Uma Ferramenta para Automação de Testes de Cibersegurança em Redes SDN

Ryan M. S. Leal, Johan K. E. Freitas, Francisco A. C. Albuquerque Júnior, Waslon T. A. Lopes, Fabrício B. S. Carvalho e Iguatemi E. Fonseca

***Resumo*— Com o avanço e aumento da utilização de Redes Definidas por Software (SDN – *Software Defined Networks*), vulnerabilidades específicas para essa arquitetura têm sido cada vez mais comuns, sendo os ataques de negação de serviço, os mais comuns e mais explorados devido a característica centralizada da rede. Neste contexto, o presente trabalho apresenta uma ferramenta desenvolvida para automação de testes de cibersegurança em redes SDN.**

***Palavras-Chave*— Redes SDN, Cibersegurança, Automação de testes.**

I. Introdução

A criação dos conceitos que envolvem a arquitetura de Redes Definidas por Software (SDN - *Software Defined Networks*) traz consigo vantagens no que se refere a controle, programabilidade e automatização, fornecendo recursos cada vez mais poderosos para o gerenciamento das redes. Porém, uma nova arquitetura traz consigo também novas vulnerabilidades [1], permitindo que atacantes explorem diferentes formas de causar danos à rede e, consequentemente a seus usuários, principalmente pela natureza centralizada desse tipo de rede. Torna-se cada vez mais necessária a criação de rotinas de testes de segurança para assim garantir que a rede esteja protegida contra os principais métodos de ataque, dificultando as ações dos atacantes.

Contudo, realizar manualmente esses testes pode ser muito custoso e impreciso, especialmente quando executada por profissionais com pouco conhecimento em segurança de redes, ocasionando erros de configuração. Isso é caracterizado como um dos maiores riscos de segurança, segundo a OWASP [2], abrindo mais brechas para atacantes explorarem as vulnerabilidades. Portanto, a criação de uma ferramenta automatizada de testes de segurança faria a manutenção das redes muito mais eficiente e protegida contra falhas humanas, possibilitando ainda a inclusão em esquemas de orquestração da rede, tornando-se parte do ciclo de gerenciamento automatizado.

A. Redes SDN

As redes SDN representam uma arquitetura baseada na capacidade de prover programabilidade para aplicações de rede e na centralização da lógica de controle, promovendo a separação da rede em planos. Os *switches* e roteadores fazem parte do plano de dados e o controlador da rede encontra-se no plano de controle [3], essa separação beneficia atividades de gerenciamento e manutenção das redes, promovendo facilidades no monitoramento da rede e diminuindo a carga gerada no *switch*, visto que este não é mais o responsável por realizar as operações da lógica de controle e regras. Para gerar a comunicação entre os diferentes planos, são utilizados protocolos como OpenFlow e a linguagem de programação P4 (*Programming Protocol-independent Packet Processors*).

Ryan M. S. Leal, Johan K. E. Freitas, Francisco A. C. Albuquerque Júnior, Iguatemi E. Fonseca, Centro de Informática, Universidade Federal da Paraíba, João Pessoa-PB, e-mail: ryanleal@cc.ci.ufpb.br, iguatemi@ci.ufpb.br; Waslon T. A. Lopes, Fabrício B. S. Carvalho, Centro de Energias Alternativas e Renováveis, Universidade Federal da Paraíba, João Pessoa-PB. Este trabalho foi parcialmente financiado pela RNP, CAPES e CNPq.

B. Vulnerabilidades em Redes SDN

Com a centralização da rede no controlador, as redes SDN são mais suscetíveis a ataques DoS (*Denial of Service*), em que o atacante visa esgotar recursos e interromper o funcionamento da rede, podendo definir como alvos os planos de dados, de controle e a aplicação. Um outro fator que acaba ampliando a vulnerabilidade a esse tipo de ataque é o uso da memória TCAM (*Ternary Content-Addressable Memory*) nos *switches* de redes SDN, as quais são mais eficientes, porém mais caras, sendo assim, implementadas em uma quantidade limitada [4].

