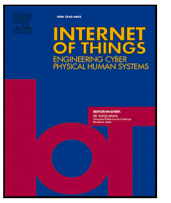
Internet of Things 24 (2023) 100899

Contents lists available at ScienceDirect

Internet of Things

journal homepage: www.elsevier.com/locate/iot



Innovative security and compression for constrained IoT networks

Anthony Feijoo-Añazco [a], Dan Garcia-Carrillo [b,*], Jesús Sanchez-Gomez [a], Rafael Marin-Perez [a]

[a] Department of R&I, Odin Solutions S.L., Murcia 30009, Spain
[b] Department of Computer Science and Engineering, University of Oviedo, Campus de Viesques 33204 Gijón, Asturias, Spain

ARTICLE INFO

Keywords:
Security
IoT
CoAP
OSCORE
SCHC
LPWAN

ABSTRACT

Low-Power Wide-Area Networks (LPWAN) are a type of wireless communication technologies that enable efficient data transfer between low-power devices. This article explores the challenges of ensuring security in LPWANs and proposes a solution that combines two key mechanisms: Object Security for Constrained RESTful Environments (OSCORE) and Static Context Header Compression (SCHC). LPWANs are increasingly being used in Internet of Things (IoT) applications, but the features that endow these technologies with low power consumption and long range also make it necessary to adapt protocols and innovate in the approaches to achieve security while maintaining communications viable in technologies like these, in some cases with severe limitations. OSCORE provides end-to-end security for constrained devices, while SCHC enables efficient IP-based transmission over LPWANs. By combining these two protocols, the proposed solution aims to address the security and efficiency challenges of LPWANs, ensuring that IoT devices and data are protected from unauthorized access and manipulation. The article also presents experimental results demonstrating the feasibility and effectiveness of the proposed approach using a real-life scenario, with industrial-grade hardware. The study concludes that the use of OSCORE and SCHC in LPWAN networks can significantly enhance the security and privacy of IoT devices.

1. Introduction

High expectations placed on the Internet of Things (IoT) paradigm explain the exacerbated increase of IoT device density.[1] If we can foresee that IoT will become a critical element of human infrastructure in the future (e.g. Intelligent Transportation Systems, Smart Agriculture, Industry 4.0), we must ensure that these assets are safe from threats. Any vulnerability introduced in today's planning will expose the global production capacity to crippling blows in the future. The main goal of this work is to address the challenges in LPWAN communications by making use of state-of-the-art open standards in order to achieve an efficient and secure IoT message exchange scheme using Static Context Header Compression (SCHC) and Object Security for Constrained RESTful Environments (OSCORE), to provide an easily replicable (open standardization, seamless integration) and secured protocol stack and network architecture are suggested to make it possible for IoT devices to be quickly and easily deployed on Smart scenarios without the need for specialized knowledge or equipment. IoT has shown to be able to offer improved solutions for the modernization of Smart Cities and Smart Agriculture deployments. Specially through the new set of technologies known as LPWAN, which are capable of covering large regions, with low bandwidth, and potentially extremely small packet and application-layer data sizes,

* Corresponding author.
E-mail address: garciadan@uniovi.es (D. Garcia-Carrillo).

[1] https://iot-analytics.com/number-connected-iot-devices/.

https://doi.org/10.1016/j.iot.2023.100899
Received 12 May 2023; Received in revised form 7 July 2023; Accepted 2 August 2023
Available online 16 August 2023

and extended battery life. These new technologies made possible to monitor and manage city infrastructures and agriculture fields through autonomous wireless sensor networks, the performance of remote control actions on actuators and enhanced the data availability for Decision Support Systems (DSS) and complicated event detection systems, thanks to the increasing smartness level. All this to go even further in the optimization process of irrigation, spraying, tree trimming, pest management, etc. Nevertheless, a look at the current landscape shows that the IoT paradigm might not be mature enough to bear the responsibility that will fall on it. Solutions aimed at the two above-mentioned sectors are subjected to unfavourable conditions, reduce band that make Smart Scenarios a difficult research field where no consensus of the optimal solution existed, as this industry is so recent that no guidelines were available until recently. Thus, flagship technologies currently used do not meet the expectations needed to be one of IoT's future-driving foundations, an issue that is also present in other economical sectors where IoT is being adopted and that we should not keep relegating to the background. The contributions in this study will demonstrate that LPWAN-based use cases, based on existing standards, may accomplish effective and secure communications while keeping the key features of these technologies, namely, a reduced demand in the amount of data sent over the air.

The remainder of this work is structured as follows: Section 2 studies the Internet Engineering Task Force (IETF) standardized protocols jointly used with LPWAN. Section 3 showcases the design of a secure and data-efficient LPWAN communication stack using state-of-the-art protocols. Section 4 elaborates on the development and adaptation of the necessary mechanisms to implement the protocol stack and design of an architecture based on the stack with an added IPv6 adaptation layer. Additionally, real-life deployment of the architecture, testing, and evaluation of the performance of the developed solution is shown. Finally, Section 5 finalizes the work, presenting the conclusions and future ways.

2. Related work

The Internet Engineering Task Force (IETF) is the standard development organisation (SDO) focused on interoperable Internet protocols. It is structured in several Working Groups (WGs), addressing different areas or topics. The results of these WGs are established standards that are now used as the foundation for many other standardization efforts. These efforts are not limited to the IETF but also include other organizations focused on IoT scenarios such as the 3rd Generation Partnership Project (3GPP),[2] Thread,[3] the Industrial Internet Consortium,[4] OMA SpecWorks,[5] OneM2M,[6] the Open Connectivity Foundation,[7] Fairhair Alliance,[8] and the Lightweight M2M standard (LwM2M) by OMA SpecWorks [1,2]. The resource-centred paradigm lies at the core of the IETF's vision for accessing and sharing information originating from constrained end-devices [3]. This paradigm enables managing sensors and actuators through their corresponding resources, such as the `/temp` resource for a device's temperature sensor or the /window-latch resource for opening or locking a window frame latch. Each resource is uniquely identified by a URI, such as `coap://2001::1/temp`, which allows web-based request-and-answer interactions.

The IETF categorizes protocol families into two groups: data transmission plane protocols and mechanisms, such as CoAP or Concise Binary Object Representation (CBOR) [4], and security mechanisms that protect the integrity and confidentiality of exchanges, such as for Object Security for Constrained RESTful Environments (OSCORE) [5], and Datagram Transport Layer Security (DTLS) [6]. It is for the aforementioned reasons, that this work focuses exclusively in the openly standardized efforts by international SDOs. One of the main design goals of this work is to keep the contribution aligned with the ongoing efforts by the IETF.

Certainly, securing communications is not a new goal, and there are standards that have being used for this purpose as well as innovative research work surrounding them. Examples of these are, for instance, the use of IPsec or TLS, typical protocols for standard Internet-based communications. When IoT comes into the picture, the classical protocols and methodologies to secure the communications have to adapt to the constraints of the IoT landscape, such as reduced computational power of Microcontroller Units (MCUs), limited memory (RAM and ROM), typically with reduced bandwidth capabilities, as well as reliance on battery operated platforms. These constraints promote the use of different protocols such as DTLS instead of TLS because of the prevalence of UDP in IoT. Even though DTLS has been predominant in IoT from the start, there were some drawbacks in the literature regarding DTLSv1.2 concretely, that made other alternatives interesting as well. Such is the case of OSCORE, which featured a security association protocol that was not tied to a specific authentication and key agreement protocol as DTLS was, which needs the DTLS handshake to run, to enable the DTLS protocol.

Plus, the DTLS handshake exchange was not favoured due to its taxing requirements [7]. DTLS is a protocol that can protect any application layer that runs on top of it, whilst it has the downside that where proxies are in place, the client and the server cannot achieve end-to-end security. For the specific context of IoT, where one of the most representative application protocols, the Constrained Application Protocol (CoAP), DTLS is stated to be the protocol to secure it in the original RFC [8], but as the use cases where end-to-end security as well as low overhead was a requirement, an application layer security protocol specific for CoAP was designed, namely, OSCORE [5].

[2] http://www.3gpp.org/.
[3] http://threadgroup.org.
[4] http://www.iiconsortium.org.
[5] http://www.omaspecworks.org/ipso-alliance.
[6] http://www.onem2m.org.
[7] http://openconnectivity.org/.
[8] http://www.fairhair-alliance.org/.

One step further, CoAP is also expected to be used in the very constrained LPWAN networks, where an additional twist is needed to add support to certain protocols in these networks. Such is the case of using Static Context Header Compression (SCHC) [9] in these networks, a protocol that allows reducing the overhead of the protocols, enabling their use in the aforementioned LPWAN networks.

The research around LPWAN with SCHC is mostly focused on the evaluation of the first RFC that describes the SCHC mechanisms, in contrast with the application of SCHC to the OSCORE specifics surrounding the CoAP protocol. Aguilar et al. [10] evaluate the energy consumption of SCHC fragmentation over Sigfox, CoAP in this case is not analysed. Muñoz et al. [11] evaluate the performance of SCHC packet fragmentation, but in this case CoAP is also not part of the objectives of the study. Sanchez-Gomez et al. [12] study the performance of SCHC compression and fragmentation in LoRaWAN, tackling the case of no compression, IPv6/UDP compression, and going up to IPv6/UDP/CoAP compression. The analysis of OSCORE is left outside the work. Moons et al. [13] study the use of SCHC to achieve an optimized protocol stack in multimodal LPWAN solutions. They reach the application layer, i.e. CoAP, in their study, but the security based on OSCORE is not part of their objectives. In terms of software, development projects and alike, we find that there are some libraries that provide support to a limited extent. The implementation OpenSCHC[9] and the work in [14] implement the RFC8724, and does not reach the application layer. The work in [15] go a bit further and implement SCHC and compare it with 6LoWPAN.

In this sense, to the best of the author's knowledge, there is no current implementation and evaluation of the OSCORE-protected CoAP exchanges over SCHC in LoRaWAN, which is the main contribution of this article.

To future-proof upcoming Smart City and Smart Agriculture scenarios, several challenges must be faced by state-of-the-art IoT solutions and deployments.

- Compatibility of different vendors' heterogeneous IoT systems and devices. The usage of several protocols and mechanisms by existing devices, processing units, and DSS platforms, makes interactions among them extremely challenging. Open, standardized protocols are essential for enabling interoperability between components and deployments using technologies from various vendors. New networked and interoperable ecosystems are replacing isolated solutions with vendor dependence in Smart Cities and Smart Agriculture.
- Due to the distributed and physically dispersed architecture created by the components and services from various suppliers, these specialized networks are vulnerable to security and privacy risks. Numerous large IoT device deployments and networks have increased the number of security attack vectors available. Attackers may take advantage of these devices, which operate for long periods of time without supervision. This specifically impacts administrators who deploy IoT devices without specialized knowledge or tools. In order to strengthen next-generation IoT services for the lucrative commercial area of Smart Cities and Smart Agriculture, standardized protocols are essential to securely perform activities such as an authenticated key exchange and end-to-end object encryption.
- Due to the absence of power grids and wired communication infrastructure in dispersed geographical areas in agricultural zones and city scenarios, scalable and low-power wireless communications are essential. Rural locations lack access to common Internet connectivity technologies like Wi-Fi or cellular networks. To enable connectivity with the application platforms set up in the cloud, IoT devices need long-range low-power wireless networks and lightweight protocols.

Breaking down the challenges listed above, we see that two of these issues are inherited from the general IoT problem: standardization and security. However, a new problem specific to this sector arises. The use of constrained devices over large and interference-prone geographical areas requires lightweight-focused technologies specially designed to be applied in these cases such as LPWAN, MQTT, LWM2M, CoAP, etc.

3. Background

In this section, we will delve into the background and details of the two main layers that concern LPWAN networks, the link layer and the application layer.

LPWANs are a series of technologies that enable long-distance wireless communications at the cost of extremely low bandwidth in some cases. To make full use of LPWANs, Internet connectivity is necessary and therefore considering security in the communications. But as the IP protocol stack was never really oriented to energy efficiency, there are some serious incompatibilities regarding its application on LPWAN architectures. Nevertheless, through mechanisms such as compression, we are able to integrate LPWANs into the Internet.

3.1. Low-power wide area networks

Many IoT use cases (such as Smart City or Smart Agriculture) require connectivity of devices over a large geographical area, rendering technologies like short-range radio and cellular communications (high power draws) unsuitable. LPWAN are a set of state-of-the-art technologies that overcome these shortcomings.

LPWAN use a type of wireless communication designed to allow low-power long-range communications with end-devices, although at a lower bit rate, thus being specially suited for sensor networks. Fig. 1 offers a comparison that contextualizes LPWAN against other technologies commonly used in Smart Cities and Agriculture. Some of the LPWAN characteristics of this communication technology are:

[9] https://github.com/openschc/openschc.

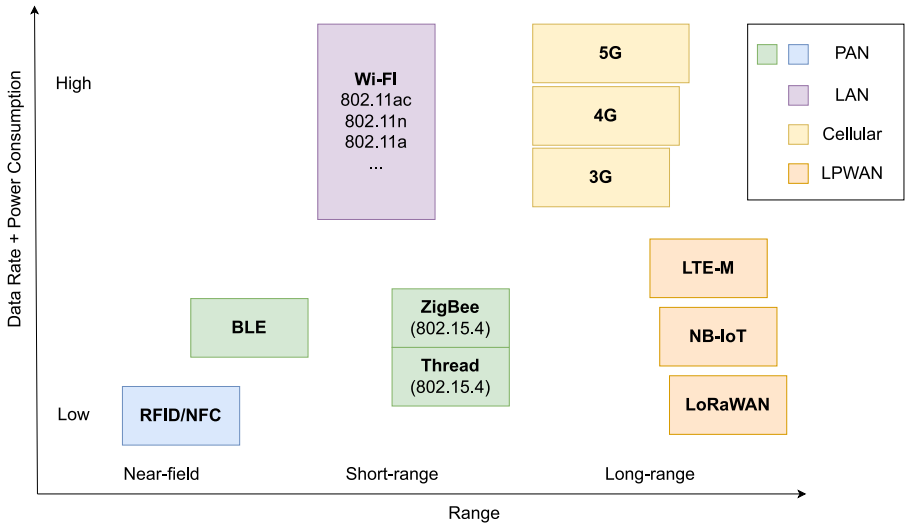


Fig. 1. LPWAN *datarate/range* compared to other wireless communication technologies [16].

