UNIVERSIDADE FEDERAL DO PARÁ
INSTITUTO DE CIÊNCIAS EXATAS E NATURAIS
FACULDADE DE COMPUTAÇÃO
GRADUAÇÃO EM SISTEMA DE INFORMAÇÃO

LAERTH LASERINO PINTO MONTEIRO

MAPEAMENTO PROATIVO DOS HOSTS EM REDES DEFINIDAS POR SOFTWARES

BELÉM-PA

Abril/2017

LAERTH LASERINO PINTO MONTEIRO

MAPEAMENTO PROATIVO DOS HOSTS EM REDES DEFINIDAS POR SOFTWARES

Trabalho de Conclusão de Curso apresentado no curso de Bacharel em Sistema de Informação da Universidade Federal do Pará, como requisito parcial para obtenção do grau de Bacharel em Sistema de Informação.

Orientador: Prof.º Dr. Antônio Abelém

BELÉM-PA

Abril/2017

...

LAERTH LASERINO PINTO MONTEIRO

MAPEAMENTO PROATIVO DOS HOSTS EM REDES DEFINIDAS POR SOFTWARES

Trabalho de Conclusão de Curso apresentado no curso de Bacharel em Sistema de Informação da Universidade Federal do Pará, como requisito parcial para obtenção do grau de Bacharel em Sistema de Informação.

Aprovada em: __/__/____

BANCA EXAMINADORA

Prof.º Dr. Antônio Abelém
Universidade Federal do Pará
Orientador

Prof. Dr. Eduardo Cerqueira
Universidade Federal do Pará

Prof. MSc. Vagner Nacimento
Universidade da Amanzônia

Aos meus Pais

Agradecimentos

Dedico meus sincero agradecimentos para:

Professor Dr. Antônio Abelém pela oportunidade de trabalhar em um grupo de pesquisa GERCOM e pelo direcionamento acadêmico. Ao MSc. Fernando Farias que ajudou no início da proposta, e me deu um tema. Ao MSc. Vagner Nacimento que me abriu as portas para o grupo e contribuiu com ideis durante o andamennto da proposta e na desfesa. Ao Dr. Bily Anderson Pinheiro que ajudou em momentos cruciais e sempre nas horas de grande raciocínio e reflexão. Ao meu caro amigo Paulo Almeida. Aos caros André Bahia, Alexander Nunes, Airton Ishimori e aos demais parceiro do GERCOM.

A minha noiva Naila Provenzano pelo apoio e paciência.

Aos meus Professores e Professoras.

Aos meus pais Raimundo Monteiro e Leopoldina Pinto.

Resumo

Resumo do trabalho de Conclusão de Curso apresentado no curso de Bacharel em Sistema de Informação da Universidade Federal do Pará, como requisito parcial para obtenção do grau de Bacharel em Sistema de Informação.

Mapeamento Proativo dos Hosts em Redes Definidas por Softwares

Orientador: Prof.º Dr. Antônio Abelém
Palavras-chave: Redes Definidas por Softwares, Segurança, Controle de Acesso.

As Redes Definidas por Software separam o plano de controle do plano de dados, provendo visão geral da rede e alta programabilidade. A adoção deste paradigma tem crescido em empresas e universidades. A garantia de monitoramento constante na rede, controle de acesso e guarda de informações dos hosts que usam os serviços, são algumas formas de melhorar a segurança da rede. Desta forma, propomos uma Aplicação de Mapeamento Proativo dos Hosts para Redes Definidas por Softwares, assim mantendo informações dos hosts e dispositivos persistentes no banco de dados. Os Resultados mostram o correto funcionamento da aplicação, onde as informações do hosts continuam armazenadas mesmo depois que este desassociam-se da rede e o controlador é desligado.

Abstract

Summary of Course Completion of work presented in the course of Bachelor of Information System of the Federal University of Pará, as a partial requirement for the degree of Bachelor of Information System.

Proactive Mapping of Hosts on Software Defined Networks

Advisor: Prof.º Dr. Antônio Abelém
Key words: Software Defined Networks, Security, Access Control.

Software Defined Networks separate the data plan control plan, providing network overview and high programmability. The adoption of this paradigm has grown in companies and universities. The assurance of constant network monitoring, access control, and storage of information from hosts using the services are some ways to improve network security. In this way, we propose a Proactive Mapping Application of Hosts for Software Defined Networks, thus keeping information of hosts and persistent devices in the database. The results show the correct operation of the application, where the information of the hosts remains stored even after it disconnects from the network and the controller is turned off.

Sumário

Lista de Abreviaturas

SDN Software Defined Networks
TCP Transmission Control Protocol
IP Internet Protocol
IF Internet of the Future
IEEE Institute of Electrical and Electronics Engineers
SDWN Software Defined Wireless Networks
LAN Local Area Network
VLAN Virtual Local Area Network
QoS Quality of Service
EAP Extensible Authentication Protocol
DOS Denial of Service
AP Access Point
GNU Gnu is Not Unix
GPL General Public License
IMP Interface Message Processor
CIDR Classless Internet Domain Routing
NAT Network Address Translator
Intserv Servicos Integrados
Diffserv Servicos Diferenciados
OVS Open vSwitch
OVSDB Open vSwitch Database
MVC Model-View-Controller
CLI Command-Line Interface
MAC Media Access Control
CLNP Connectionless Network Protocol
XCP Explicit Control Protocol
ONOS Open Network Operating System
DHCP Dynamic Host Configuration Protocol
RADIUS Remote Authentication Dial In User Service Support

ARPANET (Advanced Reserach Projects Network)

Lista de Figuras

CAPÍTULO 1

Introdução

Segundo Jim Kurose, a Internet de hoje é provavelmente o maior sistema de engenharia já criado pela humanidade, com centenas de milhões de computadores conectados, enlaces de comunicação e comutadores. Bilhões de usuários que se conectam através de *notebooks*, *tabletes*, *smartphones*, e uma série de dispositivos como sensores, webcams, consoles para jogos, quadros de imagens e até mesmo máquinas de lavar sendo conectadas [1].

A Internet é usada para uma grande variedade de propósitos comerciais e não comerciais. Para muitas pessoas é uma ferramenta de trabalho crucial, para outras, seu local de trabalho. Porém, para a grande maioria, é um eficiente meio de comunicação, entretenimento ou plataforma educacional. Ou seja, a Internet é um enorme sucesso mundial e vem mudando a forma como interagimos, trabalhamos e nos divertimos [2].

1.1 Visão geral

A arquitetura da Internet, projetada há aproximadamente 40 anos, sofreu muitas mudanças nos anos recentes para incluir novas funcionalidades, as quais não foram previstas no projeto inicial. Essa evolução até os dias atuais tem sido sempre por meio de adaptações de protocolos, ditos muitas vezes como "remendos", buscando uma melhoria momentânea para algo em específico [2].

Neste contexto existe uma constante demanda por melhorias na infraestrutura e a necessidade de novas tecnologias executando no núcleo da rede. Contudo, há uma limitação na velocidade desses avanços que, em grande parte, se dá pelo forte atrelamento entre hardware e software. Na maioria das vezes o hardware é fabricado com o respectivo software instalado e com todos os seus recursos pré-definidos. Qualquer extensão de suas

funções só poderá ser executada pelo fabricante e, muitas vezes, mediante a aquisição de licenças de uso [3].

Além disso, a tecnologia básica IP é a causa das suas próprias limitações, que se tornam cada vez mais evidentes. A noção central de tamanho único, que requer tratamento idêntico para todos os fluxos de informação na Internet ao nível do pacote IP, não é desejável e nem necessariamente econômica, especialmente quando certas classes de aplicação, tais como de mídia interativa ou de acesso remoto a instrumentos científicos que, requerem garantias de qualidade de serviço *(Quality of Service - QoS)* desnecessárias para a maioria de outras aplicações.

Devido às dificuldades encontradas recentemente na rede, existe um consenso de que a Internet precisará ser reformulada, criando a Internet do Futuro. Essa nova Internet deve manter os princípios que levaram ao sucesso atual, tais como a facilidade para implantação de novas aplicações e a adaptabilidade de seus protocolos, mas deverá possuir conceitos novos, tais como auto cura e autogerenciamento, podendo absorver alguns princípios de inteligência e conhecimento [4].

Redes Definidas por Softwares surgem com um novo paradigma, onde a lógica de funcionamento de uma rede é repensada e o controle deixa de ser distribuído entre os dispositivos, que perdem sua autonomia e passam a obedecer às decisões de um Controlador. Assim a rede transita para um comportamento dinâmico, possibilitando mudanças em grande escala nas regras de encaminhamento de pacotes em cada dispositivo através de um ponto central de gerenciamento.

1.2 Motivação e Desafios

As motivações encontradas para a elaboração deste trabalho surgiram da possibilidade do desenvolvimento de uma aplicação de Mapeamento Proativo dos Hosts em Redes Definidas por Softwares, que garanta: a segurança no controlador, segurança da comunicação entre o controle e o plano de dados, e a confiança entre os componentes da rede.

As outras motivações são: poucas soluções deste tipo que trabalha nativamente em um controlador de Redes Definida por Software e a possibilidade de simplificar o modelo de controle de acesso de dispositivos em uma rede.

