Firmware Update Attacks and Security for IoT Devices

Survey

Meriem Bettayeb[1]
Department of Computer and Electrical Engineering
University of Sharjah
Sharjah, UAE
mera-algeria@hotmail.com

Qassim Nasir[2]
Department of Computer and Electrical Engineering
University of Sharjah
Sharjah, UAE
nasir@sharjah.ac.ae

Manar Abu Talib[3]
Department of Computer Science
University of Sharjah
Sharjah, UAE
mtalib@sharjah.ac.ae

ABSTRACT

The increasing vulnerabilities found in Internet of Things (IoT) devices have raised the need for a solid mechanism of securing the firmware update of these connected objects, since firmware updates are one way to patch vulnerabilities and add security features. This survey analyses the types of attacks that target the firmware update operation in IoT devices and the available secure firmware update methods for IoT devices in the literature between 2004 and 2018. In addition, several popular firmware analysis and vulnerability detection tools are presented. We believe this paper will open the possibility for firmware analysis, attacks and security and therefore help researchers to develop new mechanisms to protect the embedded systems.

CCS CONCEPTS

• Security and privacy → Embedded systems security • Security and privacy → Hardware reverse engineering • Security and privacy → Hardware attacks and countermeasures • Social and professional topics → Centralization / decentralization

KEYWORDS

Firmware Update, Internet of Things (IoT), Security, Attacks

ACM Reference format:

Meriem Bettayeb, Qassim Nasir and Manar Abu Talib. 2019. Firmware Update Attacks and Security for IoT Devices: Survey. In *Proceedings of the ARABWIC 6th Annual International Conference on Arab Women in Computing (ArabWIC2019). ACM, Rabat, Morocco, 6 pages.* https://doi.org/10.1145/3333165.3333169

ArabWIC 2019, March 7–9, 2019, Rabat, Morocco

ACM ISBN 978-1-4503-6089-0/19/03…$15.00
https://doi.org/10.1145/3333165.3333169

1 Introduction

The Internet of Thing (IoT) is a recent revolution in technology that is quickly reshaping our future. This technology allows communication and interaction between small embedded devices and low power devices in general, thus greatly increasing the potential usefulness of such devices. IoT has become the main term to describe billions of devices that are connected to each other via the internet, where it contains everything from wearable devices and smart home appliances to industrial control systems, medical devices and automobiles [1].

However, security remains the greatest obstacle to the growth and widespread use of IoT. These methods, which secure all devices from attacks and interference and can affect user privacy or dangerously threaten public safety, are still not well established. Therefore, there is strong motivation for secure firmware (FW) update implementations that are suitable for the limitations and unique challenges of IoT devices [2].

IoT devices have basic characteristics such as: communication interfaces, network connectivity and System On a Chip (SOC) designs. Communication interfaces include GPIO pins, I2C or SPI, while network connectivity can be achieved wirelessly or via Ethernet. The SOC architecture can be based on ARM or MIPS CPUs. The firmware controlling the behavior of the embedded system should be secure and regularly updated to address any vulnerabilities on the device [3].

firmware filesystems can contain hardcoded and sensitive data. We should start by identifying any vulnerable firmware files of devices currently on the market to improve the security of IoT devices. Firmware analysis tools commonly check for the architecture of a device, it's file system, any buffer overflows and secret information such as passwords, certificate information or database addresses.

In this survey, we investigate the literature for attack types that target FW updates in IoT devices and what counter security measures can be used to better protect the FW update process. Different types of FW update attacks are summarized and secure FW updates are categorized into 'centralized' and 'distributed'. The structure of the paper is as follows: Section 2 introduces the FW update process, its requirements and attack categories. Our

methodology is described in section 3, while Section 4 answers the research questions. The conclusion is presented in section 5.

2 Firmware update process

Firmware update process is a common procedure in IoT devices. However, to implement a secure FW update in embedded devices, a specific method must be considered. In this section, we show the requirements of the update process for a secure FW download and the challenges that can tackle this process.

2.1 Secure Firmware Update Requirements

We will state the requirements for secure firmware update operation and possible attacks on the firmware update before presenting the methods currently available for securing the firmware update process.

