

Performance Evaluation of Publish-Subscribe Systems in IoT Using Energy-Efficient and Context-Aware Secure Messages

Norisvaldo Ferraz Junior (✉ norisjunior@usp.br)

Polytechnic School of Universidade de São Paulo (USP) <https://orcid.org/0000-0003-4239-512X>

Anderson AA Silva

Polytechnic School of Universidade de São Paulo (USP)

Adilson E Guelfi

Universidade do Oeste Paulista

Sergio T Kofuji

Polytechnic School of Universidade de São Paulo (USP)

Research Article

Keywords: IoT, WSN, publish-subscribe, context-aware messages, security

Posted Date: June 1st, 2021

DOI: <https://doi.org/10.21203/rs.3.rs-387836/v1>

License:   This work is licensed under a Creative Commons Attribution 4.0 International License.

[Read Full License](#)

RESEARCH

Performance evaluation of publish-subscribe systems in IoT using energy-efficient and context-aware secure messages

Norisvaldo Ferraz Junior^{1*}, Anderson AA Silva^{1,2,3,4}, Adilson E Guelfi⁵ and Sergio T Kofuji¹

Abstract

Background: The Internet of Things (IoT) enables the development of innovative applications in various domains such as healthcare, transportation, and Industry 4.0. Publish-subscribe systems enable IoT devices to communicate with the cloud platform. However, IoT applications need context-aware messages to translate the data into contextual information, allowing the applications to act cognitively. Besides, end-to-end security of publish-subscribe messages on both ends (devices and cloud) is essential. However, achieving security on constrained IoT devices with memory, payload, and energy restrictions is a challenge.

Contribution: Messages in IoT need to achieve both energy efficiency and secure delivery. Thus, the main contribution of this paper refers to a performance evaluation of a message structure that standardizes the publish-subscribe topic and payload used by the cloud platform and the IoT devices. We also propose a standardization for the topic and payload for publish-subscribe systems.

Conclusion: The messages promote energy efficiency, enabling ultra-low-power and high-capacity devices and reducing the bytes transmitted in the IoT domain. The performance evaluation demonstrates that publish-subscribe systems (namely, AMQP, DDS, and MQTT) can use our proposed energy-efficient message structure on IoT. Additionally, the message system provides end-to-end confidentiality, integrity, and authenticity between IoT devices and the cloud platform.

Keywords: IoT; WSN; publish-subscribe; context-aware messages; security

Introduction

The number of Internet of Things (IoT) devices connected to the Internet continually increases as well as the data produced by these devices [1, 2, 3]. The IoT devices provide the services of sensing, monitoring, and automation of activities [4].

Continuous sensory data gathering is essential to support the IoT applications, where devices periodically sense their environment and send the data to the cloud [5]. However, the energy consumption must be in focus [6, 7], either concerning devices' battery life duration or the energy efficiency of the cloud platform.

Thus, standardized end-to-end messages between the cloud platform and the IoT devices are essential, whether for the cloud to send commands or for the devices to send their measurements [8, 9].

In this scenario, to transmit the IoT application data, various Machine-to-Machine (M2M) protocols can be used according to [10, 11, 12, 13, 14, 15]: Constrained Application Protocol (CoAP), Message Queue Telemetry Transport (MQTT), eXtensible Messaging and Presence Protocol (XMPP), Advanced Message Queuing Protocol (AMQP), and Data Distribution Service (DDS). CoAP is a request/response RESTful protocol derived from the Hypertext Transfer Protocol (HTTP) and is designed for constrained devices [10, 15]. CoAP-enabled devices require a request to send data to the cloud, for example. However, when cloud-connected, an important feature of devices is to transmit data without waiting for a request. Therefore, the publish-subscribe systems (XMPP, MQTT, AMQP, and DDS) provide the ability to periodically send data to the cloud [12, 13, 14] without the need for receiving a request. Publish-subscribe systems connect endpoints or clients (IoT devices and the cloud platform, for instance) using a Broker.

Problem fundamentals and contribution

Publish-subscribe systems ensure data delivery to the cloud using TCP as transport and provide two re-

* Correspondence: norisjunior@usp.br

¹Laboratório de Sistemas Integráveis, Polytechnic School of Universidade de São Paulo (USP), São Paulo/SP, BR

Full list of author information is available at the end of the article

sources to the messages: the topic (a hierarchical list indicating which context the data sent refers to) and the payload.

IoT applications' data must contain some meaningful context to facilitate interoperability and to help in reducing the processing effort of cloud applications [16, 17, 18]. The context must be present to provide metadata (identification of devices and their sensors, deployment location) to the IoT applications' data.

The context-aware requirement attests to issues related to publish-subscribe systems: the absent standardization of the topic and payload. Although the unstructured nature provides flexibility, it also has drawbacks: topic and payload vary from tens to hundreds of bytes, as observed in the works of [12, 19, 20, 21].

