verificação de conexão entre os hosts da rede para verificar o funcionamento da rede, tipos de pacotes que estão sendo enviados, se há falhas nas conexões, velocidade de conexão entre cliente-servidor.
- ✓ Interromper: Esta funcionalidade interrompe toda uma estrutura de rede, isto é, interrompe todos os hosts, controladores e switches OpenFlow integrantes da rede.

✓ Limpar: Esta funcionalidade exclui todas as informações presentes na interface em uso.

✓ Finalizar: Opção que encerra a ferramenta e fecha a janela.

IV.6 DEMONSTRAÇÃO DO TESTE DE PING NA INTERFACE

Neste procedimento foi realizado um teste de *Iperf* em cada *host* presente na topologia, especificando o intervalo de tempo, a quantidade de *bytes* transferidos e também a velocidade de banda em *Gbits* transferidas por segundo. Essas informações podem ser confirmadas clicando no botão gráfico de *Iperf*.

A figura 10 mostra a interface gráfica que apresenta o funcionamento do teste de ping. Nesta janela, em cada host, é exibido o tipo de pacote transmito, a quantidade enviada, o tempo necessário para envio e recebimento, o endereço de ip de destino para envio e também quantos pacotes foram recebidos.

Para a execução do teste de *Iperf* na interface, foi necessário a ativação desta funcionalidade no botão localizado na parte superior da interface, teste de iperf, como é demonstrado na figura 11.

