Análise de vulnerabilidades em aplicações que usam API REST

Marcelo D. S. Leite, Francisco S. de Sousa Júnior e Iguatemi E. Fonseca

***Resumo*— Cada vez mais sistemas críticos e de alta escala estão se conectando e a interoperabilidade entre sistemas nunca foi tão desenvolvida. As APIs REST são uma forma de expor serviços web e estão sendo fortemente utilizadas para a comunicação de sistemas heterogêneos. Entretanto, a falta de boas práticas tanto no seu uso quanto no desenvolvimento desses serviços está expondo falhas de segurança, comprometendo assim os sistemas aos quais esses serviços estão sendo expostos. Este trabalho apresenta uma análise na exploração de vulnerabilidades nas APIs REST e faz, como estudo de caso, diversos testes experimentais na arquitetura do controlador SDN ONOS (*Open Network Operating System*) de uma rede 5G OpenRAN.**

***Palavras-Chave*— OpenRAN, 5G, SDN, API REST, OWASP TOP 10 API Security**

***Abstract*— More and more critical and high-scale systems are connecting, and interoperability between systems has never been so well-developed. REST APIs are a way to export web services and are heavily used for communication between heterogeneous systems. However, the lack of good practices in both the use and development of these services exposes security flaws, thus compromising the systems to which these services are being exposed. This work presents an analysis of the exploitation of vulnerabilities in the REST API and, as a case study, makes several experimental tests on the architecture of the SDN driver ONOS (Open Network Operating System) of a 5G OpenRAN network.**

***Keywords*— OpenRAN, 5G, SDN, REST API, OWASP TOP 10 API Security**

I. Introdução

A interface de comunicação API REST é uma tecnologia que utiliza HTTP e disponibiliza serviços web, permitindo a comunicação entre sistemas e dispositivos com um forte poder de interoperabilidade. De acordo com relatórios de segurança de APIs providos pelo Google [1], o número de ataques nas APIs REST está crescendo fortemente. Este relatório aponta que 50% das organizações experimentaram um incidente de segurança em APIs REST nos últimos 12 meses e que a fonte de potenciais ameaças nas APIs REST são 40% em APIs mal configuradas e 35% em APIs sem a devida manutenção, seja em testes de software, seja em validações de segurança. Dado esse cenário, qualquer vulnerabilidade encontrada nesses serviços tem um alto impacto para os sistemas.

No trabalho realizado em [2], 52 API's REST foram testadas. Seguindo um critério definido pelos autores, foi demonstrado que quase metade dos serviços continha problemas de implementação dos serviços APIs REST acarretando assim problemas de segurança, comprometendo os sistemas que os hospedavam. O paradigma de Redes Definidas por Software (SDN – *Software Defined Networks*) [3] permite controlar uma rede de maneira centralizada, por meio de aplicativos de software, independentemente da tecnologia subjacente utilizada.

O cenário de ameaças na arquitetura SDN tem sido alvo de estudos recentes, como mencionado em [4], [5] e [6]. De acordo com [6], as ameaças nesta arquitetura podem ser divididas em 7 grandes grupos: vazamento de dados, modificação de dados, negação de serviço, problemas de configuração, aplicativos comprometidos e maliciosos, acesso não autorizado e segurança SDN ao nível de sistema. Estas ameaças são mapeadas em preocupações de segurança como: autenticação, sigilo e integridade. O controlador SDN ONOS [7] é um controlador que atende o paradigma SDN gerenciando seus componentes e permite uma comunicação tanto externa quanto interna de alguns protocolos, como, por exemplo, o API REST. Os controladores SDN utilizam APIs REST como uma forma de gerenciamento dos seus serviços, entretanto, por conta de sua ampla utilização e da importância estrutural dos controladores SDN em uma infraestrutura de rede, este ponto acaba se tornando um possível vetor de ataque. Com isso, a principal contribuição deste artigo é a realização de uma análise de vulnerabilidade das APIs REST e executando, como estudo de caso, experimentos no popular controlador SDN ONOS em uma rede 5G OpenRAN. Uma análise qualitativa preliminar, na qual indicou-se caminhos para o estudo da problemática em pauta neste artigo, foi realizada recentemente [8].