The following conditions need to be met in order to install a new FW securely and robustly [4]:

- **Request for update**: The IoT device receives an external FW update request from an authorized entity that handles the FW update operation.
- **Authorized flash driver**: It is required to make sure that the entity who is handling the FW update process is the same entity who is sending the commands, such as loading the flash driver into the RAM or flash data memory.
- **Authentic firmware**: The FW needs to be checked to make sure it is not malicious and it is a compatible.
- **Authorized parts**: The authorized devices and analysis tools ensure the secure FW update process.
- **Rollback mechanism**: An appropriate rollback mechanism must take place to facilitate a robust update in case of a failure. The failure can be a failed update or the detection of malicious or compromised FW.

2.2 Attacks on Firmware Update Categories

The threats to the FW update operation can be divided into the following categories [5]:

- **Reverse engineering**: The ability to extract the FW and analyze it to get sensitive information without the need to access the device.
- **Firmware modification**: The attacker injects extra code in FW so that unauthorized operations can be performed.
- **Obtaining access authorization**: Some devices need authorization for external devices to communicate with them. If an attacker can gain that authorization, he will be able to conduct different types of attacks on the victim devices.
- **Installing unauthorized firmware**: A malicious party or a legal party with malicious purposes will try to install an unauthorized FW (either a malicious or stolen legitimate FW) to the device. Once the unauthorized images are installed on the device, the attacker can potentially conduct additional attacks.
- **Unauthorized device**: An illegal device may pretend to be a legal device and get an authentic copy of the FW. This can possibly result in privacy issues and losses to the device's manufacturer, along with the potential for further malicious activity.

3 Methodology

The methodology in this research is starting by specifying the research questions related to firmware update threats and security for IoT devices then setting search keywords that were taken from the research questions (RQs). After that, we select the resources and the platform search to identify the selection criteria that will narrow down and filter our search as illustrated in Figure 1.

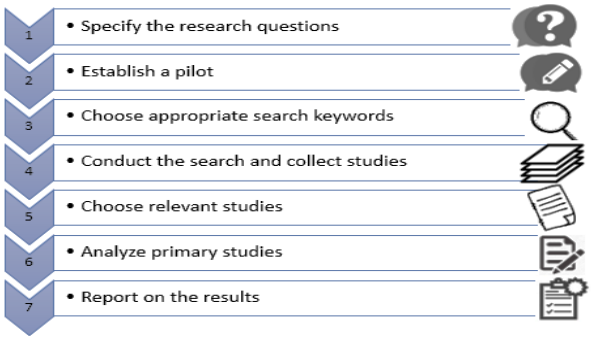

Figure 1: Research methodology

The research questions to be addressed by this study are:

- RQ1. What are the types of attacks on the firmware update process in IoT devices?
- RQ2. What are the security methods used to protect firmware updates of IoT devices?
  - RQ2.1: What are the centralized network solutions for a secure firmware update?
  - RQ2.2: What are the distributed network solutions for a secure firmware update?
- RQ3. What useful firmware analysis tools are available?

Our paper provides knowledge about FW analysis and security by answering the RQs.

3.1 Search Process

We collected data from specific conferences and journals and the study is based on the following search process. The Boolean OR operator and the Boolean AND operator were used with the search words taken from the research questions. The following search string shows all the search terms used in this survey: ("firmware" **OR** "software") **AND** ("update" **OR** "upgrade" **OR** "Over-The-Air" **OR** "OTA") **AND** ("analysis" **OR** "attack" **OR** "tool" **OR** "security" **OR** "vulnerabilities" **OR** "reverse engineering") **AND** ("IoT" **OR** "internet of things" **OR** "embedded devices").

The resources of this review are limited by the following inclusion and exclusion criteria.

3.2 Inclusion Criteria

- Papers from 2004 to 2018.

- Sources from reputable journals: IEEE, ACM, Elsevier, Springer, Science Direct.
- Sources from security related conferences.
- Open source tools.

3.3 Exclusion Criteria

- Online websites.
- Informal literature surveys.

4 Survey Results and Discussion

In this section, we discuss our findings on the predefined research questions.

4.1 Types of attacks on the firmware update in IoT devices

This section surveys most the famous attacks on the embedded devices' FW. Firmware modification attack is one of the most dangerous attacks, as it has been shown to occur on different embedded systems such as: telecommunication infrastructure, SCADA and PLC systems [6], laptop battery controllers, medical devices, network interface cards and automated teller machines (ATM). Many real-life examples show how dangerous the FW modification attack can be.

