International Research Journal of Engineering and Technology (IRJET) e-ISSN: 2395-0056
Volume: 12 Issue: 01 | Jan 2025 www.irjet.net p-ISSN: 2395-0072

Analyzing Router Firmware for Potential Security Weaknesses

Tamilselvan C[1], Harshal Valvi[2], Arun Mohan[3], Dr. Sini S Nair[4]

[1]*Intern,ITG-Dept, National Institute of Electronics & Information Technology,Calicut.*
[2]*Intern,ITG-Dept, National Institute of Electronics & Information Technology,Calicut.*
[3]*Project Associate,ITG-Dept, National Institute of Electronics & Information Technology,Calicut.*
[4]*Scientist,ITG-Dept, National Institute of Electronics & Information Technology,Calicut.*

Abstract - *Routers play a quintessential role in networking and communication to the internet as they serve as the gateway between the user devices and the web. Routers are however subject to risks with their firmware as vulnerabilities within it can be exploited by intruders to break into a secure network. An efficient technique for encapsulating such security risks is static analysis which aims to pinpoint security holes within the code without attempting to run it. This research demonstrates the use of Binwalk, Ghidra or any dismantling tool to pin point common router weaknesses such as compromised security through the use of hardcoded passwords, lack of encryption or neglecting to validate inputs. The paper also delves into other mitigation techniques through secure programming, better user identification, and consistent updates to the firmware. The findings counter potential embedded threats to routers through providing safety and security protocols.*

Key Words*: *Static analysis, Router devices, Firmware Analysis, Firmware Vulnerabilities.

1.INTRODUCTION

A router is a device that enables multiple devices access the internet and also acts as a mediator for data packet transfer between different networks. It also assists in communication by determining the optimal data route, which guarantees safe and quick delivery of data.[1].

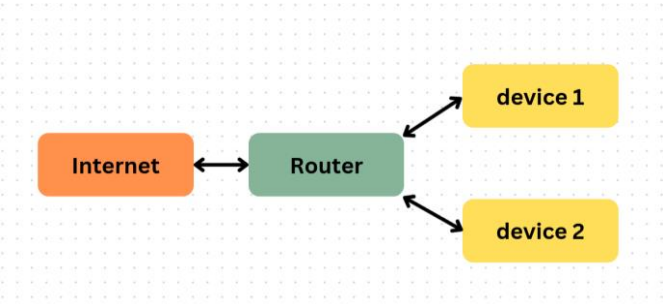

Fig -1: Router Connecting Devices to the Internet

Firmware is the program that is written at the hardware level of a device for performing basic operations. It serves as a mediator between the hardware and high-level software, ensuring the device is functioning as designed[2].Router firmware is the program that is written at the core level of the router device so as to enable effective sending and receiving of data between the two networks. It supplements the router by connecting the device's hardware to its networking functions, allowing the router to securely and effectively perform its tasks[3].Most people tend to disregard the security aspect of router firmware; However, weaknesses in this firmware can endanger the whole network. Such vulnerabilities, when abused, allow an intruder to hijack the administration of the router, changing the paths of the data in the network, retrieving secret information and in some cases, using other equipment within the network to conduct attacks[4].

Static analysis is an approach that deals with security vulnerabilities and exploits by analyzing the source code or binary files of a program without executing it. Think of it as inspecting the architecture of the software to find potential security risks and embedded flaws. It is very different from dynamic analysis which observes an application in runtime. Static analysis detects programming errors such as logic flaws, insecure code errors and buffer overflows which can be misused by the software attackers [5]. In contrast, Static analysis significantly helps in locating software weaknesses before an adversary can leverage them. Usually, router firmware is compiled to binary files and is thus vetting it is somewhat of a complicated task [6]. Static analysis tools such as Ghidra, IDA Pro and Binwalk, however, are capable of identifying hard-coded passwords, insecure communication protocols and buffer overflow vulnerabilities that are preset in a firmware as well as reverse engineer it. Security weaknesses in an application can be located through static analysis which then optimally enables tighter security patches and stronger firmware updates [7].

