D6.2

Security Enhanced OS Software

Project number	830892
Project acronym	SPARTA
Project title	Strategic programs for advanced research and technology in Europe
Start date of the project	1st February, 2019
Duration	36 months
Programme	H2020-SU-ICT-2018-2020

Deliverable type	Other
Deliverable reference number	SU-ICT-03-830892 / D6.2 / V1.0
Work package contributing to the deliverable	WP6
Due date	January 2021 - M24
Actual submission date	29th January, 2021

Responsible organisation	INRIA
Editor	Emmanuel Baccelli
Dissemination level	PU
Revision	V1.0

Abstract	This deliverable showcases our security mechanisms enhancements to the open source IoT operating system RIOT.
Keywords	Embedded, Low-power Network Protocols, Operating System Software, Security

This project has received funding from the European Union's Horizon 2020 research and innovation programme under grant agreement No 830892.

Editor

Emmanuel Baccelli (INRIA)

Contributors (ordered according to beneficiary numbers)

Francisco Molina, Koen Zandberg, Alexandre Abadie, Timothy Claeys, Malisa Vucinic, Thomas Watteyne (INRIA)

Gabriele Restuccia, Giuseppe Bianchi (CNIT)

Reviewers (ordered according to beneficiary numbers)

Philippe Massonet (CETIC)

Alexandra Kobekova (UBO)

Executive Summary

In this deliverable, we showcase the output of our work as open source implementations and integrated with mainstream RIOT, which we forked in a GitHub repository which we make available online. The focus of this fork of RIOT (based on Release 2020.10) is to:

- Highlight some of **our on-going contributions upstream to RIOT** master branch: hundreds of commits since the beginning of H2020 Sparta including, but not limited to, SUIT-compliant secure OS software update support, and support for secure 6TiSCH networking.
- Offer a sneak peek at **complementary security mechanisms we developed** but that are not (yet) integrated in RIOT master branch, and how this platform is used in the context of Sparta, e.g. a prototype of minimal virtual machines for software module hosting and isolation.

Table of Contents

Chapter 1 Introduction

RIOT is an open source general-purpose operating system for low-power IoT devices, which we co-founded, contribute continuously to. For more information on RIOT, see the master branch, and our prior publication.

We here showcase the output of our work as open source implementations and integrated with mainstream RIOT, which we forked in a repository available online at https://github.com/future-proof-iot/RIOT/tree/H2020-Sparta-Deliverable-D6-2

The focus of this fork of RIOT (based on Release 2020.10) is to:

- highlight some of **our on-going contributions upstream to RIOT** master branch: hundreds of commits since the beginning of H2020 Sparta including, but not limited to, SUIT-compliant secure OS software update support, and support for secure 6TiSCH networking.
- offer a sneak peek at **complementary security mechanisms we developed** but that are not (yet) integrated in RIOT master branch, and how this platform is used in the context of Sparta, e.g. a prototype of minimal virtual machines for software module hosting and isolation.

Currently, among the features we mention below, only SUIT and secure 6TiSCH support have been merged into the master branch of RIOT. The other functionalities we mention are only present in this fork for now. In next steps, we plan to upstream most of the functionalities we develop to the master branch of RIOT.

Chapter 2 SUIT-Compliant Secure IoT Software Updates

We co-author the SUIT standard proposed by IETF to secure IoT software updates. The SUIT specifications define a security architecture, and the necessary metadata and cryptography to secure software updates, applicable on microcontroller-based devices, such as the ones RIOT runs on. We integrate SUIT support for secure firmware updates as described in our publication on this topic. We support a SUIT-compliant workflow as depicted in Fig. 1 below, which is the blueprint for the AirMonitor demo, shown in a video tutorial we provide.

To try this functionality on your own hardware (an nrf52840dk board for instance) start with the suit_update example (see the online GitHub repo we provide at https://github.com/future-proof-iot/RIOT/tree/H2020-Sparta-Deliverable-D6-2).