For instance, the PsycoB0t [7] was the first router botnet that altered the FW of approximately 85,000 home routers and resulted in denial of service attacks. Other than that, some ATM models were also attacked by altering their FWs and causing financial loss [8]. Moreover, Mac laptop battery controllers had FW modification attacks which are done against them [9], while the Lexmark printers [10] were targeted in PostScript based attacks which caused memory inspection and arbitrary modification.

Cui et al. [11] implemented a malware in LaserJet printers that were then used to perform network reconnaissance, data exfiltration and propagation to embedded devices. The authors showed that the signing the FW update does not provide enough protection from the FW modification attacks.

Ling et al. [12] injected malicious code into a smart plug, giving them control over it. The compromised FW opened a reverse back channel to the attacker's server and generated a reverse shell. As a result, the attacker accessed the plug remotely and potentially carried out more attacks.

The work presented in [13] explains how FW updates can be exploited to spread malware even when the FW cannot be reverse engineered. This is done by forcing the devices into a perpetual update cycle. In addition, a tool is presented that incorporates hashing, access control to check whether a given file, usually FW, is genuine or not.

The authors in [14] perform a FW modification attack by injecting code into a previously downloaded FW, while also making sure to edit CRC checks accordingly. However, this attack used the official update channel, where the faulty FW is injected when the device checks for an update. The paper also describes how the presence of an official update can be faked using a TLS proxy, thus allowing the ability to update the FW when wanted.

The authors in [15] use a building-blocked reference model to create a novel IoT attack model encompassing the physical, protocol, data and software attack surfaces present in IoT devices. Additionally, several goals are set for IoT security and attack taxonomy is defined for each attack surface. The paper includes a complete section that deals with the software-based attacks, which are separated into operating system, application and FW-based attacks, where the FW attacks are categorized into: control hijacking, reverse engineering, eavesdropping and malware.

Table 1: Centralized solutions for secure firmware update

Application	Remote firmware update for constrained embedded systems [16]
	The design of monitoring system based on GPRS [17]
	Microcontroller based remote updating system using voice channel of cellular network [18]
	Secure firmware validation and update for consumer devices in home networking [19]
	Wireless monitoring, controlling and firmware upgradation of embedded devices using Wi-Fi [20]
	A smartphone connected software updating framework for IoT devices [21]
	A protocol for secure remote update of run-time partially reconfigurable systems based on FPGA [22]
	Firmware Confidentiality and Rollback for Vehicles [5]
	Secure firmware update over the air in the Internet of Things focusing on flexibility and feasibility proposal for a design [23]
	Software implementation of a secure firmware update solution in an IoT context today [3]
	SCUBA [24]
Remote attestation	Secure code update for embedded devices via proofs of secure erasure [25]
	Secure erasure and code update in legacy sensors [26]
	Efficient proofs of secure erasure [27]
	ASSURED [28]
Companies	Atmel company
	Texas Instrument with their crypto-Bootloader
	The Czech company Jablotron with the open source BigClown project
	Thales company with Hardware Security Modules (HSM)

4.2 Security methods used to protect the firmware update of IoT devices

In this section, we discuss the available solutions for secure FW update. By analyzing the collected resources, we categorized the related work into centralized (server-client) and decentralized solutions (Blockchain).

Table 2: Decentralized solutions for secure firmware update

Approach	Advantages	Disadvantages	Possible Improvements
Firmware verification of embedded devices based on a blockchain [29]	• The effects of known vulnerabilities are minimized as the device is kept updated.	• As the number of IoT devices are increasing, excessive network traffic will occur when downloading the firmware at the same time from a specific FW update server. • Message signature is done by asymmetric cryptographic algorithm which involves hard and complex key management and protection.	• Use peer to peer network model instead of client-server model to transfer the firmware file.
Blockchain-based secure firmware update for embedded devices in an Internet of Things environment [30]	• Minimize attack window time. • Reduce the attacks targeting firmware vulnerabilities on IoT devices.	• When the verification nodes increase, cryptographic key distribution costs increase. • Provided firmware can be corrupted. • Nodes might be malicious. • The broadcast message of the version-check request causes unnecessary network traffic. • Model security has not been formally verified.	• Implement a tracker-less BitTorrent system.
Blockchain-based firmware update framework for Internet-of-Things environment [31]	• Reserve integrity of distributed FW.	• No consideration for the security between the vendor and vendor node. • No consideration for the security of the communication between the IoT device and the gateway.	• Add attestation between the IoT and gateway. • Add a gateway to the Blockchain network.
An architecture to enable secure firmware updates on a distributed trust IoT network using blockchain [32]	• Establishes trust between the nodes of the Blockchain network. • Provides additional security to the IoT devices.	• No algorithms or mathematical models were provided.	• Provide a detailed algorithm and implementation details for the architecture.