1.3 Objetivos

Este trabalho tem como objetivo geral o desenvolvimento de uma aplicação de Mapeamento Proativo dos Hosts em Redes Definidas por Softwares. O aplicativo será capaz de prover o armazenamento das informações dos usuários conectados ao roteador modificados com o sistema operacional embarcado baseado em Linux, o *OpenWRT*. Entre

os objetivos específicos pretende-se:

- Realizar um levantamento dos trabalhos relacionados á proposta;
- Desenvolver aplicação de Mapeamento Proativo dos Hosts em Redes Definidas por Softwares;
- Realizar o desing da interface de usuário do serviço;
- Montar ambiente de teste real.

1.4 Organização do texto

A organização do trabalho está dividida da seguinte maneira: O capítulo dois abordará Internet do Futuro e sua Evolução, Redes Definidas por Software (SDN), protocolo *OpenFlow*, controlador ONOS, Desafios de Segurança em SDN, sistema operacional embarcado *OpenWRT* e comutador virtual *Open VSwitch*. No capítulo três será explanado os trabalhos relacionados, Ethane, Resonance e AuthFlow. No capítulo quatro será descrita a proposta do trabalho. No capítulo cinco a apresentação dos resultados e no capítulo seis as conclusão e referências bibliográficas.

CAPÍTULO 2

Internet do Futuro

Neste capítulo é apresentada uma visão geral sobre Internet Atual e sua Evolução, Redes Definidas por Software (SDN), Protocolo OpenFlow, Controlador ONOS, Desafios de Segurança em SDN, sistema Operacional Embarcado OpenWRT e Comutador Virtual Open vSwitch.

2.1 Internet Atual e sua Evolução

A Internet foi projetada dando ênfase à generalidade e heterogeneidade na camada rede. Sua estrutura é baseada nos princípios de um núcleo de rede simples e transparente com a inteligência nos sistemas finais que são ricos em funcionalidades. Além disso, é baseada na premissa de ser descentralizada e dividida em múltiplas camadas autônomas [5].

A primeira rede de comutação de pacotes foi a ARPANET *(Advanced Reserach Projects Network)* encomendada pelo Departamento de Defesa dos Estados Unidos em 1969. Era a época da "Guerra Fria" e o governo norte-americano, temendo um ataque soviético ao Pentágono, tinha como objetivo desenvolver uma rede de comunicação que não o deixasse vulnerável. As redes telefônicas eram organizadas segundo uma hierarquia que levava a pontos centrais, de forma que o ataque a alguns desses pontos poderá desestruturar toda a rede. Assim, a ARPANET foi projetada e construída para ser uma rede altamente distribuída e tolerante a falhas, utilizando comutação de pacotes.

Os enlaces da ARPANET eram de apenas 56kbps, onde cada nó, constituído por uma estação e um comutador chamado de Interface Message Processor (IMP), deveria ser conectado a dois outros nós para criar caminhos alternativos e, com isto, garantir a confiabilidade no caso de falhas em alguns nós. Os IMPs eram responsáveis pela divisão

dos dados em pacotes e pelo encaminhamento dos dados. Foram criados protocolos de rede e de estação para permitir a comunicação na rede. Os protocolos de rede eram caracterizados pelo protocolo de comunicação entre IMPs vizinhos e pelo protocolo de comunicação entre o IMP de origem e o IMP de destino.

No início da década de 70, vários nós foram inseridos na ARPANET o que tornou aparente as dificuldades para interconexão de diferentes redes. Assim, Vint (Vinton Gray) Cerf e Bob (Robert) Kahn propuseram o Transmission Control Program (TCP), que introduzia o conceito de gateways que servem para interconectar duas redes separadas de comutação de pacotes. Além disso, o TCP especificava a criação e destruição de conexões lógicas entre processos utilizando pacotes de diferentes tamanhos, tratava e recuperava erros de transmissão e falhas de sequenciamento de pacotes, além de realizar controle de fluxo e a verificação de erros fim-a-fim. Esse programa também tratava o endereçamento dos nós e o encaminhamento de pacotes, constituindo-se o marco para a criação da Internet.

Posteriormente, no início da década de 80, o Transmission Control Program foi subdividido em dois protocolos, o Transmission Control Protocol (TCP) e o Internet Protocol (IP), sendo responsáveis pelo transporte e encaminhamento dos dados trafegados na rede. Assim, nascia o modelo TCP/IP, que se tornou o modelo de referência de arquitetura da Internet. Muitos consideram a ARPANET a mãe da Internet e a criação do modelo TCP/IP a origem da Internet atual [6].

A arquitetura TCP/IP está organizada em quatro camadas: a camada de aplicação, camada de transporte, camada de rede e a camada de enlace/físico.

Figura 1: Protocolo TCP/IP

Camada de Aplicação: tem a função de prover serviços para as aplicações criando a comunicação, por meio de uma rede, com outras aplicações. Camada de Transporte: 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. Camada de Rede: é responsável pelo roteamento dos pacotes entre origem e destino. Camada de Enlace/Física: 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.

A utilização do modelo em camadas, a transparência e o princípio fim-a-fim permitiram o crescimento da Internet. No entanto o fato da sua popularidade e do seu tamanho terem crescido absurdamente durante os últimos anos levou a incapacidade de

sua estrutura suportar o grande número de usuários. Hoje as mesmas causas que levaram ao crescimento da Internet são as causas do seu engessamento e restrição do seu crescimento.

A capacidade da arquitetura da internet para absorver remendos está se esgotando, para superar esta limitação de Internet, precisará ser reformulada, criando a Internet do Futuro. Essa nova Internet deve manter os princípios que levaram ao sucesso atual, tais como a facilidade para implantação de novas aplicações e a adaptabilidade de seus protocolos, mas deverá possuir conceitos novos, tais como autogerenciamento, mobilidade e escalabilidade, podendo absorver alguns princípios de inteligência e conhecimento. Foram propostos alguns remendos para aplicar na arquitetura SDN:

- Classless Internet Domain Routing (CIDR);
- Network Address Translator (NAT);
- Serviços Integrados (Intserv);
- Serviços Diferenciados (Diffserv);
- IP Móvel (Mobile IP).

Existem duas possíveis abordagens para Internet do Futuro:

- Evolucionária
  - Compreender o comportamento da Internet atual;
  - Identificar os problemas existentes ou emergentes e resolve-os sobre duas limitações: compatibilidade e implantação gradual.
- Lousa Limpa (Clean Slate) Concepção de uma nova arquitetura que é significativamente melhor (em termos de desempenho, segurança...) sem estar limitado pela arquitetura da Internet atual. "Clean-slate" não é algo novo, já foram desenvolvidos protocolos e arquiteturas tais como: *CLNP, XCP, Nimrod*, etc. Não funcionam muito bem com o sistema legado. Para validar e adequar uma proposta nova em Internet do Futuro são necessários ambientes experimentais (testbeds):
  - Cujo tráfego não interfere com o tráfego de produção da Internet atual;
  - Permitem avaliar e medir o impacto técnico, quando possível, social e econômico das abordagens;
  - Permitem observar comportamentos e anomalias que caracterizam ambientes de produção (em escala grande).

2.2 Redes Definidas por Software - SDN

Com o enorme crescimento e evolução da Internet, observamos que sua arquitetura não evoluiu suficientemente nos últimos anos. Embora todas as modificações realizadas, a arquitetura de Internet já não está mais atendendo as demandas das novas aplicações.

Toda essa evolução das redes de computadores, 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. 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 [7].

Uma das formas de se prover programabilidade às redes é por meio da implementação de Redes Definidas por Software. Redes Definidas por Software (SDN), ou redes programáticas, são redes cujo base física é composto por equipamentos de propósito geral e a função de cada equipamento, ou conjunto de equipamentos, é realizada por um software especializado.

As Redes Definidas por Software sugerem a separação entre o plano de dados (também chamado de plano de encaminhamento) responsável pelo encaminhamento dos pacotes com base em regras, e o plano de controle, responsável pelo controle da rede em geral, permitindo ao controlador o gerenciamento das entradas da tabela de encaminhamento e das regras associadas ao tráfego desejado, dessa maneira proporcionando uma melhor visão global sobre toda a rede e ficando a cargo dele toda a inteligência da rede [8].

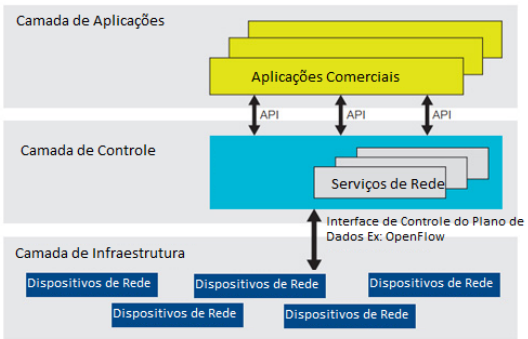

Figura 2: Redes Definidas por Software (SDN)

Segundo JUNIPER (2013) as SDNs, quando comparadas com as redes tradicionais, apresentam alguns princípios e benefícios, listados abaixo:

- Separar de forma clara o software das redes em quatro camadas: Gerenciamento, serviços, controle e encaminhamento.