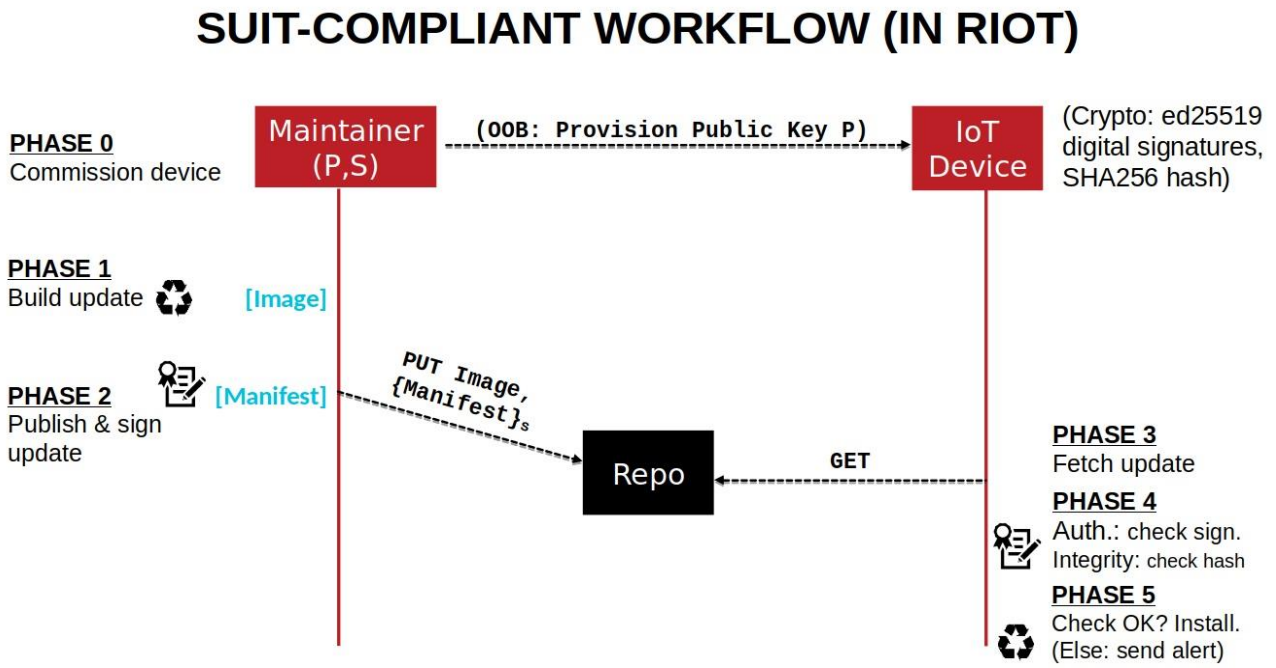

Figure 1: SUIT-compliant software update life-cycle.

As shown in Figure 1, the workflow consists in a preliminary phase (Phase 0) whereby the maintainer produces and flashes the IoT devices with commissioning material: the bootloader, the initial image, and authorized crypto material. Once the IoT device commissioned, the maintainer can trigger iterations through cycles of phases 1-5, whereby the authorized maintainer can build a new image and sign the corresponding standard metadata (the SUIT manifest) which can be transferred to the device over the network via a repository (CoAP resource directory), and upon cryptographic verification on-board the device, the new image is installed and booted.

Using the mechanisms specified by SUIT, our security-enhanced OS can for example mitigate the below attacks.

Tampered Firmware Update Attacks – An attacker may try to update the IoT device with a modified and intentionally flawed firmware image. To counter this threat, our prototype based on SUIT uses digital signatures on a hash of the image binary and the metadata to ensure integrity of both the firmware and its metadata.

Unauthorized Firmware Update Attacks – An unauthorized party may attempt to update the IoT device with modified image. Using digital signatures and public key cryptography, our prototype based on SUIT ensure that only the authorized maintainer (holding the authorized private key) will be able to update de device.

Firmware Update Replay Attacks – An attacker may try to replay a valid, but old (known-to-be-flawed) firmware. This threat is mitigated by using a sequence number. Our prototype based on SUIT uses a sequence number, which is increased with every new firmware update.

Firmware Update Mismatch Attacks – An attacker may try replaying a firmware update that is authentic, but for an incompatible device. Our prototype based on SUIT includes device-specific conditions, which can be verified before installing a firmware image, thereby preventing the device from using an incompatible firmware image.