Two categories of devices can use publish-subscribe systems: constrained (ultra-low-power) and unconstrained (high capacity). Both enable remote sensing capabilities delivering IoT data to the cloud platform. The ultra-low-power IoT devices are resource-constrained in processing (16 MHz MCU, for example, once are microcontroller-based), memory (\sim 128KB ROM/Flash, \sim 20KB RAM), battery (3V), and payload (\sim 127 to 256 bytes depending on the protocol) [10]. For this reason, these devices require specific link-layer protocols, such as LoRa and SigFox - for Low-Power Wide Area Network (LPWAN) or Wireless Sensor Networks (WSN) - the latter mainly composed by Low-Power Wireless Personal Area Network (LoWPAN). Conversely, the unconstrained devices commonly utilize (Wireless) Local Area Network (W)LAN, and they are robust enough to use any protocol, whether for messages or security. However, if battery-powered, the energy only lasts a few hours, as observed by [19, 19, 10].

Regarding security between the publish-subscribe clients and the Broker, the Transport Layer Security (TLS) provides robust security, and it is the primary option. However, considering the publish-subscribe systems' architecture, even the Transport Layer Security (TLS) is not suitable to provide end-to-end security once the IoT data is secure only between the client and the Broker and not between the clients. We observe in the literary works the use of more than one security control [22].

Therefore, the publish-subscribe systems need a message structure to provide end-to-end security because, if the payload does not have security, the Broker can read the messages. Additionally, only the unconstrained devices can use TLS, as observed in [23]. The use of TLS is not an option for ultra-low-power IoT devices given its constrained nature [24, 25]. Nonetheless, even without TLS, the ultra-low-power devices must transmit end-to-end

secure publish-subscribe messages, using a security mechanism on the message. In this regard, LPWAN devices do not provide end-to-end security once they need a gateway to connect the devices to the Internet (and consequently to the cloud). LPWAN devices also only send raw data to the gateway and do not enable the devices to use any publish-subscribe protocol. LoWPAN devices, in turn, are natively connected to the Internet (using 6LoWPAN) and can use the publish-subscribe systems to transmit data to the cloud.

Thus, the main contributions of this paper can be summarized as:

- An energy-efficient, context-aware, and end-to-end secure publish-subscribe messages (LWPubSub) for cloud-connected devices;
- An extensive evaluation of the message system using both ultra-low-power devices and high-capacity devices;
- A resource comparison (focused on energy consumption) among devices that uses publish-subscribe systems;
- A comparison of message size among publish-subscribe systems for high capacity and ultra-low-power devices;
- A comparison between the main publish-subscribe systems (AMQP, DDS, and MQTT) when communicating between heterogeneous devices (high-capacity and ultra-low-power) and the cloud platform.

Figure 1 presents the overview of the use of IoT devices and the interconnection to the cloud using the LWPubSub message structure on publish-subscribe systems.

Figure 1: LWPubSub context-aware and secure message using publish-subscribe systems

Article organization

The remainder of this paper is organized according to the following structure. “*Related Work*” section introduces publish-subscribe systems for IoT, focusing on the necessity of standardization of the messages, reviewing and discussing relevant techniques. “*System model*” section presents the proposed structure of context-aware and secure messages for publish-subscribe systems. “*System evaluation and experiments*” section presents the experimental scenario and the parameters for evaluating our proposed message system. “*Results and discussion*” section presents the results for both ultra-low-power and high capacity devices. Finally, “*Conclusion*” section provides the conclusion and future work suggestions.

Table 1: Overview of the main requirements of platforms of IoT devices

Constrained IoT devices					
Platform	Battery lifetime [26, 10]	Data rate [27, 26]	Range [27, 26]	Payload [27, 26]	End-to-end to the cloud
LoRa	Years	50 kbps	20 km	243 bytes	No
SigFox	Years	100 bps	20 km	12 bytes	No
6LoWPAN	Months	250 kbps	100 m	127 bytes	Yes
Unconstrained IoT devices					
Platform	Battery lifetime [19, 20]	Data rate	Range	Payload	End-to-end to the cloud
SBC	Hours	>54 Mbps (WLAN)	100 m	2312 bytes	Yes

Related Work

This section reviews the related work regarding publish-subscribe systems for IoT applications to clarify our contributions.

IoT applications' devices

Remote sensing provides sensing and actuation capabilities to IoT devices. These tasks involve, for instance, measuring environmental conditions (inside/outside temperature and humidity, smoke presence, gases, among others) and health information (for wearables) [28, 3].

The devices send these data to the cloud platform. The LPWAN and LoWPAN devices provide lower energy consumption when compared to unconstrained devices such as Single Board Computers (SBC) [19, 20, 21]. In Table 1, we compare the functions and capabilities for remote sensing regarding constrained and unconstrained devices. For LoWPAN, 6LoWPAN is the main protocol for routing packets to the Internet.