- Extremely energy efficient. Devices are able to run for years with a single charge of battery (e.g. 2500 mAh).
- Long range. With coverage range up to several km, giving coverage to entire cities with 1–3 base stations [16,17]
- Low per-device cost. Aimed at massive deployments.
- Highly scalable. These technologies do efficient use of the radio bandwidth resources.

Many LPWAN technologies are available and are extensively used, although many of them foster isolated deployments, as all communications use vendor-specific software and wireless connectivity that is bound to a single third-party managed backend. The most common market-ready LPWAN technologies are discussed in [18,19]. These include Narrowband Internet of things (NB-IoT), IEEE 802.15.4k and 802.15.4w, and LoRaWAN.

We acknowledge that many works in the literature rely on SigFox as a popular LPWAN solution for their contributions. However, the SigFox company recently filled for bankruptcy[10] and was acquired by a third party. For these reasons, and due to the unpredictability of what will happen with SigFox, it was purposely left out of the analysis performed in this work.

Fig. 2 showcases the generic LPWAN architecture identified by the IETF after evaluating the common traits found in all the popular market-ready LPWAN technologies, as found in Request for Comments (RFC) [19]. The generic LPWAN follows a star-of-stars topology, which relies on a centralized Network Gateway that routes all traffic, to and from the devices. In this architecture, devices are the sensors and actuator themselves, often built using resource-constrained battery-power MCUs, deployed in large geographical areas. Next, the Radio Gateways receive all the communication from these devices, forwarding the messages to the central component. The Network Gateway is the interconnection node between devices and the non-constrained side of the network. The LPWAN-AAA server is an abstraction that represents the component in charge of handling secure authentication and authorization procedures. And lastly, the Application Servers offer an entry point to access the device in a standardized way (e.g. a REST API) while hiding the technical details of the network or technology.

Long Range Wide Area Network (LoRaWAN) is a LPWAN technology defined by the LoRaWAN Alliance [20,21]. Its development is addressed by a consortium of relevant IT companies such as Cisco, Semtech, and IBM, among others. This technology establishes a protocol architecture consisting of a Physical (PHY) layer, a Medium Access Control (MAC) layer, and customer applications on top of the MAC layer. The PHY layer uses LoRa, a proprietary Chirp Spread Spectrum (CSS)-based radio technology owned by Semtech. The MAC layer is offered as an open specification [21].

Since the MAC layer is an openly-available specification, this fosters a myriad of market solutions and services, allowing each vendor to develop their own products. Additionally, there are available Free and Open Source Solutions that implement several components of the LoRaWAN ecosystem, such as the back-end server or the device libraries. In the Murcia region, LoRaWAN is able to achieve communication ranges up to 18.5 km in rural scenarios, as well as up to 6.5 km in urban scenarios [17]. These results could be extrapolated to the rest of the European global region, since LoRaWAN employs the same EU 868 ISM radio band.

Thanks to the employment of the LoRa radio technology, LoRaWAN communications have a lower per-device price, since the LoRa radio chips (e.g. Semtex SX1276) can cost as low as 5 Euro. The LoRa radio modulation has a high tolerance against adverse radio conditions – e.g. large building structures, or terrain obstacles – further improving the packet delivery ratio when compared to other radio modulation schemes, like Frequency Shift Keying.

[10] https://techcrunch.com/2022/01/27/sigfox-the-french-iot-startup-that-had-raised-more-than-300m-files-for-bankruptcy-protection-as-it-seeks-a-buyer/.

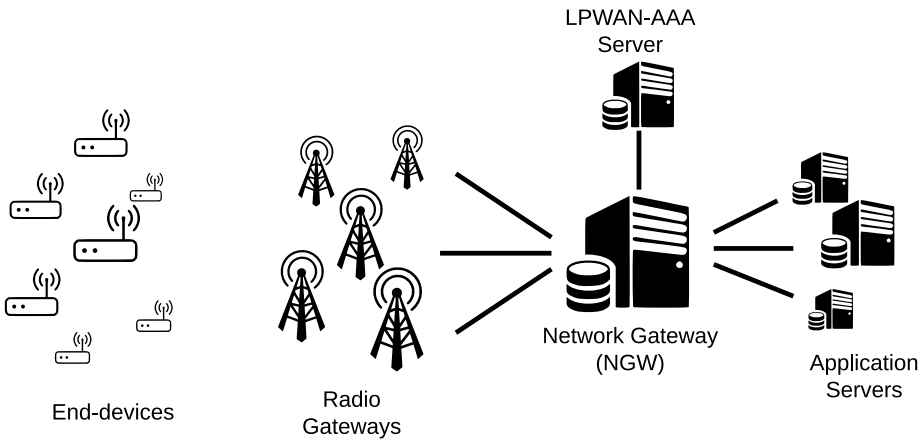


Fig. 2. Architecture of LPWANas identified in RFC 8376.
Source: [19].

Additionally, in Europe, LoRaWAN employs the EU ISM 868 MHz radio band, which is an unlicensed radio band. This is, the transmission of packets in these frequencies is allowed without paying a licence fee. As opposed to cellular technologies (4G/5G) where the clients need to pay a monthly fee to be able to employ the licensed radio bands that sustain those technologies. However, while free, its usage is regulated and this imposes severe limitations on the allowed use of bandwidth, restricting payloads to lengths of 51–242 bytes. This is tightly related to the LoRaWAN Data Rate (DR) [22]. The DR determines the speed of data radio-transmission in the LoRaWAN network. It is measured in bits per second and is influenced by two factors: bandwidth and spreading factor. Spreading factor (SF) determines signal spread in the frequency domain, with SF7 providing the highest data rate and SF12 the lowest. The bandwidth can be either BW125 or BW250, with the wider bandwidth allowing for greater data transmission. From the combination of these two variables there are seven different DRs, ranging from DR0 (lowest speed/highest range) to DR6 (highest speed/lowest range). Choosing the appropriate data rate depends on factors such as device-to-gateway distance, desired battery life, and available network bandwidth. Devices far from the gateway prioritize coverage range and use lower data rates for reliable communication, while devices close to the gateway or requiring higher throughput can use higher data rates for faster data transfer. The choice of data rate is crucial in meeting application requirements.

The maximum data payload (FRMPayload) is not arbitrary and varies depending on the DR. This is due to the fact that the FRMPayload is calculated for each DR based on the maximum allowed (different on each regional specification) on-air transmission time: if a transmission is in progress but the transmission window ends the radio link must be freed.

Hence, choosing a technology based on unlicensed radio bands presents a trade-off due to the relatively smaller quality of service provided by these networks in terms of bandwidth usage allowed per device.

LoRaWAN categorizes devices in three separate classes, namely Class A, B, and C. Class A (All) devices are always in a low-power mode (sleep), and only exit sleep mode sometimes throughout the day, to transmit a data message. After each uplink transfer, two receiving windows are opened before the device resumes its sleep mode. This class is designed for sensor devices that run on batteries and is focused on power efficiency. Actuator nodes are directed to employ Class B (Beacon). It incorporates the entire Class A functionality and periodically opens receiving windows to permit the reception of downlink messages. Lastly, devices in the Class C (Continuous) category are those that have no power restrictions because they are constantly listening for messages to be received.

3.2. Application layer in IoT: CoAP

By achieving the all-important network interoperability, the concept of "resource" also becomes extended. A resource is *"anything important enough to be referenced as a thing in itself"*, traditionally something that can be stored on a computer and represented as a stream of bits. But by making it possible to reference any IoT agent, the closeness to physical objects increases, thus a resource is not restricted to the digital world any more, appearing the concept of a physical resource (non IoT objects that can be interacted through IoT agents). To make use of these constrained devices they need to be integrated into a RESTful environment, client–server architecture with the use of URIs. E.g. Accessing a solenoid valve with `coap://123.112.123.1/valve_close` or the temperature of a room with `http://134.15.134.12/temperature`.

The Constrained Application Protocol (CoAP) [8] is an application protocol aimed at low-power low-bandwidth networks analogous to Hyper Text Transfer Protocol (HTTP), its non-constrained counterpart. With this protocol, we enable IoT devices to support the web-based interaction paradigm of resource management – create, get, update, delete – while reducing the length of the headers and the computational requirements for message processing. As we see in Fig. 3 the fixed header (always present) is 4 bytes long with some fields bitwise encoded. The rest of the fields are optional.

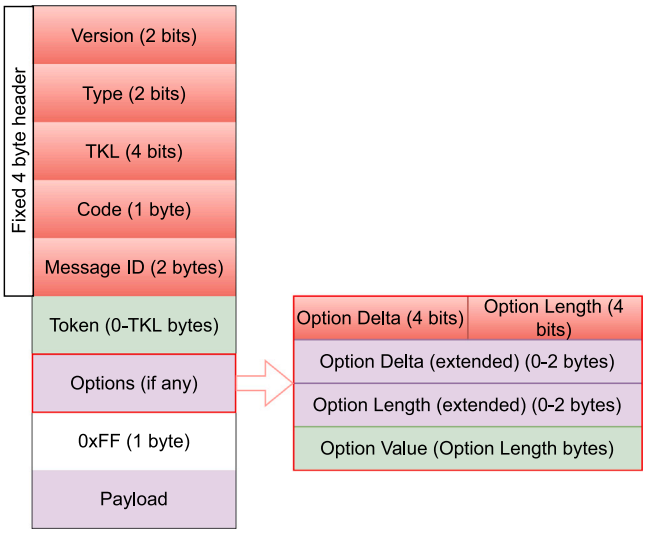


Fig. 3. Structure of a CoAP message.

In the typical usage scenario CoAP is very similar to HTTP. End-points with the role of "client" send requests to other end-points with the role of "server" in order to execute actions on the resources identified with the URI sent by the client in the request.

These CoAP exchanges are not securely performed over the Transmission Control Protocol (TCP) transport protocol but are performed asynchronously using User Datagram Protocol (UDP). CoAP needs to provide a method of communication in which resource consumption is minimal, so establishing a virtual circuit over TCP to secure each connection between hosts is not considered energetically or bandwidth optimal. Still, CoAP needed to somehow provide security to the exchanges, in the form of:

- Confidentiality. Only the sender and receiver should be able to understand the messages exchanged.
- Authentication. Any host must be able to be certain that the host they are communicating with is indeed who they say they are.
- Integrity. Sender and receiver must be able to be certain that any message they have received has not been altered.

3.2.1. CoAP with DTLS

CoAP first resorted to the use of the Datagram Transport Layer Security (DTLS) protocol [6], derived from the well-known Transport Layer Security (TLS) protocol, which made it possible to secure a communication channel even if UDP is used as the transport layer protocol (as in the case of CoAP). This protocol provides the three elements necessary for security, allowing the use of different suites or methods of cryptographic material exchange. However, this protocol has an important characteristic, namely that the encryption happens at the transport layer, hiding the information of the application layer from intermediate network components.

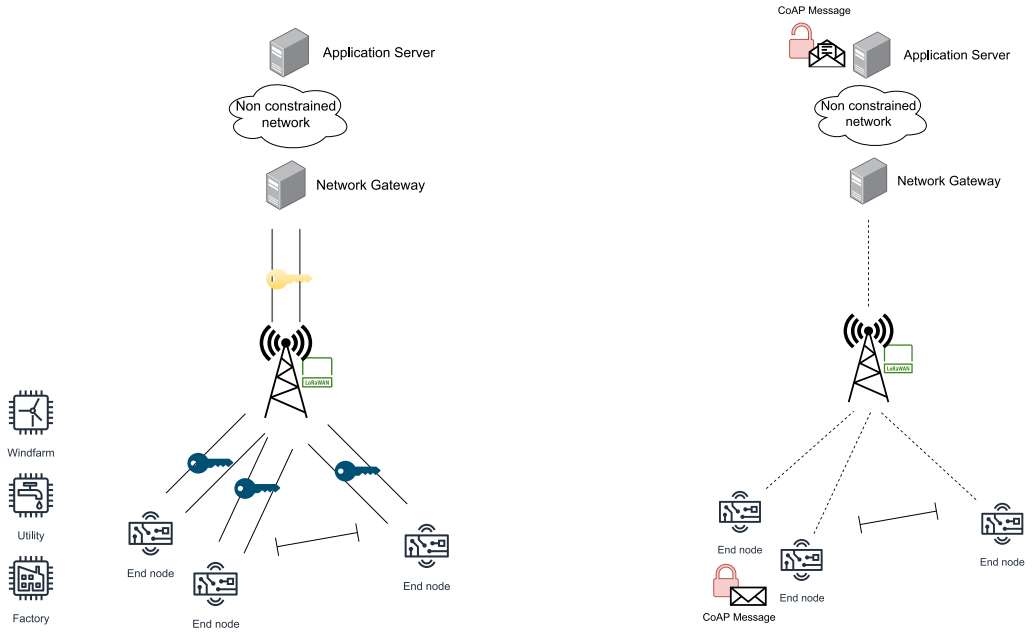
A packet passing through a middle box can undergo many treatments — e.g. a simple case would be a proxy agent routing a packet. It is likely that the router needs the value of some field of the CoAP message, but since the whole packet is encrypted, it is not readable. Therefore, it will be necessary for the DTLS channels to be created node-by-node (i.e. hop-by-hop security is being provided), as shown in Fig. 4(a), so the values are accessible. This exposes the communication to a major vulnerability at each middle box.

It is clear that DTLS has some deficiencies: the packet loses the at-rest protection, it is not focused on constrained devices, it requires negotiations between each pair of nodes in the path to establishing the DTLS channel, etc. What is sought is a method that provides end-to-end security, and this is precisely the origin of a CoAP extension protocol, OSCORE.