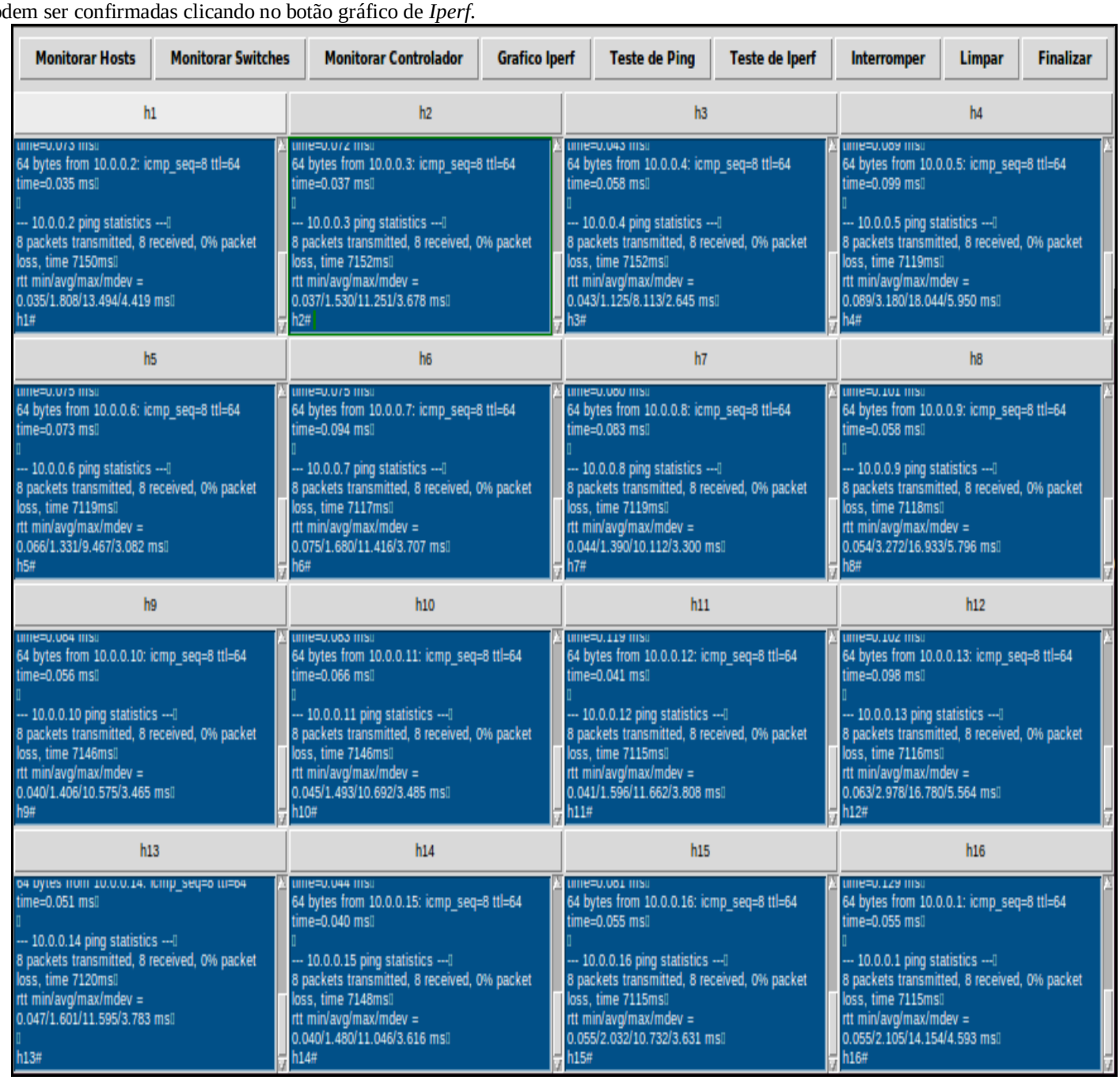

Figura 10: Interface gráfica que apresenta o funcionamento do teste de *ping.*
Fonte: Autores, (2018).

A figura 11 mostra o resultado para a Interface gráfica que apresenta o teste de iperf.

