Journal of Internet Services and Applications, 2024, 15:1, doi: 10.5753/jisa.2024.3834

Uncovering Hidden Risks in IoT devices: A Post-Pandemic National Study of SOHO Wi-Fi Router Security

Osmany Freitas [**Instituto Tecnológico de Aeronática** | ***osmany@ita.br***]
Françoa Taffarel [**Instituto Tecnológico de Aeronática** | ***taffarel@ita.br***]
Aldri Luiz dos Santos [**Universidade Federal de Minas Gerais** | ***aldri@dcc.ufmg.br***]
Lourenço Alves Pereira Junior [**Instituto Tecnológico de Aeronática** | ***ljr@ita.br***]

Divisão de Ciência da Computação, Instituto Tecnológico de Aeronáutica (ITA), São José dos Campos, SP, Brazil.

Received: 9 April 2024 • **Accepted:** 17 July 2024 • **Published:** 16 October 2024

Abstract This study thoroughly analyzes the cybersecurity status of Small Office/Home Office (SOHO) Wi-Fi routers. These routers are crucial but frequently overlooked elements in network infrastructure, particularly in light of the impact of the COVID-19 pandemic on network security. The pandemic has led to shifts in network usage patterns, blurring traditional security perimeters and extending them into private residences, creating additional points of vulnerability in urban environments. Our nationwide research evaluated an extensive dataset of router brands and models currently used at scale. We measured the prevalence of known vulnerabilities, assessed the currency of userspace and kernel software versions, and compared the security robustness of proprietary firmware against open-source alternatives. Our findings reveal a concerning landscape of widespread vulnerabilities and outdated software components, posing latent risks to end-users. The results indicate a predominance of Linux on MIPS and ARM architectures, with an average delay of 5 to 10 years between the release of the kernel and the implementation of the most recent firmware versions. As a result, we observed an average of 1344 and 72 vulnerabilities in the kernel and applications. One significant discovery from our research is that replacing the manufacturer's original firmware with open-source alternatives, such as DD-WRT, OpenWrt, and Tomato, can substantially enhance the security of the software stack. This enhancement results in improvements of up to 97% in the case of binaries and 98.42% in the kernel. Our research helps increase cybersecurity awareness by pinpointing critical home network environment weaknesses and alerting the need for more rigorous security practices in producing and maintaining SOHO routers. This investigation also allowed the report of a new remote code execution vulnerability (disclosed in CVE-2022-46552).

Keywords: Cybersecurity, SOHO Wi-Fi Routers, Network Security, Open-source Firmware, COVID-19, Network Perimeter, Vulnerability Analysis

1 Introduction

In recent years, remote work has become a trend, and it is unlikely that the world will revert to a purely in-person task execution model within organizations [WEFORUM, 2022]. The COVID-19 pandemic has accelerated the transition to remote work environments and distance education, highlighting the importance of understanding and mitigating emerging vulnerabilities in smart cities [Alfonso *et al.*, 2021].

The ease of forming geographically distributed teams has expanded as connectivity services and the ubiquity of Wi-Fi routers [Romana *et al.*, 2020] have made distance a surmountable obstacle. These routers enable connections between Internet of Things (IoT) devices and the access network with Internet service providers. However, organizations relaxed security policies and mechanisms previously based on network assets located within the physical perimeter due to the heterogeneity of remotely located devices. This sudden and massive shift has imposed unprecedented demands on network infrastructures, resulting in relaxed security practices at the edges of networks, which were already vulnerable. Hence, expanding the security perimeter to include many home office devices, often with inadequate default settings and irregular security updates, has broadened attack surfaces and increased the risk of security compromises.

A Palo Alto report on IoT device vulnerabilities released in 2020 [Networks, 2020] indicates that over 50% of devices used worldwide are vulnerable to medium to high severity attacks. With an 18% increase in Internet of Things (IoT) device connections in 2022, reaching 14.3 billion devices, and the forecast to hit 16.7 billion in 2023 [IoT Analytics, 2023], the security of these systems remains a constant concern. Therefore, a quantitative study for measuring the current customer-of-the-shelf products helps increase the situation awareness that common civilians experience in their everyday lives. In particular, the cybersecurity level in the urban context is pertinent, especially when considering smart homes and IoT devices to assist in daily tasks.

Organizations grapple with complex challenges as IoT devices proliferate, including managing a diverse array of devices without standards and levels. Ensuring ongoing monitoring for threats is challenging due to inadequate security features and unencrypted data, navigating poorly designed networks, and striving to meet diverse regulations. Moreover, these issues have roots in insufficient computing resources capable of running comprehensive protection solutions and unqualified users' limited understanding of IoT risks; thus, implementing advanced IoT security measures is critical for the current digital transformation.

Therefore, considering the spread of devices globally, incidents of this nature represent potential economic, geopolitical, and social risks. Detecting vulnerabilities in routers relies on techniques for automatically identifying flaws in the firmware of devices [Feng *et al.*, 2022; Redini *et al.*, 2020; Zheng *et al.*, 2019], analyzing their systems, binaries, and kernel, and employing methodologies such as static analysis, dynamic analysis, symbolic execution, and emulation. Analyzing vulnerabilities is crucial to quantifying risks in the digital ecosystem.

However, only a few studies attempt to measure the threats within the embedded systems of these devices, often restricting themselves to proprietary tools, [ACI, 2018], or solely focusing on kernel analysis, [Helmke and Dorp, 2022]. In our work, we expand on this approach by developing a systematic, educational method that utilizes open-source and free tools. This approach encompasses the analysis of binaries, kernels, and source code, providing a comprehensive understanding of the security landscape.

In order to provide a comprehensive landscape and support new research, this paper examines the range of Wi-Fi routers within the small office/home office category available in the Brazilian market. It delves into the most accessible devices, discussing various models and manufacturers. The initial phase entailed gathering the latest firmware version for each model, while also identifying the operating systems and service binaries, including those responsible for managing the web interface of the devices. Following this, our study proceeds with a static analysis of routers to map existing vulnerabilities and indicators of compromise that could lead to the discovery of vulnerabilities. Utilizing a source code analyzer allowed us to pinpoint numerous code flaws, particularly in web pages. These instances propelled us into the second phase of our research, which involved developing a dynamic approach to automatically validate such occurrences at scale through emulation, thereby providing a comprehensive approach to assessing router security.

The core contribution of this work consists of

1. A National Scope: Our study is based on an extensive, nationwide dataset of SOHO Wi-Fi routers, providing a comprehensive overview of the current state of router security.
2. COVID-19 Context: We contextualize our findings against the backdrop of the pandemic, which has seen a seismic shift in network perimeter definitions, with a novel focus on urban computing environments.
3. Comparative Analysis: By comparing proprietary and open-source firmware, our research offers actionable insights into the security benefits of open-source solutions in urban computing infrastructures.
4. Implications for Policy and Practice: Our findings underscore the need for policy changes and best practices in producing and maintaining SOHO routers, with implications for urban computing stakeholders and end-users.
5. Innovative Methodology: We devised a scalable methodology for large-scale cybersecurity assessments, combining automated scanning with manual analysis to gauge router vulnerabilities accurately, allowing us to report a new vulnerability reported in CVE-2022-46552.

