Mitigação de ataques a rede através das Redes Definidas por Software

Guilherme Zanatta Tocchetto, Wagner dos Santos Marques, Paulo Silas Severo de Souza[1]

[1]Pontifícia Universidade Católica do Rio Grande do Sul (PUCRS)
Caixa Postal 1429 – 90.619-900 – Porto Alegre – RS – Brazil

{guilherme.tocchetto, wagner.marques.001, paulo.silas}@acad.pucrs.br

Abstract. *Redes definidas por software têm ganhado popularidade por prover maior flexibilidade e controle através da separação dos planos de controle e planos de dados. Ao passo que SDNs se baseiam em software e a inteligência da rede é centralizada no controlador, vulnerabilidades como adulteração de fluxos da rede no plano de dados. Neste sentido, este trabalho apresenta uma abordagem que realiza a detecção e mitigação de ataques DHCP Starvation através de funcionalidades providas pelo controlador Ryu. Experimentos foram realizados, e os resultados mostraram a eficácia da estratégia proposta.*

Resumo. *Redes definidas por software têm ganhado popularidade por prover maior flexibilidade e controle através da separação dos planos de controle e planos de dados. Ao passo que SDNs se baseiam em software e a inteligência da rede é centralizada no controlador, vulnerabilidades como adulteração de fluxos da rede no plano de dados. Neste sentido, este trabalho apresenta uma abordagem que realiza a detecção e mitigação de ataques DHCP Starvation através de funcionalidades providas pelo controlador Ryu. Experimentos foram realizados, e os resultados mostraram a eficácia da estratégia proposta.*

1. Introdução

A demanda por recursos computacionais em diferentes cenários têm crescido de forma significativa com o aumento da utilização de tecnologias como computação na nuvem e o surgimento de novos paradigmas como Internet das Coisas, onde diversos dispositivos espalhados pelo ambiente trabalham em conjunto para fornecer maior qualidade de serviço aos usuários. Um dos elementos fundamentais para o funcionamento adequado dessas abordagens são as tecnologias de redes de computadores, que são responsáveis por fornecer a conectividade entre os dispositivos.

Nesse contexto, redes definidas por software (SDN - *Software-Defined Networks*) têm ganhado popularidade por prover maior flexibilidade e controle sobre a rede através da separação dos planos de controle e planos de dados, centralizando a inteligência de rede em dispositivos chamados controladores (FEAMSTER, 2013). Não obstante aos benefícios trazidos por SDNs, existe uma preocupação em assegurar a segurança desse tipo de rede, ao passo que uma vez o atacante tendo acesso ao controlador, é possível alterar o comportamento dos dispositivos da rede (ALSMADI; XU, 2015).

Neste artigo, apresenta-se uma abordagem que realiza a detecção e mitigação de ataques DHCP Starvation em redes definidas por software utilizando funcionalidades do controlador Ryu. O restante deste artigo está organizado da seguinte maneira: nas Seções

2 e 3 é feita uma discussão sobre redes definidas por software, protocolo DHCP e sobre o ataque DHCP Starvation, na seção 4 são apresentados trabalhos relacionados, nas Seções 5 e 6 apresenta-se a abordagem para detecção e mitigação de ataques DHCP Starvation, na Seção 7 são apresentados os resultados obtidos e na Seção 8 são apresentadas as considerações finais.

2. Referencial Teórico

Por consequência da utilização de redes de computadores para diversos fins, percebe-se uma demanda por novas estratégias de planejamento e gerenciamento de redes, ao passo que tais tecnologias são utilizadas desde em ambientes urbanos através de redes domésticas (onde há uma preocupação com interferência de sinal e balanceamento de carga) até em ambientes empresariais de larga-escala, onde fatores como estratégias sofisticadas de segurança e disponibilidade são essenciais. Neste contexto, redes definidas por software emergem com a proposta de uso de *Application Programming Interfaces* (APIs) padronizadas de modo a permitir que redes sejam definidas, configuradas e gerenciadas via software.