Router firmware has been seen to have various vulnerabilities such as buffer overflow errors which allow an attacker to run arbitrary code in the device [8]. Most routers however, contain hard coded username and passwords that can easily be exploited if found, and multiple firmware versions contain weak and obsolete communication protocols. These vulnerabilities make routers highly sought targets for those looking to disrupt networks [9].

A couple of solutions can be utilized for router firmware static analysis in order to examine vulnerabilities in router firmware and they include tools that assist in decompiling or reverse engineering router firmware[10]. For instance, there are tools such as Binwalk which are useful for binary

firmware image analysis, while IDA Pro and Ghidra can help in disassembling and compiling code analysis. These tools provide a comprehensive understanding of the firmware's internal structure, enabling researchers to identify vulnerabilities that may have otherwise gone undetected.[14].

This research paper explores the static analysis of security vulnerabilities in router firmware, focusing on the identification and mitigation of common security issues. The study highlights the significance of router firmware security, as vulnerabilities in this software can lead to serious network breaches, unauthorized access, and attacks.

2. LITERATURE REVIEW

Visoottiviseth et al. [11] argue that the security of router firmware, which is increasingly critical due to its dependency on IoT, can be analyzed effectively using Firmaster: an analysis tool for home router firmware.. As of today, many routers are still susceptible to several types of attacks including SQL injection, buffer overflow and poor password management which can result in, data theft, and unauthorized control and use as a bot to partake in DDoS attacks. Even though companies provide firmware updates to deal with the issues, a huge number of people do not bother to keep their devices updated which results in exposure to threats. To maintain strong router firmware security, it is essential to have a system of constant maintenance through regular updating, assessments of weaknesses in the firmware, and configurations that would protect the users and their information against cyber threats.

Feng et al. [12] explain how attack vectors can exploit router firmware vulnerabilities, emphasizing the importance of detecting vulnerabilities in IoT device firmware. One of the Attack vectors involved is a weak web interface that allows manipulation , such as injecting an SQL command or cross-side scripting. Another major source of Attack continues to be altered default logon names and passwords, which attackers can easily circumvent with rudimentary measures. Moreover, inadequate security of the update process can be exploited to implant malicious code during updates. Open ports and strong encryption protocols are also means for an attacker to capture confidential information or take over the router. They stress the need for constant update of router firmware, effective authentication and secure update mechanisms to avoid exploiting the router.

According to Xie et al. [13], modern cybersecurity and privacy are significantly impacted by the integration of various modern devices into the internet, highlighting the importance of vulnerability detection in IoT firmware. Special attention must be delivered towards the detection and alerting against zero day attacks - which in essence, require the rigorous detection of possible vulnerabilities. Analyzing embedded firmware on specialized architectures is an extremely tedious task one that traditional methods fail to accomplish perfectly. These include static analysis which is crucial in the sense that allows for vulnerabilities to be found without executing the firmware. symbolic execution on the other hand examines paths in code to detect logical errors while fuzzing provides a more dynamic approach through testing the mechanisms of a firmware. This is however fused with extensive testing which in simple terms means utilizing both dynamic and static means for a better detection rate. A particular example of this is authentication bypass flaws along with other firmware weaknesses, while also combining static analysis within fuzzing and the hybrid approach. But given the empirical results showing the reality of both known and unknown firmware vulnerabilities, it still has the potential of shifting the paradigm in terms of device security as it pertains to new threats.

Adithyan et al. [14] emphasize that security breaches in router OS were checked using static analysis, which involves examining the router code without executing it, highlighting the techniques used in reverse engineering and backdooring router firmwares. Binwalk is one such tool that is commonly used to extract information from firmware images. This technique possesses the capability of locating embedded files, firmware signatures, and compressed data within a firmware file. Security researchers are then able to review the code for any hardcoded credentials or un-updated libraries. Also, firmware binary binaries can be reverse engineered with tools, such as IDA Pro and Ghidra, and disassembled and decompiled which gives insight into how the code is structured. These tools make it possible for researchers to review the logic and find weaknesses and cases of potentially exploitable malicious code or backdoors. Security experts can then utilize all these static analysis techniques and tools to make sense of how the router firmware operates to detect problems that could go unnoticed till the router goes live.