We organized the remaining text as follows: Section 2 discusses related works. Section 3 presents the Methodology used in this research. Next, Section 4 provides a descriptive analysis of the firmware images under study, and in Section 5, the static analysis results are discussed. Section 6 proposes an approach for automatic validation of vulnerabilities in source code. Section 7 presents some considerations and recommendations given the results found, and finally, Section 8 concludes the work.

2 Related works

The digital transformation brought about by IoT carries inherent challenges, primarily in the domains of Privacy and Security. According to Wright *et al.* [2021], research on vulnerabilities in IoT devices has increased through significant investments to find flaws, especially in wireless routers. Thus, researchers are using vulnerability analysis to enhance the security of these devices. However, they face challenges such as the complexity of cyber threats and the lack of strict security standards. Hence, it becomes crucial to assess the current state of the technologies that make up this ecosystem, including identifying existing vulnerabilities.

The study conducted by Fiorenza *et al.* [2020] analyzes the use of HTTPS and vulnerabilities found in the implementations of Brazilian websites. Additionally, the case study by Ponce *et al.* [2022] serves as an example, using OSINT techniques (open-source Intelligence: techniques that collect and analyze the information available from public sources) to enumerate vulnerabilities in Brazilian Internet devices. This approach offers awareness to bolster security for systems affecting a large segment of society.

In a security context, SOHO routers, the most commonly used gateways to connect IoT devices, are highly relevant for vulnerability analysis and the associated risks. This motivation led to the proposal by Helmke and Dorp [2022], which used static analysis of firmware to gain analytical insight into the security of routers sold in the European market, albeit emphasizing the kernel and without revealing the manufacturers analyzed. Taking a similar approach, ACI [2018] provided an analysis focused on equipment in the North American market. Although the study used proprietary tools and an opaque methodology, it offered valuable insights applicable to the Brazilian context.

The work by Qin *et al.* [2023] introduced the `UCRF`, which mutually performs static analysis on the binary of the web interface's back-end of routers and subsequently executes a dynamic fuzzer to discover vulnerabilities. However, the proposal is limited to analyzing ten physical routers, resulting in low scalability. Possessing physical devices for vulnerability analysis can incur a prohibitive cost, especially when it involves a comprehensive repository of devices. Firmware emulation is invaluable for researchers to mitigate the reliance

Table 1. Related Works

Frameworks	Analysis			Web-Fuzzing Models		Scalability
	Static	Web Source Code	Dynamic	Zero-day	1-day	
Firmadyne (2016)			x	x	x	x
Costin (2016)	x	x	x	x		x
ACI (2018)	x					x
Firm-AFL (2019)			x	x		x
FirmAE (2020)			x	x	x	x
Toso (2021)	x					x
UCFR (2022)	x		x	x		
Helmke and Dorp (2022)	x					x
ALEmu (2023)			x			x
Our Work	**x**	**x**	**x**	**x**	**x**	**x**

on physical hardware and improve the efficiency of vulnerability analysis.

In the emulation domain, pioneering frameworks like `Firmadyne` by Chen *et al.* [2016], `FirmAE` by Kim *et al.* [2020], and `ALEmu` by He *et al.* [2023] lead the way in innovation. These frameworks leverage the re-hosting technique to run firmware in an emulated environment. However, although the application of this technique in studies such as those by Toso and Pereira [2021] and Zhang *et al.* [2021], vulnerability verification has been absent against known vulnerabilities and an analysis of the web-based source code used for the device [Costin *et al.*, 2016].

Table 1 provides a detailed comparison of related works in the context of web-fuzzing models, along with their analysis capabilities and scalability. The table highlights the diverse approaches taken by frameworks over the years, such as static analysis, web source code analysis, and dynamic analysis, in addition to their effectiveness in detecting zero-day and 1-day vulnerabilities. The intersection of this information underscores the importance of methods that integrate both static and dynamic analyses, as well as the detection of both zero-day and 1-day vulnerabilities while ensuring scalability. Furthermore, our research encompasses all these features, establishing itself as a comprehensive work that advances the integration approach and can potentially enhance vulnerability detection in router web systems.

3 Static Analysis Methodology

This section details the process applied for conducting the static analysis. Figure 1 represents the steps of the Methodology: (1) obtaining firmware images from manufacturer websites; (2) extracting the kernel and file system content; (3) creating a repository on GitHub; (4) automated code analysis; (5) static analysis with an emphasis on identifying kernel and binary versions, as well as checking for passwords and private keys; and finally, (6) searching for known vulnerabilities through the public Common Vulnerabilities and Exposures (CVE) database.

The scope of this work was SOHO Wi-Fi routers; however, due to the impossibility of accurately identifying the models in use in homes and establishments, as well as the unavailability of firmware from devices provided by internet service providers, we adopt the same strategy used by ACI [2018] and Helmke and Dorp [2022], which relies on models available for purchase in the market. Initially, we used the criterion of Market Share to identify the models considered in the selection of this work, considering the equipment available for sale in the most relevant e-commerce platforms in the country and, therefore, accessible to users. As a source of research, the major electronics e-commerce platforms in Brazil were considered (Figure 2), based on Market Share indicators considering sales revenue, as published by Conversion [2022]. Figure 3 shows the market share of manufacturers in the national market, with a total of 19 brands, where TP-Link, D-Link, and Intelbras hold more than 50% of the models available for sale in the main e-commerces.

3.1 Evidence Collection

The identification of available SOHO Wi-Fi routers in the market was performed with the assistance of web crawlers using the Python library `Scrapy`, which is responsible for collecting data such as model name, manufacturer, sales statistics, ratings, and prices. The data collection for this study was carried out in August and October of 2022, resulting in 158 router models available on market.

Subsequently, the firmware image acquisition could be done through physical extraction of the device; however, this procedure requires hardware access, which would compromise the research scale and make the proposed study unfeasible. For this reason, searching for digital sources to acquire this resource becomes the best strategy for this research. Thus, the firmware images were downloaded exclusively from the official manufacturer websites, avoiding acquiring any images modified by third parties that could contain artifacts or configurations different from those originally found on the equipment. In this process, only the most recent firmware versions for each model were considered, enabling analysis in the safest environment provided by the manufacturers.