No modelo tradicional de redes de computadores, dispositivos como roteadores e *switches* são constituídos por um plano de dados, que é responsável pelo encaminhamento dos pacotes, e por um plano de controle, que inclui funções responsáveis pela configuração do roteamento, redirecionamento do tráfego, controle de acesso, e assim por diante. Por outro lado, em redes definidas por software seguem a premissa de centralizar a inteligência da rede (desempenhada pelo plano de controle) em um único dispositivo de rede. Deste modo, dispositivos como *switches* e roteadores passam a conter somente plano de dados, e o processo de roteamento passa a ser feito por dispositivos chamados de controladores (KIRKPATRICK, 2013). A Figura 1 ilustra a diferença entre redes tradicionais e redes definidas por software.

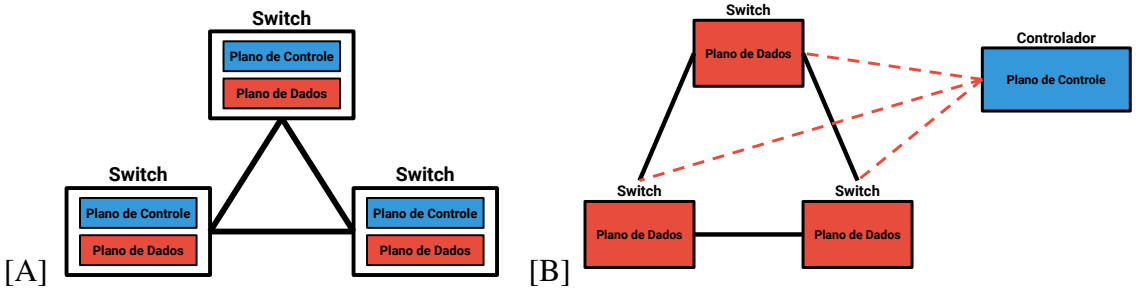

Figura 1. Comparação entre o modelo de redes tradicional (A) e redes definidas por software (B).

Através abordagem utilizada por redes definidas por software, é possível controlar cada fluxo na rede através da definição de regras que podem exercer controle sobre o funcionamento dos componentes da rede como switches e roteadores. Por exemplo, é possível obter maior velocidade na entrega dos pacotes e maior eficiência no uso dos recursos através do uso de estratégias de distribuição da carga de trabalho dentre os dispositivos de rede. Além disso, o uso de SDN facilita o gerenciamento da rede, ao passo que operadores podem, por exemplo, acessar os dispositivos controladores remotamente, podendo obter informações e realizar modificações nas características da rede em tempo de execução (HU; HAO; BAO, 2014).

A estratégia de centralização da inteligência da rede adotada em arquiteturas SDN também trás desafios como confiabilidade e segurança (BENZEKKI; FERGOUGUI; ELALAOUI, 2016). Por exemplo, diversos trabalhos já foram realizados com o foco em vulnerabilidades relacionadas à adulteração de fluxos da rede no plano de dados, que podem ser usados para atacar dispositivos de encaminhamento de pacotes ou mesmo controladores. Uma vez o atacante conseguindo acesso a um dispositivo controlador, é possível reconfigurar toda a rede de modo a extrair informações de seus usuários (KREUTZ et al., 2015). Neste sentido, há uma preocupação em desenvolver mecanismos que assegurem a segurança desse tipo de rede.

3. DHCP Starvation

Um dos elementos fundamentais em redes de computadores são os protocolos, que são responsáveis por definir regras e convenções que viabilizam a conexão entre elementos na rede. Um dos protocolos mais conhecidos atualmente é o DHCP (*Dynamic Host Configuration Protocol*), que é responsável pela configuração automatizada de informações de rede como endereço IP e máscara de subrede, e endereços DNS para clientes em uma rede. O protocolo DHCP não só facilita a configuração dos dispositivos da rede através de endereços IP, mas também provê mecanismos que evitam conflitos entre os endereços IP de dispositivos na rede (MUKHTAR; SALAH; IRAQI, 2012).

