Firewall, Teleinformática e Redes 2: Projeto 2 - 2023/1

Aquila Macedo Costa - 202021800
Igor Laranja Borges Taquary - 180122231
Matheus Lopes de Souza - 190043831

I. INTRODUÇÃO

O avanço das tecnologias de redes tem transformado a forma como nos comunicamos e interagimos em nosso mundo digital. Com o crescente uso de dispositivos móveis e a virtualização de servidores, tornou-se fundamental encontrar soluções que permitam uma gestão mais flexível, segura e automatizada das redes modernas.

Neste contexto, o Software Defined Networking (SDN) surge como uma abordagem promissora, oferecendo maneiras inovadoras de lidar com os desafios enfrentados pelas redes atuais. Através do SDN, é possível criar redes mais dinâmicas e adaptáveis, permitindo uma melhor alocação de recursos, garantindo a qualidade de serviço e fortalecendo a segurança das informações.

O presente projeto tem como objetivo explorar os benefícios do SDN, utilizando o simulador Mininet, habilitado para OpenFlow, para construir uma rede virtual e aplicar um conjunto de estratégias de segurança. Através deste relatório, apresentaremos as etapas realizadas no projeto, os resultados experimentais obtidos e o impacto das soluções implementadas.

II. FUNDAMENTAÇÃO TEÓRICA

A. SDN (Software Defined Networking)

O SDN, é uma abordagem de arquitetura de rede que separa o plano de controle (control plane) do plano de dados (data plane) em dispositivos de rede, como switches e roteadores. Isso significa que o controle da rede é centralizado em um controlador SDN, enquanto os dispositivos de rede (como switches e roteadores) executam funções de encaminhamento e comutação do plano de dados. Essa separação permite que o controlador SDN tome decisões inteligentes sobre como os pacotes de dados são roteados e processados, com base em políticas e regras definidas pelo administrador da rede.

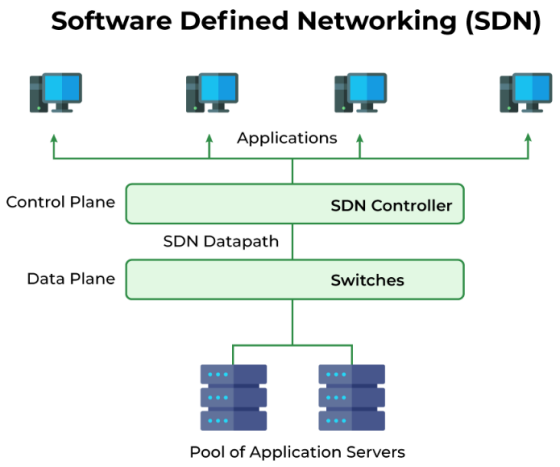

Fig. 1. Imagem simplificada sobre SDN

B. Mininet e POX

O Mininet é uma plataforma de emulação de rede de código aberto que permite criar redes virtuais em um único computador. É amplamente utilizado em pesquisas de redes de computadores, oferecendo a capacidade de testar novos protocolos e algoritmos sem a necessidade de uma rede real. Usando recursos de virtualização, o Mininet cria hosts e switches virtuais, possibilitando a análise detalhada do tráfego de rede.

Geralmente, o Mininet é usado em conjunto com controladores SDN, como o OpenFlow, para permitir um controle mais flexível sobre o comportamento da rede emulação.

O POX é um controlador de rede de código aberto desenvolvido em Python, projetado para trabalhar com redes definidas por software (SDN - Software-Defined Networking). Ele fornece uma estrutura flexível e extensível para desenvolver controladores SDN personalizados, permitindo que os desenvolvedores implementem e testem algoritmos e protocolos de rede em ambientes controlados.

