Análise de vulnerabilidades em larga escala nos Roteadores Wi-Fi por meio de Web-Fuzzing

Françoa Taffarel [1], Osmany Barros de Freitas[1] e Lourenço Alves Pereira Junior[1]

[1]Divisão de Ciência da Computação – ITA – São Jose dos Campos, SP – Brazil

{osmany,taffarel,ljr}@ita.br

Abstract. *Wireless routers have advanced to ensure connectivity between IoT devices and the Internet. This evolution has also increased the importance of security analysis, due to the growing targeted or mass cyberattacks by malicious agents. However, a constraint in conducting these analyses on a large scale is the need for access to the physical device. In this article, we present a semi-automated methodology that combines the emulation of router firmware images with web-fuzzing of the web interface using* `Nuclei`*. The initial results were the identification of 6,293 possible flaws, the creation of 27 templates for* `Nuclei` *verification, and validation of CVE-2022-46552.*

Resumo. *Os roteadores sem-fio progrediram para garantir a conectividade entre os dispositivos IoT à Internet. Essa evolução também aumentou a importância de análises de segurança, devido aos crescentes ataques cibernéticos direcionados ou em massa por agentes maliciosos. No entanto, uma restrição na realização dessas análises em larga escala é a necessidade de acesso ao dispositivo físico. Neste artigo, apresentamos uma metodologia semiautomatizada que combina a emulação de imagens de firmware de roteadores com o web-fuzzing da interface web utilizando o* `Nuclei`*. Os resultados iniciais foram a identificação de 6.293 possíveis falhas, a criação de 27 templates do* `Nuclei` *e a validação bem-sucedida do CVE-2022-46552.*

1. Introdução

Com o crescimento de 18% nas conexões de dispositivos *Internet of Things* (IoT) em 2022, alcançando 14,3 bilhões de aparelhos, e a previsão de atingir 16,7 bilhões em 2023 [Analytics 2023], a segurança desses sistemas é uma preocupação constante. Isso é especialmente relevante em casas inteligentes, onde dispositivos IoT auxiliam em tarefas diárias. Os roteadores sem-fio, essenciais para a conectividade deste dispositivos IoT [Freitas et al. 2023], foram alvos, como a *botnet* Mirai ainda ativa em 2023 [ZDI 2023].

Assim, cada vez mais a análise de vulnerabilidades em roteadores sem-fio desempenha um papel essencial na segurança cibernética, possibilitando a adoção de medidas preventivas a fim de minimizar riscos e impactos aos usuários, além de atender a conformidades regulatórias [ANATEL 2023]. Soma-se ainda a crescente preocupação de autoridades públicas almejando a melhor gerência da segurança cibernética do Brasil por meio da criação da Política Nacional de Cibersegurança (PNCiber) [GSI-PR 2023].

Portanto, este artigo descreve uma metodologia semiautomatizada capaz de auxiliar na descoberta de vulnerabilidades na *interface web* por meio da análise dinâmica

de roteadores sem-fio em larga escala. A contribuição proposta pelo artigo é a integração da capacidade de emulação de imagens de *firmware* de roteadores em larga escala por meio do *framework* `FirmAE` com a análise de vulnerabilidades usando a técnica de *web-fuzzing* com *templates* da ferramenta `Nuclei`. Além disso, outra contribuição consiste na criação de *templates* específicos para o contexto de roteadores sem-fio a partir da análise de código-fonte e de vulnerabilidades conhecidas. Esses modelos, construídos com base em Yet Another Markup Language (YAML), têm a finalidade de estabelecer os procedimentos para o envio e processamento das requisições HTTP.

2. Trabalhos Relacionados

De acordo com [Wright et al. 2021], a pesquisa em vulnerabilidades de dispositivos IoT aumentou por meio de grandes investimentos para encontrar falhas, especialmente em roteadores sem-fio. Dessa forma, pesquisadores estão usando análise de vulnerabilidades para melhorar a segurança desses dispositivos. Contudo, enfrentam desafios como a complexidade das ameaças cibernéticas e a falta de padrões de segurança rigorosos.

