Holistic IoT Architecture for Secure Lightweight Communication, Firmware Update, and Trust Monitoring

Jesus Sanchez-Gomez
Dept. of Information and Communications Engineering
University of Murcia
Murcia, Spain
jesus.sanchez4@um.es ORCID 0000-0003-2673-3790

Rafael Marin-Perez
Dept. of Research and Innovation
Odin Solutions S.L.
Murcia, Spain
rmarin@odins.es ORCID 0000-0002-8521-1864

Mirko Ross
Dept. of Direction
Digital Worx GmbH
Stuttgart, Germany
m.ross@digital-worx.de

Antonio Fernando Skarmeta Gomez
Dept. of Information and Communications Engineering
University of Murcia
Murcia, Spain
skarmeta@um.es ORCID 0000-0002-5525-1259

***Abstract*—IoT applications have recently proliferated due to their applicability in several fields, as well as the growing number of enabler technologies. For this reason, the landscape presents vast deployments formed by end-devices with heterogeneous capabilities or requirements. Low-power wide-area communication technologies have partially filled the gap for low-bandwidth low-cost IoT devices that are placed in vast coverage areas without a power-grid or cellular signal. However, these technologies seldom offer interoperable solutions to manage security-related tasks, such as monitoring cybersecurity attacks or firmware update distribution. Thus, there is a need for a human-centric platform that enables trust-worthy management of large heterogeneous IoT networks. In this work, we present a solution that enables trust monitoring and firmware update distribution employing novel open standardization efforts designed for constrained devices. The presented solution leverages on LO-CoAP-EAP, a novel lightweight bootstrapping protocol, LoRaWAN, a widespread long-range communication technology, SCHC, an IPv6 header compression and fragmentation mechanism, OSCORE, an end-to-end application-layer protection, IPFS a peer-to-peer decentralized storage solution, as well as a Hyperledger, a distributed ledger technology for secure validation of the distributed contents.**

***Index Terms*—LPWAN, SCHC, CoAP-EAP, IPFS, Blockchain**

I. Introduction

The Internet of Things (IoT) has recently become one of the most important topics in social, economic and technological advancement. IoT connectivity allows reaching the physical domain through connected devices including sensors and actuators. There are solutions constantly appearing in the global market, improving the performance and connectivity for small form-factor, battery-powered devices. For this reason, there is an estimated exponential growth in the amount of devices connected to the Internet. The IoT application include a wide variety of verticals, such as Smart Cities, Smart Buildings, Industry 4.0, e-Health, and Intelligent Transportation Systems (ITSs), among others. Numerous IoT scenarios leverage on Machine Type Communications (MTC), where low-power devices are deployed to monitor different environment parameters, during long periods of time. Some of these devices are also capable of actuating over their environment. Typically, MTC implies the use of a centralized platform where a higher level of orchestration by the deployment administrators can be achieved.

Low-Power Wide-Area Networks (LPWANs) [1] are capable of providing wireless coverage range to large areas with a reduced amount of base-stations. They allow the connectivity of rural or remote regions without cellular signal. Additionally, In some scenarios, the lack of power-grids or wired communication mediums force IoT devices to employ low-power low-bandwidth communication technologies. With the arrival of LPWANs, the IoT gap for those applications has been partially solved. However, these notable characteristics are achieved at the expense of having a highly constrained communication channel. It is designed to support a relatively reduced number of short-packets per device, per day. This limitation can be noticed in the most popular LPWAN technologies that employ unlicensed radio-bands, I.e., SigFox, LoRaWAN [2], or NB-IoT. As a consequence of the above-mentioned limitations, LPWAN technologies prefer network stacks with a reduced

This work has been supported by the European Commission under IoTCrawler (Grant No. 779852), Plug-n-Harvest (Grant No. 768735), EU IoTrust (Grant No. 825618), PHOENIX (Grant No. 893079), PRECEPT (Grant No. 958284), 5GASP (Grant No. 101016448), and INSPIRE-5Gplus projects; by the Spanish Ministry of Science, Innovation and Universities, under GUARDIAN (Grant No. TSI-100110-2019-20) projects; by the Spanish Ministry for the Ecological Transition and the Demographic Challenge under the MECANO project (Grant No. PGE-MOVES-SING-2019-000104); and by Seneca Foundation in Murcia Region under 20751/FPI/18 which is partially funded by FEDER funds.

number of lightweight protocols [3].