3.2 Content extraction

After creating the local database, the images were unpacked, and the kernel and file system content were extracted. The file system contains the necessary files for conducting static analysis, such as binaries, configuration files, scripts, and files related to web pages responsible for managing the de-

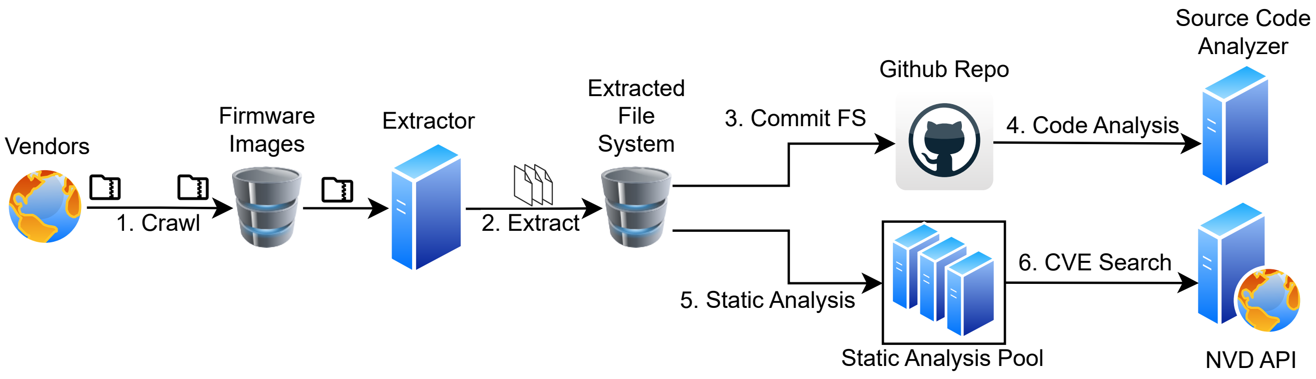


Figure 1. Static methodology for firmware analysis

vices. It is worth noting that the success of this step depends on the firmware construction project since some manufacturers use their formats for producing images. In contrast, in other cases, compression algorithms with obfuscated or encrypted variations are used.

In this research, the FirmAE framework's extractor, which is based on Binwalk, is capable of identifying and extracting files from systems. Binwalk employs the technique of pattern matching in file headers to remove them from the firmware image. During the extraction process, updates were made to the internal packages of the docker image used by the FirmAE extractor, resulting in a 17% improvement in the success rate of file extraction.

3.3 Static Analysis

This work proposes the mapping of known vulnerabilities identified by their CVE IDs. Queries were conducted through an API [1] provided by the National Vulnerability Database (NVD), which is synchronized with MITRE, the organization responsible for overseeing the CVE program. The Common Vulnerability Scoring System (CVSS), linked to CVE, is a system capable of measuring the severity of a vulnerability according to its criticality.

Since this work primarily focused on static analysis without full firmware emulation, detecting the binary version was processed by running the file in user emulation mode - QEMU User Space Emulation. Parameters were inserted to extract the binary version and check for CVEs.

Regarding the kernel, assigning CVEs based solely on the version is insufficient for accurately mapping the vulnerabilities associated with the analyzed devices. Heterogeneity in kernels, architecture-specific or driver-specific flaws, and patches applied directly by manufacturers are some of the challenges of this task. Additionally, there are inherent factors in the CVE repository that lead to various instances of false positives. Situations exist, for example, where a CVE was registered without specifying the vulnerable kernel version. Another recurrent case relates to flaws reported without considering the temporal difference between kernel versions. For instance, direct queries may view version 4.18 from August 12, 2018, posterior to kernel version 4.4.241, which is from October 29, 2020, leading to a mistaken attribution of a flaw to a more recent version.

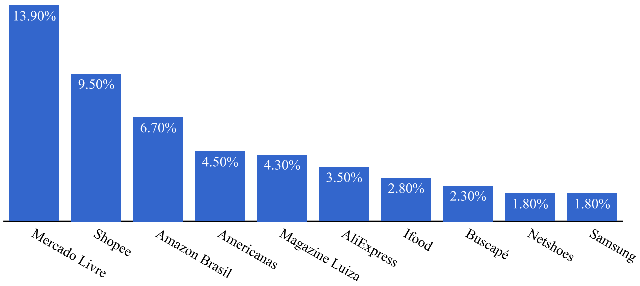


Figure 2. E-commerce's Market Share

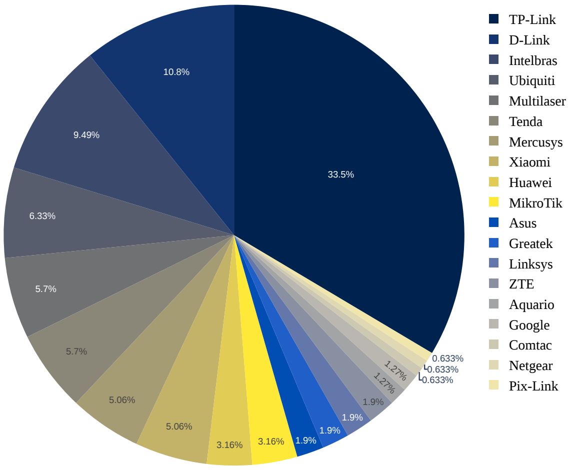


Figure 3. Vendors Market Share

The enumeration of vulnerabilities for the kernel remains an open problem, and some studies propose techniques to obtain more reliable results. Two promising approaches presented by Helmke and Dorp [2022] are file-based matching and commit-based matching. The first considers identifying the files affected by the flaw and checking for their presence in the analyzed kernel. The second evaluates the correction patches in each kernel release, an initiative of the Linux Kernel CVEs project. However, both perspectives still have limitations. This research, on the other hand, focused on three false positive mitigation techniques that consider: (i) the CVE date about the kernel release; (ii) flaws associated with specific architectures; and (iii) discarding results in which the flaw affects third-party software rather than the kernel specifically. Therefore, the Methodology's proposal presents uniform estimates for all samples.

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

3.4 Automatic code analysis

Routers, in general, provide a web interface for configuring and managing their embedded systems. Analyzing the code of these web pages can reveal vulnerabilities that become essential entry points for malicious actors. Therefore, to improve the efficiency of vulnerability scanning in the collected firmware, an approach that utilizes pattern matching to detect potentially vulnerable functions in the code files, primarily web pages, present in the extracted file systems from the obtained images was employed.

This paper used the free version of Semgrep[2] among the available analyzers because it integrates easily with GitHub and offers straightforward reproducibility. It's a lightweight, open-source static analysis tool focused on security analysis. Semgrep defines a set of rules or patterns that developers can use to scan their codebase for potential security vulnerabilities, code quality problems, or coding standard violations, for example. These rules are written in a simple, human-readable syntax. One of the critical advantages of Semgrep is its flexibility and extensibility. Developers can create custom rules tailored to their specific codebase and security requirements. Semgrep employs signatures to find potential flaws. According to Kluban *et al.* [2022], the main advantage of using these rules lies in transparency and adaptability, enabling Semgrep to achieve a 98% match rate for vulnerable functions in the dataset in their study on JavaScript function measurement.

4 Firmware Descriptive Analysis

The descriptive analysis aims to offer an overview of the preprocessing results of the data acquired for the study. It aims to provide a comprehensive understanding of the dataset used in the research by quantifying and characterizing the binaries, kernels, and configurations identified in the samples.

4.1 Official firmware images

A total of 133 firmware images were collected from 13 manufacturers in various formats, including raw or compressed formats (e.g., .gz, .rar, and .zip), resulting in a total data size of 1.4 GB.

However, some manufacturers do not make their firmware available for download. In general, such companies adopt an Over-the-air (OTA) update policy, a remote distribution method directly from the developer, either automatically or with user intervention through an application or device management interface.