O processo de gerenciamento de endereços em uma rede controlada pelo protocolo DHCP é realizado por servidores DHCP, que são responsáveis por receber requisições dos clientes na rede e atribuir endereços aos mesmos. A atribuição de endereços realizada por servidores DHCP inicia-se quando o servidor recebe uma requisição de um cliente na rede. Na sequência, o servidor realiza uma busca em uma tabela que contém a relação entre os usuários (identificados pelos seus endereços MAC (*Media Access Control*)) e endereços IP, assim como outras informações como o instante de alocação do endereço IP ao cliente e o *lease time*, que é o tempo no qual o endereço estará reservado para cliente. Após identificar um endereço IP não designado a algum cliente, o servidor DHCP faz então a atribuição do mesmo ao cliente que realizou a solicitação de IP. Conforme ilustrado na Figura 2, o processo de atribuição de um endereço de IP a um cliente realizado por um servidor DHCP é realizado em quatro etapas:

1. **Discovery:** o cliente que deseja obter um endereço IP envia uma mensagem broadcast na rede informando que deseja obter um endereço IP.
2. **Offer:** um servidor DHCP recebe a mensagem DHCP *discovery* do cliente e verifica em sua tabela de endereços se existem endereços disponíveis. Caso haja algum endereço IP disponível, o servidor envia uma mensagem para o cliente contendo o endereço de IP sendo oferecido, o IP do servidor e o *lease time*.
3. **Request:** uma vez que o cliente escolhe uma das ofertas de IPs dos servidores DHCP disponíveis na rede, o cliente envia uma mensagem informando que a oferta foi aceitada.
4. **Acknowledgement:** quando o servidor DHCP recebe a mensagem confirmando que o cliente aceitou sua oferta de endereço IP, é feita uma modificação na tabela de endereços de modo a marcar o endereço IP atribuído ao cliente como reservado. A partir desse momento, o cliente é capaz de realizar troca de mensagens na rede com seu novo endereço IP.

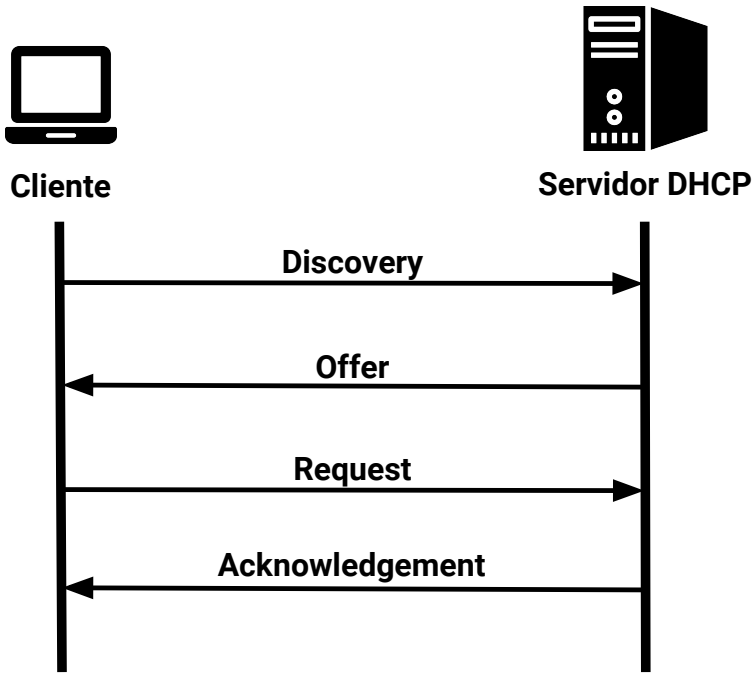

Figura 2. Processo de atribuição de endereços IP a clientes na rede realizado por servidores DHCP.