LPWAN devices are usually installed in hard-to-reach locations, and operate during months or even years without direct human supervision. Hence, they do not typically include keypads or displays. Thus, they have a relatively long operation life-cycle of unattended operation, during which, LPWAN devices are rarely monitored or updated directly by an operator. This issue is only exacerbated when security update patches are extremely hard to distribute remotely by employing the low-bandwidth datarates commonly present in LPWANs. Also, common scenarios include the deployment of hundreds or even thousands of these devices. Due to this massive aspect of IoT, security vulnerabilities present a severe challenge to solve once the deployment stage is finished, due to the potential cost of directly updating the software of each unit. Additionally, the response time to update all the affected devices must be as short as possible, since it is very unlikely that attacked devices can be remotely updated once the original firmware has been maliciously tampered. Therefore, security is an integral part of every IoT platform, during the early design stages of each project. Not only the legitimate owner of the device might loose its control, but also the tampered units are typically employed as botnets in malicious Distributed Denial of Service (DDoS) attacks, which only exacerbates the need for efficient and timely patch updates.

However, existing IoT products fail to provide human-centric and trustworthy applications covering security and privacy requirements specified in international legal regulations, such as: (i) GDPR[1] for data privacy and protection of personal data, (ii) NIS Directive[2] for inter-network operations, and (iii) ENISA Regulation for ICT Cybersecurity Certification[3]. Moreover, to maintain IoT networks, open and trustworthy solutions are required for detecting vulnerabilities in IoT software and hardware, and provide a robust firmware update management. However current solutions [3] are complex, vendor-specific, or not openly standardized. IoT deployments are characterized by devices with heterogeneous computational capabilities and network requirements. Thus, solutions that enable interoperation through open standardized technologies are required to improve cohesion and reduce network management overhead.

In this work, we present a novel architecture to tackle these challenges, based on a trustworthy, open, and human-centric solution to setup and maintain IoT networks. This proposal is based on the integration of a novel bootstrapping protocol, Peer-to-Peer file storage solutions, and Distributed Ledger technologies in order to provide secure initialization of IoT devices, vulnerabilities detection, and software patching. The proposed solution is based on recent standardization and research activities by the Internet Engineering Task Force (IETF) and peer-to-peer InterPlanetary File System (IPFS) with Distributed Ledger Technology, to ensure decentralized IoT networks. The main expected outcome will be a standards-based solution with open-source stacks for IoT devices and distributed services platform that will be supported by real-world pilot validation.

The remainder of this work is organized as follows. Section II provides background regarding bootstrapping and secure packet transmission over long-range wide-are technologies. Section III introduces the proposed secure lightweight communication, trust monitoring, and firmware update architecture. Finally, Section IV closes the paper and indicates future ways.

II. Background

Manual re-configuration of end-devices is prohibitively expensive after the deployment stage. Devices operate remotely from the moment they are installed in their definitive location. Then, all the following device configuration procedures are done over radio. The first dynamic procedure performed by IoT devices is known as secure bootstrapping. This involves authentication, authorization and key agreement operations, which are vital to control network resources and communications. Confidentiality and authentication processes of popular LPWAN technologies are designed with their radio layer in mind. As a result, they are vendor-specific, which disables interoperability with other architectures or technologies. This design choice is motivated by the severe restrictions in bandwidth, that do not allow exchanges of packets larger than a few 10s or 100s of bytes. For this reason, LPWAN interoperable security mechanisms present a challenge, since common technologies that work over IP network layer are prohibitive in this context. IP-based technologies typically employ a standardized protocol such as RADIUS [4] or Diameter [5] to carry Extended Authentication Protocol (EAP) messages. However, these require the transmission of relatively large packets, as well as a high number of exchanges. Thus, there is still a need for a standardized lightweight protocol that permits IP-based interoperability among constrained networks.