In Fig. 2, we present a graphic representation comparison of the IoT devices presented in Table 1. While being the most energy-efficient, we observe that LoRa and SigFox devices do not provide end-to-end connection to the cloud. In turn, SBC and 6LoWPAN devices provide end-to-end data transmission to the cloud. 6LoWPAN devices have an advantage over battery life, which lasts for months compared to hours of SBC devices.

Figure 2: IoT device's platform comparison regarding battery life, end-to-end data transmission to the cloud, payload length, QoS, range, and scalability

The 6LoWPAN devices are embedded systems restricted in processing, memory, payload, and energy resources [29]. The connection to the Internet (as stated in RFC 6282) requires the use of the 6LoWPAN Border Router (6LBR) [30]. The 6LBR performs IPv6 packet routing - using the IPv6 Routing Protocol for Low-Power and Lossy Networks (RPL - RFC 6550) - with compressed headers on IEEE 802.15.4 networks

(MTU 127 bytes) - the 6LBR is more robust equipment when compared to other nodes in the network.

SBC is microprocessor-based, and this type of device can run general-purpose Operating Systems (OS), like Linux. Thus, SBC is the leading platform used to evaluate IoT frameworks and message systems, as observed in [19, 20, 21, 31].

Message standardization

In publish-subscribe systems, the topic refers to the context of the payload, and both need standardization to enable interoperability and the correct processing of data. The works of [32, 33] proposes standardization for the topic, using different approaches, but none of the presented works are concerned with interoperability; in contrast, the goal of their work is only to present a specific format.

Tantitharanukul et al. [33] presents the difficulty of a subscriber understanding non-standardized topics for open smart cities. Therefore, the topic structure proposed by [33] requires, at least, the fields "*Objective/Location/Owner*" and other sub-fields.

Vrettos et al. [32], in turn, proposes topics concerning "*networkName/ nodeID/ country/ districtState/ city-Town/ areaDescription/ area/ building/ room/ control*" - a too large topic for remote sensing by ultra-low-power devices. Both works expect clients with direct access to the Broker.

Publish-subscribe systems

The XMPP, DDS, MQTT, and AMQP are the main protocols used by publish-subscribe systems [10, 11, 14, 13, 34]. These systems are not suitable to discover services to configure the IoT devices automatically [21], and the cloud platform plays an important role at the orchestration and provisioning of the devices which use these message system.

Publish-subscribe systems enable the interaction of multiple endpoints (Broker, publisher, and subscriber) [19, 21]. Interactions between clients are moderated by the Broker, so clients do not need to know each other to exchange messages [21]. A client is a peer, an application, or a device that exchanges application messages about a given topic with another

client. The payload is topic-dependent and does not follow any standard. As presented by [35, 36] it is common to use JSON at the payload-agnostic publish-subscribe systems.

XMPP is a communication protocol based on XML, which supports multiple patterns, including asynchronous messaging and publish-subscribe. Therefore, according to [25], data communication requires optimized Internet application protocols. XMPP has its primary use as Instant Messaging (IM), though it also can be used to deliver data between devices and the cloud. However, as presented by [37], XMPP is not originally designed for IoT applications, and it cannot be deployed in its current form on constrained devices. Besides, XMPP does not support acknowledged communication. The main obstacle regarding XMPP is the use of XML language, which causes longer messages, therefore, consuming a larger bandwidth [14].

MQTT is a publish-subscribe messaging protocol and is suitable for devices with resource constraints and networks with low bandwidth and high latency [15]. Its simplicity and small header size compared to other protocols (XMPP, AMQP, and DDS) make it one of the most important options for IoT applications [14]. MQTT defines three Quality-of-Service (QoS) levels for the messages: 0 (at-most-once), 1 (at-least-once), and 2 (exactly-once). This flexible approach assigns complexity to the Broker, resulting in a lightweight header and a small code footprint [38]. MQTT was also designed for asynchronous communications, where subscriptions or publishing from different entities take place in a parallel order [39]. Similar to XMPP, the MQTT protocol uses TCP as a transport. Client communication occurs by the Broker's request to subscribe or publish messages, specifying the topic and payload fields of the message. Thus, clients, who want to receive a message, subscribe to a specific topic with the message delivered on the payload [3, 21, 14]. Topics are hierarchically organized in a tree structure using the "/" separator.

AMQP (like MQTT) is another ISO/IEC messaging protocol, developed initially for banking services [38, 34]. Its main purpose is to manage the queues receiving a large number of transactions and to deliver them reliably later. Message queues are queues where messages are routed to and keep them there until the corresponding subscriber reads them. The current version of AMQP enables peer-to-peer communication, not only client-broker communication like MQTT. However, AMQP requires the higher header amongst the other publish-subscribe systems, and its use for IoT applications is mainly restricted to servers or high capacity equipment [14].