Os ataques DoS podem ser realizados com o uso de múltiplos dispositivos, gerando os ataques DDoS (*Distributed Denial of Service*), que podem utilizar métodos de alta ou baixa taxa, dependendo da do tráfego de rede gerado, no segundo caso, ao usar tráfego menor, torna-se mais difícil de detectar o ataque. Um dos maiores exemplos de DDoS em SDN é o Slow TCAM, ataque que usa uma *botnet* para gerar comunicações únicas e visa mantê-las ativas por meio de um tráfego mínimo para não ser removida por tempo limite - assim, o número de regras de encaminhamento enchem a tabela e os novos pacotes de usuários legítimos são ignorados, gerando a negação de serviço. Esse ataque pode ainda evoluir para o *Slow-Saturation* [4], caso o atacante gere estresse na comunicação entre *switch* e controlador, utilizando alta e baixa taixa de forma simultânea com sua botnet, gerando instabilidades e negação de serviço.

II. Desenvolvimento da Ferramenta

Visando facilitar o ciclo de manutenção de redes SDN, foi criada uma ferramenta que facilita o processo de testes de segurança. Seu funcionamento é focado na modularização, de forma que ataques possam ser inseridos e retirados sem necessidade de alterar todo o código, podendo ser feito adicionando os *scripts* de ataques na pasta e alterando os dados dos arquivos de configuração, indicando quais ataques serão

utilizados e seus respectivos parâmetros, como é exibido no diagrama da Figura 1. Para implementação dessa ferramenta foi utilizada a linguagem de programação Python na versão 3.x, utilizando algumas de suas bibliotecas como *subprocess* para executar ferramentas de segurança presentes no Kali Linux, *Scapy* para criar *scripts* personalizados de ataques e *logging* para gerar os *logs* de execução dos ataques, os quais são utilizados como saída da ferramenta, exibindo mensagens em diferentes níveis de atenção que são exibidos no *console* e também são armazenados em um arquivo de resultados.

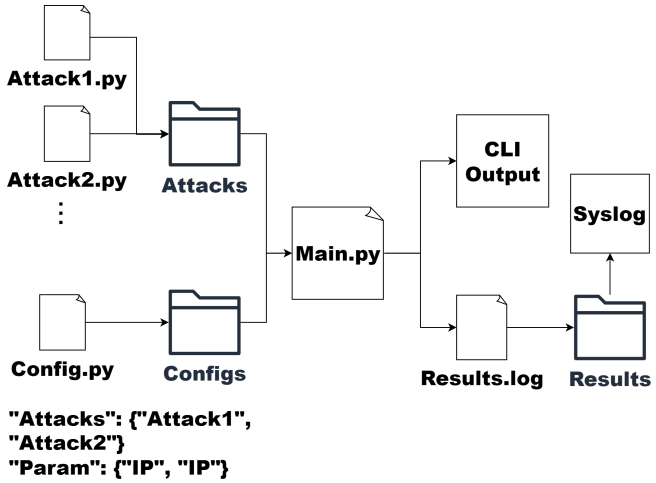

Fig. 1. Diagrama de funcionamento da ferramenta de testes de segurança.

O arquivo de configuração, conforme exibido na figura, utiliza dicionários e listas para armazenar as configurações do teste a ser executado. A ferramenta base utiliza tais dados para executar os *scripts* selecionados e que estão na pasta de ataques, permitindo a execução automática dos *scripts* configurados, os quais utilizam tanto ferramentas de segurança presentes no sistema do usuário quanto as criadas de forma personalizada, gerando a saída via *console* e arquivo de *log*. Estes *logs* de resultados podem ser utilizados por uma ferramenta de *Syslog* para gerar uma exibição mais detalhada e visual das informações, o que facilita a tomada de decisão por parte da equipe de manutenção. Além disso, explora-se a possíbilidade de adicionar sugestões de mudanças na rede, de acordo com as vulnerabilidades e erros encontrados naquela sessão de testes.