A. LoRaWAN transmission technology

LoRaWAN [2] has been chosen for this research proposal due to its strengths in the context of heterogeneous and scalable IoT scenarios. LoRaWAN is an LPWAN technology that has recently gained a lot of attention from both industry and academia. It is developed by the LoRa Alliance[4], has an open specification [2], and is supported by numerous market-ready products and network deployments around the globe. LoRaWAN is targeted to low-power, low-bandwidth, and low-cost IoT device solutions. It employs the sub-1GHz unlicensed Industrial, Scientific, and Medical (ISM) radio band in each region, tied to severe regulations such as a duty-cycle of 1%. For this reason, messages in LoRaWAN communications have a maximum payload size in the order of 10s of bytes. It employs a star-of-stars architecture, where all the end-device communications go through a central network server that

[1] http://data.europa.eu/eli/reg/2016/679/oj
[2] http://data.europa.eu/eli/dir/2016/1148/oj
[3] http://data.europa.eu/eli/reg/2019/881/oj
[4] https://www.lora-alliance.org/

routes all the application level data, as well as maintaining an overall network health through Medium Access Control (MAC) commands. LoRaWAN off-the-shelf security includes a network join procedure known as Over-The-Air Activation (OTAA), is based in a pre-shared root key. However, the OTAA procedure itself is vendor-specific, and the root key is shared by the end-device and the central LoRaWAN platform, thus the confidentiality of all the exchanged packets depends on the particular deployment security policies. This is a current challenge because regular off-the-shelf LoRaWAN assumes trust in the central server. If it gets compromised, all the following messages may be decrypted and tampered with.

B. Secure Transmission of IPv6 packets over LoRaWAN

1) SCHC: Different LPWAN technologies employ radio communications over unlicensed radio-bands. Hence, they rely on simple and vendor-specific network stacks, where application data is directly transported over the MAC layer. They lack a complete IP network stack middelware. This contradicts the vision of IoT, where interoperability over IP networks is mandatory for end-devices. The IoT paradigm aims at the interconnection of heterogeneous devices through low-bandwidth networks. This demands the use of novel mechanisms and protocols in order to enable a seamless and secure integration of these devices with the rest of the Internet. Due to the vast number of connected IoT devices, which is expected to grow exponentially, the IPv6 address space is necessary to accommodate such a large amount of devices. However, IPv6 header sizes are prohibitively large for the typical LPWAN data-rate characteristics. This hinders the direct integration of LPWAN-based devices with other deployments. To this end, the Internet Engineering Task Force (IETF) has developed the novel Static Context Header Compression (SCHC) [6] technology. SCHC provides a header compression and fragmentation mechanism for IPv6/UDP packets. Also, a new scheme to compress CoAP headers with SCHC is also currently under standardization [7]. SCHC exploits certain characteristics of LPWANs in order to perform efficient header compression, namely: (i) the star-of-stars topology found in LPWANs guarantees that the source and destination does remains even if the path changes, and (ii) device behaviour is unlikely to change during its regular life-cycle. As a consequence, data-flows are commonly well-known during the design and development stage of IoT MTC scenarios. Thanks to SCHC, devices are able to interoperate with IPv6 hosts connected to the Internet while improving radio spectrum usage and power efficiency.

SCHC is placed as an adaptation layer between the MAC and the network layers, and is independent from the LPWAN technology below it. Since not all LPWAN tehcnologies are capable of accommodating the IPV6 required maximum transmission unit of 1280 bytes in a single message, SCHC includes an optional fragmentation and reassembly mechanism. This procedure is tailored to the LPWAN's low data-rate requirements Overall, the best contribution of SCHC is hiding the technical details and limitations of underlying LPWANs, to enable the integration of end-devices through standardized Internet protocols.

2) OSCORE: The Object Security for Constrained RESTful Environments (OSCORE) [8] is employed in this proposal during the second stage of operation, shown in Fig. 2.b. OSCORE is a protocol extension of CoAP that addresses end-to-end security. It is a technology that secures individual CoAP packets at an object level, as opposed to other related technologies that perform a handshake and establish a security session. Hence, it maintains end-to-end security even when CoAP proxies are found in the route. OSCORE offers integrity, authenticity, and confidentiality mechanisms at a message level. To do so, the plain CoAP message gets encapsulated as a compact authenticated and encrypted OSCORE object. Other countermeasures offered include response delay attacks, and response mismatch attacks, through secure binding mechanisms among corresponding messages identified by their authenticated parameters. One of the highlighted improvements of OSCORE over related technologies, like Data Transport Layer Security (DTLS), is presented in its capacity to secure multicast transmissions. As opposed to session-oriented protocols OSCORE encrypts the messages at a packet level. As a consequence, multicast transmissions are efficiently supported, even in vast IoT deployments, where the same message must be securely broadcasted to hundreds or even thousands of devices. For this reason, OSCORE is considered as the indisputable choice for open standardized solutions in group communications within IoT applications [9]. Additionally, OSCORE is an enabler for the device-management Lightweight Machine-to-Machine (LwM2M) protocol, by the Open Mobile Alliance (OMA)[5], or Open Connectivity's Fairhair[6].