- Centralizar os aspectos relevantes dos planos de gerenciamento, serviços e controle para simplificar o design da rede e reduzir os custos operacionais.
- Padronizar protocolos propiciando a interoperabilidade e o suporte heterogêneo de fornecedores com mais opções e redução de custos.
- Possibilidade de utilizar em conjunto com SDN a tecnologia Cloud Computing para flexibilização e escalabilidade da rede, melhorando a infraestrutura e agregando custo-benefício a organização.

Para estabelecer a comunicação entre o plano de controle e o plano de encaminhamento foi preciso criar e padronizar um protocolo, sendo então estabelecido o protocolo OpenFlow, um protocolo aberto que possibilita o desenvolvimento de mecanismos programáveis baseado em tabelas de fluxos em diferentes switches e roteadores, ele estabelece uma comunicação segura entre os switches e o controlador, o qual utiliza esse canal para monitorar e estabelecer fluxos conforme a inteligência estabelecida pelo software [9].

2.3 Protocolo OpenFlow

O OpenFlow é um arcabouço que permite o gerenciamento da tabela de encaminhamento de comutadores Ethernet, desacoplando a lógica de controle do equipamento do hardware de encaminhamento de pacotes, permitindo assim o conceito chamado de SDN [7].

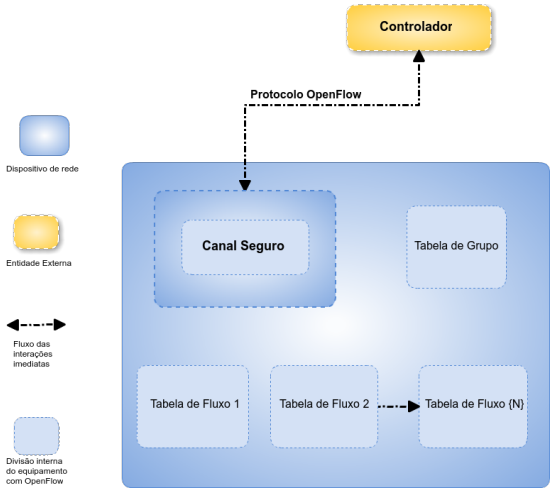

Figura 3: Arquitetura do switch OpenFlow

Uma das vantagens em se utilizar uma arquitetura OpenFlow é a flexibilidade que ela proporciona para se programar de forma independente o tratamento de cada fluxo da rede e como ele deve ou não ser encaminhado pela rede. De uma maneira mais prática o OpenFlow determina como um fluxo pode ser definido, quais serão as ações que podem ser realizadas para cada pacote pertencente a este fluxo e qual é o protocolo de comunicação

entre o controlador e os comutadores utilizados para realizar as definições de fluxo e ações. Dessa forma uma entrada na tabela de fluxos de um comutador OpenFlow é formada pela união da definição do fluxo e um conjunto de ações a ele determinadas [10].

Um fluxo é constituído pela definição dos valores de um ou mais campos do cabeçalho do pacote a ser processado pelo dispositivo OpenFlow. Dessa forma os campos do cabeçalho descrevem o fluxo que por sua vez descrevem quais pacotes combinam com aquele fluxo. Esses campos formam uma tupla de doze elementos que reúne características dos protocolos da camada de enlace, de rede, e de transporte segundo o modelo TCP/IP. Cada fluxo contém regras, para a definição dos pacotes, ações e contadores estatísticos a ele associados [10].

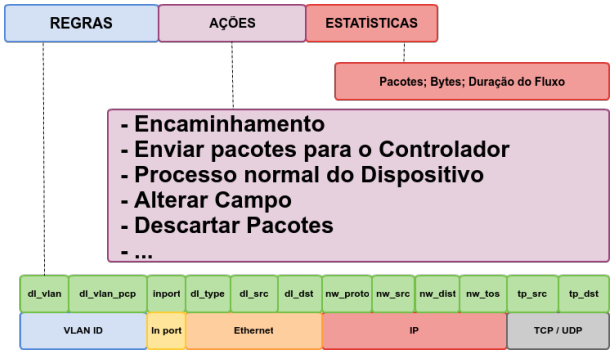

Figura 4: Definição de um fluxo na arquitetura OpenFlow

2.4 Controlador ONOS

O ONOS é o primeiro controlador para redes de código aberto escrito em Java, que visa especificamente o provimento de serviços de redes críticas. ONOS foi construído propositalmente para prover alta disponibilidade, escalabilidade e desempenho para demandas na rede. O ONOS disponibiliza na camada superior uma API, facilitando o desenvolvimento de aplicações de rede para o controlador e na sua camada inferior abstrai e disponibiliza interfaces que faz controle de dispositivos de rede com OpenFlow e dispositivos legado.

Colaboradores de Stanford, Berkeley e Nicira desenvolveram alguns controladores de redes durante os últimos sete anos, incluindo *NOX, Beacon, POX*. Esses controladores foram criados para demonstrar o potencial que tinha SDN. Muito foi aprendido com esses controladores além das aplicações desenvolvidas para os mesmos e a demonstração dessas em execução. Porém, é importante entender que esses controladores não foram desenvolvidos para fins comerciais e sim experimentais. Eles em sua maioria não tem as características necessárias como escalabilidade, alta disponibilidade (ou redundância) e alto desempenho. Assim o controlador ONOS foi criado para atender essas necessidades.

O ONOS foi desenvolvido para possibilitar os desenvolvedores de aplicações para redes conhecendo o funcionamento do hardware proprietário e permitir que os gerentes de redes se libertem da complexidade das interfaces e dos protocolos proprietários

A sua arquitetura foi definida desde o início mantendo o foco no provimento de serviços. E podemos destacar como principais funcionalidades:

- Núcleo distribuído e controle redundante, permitindo a alta disponibilidade do sistema em caso de algum controlador da rede ficar indisponível;
- Northbound é a camada superior do controlador que fornece uma API para desenvolvimento de aplicações diversas para serem executadas no controlador;
- Southbound é a camada baixa do controlador, a interface de comunicação com os dispositivos da rede, sendo responsável por abstrair o complexo gerenciamento tanto de dispositivos legado, quanto de dispositivos que executam o OpenFlow.
- A modularização é uma característica do ONOS que permite o fácil desenvolvimento, manutenção, depuração e atualização das aplicações que facilita o provimento de serviços entre os próprios módulos.

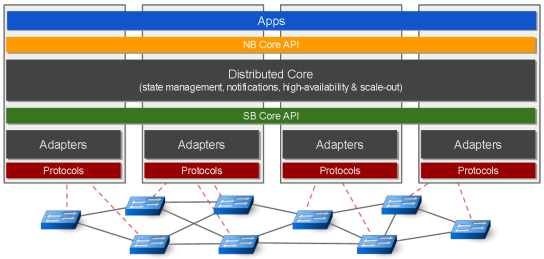

Figura 5: Arquitetura ONOS

Pelo fato do ONOS ser um controlador distribuído ele garante uma alta disponibilidade. A redundância do controle pode ser realizada em forma de cluster como é exibido na imagem abaixo.

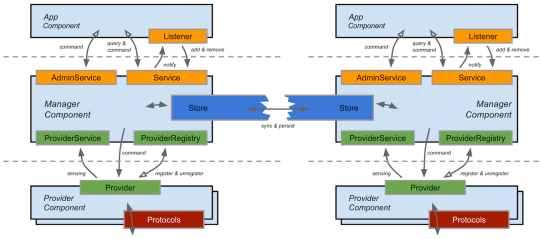

Figura 6: Cluster com ONOS

2.4.1 Módulos ONOS

A seguir, relações bem definidas dos módulos, base para personalização e dependências cíclicas.

Como podemos observar, o ONOS mantém um compartilhamento da base de dados em sincronia entre os controladores, permitindo assim que no caso de indisponibilidade de um

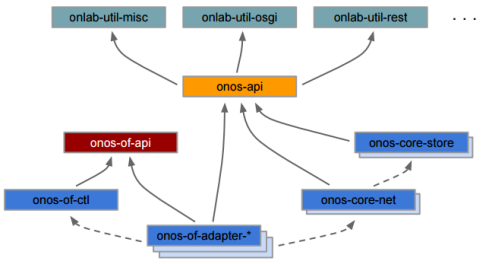

Figura 7: Módulo ONOs

controlador da rede os demais assumem a gerência dos dispositivos que se encontravam fora de sua gerência.

Assim o ONOS prova-se como uma excelente ferramenta do paradigma de Redes Definidas por Software, no sentido de solucionar problemas existentes em controladores mais antigos, trazendo uma proposta voltada ao mercado, com características importantes como alta disponibilidade, escalabilidade e alto desempenho.

2.5 Desafios de Segurança em SDN

As redes de computadores, tanto as tradicionais quanto as SDN, necessitam de mecanismos de segurança. Um dos mecanismos de segurança aplicado as redes de computadores é a autenticação de portas, a qual restringe o acesso não autorizado de dispositivos a uma rede local [11]. As estações finais que se conectam a uma rede de computadores nem sempre serão confiáveis, podendo apresentar inúmeras vulnerabilidades. O protocolo OpenFlow, apesar de largamente utilizado em SDN, não possui um mecanismo nativo para autenticação segura da origem. Por padrão, as estações finais se autenticam a uma rede OpenFlow através da validação de seus endereços MAC (Media Access Control) ou IP (Internet Protocol), com base na lógica de comutação de pacotes definida pelo administrador da topologia.