Due to some protective measures implemented by developers, it was possible to extract information from 80 samples, achieving a success rate of 60.15%. This success rate covers 11 out of the 13 manufacturers. Among the analyzed models, the MIPS architecture predominates in 83.75% of cases, while ARM represents the remaining 16.25

All processed images use Linux as the embedded operating system, with eight kernel versions. Figure 4a shows the oldest kernel version found is 2.4.20 from 2002, used by the Linksys WRT54G model, while the most recent is 5.6.3 from 2020, present in Mikrotik models. Versions 2.6.x and 3.10.x, which reached the end of support in 2016 and 2017, respectively, represent 71% of the kernel samples present in firmware images.

In contrast to the kernel release date, the firmware's release date (comprising the kernel and all software included in the distribution) shows that the oldest version dates back to 2010, while in the year of this study, 2022, 27 versions were released. Of the processed images, 61.3% were made available between 2020 and 2022. On the other hand, the analysis of the kernel version timeline reveals that the most recent ones are from 2020. The year 2009 is most represented in the dataset due to kernel version 2.6.31 in 21 models. There are also firmware images that use kernels from 2002 and 2006, for example.

Figure 4b overviews the manufacturer's profile, indicating the time difference between the firmware release and the kernel version used. In some images, this difference spans up to 12 years, while in 48 firmware images, at least six years were observed between their release and the kernel version found. While Mikrotik shows an average of 2 years, there are seven other manufacturers with an average exceeding five years, highlighting the obsolescence of the kernel used in their embedded systems. The average time difference across all samples was 2,556 days between the firmware release and the adopted kernel.

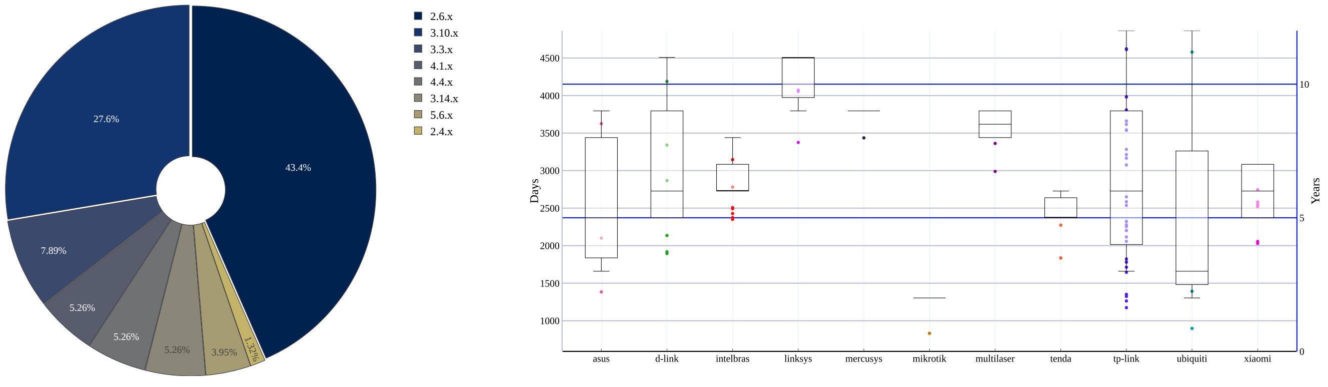
This work also identified security issues with the `/etc/passwd` and `/etc/shadow` files, which are responsible for storing user account credentials in Unix-based systems. In a post-exploitation scenario, attackers often use these files to search for passwords for use in new remote accesses. Throughout the analysis, 14 distinct user hashes, including `admin`, `root`, `user`, and `support`, were collected from 36 firmware images of 4 manufacturers. All these hashes use `md5` as the encryption algorithm, and the most common passwords found were `1234`, `admin`, and `sohoadmin`.

In general, routers support HTTPS for accessing web administration pages, and this functionality is implemented using the SSL/TLS protocol. However, some manufacturers pre-generate private keys in the device firmware. As a result, for an attacker to intercept this communication, they only need to obtain the firmware image and extract the private key, carrying out the well-known ”man-in-the-middle” attack. In 15 samples, 88 private keys were found, corresponding to models from 6 manufacturers.

4.2 Open-Source firmware images

The use of open-source code firmware emerges as an alternative to official versions, giving users greater control over the features available on their devices and the possibility of configuring them more suitably for their use cases. Among the most popular options are DD-WRT, OpenWRT, and Tomato. DD-WRT is a project focused on firmware stability and security for various routers. On the other hand, OpenWRT is an older project and the only one on the list with exclusively non-free binaries. As a result of this strategy, several routers lack full support due to the requirement of non-free drivers

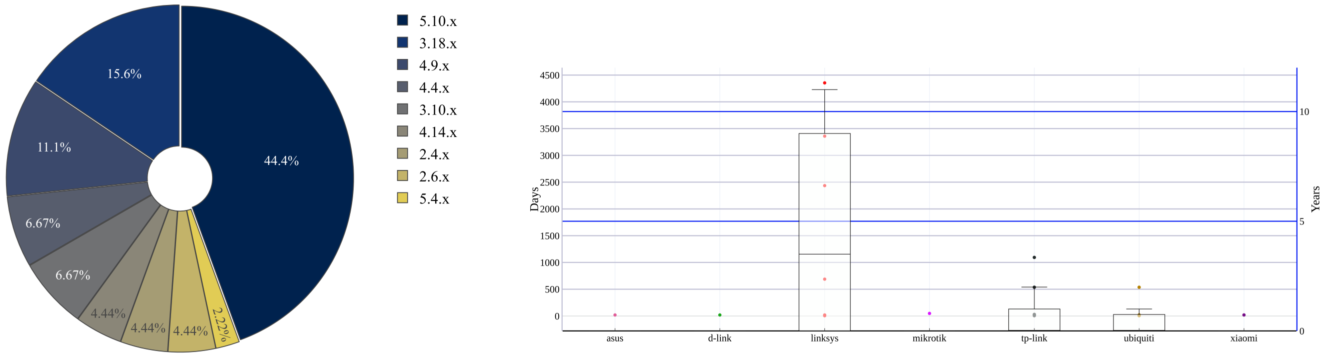
[2] https://github.com/returntocorp/semgrep



(a) Kernel versions

(b) Temporal difference between firmware and kernel release

Figure 4. Official firmware characterization



(a) Kernel versions

(b) Temporal difference between firmware and kernel release

Figure 5. Open-source firmware characterization

for their operation. Tomato offers a more straightforward version and a more modern and user-friendly interface. Still, it has a smaller development community and a limited list of supported models, as it is only compatible with devices featuring Broadcom chipsets.

To compare the results obtained with official firmware, the router models supported by open-source projects were identified. Firmware was obtained for two models compatible with Tomato, 14 with DD-WRT and 30 with OpenWrt, covering 1, 3, and 7 manufacturers, respectively. The models E900 and WRT54G, both from Linksys, were the only ones supported by all three projects. Following the same Methodology applied to official versions, the most recent version of each project was considered. The collected samples totaled 297 MB, with images ranging from 1.7 to 16.5 MB. Regarding architecture, there was only one ARM model, the TP-Link Archer C8.