Catuogno et al. [15] report several mitigation strategies for addressing the risk of vulnerabilities in router firmware, focusing on secure firmware updates and their challenges and solutions. Regularly updating the firmware of a device is among the most efficient ways to reduce the chances of exploitation, for this practice fixes the known weaknesses and secures the device from newly identified weaknesses as well. Manufacturers should set up secure coding guidelines, for instance, input validation and boundary checking, to minimize the chances of having commonplace vulnerabilities such as command injections and buffer overflows. Another effective method for minimising the risk of remote attacks is the deactivation of unutilized services and limiting remote access management to trusted networks or VPNs. Deploying

authentication controls such as biometric systems and multi-factor authentication (MFA) would create further barriers for users who attempt to gain access to resources that they do not have permission to. Furthermore, the owners of routers ought to change the default security settings and configuration options regularly, for instance, by disabling Universal Plug and Play (UPnP) that can automatically open ports which increase the chances of the router being attacked. Deploying firewalls and segmenting the network can help in preventing critical devices from being connected to potentially attacked routers, thereby reducing the possible harm caused by an intruder after exploitation of an RCE vulnerability.

3. METHODOLOGY

The objective of this research is tailored on the evaluation of weaknesses inherent in the router firmware. This is achieved through an independent examination of some selected router firmware samples. The methodology is a combination of the systematic actually gathering of the firmware from given sources and the setting up of an appropriate analysis environment. In addition, a structural method to analyze the firmware is applied to locate any security vulnerabilities. The major activities are selection of the router firmware, preparation of the analysis environment, exploration of security weaknesses, and documentation of the results to ensure a deeper understanding of the weaknesses discovered. The ultimate goal is to find weaknesses and give suggestions on how firmware security procedures can be improved.

3.1 Firmware Selection

Firmware samples were collected from official vendor websites and public repositories, prioritizing popular router models and versions with a history of vulnerabilities.

CVE ID	Router Version	Description
CVE-2024-22853	D-LINK Go-RT-AC750 GORTAC750_A1_FW_v101b03	Hardcoded password allows remote root access via telnet.
CVE-2024-37630	D-Link DIR-605L v2.13B01	hardcoded password vulnerability in/etc/passwd
CVE-2024-0769	D-Link DIR-859 1.06B01	Path traversal vulnerability.
CVE-2024-33111	D-Link DIR-845L router <=v1.01KRb03	/htdocs/webinc/js/bs_c_sms_inbox.php: XSS vulnerability
CVE-2024-33110	D-Link DIR-845L router v1.01KRb03	/getcfg.php: Vulnerable to permission bypass.

3.1 Environment setup

A secure environment was established for firmware analysis, utilizing tools for extraction and analysis along with emulators for testing. Necessary software, including firmware unpacking utilities and debugging tools, was configured. **Binwalk:** Used for extracting and unpacking firmware images to analyze their contents.

Ghidra: A reverse engineering tool to decompile and examine binary code for vulnerabilities.

John the Ripper: A powerful password cracking tool used to test the strength of firmware credentials and identify weak or default passwords.

3.2 Vulnerability Exploration

Vulnerability exploration involved systematically examining the firmware for potential security flaws. This process included identifying weaknesses in the router.