C. Firmware Updates Over The Air

Another key feature of current heterogeneous IoT scenarios is a trusted end-device software update management procedure that is able to distribute new versions of binary code wirelessly. This paradigm is known as Firmware Updates Over-The-Air (FUOTA). After a new patch, devices will receive and verify the updated binary code from a trusted platform over the radio technology. After the authenticity and integrity validation, the end-device reprograms itself with the new firmware and restarts operation. Due to the limited data-rate of LPWAN technologies, delta mechanisms are preferred, sending only the minimum the different bits of data required to locally rebuild the whole updated firmware.

III. Proposed Architecture

This section describes the proposed architecture, presented in Fig. 1. It is an energy-efficient suite for constrained low-power environments that combines a novel lightweight bootstrapping process; OSCORE for end-to-end data protection; SCHC for efficient header compression and fragmentation; Peer-to-Peer storage solutions for the scalable distribution of firmware updates; as well as distributed ledger technologies

[5]https://omaspecworks.org/

[6]https://openconnectivity.org/developer/specifications/fairhair/

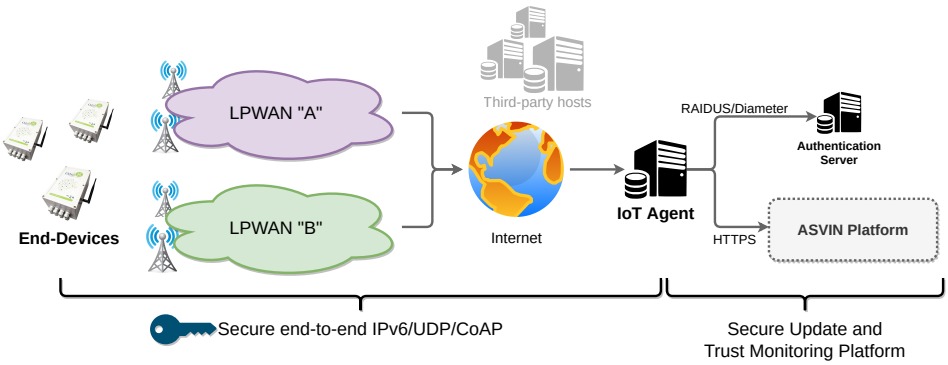

Fig. 1. Proposal architecture.

to secure the integrity of distributed content. By leveraging on these standardized protocols, this architecture enables energy-efficient security and interoperability of distributed and heterogeneous low-power wireless networks to enable trust monitoring and secure update firmware distribution.

In this proposal, the lightweight Low-Overhead EAP over CoAP (LO-CoAP-EAP) [10] protocol is chosen to perform authentication and key agreement. LO-CoAP-EAP leverages on three open standards to guarantee interoperability, namely: (i) the Constrained Application Protocol (CoAP) [11] to encode application payloads, (ii) EAP for the encapsulation of authentication messages, and (iii) is based on the Authentication, Authorization, and Accounting framework (AAA) [12], which provides scalability and homogeneity to perform security network management tasks.

In this work, we focus on two different stages in the IoT end-device life-cycle. These two stages are depicted in Fig. 2. First, devices perform a bootstrapping process when they are deployed. This process includes a secure authentication and key agreement exchange, shown in Fig. 2.a. Once the device successfully authenticates itself, derived session keys are obtained by both end-points. Next, the device switches over to the regular operation stage, Fig. 2.b, where all communications are secured through OSCORE.

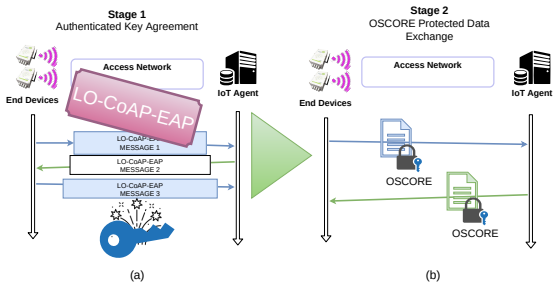

Fig. 2. End device operation stages: (a) secure authentication stage, and (b) secure update interaction.