A centralização do plano de controle traz inúmeras vantagens, como programabilidade, lógica centralizada e visão global da rede [12]. Tais atributos representam significativos benefícios, porém aumentam a exposição do controlador e suas aplicações a ataques de negação de serviço(Ex.: DOS), e interceptação de fluxos. Em uma rede OpenFlow, todo pacote é analisado, caso o pacote possui cabeçalho não associado aos fluxos existentes, é enviado para inspeção do controlador. Caso algum comutador da topologia envie uma grande quantidade incomum de novos cabeçalhos de pacotes para o controlador, poderá ter seus recursos de processamento esgotados [13].

A necessidade de autenticação dos elementos de rede não é exclusividade de uma SDN, tão pouco um novo paradigma, redes convencionais sem separação do plano de dados e controle apresentam o mesmo tipo de problema [13]. A questão peculiar em uma SDN é o aumento da criticidade desse fato, pois estando um nó da rede comprometido,

alvos como o controlador e estação de gerenciamento podem ser alcançados, tornando a rede vulnerável [13]. Desta forma propõe-se uma aplicação nativa do controlador para realizar controle de acesso.

2.6 Sistema Operacional Embarcado OpenWRT

O OpenWRT é uma distribuição Linux para Sistemas Embarcados, em que seu desenvolvimento dispõe de um sistema de arquivos totalmente escrito com gerenciamento de pacotes, e não apenas um firmware simples com um único processo, isso proporciona uma grande facilidade na seleção e configuração das aplicações fornecidas e permite que o colaborador/usuário customize o dispositivo usando apenas pacote.

Para colaboradores o OpenWRT é a estrutura ideal para construir uma aplicação sem ter a preocupação de construir um firmware completo em torno dela. Para o usuário isto significa a habilidade de total customização, para utilizar o dispositivo de maneira nunca antes vista. Com a liberação dos códigos fontes Linux da série de roteadores *Linksys WRT54G/GS* surgiram grandes números de *firmwares* modificados para estender as funcionalidades em várias maneiras.

O projeto começou em janeiro 2004. As primeiras versões de OpenWrt foram baseadas em fontes de Linksys GPL para WRT54G e no buildroot do projeto uclibc. Esta versão foi definida como OpenWrt versão estável. Há ainda muitas aplicações de OpenWrt, como os *Freifunk-Firmware*, que são baseados nesta versão. No começo de 2005 alguns novos desenvolvedores se juntaram a equipe. Após alguns meses de desenvolvimento fechado a equipe decidiu publicar as primeiras versões experimentais do OpenWrt.

As versões experimentais usam sistemas bastante customizado, construída e baseada na versão 2 do *buildroot*, o *buildroot2*, também do projeto *uclibc*. OpenWrt usa fontes oficiais do kernel GNU/Linux e apenas adiciona patches para o sistema de chip e drivers de interface de rede. A equipe de desenvolvimento tenta re-implementar a maior parte do código proprietário no interior dos GPL dos diferentes fornecedores. Existem ferramentas livres para escrever novas imagens de *firmware*, para configurar o chip de LAN sem fios *(wlcompat / wificonf)* e programar o interruptor de VLAN.

2.7 Comutador Virtual Open vSwitch

Open Virtual Switch (Open vSwitch ou OVS) é um software multiplataforma de código aberto que emula switches virtuais permitindo que máquinas virtuais se comuniquem entre si ou encaminhar o tráfego para uma rede física.Também fornece funcionalidades avançadas como VLANs, tunelamento e suporte nativo da tecnologia OpenFlow, permitindo desta forma a criação de redes definidas por software (SDN).

Ser programável é uma característica fundamental para redes SDN, no entanto

os protocolos OpenFlow e OVSDB foram implementados no Open vSwitch para permitir a automatização de rede através de programas.

OVS pode ser controlado e programável através do protocolo de gerenciamento OVSDB e da interface padrão de comunicação OpenFlow. OVS foi desenvolvido para ser compatível com diversas plataformas e hardware, provê suporte a interfaces padrão de gerenciamento, como *sFlow, NetFlow* entre outros (Open vSwitch, 2014a). Os principais componentes da arquitetura do OVS são:

- *ovs-vswitchd*: a execução de um programa em background (daemon) que implementa o switch, e em conjunto com o módulo do kernel Linux faz a comutação baseada em fluxo.
- *ovsdb-server*: um servidor de banco de dados para armazenar informações de configuração do ovs-vswitchd.
- *ovs-dpctl*: uma ferramenta para configurar o módulo de comutação do kernel.
- *ovs-vsctl*: uma ferramenta para requisitar e atualizar a configuração do ovsvswitchd.
- *ovs-appctl*: uma aplicativo que envia comandos para daemons do Open vSwitch.
- *ovs-ofctl*: uma ferramenta que consulta, controla tabela de fluxo dos switches e controladores OpenFlow.

CAPÍTULO 3

Trabalhos Relacionados

Este capítulo apresenta os principais trabalhos encontrados na literatura relacionado a aplicação de controle de acesso para redes definidas por software e outras soluções que garantem a segurança de uma infraestrutura SDN. Os trabalhos relacionados listados estão associados a um ou mais aspetos considerados neste trabalho.

3.1 Ethane

Uma arquitetura que visa a autenticação de nós em uma rede definida por software, também apresenta uma política de autenticação baseado em fluxos. Ethane permite que os gerentes definam uma única rede "fine-grain", nela o controlador é quem realiza o processo de autenticação dos dispositivos finais da rede. A fim de proteger a rede e a informação que nela viaja Ethane estipula três princípios fundamentais para tornar a rede mais facilmente administrável.

- A rede deve ser gerida tendo em conta políticas de segurança estabelecidas a um nível mais elevado e a partir de pontos de acesso que não envolvam os equipamentos ativos de rede;
- Devem também ser as políticas a determinar todo o caminho que os pacotes percorrem ao longo da rede, ainda que forçosamente sejam obrigados a passar por um destino intermédio específico;
- A rede deve impor uma forte ligação entre os pacotes e a sua origem, de forma a poderem ser sempre rastreados.

3.2 Resonance

Um sistema de segurança para controle de acesso dinâmico em redes empresariais. Com esse mecanismo controlador não participa no processo de autenticação. Resonance utiliza Web Service para fazer autenticação e autorização dos hosts. A segurança da rede corporativa é tipicamente reativa, e depende muito da segurança dos hosts e "middleboxes" caixas de meio. Eles propuzeram Resonance, um sistema para proteger redes corporativas, onde os próprios elementos de rede aplicam políticas de controle de acesso dinâmicas baseadas em informações de nível de fluxo e alertas em tempo real [14].

A ressonância usa switches programáveis para manipular o tráfego em camadas inferiores; Os comutadores tomam decisões (ex:, descartando ou redirecionando tráfego) para impor políticas de segurança de alto nível com base em informações de políticas de segurança de nível mais alto e de sistemas distribuídos de monitoramento e inferência.

3.3 AuthFlow

Um mecanismo de autenticação e controle de estações finais baseado na credencial da estação. O AuthFlow apresenta duas contribuições principais: (i) a autenticação das estações finais diretamente na camada de enlace; e (ii) a associação das credenciais de acesso de uma estação aos fluxos pertencentes a essa estação. Segundo [Mattos et al. 2014] a ideia principal do mecanismo AuthFlow é realizar a autenticação usando protocolos da camada de enlace, fazendo o mapeamento da identidade usada na autenticação em fluxos criados por uma dada estação autenticada. Para tanto, o mecanismo proposto usa o padrão IEEE 802.1X e o Extensible Authentication Protocol (EAP). O EAP encapsula as trocas de mensagens de autenticação entre a estação suplicante e um servidor de autenticação RADIUS [15].

CAPÍTULO 4

Proposta

Este capítulo apresenta funcionamento da proposta: Uma Aplicação de Mapeamento Proativo dos Hosts em Redes Definidas por Softwares, seu contexto, visão geral e sua aplicação no cenário de mapeamento dos hosts.

4.1 Visão Geral

Mapeamento Proativo dos Hosts em Redes Definidas por Softwares é uma das plicações do controlador ONOS, escrito em Java para realizar mapeamento dos hosts e roteadores em uma rede.

Uma aplicação para Redes Definidas por Softwares que realiza mapeamento das informações dos hosts de usuários na rede, possibilita a aplicação de regras necessárias para manter o funcionamento da rede de acordo com as políticas definidas para este ambiente. Pode-se realçar o fato de ter como base SDN para melhor segurança e baixo investimento.

A aplicação para redes definidas por softwares tem o propósito de salvar no banco de dados SQL todas as informações de um determinado host que acessar na rede. Para isso o mecanismo de mapeamento funciona da seguinte maneira. O usuário fornece o *MAC* do host para o Administrador adicionar na tabela de fluxo de *OpenFlow*, nesse primeiro momento o Administrador faz autorização manualmente dos hosts para acesso a internet, próximo passo será implementado *RESTful e Web*, assim o usuário poderá fornecer seus dados e *MAC* pelo *Browser*.