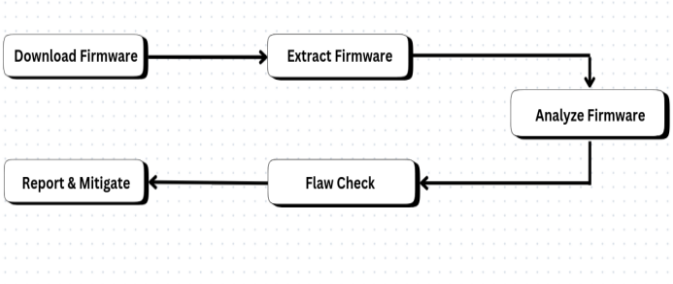

Fig -2: Vulnerability Exploration Flow for Firmware

3.2.1 Download Firmware

The first step is getting the firmware under scrutiny. Device firmware can be obtained either through a vendor's website, a device upgrade page or through actual extraction from the device itself. The source of the firmware must be proper to prevent contamination of the results of the research. This is the very stage from which the further analysis will begin since the quality and version of the firmware is essential to the research.

3.2.2 Extract Firmware

Once the firmware is downloaded, the next step is to extract its contents. This involves unpacking the firmware image to access the underlying file system, binaries, and other critical components. Specialized tools such as binwalk or manual techniques are employed to extract and reconstruct the file system. This step is essential to prepare the firmware for thorough inspection and analysis.

3.2.3 Analyze Firmware

Here, the extracted firmware is thoroughly analyzed. Vulnerabilities of a device through software are identified

using automated tools and also through manual tools. Most of the times hard coded passwords, API keys, libraries, and other weak configurations are used as the areas of concern. Thus the aim of the analysis is to zero-in on the vulnerabilities surgical devices have which can be easily exploited by attackers, jeopardizing device security and user data.

3.2.4 Flaw Check

At this stage, all flaws that were previously revealed are collected in a continuous fashion and provided in detailed reports that include types of defects as well as their degree of seriousness and level of influence. These flaws can be detected with the help of various tools, such as vulnerability scanners, reverse engineering systems, and static analysis applications. The report is also an important document for stakeholders to verify the dangers posed by the firmware.

3.5 Report and Mitigate

The last step relates to rectifying the flaws that are already discussed. In some cases, it is suggested that the for what recall of the firmware, an update of the non-secured libraries, changing encryption algorithms for more detail, or modification of factory settings should be performed. In this practice, after mitigations are performed, the modified firmware is retested; the idea is to check that the attempts to fix the earlier problems do not lead to introducing other problems. This step is crucial to mitigating the risk of the device.

4. EXPERIMENTAL AND RESULT

The Experimental phase analyzed five different router firmware versions to identify vulnerabilities. The analysis uncovered vulnerabilities in all four firmware, with varying levels of severity and impact.

4.1 Hardcoded Password Root Access Via Telnet

The firmware file **GORTAC750_A1_ FW_v101b03.bin** was downloaded from the manufacturer's website and analyzed using Binwalk, a firmware analysis tool. Binwalk enabled the extraction of embedded files and inspection of the firmware structure to uncover potential vulnerabilities.

Fig 3 shows the Binwalk analysis results, decomposing the firmware into its components.

Fig -3: Extract the firmware

The firmware was analyzed using Binwalk, revealing a hardcoded Telnet password in the /squashfs-root/etc/init0.d/ directory.The S80telnetd.sh script contains the command telnetd -l /usr/sbin/login -u Alphanetworks:$image_sign -i br0, which reveals the presence of hardcoded credentials (Alphanetworks) and references $image_sign. This configuration poses significant security risks by enabling unauthorized Telnet access.**Fig. 4** illustrates S80telnetd.sh script containing the vulnerable command.

Fig -4: Script file telnet

The hardcoded password, as shown in **Fig. 5**, is stored in squashfs-root/etc/config/image_sign, posing a security risk.

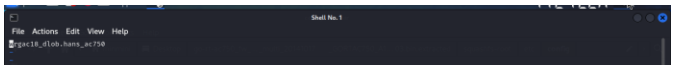

Fig -5: hardcoded password of telnet

4.2 Hardcoded Password Vulnerability

The firmware file DIR605LB2_FW213WWB01.bin was analyzed using Binwalk to extract its contents, as shown in **Fig. 6.**

Fig -6: Firmware extraction

Within the extracted firmware contents, a hardcoded access password was located in the squashfs-root-0/etc/. The hardcoded value root:abSQTPcIskFGc:0:0: root:/:/bin/sh and nobody:x:99:99:Nobody:/: were found. To retrieve the plaintext password, the hashed value was processed using John the Ripper, a robust password-cracking tool. **Fig. 7** illustrates the successful recovery of the plaintext access password, demonstrating the vulnerability posed by hardcoded credentials.