Nesse contexto, é possível adotar diversas técnicas como a análise estática das imagens de *firmware* dos roteadores que foi utilizada por [Helmke and vom Dorp 2022], [ACI 2018] e [Freitas et al. 2023] para caracterizar, respectivamente, vulnerabilidades de roteadores sem-fio mais comuns no mercado europeu, norte-americano e brasileiro. Além disso, os autores [Zheng et al. 2019], [Redini et al. 2020] utilizam a técnica de análise dinâmica em seus *frameworks* para localizar vulnerabilidades em imagens de *firmware*. No entanto, nenhum desses trabalhos realiza a análise do código-fonte do sistema de arquivos com ênfase na *interface web*. Com a mesma motivação, o trabalho de [Qin et al. 2023] apresentou o `UCRF` que realiza de forma mútua a análise estática no binário do *back-end* da interface web dos roteadores e posteriormente executa dinamicamente um *fuzzer* para descobrir vulnerabilidades. Porém, a proposta limita-se à análise de 10 roteadores físicos e, consequentemente, possui baixa escalabilidade.

Com o objetivo de reduzir a dependência de dispositivos físicos e melhorar a capacidade de análise de vulnerabilidades, a emulação de imagens de *firmware* tornou-se uma solução valiosa para os pesquisadores. Neste contexto, a vanguarda da emulação em larga escala é representada por *frameworks* como `Firmadyne` [Chen et al. 2016], `FirmAE` [Kim et al. 2020] e `ALEmu` [He et al. 2023]. Esses *frameworks* aplicam a técnica de *re-hosting* para executar o *firmware* em um ambiente emulado. No entanto, o [Zhang et al. 2021], que utiliza o framework `FirmAE`, não realiza a análise de código-fonte e seu *web-fuzzer* possui baixa escala por possuir poucas regras de verificação para validar as mais recentes vulnerabilidades.

Assim, o principal desafio deste trabalho em andamento não se limita apenas à integração das técnicas de emulação e análise de vulnerabilidades, mas também abrange a busca por escalabilidade e a capacidade de validar vulnerabilidades. Para isso, será utilizada a ferramenta `Nuclei` que desempenha um papel fundamental neste processo possibilitando a detecção automatizada e em escala de vulnerabilidades em aplicações, infraestruturas e produtos, graças à estrutura modular de seus *templates* de execução de *web-fuzzing* [Solanki 2023]. Esses *templates* serão alimentados com evidências oriundas da análise do código-fonte do sistema de arquivos das imagens de *firmware* possibilitando descobrir novas vulnerabilidades (*zero-days*). Além disso, será possível também a validar

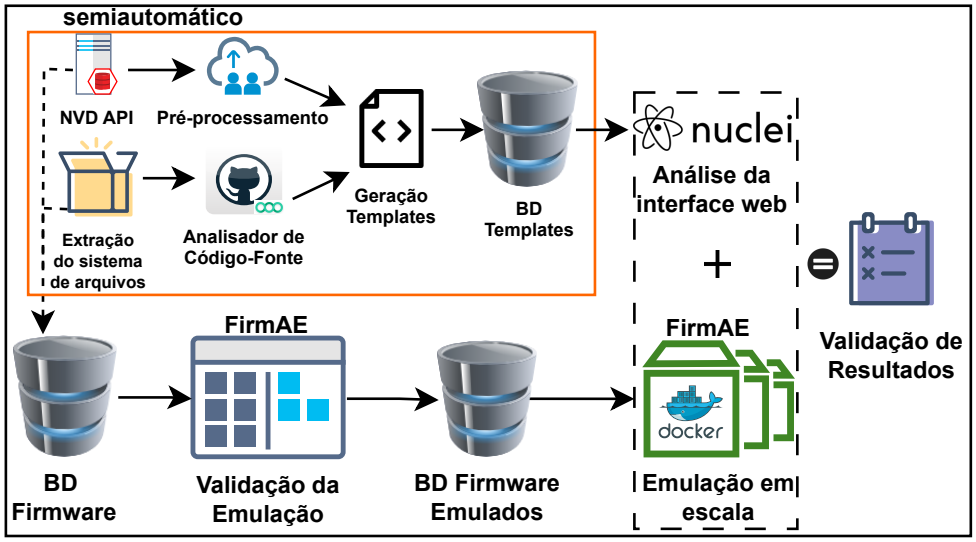

Figura 1. Metodologia

em outros dispositivos vulnerabilidades anteriormente reportadas (*1-day vulnerabilities*) utilizando os seus dados em *templates* específicos para o contexto de roteadores sem-fio.

