Design and Evaluation of a Method for Over-The-Air Firmware Updates for IoT Devices

Maria Júlia Berriel de Sousa[1], **Juliana Freitag Borin**[1]

[1]Instituto de Computação – Universidade Estadual de Campinas (UNICAMP)
Campinas – SP – Brazil

`m117964@g.unicamp.br, juliana@ic.unicamp.br`

Abstract. *Internet of Things (IoT) has been gaining a lot of attention in the last few years and despite the large amount of investments, there are still many challenges to be overcome within all parts of IoT architecture. One challenge is how to develop a long-lasting application. With regard to devices, one of the main contributors to longevity is the ability to update devices firmware. In practical terms, software and firmware updates over-the-air (OTA) are a crucial part of IoT architecture, along with the capacity to manage them remotely.*
This dissertation explains the IoT architecture with a layered approach and how updates OTA are part of it. As a solution to this problem, the design and implementation of an OTA firmware update method is presented. The method is based on the standard architecture recently proposed by the Internet Engineering Task Force (IETF), in the form of RFC 9019. The proposed solution is evaluated through two testbeds: one with 20 constrained IoT devices connected to an open source IoT cloud platform through a WiFi network and the other one with a set of 75 devices spread in a large geographic area as part of a real world application. Results show that the proposed solution is suitable for constrained devices and has little impact on the network traffic.

1. Problem Statement and Motivation

Internet of Things (IoT) is becoming a more common and more established field of information technologies as it has the potential to be applied in many areas, such as at home, in the industry, in cities, in agriculture and more [Ahmed et al. 2016]. Meanwhile, there is a vast amount of research devoted to IoT and its components, which include different parts of the whole IoT infrastructure: from the smart devices collecting data or interacting with the environment to the service that allows the data to be interacted with or even the method for transmitting that data.

With regard to the IoT devices themselves, a very important feature is the ability to update the software or firmware running on them. Updating allows for correction of bugs (specially security bugs), to add new features to augment functionality, and improve performance. All of these can extend device's useful life, saving costs in replacing them and avoiding the generation of e-waste.

The possible large number of devices deployed and their, sometimes, hard to access locations, make the update process by physically reaching each one of them, a costly and time-consuming task. To avoid that cost, the solution is to use *over-the-air* (OTA) updates, that can be performed remotely, leveraging the network to distribute the new code to the devices. Being such an important part of a device life cycle, software update brings its own challenges. Many works in the literature describe those challenges and possible solutions for them.

These works test the impact of updating the device, but very few consider how the network performance is affected by an update, e.g., if data is lost, how much, if any, downtime. Furthermore, each work that proposes a solution creates its own framework that satisfies the authors priorities, and compatibility with other works is not usually one of those priorities, generating many siloed solutions. Add to those the many others from companies that are compatible only within each company products and are often closed source, incompatibility appears as a large problem for the IoT industry. The straight forward solution to this problem is standardization, and a proposal by a recognized standard agency is a strong candidate for such standard. To advance in the direction of having a standardized firmware update method for devices with resource constraints, the Internet Engineering Task Force (IETF) published the Request for Comment (RFC) 9019, defining a firmware update architecture for IoT [Brendan Moran et al. 2021].

In order to be accepted as a standard, RFC 9019 needs to be implemented and tested beyond proof of concept. This work presents the design and implementation of a solution for OTA firmware update based on the IETF architecture, adding to the body of work to make the RFC a standard. By using the RFC and making the code open source, we aim to ease the adoption of OTA procedures by the industry in a manner that can be easily replicated.

2. Related Work

In the works related to IoT, the scope of each can be generalized, as an entire solution, or it can be focused on a specific part, such as devices or networking. From a general perspective, there are works that explain the IoT architecture through a layered approach. The same approach was used in this dissertation, to understand how each part works and how OTA updates fit into it.