DDS enables high-performance M2M communication using the publish-subscribe paradigm [15]. DDS is a standard presented by the Object Management Group (OMG) and is brokerless [40]. Therefore, DDS is decentralized, and publishers and subscribers exchange data directly between them. Thus, a publisher client publishes data even if there is no subscriber since publishers do not know who uses their data [14]. The DDS defines four main entities: the "Domains" (a virtual entity that allows communication of devices with the same interests), the "Publisher" with its "Data Writer" (used by the publisher to send the data), the "Subscriber" and its "Data Reader" (controlled by the subscriber to read the data), and the "Topic". In contrast to the previous protocols, DDS supports UDP and TCP, and one of its benefits is the wide range (23) of available QoS [14]. Moreover, the DDS header requires at least 56 bytes, and even the DDS for eXtremely Resource Constrained Environments (DDS-XRCE) [41], require at least 12 Bytes - 8 bytes at the header plus 4 bytes for each *submessageHeader*. For that reason, the works of [40, 10] conclude that DDS consumes at least two times more bandwidth than a protocol like MQTT.

Table 2 summarizes the key features of the publish-subscribe message systems under consideration based on the corresponding literature.

We conclude, regarding the considered publish-subscribe systems, that the following protocols produce larger packets (from the largest to the smallest): XMPP, DDS, AMQP, and MQTT.

Regarding security, high-capacity IoT devices, such as SBC, can use standard protocol (TLS, for example). However, ultra-low-power devices are not capable of using TLS. Consequently, a security mechanism must be in place to protect the data regardless of the used publish-subscribe system because the Broker is an intermediary in this communication. Thus, if the payload does not have security, the Broker can read the messages. Even using TLS, the payload needs security not to reveal data to entities other than the clients.

Thus, we propose a structure for the topic and payload for publish-subscribe systems that can benefit either constrained or high capacity IoT devices (for those capable of sending end-to-end messages to the Internet without proxies). The innovation in our message structure is that the proposed structure for topic and payload is energy-efficient and secure. The energy efficiency is twofold: for devices, the message improves battery lifetime, and for cloud platforms, it reduces network traffic and power consumption. Besides, the messages are end-to-end secure between devices and the cloud platform once only TLS cannot deal with end-to-end security between devices and

Table 2: Key features of the publish-subscribe message systems under consideration

	XMPP	MQTT	AMQP	DDS
Transport protocol	TCP	TCP	TCP	TCP/UDP
QoS support	No	3 levels	3 levels	23 levels
Header size (at least)	Variable (XML tags)	2	8	12*
Security	TLS	TLS	TLS	TLS/DTLS
Encoding format	XML	Binary	Binary	Binary
Low-Power and Lossy networks	Fair	Good	Good	Poor
Standard	IETF	OASIS	OASIS	OMG

*DDS-XRCE

cloud, considering the Broker’s presence that forwards the publish-subscribe messages.

System model

This section presents our LWPubSub message structure, providing secure and context-aware messages for high capacity (SBC) and constrained (ultra-low-power) devices transmission to the cloud platform using publish-subscribe systems. We also point out the energy-efficiency importance for both devices and the cloud platform.

Context-aware publish-subscribe messages

Publish-subscribe systems are payload-agnostic, and developers design topics and payload to meet the application needs. Nevertheless, IoT devices are diverse, ranging from high capacity to ultra-low-power. Hence, this diverse characteristic of devices poses challenges to designing a structure of topic and payload.

Considering the myriad of sensors and the absence of a standard, topic and payload range from tens to hundreds of bytes as observed in [21, 19].

Accordingly, an important discussion arises regarding standardization. First, the topic must identify the data source to the cloud platform process and store it. Cloud platforms, such as Azure [1] and AWS [2] present guidelines regarding the topic construction. However, these guidelines do not focus on the energy efficiency of the topic nor provide unique identification of devices. Using Azure as an example, we have a topic with 60 bytes. AWS, in turn, requires approximately 30 Bytes for telemetry data (topic can be greater according to the deviceID or other functions at the platform). Both platforms do not provide a standard for the unique identification of devices.

Besides, JSON is one of the main structures used when there is no standard to follow [35, 36]. Although, the JSON structure alone requires at least 7 bytes at the payload: {“”:“”} - without considering the sensor

type used. About IoT sensors, the various types of sensors require efforts to identify them. For instance, the temperature sensor data, in a JSON payload can be: {“temperature”:“15.01”}. Also, a developer can use shorter metadata, like: {“t”:“15.01”}, but it might be a pitfall because for an IoT architecture with various domains and sensors, the character “t” can represent other sensors, not only temperature.

Thus, emerged the necessity to standardize the topic and the payload. As stated by [42], standardization decreases the gaps and reduces system complexity. Our LWPubSub message structure goes in this direction once we provide a unique topic for each device and standardized metadata on the payload.

Cloud-connected devices

Considering the diverse characteristic of IoT devices presented in the sections “Introduction” and “Related Work” and the necessity of them being able to connect to the Internet (and to the cloud), the devices that meet these requirements are: SBC and WSN (6LoWPAN) devices.