3.2.2. CoAP with OSCORE

Object Security for Constrained RESTful Environments (OSCORE) [5] is a security protocol operating on the application layer. Unlike DTLS, it provides end-to-end security at object level (Fig. 4(b)). This means that the CoAP PDU generated at the sending end once encrypted will not be decrypted until it reaches the receiving end. Eliminating all unnecessary processing in the middle boxes provides the in-transit and at-rest protection that DTLS could not ensure. In addition, OSCORE is specifically designed to work in constrained environments, so it is also working with packets with reduced overhead. This is achieved by selectively protecting the strictly necessary CoAP PDU fields.

Another characteristic of OSCORE's application level protection mechanism is allowing multicast communications, which is not possible in DTLS. That is, to send the same packet to thousands of devices, you need to make thousands of DTLS exchanges. On the other hand, OSCORE relies on a scalable mechanism by which keys can be derived for a group of devices. This is vital in the



(a) CoAP over DTLS

(b) OSCORE protected CoAP

Fig. 4. Different CoAP protection systems [23].

use case of distributing a firmware update using the radio link to a massive number of devices — Firmware Update Over-the-Air (FUOTA).

OSCORE needs cryptographic material that must be pre-provided at both ends. All OSCORE's cryptographic parameters revolve around the Security Context. The security context is the set of information elements necessary to carry out cryptographic operations in OSCORE. For this reason, every pair of communication endpoints share an identical security context. Initially, the Security Context is a simple set of parameters and keying material, used to derive the complete Security Context that will be ultimately used to process the COSE objects.

However, OSCORE provides modularity when it comes to choosing the appropriate algorithms and mechanisms. OSCORE does not define in its own standard how they generate the values of the initial context. Neither does it specify an HMAC-based Key Derivation Function (HKDF) to derive the keying material or an Authenticated Encryption with Associated Data (AEAD) algorithm for the object encryption itself.

As seen in Fig. 5 OSCORE is applied to a clear CoAP message and produces an OSCORE-protected CoAP message. The message fields of a CoAP message are classified as protected or unprotected fields; protected fields are put together along the CoAP payload in a COSE Object [24] to be encrypted and integrity assured between endpoints, forming the new OSCORE protected payload (also refereed as OSCORE payload). Meanwhile, unprotected fields are left to form the new CoAP header (also referred as OSCORE header) to allow middle boxes certain support operations. As no Integrity-protected headers are currently defined in the RFC [5] it is not contemplated in the figure.

Finally, we have to clarify that DTLS not being the optimal choice to apply in a LPWAN scenario is easy to explain. DTLS and OSCORE use cases are different, as DTLS is a more polyvalent protocol, it is the weapon of choice on protocol stacks proper of resource-richer architectures. Whereas OSCORE has fewer use cases in comparison, it is a much better suited to be applied on these types of massive device deployments over constrained bandwidth scenarios.

3.2.3. Upper layers in smart city and smart agriculture

Smart City and Smart Agriculture are the main use cases that frame this contribution. Nevertheless, our solution was designed to be agnostic and present an abstraction that allows its use in several other verticals with similar characteristics — inexpensive battery-powered devices connected to sensors or actuators. Besides, each organization may utilize the technology differently — the presented design does not limit the options of what data-format to employ. Some organizations prefer to encode the payload using a customized binary format, while other structured data formats are also popular, such as JSON or XML. Conversely, the use of efficient and compact binary representations are typically preferred – e.g. CBOR or Protocol Buffers (Protobuf) [25] – due to scalability in high-performance and bandwidth-constrained environments.

The specific contents of the upper application layer payloads heavily depends on the sensors. These are some of the typical payloads for sensors included in sensor networks for Smart City and Smart Agriculture: temperature, humidity, CO2, soil moisture, solar radiation, leaf wetness, pH, vermin trap state, motion sensors, GPS position locators, air quality, noise, waste management and smart parking.

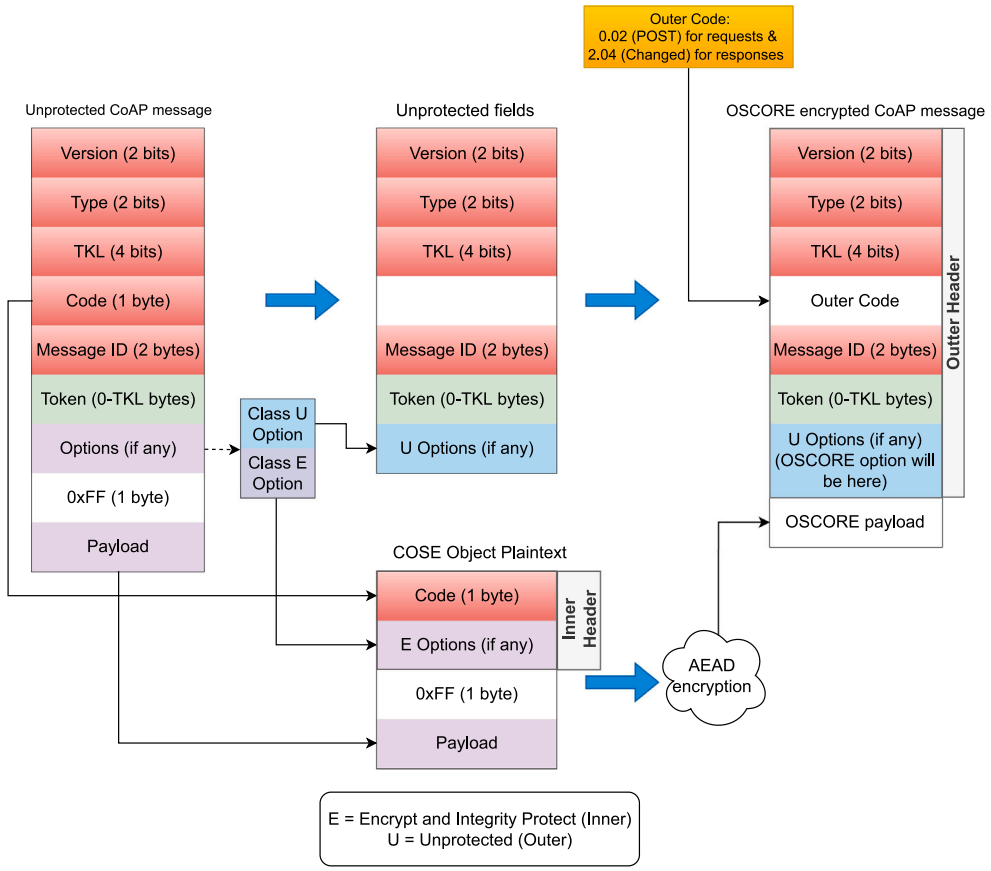


Fig. 5. OSCORE work process over CoAP message.

3.3. LPWAN gap analysis: Static context header compression

IETF work groups LPWAN/IPv6 over Networks of Resource-constrained Nodes (6Lo) have made great efforts to facilitate the integration of Internet Protocol version 6 (IPv6) as its vast address space and automated setup tools would support the expected device growth. However, the IP stack could not be replicated in IoT architectures, as it was designed for resource-rich networking environments. Consequently, the IETF was tasked with the development of a lightweight, IPv6-based protocol stack suitable for IoT-constrained devices, formed by components that have been specially designed for IoT scenarios.

Likewise, some LPWANs include security and privacy mechanisms at link level. These mechanisms are vendor-specific and do not address end-to-end security, fixing this issue outside the scope of their deployments. This is, while some LPWANs may introduce encryption mechanisms in their components, if the network is compromised, the application payload can be deciphered by the attackers. This is a noticeable shortcoming in the area of LPWAN security [26], highlighting the necessity for end-to-end encryption that is seamlessly integrable in LPWAN deployments.

As depicted by Fig. 6, Static Context Header Compression (SCHC) [9] is a state-of-the-art technology standardized by the IETF LPWAN work group to seamlessly integrate the transmission of IPv6/UDP (L3) packets over LPWAN technologies. SCHC introduces an adaptation layer clearly split in two differentiated sublayers, namely: Compression/Decompression (C/D) and an optional Fragmentation/Reassembly (F/R) mechanism. Please note that the SCHC has already been validated as an adaptation mechanism for LPWANs [27,28].

One of the identified LPWAN issues is the lack of support for native IP interoperability. They rely on vendor-specific mechanisms to abstract the communication with end-devices. However, seamless interoperation of constrained networks through the use of Internet protocols is tightly aligned with the overall vision of the IETF work groups developing standards aimed at constrained devices. As the number of IoT devices is expected to continuously grow, the IPv6 address space is considered necessary.

To do so, SCHC exploits the aforementioned LPWAN characteristics that allow efficient header compression. Thanks to the star-of-stars topology, both end-points of compression/decompression are well known in advance. Besides, since embedded firmware is unlikely to change, the developers are able to accurately predict the information flows that must be compressed.

Both ends that participate in a SCHC message exchange share in advance the same common piece of information known as a context. Each communication flow identified by the developers is represented as a compression/decompression (C/D) context rule.

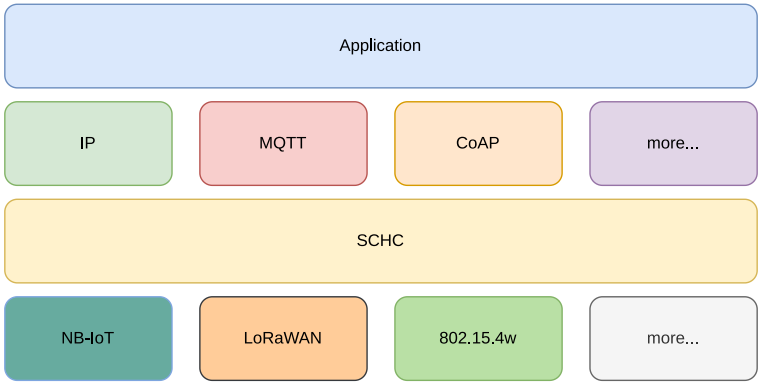


Fig. 6. LPWAN-based protocol stack when SCHC is in use.

As the name of the methodology implies, the context is static and does not require signalling messages to update or built the context at both ends. As a consequence, it is assumed that both ends share an identical copy of the copy at all times.

Additionally, IPv6 fixed maximum packet length of 1280 bytes can be a problem, as many LPWANs have a maximum packet unit in the order of tens of bytes. For instance, LoRaWAN may impose a maximum payload size of 51 bytes in its lowest data-rate configuration. For this reason, the "optional" Fragmentation/Reassembly (F/R) mechanism found in SCHC is extremely important, as it enables IPv6 support on LPWAN networks. In essence, SCHC's best characteristic is its ability to hide the architecture and interfacing details of the connected devices through any LPWAN, isolating the higher-level layers [6]. Instead, the devices are seamlessly integrated through Internet protocols.

Fig. 7 depicts the treatments that must be applied over a CoAP message, so it can be transported through a LPWAN network. Namely, it is a **message type 5** (see Section 4.4) that undergoes the SCHC treatments at two different levels: the network/transport layers and the application layer.

On this section the focus is on the work process of plain SCHC (from now on referred as **SCHC**) over network/transport level (identified by the "SCHC" light blue arrow) as this is the protocol layer SCHC was first created for. The process of SCHC over application level (identified by the "SCHC_CoAP" dark blue arrow) in this case is known as "SCHC for CoAP" (from now on referred as **SCHC_CoAP**) and it will be covered in Section 4.2.

This SCHC adaptation layer is divided into two sublayers:

- **SCHC C/D.** Identified by the "SCHC C/D" arrow.
  The IPv6 packet encapsulating the CoAP message is processed by the Compressor, generating an SCHC Packet: a packet with compressed IPv6/UDP headers (IPv6 payload is not altered during this stage). In this case, the SCHC Context's rule matching is so accurate that only 1 byte of the header is left, it's Rule Identifier.
  This stage is always applied if the SCHC context contains a rule that matches the packet structure. When the SCHC packet arrives at the receiving end, it is decompressed using the same SCHC context rules that were employed by the transmitter. As a result, an identical copy of the original IPv6 packet is obtained.
- **SCHC F/R.** Identified by the "SCHC F/R" arrow.
  If, and only if the LPWAN payload size available is not able to allocate the entire compressed SCHC packet, then a fragmentation mechanism is applied to split the SCHC packet in SCHC Tiles. On the receiving end, the tiles are reassembled to rebuild the SCHC packet.

4. OSCORE-protected CoAP over SCHC in LoRaWAN

Here we elaborate on the proposal of this article. In Section 4.1, we present the proposed architecture, which encompasses different entities involved in the secured CoAP exchange mapped to LPWAN entities. After that, in Section 4.2, we go into the details of the implementation of the proof of concept.

OSCORE messages leverages on COSE [24], which is regarded as a low-overhead mechanism, aimed at lightweight message communications. However, COSE supports a myriad of different stateless use cases, thus it is not as optimized for stateful security protocols. To partially mitigate this, RFC8613 Section 6.1 [5] defines a simple compression mechanism within the process of generating an OSCORE message, by removing redundant data. This focuses on reducing the size of the CoAP option that carries the COSE encrypted object in the OSCORE message option, as well as in the CoAP payload — which carries the COSE ciphertext.

The aforementioned OSCORE compression mechanism is able to reduce the per-packet overhead. However, it is contained in the OSCORE encoding procedure itself and outside the scope of the SCHC. Nevertheless, the SCHC CoAP RFC8824 [29] analyses in Section 6.4 these characteristics in order to further enhance the application of the SCHC mechanism for OSCORE packets.