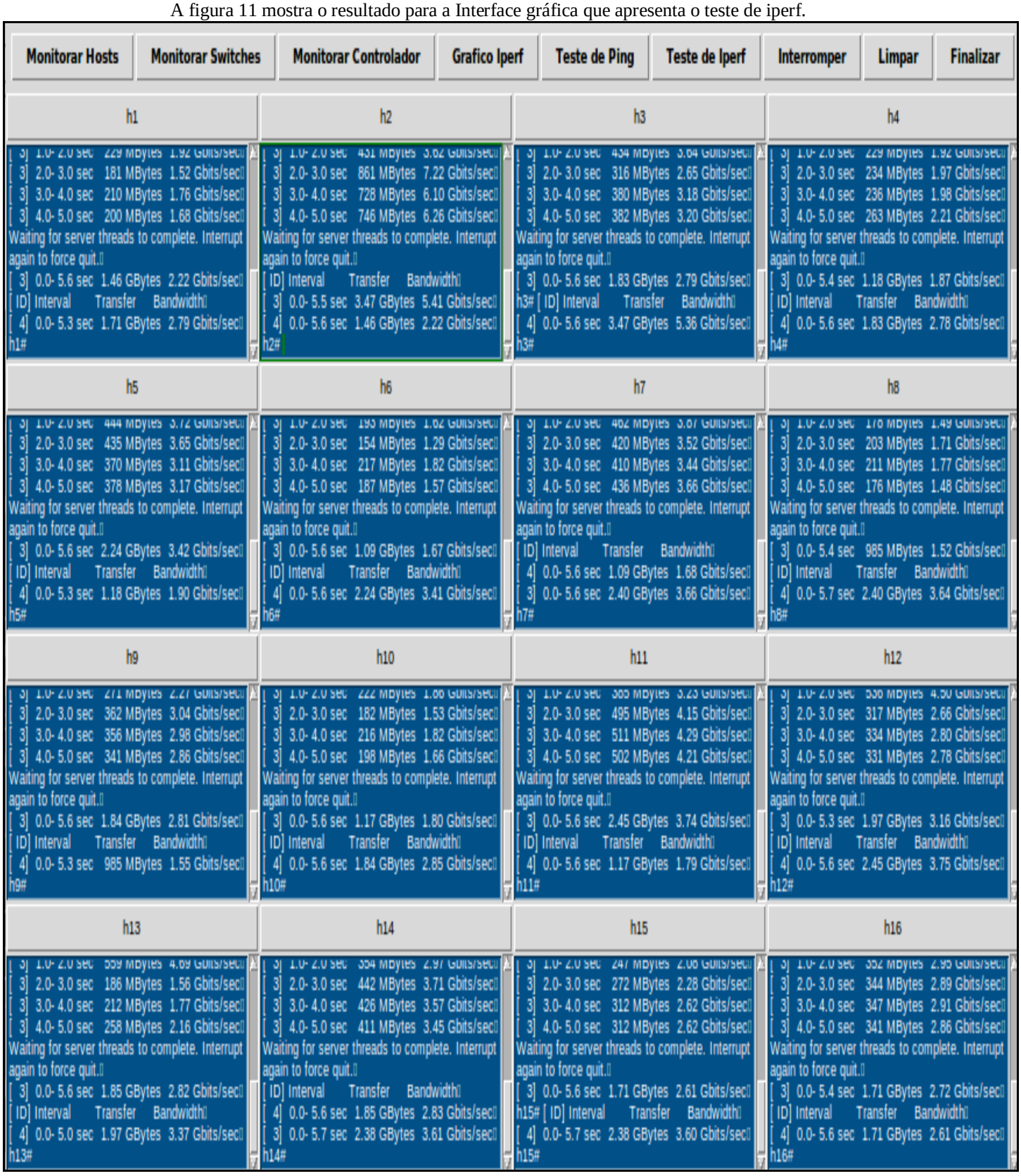

Figura 11: Interface gráfica que apresenta o teste de iperf.
Fonte: Autores, (2018).

IV.7 DEMONSTRAÇÃO DO GRÁFICO DE *IPERF* NA INTERFACE

Uma forma de verificar com mais clareza e simplicidade, os resultados dos testes de Iperf que foram apresentados na interface, o método gráfico de *Iperf* exibe a velocidade média de banda, dos dispositivos presentes na rede, em um gráfico, que pode ser visualizado ao clicar no botão gráfico *Iperf*. Conforme figura 12.

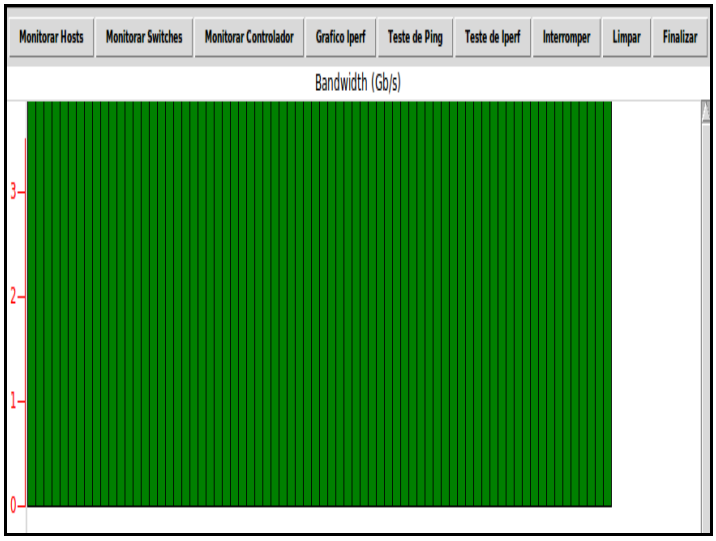

Figura 12: Resultados do teste de iperf.
Fonte: Autores, (2018).

A cada segundo é atualizado no gráfico uma barra verde, com a velocidade média de todos os *hosts*, a mesma continua aumentando enquanto não foi interrompida ou finalizados os procedimentos na rede, ou seja, é um gráfico dinâmico de monitoramento no intervalo de segundo a segundo.

V. RESULTADOS E DISCUSSÕES

Nesta seção serão apresentados os resultados percebidos após a criação e implementação da ferramenta proposta nesta pesquisa, com a topologia escolhida para realizar os testes. Como também os resultados que poderão ser aplicados de maneira geral em outros casos.