Assim que Administrador adicionar *MAC* do host suplicante na tabela de fluxo OpenFlow, a aplicação de mapeamento captura *id_host, id_onos, mac, ip_idress, vlan e location*. Salva no banco de dados criado no momento que primeiro device e host é

adicionado ao controlador ONOS, caso for primeira vez na rede, se não compara com os dados existente e realiza atualização da tabela, para não permitir redundância. Após o estágio de autorização, o Controlar ONOS permite que o host suplicante acesse os recursos da rede. Assim aplicação de Mapeamento Proativo dos Hosts em Redes Definidas por Softwares persiste os dados capiturado.

Usou-se Apache Karaf como container para o controlador ONOS(OSGi). A tecnologia OSGi é um conjunto de especificações que definem um sistema de componente dinâmico para Java. Estas especificações permitem um modelo de desenvolvimento em que os aplicativos são (dinamicamente) composta de muitos componentes diferentes (reutilizáveis).

4.2 Arquitetura e Implantação

Para definir arquitetura da aplicação de mapeamento em redes definidas por softwares usou-se modelo MVC *(Model-View-Controller)*, este modelo quebra aplicação ou até mesmo um pedaço da interface de uma aplicação, em três partes: o modelo, a visão e o controlador.

O controlador (Controller) que interpreta as entradas do mouse ou do teclado enviado pelo usuário e mapeia essas ações do usuário em comandos que são enviados para o modelo (Model) e/ou para a janela de visualização (View) para efetuar a alteração apropriada. O modelo (Model) gerencia um ou mais elementos de dados, responde a perguntas sobre o seu estado e responde a instruções para mudar de estado. O modelo sabe o que o aplicativo quer fazer e é a principal estrutura computacional da arquitetura, pois é ele quem modela o problema que está se tentando resolver. A visão (View) gerencia a área retangular do display e é responsável por apresentar as informações para o usuário através de uma combinação de gráficos e textos.

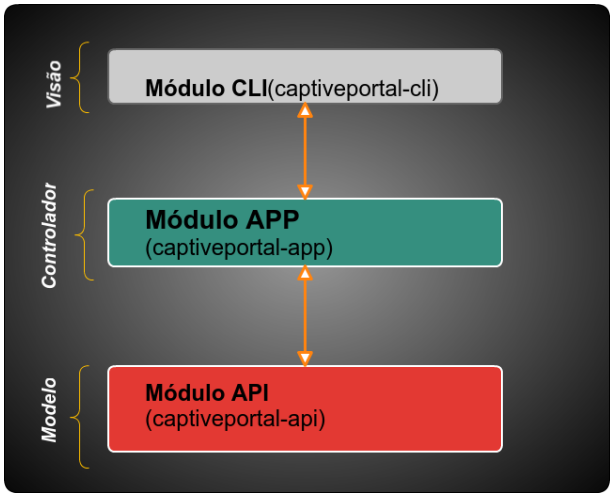

Figura 8: Arquitetura da Aplicação de Mapeamento Proativo dos Hosts em Redes Definidas por Softwares e seus Componentes.

A figura abaixo apresenta módulos e arquitetura da Aplicação de Mapeamento Proativo dos Hosts em Redes Definidas por Softwares.

1. Módulo CLI: responsável por permitir ao administrador do sistema acessar algumas funções da Aplicação de Mapeamento Proativo dos Hosts em Redes Definidas por Softwares por meio de linha de comando, a partir do modo CLI de controlador ONOS.
2. Módulo APP: principal responsável pelo controle de acesso à API.
3. Módulo API: responsável por classes da entidade do sistema, API de conexão com o banco de dados e lógica de aplicação das regras de negócio definidas pelo administrador.

4.3 Diagrama de Sequência

O diagrama de sequência permite identificar de maneira simples e concisa as sequências de atividades realizadas.

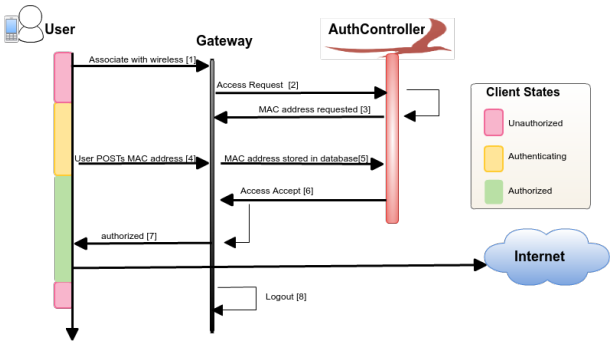

Figura 9: Diagrama de Sequência

Descrição do diagrama de sequência:

1. Usuário tenta associar na rede;
2. Device enviar requisição para Aplicação de Mapeamento Proativo dos Hosts em Redes Definidas por Softwares;
3. Aplicação de Mapeamento Proativo dos Hosts em Redes Definidas por Softwares solicita Mac do host para autenticação;
4. Usuário posta MAC do host para liberação de acesso;
5. Administrador adiciona MAC do host solicitante;
6. Acesso ao serviço liberado;
7. Usuário Autorizado;
8. Host logout.

4.4 Diagrama de Classes

O diagrama de classe apresenta as diferentes classes que fazem parte do sistema da aplicação de Mapeamento Proativo dos Hosts em Redes Definidas por Softwares e como elas se relacionam.

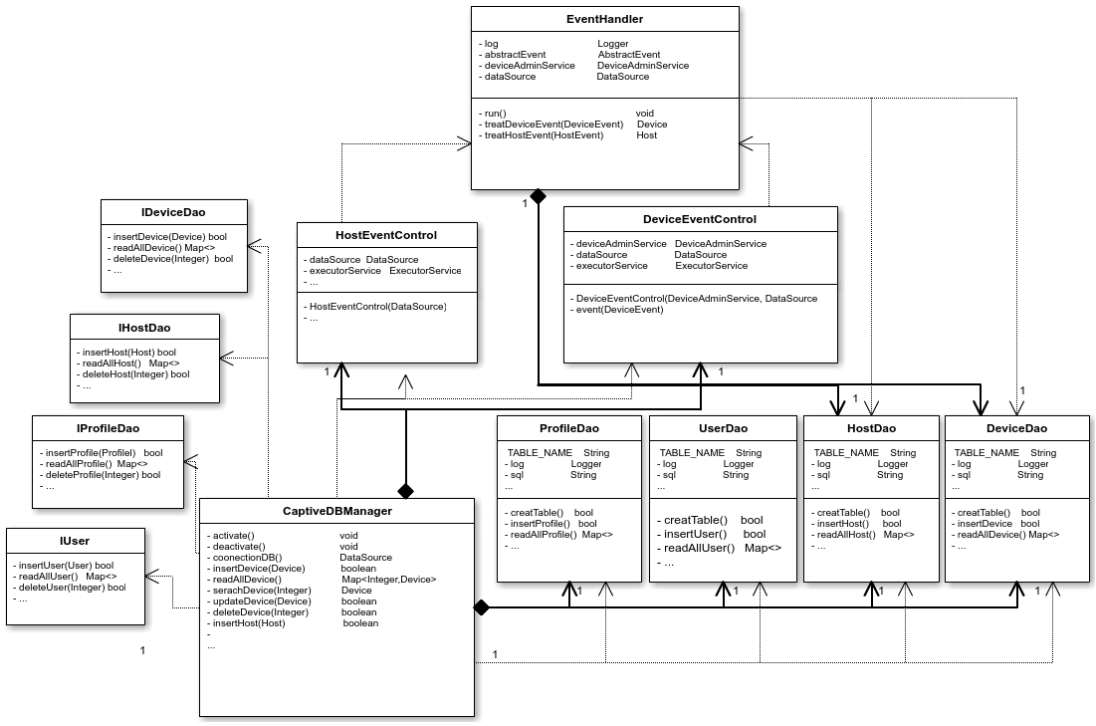

Figura 10: Diagrama de Classes

O diagrama mostrado na figura acima, apresenta CaptiveDBManager como classe principal. Nele faz-se chamada das interfaces *IDeviceDao e IHostDao* que acessam diretamente *DeviceDao e HostDao* para comunicação com banco de dados. As interfaces *IDeviceDao e IHostDao* são extremamente importantes para segurança dos dados, evitam acesso direto a classes que manipulam conexão com banco de dados. *DeviceEventControl e HostEventControl* capturam qualquer evento do controlador ONOS para controle do banco de dados. E também atravez dessas classes que aplicação sabe que chegou algum dado para ser adicionado ou atualizar.

CAPÍTULO 5

Avaliação da Proposta

Um ambiente de teste foi montado para testar Aplicação de Mapeamento Proativo dos Hosts em Redes Definidas por Softwares. O ambiente de teste depende muito do objetivo a ser alcançado,neste caso o objeto é desenvolver uma Aplicação de Mapeamento Proativo dos Hosts em Redes Definidas por Softwares.