4.2.1 Centralized firmware update solutions. We categorized IoT devices based on application, and each type requires a different FW update solution. For instance, Jurkovic and Sruk [16] carried out some experiments remotely by using a Raspberry Pi, which can be accessed from the internet by using the Transport Layer Security (TLS) protocol. It also includes a specific certificate and a key for server and client communication. Another approach to update the FW is to deliver it using a mobile network, which is a schema that contains two parties: a server that is connected to General Packet Radio Services GPRS modules and the device that needs to be updated as mentioned in [17].

A Similar concept is used in [18], except it used two cheap GPRS-enabled phones. The setup here consists of a PC controller and a remote unit containing an 8-bit microprocessor that gets updated. The disadvantage of using a mobile network with GPRS is that the speed declines logarithmically with growing distance from a GSM transmitter.

Another proposed design in [19] provides a secure FW update in home networks by using a FW manager residing in the home gateway and a server. The FW manager collects data and sends it to the server, which retrieves the FW from the vendors and gives it to the FW manager. The manager then uses a hash chain to check the integrity and authenticity of the updated FW. The weaknesses of this design are the wireless connection between the FW manager and the device getting updated, since wireless connections can be attacked in multiple ways.

Zaware and Shinde [20] used Wired Equivalent Privacy (WEP) encryption in order to update the FW using any Wi-Fi based device (PC or mobile), however, using Wi-Fi Protected Access (WPA or WPA2) produces more secure solution to the design, while the work presented in [21] used a smartphone with Bluetooth connectivity to update the FW binaries of the IoT devices. In contrast, the work in [22] uses Field-Programmable Gate Arrays (FPGAs), which have the capability of updating the hardware system remotely via the internet in a safe way. This can prevent cloning, and reverse engineering and protect and verify IP cores keys, although it is a bad choice because of the complexity of the hardware and software design compared to single-chip processor.

Mansor et al. [5] proposed an improved EVITA+ protocol which is a rollback method in case of FW update failure while keeping the confidentiality of both the old and new FW.

The work reported in [23] provides a design proposal for how a secure FW update over the air can be carried out in such a way that the users can decide on the tradeoff between security and speed depending on their specific requirements.

Aside from works which generally rely on encryption, there are several implementations in the literature that secure FW updates through remote attestation. For instance, Seshadri et al. [24]

introduced SCUBA, which uses software attestation, which is based on calculating checksum of code in a specific period of time, and providing authentication by detecting compromised nodes and recovering them, if possible, or blocking the node.

Perito and Tsudik used Proofs of Secure Erasure (PoSE-s) mechanism in [25] to securely update the FW. This technique operates by having the prover performs secure erasure before downloading the new update into a clean slate, where secure erasure means that the verifier will fill the prover's memory with uncompressible randomness, after which the prover replies to the verifier with a screenshot of the new memory contents. This proof ensures that the prover is ready to install the update as it is in a safe state.

[26] and [27] both improved upon the PoSE technique by reducing time, bandwidth and energy requirements. However, the problem with the PoSE method is that it is deployed in one hop environment between the prover and verifier and does not provide any fault tolerance, since it is not possible to revert back to the original FW version in case of a failure. On the other hand, ASSURED [28] supports robust updates and the prover has the ability to isolate suspicious FW, and as such secure erasure is not required.

There are different companies that provide solutions for secure FW updates, such as Texas Instrument with their crypto-Bootloader, the Czech company Jablotron with the open source BigClown project and Thales company with their Hardware Security Modules (HSM) [3]. Table 1 summarizes all the types of centralized architecture of secure FW update.