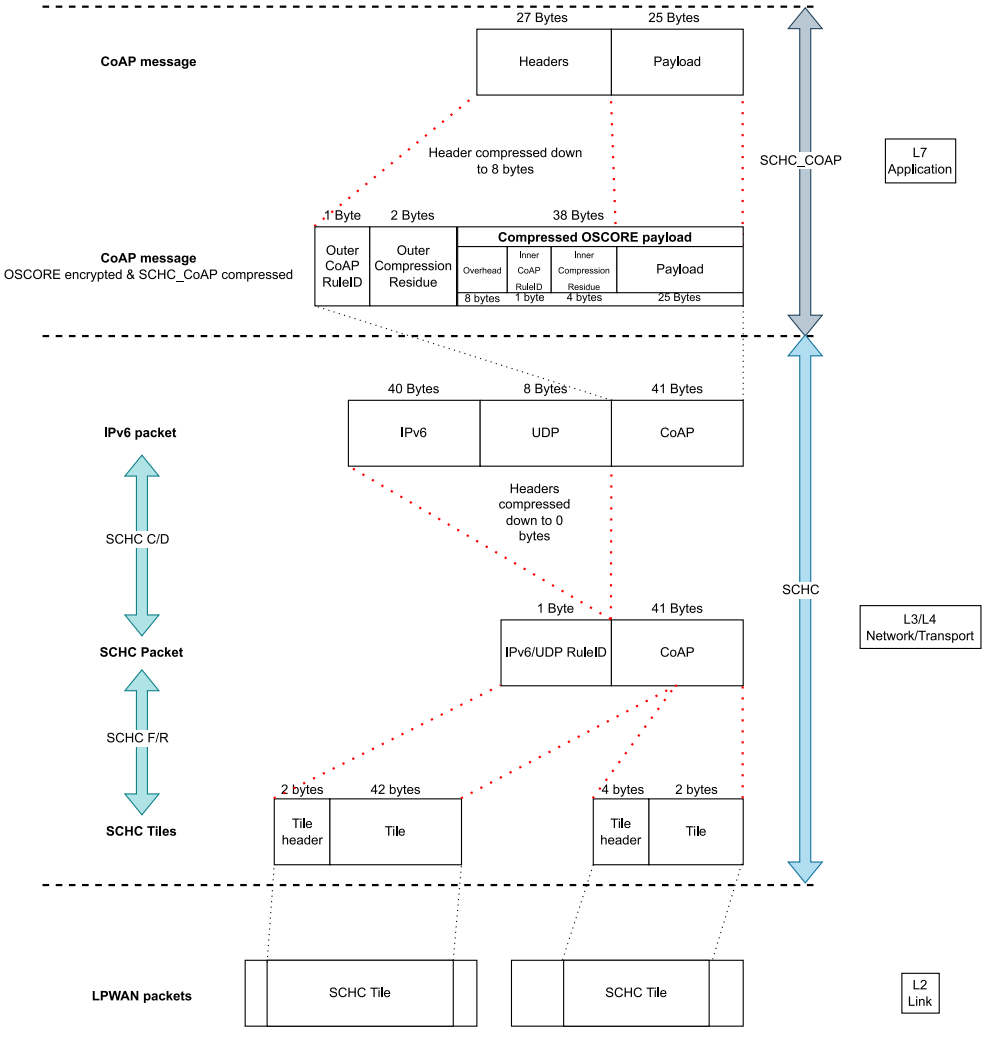


Fig. 7. Data treatment from L7 to L2 layer [27].

OSCORE capitalizes on the CoAP features that enable proxy and caching functionalities to fully achieve end-to-end encryption for CoAP messages. In order to do so, OSCORE messages differentiate among two different data units, namely: (i) the data that is needed in order to achieve proxy and caching functions, and (ii) the ciphered data that is protected by the encryption mechanism, and it is to be secured. These are already identified as outer context and inner context, respectively, in Section 7.2 of RFC8824 [29].

While briefly describing the desired characteristics at a high conceptual level, there are no clear definitions on how to implement this concept. Currently, there are no architecture proposals or data flows available in the IETF standardization results. To solve this gap, in this work, the problem of implementing the aforementioned concept of achieving CoAP object-level protection while retaining proxy and caching features is further analysed. As a result, a devised architecture design has been achieved, leveraging on the aforementioned technologies and constraints. This research proposal is showcased in Section 4.1, where each component is defined, specifying the employed communication protocols and data formats, as well as the features allocated to each one.

4.1. Proposed architecture

This architecture follows the guidelines of the IETF's LPWAN working group, and it is based on a protocol stack that makes use of LoRaWAN, SCHC, IPv6, OSCORE, CoAP_SCHC and CoAP protocols. As depicted in Fig. 8, this architecture provides a seamless integration of the OSCORE security protocol for CoAP over the IPv6-adapted LPWAN protocol stack.

The architecture presents a bidirectional dataflow: uplink communications from end-devices to a network/application server and vice versa (downlink).

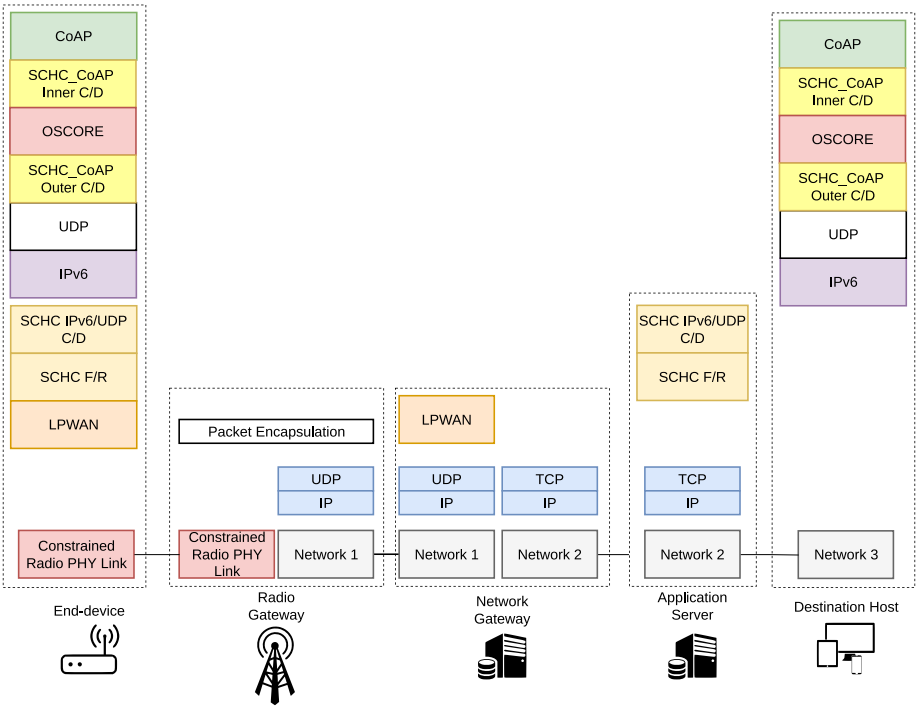


Fig. 8. Proposed architecture.

End-device: Represents any LoRa client device (e.g., sensors, actuators, etc.) that sends and/or receives information. Communicates with Radio-Gateways. It is the constrained end-point of the LPWAN system, SCHC F/R mechanism, SCHC IPv6 C/D mechanism, SCHC CoAP C/D mechanism and OSCORE encryption/decryption mechanism.

Radio-Gateway A transceiver antenna on the infrastructure side. On the uplink flow, the end point of the constrained link, it captures the end-devices traffic and sends it to NGW via IP. On the downlink flow, it is the starting point of the constrained link, receives the application side traffic and sends it to the convenient end-devices. It is a middle agent of the LPWAN system.

Network Gateway: Shortened to NGW. It acts as a router as it is the node directing the Radio-Gateway traffic towards/from the Internet. It is the non-constrained end-point of the LPWAN system. In the uplink part, the NGW encapsulates each SCHC fragment in an IP packet and routes it to the corresponding destination. On the downlink flow, it receives traffic from the internet and routes it to its corresponding Gateway.

Application Server: Application-layer processing agent. This node acts as the non-constrained end-point of the SCHC F/R mechanism and SCHC IPv6/UDP C/D mechanism. From a broad viewpoint, from this point onwards to the destination host, the LPWAN-related adaptations layer introduced by SCHC is not further required. On the uplink flow, this node reassembles the SCHC fragments received from the Network Gateway, reconstructing a compressed IPv6/UDP packet whose headers will be decompressed and sent outbound to his destination host. On the downlink flow, it compresses the headers from the IPv6/UDP packets heading to the end-devices, fragments the resulting SCHC packet and sends it to the constrained network.

Destination Host: It is the non-constrained end-point of the SCHC CoAP C/D mechanism and OSCORE encryption/decryption mechanism. Although this server will not store or process the CoAP message, it is refereed as “destination host” as from the constrained endpoint viewpoint, it is the host to reach out for, in order to communicate with the outside non-constrained networks. From the non-constrained endpoints viewpoint, this is the “LPWAN gateway” as any traffic sent to this host will route to the end-nodes as if they had native IPv6 support. The logic of this agent could be deployed alongside the “application server”.

4.2. SCHC for OSCORE

The proposed contribution follows the suggested procedure found in section 2 and section 7.2 of the “SCHC for CoAP” RFC [29], which consists in dividing the SCHC_CoAP compression process into two stages: **Inner Compression and Outer Compression**. To apply the compression using this approach, it is also needed to change the vision of the CoAP header structure. Instead of looking at a CoAP header as an absolute unit, it is split into two parts:

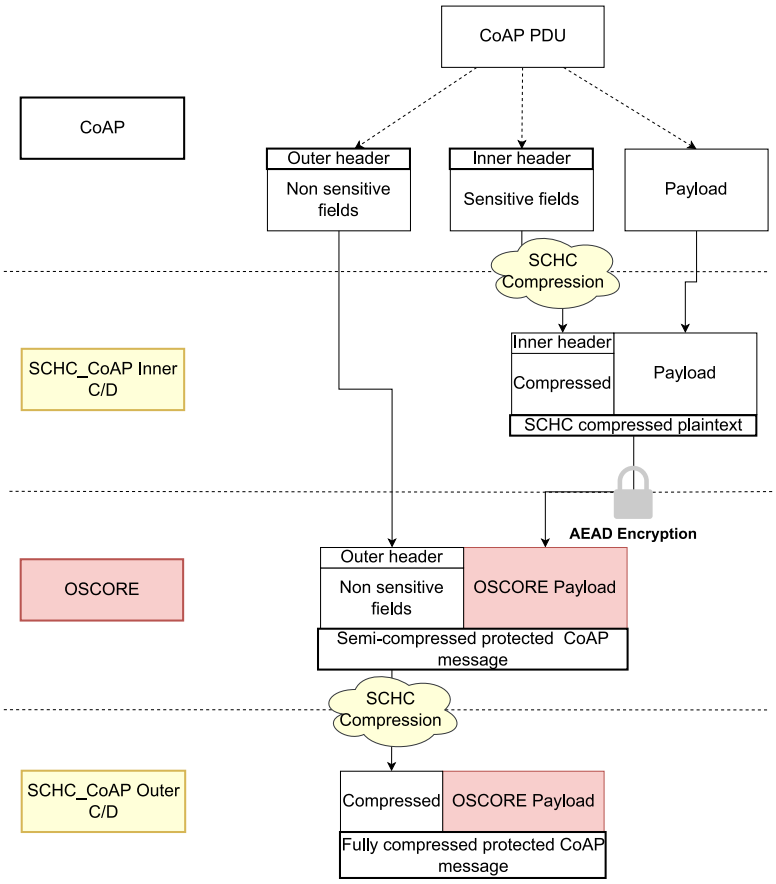


Fig. 9. CoAP message undergoing application of OSCORE and SCHC_CoAP mechanism.

- **Outer Header:** Formed by the CoAP header fields that do not include sensitive data, they will later become the OSCORE header . Outer, as it is the outermost header in the OSCORE message format hierarchy. It will occupy the earliest position when encoding an OSCORE message, followed by the OSCORE encrypted payload. This header is mostly formed by fields withdrawn from the original CoAP header and put together as the OSCORE header.
  As the OSCORE messages have the same header structure as CoAP messages, this header can be compressed with any standard SCHC library.
- **Inner Header:** Formed by the fields containing sensitive data from the CoAP headers, it will be later encrypted together with the CoAP payload. When extracted from the original CoAP header, the fields are put together, forming a new "reduced" header. If the encryption from an OSCORE message is somehow cleared out, the underlying virtual structure of the OSCORE message would be:

  – Header fields including non-sensitive data (Outer header)
  – Fields containing sensitive data (Inner header)
  – CoAP Payload

  Therefore Inner as it is the innermost header in the OSCORE message format hierarchy.

Using this SCHC_CoAP compression scheme allows for compression of both the **resulting OSCORE message after encryption (Outer Header compression)** but also the **Plaintext before encryption (Inner Header compression)**, as shown in OSCORE SCHC_CoAP [29, section 7.2]. Thus, allowing the compression of the "will be" OSCORE payload. This scheme will be reversed when decrypting: first, the initial decompression of the OSCORE header before decryption (Outer Header decompression), and later, the decompression of the resulting plaintext after decryption (Inner Header decompression). For each stage of C/D, a SCHC_CoAP rule is needed, as their fields will not be the same: a rule used for the C/D of the Outer Header and another rule is used for the C/D of the Inner Header.

Out of the two compression stages explained above, the **management of Inner Header** is the main issue when it comes to the incompatibility of regular SCHC_CoAP when applied to OSCORE-protected messages. The Inner header virtually exists only during a short period of time while inside the working process of the OSCORE reordering and encryption/decryption of CoAP messages:

- Before encryption: The sensitive fields are withdrawn from the CoAP header and put together forming the Inner Header. It is at that specific step of the procedure that the SCHC compression must be applied, before it is put together with the CoAP payload and the encryption process begins. In Fig. 9, the SCHC compression between "COAP" and "SCHC_CoAP inner C/D" layers represents this step.
- After decryption: When the OSCORE payload is decrypted, a plaintext is obtained. If the SCHC Compression had been applied on the encrypting end, this plaintext would be formed by a **SCHC compressed Inner Field** and a CoAP payload put together. The SCHC decompression must be applied right after splitting the CoAP payload from the compressed Inner Header, before OSCORE's rebuild process of the original CoAP header begins (using the fields from the Inner Header).

Within the IP stack, SCHC comprises an adaptation layer located lower than OSCORE, therefore OSCORE is completely unaware of SCHC actuation. Any previously developed OSCORE library assumes that the Inner Header only needs encryption on an outbound message and decryption on an inbound message. Hence, this is why there are no available methods to treat the Inner Header once the OSCORE encryption has started. In this work, this has been changed by applying enhancements over both OSCORE and SCHC libraries, allowing SCHC's actuation over the Inner Header mid through OSCORE's encryption/decryption procedure, which solves the difficulty of applying changes on the Inner Header.