5.1 Caso de Uso

Esse diagrama classe documenta o que o sistema faz do ponto de vista do usuário e administrador. Em outras palavras, ele descreve as principais funcionalidades do sistema e a interação dessas funcionalidades com os usuários.

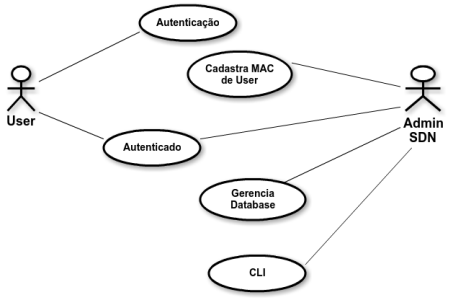

Figura 11: Diagrama de Caso de Uso

Usuário solicita autenticação para usar os recursos da rede, administrador SDN por sua vez adiciona MAC do host na tabela de fluxo OpenFlow, acesso ao serviço permitido, dados do host salvo no banco de dados.

5.2 Descrição da simulação

O ambiente de teste é mostrado na figura a baixo, pode-se observar a distribuição dos APs OpenFlow, host e controlador.

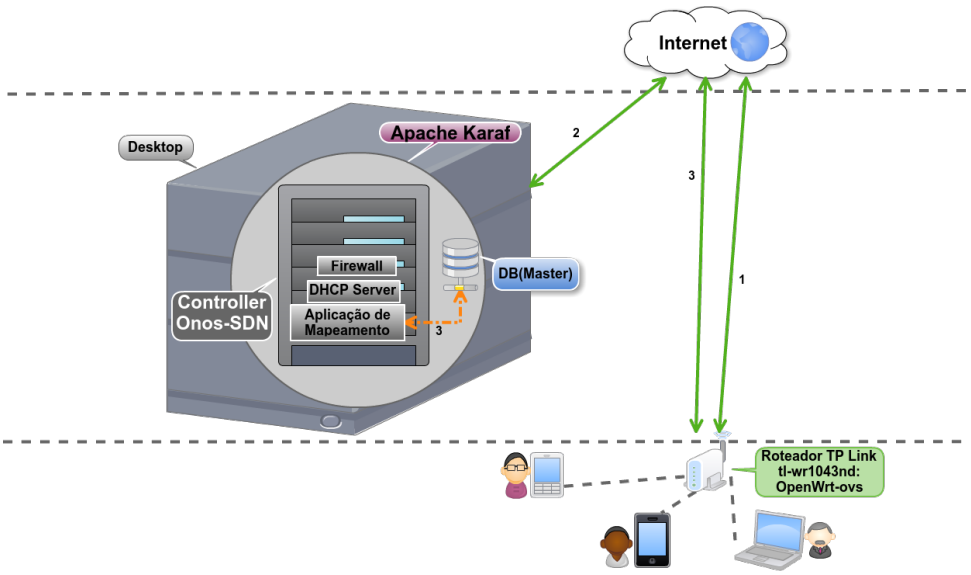

Figura 12: Ilustração do cenário de experimento da Aplicação de Mapeamento Proativo dos Hosts em Redes Definidas por Softwares.

O protótipo da Aplicação foi implementado no cenário acima, que possui um desktop, onde funciona o controlador ONOS e suas aplicações. Dentre aplicações do controlador ONOS destaca-se Firewall, DHCP Server e Aplicação de Mapeamento Proativo dos Hosts em Redes. Dentro Apache Karaf enconta-se banco de dados SQLite. O roteadore está configurado com OpenWrt e *Open vSwitch*.

O ambiente experimental é composto de uma rede sem fio com um roteador OpenFlow e equipamentos que possuem uma interface de redes sem fio, como computares portáteis e smartphones.

As configuração do plano de dados foi realizada de forma manual, com comandos *ovs-vsctl e ovs-ofctl*. Para setar as configurações no roteador *TP Link*, precisou-se conectar em uma interface cabeada de uma das portas do roteador *TP Link*, previamente confirado com um determinado endereço IP, via utilitário ssh *(Secure Shell)*. Uma vez que se esteja logado no roteador, pode-se definir as regras *ovs-vsctl* e *ovs-ofctl* utilizando o plano de controle.

ovs-ofctl add-flow br-wlan0 in_port=5,dl_src=80:6c:1b:f8:71:0b,action=output:1

ovs-ofctl add-flow br-wlan0 in_port=1,dl_dst=80:6c:1b:f8:71:0b,action=output:5

ovs-vsctl set-controller br-wlan0 tcp:10.126.1.247:6633

ovs-vsctl add-port br-wlan0 wlan0 – set Interface wlan0 ofport_request=5

Deste modo, nos campos *dl_src* e *dl_dst* pode-se definir em que host a regra será aplicada. Também no campo tcp pode-se definir que máquina será o controlador. No ambiente de teste cada interface da rede tem um endereço IP específico configurado.

5.3 Analise dos Resultados

As figuras a seguir mostram os resultados da Aplicação de Mapeamento Proativo dos Hosts em Redes Definidas por Softwares. E alguns comandos CLI para banco de dados SQLite. Repara-se nas interfaces a baixo, o Controlador ONOS e as informações dos host e Device: *MAC*, *IP* e *VLAN*.

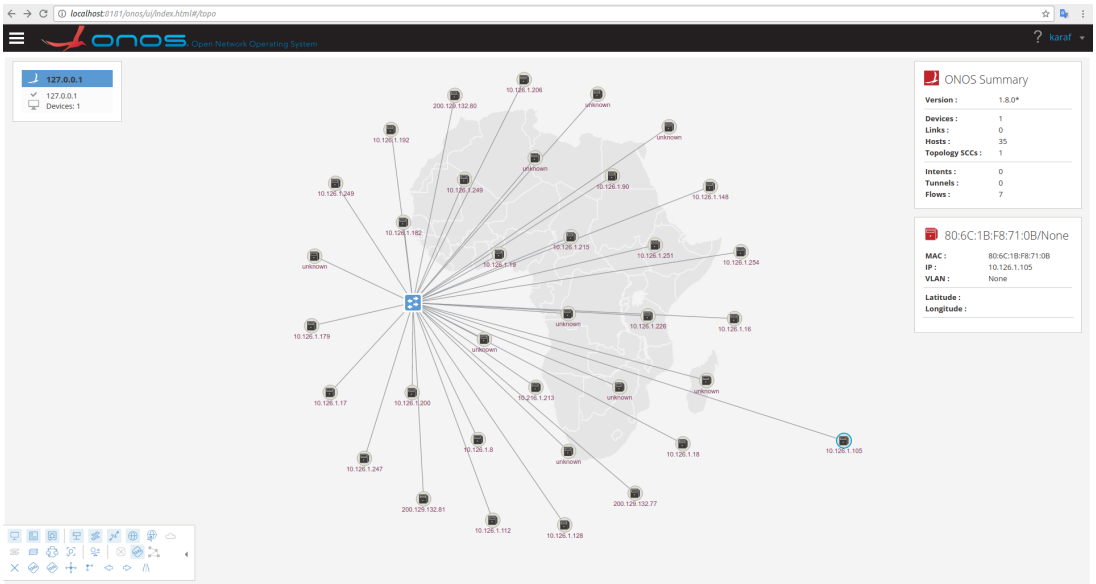

Figura 13: Interface de Controlador ONOS, destacando host com IP"10.126.1.105"

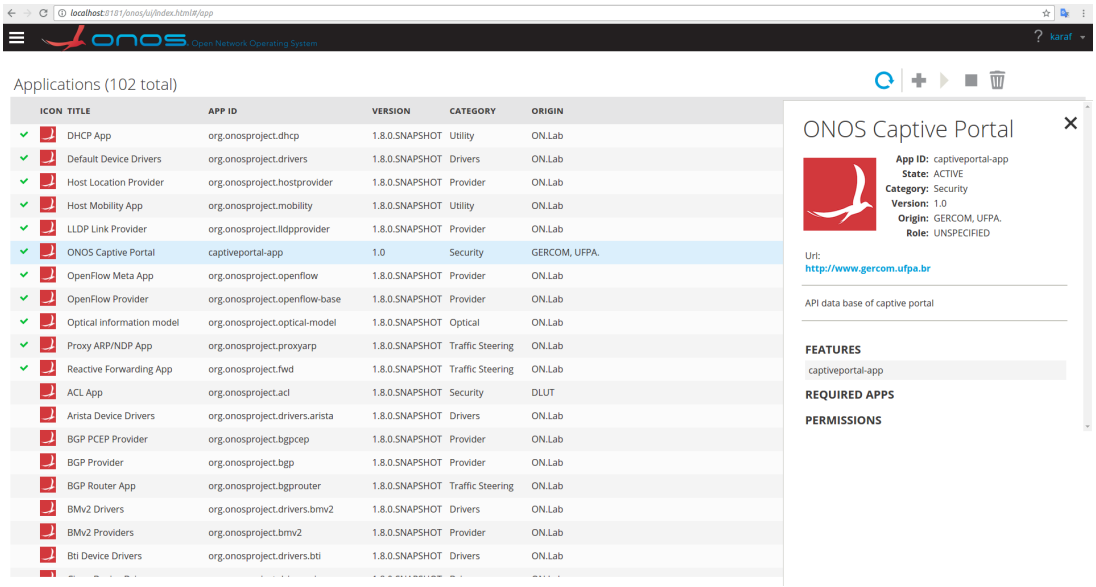

Figura 14: Aplicação de Mapeamento Proativo dos Hosts em Redes Definidas por Softwares com nome de ONOS Captive Portal, justamente como pretente-se usar as informações e dados dos hosts armazenados futuramente.