No ataque DHCP Starvation, o atacante utiliza ferramentas de MAC spoofing para alterar seu endereço MAC e realizar diversas requisições de solicitação de IPs ao servidor DHCP de modo a esgotar a lista de endereços disponíveis. Como a tabela de endereços de IP de servidores DHCP consiste em uma relação entre endereços os IP e os endereços MAC dos clientes, ao trocar seu MAC a cada requisição de IP, o atacante evita que o servidor DHCP identifique que mais de uma requisição de IP fora realizada pelo mesmo dispositivo. Deste modo, o servidor atribuirá todos seus endereços disponíveis ao atacante, que nunca fará uso dos mesmos. Assim, o servidor DHCP ficará temporariamente incapaz de atribuir novos endereços aos clientes na rede.

4. Trabalhos Relacionados

A segurança em ambientes SDN tem sido objeto de inúmeras pesquisas. Raghav e Dua (2017) enfatizam a relevância de técnicas para aumentar a segurança de redes definidas por software. Os autores apresentam no estudo algumas das falhas de segurança na arquitetura SDN e quais são os diferentes tipos de ataque possíveis em SDN. Em seguinda, desenvolveram um método baseado na intersecção de um conjunto de regras para realizar a detecção dos ataques. A topologia criada, de acordo com os autores, é uma topologia de propósito geral que pode ser estendido com mais switches e hosts. Para realização dos testes, o simulador Mininet foi utilizado em conjunto com o controlador Ryu.

Tselios, Politis e Kotsopoulos (2017) fornecem uma visão geral dos problemas comuns de segurança da SDN quando tal ambiente é adota junto às nuvens IoT. Logo, os autores descrevem os princípios de design recentemente introduzido paradigma Blockchain e defendem razões que tornam Blockchain como um fator de segurança significativo para soluções onde SDN e IoT estão envolvidos. De acordo com os autores, especialmente

quando SDN é usado para suporte a elementos de rede relacionados à Internet das Coisas, preocupações de segurança adicionais aumentam, devido à vulnerabilidade elevada de tais implantações possuem, a tipos específicos de ataques e à necessidade comunicação inter-nuvem que é comumente exigida por dispositivos IoT.

Ibdah et al. (2017) avaliam o impacto de diferentes ataques cibernéticos que podem direcionar a comunicação de rede inteligente, que é implementada como uma rede definida por software na operação do sistema de rede inteligente em geral. Foram definidos diferentes cenários de ataque, incluindo ataques DDoS, highjacking de localização e sobrecarga de link contra redes SDN de diferentes tipos de controladores, incluindo POX, Floodlight e RYU. Os resultados obtidos indicaram que os sistemas smartgrid habilitados para SDN são vulneráveis a diferentes tipos de ataques. O mininet foi o simulador escolhido para a realização dos testes.

Hussein et al. (2016) propuseram uma abordagem de design de segurança SDN com o intuito de garantir um bom equilíbrio entre o desempenho da rede e os recursos de segurança. No mesmo contexto, os autores mostram que a abordagem pode ser usada para evitar ataques DDoS visando o controlador ou os diferentes hosts na rede, bem como rastrear a origem do ataque. A solução está na introdução de um terceiro plano, denominado plano de segurança, além do plano de dados, responsável por encaminhar os pacotes de dados entre os switches SDN e paralelo ao plano de controle, responsável pela troca de regras e dados entre os switches. A avaliação indicou a capacidade do sistema proposto de impor diferentes níveis de segurança em tempo real definidos pelo usuário com baixa sobrecarga.

Diferente dos trabalhos apresentados, este trabalho visa verificar a viabilidade de utilizar o controlador Ryu para identificar e mitigar o ataque DHCP Starvation em um ambiente SDN. Semelhante aos trabalhos citados, o controlador Ryu foi utilizado com o simulador Mininet. Tais ferramentas foram adotadas devido sua ampla adoção no meio acadêmico com o intuito de verificar a viabilidade do SDN em diferentes cenários.