To avoid network addressing errors, **Outer C/D Context and Inner C/D Context** were created to further extend the concept of **C/D Context**, allowing to group the rules on one context or the other depending on the rule's header applicability.

4.2.1. OSCORE option's compression

The OSCORE option must also be addressed. Encoded in the value of this option lies a compressed COSE object (containing flags, nonces, sequence numbers, etc.) as described in Section 2 of the OSCORE standard [5]. This is essential to initialize any encrypted client–server communication, and this object must be carefully managed, as it is bitwise codified, preventing the application of standard SCHC_CoAP C/D.

A method for C/D specially designed for this value can be seen as unnecessary. However, this is far from being the case since depending on the complexity of the keys used to initialize the communication, the OSCORE option's value can be several bytes long, as shown in the OSCORE standard, Section 6.1 [5]. This **extension of SCHC_CoAP** has also been included in the developed solution.

The **RFC "SCHC for CoAP"** Section 6.4 [29] covers the compression of this option's value, stating that the creation of four new **Field Identifiers** is enough to enable the application of SCHC over this COSE object. These four field identifiers are:

CoAP OSCORE_flags: Covers the first byte of the object, containing the OSCORE flag bits.

CoAP OSCORE_piv: Directly corresponds to the piv.

CoAP OSCORE_kidctx: Covers the piv's following 1 byte, which encodes the 'kid context' length and the 'kid context' itself.

CoAP OSCORE_kid: Directly corresponds to the kid.

4.3. Testbed

Here, we describe the hardware components employed in the real-life testbed used to validate and evaluate the impact of compressed and encrypted communications on constrained devices. The deployment was done using an end-device with LoRaWAN class A/C capabilities, a LoRaWAN radio gateway, a LoRaWAN Network Gateway and two servers, one acting as the Application Server, i.e. an SCHC middleware agent between the Internet and the SCHC masked network, and the other server acting as the Destination Host i.e. an OSCORE-CoAP_SCHC enabled application decrypting, decompressing and accessing the end-device data, both connected to the Internet.

- End-device The end-device is built around a SmartEverything Fox board powered by an ATMEL SAMD21 Ultra-low-power ARM Cortex-M0+ microcontroller. This microcontroller counts with 256 kB of programmable flash storage and 32 kB of main memory. Therefore, this device is classified as a class 2 constrained device according to the IETF classification [30]. This device is analogous to the more widespread and popular Arduino Zero board, sharing almost identical technical characteristics.
  To this board we connect a RN2483 radio chip designed by Microchip. This chip features LoRaWAN Class A and C capabilities and is prepared for transmission on the European radio band 868 MHz ISM. To this radio chip an antenna is attached, 2.2 dBi omnidirectional, transmitting at the highest (14 dBm) output power possible.
- Radio-Gateway or base-station The Radio-Gateway is the RisingHF RHF2S208 LoRaWAN gateway. Through the Semtech's SX1301 baseband chip, it processes the end-device's radio transmissions. This radio chip was attached to a 5 dBi omnidirectional antenna.
- Network Gateway The Network-Gateway is located in a server working on an Intel i5-7400 CPU running at 3.00 GHz with 8 GiB of system memory. This lorawan server v1.0 was set up using Chirpstack, a Free Open Source Software (FOSS) solution.[11]

[11] https://www.chirpstack.io/.

Table 1
Message type 1 fields.

CoAP field	Value
Version	1
Type	(0 × 0) Confirmable
Token length	0
Code	(0 × 01) GET
Message ID	0 × 7d35
Total header size	5 bytes
Payload	1 byte

- Application server and destination Host The application server and end-user application are executed on the same machine. This machine runs on an Intel i7-4790 CPU running at 3.60 GHz with 16 GiB of system memory.
  The SCHC development that is found on the application server and destination host, as well as the end-device, was majorly developed in C++. The new SCHC functionality introduced in this work enhances and leverages on the work presented in [27][12] (network-layer SCHC only). Meanwhile, the OSCORE functionality, implemented in the end-device and the destination host, is based and extends for compatibility with SCHC_OSCORE the work presented in [31].[13]

4.4. Evaluation of SCHC in OSCORE

The following proposed scenarios are aimed at reflecting message exchanges in a situation of deployment of IoT devices for a Smart Agriculture or a Smart City deployment use case. Please note that this contribution focuses on the impact of applying compression to the headers, in the case of OSCORE, before encryption. For this reason, the CoAP payloads envisioned in the following scenarios have arbitrary contents that reflect a wide array of common lengths found in Smart Cities and Smart Agriculture scenarios. For this reason, the optimization of the payload length it is outside the scope of this work it is considered an opaque array of bytes. Each scenario description below includes a discussion about its goal. On the one hand, message types 1–3 are for demonstration purposes only to clarify what is the impact of applying the proposed SCHC mechanism and OSCORE encryption to messages with different characteristics. On the other hand, message types 4–5 are designed to represent the nature of messages generated in the chosen Smart Cities and Smart Agriculture verticals. This is, they are aimed at representing typical sensor and actuator payloads, ranging from a few parameters to lengthier ones, such as a weather station.

In these scenarios, OSCORE communications are considered bidirectional — i.e. both uplink and downlink messages are considered. However, either if uplink or downlink transmissions are more predominant is entirely up to the particular use case scenario. In the case of Smart Agriculture and Smart Cities, the majority of the radio band occupation is typically found in uplink transmissions, where a massive deployment of constrained devices transmits small data units containing their sensor measurements. Whereas the downlink messages are aimed at triggering some kind of actuation by the device or changing a configuration, thus, are less frequent on a day-to-day basis.

As a consequence, LPWANs typically favour the performance of uplink transmissions, especially in the case of those employing unlicensed radio bands, where maximum transmission per hour per device is regulated and limited — in these scenarios, if the base stations need to continuously communicate with hundreds of devices, they will quickly expire their allocated bandwidth. These key ideas have been adapted to the web-based interaction allowed by CoAP and are represented in the following scenarios.

Message type 1. CoAP message with no options and 1 byte payload.

Use case: Base-line message. A small CoAP message with no real-world usage.

Goal of this scenario: Expose the amount of overhead added by OSCORE encryption, which will be the same for every other protected CoAP message independently of its size (10 bytes + x bytes OSCORE option + y bytes OSCORE option value), as the OSCORE context used to encrypt will be always the same one. Use it as a reference to indicate how SCHC is not applicable to the payload content (and part of the overhead) (see Tables 1 and 2).

Message type 2. CoAP message with 1 option and no payload.

Use case: CoAP Request launched by the non-constrained client (OSCORE server) in downlink direction, requesting a resource (e.g. an XML file with sensor measurements) from a constrained device (a soil moisture sensor at a farm).

Goal of this scenario: Compare hot the SCHC Compression effectiveness varies when the CoAP header is greater than the payload (see Tables 3 and 4).

[12] https://github.com/JesusSanchez456/SCHC_Client.
[13] https://github.com/Fraunhofer-AISEC/uoscore-uedhoc.

Table 2
Message type 2 fields.

CoAP field	Value
Version	1
Type	(0 × 0) Confirmable
Token length	0
Code	(0 × 01) GET
Message ID	0 × 7d35
Option ID 1	URI_PATH
Option value 1	temp.xml
Total header size	13 bytes

Table 3
Message type 3 fields.

CoAP field	Value
Version	1
Type	(0 × 0) Confirmable
Token length	0
Code	(0 × 01) GET
Message ID	0 × 7d35
Option ID 1	ETAG
Option value 1	0 × 24A0BB68
Option ID 2	URI_PATH
Option value 2	~sensors
Option ID 3	URI_PATH
Option value 3	Weather
Option ID 4	ACCEPT
Option value 4	(0 × 50) application/json
Total header size	28 bytes

Table 4
Message type 4 fields.

CoAP field	Value
Version	1
Type	(0 × 0) Confirmable
Token length	0
Code	(0 × 41) 2.05 Content
Message ID	0 × 7d36
Option ID 1	ETAG
Option value 1	0 × 24A0BB68
Total header size	10 bytes
Payload size	25–100 bytes

Message type 3. **Use case:** CoAP Request launched by the non-constrained client in downlink direction, requesting a resource (a file with the specific readings of **one of the sensors**) to a constrained device where several readings have to be transmitted each time — this is common in sensors where several physical parameters are measured, commonly weather stations (e.g. wind vector, wind velocity, temperature, air humidity, solar radiation, rainfall precipitation).

In this case there are 2 URI-PATH options because the URI is compound " sensors/weather", so it will be composed by two parts.

Included is the Etag option, a resource-local identifier for the local cache in the weather station. In this scenario, it is supposed that the non-constrained server requests readings from the weather station. Then, the sensor generates an answer JSON with the readings and calculates the Etag value as an opaque hash, answering with a CoAP RESPONSE containing the JSON and its hash. Next, the server saves the Etag in its local cache. Later, when the server requests the resource again, it adds the URI of the resource but this time it also adds the previous Etag. Next, when the CoAP REQUEST reaches the sensor, it will check against its value cache with the Etag sent to it in the request. Finally, if the readings have not changed, the sensor will not send the resource and simply answer with a 2.03 Valid code. In this case, the Etag itself is 4 bytes long. This options if extensively described in [9, Section 5.10.6.]

Finally, the Accept option is sent, indicating the format in which the resource is requested. In this case, "application/json".

Goal of this scenario: It extends the analysis from message type 2, being this message analogous but going from one option to four options. It will be analysed here what is the impact of SCHC, since a large amount of bytes is dedicated to the CoAP headers.

Message type 4. CoAP message with 1 option and variable payload.

Use case: CoAP Response launched by the constrained device in the Uplink direction containing in its payload the requested resource (readings obtained) accompanied by the Etag option.

Goal of this scenario: Reference probably one of the most common scenarios, a message with some compressible part but with a payload that will remain fixed. Compare it with previous message types to explain how we ended up arriving at the results we obtained.

Table 5
Message type 5 fields.

CoAP field	Value
Version	1
Type	(0 × 0) Confirmable
Token length	0
Code	(0 × 41) 2.01 Created
Message ID	0 × 7d36
Option ID 1	ETAG
Option value 1	0 × 24A0BB68
Option ID 2	LOCATION_PATH
Option value 2	g1_moist.xml
Option ID 3	CONTENT_FORMAT
Option value 3	(0 × 29) application/xml
Option ID 4	MAX_AGE
Option value 4	1
Total header size	27 bytes
Payload size	25–100 bytes

Table 6
Test-bed scenarios.

Scenario	Direction	CoAP type	Code	TKL	CoAP options	Payload length
Message type 1	Downlink	CON	0.01 GET	0	0	1
Message type 2	Downlink	CON	0.01 GET	0	1	0
Message type 3	Downlink	CON	0.01 GET	0	4	0
Message type 4	Uplink	CON	2.05 Content	0	1	25
	Uplink	CON	2.05 Content	0	1	50
	Uplink	CON	2.05 Content	0	1	100
Message type 5	Uplink	CON	2.01 Created	0	4	25
	Uplink	CON	2.01 Created	0	4	50
	Uplink	CON	2.01 Created	0	4	100

Several samples will be taken with variations in payload length to represent the applicability of the project to IoT devices of different capacities:

- 25 bytes: a temperature sensor of a deployment in a (Smart Building).
- 50 bytes: water quality sensor of a farm (Smart Agro)
- 100 bytes: readings from a meteorological station (Smart City)

Message type 5. CoAP message with 4 options and variable payload. It includes the fields (see Table 5):

Use case: CoAP Response triggered by the constrained device in the Uplink direction.

It is generated in response to a "GET" sent by the client. In this case, the GET message is used to signal the sensor to process its collected data in order to generate a file which will be made available as a resource. The CoAP response would be used to confirm that the resource has been created, containing in its payload the resource created (readings obtained).

Options such as the "Content-Format" are added to indicate in which format the resource contained in the payload is encoded. The "Location-Path" is also added, indicating the path to the new resource where the endpoint has left the collected data. Finally, the "Max-Age" is added, indicating the maximum time that the resource can be considered valid.

Goal of this scenario: Compare with the previous message type. This case study is almost identical to the previous one, but significantly increases the uncompressed message size by adding 3 more options. Applying compression will result in packets of exactly the same size as in the previous message type, with the original size being larger.

A summary of all the aforementioned scenarios is showcased in Table 6.

4.4.1. Results

This section shows the acquired results from running all the scenarios defined in the previous section. These results showcase the impact of applying the SCHC for OSCORE mechanism to these packets.

Please note that all of these results focus on showcasing the impact with regard to the CoAP Protocol Data Unit (PDU) — i.e. the first byte shown in the comparisons corresponds to the first byte of the CoAP header. In previous works [27,28], it is assumed that the IPv6/UDP headers have been compressed to a single byte using the regular SCHC mechanism, as shown in the SCHC RFC Appendix A, Figure 26 [9], where the entirety of IPv6/UDP headers are compressed.