3. Metodologia para validação de vulnerabilidade em escala

A Figura 1 expõe a metodologia proposta por esta pesquisa, a qual é fundamentada na emulação e análise de vulnerabilidade, em escala, das imagens de *firmware*, presentes em um banco de dados, por meio da técnica da *re-hosting*. As imagens presentes nesse repositório são obtidas com auxílio de *web crawlers* a partir das páginas dos fabricantes, ou de modo manual diretamente do dispositivo físico através de interfaces como JTAG, UART e USB. Nesse contexto, o *framework* `FirmAE` executa as heurísticas para checagem de emulação a fim de obter informações sobre quais imagens são emuláveis.

Em seguida, são aplicadas as funções do `FirmAE` para extração de sistema de arquivos das imagens disponíveis, sendo aproveitadas nessa fase as boas práticas implementadas por [Freitas et al. 2023]. O resultado da extração é enviado para um repositório no `Github`. Posteriormente, esse conteúdo será analisado pela ferramenta `Semgrep`[1], que utiliza uma abordagem de pesquisa de padrões para detectar funções vulneráveis no código-fonte encontrado no conteúdo extraído dos sistemas de arquivos. Os resultados reportados pelo `Semgrep` são considerados indícios de possíveis vulnerabilidades que necessitam de validação. Esses resultados incluem informações como a localização da vulnerabilidade em determinado arquivo, a severidade e os detalhes sobre as regras correspondente a detecção. Após uma análise manual desses achados, são preparados *templates* da ferramenta `Nuclei`, a fim de validar as possíveis falhas em toda a base de dados, proporcionando uma análise abrangente e precisa das vulnerabilidades.

Para avaliar de forma eficiente numerosas imagens de *firmware*, é realizada a paralelização da emulação, utilizando contêineres com a ferramenta `Docker`. Cada imagem é emulada de forma independente em um contêiner, que possui internamente os pacotes e dependências utilizados para emulação. Isso garante uma emulação confiável

[1]https://github.com/returntocorp/semgrep

mesmo em situações com múltiplas interfaces de rede. Por fim, com o *firmware* emulado com sucesso e a interface web acessível, o `Nuclei` executa os *templates* disponíveis na base de dados, utilizando requisições HTTP para realizar a verificação das vulnerabilidades e geração de um relatório final.

3.1. Criação de *templates*

O `Nuclei` utiliza *templates* do tipo YAML, que definem como as solicitações HTTP serão enviadas e processadas. A Código 1 expõe um exemplo de *template* YAML utilizado no `Nuclei`, nota-se que além de ser um formato simples para leitura humana, permite identificar como será o seu processo de execução. Os principais benefícios que são aproveitados por esta proposta são a flexibilidade e a personalização, permitindo a definição de testes e parâmetros de forma clara e concisa. Essa flexibilidade permite a detecção abrangente de vulnerabilidades, abordando diversos tipos de ataques, incluindo injeção de comandos, *cross-site scripting* e vazamento de dados sensíveis.

Os desenvolvedores do `Nuclei`disponibilizam em seu repositório[2] diversos *templates* para distintas categorias. No entanto, os roteadores sem-fio possuem páginas de gerenciamento web com características inerentes aos seus equipamentos, exigindo que os *templates* sejam específicos a fim de otimizar os resultados do *fuzzing*.