LPWAN devices (such as LoRa and SigFox) require a proxy that receives the message from the devices and forwards them to the cloud, breaking end-to-end communication (and security).

Energy-efficiency

Energy efficiency is a crucial characteristic for the devices and the cloud platform. As presented by [43], the cloud computing metrics use the amount of data traffic to calculate the Key Performance Indicators (KPIs) related to energy efficiency and Greenhouse Gases (GHG) emissions.

Regarding corporate-level metrics, we observe some organizations using the metric *Carbon Intensity*, represented by the division of Scope 1 and 2 emissions by the total amount of data transported over its network ($CO_2/Terabyte$) [43], as presented in Eq. 1.

$$Carbon\ Intensity = \frac{GHG\ emissions\ Scope\ 1\ and\ 2}{Terabytes\ of\ data\ traffic} \quad (1)$$

[1]<https://docs.microsoft.com/en-us/azure/iot-hub/iot-hub-mqtt-support>

[2]<https://docs.aws.amazon.com/whitepapers/latest/designing-mqtt-topics-aws-iot-core/designing-mqtt-topics-aws-iot-core.html>

The Power Usage Effectiveness (PUE) is a metric that shows progress in data center energy efficiency [44], according to Eq. 2, and it is widely used to observe data center efficiency, recommended by the GHG Protocol [44].

$$PUE = \frac{\text{Total Facility Power}}{\text{IT Equipment Power}} \quad (2)$$

Moreover, concerning the equipment-level metrics at the cloud platform, the Energy Consumption Rating Weighted (ECRW) represents the energy necessary to move n Gbps of data in Watt/Gbps [43], according to the Eq. 3.

$$ECRW = \frac{((\alpha \times E_f) + (\beta \times E_h) + (\gamma \times E_i))}{T_f} \quad (3)$$

where T_f is the maximum throughput (in Gbps); E_f is the energy consumption (in Watts) at maximum capacity; E_h is the energy consumption at half capacity; E_i is the energy consumption in idle; $\alpha = 0.35$, $\beta = 0.4$, and $\gamma = 0.25$ are the coefficients to represent the mixed mode of operation.

We observe in Eq. 1, 2, and 3 that the number of bytes of the messages directly affect the energy consumption of the cloud platform. Thus, when constructing the topic and the payload, it is necessary to optimize the required bytes of a message.

Therefore, we conclude that reducing the number of bytes transmitted at each publish-subscribe message decreases the energy consumption, PUE, and Carbon Intensity of a cloud platform.

About devices, attention to the number of bytes remains equally important.

The energy consumption of IoT devices depends on the platform. High-capacity devices (with the 802.11 interface on) have the same energy consumption when performing sensing tasks (except when using CPU at 100% constantly - which is not the case of sensing data) [45]. So, each characteristic of the devices is considered to consume an amount of energy, for example, video output and Bluetooth, to calculate high capacity device's energy consumption. Moreover, high-capacity devices used for remote sensing only require processing and transmission capability. Thus, devices must operate heedlessly.

On the other hand, ultra-low-power devices have more granularity to measure energy consumption. As presented by [46, 47] TX and RX are the resources that require more energy on these devices. Also, it is important to observe the CPU and Low Power Mode (LPM) energy consumption on them.

Moreover, for ultra-low-power devices, the application also needs to consider the 127-bytes MTU of 6LoWPAN devices.

Message system

Therefore, we present in Fig. 3 the proposed structure of topic and payload for publish-subscribe systems. The domains are the vertical markets [3] that enable the wide range of IoT architectures. A given IoT architecture, can contain the following domains: school, healthcare, transportation, among others (this work does not restrict the design of the domain, one can use a numeric sequence). Each domain has its own devices, uniquely identified by the *deviceID* metadata. Our proposal regarding *deviceID* is that the unique identification of the device must come from the MAC address. For that reason, the tuple */domain/deviceID* uniquely identify a device at an IoT Architecture.

Regarding the sensors and actuators of a device, we propose the use of the IPSO registry [48], using the tuple *objectID/instanceID* to uniquely identify the devices' sensors, where *objectID* is the type of the sensor and *instanceID* is the sensor itself.

The use of IPSO objects requires at most 5 bytes to identify a sensor, plus an additional byte to identify the instance of the sensor (in cases where a device has two or more sensors of the same type). For instance, assuming a device with two LEDs, IPSO defines the *objectID* 3303, followed by the *instanceID* 0 for the first and 1 for the other. Thus, compared to the JSON payload, the proposed use of IPSO objects meets the context-aware requirements while being energy-efficient.

With the union of */domain/deviceID* (at the topic) and *objectIDinstanceID* (at the payload), we achieve lower byte usage for publish-subscribe systems.