The evaluation metrics proposed to measure the impact on the overall performance of the contributed solution are the compared length of the messages that must be transmitted over the constrained radio link in conjunction with the CPU time usage and energy required to perform SCHC and OSCORE operations — as described in Section 4.4.2. It has been observed by previous works that reducing the size of the transmitted messages improves the overall aggregated network performance by reducing the chance of

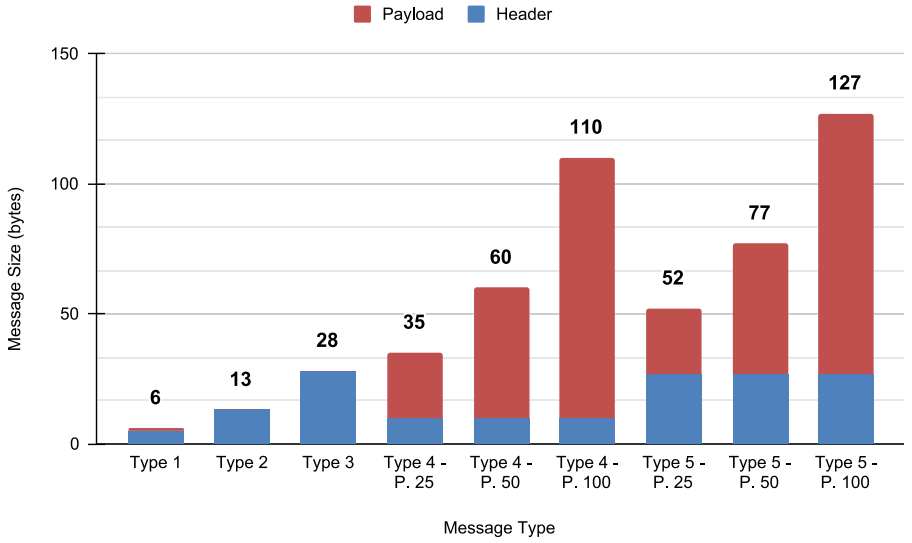


Fig. 10. All message types unencrypted and uncompressed — CoAP PDU.

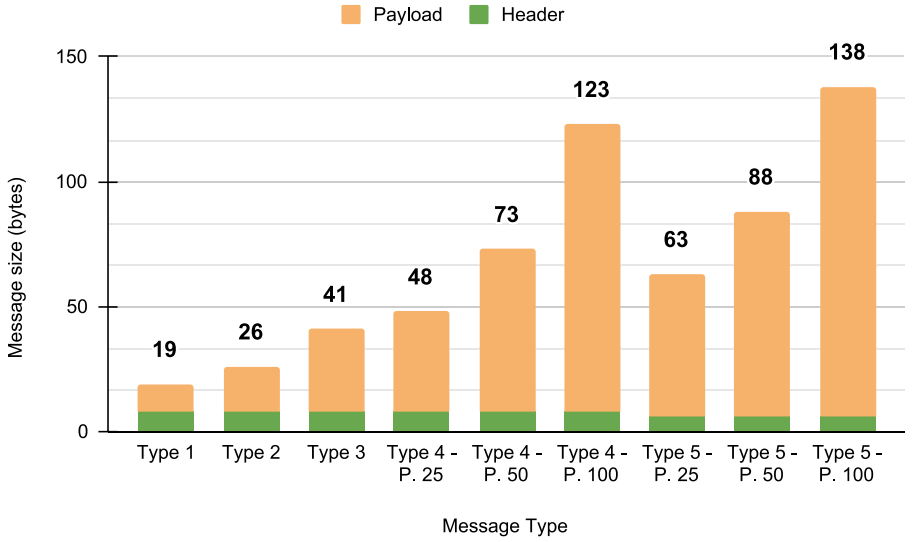


Fig. 11. All message types encrypted and uncompressed.

different messages colliding at the receptor, thus improving the packet delivery ratio [27,32]. In turn, this reduces both the necessity for fragmentation and retransmissions due to fewer lost packets. Additionally, transmitting is one of the activities that consumes the highest amount of energy in a battery-powered device. Since most of the constrained devices connected through LPWANs are going to be battery-powered, the reduction of bandwidth usage is a critical performance metric [33], which will be further improved thanks to the aggregated reduction of the packet sizes. The major aggregated performance metric for LPWANs is the number and length of packets [32].

First, Fig. 10 shows scenarios Type 1–5, showcasing the proportion of which is CoAP Headers and CoAP payload. Please note that, since SCHC is a header compression mechanism, no CoAP payload compression is achievable. As aforementioned, these payloads are considered as opaque arrays of bytes that must be carried without modifications from end to end.

Then, Fig. 11 shows scenarios Type 1–5 after applying the encryption done by OSCORE. Please note that the overall length of each message is increased compared to Fig. 10. This is due to the overhead introduced by OSCORE to achieve object encryption. Additionally, due to how OSCORE operates, the encrypted information is now transformed into a payload, including fields that were previously part of the unencrypted CoAP message.

Next, Fig. 12 presents scenarios Type 1–5 after applying both the OSCORE object encryption and the SCHC compression. Please note that the OSCORE encryption procedure has to be previously applied in order to employ the main contribution of this work.

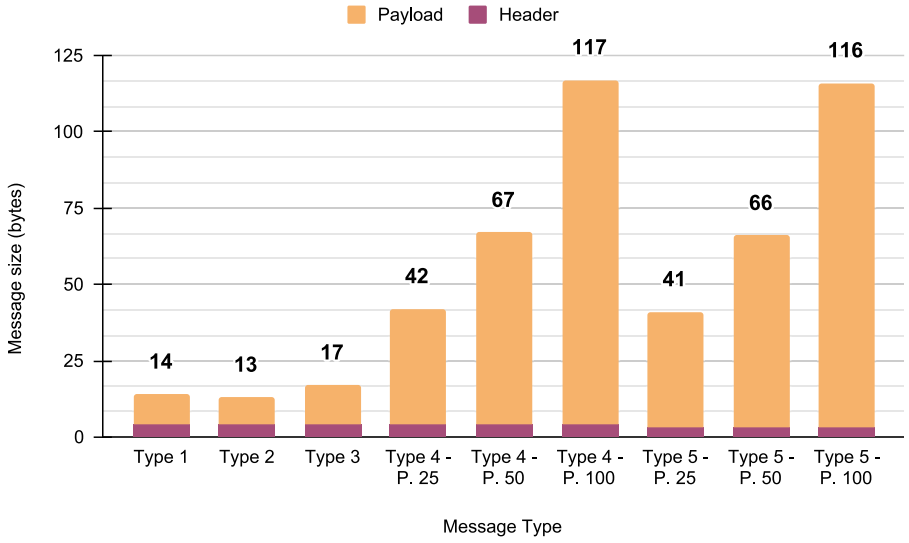


Fig. 12. All message types encrypted and fully compressed.

Table 7
CPU time usage and energy consumption of SAMD21 MCU.

Stage	t (µs)	Energy consumption (nWh)
CoAP Inner Header Compression	1819,92	18,85
OSCORE Encryption	7989,22	82,76
CoAP Outer Header Compression	3288,85	34,07
IPv6/UDP Header Compression	2025,21	20,99
Total	15 123,22	115,66

Later, Fig. 13(a)–15 showcase the complete set of experimentation results for each scenario, which include the following columns: (i) the original CoAP Protocol Data Unit (PDU), (ii) the application of the SCHC compression for IPv6/UDP/CoAP as shown in the SCHC for CoAP standard [29], (iii) the message after applying OSCORE, (iv) the encrypted message after applying outer SCHC compression only, (v) the encrypted message after applying inner SCHC compression only, and (vi) the encrypted message after applying both the outer and inner SCHC compression of OSCORE.

Finally, Fig. 16 presents the final comparison summary of previous tables side-to-side for each type, including (i) the original unencrypted and uncompressed length of the CoAP PDU, (ii) the message after applying OSCORE object encryption, and (iii) the encrypted message after applying the proposed mechanism of SCHC OSCORE inner and outer compression.

4.4.2. Evaluation of CPU time and energy consumption

In the previous section, it has been observed how for each use case, there is a reduction in packet size to a greater or lesser extent. What this reduction in size translates into in terms of CPU time, time of air (ToA) and energy consumption will be analysed below.

Table 7 and Fig. 17 showcase the data for the CPU evaluation was gathered using the real-life end-device when applying the OSCORE and SCHC_COAP treatments over a CoAP message type 5 [Section 4.4](most demanding use case, 4 options, 100 bytes payload), and SCHC C/D and F/R over the encapsulating IPv6 packet. The analysis differentiates the time that takes for applying the CoAP inner compression, the CoAP outer compression, the encryption, and the IPv6/UDP compression.

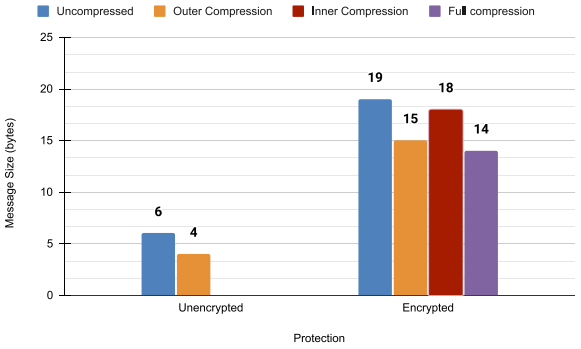
The energy consumption has been calculated from the nominal voltage (3.3 V) and intensity (11.3 mA) specified on the MCU datasheet[14] and the measured CPU time used to process the message type 5 [4.4]. The CPU time has been measured through software and exclusively reflects the time dedicated to complete that specific stage/operation

Analogous to the aforementioned results, the impact of applying OSCORE and SCHC_COAP for a message type 5 has also been studied in the non-constrained end-node. Hence, the resulting outputs are depicted in Table 8 and Fig. 18.

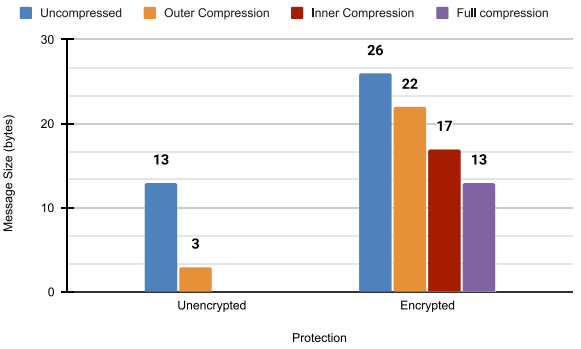
4.4.3. Evaluation of time-on-air

In the three following Tables 9, 10, 11 the ToA under different circumstances has been measured. These tables are critical when it comes down to appreciating the resource usage saving introduced through the proposed architecture.

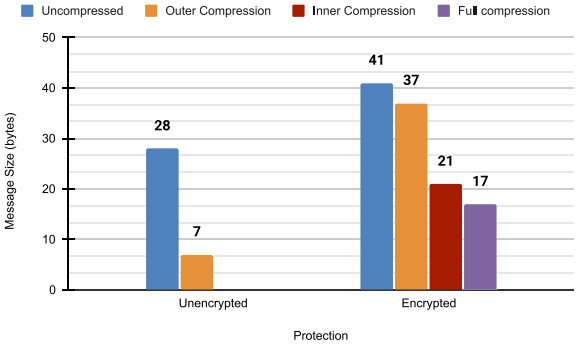
[14] https://ww1.microchip.com/downloads/en/DeviceDoc/SAM_D21_DA1_Family_DataSheet_DS40001882F.pdf.



(a) Message type 1



(b) Message type 2



(c) Message type 3

Fig. 13. Message Type 1–3 results.

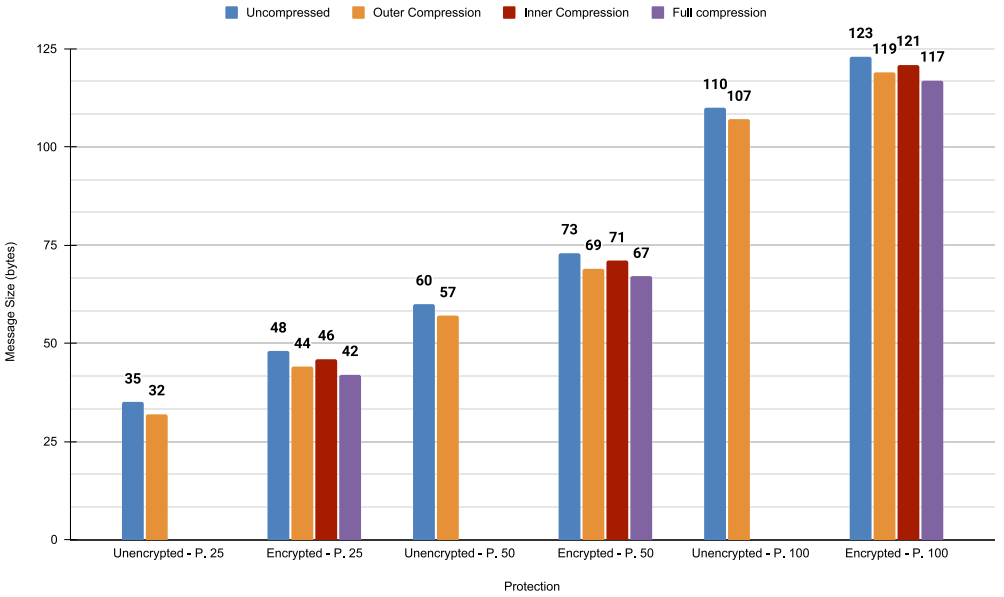


Fig. 14. Message type 4.

The ToA has been measured using the European radio band 868 MHz ISM. In this case, we have two independent variables, namely: the payload (Section 4.4) and the available Data Rates (DR).

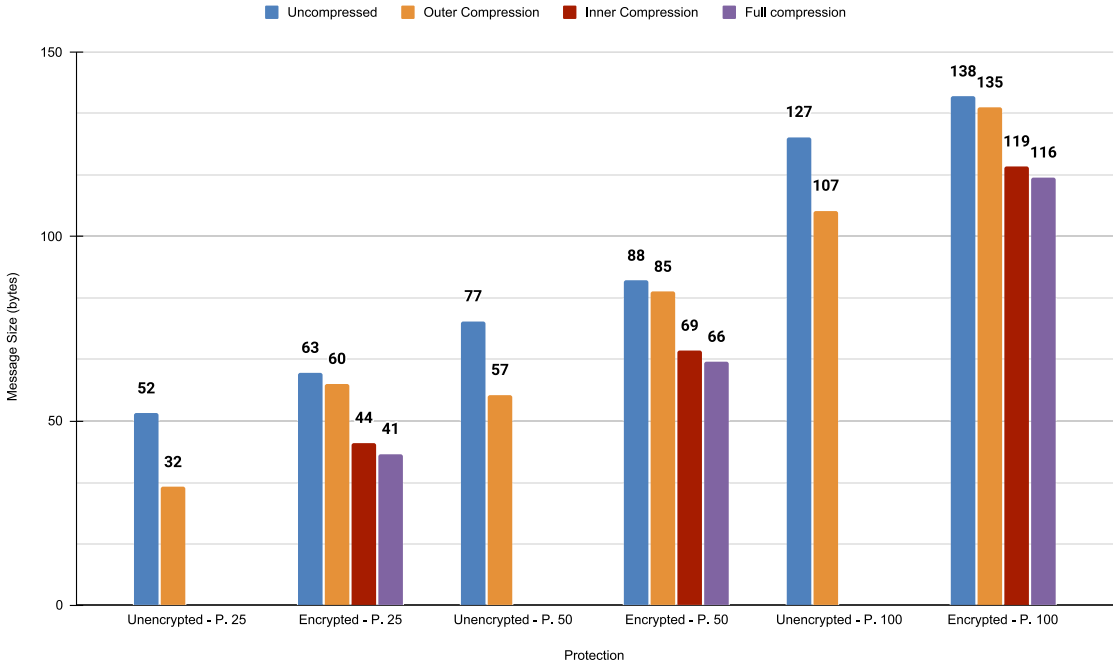


Fig. 15. Message type 5.