Figura 15: Informações do roteador controlado pelo ONOS. Alguns dados como: *ID*, *IP*, Número de portal e *Open vSwitch*.

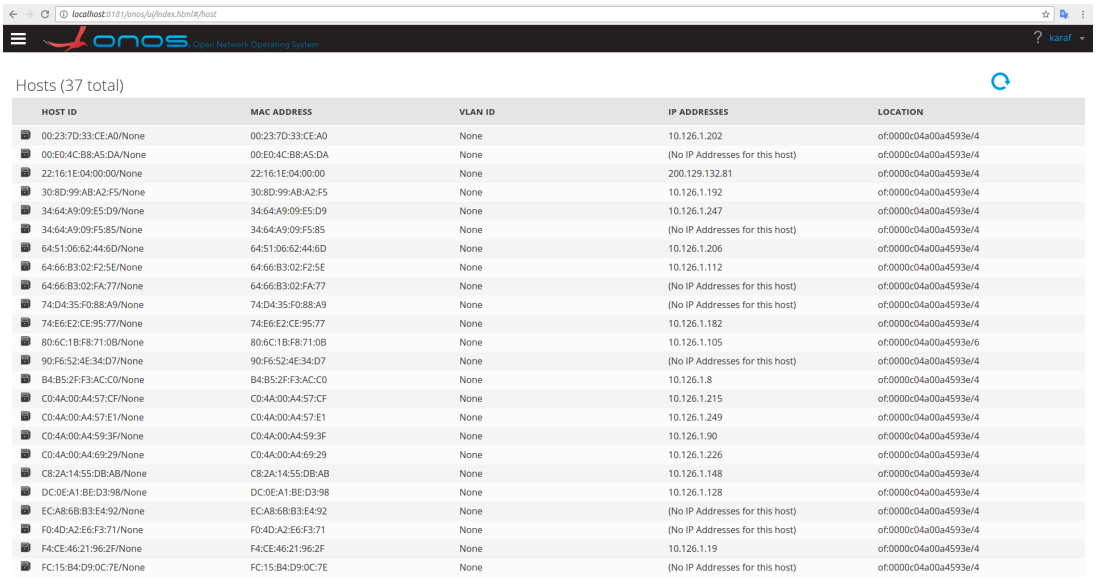

Figura 16: Quantidades e informações dos hosts conectados no Controlador, o mesmo se encontra no banco de dados.

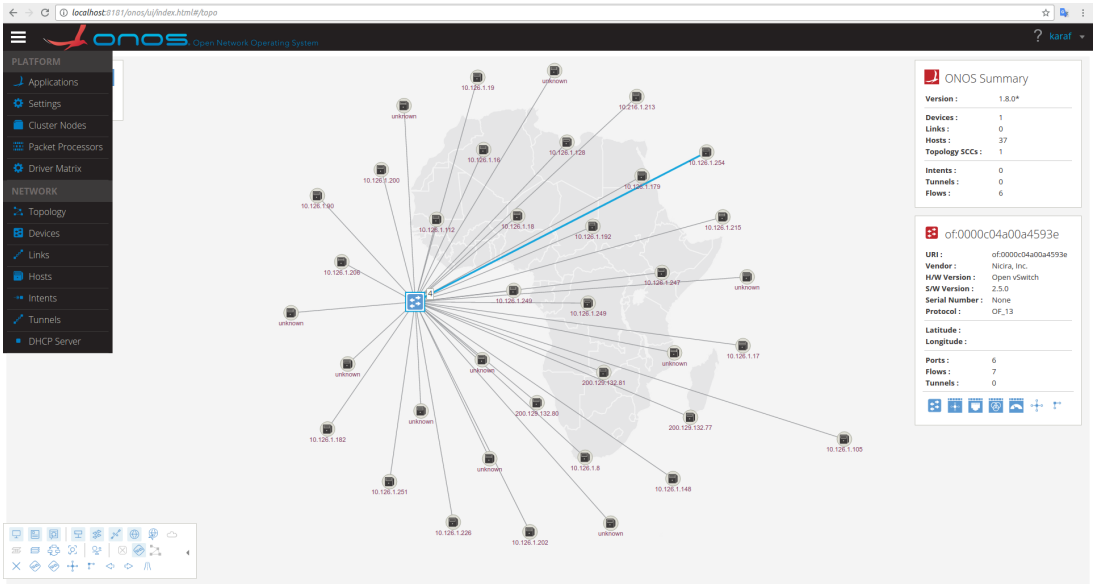

Figura 17: A esquerda em cima da interface, as ferramentas que o Controlador ONOS desponibiliza para gerenciamento e monitoramentos.

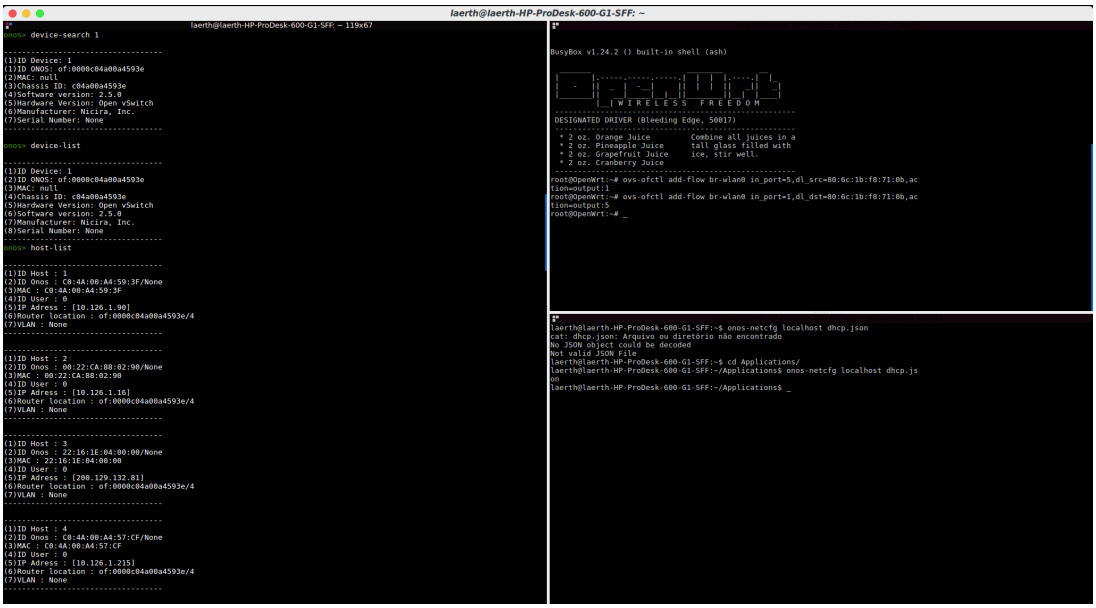

Figura 18: Terminal com alguns comandos *CLI*(*device-list e host-list*) e *ovs-ofctl*(adiciona host na tabela de Fluxo
textitOpenFlow)

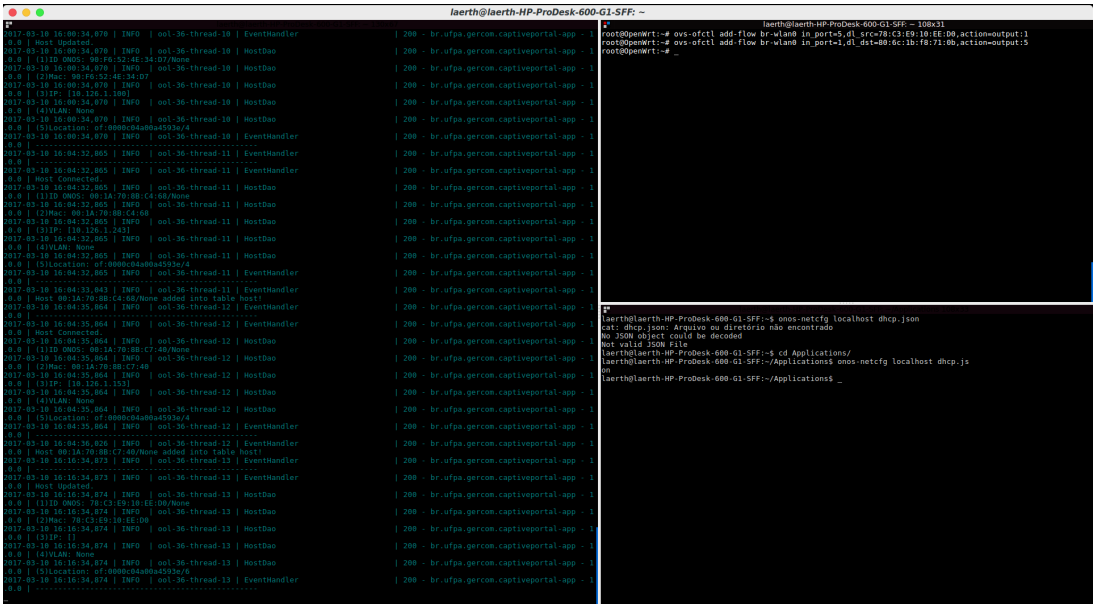

Figura 19: No lado esquerdo do terminal o log do controlador ONOS, exibindo eventos dos hosts conectado(Comando: *log:tail*)