When using MQTT, AMQP, or DDS publish-subscribe messages, the LWPubSub message structure sends the topic in plain text, which guarantees the end-to-end transmission of messages between clients without the need for an intermediate proxy.

Besides, the payload needs encryption to guarantee end-to-end security. Even using the standard best practices (TLS or DTLS), if the payload remains unencrypted, the Broker (MQTT/AMQP) or a subscriber (DDS) can read the message - resulting in data leakage. For this reason, LWPubSub applies confidentiality, authenticity, and integrity (CIA) to the payload.

Security

Without security on the payload, an intermediary between the cloud and the devices (like the Broker) can read the messages, even when using TLS. Thus, the payload requires security. Applying confidentiality

Figure 3: Structure of the LWPubSub message structure for publish-subscribe messages

only, although possible, does not offer non-repudiation of the data transmitted. For this reason, it is crucial to use the CIA on the messages, providing Authenticated Encryption (AE). The LWPubSub message structure provides CIA to the publish-subscribe message payload using the Advanced Encryption Standard (AES) Counter Mode with Cipher Block Chaining Message Authentication Code (CCM) [25]. Besides, authenticated encryption provides security against chosen-ciphertext attacks.

The AES-CCM authenticates but does not encrypt the topic (the associated data), generating the authentication tag. The LWPubSub encrypts the payload, ensuring CIA. Further, AES-CCM requires a unique Nonce, and it provides a reduced message size and absence of padding.

The LWPubSub message structure encrypts and generates the payload according to these steps:

- 1 The first 13 bytes refer to the Nonce;
- 2 The following 8 bytes refer to the Message Integrity Code (MIC) - or authentication tag;
- 3 The subsequent bytes refer to the encrypted message.

About the IoT Architecture, we consider secure from end to end the messages exchanged between the devices and the IoT Agent at the cloud platform.

System evaluation and experiments

In this section, we introduce simulation scenarios, including parameter settings. We conducted several experiments to evaluate the proposed message system.

We observe and evaluate, with the execution of the experiments, the following metrics regarding the use of LWPubSub on MQTT, AMQP, and DDS:

- Header overhead;
- Energy consumption (observation of the behavior of devices in different operating conditions, obtaining measurements in poll frequency variations using a wearable as an environmental sensor);
- Number of bytes required for topic and payload.

For the LWPubSub message structure validation and to collect the results, we use the experimental scenario presented in Fig 4.

In Table 3, we summarize the characteristics of the equipment, devices, sensors, and parameters used in our proposed message structure.

To validate the IoT architecture and considering as a premise the use of open standards, the FIWARE platform [49, 13, 50] provides the needed infrastructure

Table 3: Experiment parameters

Equipment and Devices		
Cloud platform	Microservices/docker containers	
MQTT Broker	Mosquitto on the Huawei cloud	
AMQP Broker	RabbitMQ on the Huawei cloud	
DDS	OpenDDS on the Huawei cloud	
6LBR	Launchpad CC2650	
IoT Gateway	Raspberry pi model 3B	
Ultra-low-power devices	Sensortag CC2650 Remote CC2538	
High capacity devices	Raspberry Pi model 3B Raspberry Pi Zero W	
Cloud platform parameters		
Platform	FIWARE-based	
Context-Broker	Orion 2.3.0	
Database	Mongo DB 3.6	
ultra-low-power devices features		
Resource	Sensortag	Remote
TX current	7.9 mA	24 mA
RX current	6 mA	20 mA
LPM current	0.55 mA	0.60 mA
CPU current	3.48 mA	20 mA
Microcontroller	CC2650 (48 MHz ARM Cortex M3)	CC2538 (32 MHz ARM Cortex M3)
ROM	128 KB	512 KB
RAM	20 KB	32 KB
OS	Contiki-NG	
TSCH schedule	Minimal	
High capacity devices features		
Resource	Rasp. Pi model 3B	Rasp. Pi Zero W
CPU	1.2 GHz 64-bit quad core ARM Cortex- A53	1 GHz ARM11 32- bit
Flash memory	32 GB	32 GB
RAM	1 GB	512 MB
802.11 active (headless)*	225.17 mA	95.15 mA
OS	Raspbian release 10	
<i>*measured with USB Voltmeter Tester UM24C</i>		
LWPubSub message structure parameters		
Domain	99	
Sensortag deviceId	00124b05257a	
Remote deviceId	00124b4a527d	
Rasp. Pi 3B deviceId	0012eb00f6d0	
Rasp. Pi Zero W deviceId	0012ebc894cb	
objectID	3303, 3311, and 3338	
ultra-low-power devices security	Payload with AES-CCM-8	
High capacity devices security	TLS + payload with AES-CCM-8	

for an opensource cloud. The FIWARE platform contains Generic Enablers (GE), including services called IoT Agents that receive CWIoT measurements. The Orion Context-Broker (the main GE) is the context-aware service provided by FIWARE. Contextualized data enable the correct processing of information by the cloud platform [49].