III. AMBIENTE EXPERIMENTAL E ANÁLISE DE RESULTADOS

A. Topologia

Na implementação da topologia, seguimos as recomendações da documentação do próprio Mininet sobre topologias customizadas. Foram utilizados 10 hosts, 4 switches e um controlador remoto (POX). Os hosts representam os dispositivos finais, como computadores ou outros dispositivos de rede, que estão conectados aos switches. Os switches são dispositivos intermediários responsáveis por encaminhar os pacotes de dados entre os hosts conectados a eles. O controlador remoto é uma entidade externa ao Mininet que gerencia a comunicação entre os switches.

O controlador remoto é uma parte fundamental da rede definida por software (SDN - Software-Defined Networking). Ele atua como uma entidade centralizada que gerencia e controla o comportamento dos switches na rede. A comunicação entre os switches e o controlador é estabelecida através do protocolo OpenFlow, que permite que o controlador instrua os switches sobre como encaminhar o tráfego.

B. Firewall

Um Firewall é um sistema que bloqueia e filtra o tráfego que se direciona a ele, de acordo com um conjunto de regras. Ele pode ser utilizado para proteger uma rede contra ameaças provenientes da internet. No caso de um Firewall baseado em SDN (Rede Definida por Software), utilizamos o controlador OpenFlow para gerenciar o tráfego entre os dispositivos conectados, aplicando as regras definidas para permitir ou bloquear o fluxo de dados. Nesse contexto, utilizaremos o controlador POX para estabelecer as políticas ou regras necessárias (abordaremos esse aspecto mais

adiante) e filtrar o tráfego entre os hosts utilizando os switches. A figura abaixo ilustra um exemplo de um Firewall SDN com dois hosts:

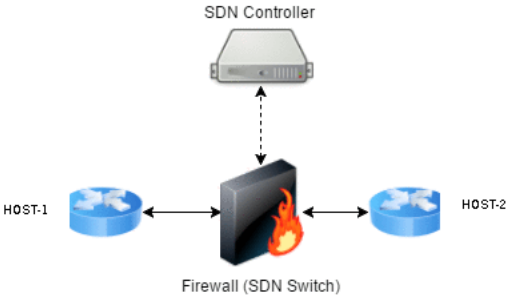

Fig. 2. Firewall SDN com dois hosts

As regras são definidas em um arquivo JSON chamado *firewall_rules_config.json* por padrão. Cada regra consiste em um conjunto de condições que devem ser atendidas para que o tráfego seja bloqueado.

```
{
    "rules": [
        {
            "tcp": {
                "dst": "21"
            }
        },
        {
            "eth": {
                "src": "00:00:00:00:00:01",
                "dst": "00:00:00:00:00:02"
            }
        },
        {
            "eth": {
                "src": "00:00:00:00:00:02",
                "dst": "00:00:00:00:00:01"
            }
        },
        {
            "eth": {
                "src": "00:00:00:00:00:04",
                "dst": "00:00:00:00:00:06"
            }
        },
        {
            "eth": {
                "src": "00:00:00:00:00:06",
                "dst": "00:00:00:00:00:04"
            }
        }
    ]
}
```

Fig. 3. firewall_rules_config.json

Ele suporta dois tipos de regras: regras baseadas em endereços Ethernet (*ETH_RULE)* e regras baseadas em pacotes TCP (*TCP_RULE*). Isso permite filtrar o tráfego com base nos endereços MAC de origem e destino e nos números de porta TCP de origem e destino.

Foram criadas duas funções de adição de regras, *add_eth_rule()* e *add_tcp_rule()*, que são responsáveis por adicionar as regras específicas de endereço Ethernet e TCP, respectivamente. As regras são adicionadas ao objeto *block*, que é posteriormente utilizado para criar o fluxo de bloqueio no switch.

O controlador POX monitora eventos de conexão e chama o método *_handle_ConnectionUp()* sempre que um novo switch se conecta à rede. Esse método verifica se o switch conectado é o mesmo definido como *router_id* na função *launch()* e, em caso afirmativo, adiciona as regras do firewall a esse switch.