The images of open-source firmware exhibited versions of the kernel from 9 distinct releases, with a significant emphasis on 3.18.x and 5.10.x, accounting for 60% of the repository, as depicted in Figure 5a. The oldest among them is 2.4.37.9, released on 01/02/2010, found in the firmware of the WRT54G model, while the most recent is 5.10.146, identified in models with version 22.02.3 of OpenWrt, available on 28/09/2022. Among the collected samples, 93% were released between 2020 and 2022.

An analysis of how up-to-date the kernel is concerning the firmware release date revealed that OpenWrt displayed a more consistent profile among the three projects. The most significant difference was observed in version 19.07.10, which used kernel 4.14.275 with a release date of 48 days ago, while in OpenWrt 21.02.0, kernel 5.4.143 had been released just nine days ago. DD-WRT produced results ranging from 5 days to a 7-year difference, while Tomato used kernel versions released between 6 and 9 years from the date of firmware compilation. When looking at the support provided by Tomato for current versions, this is only the profile of 2 models. However, the average time difference between firmware release and the kernel used, considering all samples from the three projects, stood at 411 days, as shown in Figure 5b, which is over six times shorter than the average for official firmware. Unlike official software, hashes of passwords and private keys were irrecoverable, aligning with good information security practices.

5 Static Analysis Results

The static analysis process produced results in three categories: binaries, kernel, and source code. The version identification of binaries and the kernel aimed to enumerate known vulnerabilities through CVEs. The source code analysis, primarily focused on web page content, was conducted using the Semgrep tool, which identified potential security vulnerabilities by comparing all discovered source code against known patterns.

5.1 Static Analysis in Official Firmware

The automated process of collecting versions and searching for CVEs was applied to 5894 binaries, of which 84 were vulnerable, totaling 1474 known vulnerabilities. Of those compromised, 88.1% were released over four years ago. Notably, OpenSSL 0.9.8, made available on 05/07/2005, currently has 96 vulnerabilities, according to the NVD database. It was found in the firmware V1.06B1 of the D-Link DIR-859 model and the firmware v201214 of the TP-Link Archer C20 V5 model.

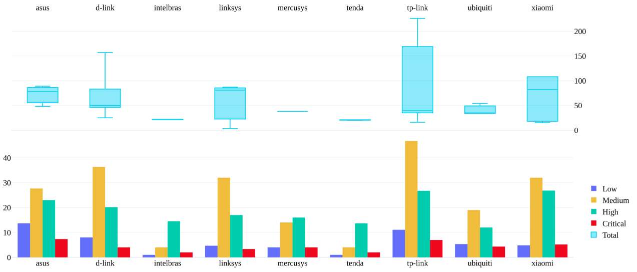


Figure 6. Average binary vulnerabilities by vendor in official firmware

The most common vulnerable binaries were `Busybox`, `Dnsmasq`, `Iptables`, `OpenSSL`, and `OpenVPN`. `Busybox` is a program that combines simplified versions of Unix utilities, commonly used in embedded systems, to provide interactivity and other features. `Dnsmasq` implements essential services such as `DHCP, DNS, and TFTP. Iptables` serves as a configuration interface for packet filtering rules in the Linux kernel. `OpenSSL` and `OpenVPN` provide SSL/TLS implementation and VPN service, respectively. Figure 6 shows the average number of vulnerabilities in binaries grouped by vendors.

The top sections of Figures 6,7, 8, and 9 present box plots constructed with the values of the average vulnerability distributions grouped by manufacturer. In absolute numbers, 12 models had at least ten critical vulnerabilities, all from TP-Link. Regarding average per device, Asus had seven crucial vulnerabilities, followed by TP-Link with 6 of higher severity. In the groups of vulnerabilities classified as High and Medium severity, Xiaomi had an average of 27, and TP-Link had 44 vulnerabilities, respectively. TP-Link topped the list with 9 out of the ten most vulnerable models, including TL-WR941HP, Archer C60 V3, Archer C1200, Archer C8, Deco M5, Deco M4, Deco M9 Plus, Deco E4, Archer C7, and, lastly, D-Link DIR-846.

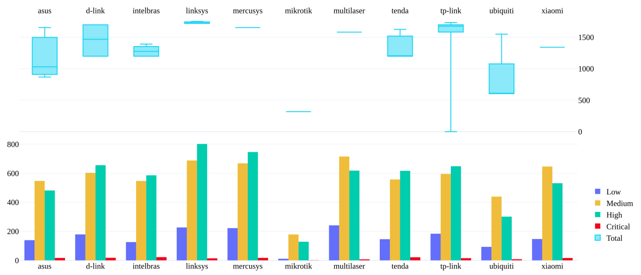


Figure 7. Average kernel vulnerabilities by vendor in official firmware

The update frequency is ineffective due to outdated firmware binaries. For example, TP-Link's Archer C7 V5 has the newest firmware from 08/11/2022. Still, it utilizes old versions of essential binaries like `proftpd` from 2011, `busybox` and kernel 3.3.8 from 2012, and `openssl` from 2015, leading to security risks. Thus, flaws like Heartbleed, Poodle, and Freak Attack were found in multiple models.

The analysis of kernel versions revealed the same pattern seen in binaries, where the firmware is released with old and even discontinued versions of the core of the system. The shortest interval observed was two years in the Ubiquiti UniFi UAP model and the Mikrotik models. However, there are cases, such as the TP-Link EAP110 and EAP115 models, with a 12-year difference. We observed an average of 1344 kernel CVEs per analyzed model. Figure 7 provides an estimate of the average vulnerabilities in the models of each manufacturer.

5.2 Static analysis in Open-source Firmware

Out of 2168 binaries analyzed across 46 images, 15 contained known vulnerabilities. These were associated with versions of `busybox, dnsmasq, iptables`, and `openvpn`, accounting for 142 vulnerabilities. The most affected was `dnsmasq 2.55` in OpenWrt version 10.03.1, compatible with the Linksys `WRT54G`, which had 22 CVEs, including three critical ones. However, this does not represent the typical binary profile in the project, which usually comprises updated versions. For instance, the latest version, 22.03.1, had only one vulnerable binary, `busybox 1.35.0`, and this version had not yet been patched.

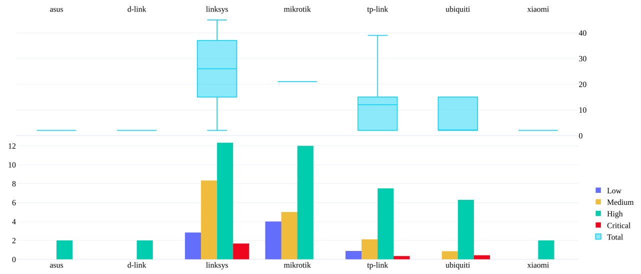


Figure 8. Average binary vulnerabilities in open-source firmware

Open-source firmware outperforms official firmware in security, showing a lower vulnerability average: 97% less for Xiaomi, 96.8% for Asus, 81.5% for Ubiquiti, 66.6% for TP-Link, and 55% for Linksys, particularly in older models, hence limiting the attack surface.