Figure 4: Experimental scenario

In the cloud platform, the domains and devices are orchestrated and provisioned, according to Table 3. Devices' sensors are (following the LWPubSub structure *objectIDinstanceID*): 33030 (HDC temperature sensor), 33110 (red LED), 33111 (green LED), and 33380 (alarm).

The LWPubSub assembles the message (topic and payload) as shown in Fig. 3. An example of topic used in our experiments is (using Sensortag *deviceID* in this example) /99/00124b05257a, followed by the encrypted payload which contains a context-aware temperature data 33030|15.01.

The publish-subscribe systems, at the cloud platform, run at the IoT Agent on the Huawei cloud. The communication of the devices with the cloud, the IoT gateway, runs on a Raspberry Pi model 3B, with an IEEE 802.11 interface connected to the Internet and an IEEE 802.15.4 interface (via Launchpad - the 6LBR) to communicate with the 6LoWPAN network.

The ultra-low-power devices of the validation scenario are the Texas Instruments Sensortag and the Zolertia Remote rev. b. To comply with the energy-efficient goal of the LWPubSub, we use the Time Slotted Channel Hopping (TSCH) as the link-layer protocol since it presents the lowest energy consumption relating to this layer on 6LoWPAN networks [51] (we use TSCH minimal schedule). We use Contiki-NG, an open-source Operating System (OS) for constrained devices. As presented in section "Related Work", ultra-low-power devices can run only MQTT at the application layer.

In contrast, high-capacity devices run MQTT, AMQP, and DDS in the experiments.

Regardless of devices' capacity, we implement AES-CCM-8 at the payload, ensuring end-to-end security between the devices and the cloud platform.

Moreover, IoT applications require different time intervals when collecting sensor measurements, ranging from wearable (which requires many measurements per minute) to environmental conditions measures (which require only a few measurements per day). Therefore, to observe the behavior of CWIoT in different operating conditions, the experiments conducted include the following poll frequencies to obtain and send measurements:

- Very high: one measurement every 5 seconds (17280 per day);
- High: one measurement every quarter of a minute (5760 per day);

- Medium: one measurement every quarter of an hour (96 per day);
- Low: one measurement every quarter of a day (4 per day);
- Very low: one measurement per day.

Results and discussion

In this section, we analyze the impact of the publish-subscribe message systems considering several significant parameters and discuss the performance of the proposed message system through experiments.

Publish-subscribe message size

About the topic, the LWPubSub message structure requires 16 bytes for sending measurements and 20 bytes to receive commands from the cloud platform. Concerning the payload presented in the section "System model", to send telemetry data like temperature or humidity, the secure payload is 32 bytes long. Thus, the entire LWPubSub message (topic+payload with end-to-end security) requires 52 bytes to send one-sensor data (like temperature).

There are differences between the header overhead of the AMQP, DDS, and MQTT protocols when using LWPubSub. We present the overhead based on the required bytes in the header and the total message size. Thereupon, we need to observe the header overhead percentage, which we calculate using the Eq. 4, considering the header size presented in the section "Related works".

$$Header\ overhead(\%) = \frac{Header}{Header + Payload} \times 100 \quad (4)$$

In Fig. 5 we observe the header overhead required by AMQP, DDS, and MQTT when using LWPubSub topic and payload. The dashed line in Fig. 5 represents the header overhead with the use of LWPubSub to send temperature data. MQTT presents the lower header overhead (3.70%), followed by AMQP (13.33%) and DDS (18.75%). It is important to highlight the relevance of the protocol overhead, given the constrained payload of ultra-low-power devices (127 bytes), where only 104 bytes are available. Thus, above 104 bytes, the message requires fragmentation, increasing the number of messages to send the data and energy consumption.

Regarding SBC, which does not have payload-size restrictions, the header overhead is not significant for larger messages. For example, the header overhead of a

Figure 5: Header overhead using LWPubSub messages

payload of 1185 bytes is below 1% for MQTT, AMQP, and DDS.

The structure design of our message proposes a short and lightweight (and yet, complete) topic. We assume the cloud platform as the main entity that publicly provides the IoT data to the users. The publish-subscribe system (MQTT, AMQP, DDS) is only accessible between the devices and the cloud platform and not by other users.

In contrast, [33] proposes a topic naming criteria for smart cities requiring users' access to the Broker. The topic structure proposed by [33] requires, at least, the fields "Objective/Location/Owner" and its sub-fields. For instance, in the example provided by the authors, a temperature measurement topic requires 61 bytes. The topic proposed by [32] requires 72 bytes - we consider "country" and "state" fields with 2 bytes to reduce the number of bytes on the topic. On the other hand, our proposed topic for sending temperature requires 16 bytes. Fig. 6 presents the topic comparison with the main related works.

Figure 6: Topic results comparison with the main related works

In the works of [33, 32], the client needs to know the Broker's address and to subscribe to the desired topics. Our work, in turn, proposes cloud computing as the primary entity which provides the IoT data to the users.