C. Análise de tráfego

A análise do tráfego entre os hosts foi realizada por meio da aplicação Wireshark, uma ferramenta que possibilita a observação e monitoramento das mensagens trocadas entre diferentes dispositivos em uma rede. Com o auxílio do Wireshark, tornou-se viável examinar em detalhes o tráfego de dados, identificando os pacotes transmitidos, bem como analisando os protocolos e conteúdos das mensagens.

No projeto foram feitos testes utilizando dois protocolos de aplicação, o HTTP (Hypertext Transfer Protocol) e o FTP (File Transfer Protocol). Em ambos os casos o protocolo da camada de transporte foi o TCP.

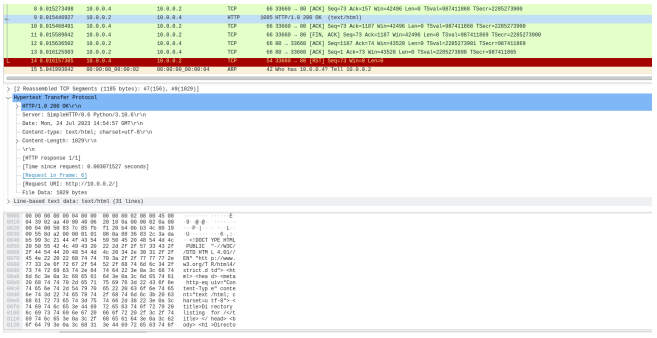

Fig. 4. Detalhes da mensagem HTTP na camada de aplicação

No protocolo HTTP (Hypertext Transfer Protocol), uma mensagem consiste em um cabeçalho e, opcionalmente, um corpo. O cabeçalho contém informações sobre a requisição ou resposta, enquanto o corpo carrega os dados da mensagem. No teste em questão (figura 4) o corpo da mensagem de resposta foi um texto HTML e o cabeçalho contendo informações como status da requisição e tipo de documento.

O overhead do encapsulamento usando HTTP pode variar dependendo do tamanho e da complexidade das mensagens HTTP, mas é geralmente relativamente baixo em comparação com outros protocolos mais complexos, como o TCP/IP.

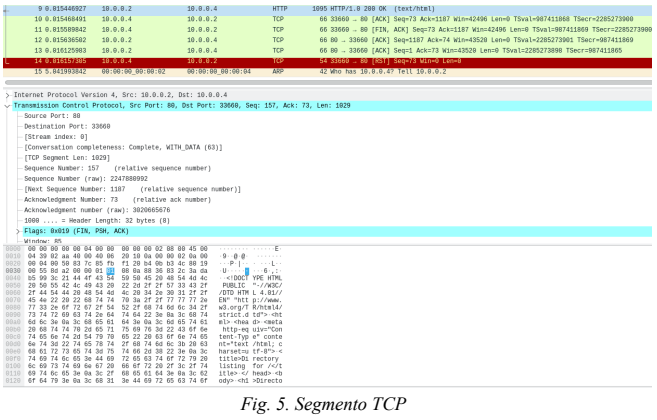

Fig. 5. Segmento TCP

D. Testes de conectividade

1. O primeiro teste do firewall consiste em testar a conectividade entre todos os nós quando não há regras implementadas (Figura 6).

```
mininet> pingall
*** Ping: testing ping reachability
h1 -> h2 h3 h4 h5 h6 h7 h8 h9 h10
h2 -> h1 h3 h4 h5 h6 h7 h8 h9 h10
h3 -> h1 h2 h4 h5 h6 h7 h8 h9 h10
h4 -> h1 h2 h3 h5 h6 h7 h8 h9 h10
h5 -> h1 h2 h3 h4 h6 h7 h8 h9 h10
h6 -> h1 h2 h3 h4 h5 h7 h8 h9 h10
h7 -> h1 h2 h3 h4 h5 h6 h8 h9 h10
h8 -> h1 h2 h3 h4 h5 h6 h7 h9 h10
h9 -> h1 h2 h3 h4 h5 h6 h7 h8 h10
h10 -> h1 h2 h3 h4 h5 h6 h7 h8 h9
*** Results: 0% dropped (90/90 received)
```