5. Etapas de Desenvolvimento

Nesta Seção é realizada a apresentação da topologia criada a fim de prover a realização dos testes no ambiente SDN. No mesmo contexto, destaca-se a estratégia de implementação da aplicação de ataque à rede, bem como ferramentas utilizadas para realização de tal ataque. Por fim, também é exposta a aplicação desenvolvida a fim de prover a detecção e mitigação do ataque.

5.1. Topologia

A topologia da rede consiste em três *hosts*, que correspondem a um servidor DHCP, um *host* que fará o disparo do ataque DHCP *Starvation* na rede (*Host 2*) e um *host* que representa um usuário comum que poderia estar utilizando a rede (*Host 1*), todos ligados a um *switch* que possui o controlador Ryu instalado no mesmo, como mostra a figura 1.

5.2. Ataque

Com o intuito de verificar as alternativas de mitigação do ataque DHCP Starvation junto ao ambiente SDN, foi desenvolvida uma aplicação maliciosa que visa realizar o envio de

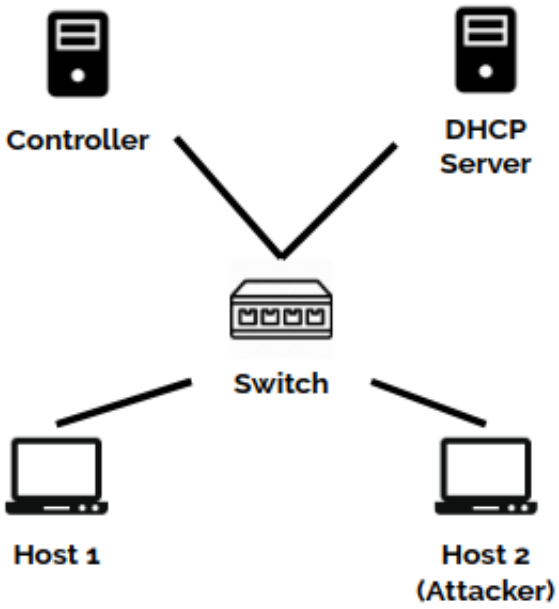

Figura 3. Topologia da Rede

pacotes maliciosos ao servidor DHCP com o intuito de tornar todos os IPs da rede indisponíveis para os novos dispositivos que tentarão realizar a conexão na rede. A ferramenta Scapy foi utilizada para a manipulação dos pacotes junto à aplicação maliciosa. Tal ferramenta consiste em um utilitário *open source* escrito na linguagem Python e comumente utilizado para realização de testes de segurança em redes de computadores (KOBAYASHI et al., 2007). Logo, a aplicação consiste em um laço de repetição que visa percorrer todos os endereços de IP definidos pelo usuário malicioso na rede, ao passo que em cada iteração, um novo pacote de DHCP request é criado e enviado para o servidor DHCP na rede. A Figura 4 apresenta o trecho de código onde o pacote é criado e enviado para o servidor DHCP utilizando a ferramenta Scapy.

```
conf.checkIPaddr = False
mac_address = RandMAC()

dhcp_request = Ether(src=mac_address, dst=broadcast)
    /IP(src="0.0.0.0", dst="255.255.255.255")
    /UDP(sport=68, dport=67)
    /BOOTP(chaddr=mac_address)
    /DHCP(options=[("message-type","request")
    ,("server_id","10.10.10.1")
    ,("requested_addr", IP_address_subnet + str(ip)),"end"])

sendp(dhcp_request)
```

Figura 4. Trecho de Código que apresenta a criação e o envio de pacotes maliciosos à rede.