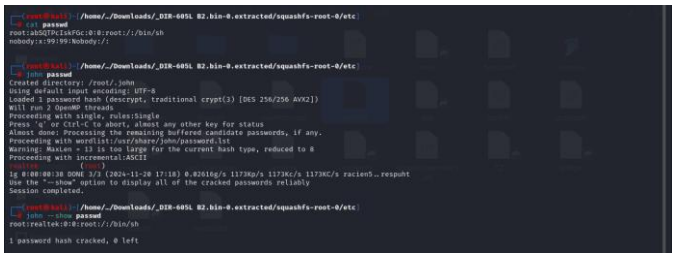

Fig -7: Hardcoded password credential

4.3 Path Traversal Vulnerability

*U*sing Binwalk, the firmware file DIR859Ax_ FW106b01_beta01.bin was analyzed, and various files, including configuration files and executables, were extracted, as shown in **Fig. 8**.

Fig -8: Extraction of firmware

Using the Gidhara reverse engineering tool, the Binwalk analysis of the firmware revealed the presence of a vulnerable file located at /htdcos/webinc/fatlady.php/. This file was identified as a potential security risk due to its inclusion in the firmware and the possibility of exploitation. **Fig. 9** illustrates the path and highlights the location of the vulnerable fatlady.php file.The vulnerability arises from improper validation or insecure coding practices in the script, which could be exploited for unauthorized access or other malicious activities.

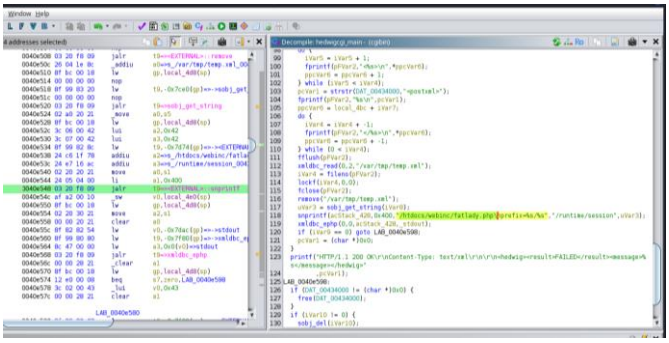

Fig -9: Gidhara reverse engineering tool

After analyzing the firmware using the Gidhara reverse engineering tool, the /squashfs-root/ htdocs/webinc/ fatlady.php file was found to contain a path traversal vulnerability. Specifically, the file includes a $target = "/htdocs/phplib/fatlady/".$service.".php" directive, which allows attackers to manipulate file paths and potentially access restricted files on the system. This vulnerability could lead to unauthorized access or modification of sensitive data.

Fig. 10 illustrates the file and highlights the vulnerable $target = "/htdocs/phplib/fatlady/".$service.".php" path traversal issue.

Fig -10: Path Traversal Vulnerability

4.4 Cross Site Scripting (XSS) Vulnerability

The firmware file DIR845LA1_FW101KRb03.bin was analyzed using Binwalk to extract its contents, as shown in **Fig. 11**. The tool uncovered several embedded files and configurations, which are essential for identifying vulnerabilities within the firmware.

Fig -11: Firmware Extraction

After analyzing the file, it was found that the squashfs-root/htdocs/webinc/js/bsc_sms_inbox.php file contains a vulnerability. Specifically, the line var get_Treturn = '<?if($_GET["Treturn"]=="") echo "0"; else echo $_GET["Treturn"];?>'; is prone to a Cross-Site Scripting (XSS) attack. The vulnerability arises from improper sanitization of user input, allowing an attacker to inject malicious scripts into the Treturn parameter. This could lead to session hijacking or unauthorized actions performed on behalf of the user. **Fig. 12** highlights the vulnerable code in the bsc_sms_inbox.php file.