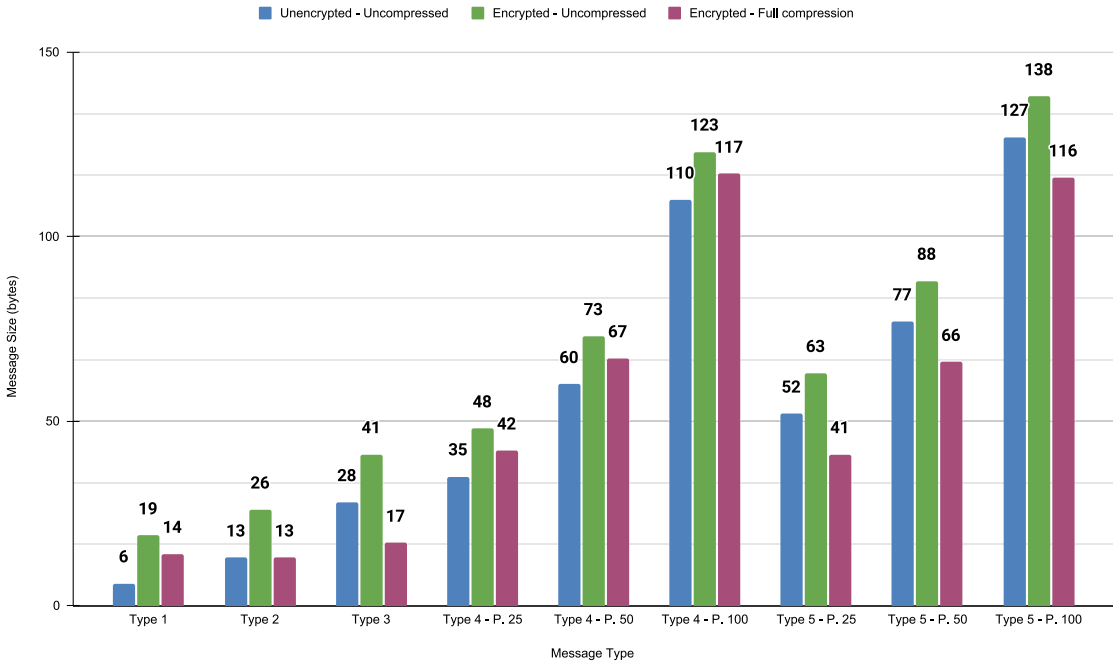


Fig. 16. Final comparison of all message types with different compression rates — CoAP PDU.

Table 9 shows the ToA when the data is transmitted as-is (No SCHC - No SCHC_CoAP - No OSCORE). Those cells marked in yellow shows the theoretically time that it would take for the corresponding LPWAN packet to be transmitted. Theoretically only, as in reality the packet cannot be transmitted through the LPWAN network as it size surpasses the FMRPayload (Section 3.1) [3] size allowed in the employed LoRaWAN radio band regulations [22] (e.g. Type 3 message has a CoAP header of 28 bytes and a 40 bytes IPv6 header surpassing the 51 bytes maximum payload size on DR0).

Table 8
CPU time usage and energy consumption of non-constrained end-node.

Stage	t (µs)
CoAP Inner Header Compression	3.53
OSCORE Encryption	17.64
CoAP Outer Header Compression	3.12
IPv6/UDP Header Compression	5.52
Total	29.81

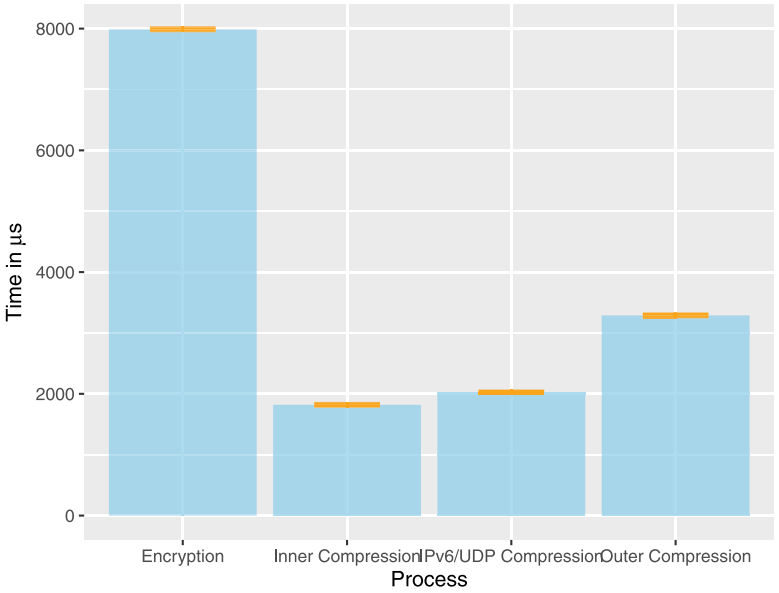


Fig. 17. Time (µs) dedicated to the different processes in the constrained node.

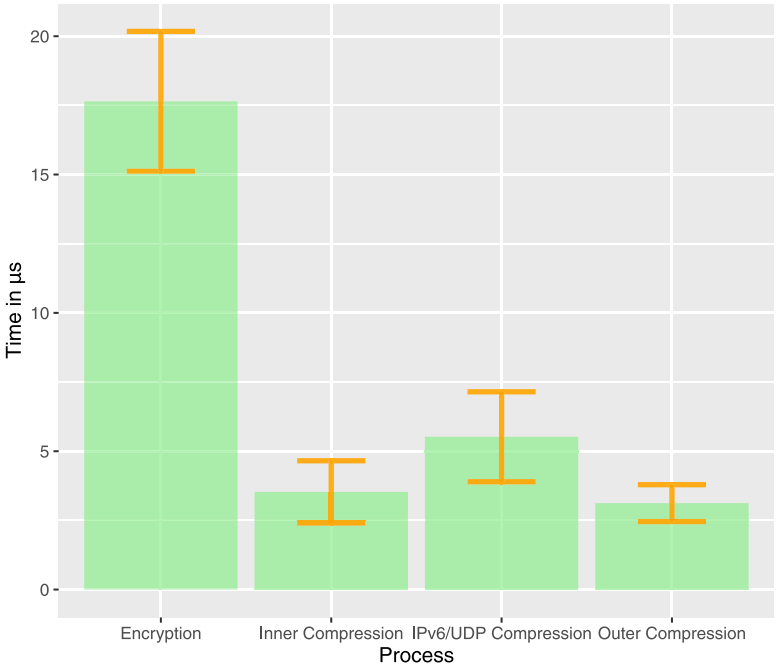


Fig. 18. Time (µs) dedicated to the different processes in the non-constrained node.

Table 9
ToA transmitting plain CoAP messages without encryption nor compression.

Message type	ToA (ms)						
	DR0	DR1	DR2	DR3	DR4	DR5	DR6
Type 1	2629,6	1396,7	657,4	349,2	195,1	107,8	53,9
Type 2	2793,5	1560,6	739,3	390,1	215,6	123,1	61,6
Type 3	3285	1806,3	862,2	451,6	256,5	143,6	71,8
Type 4 - 25 bytes payload	3612,7	1970,2	903,2	492,5	277	153,9	76,9
Type 4 - 100 bytes payload	6070,3	3280,9	1517,6	840,7	461,3	261,4	130,7
Type 5 - 25 bytes payload	4104,2	2297,9	1026	574,5	318	179,5	89,7
Type 5 - 100 bytes	6561,8	3608,6	1640,4	902,1	512,5	287	143,5

Table 10
ToA transmitting OSCORE encrypted CoAP messages with no compression.

Message type	ToA (ms)						
	DR0	DR1	DR2	DR3	DR4	DR5	DR6
Type 1	3121,2	1642,5	780,3	410,6	236	128,3	64,1
Type 2	3285	1806,3	821,2	451,6	246,3	138,5	69,2
Type 3	3776,5	2052,1	944,1	513	287,2	164,1	82
Type 4 - 25 bytes payload	3940,4	2215,9	1026	554	307,7	174,3	87,2
Type 4 - 100 bytes payload	6398	3526,7	1640,4	902,1	502,3	281,9	140,9
Type 5 - 25 bytes payload	5742,6	3117,1	1435,6	799,7	440,8	251,1	125,6
Type 5 - 100 bytes	7053,3	3854,3	1763,3	963,6	543,2	307,5	153,7

Table 11
ToA transmitting OSCORE encrypted CoAP messages with compression.

Message type	ToA (ms)						
	DR0	DR1	DR2	DR3	DR4	DR5	DR6
Type 1	1646,6	823,3	411,6	226,3	123,4	66,8	33,4
Type 2	1646,6	823,3	411,6	205,8	113,2	61,7	30,8
Type 3	1646,6	905,2	452,6	226,3	123,4	71,9	36
Type 4 - 25 bytes payload	2465,8	1396,7	657,4	349,2	195,1	107,8	53,9
Type 4 - 100 bytes payload	5251,1	2871,3	1312,8	717,8	379,4	215,3	107,6
Type 5 - 25 bytes payload	2465,8	1341,8	616,4	349,2	184,8	102,7	51,3
Type 5 - 100 bytes	5251,1	2789,4	1271,8	697,3	379,4	215,3	107,6

Table 12
RN2483 energy consumption while transmitting the encrypted CoAP PDUs.

Message type	Energy consumption (nWh)						
	DR0	DR1	DR2	DR3	DR4	DR5	DR6
Type 1	93,77	49,80	23,44	12,45	6,96	3,84	1,92
Type 2	99,61	55,65	26,36	13,91	7,69	4,39	2,20
Type 3	117,14	64,41	30,74	16,10	9,15	5,12	2,56
Type 4 - 25 bytes payload	128,82	70,25	32,21	17,56	9,88	5,49	2,74
Type 4 - 100 bytes payload	216,46	116,99	54,12	29,98	16,45	9,32	4,66
Type 5 - 25 bytes payload	146,35	81,94	36,59	20,49	11,34	6,40	3,20
Type 5 - 100 bytes	233,98	128,68	58,49	32,17	18,27	10,23	5,12

Table 10 shows the ToA when the data is object-encrypted, but no further optimization is applied (No SCHC - No SCHC_CoAP - OSCORE). This increases the difficulties to freely transmit data, as the introduction of OSCORE also introduces an overhead, which in this case renders message type 1 unsuitable for transmission on certain DRs.

Finally, Table 11 shows the results gathered when using the fully optimized solution (SCHC - SCHC_CoAP - OSCORE). In this particular table, cells in yellow mark those cases where the packet would not be able to be transmitted if SCHC F/R mechanism was not available.

4.4.4. Evaluation of RN2483 energy consumption

On these two last tables, the energy consumption of the radio chip is observed. Depending on the DR used, the used energy varies(more ToA equals more consumption). The consumption is expressed in nanowatts per hour (nWh).

Table 12 contains the energy consumption of the chip when the transmitted data has not received any prior treatment (No SCHC - No SCHC_CoAP - No OSCORE). Meanwhile, Table 13 contains the energy consumption of the chip when the data has been properly securitized and compressed to be sent through the LPWAN network (SCHC - SCHC_CoAP - OSCORE).

Table 13
RN2483 energy consumption while transmitting the encrypted and compressed CoAP PDUs.

Message type	Energy consumption (nWh)						
	DR0	DR1	DR2	DR3	DR4	DR5	DR6
Type 1	58,72	29,36	14,68	8,07	4,40	2,38	1,19
Type 2	58,72	29,36	14,68	7,34	4,04	2,20	1,10
Type 3	58,72	32,28	16,14	8,07	4,40	2,56	1,28
Type 4 - 25 bytes payload	87,93	49,80	23,44	12,45	6,96	3,84	1,92
Type 4 - 100 bytes payload	181,40	99,47	46,81	25,60	13,53	7,68	3,84
Type 5 - 25 bytes payload	87,93	46,88	21,98	12,45	6,59	3,66	1,83
Type 5 - 100 bytes	181,40	99,47	45,35	24,86	13,53	7,68	3,84

4.4.5. Discussion

Analysing the acquired results, as shown in Fig. 16, it can be observed that when applying the SCHC OSCORE compression, there is always an improvement in message length with regard to the encrypted version. Most noticeable with Type 2 and Type 3, with compression ratios of 50.00% and 41.46% respectively. As a consequence, these results validate the application of the SCHC-OSCORE mechanism whenever encryption is employed.

Furthermore, for some instances, the SCHC-OSCORE compression yields shorter message sizes than their unencrypted counterpart CoAP message — this is the case for Message Types 3–5. This is a highlight of the proposed contribution, which validates that the use of object encryption does not necessarily impose a bandwidth penalty.