After conducting a study on OTA solutions, a total of 17 IoT-specific options were analyzed and juxtaposed against one another as well as the solution introduced in this research. The comparison was presented in the form of a table (Section 3.1 of the dissertation) that examined various factors, including communication protocols, security measures, node addressing, fault tolerance, power consumption, device management, device hardware, device classification, and experimental scenarios.

The comparison yielded several noteworthy findings, including the fact that most experiments were merely proof of concept, conducted on a single board. Only a small number of authors carried out tests on a testbed featuring multiple devices, with studies [Unterschutz and Turau 2012, Stathopoulos et al. 2003] and our own being among them. Additionally, hardly any of the studies specified their solutions as being applicable to any type of hardware, whereas our solution was successfully implemented and tested on two different devices. It is worth noting that the RFC, feedback mechanism, and IoT platform upon which our solution is based are all hardware-agnostic, which means that it could potentially be implemented on even more resource-constrained devices. Finally, the only works that featured testbeds that spanned a wide area were [Unterschutz and Turau 2012] and our own.

2.1. Updates Over the Air

Given the importance of having devices running up-to-date software, any IoT architecture should have an over the air updating mechanism, furthermore, it should be integrated with the architecture from the beginning.

Updates over the air is the whole process from developing a new firmware version, introducing it to the architecture via the mechanisms available, until it reaches all the pertinent devices, with it installed and correctly running. Despite all it envelops, the focus of this work is in the later part. Considering that, there are two key points to the process: dissemination and activation [Bauwens et al. 2020, Unterschutz and Turau 2012].

Dissemination is how the new firmware is transported across the network to the end devices. One aspect is its topology, which has a big influence in how the dissemination happens. In a mesh network, sending large amounts of data over multiple hops is a big challenge, for this reason mesh networks are not a common topology, tough there is research into a reliable broadcast algorithm for file transmission [Yang et al. 2009]. Another possibility is a star topology, where devices communicate with a gateway, in this case, the gateway can be leveraged to coordinate the update in its own local network. Finally, devices might connect directly to a base station, fetching the update directly from the update server.

Once the update is ready to go, an important architectural decision that influences how it is disseminated is the mode in which it is initiated. The update server might initiate it by notifying the target devices, or the devices might query the server for a new update, or it can be a hybrid of the two modes [Brendan Moran et al. 2021, El Jaouhari and Bouvet 2022, Gupta and van Oorschot 2019].

Meanwhile, activation regards the devices and how they handle receiving an update. Depending on the device's capabilities, an update might happen in an "update mode" where its other functionalities stop until the update is complete, or it may be capable of handling the update in parallel with its other functions. Device behavior might also change depending on the type of update, that is, which part of the firmware is being updated. It can simply replace the entire firmware, or it can be partial [Ruckebusch et al. 2018]. A partial update can be a differential patch, which is usually complicated for constrained devices to handle, or it can be modular, where a specific part of the firmware is replaced, such as the network stack, an application. Replacing the bootloader is also a possibility, but it is less common, since a failure may cause the device to be unable to boot. In some cases, no code is replaced, only configurations, for example, frequency which data is collected, which server to send the data, etc. It also involves fault tolerance and security measures, to ensure the update will not disrupt any device that is working normally.

For an update to happen, first a new version of the firmware needs to be developed and made available to devices. This process can include multiple parties, from the device manufacturer, distributor, original equipment manufacturers (OEM), ISPs [El Jaouhari and Bouvet 2022], to end users, all can have a stake in the creation and distribution of a new firmware. This factor sometimes is the main reason updating IoT devices is not a more common practice, since companies may go out of business or simply decide to stop supporting an older device in favor of a newer one. Most developers do not account for ownership changes when developing a new product, even though there are solutions for this problem [Gupta and van Oorschot 2019].