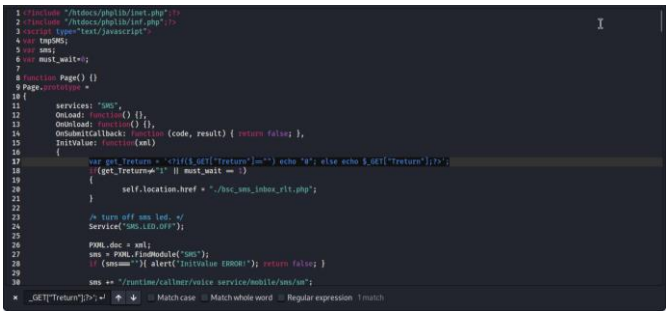

Fig -12: Cross Site Scripting (XSS)

4.5 Permission Bypass Vulnerability

The analysis of the firmware file DIR845LA1_FW101KRb03.bin using Binwalk, as shown in **Fig. 13**, led to the extraction of key components. These included configuration files and executables, which were thoroughly examined for potential vulnerabilities.

Fig -13: Extraction of Firmware

After analyzing the firmware using Binwalk, it was observed that certain files, including /squsn/getcfg.php, contain critical vulnerabilities that can be exploited by attackers. Among these vulnerabilities, a significant permission bypass flaw was identified, which could compromise the overall security of the system by allowing unauthorized access to sensitive areas. The vulnerability arises from the code:

```
if ($_GLOBALS["AUTHORIZED_GROUP"] == "") {  return 0;}

    if ($_GLOBALS["AUTHORIZED_GROUP"] < 0) {  return 0;}return 1;
```

This vulnerability arises from the improper handling and validation of the AUTHORIZED_GROUP variable within the code, which fails to enforce robust access control mechanisms. The flawed logic in the code does not adequately verify the authorization status, enabling attackers to manipulate system logic and potentially gain access to restricted functionalities or sensitive information. As a result, this vulnerability poses a serious threat to the integrity and confidentiality of the system. **Fig. 14** illustrates the vulnerable code in getcfg.php, highlighting the insecure permission validation.

Fig -14: Permission by pass vulnerability

Based on the analysis of five router firmware versions, there are key vulnerabilities that appeared in devices along with their severity levels. Within the first firmware, there was a risk with hardcoded plaintext telnet credentials which posed a threat for unauthorized access. The encrypted credentials within the second firmware could be decrypted due to weak encryption techniques where John the Rip was used. Within the third firmware, there was a PHP script that had not sanitized user inputs and the device users had path traversal vulnerability which allowed them to reach files which were not authorized. The fourth firmware surfer from a cross-site scripting vulnerability which was present in a PHP script. This php script enabled users to inject malicious JavaScript onto the web interface of the router and allowed them to steal data or hijack their sessions. Lastly, the fifth firmware contained scribes of control that had permission circumvention vulnerability enabling users to access features that were supposed to remain restricted. The findings provide important information about the vulnerabilities in authentication, encryption, input validation and access control that may lead to further compromise of system security. In tandem, these concerns bring to light the importance of secure practices during the development of firmware.

5. CONCLUSIONS

The router's firmware was analyzed, and several security flaws were found including compulsory passwords, cross-site scripting, XSS, and permission bypass. Reverse engineering and systematic static analysis of many firmware samples found a large number of coding mistakes, indicating a general trend of negligent coding practices in the firmware industry. The results of their work point toward the necessity of addressing security in the firmware, particularly during its development phase, to fortify the protection against hardcoded credentials, improper access controls, and faulty inputs. Such tools as Ghidra and John the Ripper, which are used as cracks, also demonstrated their significance in research activities in cybersecurity.

To reduce the risks and threats, security must be incorporated into the Development Life Cycle, Regular patches for the vulnerabilities found must be performed, and educating the developers to code securely. For further work, it might be possible to design automatic usage and systems for vulnerability testing in the firmware and recommend ways to improve router security.