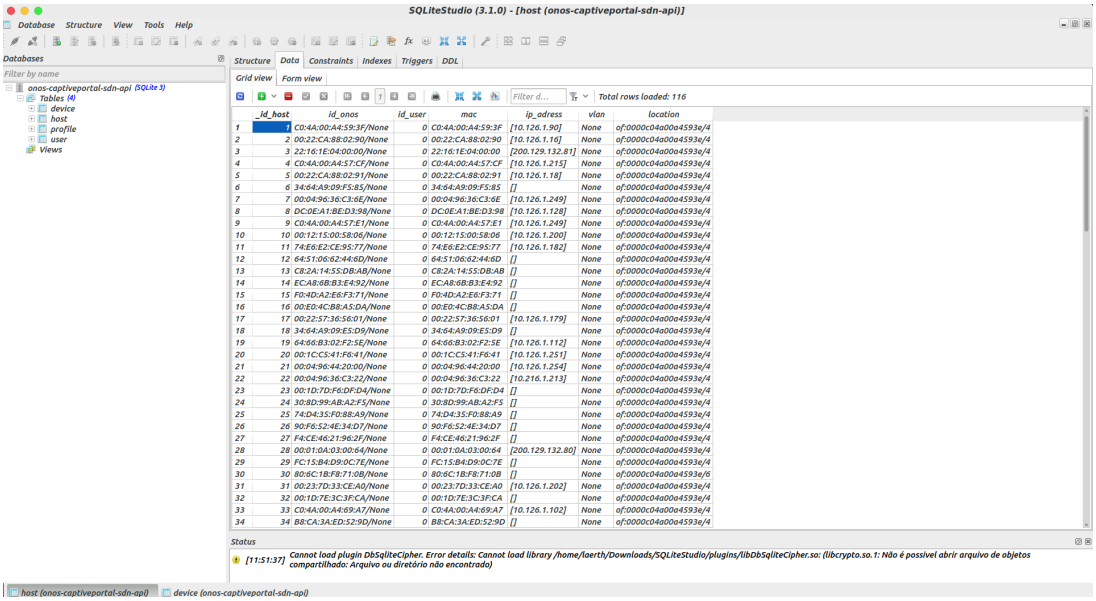

Figura 20: Banco de Dados com Informações dos hosts

CAPÍTULO 6

Conclusões

Este trabalho desenvolveu uma aplicação para Mapeamento Proativo dos Hosts em Redes Definidas por Softwares e foi montado um cenário de teste, onde foi testado e avaliado devidamente. O objetivo definido foi alcançado, aplicação funciona.

Com aplicação funcionando pode-se ver as etapas mais importantes do desenvolvimento funcionando, apache-Karaf como container para ONOS e aplicação de mapeamento proativo dos Hosts como uma aplicação do ONOS. A Comunicação entre controlador e roteador usando protocolo OpenFlow, onde pude manipular as tabelas de Fluxos do OpenFlow. Resumidamente os resultados foram satisfatório, questão como persistência dos dados dos hosts resolvido.

Como trabalho futuro, pretende-se implementar uma coluna na tabela de banco de dados, a fim de contar quantas vazes um determinado host acessou a rede. E também a parte da aplicação, que realiza controle de acesso e monitoramento, incluíndo módulo RESTful e Web. Uma vez que ele estiver funcionando, junto com protocolo de autenticação seguro, proposto pelo Leonardo da Costa (aluno de mestrado no GERCOM) a aplicação poderá inserie regras, necessárias para manter o funcionamento da rede de acordo com as políticas definidas para este ambiente.

Referências

[1] J. F. Kurose, K. W. Ross, C. M. Hierro, Á. P. d. M. y Pablo, and L. Marrone, *Redes de computadoras: un enfoque descendente*. Addison Wesley, 2010.

[2] F. N. Farias, J. M. D. Júnior, J. J. Salvatti, S. Silva, A. J. Abelém, M. R. Salvador, and M. A. Stanton, "Pesquisa experimental para a internet do futuro: Uma proposta utilizando virtualização e o frame-work openflow," *XXIX Simpósio de Redes de Computadores e Sistemas Distribuídos-SBRC*, 2011.

[3] M. Moraes, B. Pinheiro, V. Nascimento, and A. Abelém, "Redes sem fio de multiplos saltos definidas por software," in *IV Workshop de Pesquisa Experimental da Internet do Futuro*, 2013.

[4] M. D. Moreira, N. C. Fernandes, L. Costa, and O. Duarte, "Internet do futuro: Um novo horizonte," *Minicursos do Simpósio Brasileiro de Redes de Computadores-SBRC*, vol. 2009, pp. 1–59, 2009.

[5] D. Clark, R. Braden, K. Sollins, J. Wroclawski, and D. Katabi, "New arch: Future generation internet architecture," DTIC Document, Tech. Rep., 2004.

[6] B. A. Forouzan and S. C. Fegan, *Protocolo TCP/IP-3*. AMGH Editora, 2009.

[7] C. E. Rothenberg, M. R. Nascimento, M. R. Salvador, and M. F. Magalhães, "Openflow e redes definidas por software: um novo paradigma de controle e inovação em redes de pacotes," *Cad. CPqD Tecnologia, Campinas*, vol. 7, no. 1, pp. 65–76, 2010.

[8] G. Marchesan and R. D. Medina, "Simulando cenários para redes definidas por software," *Simpósio de Pesquisa e Desenvolvimento em Computação*, vol. 1, no. 1, 2016.

[9] N. McKeown, T. Anderson, H. Balakrishnan, G. Parulkar, L. Peterson, J. Rexford, S. Shenker, and J. Turner, "Openflow: enabling innovation in campus networks," *ACM SIGCOMM Computer Communication Review*, vol. 38, no. 2, pp. 69–74, 2008.

[10] L. R. Costa, "Openflow e o paradigma de redes definidas por software," 2013.

[11] L. G. B. UEPG and D. C. F. J. UEPG, "Autenticação ieee 802.1 x em redes de computadores utilizando tls e eap," 2008.

[12] T. D. Nadeau and K. Gray, *SDN: Software Defined Networks: An Authoritative Review of Network Programmability Technologies.* "O'Reilly Media, Inc.", 2013.

[13] D. Kreutz, F. Ramos, and P. Verissimo, "Towards secure and dependable software-defined networks," in *Proceedings of the second ACM SIGCOMM workshop on Hot topics in software defined networking.* ACM, 2013, pp. 55–60.

[14] A. K. Nayak, A. Reimers, N. Feamster, and R. Clark, "Resonance: dynamic access control for enterprise networks," in *Proceedings of the 1st ACM workshop on Research on enterprise networking.* ACM, 2009, pp. 11–18.

[15] D. M. F. Mattos, O. C. M. B. Duarte, and R. de Janeiro-RJ-Brasil, "Authflow: Um mecanismo de autenticaç ao e controle de acesso para redes definidas por software," 2014.

ANEXO A – Configurações e Interfaces de OpenWRT

ovs-vsctl show
489fd0fa-f00a-495b-bf55-404f8f91b671
Bridge "br-wlan0"
Controller "tcp:10.126.1.224:6633"
is_connected: true
Controller "tcp:10.126.1.247:6633"
is_connected: true
Port "eth1.4"
Interface "eth1.4"
Port "wlan0"
Interface "wlan0"
Port "br-wlan0"
Interface "br-wlan0"
type: internal
Port "eth1.6"
Interface "eth1.6"
Port "eth1.5"
Interface "eth1.5"
Port "eth1.3"
Interface "eth1.3"

ANEXO B – Configuração de Bridge e Portas

ovs-ofctl show br-wlan0
1(eth1.4): addr:c0:4a:00:a4:59:3e
config: 0
state: 0
speed: 0 Mbps now, 0 Mbps max
2(eth1.5): addr:c0:4a:00:a4:59:3e
config: 0
state: 0
speed: 0 Mbps now, 0 Mbps max
3(eth1.3): addr:c0:4a:00:a4:59:3e
config: 0
state: 0
speed: 0 Mbps now, 0 Mbps max
4(eth1.6): addr:c0:4a:00:a4:59:3e
config: 0
state: 0
speed: 0 Mbps now, 0 Mbps max
5(wlan0): addr:c0:4a:00:a4:59:3e
config: 0
state: 0
speed: 0 Mbps now, 0 Mbps max
LOCAL(br-wlan0): addr:c0:4a:00:a4:59:3e
config: POR_DOWN
state: LINK_DOWN
speed: 0 Mbps now, 0 Mbps max
OFPT_GET_CONFIG_REPLY (xid=0x4): frags=normal miss_send_len=0