The update process can also follow one or many policies that are decided by an authorized operator. For instance, how to select devices to be updated, by a specific type of hardware, or geographic location, by client, or any other criteria. Policies might refer to how the update is applied, for example the activation of a new firmware version might be deferred to a specific moment in time, as to not interrupt a service during working hours. They may also refer to the severity of the update, from trivial to urgent [Thantharate et al. 2019].

In order to coordinate, enforce policies and make the update happen, an essential part of the architecture is the device management. It is the element responsible for keeping track of devices information relevant to updating, such as firmware version, in addition to log and report any feedback regarding successes or failures.

Beyond using the device management infrastructure, security mechanisms are also very important, as the consequences of lack of security can be devices compromised at the firmware level.

3. Contributions

At first, a solution for an OTA procedure based on RFC 9019 specifications was designed, alongside a qualitative evaluation of the implemented solution based on the architecture requirements proposed by the IETF (section 4.3 of the dissertation), and a detailed explanation of the process implementation on the devices. In a more technical manner, the work includes:

- the definition of the messages exchanged between the entities involved in the update process: this contribution was presented in the form of a state machine that depicts the decisions a device make during the update process as well as the exchanged messages and in the form of a sequence diagram of messages exchanged between device and IoT platform during the update process (Figures 4.1 and 4.2, respectively, in the dissertation);
- the design and implementation of the firmware for the IoT devices: this contribution is detailed in Algorithms 1 through 6 in Section 4.1.2 of the dissertation and is accompanied by open-source software that includes all of the code[1].

The implemented firmware provides functionality for communicating with an open source IoT cloud platform developed by Konker Labs[2]. All the communication related to the update process is performed using HTTP. The advantage of using HTTP for the update is that it allows for more bytes per packet, making the update process faster when compared to IoT protocols such as MQTT, especially for the firmware download. Additionally, for the purpose of easing the adoption of OTA updates by the industry at large, using a protocol network administrators are familiar with is a facilitator. As an example, HTTP is usually allowed through a company firewall, while MQTT or CoAP might not be.

In order to assess the effectiveness of the OTA procedure, experiments were conducted to evaluate its impact on various metrics. These metrics included the amount of

[1]`https://github.com/lmcad-unicamp/libKonkerESP`

[2]`https://prod.konkerlabs.net/registry`

City	**Number of devices**
São Paulo	32
Guarulhos	7
Osasco	5
Santo André	4
Jundiaí, Barueri, Sorocaba	3
Carapicuiba, São Jose dos Campos, São Bernardo do Campo	2
Itapevi, Diadema, Cajamar, Caraguatatuba, São Roque, Taboão da Serra, Franco da Rocha, Votorantim, Bragança Paulista, Itatiba, Mogi das Cruzes, Mauá	1

Table 1. Distribution of devices by city

data transmitted in both bytes and packets, latency, available memory, CPU utilization and Wi-Fi signal strength at each device, and the duration of each step of the update process. The proposed solution was evaluated in two different scenarios. The first scenario involved a testbed with up to 20 constrained IoT devices in a Local Area Network (LAN) setting, which was designed to test the architecture's applicability by utilizing real devices in a scenario that companies may encounter when implementing IoT solutions. The second scenario involved devices that had already been deployed and were serving as gateways to IoT devices in a Wide Area Network (WAN) setting for a restaurant chain. The list of devices used in each city is detailed in Table 1. In both scenarios, Wi-Fi was used as the wireless access network.

In summary, the main contributions of this work are:

- design and implementation of an OTA software update solution based on the IETF specification;
- open-source library containing the implemented code;
- evaluation of such solution in both a LAN scenario with constrained devices and a real world WAN scenario with non-constrained devices.

3.1. Results

In both experiments scenarios, several measurements of the devices and network behavior were taken. The reasoning for each measurement was explained, as well as the results obtained, in the form of graphs and tables. An analysis of the results presented was also made.