III. Testes realizados

Com a implementação base da ferramenta, foram executados testes em ambiente local, para avaliação do seu desempenho. Para sua criação, foram utilizadas duas máquinas virtuais, uma com o Xubuntu 22.04 (na qual foi executado o controlador ONOS 3.0.0) e uma máquina Kali Linux 2023.4 (a qual executou o emulador de redes SDN, Mininet [5][6], por meio do qual foi emulada uma rede com topologia em árvore de profundidade 3, controlada pelo ONOS). Arbitrariamente foi escolhida uma das máquinas *host* e por meio dela foi executada a ferramenta, a qual testou dois ataques de inundação: *macof*, que gera inundação de pacotes ARP (*Address Resolution Protocol*) com endereços MAC randômicos; e *Hping*, a qual gera pacotes TCP/UDP/ICMP (TCP - *Transmission Control Protocol*, UDP - *User Datagram Protocol*, ICMP - *Internet Control Message Protocol*) personalizados e que permite também gerar inundações (tudo isso controlado por meio de seus parâmetros, que podem ser selecionados por meio do arquivo de configuração da ferramenta criada). Na Figura 2 é apresentada a saída nos logs da ferramenta após a execução do teste com *timeout* de 30 segundos para cada ataque. Foi realizado um comparativo de tempo de execução dos testes manualmente e com o uso da ferramenta, obtendo 1min40s para os testes manuais e 1min05s com o uso da ferramenta, demonstrando o quão potencialmente pode-se obter menos gastos de tempo conforme mais complexo for o cenário de testes que deseja-se representar. No caso de um teste com 2 ataques obteve-se um ganho de 35 segundos, i.e., um aumento de 53,8% no tempo de execução. Com mais ataques, os ganhos de tempo podem ser ainda maiores.

```
2024-06-03 21:47:42,377 - __main__ - INFO - Executando ataque:
macof_attack com parâmetros: {'interface': 'h5-eth0'}
2024-06-03 21:47:42,380 - attack_scripts.macof_attack - INFO -
Executando macof na interface h5-eth0
2024-06-03 21:48:12,575 - attack_scripts.macof_attack - INFO -
Execucao macof na interface h5-eth0 atingiu tempo limite
2024-06-03 21:48:12,661 - __main__ - INFO - Ataque macof_attack
concluído com sucesso: None
2024-06-03 21:48:12,664 - __main__ - INFO - Executando ataque:
hping_attack com parâmetros: {'target': '10.0.0.5', 'interface':
'h5-eth0'}
```

Fig. 2. Saída do arquivo de log após execução do teste.

IV. Conclusões

A ferramenta automatizada construída pode ser muito útil na rotina de manutenção de uma rede SDN, podendo ser integrada nos ciclos de CI/CD (CI - *Continuous Integration*, CD - *Continuous Delivery*) e sendo facilmente modificada para adicionar novos *scripts* de ataques, de forma a personalizar seu uso. Além disso, a integração com uma ferramenta de *Syslog* pode facilitar ainda mais a visualização dos resultados.

Como continuação deste trabalho, a ideia é melhorar as funcionalidades existentes, adicionar novos *scripts* de ataques, realizar a integração com sistema de *Syslog* (gerando uma visualização gráfica dos resultados, com possibilidade de adicionar sugestões de melhorias na rede) e, por último, pretende-se aplicar a ferramenta em um ambiente de rede 5G OpenRAN, testando sua utilidade em um cenário real e inovador.

Referências

[1] J. C. Correa Chica, J. C. Imbachi, and J. F. Botero Vega, "Security in SDN: A comprehensive survey," *Journal of Network and Computer Applications*, vol. 159, June 2020.

[2] O. Foundation, "Top 10 web application security risks," 2023. `https://owasp.org/www-project-top-ten/` [Accessado: 03 de Junho de 2024].

[3] T. Li, J. Chen, and H. Fu, "Application scenarios based on SDN: An overview," *Journal of Physics: Conference Series*, 2019.

[4] T. A. Pascoal, Y. G. Dantas, I. E. Fonseca, and V. Nigam, "Slow TCAM exhaustion DDoS attack," in *IFIP International Conference on ICT Systems Security and Privacy Protection*, pp. 17–31, Springer, 2017.

[5] M. P. Contributors, "Mininet - an instant virtual network on your laptop (or other pc)," 2022. `https://mininet.org/` [Acessado: 05 de Junho de 2024].

[6] I. A. Salti and N. Zhang, "Link-guard: An effective and scalable security framework for link discovery in sdn networks," *IEEE Access*, vol. 10, pp. 130233–130252, 2022.