V.1 FACILIDADE DE GERENCIAMENTO

A ferramenta criada nessa pesquisa foi muito importante porque se apresentou como uma alternativa viável para resolver o problema proposto de gerenciamento de uma Rede Definida por *Software* com uma grande quantidade de *hosts*, *switches* envolvidos e um controlador principal. Não foi necessário o acesso ao *hardware* dos equipamentos, por conta da codificação em *Python* que trabalha em conjunto com a biblioteca do *Mininet*, criando assim uma camada virtual de *software* que simulou o *hardware* real dos equipamentos.

V.2 APRIMORAMENTO DE USABILIDADE AO USUÁRIO FINAL

Com base no trabalho proposto, monitorar 16 *hosts* consiste em uma tarefa extensa, visto que é necessário executar os comandos de teste de *host* a *host*, ou seja, se fosse necessário visualizar o *ping* entre um *host* e outro, deveria ser implementado explicitamente esse código, levando assim uma demanda alta de tempo para a sua conclusão e isso é ruim.

A ferramenta criada fez essa tarefa de forma automática e rápida. O benefício que o sistema trouxe para essa situação, foi que ao realizar o clique no botão, ocorreu de forma simples e eficaz, o teste de todos os *hosts* com base nos parâmetros de teste de *ping* definidos no código em *Python* utilizado no trabalho.

Para monitorar os *switches*, segue a mesma linha de raciocínio, ou seja, bastou apenas o clique do botão de teste correspondente e foi possível verificar o teste acontecendo em todos os *switches* que foram definidos na nossa topologia.

A interface desenvolvida foi uma forma de facilitar a usabilidade de uma série de tarefas ao usuário da ferramenta, abstraindo assim a necessidade de o usuário ter que saber programar em *Python*. O sistema realizou todas as chamadas necessárias para monitoramento e controle de rede através de botões de fácil acesso e intuitivos.

V.3 ADMINISTRAÇÃO DE REDES HETEROGÊNEAS

O sistema proposto tem a capacidade de gerenciar diferentes dispositivos, como por exemplo, computadores de marcar diferentes, sistemas operacionais diferentes, smartphones, tablets, videogames entre outros dispositivos que podem se conectar à internet, porque utiliza a vantagem de programação das redes a fim de resolver o problema de integração e também pelo fato de esta ferramenta ser baseada na tecnologia de Redes Definidas por *Software*, as quais possuem um protocolo padrão de comunicação.

V.4 REDUÇÃO DE TEMPO E CUSTO DE IMPLANTAÇÃO

A redução de tempo e custo é uma consequência da implementação de uma Rede Definida por Software, pelo motivo de que sua utilização remove uma série de problemas que eram vivenciados no modelo tradicional de trabalho com redes. Ou seja, as Redes Definidas por Software agilizam a forma como se criam sistemas de trabalho relacionado a redes em geral.

Eliminando o acoplamento com o hardware e criando assim uma nova oportunidade de mercado, um novo paradigma que só tende a crescer acompanhando as últimas tendências do mercado.

VI. CONCLUSÃO

Esta pesquisa apresentou, a viabilidade, as aplicações, o funcionamento e os benefícios das Redes Definidas por "Software" no contexto atual, o qual é predominantemente heterogêneo por conta das diversas marcas de dispositivos tecnológicos e seus "Softwares" embarcados, o que dificulta a comunicação entre esses equipamentos.

As Redes Definidas por "Software" permitem uma comunicação padrão, o que resolve o problema de integração e facilitam o gerenciamento de dispositivos distintos.

Esta pesquisa também apresentou uma ferramenta construída aos moldes das características das Redes Definidas por "Software" para administrar os dispositivos conectados em rede, a qual foi aplicada em uma topologia criada no "Software Mininet", contendo 16 "hosts", 5 "switches" e 1 controlador.

A linguagem de programação utilizada nesse estudo foi elaborado um software na linguagem de programação *Python*, com o intuído de colocar em prática todo o benefício que é proporcionado por esse novo paradigma de trabalho com redes.