4.2.2 Distributed firmware update solutions. In IoT devices, attackers seek to alter the software of the embedded devices by using malicious codes in malware. To protect IoT devices from such attacks, some researchers started to integrate Blockchain technology with IoT to perform secure FW updates. Paper [33] shows the applicability of Blockchain technology to guarantee the security of data sent between the nodes of an IoT network. Lee et al. [29] proposed a secure scheme to verify IoT's FW, where the FW update server will update the FW of an IoT device if it did not have the latest version of FW. Otherwise, the Blockchain nodes will verify the integrity of the FW.

In [30], the client-server model is used for FW sharing to a peer-to-peer network architecture, which can be implemented using BitTorrent for FW transmission. Yohan et al. [31] presented a FW update framework that consists of five elements: vendor repository, vendor node, passive node, IoT gateway, and IoT device. The design is based on two operational cases: the process of creating the FW update contract and the process of distributing that FW update by using a PUSH based FW update method, which is a method used to deliver FW update from a vendor to the IoT devices.

Roy et al. [32] propose a secure FW update design in a distributed, trust-based Blockchain IoT network which has four entities: Firmware Author (FA), Author Application Server (AuAS), Vendor Distribution Server (VeDS) and Payload Distribution Server (PDiS). If any node in the Blockchain network is compromised, it will be discarded from the Blockchain network, while a ping is used to check the integrity of each node on the network. The gateway, which is a node in the blockchain that is connected to the IoT devices, will check the signatures signed by the FA and VeDS before downloading the firmware through the peer-to-peer torrent protocol. Table 2 summarizes the available decentralized solutions for secure FW update which are build using blockchain network.

4.3 Firmware analysis tools

In recent years, classical software analysis methodologies have been utilized in FW analysis and vulnerability detection [34]. Most of these tools are published as well as being presented at academic conferences, however, someone cannot depend only on these tools for FW attacks detection. Table 3 shows the most popular open source tools for FW analysis.

Table 3: Firmware analysis tools

Tool	Description
Binwalk [35]	Extract FW to gain access to secret files.
Firmware Reverse Analysis Konsole (FRAK) [36]	automate FW unpacking, analysis, modification and repacking operations.
Firmware Analysis and Comparison Tool (FACT) [37]	Automate FW Security analysis for elements such as: routers, IoT devices, UEFI, webcams and drones.
Firmadyne [38]	Perform emulation and dynamic analysis to FW in embedded devices.
Firmware-Mod-Kit [39]	Facilitate easy teardown and reconstruction of FW for different embedded devices.
Firmwalker [40]	Bash script used for searching through the mounted FW to discover passwords, configuration or script files, and Uniform Resource Locator (URL), and others.
Firmware Analysis Toolkit (FAT) [41]	A toolkit that analyzes embedded device FW to identify vulnerabilities. This was originally built to be used in the "Offensive IoT Exploitation" training conducted by Attify.
BIN2BMP [42]	A python script that visualizes binary data in a graphical form, which can help when working with and reverse engineering FW.
Radare2 [43]	A complete framework for reverse-engineering; consists of a set of small units executed from the command line.
Interactive Disassembler (IDA Pro) [44]	A commercial tool that does automatic code analysis, using cross-references between code segments and information known about the parameters of API calls, and other information.
Firminator [45]	FW vulnerability scanner which provides static and dynamic analysis of FW.
Avatar [46]	Analyze FW using fuzzing and can find hard-coded backdoors.
FIE [47]	compute symbolic states on FW to detect vulnerabilities of MSP430 microcontrollers.
IoT Inspector [48]	Scan the whole FW of IoTs to find out whether they are vulnerable to common attacks.

FirmUp [49]	Static detection of possibly widespread vulnerabilities in FW with high efficiency.
Angr [50]	Perform static and dynamic analysis.
ReFirm Labs: IoT firmware security startup [51]	Provide FW validation in the IoT devices by finding the security flaws of the FW and mitigate them.
Firmware.Re [48]	Analyze most of ROM to detect flaws and all types of embedded malware.

5 Conclusion

This survey explored firmware update exploits and different centralized and decentralized security techniques. Furthermore, it summarized firmware analysis and vulnerability detection tools that are used by many researchers in their work. All papers included here are extracted from the literature between 2004 and 2018. We have accomplished the objectives of this survey and answered the research questions presented in this paper.

ACKNOWLEDGMENTS

We would like to thank the University of Sharjah, Sharjah Islamic Bank, Dubai Electronic Security Center (DESC) and OpenUAE Research and Development Group for funding this research study.