O restante deste artigo está organizado como a seguir. A Seção II traz uma breve descrição da problemática encontrada em um cenário de vulnerabilidades nas APIs REST. A metodologia, os resultados experimentais e as ferramentas utilizadas são apresentados na Seção III. As conclusões e comentários finais são mostrados na Seção IV.

II. A problemática de Segurança nas APIs REST

Por questão de interoperabilidade entre sistemas, a utilização de API REST está bastante popular e cada vez mais integrada entre os sistemas de software. Por sua vez, aspectos de segurança em sua utilização estão cada vez mais sendo ignorados, trazendo assim uma brecha de segurança bastante impactante e refletindo, portanto, na segurança desses sistemas [2]. Partindo de um ataque de cibersegurança clássico, vários passos são necessários até que o atacante possa enfim acessar e explorar uma brecha de segurança. O ataque tem como primeiro passo o reconhecimento do alvo, no qual se inicia

Marcelo D. S. Leite, Francisco S. de Sousa Júnior e Iguatemi E. Fonseca, Centro de informática, Universidade Federal da Paraíba, João Pessoa-PB, e-mail: marcelodinizsl@gmail.com; franciscosantana203@gmail.com; iguatemi@ci.ufpb.br.

uma varredura do IP do host, portas e serviços disponibilizados pelo mesmo. Uma vez que esta informação é descoberta, um segundo passo se inicia, no qual é definida uma melhor forma para atacar tal alvo. Tendo sucesso, o atacante acessa o host e então se inicia o passo de infiltração neste sistema. Em seguida, o atacante precisa ficar fora de ser detectado por sistemas de detecção de intrusão e elevar os privilégios do seu status de usuário logado corrente. Finalmente, com todos esses passos, o atacante pode, enfim, acessar e explorar a falha de segurança que é o ponto-alvo inicial de seu ataque.

Um ataque proveniente de uma brecha de segurança em uma API REST não segue este passo a passo de um ataque clássico, pois um determinado serviço já está sendo exposto em forma de *endpoint*. Dependendo de como um determinado sistema foi arquitetado, não é nem necessário a autenticação do usuário para acessar e consumir este recurso exposto pelo *endpoint*, permitindo assim uma maior facilidade na exploração e descoberta de uma possível brecha de segurança.

Por conta deste crescente cenário inseguro, o projeto *OWASP (Open Web Application Security Project)*, a qual é uma organização sem fins lucrativos dedicada à segurança de projetos de software, criou uma lista das TOP 10 falhas de segurança mais comuns em se tratando de API REST. Tal lista é atualizada em média a cada 2 anos e é chamada de *OWASP TOP 10 API Security*, que serve como conjunto de boas práticas e orientação no desenvolvimento de interfaces que utilizam APIs REST e suas aplicações *web*. O *Top 10 REST API* de vulnerabilidade divulgada pela OWASP é [9]:

1) Autorização ao nível de objeto quebrado;
2) Autenticação quebrada;
3) Autorização de nível de propriedade de objeto quebrado;
4) Consumo irrestrito de recursos;
5) Autorização de nível de função quebrada;
6) Acesso irrestrito a fluxos de negócios sensíveis;
7) Falsificação de Solicitação do Lado do Servidor (SSRF);
8) Configuração incorreta de segurança;
9) Gerenciamento de estoque inadequado;
10) Consumo inseguro de APIs.

III. Metodologia e Resultados Experimentias

A. Metodologia

Como ilustrado na Figura 1, a ideia do trabalho foi analizar vulnerabilidades nas APIs REST. Como estudo de caso, foi investigado os *endpoints* disponibilizados pelo controlador SDN ONOS. Para isso, a metodologia escolhida foi o uso de técnicas e ferramentas de pentest, bem como criar scripts de ataques, utilizando a linguagem de programação Python, para explorar vulnerabilidades presentes na lista de vulnerabilidades do *OWASP TOP 10 API Security* [9]. Os experimentos foram conduzidos em ambiente real, i.e., no Testbed do Projeto OpenRAN@Brasil da RNP e CPqD [10].