Atualmente, esta pesquisa propõe a elaboração manual de *templates* específicos para o contexto de roteadores sem-fio, utilizando como base duas distintas fontes de dados. A primeira consiste em dados de vulnerabilidades previamente identificadas em roteadores. A seleção das vulnerabilidades a serem abordadas nos modelos pode ser conduzida por meio de consultas à API[3] fornecida pelo *National Vulnerability Database* (NVD), que mantém sua base de dados sincronizada com o *Common Vulnerabilities and Exposures* (CVE), administrado pela MITRE. Esse procedimento possibilita a verificação da presença dessas mesmas falhas em outras imagens armazenados no repositório. A segunda fonte para a geração de *templates* se baseia na análise minuciosa dos achados reportadas pelo analisador de código-fonte `Semgrep`, em busca de evidências que permitam extrair os parâmetros e condições para reprodução da requisição HTTP que irá compor o modelo que será utilizado pelo `Nuclei`. Essa estratégia proporciona a descoberta de novas vulnerabilidades.

```
id:CVE-2022-46552
info:
    name: CVE-2022-46552
    author: LabC2DC-ITA
    severity: high
    description: RCE vulnerability via the lan(0)_dhcps_staticlist parameter
    http:
        - raw:
            - |
              POST /HNAP1/ HTTP/1.1
              Content-Type: application/json
              Accept: application/json
              Content-Length: 123
              SOAPACTION: "http://purenetworks.com/HNAP1/SetIpMacBindSettings"
              Connection: close
              {"SetIpMacBindSettings"{"lan_unit":"0","lan(0)_dhcps_staticlist":"1,
              $(id>rce_confirmed),02:42:d6:f9:dc:4e,192.168.0.15"}}
```

Código 1: Uma parte do *template* para validação do CVE-2022-46552.

[2]https://github.com/projectdiscovery/nuclei-templates

[3]https://nvd.nist.gov/vuln/search

Tabela 1. Resultados preliminares

Base de dados inicial	Fabricante	Imagens Emuláveis	Resultados `Semgrep`	*Templates* Criados
846	TP-Link	144	2509	23
967	D-Link	75	3784	4

4. Resultados Preliminares

Todos os experimentos foram conduzidos em um servidor com quatro CPUs Intel® Xeon® E3-1225-v6-3.30GHz, 32 GB de RAM DDR4 e 4 TB de HD. Com sistema operacional `Ubuntu 20.04`, com o `PostgreSQL 12.15` e o `Docker v20`. A base de dados utilizada para testar a metodologia é composta por imagens de *firmware* dos fabricantes `TP-Link` e `D-Link` das quais 1748 são provenientes dos estudos feitos por [Toso and Pereira 2021] e outras 65 da base utilizada por [Freitas et al. 2023]. Em seguida desta base, a ferramenta `Semgrep` analisou o sistema de arquivos extraído pelo *framework* `FirmAE` e reportou, respectivamente, os valores de 2509 e 3784 indícios de vulnerabilidades para os fabricantes `TP-Link` e `D-Link`.

Atualmente, a base de dados de templates é composta por um total de 27 modelos, dos quais 26 são provenientes de vulnerabilidades anteriormente conhecidas e um foi gerado por meio da análise manual dos resultados do `Semgrep`. A ênfase desta análise foi colocada em 369 possíveis falhas consideradas de alta criticidade, particularmente aquelas relacionadas à execução remota de código.

Durante a emulação e análise de vulnerabilidades, os resultados preliminares indicaram que 219 (12%) da imagens de firmware foram emuladas simultaneamente por meio da paralelização por contêineres e a ferramenta Nuclei executando os *templates* gerados obteve êxito em validar a vulnerabilidade de execução remota de comandos com privilégios de super-usuário `root` no roteador D-Link DIR-846. A falha resulta da injeção de código malicioso via requisição POST devido à falta de sanitização no arquivo `SetIpMacBindSettings.php`. O fabricante foi notificado por canal oficial e a falha recebeu a numeração CVE-2022-46552 [Mitre 2023]. Os dados dispostos na Tabela 1 expõem os valores relacionados à aplicação da metodologia proposta.

5. Conclusões

Nesta artigo, foi exposto uma metodologia semiautomatizada para descoberta de vulnerabilidades em larga escala em roteadores sem-fio por meio da análise dinâmica de vulnerabilidades de suas *interfaces web*. Esta metodologia tem como objetivo integrar as capacidades de emulação do *framework* `FirmAE` e o poder de detecção de vulnerabilidades da ferramenta `Nuclei`. Por se tratar de uma pesquisa em andamento[4], como trabalhos futuros, pretende-se apresentar novas informações relevantes como as limitações, os tempos de execução e mais detalhes sobre a configuração e experimentação da metodologia. Soma-se ainda, a expansão do repositório de *templates* para detecção de vulnerabilidades, utilizando tecnologias de Processamento de Linguagem Natural (NLP). Além disso, técnicas de análise de similaridade de código podem ser empregadas para identificar trechos suspeitos comuns em diferentes imagens de *firmware*.