Figure 8 illustrates the average number of vulnerabilities found in binaries from three open-source projects by the manufacturer. While outdated models contribute to higher indices, the results indicate fewer vulnerabilities than in official firmware versions. Critical vulnerabilities related explicitly to *heap* or *buffer overflow* issues potentially leading to remote command execution were found only in Linksys, TP-Link, and Ubiquiti models.

Updating to the latest kernel versions significantly enhances security, substantially reducing vulnerabilities across various models. Specifically, Asus showed in Figure 9 98.42% decrease, D-Link 98.22%, Xiaomi 98.05%, TP-Link 96.23%, Ubiquiti 75.45%, Mikrotik 68.5%, and Linksys 46.5%. This trend is consistent with the binary analysis, indicating that newer kernel versions offer a more secure environment.

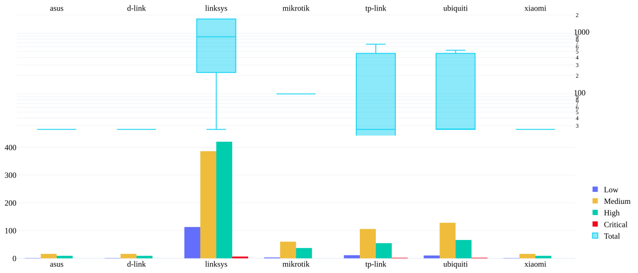


Figure 9. Average kernel vulnerabilities in open-source firmware

5.3 Semgrep Tool Report

The tool reported 3921 and 917 possible flaws in analyzing official and open-source firmware images that need validation. These flaws are categorized according to the Common Weakness Enumeration (CWE), a list developed by MITRE to enumerate common types of vulnerabilities.

Table 2. Top 25 MITRE CWEs found in official firmware images. criticality refers to the Likelihood of exploitation.

Official	Criticality	#	Top 25	*Open-source*	Criticality	#	Top 25
CWE-79	HIGH	1621	2	CWE-79	HIGH	411	2
CWE-20	HIGH	545	4	CWE-20	HIGH	227	4
CWE-22	HIGH	64	8	CWE-798	*unknown*	108	15
CWE-798	*unknown*	168	15				
CWE-94	MEDIUM	20	25				

In this context, Table 2 presents the CWEs in the Top 25 MITRE that were results found in the analysis. The most frequent one was CWE-79, related to Cross-site Scripting, ranking second on the MITRE Ranking, accounting for about 41.3% of the findings identified by Semgrep. Regarding open-source firmware, the same CWE-79 also leads with more findings, representing around 44.8%. CWEs have a scoring system that uses the criticality of a vulnerability in its calculation to identify the probability of successful exploitation by an attacker. Thus, this work used the probability of CWE exploitation to classify the risks associated with the firmware in the database.

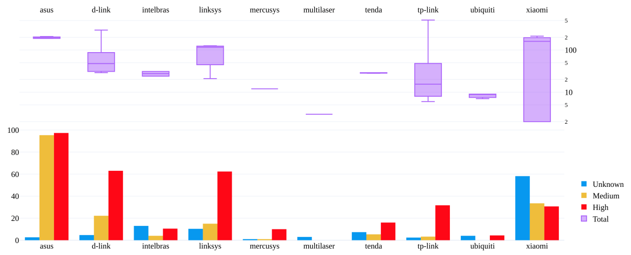


Figure 10. Average CWE in official firmware images

Figure 10 presents the average Criticality of CWEs detected by Semgrep across manufacturers, with Asus showing higher criticality in official firmware. The boxplots within Figure 10 illustrate the potential exploitability of CWEs per manufacturer, suggesting that open-source firmware typically exhibits fewer signs of exploitability than official versions. Figure 11 reflects a consistent criticality across open-source firmware, typically indicating lower exploitability—except for Ubiquiti, where vulnerabilities rise from 8 in official to 19 in open-source.

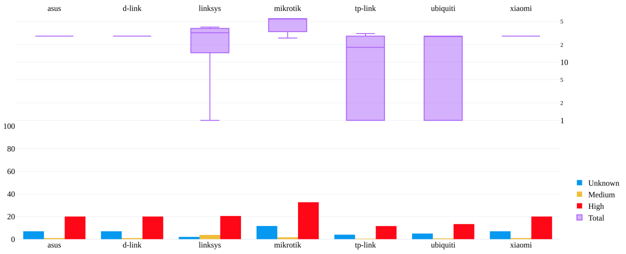


Figure 11. Average CWE in open-source firmware images

6 Vulnerability Validation Proposal

As mentioned in Section 3.4, we used the Semgrep tool to analyze the source code found in the content extracted from firmware images. Semgrep uses patterns to scan the code for potential security vulnerabilities reported as findings. However, these findings are only evidence and require validation to confirm the vulnerability. This entails simulating or executing the code routine associated with the suspicious code snippet to determine if the flaw is exploitable. As a result, we developed a second phase of our research, a dynamic approach now, which includes a method to test and confirm which of these findings truly represent vulnerabilities.

6.1 Methodology for vulnerability validation at scale

Figure 12 displays the Methodology proposed by this research, which is based on the emulation and vulnerability analysis, at scale, of firmware images present in a database through the re-hosting technique. The `FirmAE` framework uses heuristics to check which images can be emulated. Subsequently, the functions of `FirmAE` are applied to extract the file system from the available images, benefiting from the best practices implemented by Freitas *et al.* [2023].

The extraction result is sent to a repository on `GitHub`. Later, this content will be analyzed by the `Semgrep` tool to indicate potential vulnerabilities that require validation. These results include information such as the location of the vulnerability in a particular file, the severity, and the details of the rules corresponding to the detection. After a manual analysis of these findings, `Nuclei` tool templates are prepared to validate possible flaws across the entire database, providing a comprehensive and accurate analysis of the vulnerabilities.

To efficiently evaluate numerous firmware images, the emulation is parallelized using containers with the `Docker` tool. Each image is emulated independently in a container, which internally contains the packages and dependencies used for emulation. This ensures reliable emulation even in situations with multiple network interfaces. Finally, with the firmware successfully emulated and the web interface accessible, `Nuclei` executes the templates available in the database, using HTTP requests to perform the vulnerability verification and generate a final report.