REFERENCES

[1] L. Da Xu, W. He, and S. Li, "Internet of things in industries: A survey," *IEEE Trans. Ind. informatics*, vol. 10, no. 4, pp. 2233–2243, 2014.
[2] S. M. Chowdhury, A. Hossain, and S. Debnath, "Impact of Error Control Code on Characteristic Distance in Wireless Sensor Network," *Wirel. Pers. Commun.*, 2017.
[3] L. Kvarda, P. Hnyk, L. Vojtech, and M. Neruda, "Software implementation of secure firmware update in IoT concept," *Adv. Electr. Electron. Eng.*, vol. 15, no. 4 Special Issue, pp. 626–632, 2017.
[4] S. Schmidt, M. Tausig, M. Hudler, and G. Simhandl, "Secure Firmware Update Over the Air in the Internet of Things Focusing on Flexibility and Feasibility Proposal for a Design," in *Internet of Things Software Update Workshop (IoTSU), At Dublin*, 2016, no. June.
[5] H. Mansor, K. Markantonakis, R. N. Akram, and K. Mayes, "Don't Brick Your Car: Firmware Confidentiality and Rollback for Vehicles," in *Availability, Reliability and Security (ARES), 2015 10th International Conference, IEEE*, 2015, pp. 139–148.
[6] T. Rad, "Vulnerabilities in Correctional Facilities," 2011.
[7] Dronebl, "Network Bluepill," 2008.
[8] B. Jack, "Jackpotting Automated Teller Machines Redux," *Black Hat USA*, 2010.
[9] C. Miller, "Battery firmware hacking," *Black Hat USA*, pp. 3–4, 2011.
[10] A. Costin, "PostScript(um–you've been hacked)," *28C3*, 2011.
[11] A. Cui, M. Costello, and S. J. Stolfo, "When Firmware Modifications Attack : A Case Study of Embedded Exploitation," 2013.
[12] Z. Ling, J. Luo, Y. Xu, C. Gao, K. Wu, and X. Fu, "Security Vulnerabilities of Internet of Things: A Case Study of the Smart Plug System," *IEEE Internet Things J.*, vol. 4, no. 6, pp. 1899–1909, 2017.
[13] C. Hawk, J. Hyland, R. Rupert, M. Colonvega, and S. Hall, "Defending Against Firmware Cyber Attacks on Safety-Critical Systems," *Chiropr. Osteopat.*, vol. 14, no. 1, p. 3, 2006.
[14] J. Rieck, "Attacks on Fitness Trackers Revisited : A Case-Study of Unfit Firmware Security," pp. 33–44, 2016.
[15] H. A. Abdul-ghani, D. Konstantas, and M. Mahyoub, "A Comprehensive IoT Attacks Survey based on a Building-blocked Reference Model," no. April, 2018.
[16] G. Jurković and V. Sruk, "Remote firmware update for constrained embedded systems," *2014 37th Int. Conv. Inf. Commun. Technol. Electron. Microelectron. MIPRO 2014 - Proc.*, no. May, pp. 1019–1023, 2014.
[17] H. Yaling, "The design of monitoring system based on GPRS," pp. 1–4, 2016.
[18] S. Dalai, B. Chatterjee, D. Dey, S. Chakravorti, and K. Bhattacharya, "Microcontroller based remote updating system using voice channel of cellular network," *2015 IEEE Power, Commun. Inf. Technol. Conf.*, pp. 11–16, 2015.
[19] B. C. Choi, S. H. Lee, J. C. Na, and J. H. Lee, "Secure firmware validation and update for consumer devices in home networking," *IEEE Trans. Consum. Electron.*, vol. 62, no. 1, pp. 39–44, 2016.
[20] P. G. Zaware, "Wireless Monitoring , Controlling and Firmware upgradation of embedded devices using Wi-Fi," pp. 2–7, 2014.
[21] S. G. Hong, N. S. Kim, and T. Heo, "A smartphone connected software updating framework for IoT devices," *Proc. Int. Symp. Consum. Electron. ISCE*, vol. 2015–Augus, pp. 2–3, 2015.
[22] T. Thanh, T. H. Vu, N. Van Cuong, and P. N. Nam, "A protocol for secure remote update of run-time partially reconfigurable systems based on FPGA," *2013 Int. Conf. Control. Autom. Inf. Sci. ICCAIS 2013*, no. November 2013, pp. 295–299, 2013.