Comparing both the LAN and WAN experiments, the differences between a controlled environment and an uncontrolled one became clear. In the LAN experiment all devices updated without a problem, but the same did not happen in the WAN experiment. It highlights the difficulties of deploying devices on the wild, where there is little or no control over the connection, or even if the device is turned on or off. Even collecting the information proved difficult when devices performed the update, but did not respond when requested for information later.

Despite this, the WAN experiments had a high success rate, and where it failed it was possible to identify the most likely cause with the information provided by the messages (or lack thereof) from the designed process. This data can provide mitigation or solutions for these problems, helping improve the solution that is built on top of the RFC architecture.

Results showed no significant impact of the update process on the network traffic or on the application running on the devices. They also highlight the difficulties of testing and deploying IoT solutions in a professional setting. To the best of the authors' knowledge, this is the first work to implement and test a firmware update solution based on the IETF architecture in multiple IoT devices in two different settings, one of them being a real world scenario.

The disparity of success rates in the LAN and WAN settings highlight the difference between theory and practice. In one setting, there was only one entity making decisions and with total control of all the parts. This is important for testing the viability of a solution, but it is incomplete when considering solutions that will be used by companies and people. The WAN setting was a more approximate demonstration of this scenario, where multiple entities had control over parts of the architecture: the restaurant chain had control over devices' usage, the authors of the firmware design and implementation, and even the platform had more stages to go through before adding a new feature. Though this setting does not emulate multiple decision-making parties, such as devices manufactures and vendors, clients or data analysts.

4. Publications

The following paper was written and published as a result of this research:

- Maria Júlia B. de Sousa, Luis Fernando Gonzales, Erick Ferdinando, Juliana F. Borin. Over-The-Air Firmware Update for IoT Devices on the Wild. *Internet of Things*, 19-C:100578, August 2022. [JCR Impact Factor = 5,711, Qualis A1, Scopus 97th percentile]

5. Conclusion

Updating the software of IoT devices is a critical part of the IoT life cycle in order to keep devices relevant with new features and secure against attackers. This work presented the advantages of maintaining devices up-to-date and the possible consequences of failing to do so. The challenges faced by IoT implementations as a whole were also presented, and a layered architecture was introduced as a way to visualize the elements of an IoT solution and how to tackle said challenges. Each layer purpose was described, alongside the technologies that are used in each one.

A summary of the state-of-the-art of OTA software update solutions and research for IoT was given, where challenges specific to OTA were explored, with special attention to fault tolerance and security issues and solutions. A table comparing the discussed solutions was used to consolidate the main points of research, as well as comparing these solutions with this work. The OTA architecture proposed by the IETF SUIT working group, in the form of RFC 9019, was explained in more details, alongside its accompanying manifest proposal.

One of the main advantages of the RFC architecture was the flexibility to apply it to very different settings and adapt to divergent requirements without becoming incompatible, where more specific solutions might fall short when used in different contexts, requiring re-engineering of the solution. In this case, despite quite different code implementations, both had the same underlying process and flow, without the need to reformulate the architecture.

Given this context, this dissertation proposed an OTA firmware update solution for constrained IoT devices based on the IETF standard architecture. The design of the proposal is described, explained and illustrated by diagrams, a state machine diagram and a sequence diagram for the message flow among the involved entities. At the implementation level, the algorithms for the IoT devices were discussed by presenting the pseudocode.

One form of evaluating the solution was a qualitative comparison with requirements from earlier versions of the RFC, as well as discussing how the devices handle fault detection and correction, and which security requirements are fulfilled.

With the goal of evaluation the proposed solution performance, two different scenarios were tested: one testbed including up to 20 IoT devices in a LAN setting, with four different scenarios, and another with 75 devices deployed in different cities as part of a real application monitoring restaurants, with two different scenarios. All the developed code has an open source license and can be found on GitHub. Results show that it is feasible to implement this solution in real world scenarios and that it does not interfere with the application running on the infrastructure. The success rate of the LAN experiment was 100% and the WAN, an average of 75%, due to connection conditions.