A. End-Device

It is a small form-factor hardware which sits on the edge of an LPWAN network. It consists of a microcontroller, radio module, and sensor peripherals. These end-devices are typically installed in hard-to-reach locations under adverse conditions. Additionally, end-devices are relegated to simple monitoring and actuation tasks that deposits the obtained data in a centralized platform. The end-device shall incorporate a LoRaWAN module. Upon deployment, the end-device establishes a secure communication link with the LoRaWAN server through the LoRaWAN OTAA scheme [2]. Once connected to the LoRaWAN network, it performs the LO-CoAP-EAP authentication and key agreement procedure together with the IoT agent. As a result, the end-device obtains a session key employed for further communications with the secure update platform, through OSCORE protected messages.

B. IoT Agent

It is a trusted component that acts as a bridge between the constrained and non-constrained side of the end-to-end interaction, placed within the secure update and trust monitoring platform. In essence, it acts as a bridge among the low-power network and the platform. Hence, its IPv6 network address must be pre-provided to the end-devices, and must continuously listen to UDP/CoAP requests. During the authentication and key agreement stage, the Iot Agent plays the role of authenticator in the AAA framework. Thus, all the LO-CoAP-EAP messages transmitted by end-devices will be addressed to it. Then, the received payload is encapsulated in a non-constrained authentication protocol, such as RADIUS or Diameter. The resulting re-encapsulated EAP payload is forwarded to the authentication server, the end component of the AAA architecture. Likewise, the IoT Agent will do the reverse procedure for downlink messages. As a result, it will attain a set of session keys that will be employed for the messages in the following exchanges. During the operation stage, it acts as the end-point of the OSCORE communications, embedding the end-device CoAP requests embedded within OSCORE messages to the REST APIs exposed by the different elements of the asvin platform. Due to the lack of CoAP and OSCORE support by these end-points, the IoT Agent acts as a translator among the constrained and non-constrained side of the interaction.

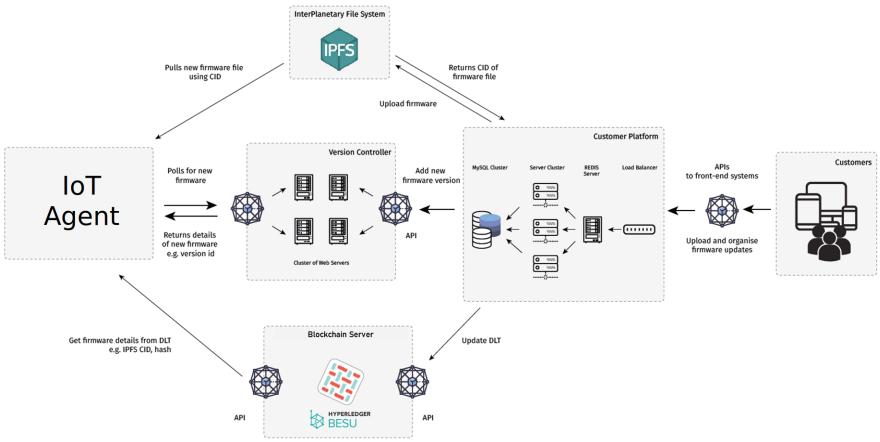

Fig. 3. asvin Platform Architecture

C. Authentication Server

It presents the final end-point of the AAA architecture, serving as an central administrative point for network management. It hides the technical details of each end-device capabilities by employing the EAP technology that grants flexibility to the authentication process. The authentication servers chooses the adequate authentication and key agreement methodology for each device. Hence, administrators only have to provide the cryptographic material and security policies, in a homogeneous and normalized manner. After each end-device is deployed, it performs a bootstrapping process with the authentication server. As a result, both end-points of the communication authenticate themselves mutually and fresh sessions key are attained.

D. asvin Platform

The asvin platform[7] is a Platform-as-a-Service (PaaS) to facilitate over-the-air security patches for IoT devices using novel decentralized and distributed technologies. It provides a complete solution for device, security patches and rollout management. It is comprised of four components as shown in Fig. 3, each of them tailored to efficiently perform a specific set of tasks.

The asvin platform components securely communicate among them through HTTPS, a secure extension of HTTP that leverages on TLS/SSL. Public key certificates are used to authenticate the components, as well as to secure the exchanges between external clients. The IoT agent forwards end-device messages to the platform using the exposed REST API end-points. Also, the asvin platform can directly communicate with each end-device through their unique IPv6 address via the IoT agent's secure CoAP/HTTP translation.