Chapter 3 Secure Low-power IoT Networking

We integrated OpenWSN, the standards-compliant open-source implementation of the 6TiSCH network stack, as described in our publication on this topic.

To try out this functionality, start by trying out the steps described in the dedicated documentation, which we also show in a video tutorial we provide, which demonstrates secure network joining with CoJP (Constrained Join Protocol) over IEEE 802.15.4 radio and 6TiSCH, with OSCORE context establishment and routing tree formation with RPL, on OpenMote nodes.

(See the online GitHub repo we provide at https://github.com/future-proof-iot/RIOT/tree/H2020-Sparta-Deliverable-D6-2).

We plan to use this base, purposely designed to be extensible above the libraries providing OSCORE and COSE support, to implement and integrate complementary upcoming secure IoT protocols such as EDHOC (Ephemeral Diffie-Hellman Over COSE) which build upon OSCORE and COSE.

Furthermore, we use this platform to benchmark and compare different secure IoT protocols stacks, for example as described in our other publication on this topic comparatively evaluating DTLS1.3.

Chapter 4 Low-power Virtual Machines using rBPF

We designed rBPF, a register-based VM based on extended Berkeley Packet Filters (eBPF). In our publication on this topic, we show that rBPF execution time overhead is tolerable for low-throughput, low-energy IoT devices. We further show that, using a VM based on rBPF requires only negligible memory overhead (less than 10% more memory). Compared to an alternative such as WebAssembly for microcontrollers (Wasm3) which requires 100% overhead, rBPF is thus a promising approach to host small software modules, isolated from OS software, and updatable on-demand, over low-power networks.

This functionality is not yet available in the master branch of RIOT, but it is available in this fork. To try out this functionality, start with the gcoap_bpf example (see the online GitHub repo we provide at https://github.com/future-proof-iot/RIOT/tree/H2020-Sparta-Deliverable-D6-2).

Chapter 5 Conclusion

In this deliverable we showcase secure software update capabilities, secure low-power network capabilities, and minimal virtual machine isolation mechanisms applicable on low-power microcontrollers. The output of our work is available as open source implementations and integrated with mainstream RIOT, which we present in a fork of the main branch, hosted in a repository available online. We also committed upstream secure IoT software update mechanisms and secure IoT networking mechanisms to the main branch of RIOT.

We plan to continue using a similar workflow with our upcoming contributions in the context of H2020 Sparta. We will upstream as much as possible our output to the master branch of RIOT, and in parallel, we will publish and maintain complementary open source modules in the fork, to showcase their potential integration.

Chapter 6 List of Abbreviations

Abbreviation	Translation
SUIT	Software Updates for Internet of Things
6TiSCH	IPv6 over the TSCH mode of IEEE 802.15.4e
OSCORE	Object Security for Constrained RESTful Environments
COSE	CBOR Object Signing and Encryption
RPL	IPv6 Routing Protocol for Low-Power and Lossy Networks
EDHOC	Ephemeral Diffie-Hellman Over COSE
DTLS	Datagram Transport Layer Security Protocol
rBPF	RIOT Berkeley Packet Filter
Wasm	WebAssembly for Microcontrollers

Chapter 7 Bibliography

[1] K. Zandberg et al. "Secure firmware updates for constrained IoT devices using open standards: A reality check." IEEE Access, 2019

[2] B. Moran et al. "A Concise Binary Object Representation (CBOR)-based Serialization Format for the Software Updates for Internet of Things (SUIT) Manifest." IETF Internet Draft draft-ietf-suit-manifest-09, 2020

[3] K. Zandberg, E. Baccelli. "Minimal Virtual Machines on IoT Microcontrollers: The Case of Berkeley Packet Filters with rBPF." IFIP/IEEE PEMWN, 2020

[4] G. Restuccia, et al. "Low-Power IoT Communication Security: On the Performance of DTLS and TLS 1.3." IFIP/IEEE PEMWN, 2020

[5] T. Claeys et al. "RIOT and OpenWSN 6TiSCH: Happy Together.", IFIP/IEEE PEMWN, 2020

[6] RIOT Release 2020.10

[7] 6TiSCH Tutorial

[8] Sparta D6.2 Repository https://github.com/future-proof-iot/H2020-Sparta-D6-2-Sparta-RIOT-fp