[23] S. Schmidt, M. Tausig, M. Hudler, and G. Simhandl, "Secure Firmware Update Over the Air in the Internet of Things Focusing on Flexibility and Feasibility," no. August, 2016.
[24] A. Seshadri, M. Luk, A. Perrig, L. van Doorn, and P. Khosla, "SCUBA: Secure Code Update By Attestation in sensor networks," *WiSe '06 Proc. 5th ACM Work. Wirel. Secur.*, 2006.
[25] D. Perito and G. Tsudik, "Secure code update for embedded devices via proofs of secure erasure," in *Springer*, 2010.
[26] G. O. Karame and W. Li, "Secure erasure and code update in legacy sensors," in *Springer*, 2015.
[27] N. Karvelas and A. Kiayias, "Efficient proofs of secure erasure," *SCN, Springer*, 2014.
[28] N. Asokan, T. Nyman, A. Sadeghi, G. Tsudik, and T. U. Darmstadt, "ASSURED : Architecture for Secure Software Update of Realistic Embedded Devices," *IEEE*.
[29] B. L. B, S. Malik, S. Wi, and J. Lee, "Firmware Verification of Embedded Devices Based on a Blockchain," *Springer*, vol. 199, pp. 52–61, 2017.
[30] B. L. J. Lee, "Blockchain-based secure firmware update for embedded devices in an Internet of Things environment," *J. Supercomput. Springer*, 2016.
[31] A. Yohan, N. Lo, and S. Achawapong, "Blockchain-based Firmware Update Framework for Internet-of-Things Environment," in *Conf. Information and Knowledge Engineering*, pp. 151–155.
[32] G. Gabriel, R. Roy, and S. B. R. Kumar, "International Conference on Computer Networks and Communication Technologies," in *International Conference on Computer Networks and Communication Technologies, Springer (to appear)*, 2019, vol. 15, pp. 671–679.
[33] Y. Gupta, R. Shorey, D. Kulkarni, and J. Tew, "The Applicability of Blockchain in the Internet of Things," pp. 561–564.
[34] "Awesome Firmware Security." [Online]. Available: https://github.com/PreOS-Security/awesome-firmware-security/blob/master/README.md.
[35] "Binwalk." [Online]. Available: https://github.com/ReFirmLabs/binwalk.
[36] A. Cui, "Embedded Device Firmware Vulnerability Hunting Using FRAK," *Black Hat USA*, 2012.
[37] "FACT." [Online]. Available: https://github.com/fkie-cad/FACT_core.
[38] D. D. Chen, M. Egele, M. Woo, and D. Brumley, "Towards Automated Dynamic Analysis for Linux-based Embedded Firmware," no. February, pp. 21–24, 2016.
[39] "Firmware Mod Kit." [Online]. Available: https://github.com/rampageX/firmware-mod-kit/wiki.
[40] "Firmwalker." [Online]. Available: https://github.com/craigz28/firmwalker.
[41] Attify, "Firmware Analysis Toolkit." [Online]. Available: https://github.com/attify/firmware-analysis-toolkit.
[42] "BIN2BMP." [Online]. Available: https://sourceforge.net/projects/bin2bmp/files/bin2bmp/.
[43] "Radare2." [Online]. Available: https://github.com/radare/radare2.
[44] "IDA." [Online]. Available: https://hex-rays.com/.
[45] "Firminator." [Online]. Available: https://github.com/misterch0c/firminator_backend.
[46] J. Zaddach, L. Bruno, and D. Balzarotti, "Avatar : A Framework to Support Dynamic Security Analysis of Embedded Systems ' Firmwares."
[47] D. Davidson, T. Ristenpart, and W. Madison, "F IE on Firmware : Finding Vulnerabilities in Embedded Systems using Symbolic Execution."
[48] "Firmware.Re." [Online]. Available: http://firmware.re/.
[49] Y. David and E. Yahav, "FirmUp : Precise Static Detection of Common Vulnerabilities in Firmware," *ASPLOS'18*, 2018.
[50] "Angr." [Online]. Available: https://github.com/angr/angr.
[51] "ReFirm Labs." [Online]. Available: https://www.refirmlabs.com/.