1) IPFS: The IPFS protocol is utilized to store firmware and patches. The IPFS is a content-addressable peer to peer method for storing and sharing hypermedia in a distributed file system. It solves the problem of duplicate files across the network as it exists in the HTTPS and remove redundancy. When a firmware file is stored on the network a hash is generated based on content of the firmware and is stored on a blockchain network. This unique hash is called Content Identifier (CID)[8]. Subsequently, the CID is utilized to pull the firmware from IPFS server.

2) Blockchain: The asvin platform employs distributed ledger technology to provide an extra layer of security and resiliency to the platform. All events of the platform are recorded in a shared ledger. The distributed ledger also contains critical meta data information of devices and firmware. The blockchain infrastructure is based on Hyperledger Fabric[9] and Besu[10]. Both are private permissioned blockchain network technologies designed and developed under the Linux Foundation[11].

3) The Customer Platform: The customer platform facilitates a web-interface dashboard, Fig. 4, to manage end-devices, and their firmware updates. It acts as an abstraction layer to hide the complexity and sophistication of the decentralized and distributed ledger technologies of the asvin platform. Thus, it can be operated without specialized knowledge or tools. The customer platform delivers device management, where the network operator will be able to use the dashboard to register end-devices, group them in several classes, and review device monitoring statistics. Additionally, the user can upload firmware update files, which are stored on the platform's IPFS server. The operator can schedule a rollout of the latest patch files to the device groups. The customer platform interacts with the Hyperledger blockchain server to update the end-device firmware database and keeps the version control server updated with information from the latest firmware.

4) Version Controller: The version controller component consists of multiple nodes which have an exact copy of a web server and host identical services. Nodes form a cluster

[7]https://asvin.readthedocs.io/en/latest/

[8]https://docs.ipfs.io/concepts/content-addressing/

[9]https://www.hyperledger.org/use/fabric

[10]https://www.hyperledger.org/use/besu

[11]https://www.linuxfoundation.org/

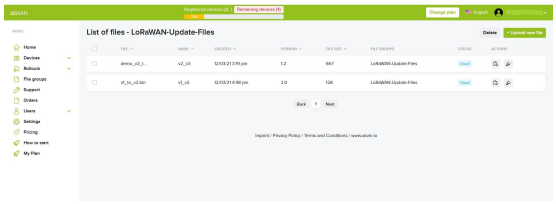

Fig. 4. Customer platform — firmware and rollout management screen

with a fully functional web server that can serve a request independently. Each node has different network address, but they are hidden from other components of the asvin platform architecture. Hence, an abstraction layer is used on top of the cluster to hide complexity. This abstraction layer makes use of round-robin DNS technique for load balancing and fault tolerance. The server performs following tasks: (i) Response to end-devices, in order to register on the asvin platform, the end-devices send request to the version controller. They also poll the version controller to check for new software updates. In turn, the server responds with information of the current valid firmware and rollout id if any update is scheduled. (ii) Latest firmware version list, it maintains real-time information of different firmware versions available on data storage servers. The version controller has a list of all available firmware on the asvin platform for each end-device. Note that the version controller and customer platform share same database.

5) Trust monitoring: The threat landscape of end devices is quite large. There are multiple pain points for an end-device where it can be compromised. An attacker can steal data from the device or employed it as a bot to raise DDoS attacks. It is even more alarming when an end-device is part of a critical company network. The IETF has given guidelines like Manufacturer Usage Description (MUD)[12] to substantially reduce the threat surface of end-device to those communication intended by the manufactures. In that line, the asvin Platform provides a novel dynamic trust monitoring feature to monitor continuously the end-devices for external threats and raise an alarm on the dashboard before an end-device comes under threats. The trust monitoring service includes scanning the critical characteristics of end-devices for an example number of successful patches, number of device reboots, last heartbeat from the device and availability of the encryption key generated after the secure bootstrapping process. All these parameters play crucial role to determine end-device security. The Asvin Platform will collect these parameters from end-device in user defined configurable time interval. On the platform the parameters will be analyzed and a dynamic trust score will be calculated. The platform will also generate a weekly, monthly trust monitoring report for end-devices.