[4]https://github.com/c2dc/screen-sbseg-2023

Agradecimentos

Este trabalho tem apoio financeiro do Programa de Pós-graduação em Aplicações Operacionais (PPGAO/ITA) e da FAPESP processo #2020/09850-0.

Referências

ACI (2018). Securing iot devices: How safe is your wi-fi router? https://www.theamericanconsumer.org/. acessado em 31/05/2023.

Analytics, I. (2023). State of iot 2023: Number of connected iot devices growing 16% to 16.7 billion globally. *IoT Analytics*. Acessado em 25/05/2023.

ANATEL (2023). Ato n° 2436 - requisitos mínimos de segurança cibernética. https://informacoes.anatel.gov.br/legislacao/. Acessado em 07/05/2023.

Chen, D. D., Woo, M., Brumley, D., and Egele, M. (2016). Towards automated dynamic analysis for linux-based embedded firmware. In *NDSS*, volume 1, pages 1–1.

Freitas, O., Corrêa, F., Santos, A., and Junior, L. P. (2023). Caracterização das vulnerabilidades dos roteadores wi-fi no mercado brasileiro. In *Anais do XLI SBRC*, PA, RS, Brasil. SBC.

GSI-PR (2023). Política nacional de cibersegurança (pnciber). https://www.gov.br/gsi/pt-br/composicao/SSIC/dsic/audiencia-publica/. Acessado em 19 de junho de 2023.

He, H., Xiong, X., and Zhao, Y. (2023). Alemu: A framework for application-layer programs emulation of embedded devices. In *2023 4th ICCEA*, pages 406–411.

Helmke, R. and vom Dorp, J. (2022). Towards reliable and scalable linux kernel cve attribution in automated static firmware analyses.

Kim, M., Kim, D., Kim, E., Kim, S., Jang, Y., and Kim, Y. (2020). FirmAE: Towards large-scale emulation of iot firmware for dynamic analysis. In *ACSAC*, Online.

Mitre (2023). CVE-2022-46552. Available from MITRE, CVE-ID CVE-2022-46552. https://cve.mitre.org/cgi-bin/cvename.cgi?name=CVE-2022-46552.

Qin, C. et al. (2023). Ucrf: Static analyzing firmware to generate under-constrained seed for fuzzing soho router. *Computers & Security*, page 103157.

Redini, N., Machiry, A., Wang, R., et al. (2020). Karonte: Detecting insecure multi-binary interactions in embedded firmware. In *2020 IEEE SSP*, pages 1544–1561.

Solanki, H. V. (2023). Limiting attack surface for infrastructure applications using custom yaml templates in nuclei automation. Master's thesis, Dublin, National College of Ireland.

Toso, G. and Pereira, L. A. (2021). Enumeração de sistemas operacionais e serviços de firmwares de roteadores sem-fio. In *Anais Estendidos do XXI SBSeg*, PA, RS, Brasil. SBC.

Wright, C., Moeglein, W. A., Bagchi, S., Kulkarni, M., and Clements, A. A. (2021). Challenges in firmware re-hosting, emulation, and analysis. *ACM Comput. Surv.*, 54(1).

ZDI, Z. D. I. (2023). Tp-link wan-side vulnerability cve-2023-1389 added to the mirai botnet arsenal. *Zero Day Initiative Blog*. acessado em 30/05/2023.

Zhang, H., Lu, K., Zhou, X., et al. (2021). Siotfuzzer: fuzzing web interface in iot firmware via stateful message generation. *Applied Sciences*, 11(7):3120.

Zheng, Y., Davanian, A., Yin, H., et al. (2019). {FIRM-AFL}:{High-Throughput} greybox fuzzing of {IoT} firmware via augmented process emulation. In *USENIX Security 19*.