O servidor de DHCP adotado já implementa alguns mecanismos de segurança, tais como a realização da verificação do endereço MAC de um determinado dispositivo a

fim de evitar o envio de mais de um endereço IP válido para o mesmo dispositivo. Nesse sentido, através da ferramenta citada, é possível gerar endereços MAC diferentes para cada um dos pacotes maliciosos que serão enviados ao servidor usufruindo da função RandMAC. Como já enfatizado, a aplicação realiza o envio de pacotes DHCP *requests*, e assim, é necessário ser de conhecimento do atacante o endereço IP do servidor DHCP, visto que o envio de um pacote de DHCP *discover* foi abstraído desta implementação.

Também é definido o *range* de IPs que a aplicação maliciosa irá solicitar ao servidor DHCP. Todavia, algum dos IPs solicitados pela aplicação poderia não estar mais disponível. Nesse sentido, não é realizada a verificação de disponibilidade do endereço, evitando que a aplicação aguarde pela confirmação do endereço, como ocorreria em um solicitação normal de IP por um dispositivo sem intenções maliciosas. Logo, após o envio do pacote DHCP solicitando um IP específico, o fluxo de iteração continua e a aplicação solicita todos os endereços de IP definidos no referido *range*. Após o termino da execução da aplicação maliciosa, todos os IPs disponíveis na rede foram reservados e não estão mais disponíveis.

6. Mitigação

No cenário de testes utilizado, o Ryu foi o controlador escolhido para ser utilizado em conjunto com o simulador Mininet. Este controlador é disponibilizado de maneira *open source* e sob a licença Apache 2.0. Assim como o simulador Mininet, o Ryu é escrito completamente baseado em Python e O código fonte principal pode ser encontrado no GitHub, fornecido e apoiado pela comunidade Open Ryu. Semelhante aos outros controladores SDN, Ryu também permite a criação de pacotes pacotes OpenFlow, bem como gerenciamento relacionado à entrada e saída de pacotes. Além disso, possui uma lista abundante de bibliotecas que suportam operações de processamento de pacotes (ASADOLLAHI; GOSWAMI; SAMEER, 2018).

A etapa de mitigação do ataque consiste em duas etapas: Identificação do ataque e execuções de ações para mitigação do ataque. O DHCP Starvation, como já citado, realiza o envio de pacotes à rede com a intensão de tornar todos os endereços IPs indisponíveis. Para identificar tal ataque, é necessário implementar ações dentro do controlador a fim de realizar verificações em todos os pacotes DHCP que são enviados à rede. Logo, é possível verificar diversos atributos, tais como MAC do remetente. Nesse contexto, uma simples verificação de MAC tornaria possível verificar se um determinado dispositivo está realizando uma série de requisições de IP na rede.

No entanto, como foi realizado e enfatizado, é possível realizar o envio de pacotes maliciosos realizando a alteração do atributo que corresponde ao endereço MAC de origem. Sendo assim, identificar o ataque torna-se uma tarefa não trivial. Logo, a fim de mitigar o ataque, foi definido junto ao controlador um conjunto de endereços de MAC que poderiam solicitar endereços IP através do protocolo DHCP na rede. Nesse sentido, quando um pacote é recebido pelo controlador, uma verificação é realizada a fim verificar se o endereço de origem do pacote faz parte do grupo de endereços já definidos na rede. A Figura 5 ilustra o trecho de código que realiza a referida verificação onde autorizamos apenas um endereço MAC a pedir endereços IP por meio do protocolo DHCP. Como observado, todos os pacotes recebidos passam inicialmente por uma análise a fim de averiguado se são pacotes DHCP. Caso positivo, é verificado se o endereço MAC de

origem é um endereço conhecido. Caso positivo, o pacote é encaminhado ao destino. Caso negativo, tal pacote é descartado.

```
pkt = packet.Packet(msg.data)
dhcpPacket = pkt.get_protocols(dhcp.dhcp)
if dhcpPacket:
    if dhcpPacket[0].chaddr != '00:00:00:00:00:05':
        print "Not valid. Discarding packet.\n"
    else:
        actions = [ofp_parser.OFPActionOutput(ofp.OFPP_FLOOD)]
        out = ofp_parser.OFPPacketOut(datapath=dp, buffer_id=msg.
                                      buffer_id, in_port=msg.
                                      in_port, actions=actions)
        dp.send_msg(out)
```

Figura 5. Trecho de Código que apresenta a mitigação dos pacotes maliciosos que são enviados à rede.