6.2 Templates creation

`Nuclei` utilizes YAML-type templates, which define how HTTP requests will be sent and processed. Code 0 displays

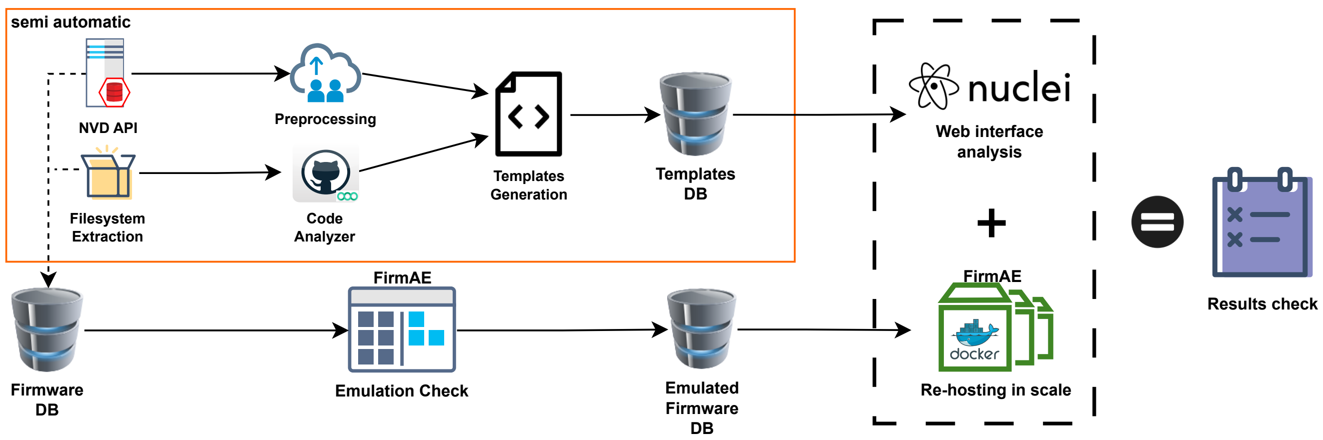


Figure 12. Methodology for dynamic analysis

an example of a YAML template used in Nuclei; it is noticeable that, besides being a simple format for human reading, it allows the identification of how the execution process will be. The main benefits leveraged by this proposal are flexibility and customization, allowing for a clear and concise definition of tests and parameters. This allows for detecting many attack types, like command injection, cross-site scripting, and data leaks.

The developers of Nuclei provide various templates for different categories in their repository[3]. However, wireless routers have web management pages with characteristics inherent to their equipment, requiring that the templates be specific to optimize the fuzzing results.

```
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"}}
```

Code 0: CVE-2022-46552 validation template

Currently, this research proposes the manual elaboration of specific templates for the context of wireless routers, using two distinct data sources as a basis. The first consists of data from vulnerabilities previously identified in routers. The selection of vulnerabilities to be addressed in the models can be conducted through queries to the API[4] provided by the National Vulnerability Database (NVD), which keeps its database synchronized with the Common Vulnerabilities and Exposures (CVE), managed by MITRE. This procedure allows for the verification of the presence of these same flaws in other images stored in the repository. The second source for template generation is based on meticulous analysis of the findings reported by the source code analyzer Semgrep in search of evidence that allows the extraction of parameters and conditions to reproduce the HTTP request that will make up the model to be used by Nuclei. This strategy enables the discovery of new vulnerabilities.

6.3 Results

For dynamic analysis, all experiments were conducted on a server with four Intel® Xeon® E3-1225 v6 3.30GHz CPUs, 32 GB of DDR4 RAM, and 4 TB of HDD. With the operating system Ubuntu 20.04, PostgreSQL 12.15, and Docker v20. The database used in the dynamic analysis test is composed of firmware images from the manufacturers TP-Link and D-Link, of which 1748 are derived from the studies conducted by Toso and Pereira [2021] and another 65 are firmware images collected by this research. Subsequently, the tool Semgrep analyzed the file system extracted by the *framework* FirmAE and reported, respectively, 2509 and 3784 indications of vulnerabilities for the manufacturers TP-Link and D-Link.

Currently, the template database comprises 27 models, of which 26 stem from previously known vulnerabilities, and one was generated through manual analysis of the Semgrep results. The focus of this analysis was placed on 369 possible flaws considered of high criticality, particularly those related to the remote execution of code.

Table 3. Preliminary Results

Database	Vendor	Emulable Images	Semgrep Results	Templates Created
846	TP-Link	144	2509	23
967	D-Link	75	3784	4

In the initial phase of emulation and vulnerability assessment, findings revealed that 219 firmware images, representing 12%, were emulated concurrently via container parallelization. The Nuclei tool effectively executed the templates, confirming a critical remote command execution vulnerability with root access on the D-Link DIR-846 router. Subsequent manual scrutiny of Semgrep flagged data unearthed a novel (zero-day) vulnerability that permits remote command execution with super-user (root) privileges on the D-LINK DIR-846 model.

An authenticated user can inject code on the router's admin page due to a lack of sanitization. The flaw in question is present in the file SetIpMacBindSettings.php, which contains the exec function that receives a variable manipulated by the user. Thus, an attacker can execute arbitrary commands by sending a malicious payload through a POST

[3] https://github.com/projectdiscovery/nuclei-templates
[4] https://nvd.nist.gov/vuln/search

request. Using the proposed methodology for automatic vulnerability validation with Nuclei templates, we could initially validate this flaw in an emulated environment and ultimately confirm its exploitation directly on a physical router. As a best practice, all related information was transmitted to the manufacturer for an update, and CVE-2022-46552 was registered [Mitre, 2023]. Table 3 summarizes the preliminary results of applying the proposed methodology.

7 Discussions

This section delves into advancements in cybersecurity awareness and offers practical recommendations based on our research findings. Vulnerability analysis in wireless routers plays an essential role in cybersecurity, enabling the adoption of preventive measures to minimize risks and impacts on users and comply with regulatory requirements [ANATEL, 2023]. Additionally, there is a growing concern among public authorities aiming to improve cybersecurity management in Brazil by creating the National Cybersecurity Policy (PNCiber) [GSI-PR, 2023].

From the analysis of the results obtained in both the first phase, static, and the second, dynamic, it is evident that there is a need to guide users to strengthen the security of their devices. In this context, a highly recommended practice for users is to update firmware, aiming to reduce the attack surface on their networks. However, it's important to note that while firmware updates are only partially effective in ensuring complete router security, they remain an essential step in bolstering defenses. In other words, manufacturers still use outdated binaries and kernels in the release of new versions. It is essential to regularly check for the availability of new versions for the device and apply the necessary software replacement; some vendors use Over-The-Air (OTA) to update your devices [Peter *et al.*, 2023] remotely. The use of default credentials is still noticeable, and reconfiguring the router's admin page is mandatory to reduce the chances of an attacker gaining immediate access to local network management. We observed that end-of-service-life (EoSL) devices are available.

Therefore, the brand, model, and product lifecycle should be subject to nationwide policies targeting cybersecurity observation. EoSL devices are no longer officially sold by their manufacturers, and the consequence of this is the need for official firmware updates for these models, leaving users with potentially more vulnerable systems. A strategic alternative to minimize the need for the cost of replacing new equipment or the use of highly vulnerable official firmware is the adoption of open-source firmware. Even though these projects need more support for all available models, the list of compatible devices is vast enough for users to find ones that meet their requirements. This work revealed lower obsolescence and vulnerabilities for open-source replacements compared to the official firmware versions. Another advantage is the possibility of updating binaries as the community releases new package versions in repositories, further minimizing vulnerabilities on the device.