De acordo com as pesquisas realizadas, foi possível notar que a tecnologia citada possui grande benefício de utilização no cenário tecnológico mundial por conta da sua extensa gama de possibilidades.

Eliminar a dificuldade criada por fabricantes que impossibilitam a comunicação dos *hardwares* de um determinado fabricante com o *hardware* de outro fornecedor foi uma grande vantagem criada pelas redes SDN, e mostra o potencial que existe de crescimento neste tipo de gerenciamento de redes.

Portanto as Redes Definidas por *Software* se adequam às expectativas esperadas pelas empresas, os quais são menor custo e qualidade de serviços e possibilitam, com eficiência, a escalabilidade da rede.

O *Mininet*, se mostrou muito eficiente em conjunto com a linguagem de programação *Python*, na criação de uma rede definida por *software* de teste com uma interface gráfica de fácil usabilidade ao usuário final utilizador da plataforma proposta. Desprendendo a necessidade de colocar os códigos manualmente, colocando toda a complexidade em botões intuitivos que acionam a funcionalidade de teste desejada pelo usuário.

VII. REFERÊNCIAS

[1] Córdova, Rodrygo Torres.**SDN–DMM: redes definidas por software para gerenciamento de mobilidade distribuído em redes,IP,móveis,heterogêneas**.

[2] Savoine, Márcia Maria et al. **Proposta de Uso de Métodos Ágeis no Gerenciamento e Implantação de Projeto de Redes sem Fio**. Anais SULCOMP, v. 8, 2017.

[3] Cardoso, Whasley S. et al. **Implantação de um Patch Panel Virtual Utilizando Redes Definidas por Software.** 2017.

[4] Giselle Silva e Faria. **Comparação das variáveis de atividade física fornecidas pelo acelerômetro actigraph gt3x e pelo aplicativo de celular google fit durante a marcha de indivíduos pós-acidente vascular encefálico.** Escola de Educação Física, Fisioterapia e Terapia Ocupacional da UFMG. Belo Horizonte. 2017.

[5] Heideker, Alexandre; Kamienski, Carlos. **Funções de Rede Virtualizadas em Plataforma de Computação em Nuvem para Cidades Inteligentes**. In: XIII Workshop em Clouds e Aplicações–WCGA. SBC. p43–56. 2015.

[6] Gupta, Arpit et al. Sdx: **A software defined internet exchange. ACM SIGCOMM Computer Communication Review**, v. 44, n. 4, p. 551-562, 2015.

[7] Rakesh, Bobba; Donald, R. Borries; ROD, Hilburn; Joyce, Sanders; Mark, Hadle; Rhett, Smith. **As Redes Definidas por Software Atendem aos Requisitos do Sistema de Controle**. AMEREN, Illinois.

[8] Guedes, Dorgival et al. **Redes Definidas por Software: uma abordagem sistêmica para o desenvolvimento de pesquisas em Redes de Computadores**. Minicursos do Simpósio Brasileiro de Redes de Computadores-SBRC,2012, v.30, n.4, p.160,210, 2012.

[9] Fernandes, Natalia Castro; Duarte, O. C. M. B**.** XNetMon: **Uma arquitetura com segurança para redes virtuais.** Anais do X Simpósio Brasileiro em Segurança da Informação e de Sistemas Computacionais, p. 339-352, 2010. Disponível.em:<http://www.lbd.dcc.ufmg.br/colecoes/sbseg/2010/0019.pdf >. Acesso em 11 de Agosto de 2017.

[10] Schaffrath, Gregor et al. **Network virtualization architecture: Proposal and initial prototype**. In: Proceedings of the 1st ACM workshop on Virtualized infrastructure systems and architectures. ACM, 2009. p. 63-72.

[11] Rabuske, Renato Antônio**. Inteligência artificial**. Universidade Federal de Santa Catarina, 1995.

[12] Mckeown, N. et al. O**penFlow: enabling innovation in campus networks**. ACMSIGCOMM Computer Communication Review, 2(38):69–74, March 2008.