7. Testes e Resultados

A fim de verificar a viabilidade da abordagem desenvolvida no ambiente SDN, diversos testes foram realizados. O primeiro teste foi realizado com o intuito de verificar se a topologia criada atendia às especificações de envio de pacotes. Nesse sentido, sem a execução do código malicioso, o controlador foi instanciado, bem como a simulação do ambiente. Um *host* foi iniciado e através deste, foi solicitado um endereço DHCP à rede. Tal host recebeu um endereço IP válido, ao passo que foi possível estabelecer a conexão com os demais dispositivos conectados à rede por meio do protocolo ICMP (Internet Control Message Protocol).

A segunda avaliação consistiu em executar a aplicação maliciosa a partir de um *host* na rede. Tal etapa foi realizada a fim de verificar se o ataque seria bem sucedido em um ambiente SDN. Nesse sentido, o controlador foi instanciado sem os condicionais que tratavam os pacotes DHCP. Logo em seguida, a simulação do ambiente foi iniciada e um *host* foi iniciado com o código malicioso. Ao final da execução do ataque, um novo *host* foi iniciado, este sem endereço IP definido, e uma requisição por IP foi realizada a partir deste novo *host*. Como esperado, tal requisição não foi atendida, ao passo que o *host* aguardou resposta por um tempo finito e o requisição não surtiu efeito. Logo, ficou evidenciado que o ataque fora bem sucedido.

Por fim, a fim de verificar se a estratégia de mitigação do ataque havia sido realizada de maneira efetiva. O controlador foi instanciado com os condicionais desenvolvidos para realizar a avaliação dos pacotes recebidos. A simulação foi iniciada e dois *hosts* foram iniciados. Um destes sem endereço IP e o outro contendo o código malicioso. Inicialmente, o ataque foi iniciado e concluído. A fim de verificar se o ataque fora bem sucedido, através do segundo *host*, foi solicitado um endereço IP à rede. O servidor DHCP respondeu com o endereço de IP solicitado, ficando evidenciado que o ataque não ocorreu de maneira bem sucedida. Assim, a abordagem utilizada junto ao controlador a fim de mitigar o ataque DHCP Starvation mostrou-se satisfatória.

8. Conclusão

Por consequência do aumento da demanda por redes de computadores para diversos fins, percebe-se também uma demanda por novas estratégias de planejamento e gerenciamento de redes. Neste contexto, redes definidas por software (SDN) emergem com a proposta de uso de APIs padronizadas de modo a permitir que redes sejam definidas, configuradas e gerenciadas via software. No entanto, a SDN não está imune a vulnerabilidades de segurança que existem atualmente nos sistemas ou que pode surgir recentemente devido a alterações no design da rede (ARBETTU et al., 2016). Nesse sentido, este trabalho visa apresentar o desenvolvimento e avaliação da viabilidade de mitigação do ataque DHCP Starvation junto ao ambiente SDN.

O ambiente SDN foi simulado através da ferramenta Mininet[1] e o Ryu[2] foi o controlador adotado durante a realização dos testes. Para simulação do referido ataque, foi utilizada a biblioteca Scapy[3] que é amplamente utilizada para fins de testes de segurança em ambientes empresariais. Todavia, como tal ferramenta utiliza Python, sua utilização junto ao cenário SDN desenvolvido foi bem sucedida. Diversos cenários de teste foram criados e os resultados iniciais foram considerados satisfatórios, ao passo que através do controlador foi possível identificar e mitigar pacotes DHCP advindos de um usuário mal intencionado junto à rede estabelecida.