8 Conclusion

The study examined Brazil's most popular SOHO routers, highlighting that the pre-installed firmware is less secure than open-source alternatives. Furthermore, as a result of this research, the web management system was found to have flaws. In light of the need to better evaluate the indications provided by the source code analyzer, an approach was proposed to automate this validation process. This methodology for large-scale vulnerability detection in wireless routers was outlined, leveraging FirmAE for emulation and Nuclei for vulnerability detection. With minimal effort, a zero-day vulnerability in the web interface of a D-Link router was discovered (CVE-2022-46552). Future research will focus on exploring other firmware content extraction methods, improving binary version detection, applying code similarity analysis to pinpoint common vulnerabilities in firmware images, conducting a broader global analysis, and enhancing the source code analysis by integrating the use of Natural Language Processing (NLP) for automating Nuclei template generation and increment database.

Declarations

Acknowledgements

This work has financial support from the Graduate Program in Operational Applications — PPGAO/ITA, the São Paulo Research Foundation (FAPESP) under grant #2020/09850-0 and #2022/00741-0, CAPES, and CNPq proc. #313641/2020-0.

Authors' Contributions

Competing interests

The authors declare that they have no competing interests.

Availability of data and materials

Data can be made available upon request.

References

ACI (2018). Securing iot devices: How safe is your wi-fi router? Available at:`https://www.theamericanconsumer.org/`. Last access: December, 2022.

Alfonso, I., Garcés, K., Castro, H., and Cabot, J. (2021). Self-adaptive architectures in iot systems: a systematic literature review. *Journal of Internet Services and Applications*, 12(1):1–28. DOI: 10.1186/s13174-021-00145-8.

ANATEL (2023). Ato n° 2436 - requisitos mínimos de segurança cibernética. Available at:`https://informacoes.anatel.gov.br/legislacao/`. Last access: May, 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. Available at:`https:`

//www.ndss-symposium.org/wp-content/uploads/2017/09/towards-automated-dynamic-analysis-linux-based-embedded-firmware.pdf.

Conversion (2022). E-commerce no brasil: conheça os principais dados, o market share, o crescimento e as principais estatísticas, com atualização mensal! Available at:https://www.conversion.com.br/. Last access: November, 2022.

Costin, A., Zarras, A., and Francillon, A. (2016). Automated dynamic firmware analysis at scale: A case study on embedded web interfaces. In *Proceedings of the 11th ACM on Asia Conference on Computer and Communications Security*, ASIA CCS '16, page 437–448, New York, NY, USA. Association for Computing Machinery. DOI: 10.1145/2897845.2897900.

Feng, X., Zhu, X., Han, Q.-L., Zhou, W., Wen, S., and Xiang, Y. (2022). Detecting vulnerability on iot device firmware: A survey. *IEEE/CAA Journal of Automatica Sinica*, pages 1–17. DOI: 10.1109/JAS.2022.105860.

Fiorenza, M., Kreutz, D., Escarrone, T., and Temp, D. (2020). Uma análise da utilização de https no brasil. In *Anais do XXXVIII Simpósio Brasileiro de Redes de Computadores e Sistemas Distribuídos*, pages 966–979, Porto Alegre, RS, Brasil. SBC. DOI: 10.5753/sbrc.2020.12338.

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. DOI: 10.5753/sbrc.2023.487.

GSI-PR (2023). Política nacional de cibersegurança (pnciber). Available at:https://www.gov.br/gsi/pt-br/composicao/SSIC/dsic/audiencia-publica/. Last access: June, 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. DOI: 10.1109/ICCEA58433.2023.10135383.

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

IoT Analytics (2023). State of iot 2023: Number of connected iot devices growing 16% to 16.7 billion globally. *IoT Analytics*. Available at:https://iot-analytics.com/number-connected-iot-devices/. Last access: May, 2023.

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 *Annual Computer Security Applications Conference (ACSAC)*, Online. DOI: 10.1145/3427228.3427294.

Kluban, M., Mannan, M., and Youssef, A. (2022). On measuring vulnerable javascript functions in the wild. In *Proceedings of the 2022 ACM on Asia Conference on Computer and Communications Security*, ASIA CCS '22, page 917–930, New York, NY, USA. ACM. DOI: 10.1145/3488932.3497769.

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

Networks, P. A. (2020). 2020 unit 42 iot threat report. Available at:https://iotbusinessnews.com/download/white-papers/UNIT42-IoT-Threat-Report.pdf. Last access: December, 2022.

Peter, C., Penning, L., Zimpeck, A., Marques, F., and Yamin, A. (2023). An approach to remote update embedded systems in the internet of things. *Journal of Internet Services and Applications*, 14(1):151–159. DOI: 10.5753/jisa.2023.3078.

Ponce, L., Gimpel, M., Fazzion, E., Ítalo Cunha, Hoepers, C., Steding-Jessen, K., Chaves, M., Guedes, D., and Jr., W. M. (2022). Caracterização escalável de vulnerabilidades de segurança: um estudo de caso na internet brasileira. In *Anais do XL Simpósio Brasileiro de Redes de Computadores e Sistemas Distribuídos*, pages 433–446, Porto Alegre, RS, Brasil. SBC. DOI: 10.5753/sbrc.2022.222341.

Qin, C. *et al.* (2023). Ucrf: Static analyzing firmware to generate under-constrained seed for fuzzing soho router. *Computers & Security*, page 103157. DOI: 10.1016/j.cose.2023.103157.

Redini, N., Machiry, A., Wang, R., Spensky, C., Continella, A., Shoshitaishvili, Y., Kruegel, C., and Vigna, G. (2020). Karonte: Detecting insecure multi-binary interactions in embedded firmware. In *2020 IEEE Symposium on Security and Privacy (SP)*, pages 1544–1561. DOI: 10.1109/SP40000.2020.00036.

Romana, S., Grandhi, J., and Eswari, P. R. L. (2020). Security analysis of soho wi-fi routers. In *2020 International Conference on Software Security and Assurance (ICSSA)*, pages 72–77. DOI: 10.1109/ICSSA51305.2020.00020.

Toso, G. and Pereira, L. A. (2021). Enumeração de sistemas operacionais e serviços de firmwares de roteadores semfio. In *Anais Estendidos do XXI Simpósio Brasileiro em Segurança da Informação e de Sistemas Computacionais*. SBC. DOI: 10.5753/sbseg$_estendido$.2021.17351.

WEFORUM, W. E. F. (2022). Employers are giving workers the work from home days they want. Available at:https://www.weforum.org/. Last access: July, 2023.

Wright, C., Moeglein, W. A., Bagchi, S., Kulkarni, M., and Clements, A. A. (2021). Challenges in firmware rehosting, emulation, and analysis. *ACM Comput. Surv.*, 54(1). DOI: 10.1145/3423167.

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. DOI: 10.3390/app11073120.

Zheng, Y., Davanian, A., Yin, H., Song, C., Zhu, H., and Sun, L. (2019). {FIRM-AFL}:{High-Throughput} greybox fuzzing of {IoT} firmware via augmented process emulation. In *28th USENIX Security Symposium (USENIX Security 19)*, pages 1099–1114. Available at:https://www.usenix.org/conference/usenixsecurity19/presentation/zheng.