6. REFERENCES

[1] *S. Keshav and R. Sharma, "Issues and trends in router design," in IEEE Communications Magazine, vol. 36, no. 5, pp. 144-151, May 1998.*

[2] Chujay Tan, Junita Mohamad-Saleh, Khairu Anuar Mohamed Zain,Zulfiqar Ali Bin Abd Aziz.Review on Firmware- ICISPC 2017: Proceedings of the International Conference on Imaging, Signal Processing and Communication.

[3] Z. Lu, C. Steinmuller and S. Mukhopadhyay, "Towards Formal Verification of a Commercial Wireless Router Firmware," *2013 IEEE 37th Annual Computer Software and Applications Conference*, Kyoto, Japan, 2013, pp. 639-647.

[4] A. Dua, V. Tyagi, N. Patel and B. Mehtre, "IISR: A Secure Router for IoT Networks," *2019 4th International Conference on Information Systems and Computer Networks (ISCON)*, Mathura, India, 2019, pp. 636-643.

[5] D. N. Kleidermacher, "Integrating Static Analysis into a Secure Software Development Process," *2008 IEEE Conference on Technologies for Homeland Security*, Waltham, MA, USA, 2008, pp. 367-371.

[6] K. Cheng *et al.*, "DTaint: Detecting the Taint-Style Vulnerability in Embedded Device Firmware," *2018 48th Annual IEEE/IFIP International Conference on Dependable Systems and Networks (DSN)*, Luxembourg, Luxembourg, 2018, pp. 430-441.

[7] B. Beckman and J. Haile, "Binary Analysis with Architecture and Code Section Detection using Supervised Machine Learning," *2020 IEEE Security and Privacy Workshops (SPW)*, San Francisco, CA, USA, 2020, pp. 152-156.

[8] C. Feng, Z. Alomari, Z. Wang, C. Zhang and U. Zakia, "Systemic Implications of CVE-2023-33246 A Closer Look at Remote Code Exploitation Mechanisms," *2024 15th International Conference on Information and Communication Systems (ICICS)*, Irbid, Jordan, 2024, pp. 1-6.

[9] E. Bertino and N. Islam, "Botnets and Internet of Things Security," in *Computer*, vol. 50, no. 2, pp. 76-79, Feb. 2017.

[10] Ul Haq, S., Singh, Y., Sharma, A. *et al.* A survey on IoT & embedded device firmware security: architecture, extraction techniques, and vulnerability analysis frameworks. *Discov Internet Things* **3**, 17 (2023).

[11] V. Visoottiviseth, P. Jutadhammakorn, N. Pongchanchai and P. Kosolyudhthasarn, "Firmaster: Analysis Tool for Home Router Firmware," *2018 15th International Joint Conference on Computer Science and Software Engineering (JCSSE)*, Nakhonpathom, Thailand, 2018, pp. 1-6.

[12] X. Feng, X. Zhu, Q. -L. Han, W. Zhou, S. Wen and Y. Xiang, "Detecting Vulnerability on IoT Device Firmware: A Survey," in *IEEE/CAA Journal of Automatica Sinica*, vol. 10, no. 1, pp. 25-41, January 2023.

[13] W. Xie, Y. Jiang, Y. Tang, N. Ding and Y. Gao, "Vulnerability Detection in IoT Firmware: A Survey," *2017 IEEE 23rd International Conference on Parallel and Distributed Systems (ICPADS)*, Shenzhen, China, 2017, pp. 769-772.

[14] A. Adithyan, K. Nagendran, R. Chethana, G. Pandy D. and G. Prashanth K., "Reverse Engineering and Backdooring Router Firmwares," *2020 6th International Conference on Advanced Computing and Communication Systems (ICACCS)*, Coimbatore, India, 2020, pp. 189-193.

[15] Catuogno, Luigi, and Clemente Galdi. 2023. "Secure Firmware Update: Challenges and Solutions" *Cryptography* 7, no. 2: 30.