Como trabalhos futuros, almeja-se a análise de novas abordagens para a mitigação do ataque apresentado. No ambiente atual, novas máquinas, sem endereço Mac registrado junto ao controlador, não poderiam se conectar à rede, o que não seria viável em um cenário onde a rede tratada seria pública ou que recebesse constantemente novos dispositivos. Para isso, almeja-se realizar o tratamento dos pacotes recebidos e o registro das portas do switch que enviaram pacotes DHCP em um curto espaço de tempo. Com isso, teoricamente seria possível bloquear portas do switch que identificassem diversas requisições HTTP advindas da mesma porta. Tal abordagem mitigaria um possível ataque de DHCP Starvation e tornaria possível que novas máquinas se conectassem à rede.

Referências

ALSMADI, I.; XU, D. Security of software defined networks: A survey. *computers & security*, Elsevier, v. 53, p. 79–108, 2015.

ARBETTU, R. K. et al. Security analysis of opendaylight, onos, rosemary and ryu sdn controllers. In: IEEE. *Telecommunications Network Strategy and Planning Symposium (Networks), 2016 17th International*. [S.l.], 2016. p. 37–44.

ASADOLLAHI, S.; GOSWAMI, B.; SAMEER, M. Ryu controller's scalability experiment on software defined networks. In: *2018 IEEE International Conference on Current Trends in Advanced Computing (ICCTAC)*. [S.l.: s.n.], 2018. p. 1–5.

BENZEKKI, K.; FERGOUGUI, A. E.; ELALAOUI, A. E. Software-defined networking (sdn): a survey. *Security and communication networks*, Wiley Online Library, v. 9, n. 18, p. 5803–5833, 2016.

[1]Mininet. Disponível em: <http://mininet.org/>.

[2]Ryu Controller. Disponível em: <https://osrg.github.io/ryu/>.

[3]Scapy. Disponível em: <https://scapy.net/>.

FEAMSTER, N. Software defined networking. *Retrieved from coursera: https://class. coursera. org/sdn-001*, 2013.

HU, F.; HAO, Q.; BAO, K. A survey on software-defined network and openflow: From concept to implementation. *IEEE Communications Surveys & Tutorials*, IEEE, v. 16, n. 4, p. 2181–2206, 2014.

HUSSEIN, A. et al. Sdn security plane: An architecture for resilient security services. In: IEEE. *Cloud Engineering Workshop (IC2EW), 2016 IEEE International Conference on*. [S.l.], 2016. p. 54–59.

IBDAH, D. et al. On the security of sdn-enabled smartgrid systems. In: IEEE. *Electrical and Computing Technologies and Applications (ICECTA), 2017 International Conference on*. [S.l.], 2017. p. 1–5.

KIRKPATRICK, K. Software-defined networking. *Communications of the ACM*, ACM, v. 56, n. 9, p. 16–19, 2013.

KOBAYASHI, T. H. et al. Using a packet manipulation tool for security analysis of industrial network protocols. In: *2007 IEEE Conference on Emerging Technologies and Factory Automation (EFTA 2007)*. [S.l.: s.n.], 2007. p. 744–747. ISSN 1946-0740.

KREUTZ, D. et al. Software-defined networking: A comprehensive survey. *Proceedings of the IEEE*, Ieee, v. 103, n. 1, p. 14–76, 2015.

MUKHTAR, H.; SALAH, K.; IRAQI, Y. Mitigation of dhcp starvation attack. *Computers & Electrical Engineering*, Elsevier, v. 38, n. 5, p. 1115–1128, 2012.

RAGHAV, P.; DUA, A. Enhancing flow security in ryu controller through set operations. In: IEEE. *Computer and Communications (ICCC), 2017 3rd IEEE International Conference on*. [S.l.], 2017. p. 1265–1269.

TSELIOS, C.; POLITIS, I.; KOTSOPOULOS, S. Enhancing sdn security for iot-related deployments through blockchain. In: IEEE. *2017 IEEE Conference on Network Function Virtualization and Software Defined Networks (NFV-SDN)*. [S.l.], 2017. p. 303–308.