The decision of deploying an update happens before anything in the scope of this work, but it does not mean the results do not provide useful insights about how to facilitate the implementation of an OTA process. Having a library that is compatible with a platform and can be used by different devices facilitate implementation, and may even be a selling point for a platform provider.

In summary, the contributions were the design and implementation of an OTA update solution based on the RFC 9019 specification, and its evaluation in two scenarios, with constrained and non-constrained devices. The code for both implementations is also available as an open-source library.

Acknowledgements

The authors would like to express their gratitude to Konker (Inmetrics S.A., grant #34-P-04058-2018) for providing financial and equipment support and the Brazilian National Council for Scientific and Technological Development - CNPq (grant #308840/2020-8) for the financial support. Additionally, they would like to extend a special thank you to researchers Luis Fernando Gomez Gonzalez and Erick Mascagni Ferdinando for their contributions to the technical discussions and for their assistance in conducting the experiments.

References

[Ahmed et al. 2016] Ahmed, E., Yaqoob, I., Gani, A., Imran, M., and Guizani, M. (2016). Internet-of-things-based smart environments: state of the art, taxonomy, and open research challenges. *IEEE Wireless Communications*, 23(5):10–16.

[Bauwens et al. 2020] Bauwens, J., Ruckebusch, P., Giannoulis, S., Moerman, I., and Poorter, E. D. (2020). Over-the-Air Software Updates in the Internet of Things: An Overview of Key Principles. *IEEE Communications Magazine*, 58(2):35–41.

[Brendan Moran et al. 2021] Brendan Moran, Brown, D., Meriac, M., and Tschofenig, H. (2021). A Firmware Update Architecture for Internet of Things. Request for comment 9019, RFC Editor.

[El Jaouhari and Bouvet 2022] El Jaouhari, S. and Bouvet, E. (2022). Secure firmware Over-The-Air updates for IoT: Survey, challenges, and discussions. *Internet of Things*, 18:100508.

[Gupta and van Oorschot 2019] Gupta, H. and van Oorschot, P. C. (2019). Onboarding and Software Update Architecture for IoT Devices. *2019 17th International Conference on Privacy, Security and Trust (PST)*, pages 1–11.

[Ruckebusch et al. 2018] Ruckebusch, P., Giannoulis, S., Moerman, I., Hoebeke, J., and De Poorter, E. (2018). Modelling the energy consumption for over-the-air software updates in LPWAN networks: SigFox, LoRa and IEEE 802.15.4g. *Internet of Things*, 3-4:104–119.

[Stathopoulos et al. 2003] Stathopoulos, T., Heidemann, J., and Estrin, D. (2003). A Remote Code Update Mechanism for Wireless Sensor Networks:. Technical report, Defense Technical Information Center, Fort Belvoir, VA.

[Thantharate et al. 2019] Thantharate, A., Beard, C., and Kankariya, P. (2019). CoAP and MQTT Based Models to Deliver Software and Security Updates to IoT Devices over the Air. In *2019 International Conference on Internet of Things (iThings) and IEEE Green Computing and Communications (GreenCom) and IEEE Cyber, Physical and Social Computing (CPSCom) and IEEE Smart Data (SmartData)*, pages 1065–1070, Atlanta, GA, USA. IEEE.

[Unterschutz and Turau 2012] Unterschutz, S. and Turau, V. (2012). Fail-safe over-the-air programming and error recovery in wireless networks. *Proceedings of the 10th International Workshop on Intelligent Solutions in Embedded Systems*, pages 27–32.

[Yang et al. 2009] Yang, Z., Li, M., and Lou, W. (2009). R-Code: Network Coding Based Reliable Broadcast in Wireless Mesh Networks with Unreliable Links. In *GLOBECOM 2009 - 2009 IEEE Global Telecommunications Conference*, pages 1–6, Honolulu, Hawaii. IEEE.