Como será mostrado nos resultados adiante, um atacante externo à rede 5G OpenRAN conseguiu acesso ao seu núcleo, por meio de técnicas de pentest, sendo capaz de executar qualquer ação com status de usuário administrador. Isso representa uma falha grave no sistema de autenticação de usuário do controlador ONOS.

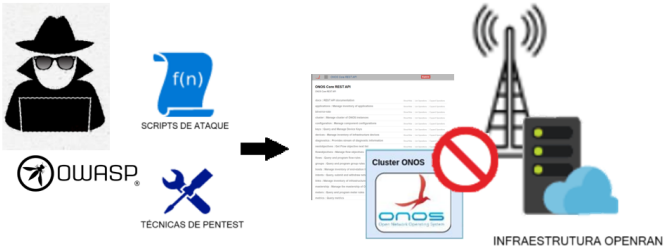

Fig. 1. Medotologia do ataque ao controlador ONOS.

B. Cenário Experimental

A Figura 2 representa o cenário testado no *testbed*, no qual foram utilizados 2 *hosts*, máquinas com a distribuição Kali Linux, utilizadas para simular os atacantes. Os *switches* e a rede SDN foram emulados usando o Mininet dentro desses *hosts*. Quanto ao controlador de rede, foram feitos experimentos tanto com 1 instância quanto com 3 instâncias do controlador SDN ONOS, na sua versão mais recente, versão 2.5.9, em um cluster no *Kubernet*, fornecendo alta disponibilidade e aproximando de um ambiente de rede real. Esta versão do Controlador ONOS não tinha nenhuma funcionalidade implementada para mitigar as vulnerabilidades, como por exemplo, o HTTPS.

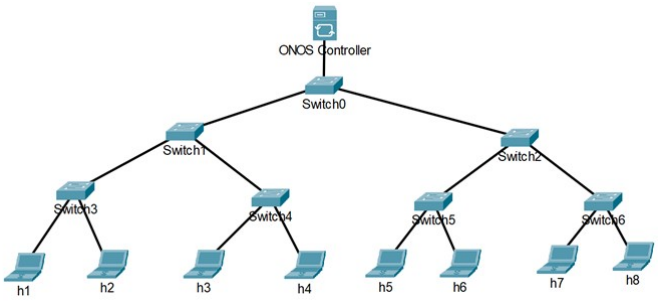

Fig. 2. Configuração do Controlador SDN ONOS no Testbed do Projeto OpenRAN@Brasil.

O ONOS é um controlador SDN de código-fonte aberto que controla como os pacotes de dados são encaminhados, gerencia componentes de rede (como *switches* e *links*), executa programas e módulos, provendo assim serviços de comunicação entre *hosts* e redes de computadores. Funciona como um sistema distribuído por meio de vários servidores, provendo suporte à tolerância a falhas sem interromper o tráfego da rede. A Figura 3 traz as APIs REST expostas pelo Controlador SDN ONOS. É possível acessar os *endpoints* da interface *web* do *swagger* [11] que implementa a especificação OpenAPI por um navegador *web*. Para ter acesso a esta interface, é necessário que o usuário esteja autenticado. Assim como qualquer ferramenta, para ter acesso a esta lista de API e consumi-la precisa estar autenticado.

C. Experimentos

Este trabalho realizou uma varredura nas APIs REST disponibilizadas pelo controlador SDN ONOS [12] com a lista de vulnerabilidades das APIs REST divulgadas pelo *OWASP TOP*