Fig. 6. Conectividade dos nós sem regras

2. O segundo teste do firewall consiste em testar a conectividade entres os nós com regras aplicadas e comparar com o primeiro teste. O ‘x’ representa os bloqueios durante o *pingall*. O caso abaixo representa a conexão quando as regras são:

- Host 1 ←x→ Host 2 (Bloqueado) (Figura 7)
- Host 6 ←x→ Host 4 (Bloqueado) (Figura 7)
- Porta 80 (Bloqueada) (Figura 8)

```
mininet> pingall
*** Ping: testing ping reachability
h1 -> X h3 h4 h5 h6 h7 h8 h9 h10
h2 -> X h3 h4 h5 h6 h7 h8 h9 h10
h3 -> h1 h2 h4 h5 h6 h7 h8 h9 h10
h4 -> h1 h2 h3 h5 X h7 h8 h9 h10
h5 -> h1 h2 h3 h4 h6 h7 h8 h9 h10
h6 -> h1 h2 h3 X h5 h7 h8 h9 h10
h7 -> h1 h2 h3 h4 h5 h6 h8 h9 h10
h8 -> h1 h2 h3 h4 h5 h6 h7 h9 h10
h9 -> h1 h2 h3 h4 h5 h6 h7 h8 h10
h10 -> h1 h2 h3 h4 h5 h6 h7 h8 h9
*** Results: 4% dropped (86/90 received)
```

Fig. 7. Conectividade dos nós com regras (Hosts)

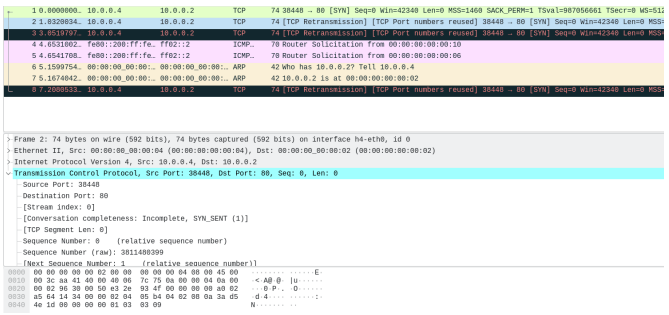

Fig. 8. Conectividade dos nós com regras (Porta 80)

IV. CONCLUSÃO

Neste projeto, foi explorado o ambiente das redes definidas por software (SDN) e sua aplicação em uma topologia criada no simulador Mininet, com suporte para OpenFlow. Desenvolvendo um aplicativo de segurança personalizado. Através da aplicação de regras no firewall, foi observado como o comportamento da aplicação entre dois hosts foi afetado, evidenciando a importância do controle eficiente e automatizado proporcionado pelas tecnologias de SDN e OpenFlow.

Com essa abordagem, pode-se experimentar e compreender a flexibilidade e a agilidade que as redes definidas por software trazem para lidar com as demandas em constante evolução de recursos, qualidade de serviço e segurança em ambientes modernos de computação e comunicação. E como o controle centralizado do tráfego em uma rede por meio de regras dinâmicas pode oferecer benefícios significativos em termos de eficiência e adaptação.

URL do vídeo: https://youtu.be/MiEsRMoIFfQ
Repositório: https://github.com/aquilamacedo/firewall-mininet

REFERÊNCIAS

[1] James F. Kurose & Keith W. Ross.. Computer Networking: a top-down Approach (8th Edition).. 8a. Addison Wesley. 2020

[2] Documentação do Wireshark <https://www.wireshark.org/docs/>

[3] Documentação do POX <https://noxrepo.github.io/pox-doc/html/>

[4] Documentação Mininet <http://mininet.org/walkthrough/>