As shown in Section 4.4.2, the cost of applying the proposed innovations is almost negligible (orders of magnitude smaller) when compared with the time-on-air savings achieved. Please note that gateways in LPWAN networks do not inspect the contents of the payloads that they forward to the non-constrained side of the network. Thus, gateways do not suffer a performance penalty when applying the SCHC or OSCORE mechanisms to the messages, because all payloads are treated as opaque. Furthermore, the payload reduction attained in the results has a positive impact on the overall network infrastructure due to the messages being shorter. With regards to the computational overhead introduced at the non-constrained end-node, Fig. 18 and Table 8 show that the time required is approximately three orders of magnitude smaller than the constrained counterpart. As a consequence, the innovation introduced in this proposal incur negligible computational overhead — although this contribution focuses on aggregated bandwidth and energy efficiency on the constrained side of the deployment as the key performance indicators.

Nevertheless, the impact of SCHC-OSCORE compression may yield different result ratios if there are more bytes dedicated to payload than headers. This is the case of Message Type 4, where the impact of SCHC-OSCORE compression is lower compared to Type 5. This is because SCHC, as a header compression mechanism, does not treat bytes in the CoAP payload, considering those an opaque set of octets that must be carried from end-to-end. Hence, since Type 4 messages have lower CoAP headers, the impact is lower.

After the result analysis, it could be stated that, overall, applying SCHC full compression improves the aggregated performance of any LPWAN deployment by reducing the required energy to transmit end-to-end secured CoAP objects, while also improving the bandwidth usage of the deployment, allowing more devices to be installed without causing radio collisions. These results can also be used by network administrators by extrapolating the acquired figures and do a better study of the amount of saved bandwidth achieved through SCHC-OSCORE compression.

5. Conclusions and future work

This work achieved the implementation of the envisioned design using a real-life test-bed which includes Arduino-based constrained devices and a real-life industry-ready LPWAN technology deployment – i.e. LoRaWAN – demonstrating that this solution can be included in resource-constrained devices, such as those based on System-on-Chip (SoC) Microcontroller (MCU) architectures. This is achievable thanks to the relatively small code footprint introduced by both the developed SCHC and OSCORE libraries.

Also, the employment of OSCORE leverages on lightweight cryptographic primitives that curve the overall cost of massive deployments, thanks to requiring low computational resources, as opposed to other heavier cryptographic mechanisms that rely on non-constrained devices or networks, such as those typically found in regular computer networks — e.g. HTTP over TLS. The resulting architecture achieved in this work is easily replicable in any LPWAN scenario, thanks to it following the generic architecture devised by the IETF [19].

The experimentation results contribute to the ongoing validation of the employment of IETF openly standardized protocols aimed at enabling LPWAN-based deployments' interoperability with IP networks. Through the combination of a state-of-the-art header compression mechanism for integrating LPWAN networks and a CoAP Object encryption, secure and private end-to-end communications are achieved with a reasonable bandwidth cost. This allows the efficient transmission of sensor and actuator messages with centralized application platform servers, as those found in the Smart City and Smart Agriculture scenarios. Besides, due to how this work contribution can be seamlessly integrated into an LPWAN architecture, the proposed result can also be employed in other verticals that benefit from the LPWAN paradigm, such as e-Health, Intelligent Transportation Systems (ITS), or Industry 4.0.

Future work is planned for the analysis and application of SCHC schemes to additional use cases, scenarios and protocols, to continue leveraging the benefits of existing protocols in LPWAN technologies.

In this sense, there are other protocols that can leverage SCHC to reduce the number of bytes sent over the air. Future work is planned for the application of SCHC to authentication algorithms, key management protocols and similar security mechanisms, in order to bring more flexibility to secure LPWAN. This has to be studied to understand where SCHC can be used, and where we are limited by the own protocols, as security-reated protocols and the tools used, may limit the applicability of compression mechanisms such as SCHC, for instance, when random values or ciphered information is exchanged.

Declaration of competing interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Data availability

Data will be made available on request

Acknowledgements

This work was supported in part by the European Commission through EU Projects, the NEPHELE under Grant 101070487, the REWIRE under Grant 101070627, the PRECEPT under Grant 958284, the MASTERPIECE under Grant 101096836; partially by the Torres Quevedo grant by the Ministry of Science and Innovation of Spain under grant PTQ2021-011897; and partially by Grant PID2020-112675RB-C44 funded by MCIN/AEI/10.13039/5011000011033 (ONOFRE Project).

References

[1] Open Mobile Alliance, Lightweight machine to machine technical specification - LwM2M, 2020, URL http://www.openmobilealliance.org/release/LightweightM2M/V1_2-20201110-A/OMA-TS-LightweightM2M_Core-V1_2-20201110-A.pdf.
[2] Open Mobile Alliance, Lightweight machine to machine - LwM2M v1.1, 2019, URL http://www.openmobilealliance.org/release/LightweightM2M/Lightweight_Machine_to_Machine-v1_1-OMASpecworks.pdf.
[3] O. Garcia-Morchon, S. Kumar, M. Sethi, Internet of Things (IoT) Security: State of the Art and Challenges, Tech. Rep., 2019, pp. 5–10, http://dx.doi.org/10.17487/RFC8576, URL https://www.rfc-editor.org/info/rfc8576.
[4] C. Bormann, P.E. Hoffman, Concise Binary Object Representation (CBOR), RFC 8949, RFC Editor, 2020, http://dx.doi.org/10.17487/RFC8949, URL https://www.rfc-editor.org/info/rfc8949.
[5] G. Selander, J.P. Mattsson, F. Palombini, L. Seitz, Object Security for Constrained RESTful Environments (OSCORE), RFC 8613, RFC Editor, 2019, http://dx.doi.org/10.17487/RFC8613, URL https://www.rfc-editor.org/info/rfc8613.
[6] E. Rescorla, N. Modadugu, Datagram Transport Layer Security Version 1.2, RFC 6347, RFC Editor, 2012, http://dx.doi.org/10.17487/RFC6347, URL https://www.rfc-editor.org/info/rfc6347.
[7] M. Vučinić, B. Tourancheau, T. Watteyne, F. Rousseau, A. Duda, R. Guizzetti, L. Damon, DTLS performance in duty-cycled networks, in: 2015 IEEE 26th Annual International Symposium on Personal, Indoor, and Mobile Radio Communications, PIMRC, IEEE, 2015, pp. 1333–1338.
[8] Z. Shelby, K. Hartke, C. Bormann, The Constrained Application Protocol (CoAP), RFC 7252, RFC Editor, 2014, http://dx.doi.org/10.17487/RFC7252, URL https://www.rfc-editor.org/info/rfc7252.
[9] A. Minaburo, L. Toutain, C. Gomez, D. Barthel, J.-C. Zúñiga, SCHC: Generic Framework for Static Context Header Compression and Fragmentation, RFC 8724, RFC Editor, 2020, http://dx.doi.org/10.17487/RFC8724, URL https://www.rfc-editor.org/info/rfc8724.
[10] S. Aguilar, A. Platis, R. Vidal, C. Gomez, Energy consumption model of SCHC packet fragmentation over sigfox LPWAN, Sensors 22 (6) (2022) http://dx.doi.org/10.3390/s22062120, URL https://www.mdpi.com/1424-8220/22/6/2120.
[11] R. Muñoz, J. Saez Hidalgo, F. Canales, D. Dujovne, S. Céspedes, SCHC over LoRaWAN efficiency: Evaluation and experimental performance of packet fragmentation, Sensors 22 (4) (2022) http://dx.doi.org/10.3390/s22041531, URL https://www.mdpi.com/1424-8220/22/4/1531.
[12] J. Sanchez-Gomez, J. Gallego-Madrid, R. Sanchez-Iborra, J. Santa, A.F. Skarmeta, Impact of SCHC compression and fragmentation in LPWAN: A case study with lorawan, Sensors 20 (1) (2020) http://dx.doi.org/10.3390/s20010280, URL https://www.mdpi.com/1424-8220/20/1/280.
[13] B. Moons, A. Karaagac, J. Haxhibeqiri, E. De Poorter, J. Hoebeke, Using SCHC for an optimized protocol stack in multimodal LPWAN solutions, in: 2019 IEEE 5th World Forum on Internet of Things (WF-IoT), IEEE, 2019, pp. 430–435.
[14] D. Wistuba, S. Céspedes, J.C. Zúñiga, R. Muñoz Lara, S. Aguilar Romero, C. Gómez Montenegro, R. Vidal Ferré, An implementation of IoT LPWAN SCHC message fragmentation and reassembly, in: SSN 2020: V School on Systems and Networks: Proceedings of the V School on Systems and Networks, SSN 2020: VitÓRia, Brazil, December 14-15, 2020, CEUR-WS. org, 2020, pp. 1–4.
[15] W. Ayoub, F. Nouvel, S. Hmede, A.E. Samhat, M. Mroue, J.-C. Prévotet, Implementation of SCHC in NS-3 simulator and comparison with 6LoWPAN, in: 26th International Conference on Telecommunications, ICT, 2019.
[16] K. Mekki, E. Bajic, F. Chaxel, F. Meyer, A comparative study of LPWAN technologies for large-scale IoT deployment, ICT Express 5 (1) (2019) 1–7, http://dx.doi.org/10.1016/j.icte.2017.12.005, Publisher: Elsevier B.V..
[17] R. Sanchez-Iborra, J. Sanchez-Gomez, J. Ballesta-Viñas, M.-D. Cano, A. Skarmeta, Performance evaluation of LoRa considering scenario conditions, Sensors 18 (3) (2018) 772, http://dx.doi.org/10.3390/s18030772, URL http://www.mdpi.com/1424-8220/18/3/772.
[18] R. Sanchez-Iborra, M.-D. Cano, State of the art in LP-WAN solutions for industrial IoT services, Sensors 16 (5) (2016) 708, http://dx.doi.org/10.3390/s16050708, URL http://www.mdpi.com/1424-8220/16/5/708.
[19] S. Farrell, Low-Power Wide Area Network (LPWAN) Overview, RFC 8376, RFC Editor Medium, 2018, http://dx.doi.org/10.17487/RFC8376, URL https://rfc-editor.org/rfc/rfc8376.txt.
[20] LoRa Alliance™, What is it LoRaWAN™ - A Technical Overview of LoRa® and LoRaWAN™, Tech. Rep., 2015, URL https://lora-alliance.org/resource-hub/what-lorawantm, Issue: November.
[21] L. Alliance, N. Sornin, M. Luis, T. Eirich, T. Kramp, O. Hersent, LoRaWAN ™ specification v1.0.2, LoRa™ Alliance (2016).
[22] L.A.T. Committee, LoRaWAN™ 1.0.2 Regional Parameters, Tech. Rep., 2017, pp. 1–55.
[23] C. Gündogan, C. Amsüss, T.C. Schmidt, M. Wählisch, IoT content object security with OSCORE and NDN: A first experimental comparison, in: Proc. of 19th IFIP Networking Conference, IEEE Press, Piscataway, NJ, USA, 2020, pp. 19–27, URL https://ieeexplore.ieee.org/document/9142731.

[24] J. Schaad, CBOR Object Signing and Encryption (COSE), RFC 8152, RFC Editor, 2017, http://dx.doi.org/10.17487/RFC8152, URL https://www.rfc-editor.org/info/rfc8152.
[25] K. Varda, Protocol Buffers: Google's Data Interchange Format, Tech. Rep., Google, 2008, URL http://google-opensource.blogspot.com/2008/07/protocol-buffers-googles-data.html.
[26] J. Sanchez-Gomez, D.G. Carrillo, R. Sanchez-Iborra, J.L. Hernandez-Ramos, J. Granjal, R. Marin-Perez, M.A. Zamora-Izquierdo, Integrating LPWAN technologies in the 5G ecosystem: A survey on security challenges and solutions, IEEE Access 8 (2020) 216437–216460, http://dx.doi.org/10.1109/ACCESS.2020.3041057, URL https://ieeexplore.ieee.org/document/9272765/.
[27] J. Sanchez-Gomez, J. Gallego-Madrid, R. Sanchez-Iborra, J. Santa, A. Skarmeta, Impact of SCHC compression and fragmentation in LPWAN: A case study with LoRaWAN, Sensors 20 (1) (2020) 280, http://dx.doi.org/10.3390/s20010280, URL https://www.mdpi.com/1424-8220/20/1/280.
[28] C. Gomez, A. Minaburo, L. Toutain, D. Barthel, J.C. Zuniga, IPv6 over LPWANs: Connecting low power wide area networks to the internet (of things), IEEE Wirel. Commun. 27 (1) (2020) 206–213, http://dx.doi.org/10.1109/MWC.001.1900215, Publisher: IEEE.
[29] A. Minaburo, L. Toutain, R. Andreasen, Static Context Header Compression (SCHC) for the Constrained Application Protocol (CoAP), RFC 8824, RFC Editor, 2021, http://dx.doi.org/10.17487/RFC8824, URL https://www.rfc-editor.org/info/rfc8824.
[30] C. Bormann, M. Ersue, A. Keränen, Terminology for Constrained-Node Networks, RFC 7228, RFC Editor, 2014, http://dx.doi.org/10.17487/RFC7228, URL https://www.rfc-editor.org/info/rfc7228.
[31] S. Hristozov, M. Huber, L. Xu, J. Fietz, M. Liess, G. Sigl, The cost of OSCORE and EDHOC for constrained devices, in: Proceedings of the Eleventh ACM Conference on Data and Application Security and Privacy, ACM, New York, NY, USA, 2021, pp. 245–250, http://dx.doi.org/10.1145/3422337.3447834, URL https://dl.acm.org/doi/10.1145/3422337.3447834.
[32] M.C. Bor, U. Roedig, T. Voigt, J.M. Alonso, Do LoRa low-power wide-area networks scale? in: Proceedings of the 19th ACM International Conference on Modeling, Analysis and Simulation of Wireless and Mobile Systems, ACM, New York, NY, USA, 2016, pp. 59–67, http://dx.doi.org/10.1145/2988287.2989163, URL https://dl.acm.org/doi/10.1145/2988287.2989163.
[33] M. Bembe, A. Abu-Mahfouz, M. Masonta, T. Ngqondi, A survey on low-power wide area networks for IoT applications, Telecommunication Systems 71 (2) (2019) 249–274, http://dx.doi.org/10.1007/s11235-019-00557-9, Publisher: Springer New York LLC.