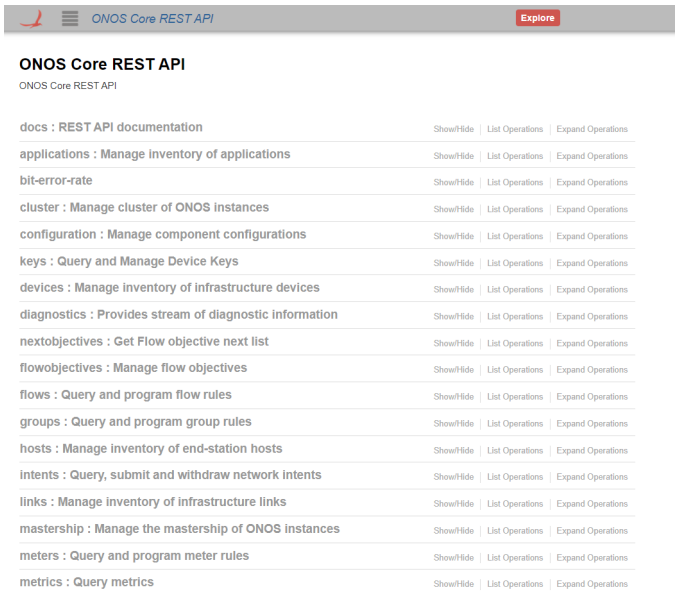

Fig. 3. APIs REST expostas pelo Controlador SDN ONOS.

10 API Security. Após essa varredura, foi identificado que, das 10 vulnerabilidades citadas pelo TOP 10, o controlador ONOS está vulnerável em pelo menos 2 delas. As vulnerabilidades listadas podem ser exploradas e têm um alto impacto no comprometimento dos serviços expostos, na integridade dos dados e no funcionamento do sistema. Para explorar estas vulnerabilidades, foram utilizados mais de um ataque, em que seus detalhamentos e explicações seguem nas seções seguintes.

Em se tratando da API.9, o controlador ONOS não utiliza versionamento de API REST logo a versão atual dos *endpoints* sempre é atualizada a cada *release* da versão do controlador ONOS, ficando assim as versões anteriores não acessíveis. Para a API.10, o controlador ONOS não consume nenhuma API externa, deixando assim esta vulnerabilidade não explorável.

Este artigo se propõe a se aprofundar em duas vulnerabilidades encontradas:

1) API.2 Autenticação quebrada;
2) API.4 Consumo irrestrito de recursos;

D. Autenticação quebrada

A vulnerabilidade autenticação quebrada (*Broken Authentication*) aborda falhas nos mecanismos de autenticação de APIs, como senhas fracas, tokens mal configurados ou expostos, permitindo que atacantes comprometam contas ou assumam identidades de outros usuários. Na exploração desta vulnerabilidade, foram criados vários *scripts* com a intenção de varrer as possibilidades de vulnerabilidades sobre autenticação.

1) Força Bruta: Para realizar o teste desta vulnerabilidade, foi fixado o nome de usuário e utilizada uma *wordlist* para tentar descobrir a senha correspondente. O teste foi realizado em uma URL alvo que fornece uma interface *web* de uma instância do controlador ONOS, com o nome de usuário definido como **onos** e utilizando a *wordlist* **rockyou.txt**. O processo de autenticação do nome de usuário e senha utiliza codificação em **base64**. No *script* de ataque, foi utilizado o arquivo **rockyou.txt**, combinando o nome de usuário para realizar um ataque de dicionário para descobrir a senha utilizada.

2) Ataque man-in-the-middle (MITM): Este ataque é realizado quando existe um agente malicioso interceptando a comunicação entre o cliente e o servidor. O objetivo deste teste foi de interceptar as requisições HTTP relacionadas à autenticação do usuário no momento do acesso ao controlador SDN ONOS por meio de usuário e senha. Para os testes relacionados com a API REST foi utilizado o Postman [13], como cliente para as APIs REST, e um navegador *web* para o acesso *web* no controlador SDN ONOS. Também foi utilizada nos testes uma ferramenta bastante utilizada no cenário de teste de penetração, chamada *Burp Suite* [15], da qual funciona como um *proxy*, capturando as requisições entre o cliente e o servidor. Com isso, as mensagens *request* e *post* são capturadas e podem ser manipuladas com objetivos específicos para a simulação de um ataque MITM.