IV. Conclusions

IoT-based technologies and applications have proliferated during recent years. However, different vendors compete to get the biggest market share, without considering open standardized protocols or architectures. Hence, vendors have yet not filled the gap for an interoperable platform that manages the distribution of security update patches over radio in a homogeneous and concise way.

In order to solve these issues, in this work we provide a trust-worthy and human-centric IoT platform to monitor vulnerabilities and manage secure update patches over a constrained network. This solution enables trust monitoring and firmware update distribution employing novel open standardization efforts designed for constrained devices. The presented solution leverages on LO-CoAP-EAP, a novel lightweight bootstrapping protocol, LoRaWAN, a wide-spread long-range communication technology, SCHC, an IPv6 header compression and fragmentation mechanism, OSCORE, an end-to-end application-layer protection, IPFS a peer-to-peer decentralized storage solution, as well as a Hyperledger, a distributed ledger technology for secure validation of the distributed contents. In future contributions, we expect to present test validation results obtained from our current in-progress development.

[12]https://tools.ietf.org/html/rfc8520

References

[1] R. Sanchez-Iborra and M.-D. Cano, "State of the art in LP-WAN solutions for industrial IoT services," *Sensors*, vol. 16, no. 5, p. 708, 2016. [Online]. Available: http://www.mdpi.com/1424-8220/16/5/708/htm

[2] LoRa Alliance, N. Sornin, M. Luis, T. Eirich, T. Kramp, and O. Hersent, "LoRaWAN Specification v1.0.2," *LoRa Alliance*, 2016. [Online]. Available: https://lora-alliance.org/resource-hub/lorawantm-specification-v102

[3] O. Garcia-Morchon, S. Kumar, and M. Sethi, "Internet of Things (IoT) Security: State of the Art and Challenges," Tech. Rep., apr 2019. [Online]. Available: https://www.rfc-editor.org/info/rfc8576

[4] A. Rubens, C. Rigney, S. Willens, and W. A. Simpson, "Remote Authentication Dial In User Service (RADIUS)," RFC 2865, 2000. [Online]. Available: https://rfc-editor.org/rfc/rfc2865.txt

[5] G. Zorn, "Diameter Network Access Server Application," RFC 7155, 2014. [Online]. Available: https://rfc-editor.org/rfc/rfc7155.txt

[6] A. Minaburo, L. Toutain, C. Gomez, and D. Barthel, "SCHC: Generic Framework for Static Context Header Compression and Fragmentation," no. 8724, apr 2020. [Online]. Available: https://rfc-editor.org/rfc/rfc8724.txt https://www.rfc-editor.org/info/rfc8724

[7] A. Minaburo and L. Toutain, "LPWAN Static Context Header Compression (SCHC) for CoAP," Internet Draft, Internet Engineering Task Force, Dec. 2021. [Online]. Available: https://tools.ietf.org/html/draft-ietf-lpwan-coap-static-context-hc-18

[8] G. Selander, J. Mattsson, F. Palombini, and L. Seitz, "Object Security for Constrained RESTful Environments (OSCORE)," RFC 8613,, jul 2019. [Online]. Available: http://tools.ietf.org/rfc/rfc8613.txt https://www.rfc-editor.org/info/rfc8613

[9] J. Park, M. Jung, and E. P. Rathgeb, "Survey for Secure IoT Group Communication," *2019 IEEE International Conference on Pervasive Computing and Communications Workshops, PerCom Workshops 2019*, pp. 1026–1031, 2019.

[10] D. Garcia-Carrillo, R. Marin-Lopez, A. Kandasamy, and A. Pelov, "A CoAP-Based Network Access Authentication Service for Low-Power Wide Area Networks: LO-CoAP-EAP," *Sensors*, vol. 17, no. 11, p. 2646, nov 2017. [Online]. Available: http://www.mdpi.com/1424-8220/17/11/2646

[11] Z. Shelby, K. Hartke, and C. Bormann, "The Constrained Application Protocol (CoAP)," Tech. Rep., jun 2014. [Online]. Available: https://www.rfc-editor.org/info/rfc7252

[12] D. Spence, G. Gross, C. de Laat, S. Farrell, L. H. M. Gommans, P. R. Calhoun, M. Holdrege, B. W. de Bruijn, and J. Vollbrecht, "AAA Authorization Framework," RFC 2904, 2000. [Online]. Available: https://rfc-editor.org/rfc/rfc2904.txt