O mecanismo de autenticação utilizado nos testes foi o *basic authentication*, o qual utiliza login e senha e é o método de autenticação que vem configurado por padrão no controlador SDN ONOS. Uma vez feita a requisição de autenticação via API REST ou *Web Browser*, a ferramenta *Burp Suite*, configurada como *proxy* para essas requisições, foi possível capturar e examinar o seu conteúdo. Na Figura 4 é possível ver que a informação do usuário e senha estão no cabeçalho da requisição usando a codificação *Base64*. Essa codificação não é uma forma de criptografia e sim uma codificação como o UTF-8 ou UTF-16, podendo assim, a informação ser mudada para outro formato e perdendo o seu sigilo.

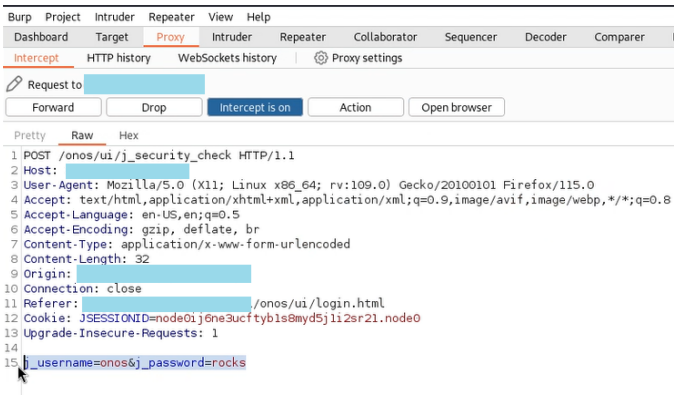

Fig. 4. Captura de requisição de autenticação pelo web browser pela ferramenta Burpsuite.

3) Sniffing de rede: Outra técnica utilizada foi um *sniffing* de rede utilizando a ferramenta Wireshark [14], no momento em que o usuário realizava o *login* na tela de autenticação do usuário do controlador ONOS pelo navegador *web*. Conforme a Figura 5, é possível verificar os pacotes relacionados a requisição de autenticação do usuário. A informação referente ao usuário e senha está capturada em texto puro, sem nenhuma codificação ou criptografia, expondo assim a vulnerabilidade deste método de autenticação padrão do controlador SDN ONOS.

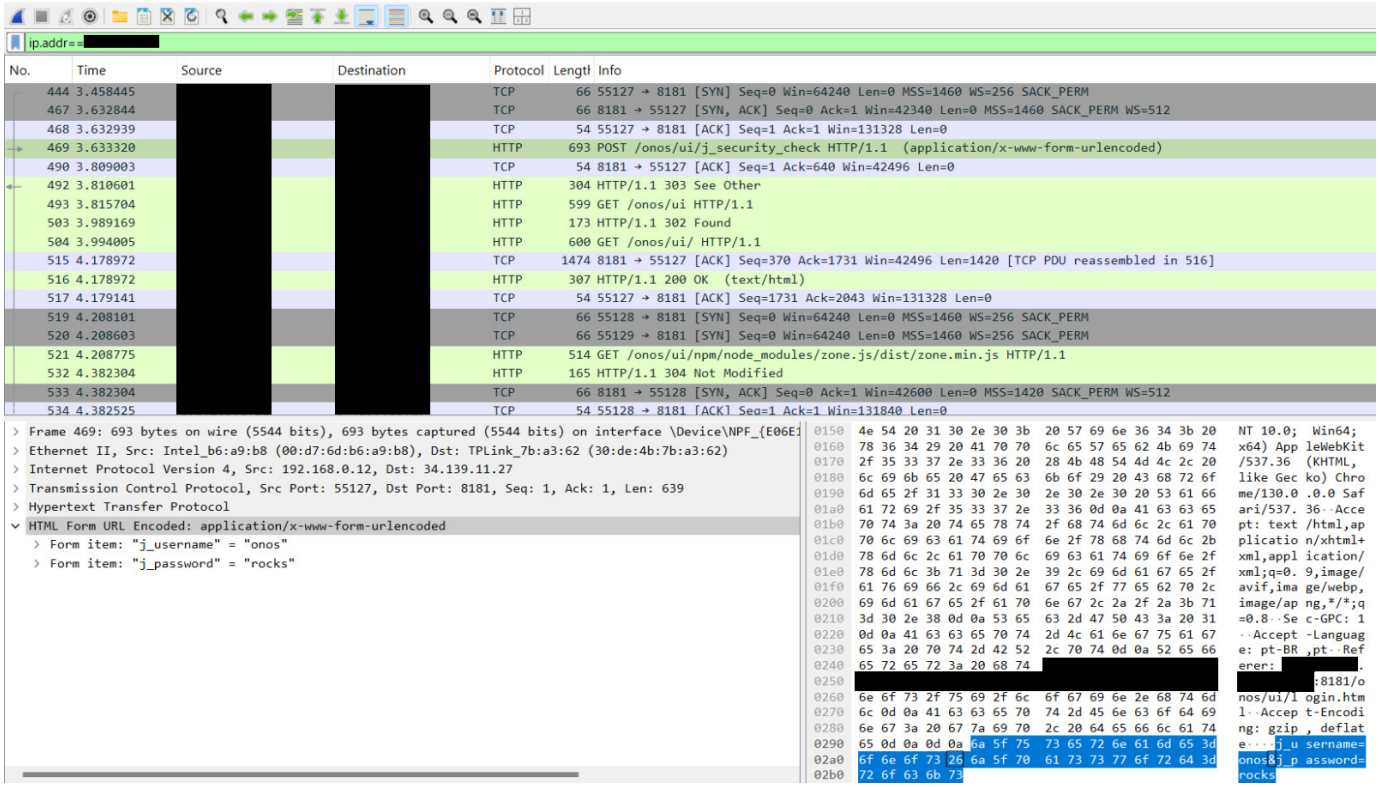

Fig. 5. *Sniffing* realizado pela ferramenta Wireshark.

E. Consumo irrestrito de recursos

O consumo irrestrito de recursos (*Lack of Resources and Rate Limiting*) trata da ausência de restrições em APIs REST para controlar o número de requisições ou o uso de recursos, permitindo abusos como ataques de negação de serviço (DoS) ou sobrecarga do sistema, comprometendo a disponibilidade. Para essa vulnerabilidade, foi utilizado o método de força bruta descrito a seguir.

1) Força Bruta por tentativa de login: Foi possível verificar que o controlador SDN ONOS apresenta uma vulnerabilidade relacionada à ausência de limitação no número máximo de requisições provenientes de uma mesma origem. Ao modificar o *script* anterior para ser executado por *threads*, e cada execução do programa ser realizada por meio de uma instância, foram utilizadas 100 instâncias e 100 *threads* para cada uma, para analisar como o servidor responde a uma inundação de requisições.

Os resultados mostrados na Figura 6 foram causados por *timeouts* nas requisições, devido à incapacidade do servidor de lidar com o volume de requisições simultâneas. Essa vulnerabilidade ocorre porque o controlador SDN ONOS não possui uma limitação no número máximo de requisições por minuto provenientes de uma mesma origem.

F. Impactos na arquitetura de uma rede 5G ou OpenRAN

Existe um mapeamento direto entre os ataques realizados neste artigo e os impactos no controlador ONOS. Consequentemente, a arquitetura de uma rede 5G ou OpenRAN também será atingida, comprometendo os serviços expostos e a quebra da integridade dos dados.

Em relação à vulnerabilidade de autenticação quebrada, o atacante, que é uma pessoa não autorizada, uma vez obtendo acesso aos *endpoints*, pode realizar ataques na integridade dos dados, nos quais os valores dos componentes são alterados, possibilitando assim o mau funcionamento da rede 5G e seus serviços. Outra possibilidade seria a reconfiguração total dos componentes da rede, como, por exemplo, os componentes mostrados na Figura 1, causando, deste modo, uma falha estrutural completa na arquitetura do Testbed ou até mesmo um comportamento malicioso desta arquitetura.

Em relação à vulnerabilidade Consumo Irrestrito de Recursos, um ataque de negação de serviço pode desencadear uma falha no funcionamento do controlador ONOS, impactando assim o comportamento da arquitetura e disponibilidade dos serviços da rede 5G ou OpenRAN. Como não existe um limitador de recursos, o processamento e memória são consumidos, provocando então um comportamento imprevisível do controlador ONOS.

IV. Conclusão

Este artigo apresentou um estudo, realizado em um ambiente real, sobre vulnerabilidades encontradas nas APIs REST tendo, como estudo de caso, o controlador SDN ONOS de uma rede 5G OpenRAN. Os testes foram executados com base na lista de vulnerabilidade *OWASP Top 10 API Security* [9]. Os experimentos foram conduzidos no *testbed* do Projeto OpenRAN@Brasil da RNP e CPqD [10]. Os resultados mostraram inicialmente duas importantes vulnerabilidades relacionadas à

Fig. 6. Mensagens de *timeout* como resposta do controlador SDN ONOS.

autenticação e ao consumo irrestrito de recursos. Uma forma de mitigar estas vulnerabilidades seria a utilização de uma forma mais sofisticada de autenticação, além do usuário e senha, sendo o padrão na instalação do controlador SDN ONOS. A utilização do protocolo HTTPS, como forma de criptografar o conteúdo do que é trafegado em suas requisições pela rede, pode ser uma alternativa para mitigação de vulnerabilidades. Outra possibilidade é a implementação de um limitador de recursos inteligente que seja capaz de analisar o tráfego de requisições e, se necessário, limitar a quantidade de requisições vindas de um determinado *host*.

REFERÊNCIAS

[1] API Security Research Report 2022 - https://cloud.google.com/resources/api-security-research-report (Acessado em Maio 2025).
[2] N. Laranjeiro, J. Agnelo and J. Bernardino, "A Black Box Tool for Robustness Testing of REST Services,"in *IEEE Access*, vol. 9, pp. 24738-24754, 2021.
[3] J, Elumalai and Subhashini, R.. (2023). Review of the SDN Architecture with Various API Controllers, in *International Conference on Research Methodologies in Knowledge Management, Artificial Intelligence and Telecommunication Engineering* (RMKMATE) 1-6, 2023.
[4] A. Kanwal, M. Nizamuddin, W. Iqbal, W. Aman, Y. Abbas and S. Mussiraliyeva, "Exploring Security Dynamics in SDN Controller Architectures: Threat Landscape and Implications,"in *IEEE Access*, vol. 12, pp. 56517-56553, 2024.
[5] M. F. ELHejazi and W. H. A. Muragaa, "Improving the Security and Reliability of SDN Controller REST APIs Using JSON Web Token (JWT) with OpenID and auth2.0,"*IEEE 4th International Maghreb Meeting of the Conference on Sciences and Techniques of Automatic Control and Computer Engineering (MI-STA)*, Tripoli, Libya, pp. 398-402, 2024.
[6] F. Bannour, S. Dumbrava and D. Lu, "A Flexible GraphQL Northbound API for Intent-based SDN Applications,"*IEEE/IFIP Network Operations and Management Symposium (NOMS)*, Budapest, Hungary, 2022, pp. 1-5, 2022.
[7] ONOS Web Site - https://opennetworking.org/onos/ (Acessado em Maio 2025).
[8] LEITE, M. D. S. ; LOPES, W. T. A. ; CARVALHO, F. B. S. ; Fonseca, I. E., "Estudo de Vulnerabilidades em API e APP de Controladores SDN ONOS", In: *XLII Simpósio Brasileiro de Telecomunicações e Processamento de Sinais*, v. 1. p. 1-2, 2024.
[9] OWASP API Security - https://owasp.org/www-project-api-security/ (Acessado em Maio 2025)
[10] OpenRAN Brasil Web site - https://openranbrasil.org.br/ (Acessado em Maio de 2025).
[11] SWAGGER. Swagger: API Documentation and Design Tools for Teams. Disponível em: https://swagger.io (Acessado em Maio de 2025).
[12] ONOS REST API - https://wiki.onosproject.org/display/ONOS/ (Acessado em Maio 2025).
[13] Postman tool - https://www.postman.com/ (Acessado em Maio 2025).
[14] Burp suite tool - https://portswigger.net/burp (Acessado em Maio 2025).
[15] Wireshark Tool - https://www.wireshark.org/ (Acessado em Maio 2025).