Regarding the ultra-low-power devices (which use only MQTT), to send one-sensor telemetry data (like temperature), the LWPubSub message (with CIA) generates a 6LoWPAN packet of 113 bytes in the experiments. Thus, there is no fragmentation of messages sent by the ultra-low-power devices (which have a messaging limit of 127 bytes). For high-capacity devices, the payload is 106-bytes long.

We present in Fig. 7 the comparison of the message size among the main related works. We consider our context-aware and secure payload (32 bytes) for all works compared; however, each work has its respective topic. The LWPubSub message presents a smaller message size, thus, requiring the transmission of fewer bytes and saving energy.

Figure 7: Message size comparison with the main related works

Energy consumption

On the subject of energy consumption, we apply the Eq. 5 to observe the energy consumption considering idle and active periods of the IoT devices; thus, the parameter n varies depending on the type of IoT device used.

$$Energy_{IoTdevice} = \sum_{i=1}^n n \times V \times I \quad (5)$$

In Eq. 5, n is the time spent for device resource, V is the voltage, and I is the current. Regarding high capacity devices, we deploy a headless Raspberry Pi model 3B and Raspberry Pi Zero W. We measure the energy consumption with the devices in idle and active (transmission and reception using the 802.11 radio). According to Eq. 5, the energy consumption presents the time spent considering $V = 5$, the current (I) as presented in Table 3, and $n = (Idle, TX/RX)$.

About the energy consumption calculation of 6LoWPAN devices (ultra-low-power), we proceed according to Eq. 5, using the Contiki-NG Energest [52] module. We calculate the energy (in millijoules) from the time spent (in seconds) of each device resource (CPU, LPM, TX, RX). The voltage is fixed at 3 V (when supplied by two AA batteries), and the current (I) for each resource is shown in Table 3, and $n = (CPU, LPM, TX, RX)$.

The SBC devices evaluated, as presented in Fig. 4, are capable of running MQTT, AMQP, and DDS. However, the ultra-low-power devices are only capable of running MQTT, given their constrained nature.

We present in Fig. 8 the energy consumption for an IoT device to send and receive one-sensor LWPubSub messages. It is important to note the y-axis limits because, for ultra-low-power devices, the energy required is under 10mJ; nevertheless, SBC devices require much more energy, reaching almost 200 mJ. Moreover, in Fig. 8, we consider the actions "send a measurement" and "receive a command", not considering the idle time.

(a) ultra-low-power (b) SBC

Figure 8: Energy consumption to send measurements and receive commands

Additionally, Fig. 9 represents the comparison of the required daily energy of ultra-low-power and SBC devices.

Besides, Fig. 9a shows the ultra-low-power devices consuming less than 400 J (Sensortag is more constrained and consumes less than 300 J). We observe then that sleep time (LPM) and listening (RX) drive ultra-low-power devices' energy consumption. The CPU and TX energy consumption barely impacts and is much the same for medium, low, and very low poll frequencies.

Regarding the energy consumption of SBC presented in Fig. 9b, it reaches approximately 8,000 J for Raspberry Pi Zero W and almost 20,000 J for the Raspberry Pi 3B, regardless of the publish-subscribe message system (MQTT, AMQP, or DDS). SBC spends most of its time in "idle", even for the poll frequency "very high".

We observe the previous comparison of energy consumption when looking at the battery life of the devices at each poll frequency. Let us consider the energy provided by two AA batteries: $30780\text{ J } (2.85\text{Ah} * 1.5\text{V} * 3600\text{secs.} * 2)$.

With this energy, the Raspberry Pi 3B and Raspberry Pi Zero W run for 1.5 and 3.7 days, respectively, regardless of the poll frequency. In contrast, ultra-low-power devices present a different behavior. Figure 10 illustrates the battery lifetime of ultra-low-power devices at the evaluated poll frequencies. The "very high" and "high" poll frequencies presented in Fig. 10 (which demand more energy) present lower battery life than the other poll frequencies. However, sending 96, 4, or 1 message per day (medium, low, and very low poll frequencies) consumes nearly the same amount of energy - even for those that wait longer between measurements such as "low" and "very low".

Thus, high-capacity devices must have an uninterrupted power supply; in contrast, ultra-low-power devices can run on battery for months, making mobility feasible.

Regarding security, for an IoT application that requires the best security practices, SBC is the device to use. On the other hand, TLS is not an option for ultra-low-power devices once they cannot run this heavy-weight security protocol. However, as presented in the section "System model", the LWPubSub message structure applies confidentiality, integrity, and authenticity to the payload.

IoT Applications

A smart healthcare IoT architecture can be benefited from the use of ultra-low-power devices using LWPubSub. A tracking system for vaccine temperature control is an example of an application. It is possible

to measure the temperature of a vaccine box from the production to the destination once the LWPubSub application periodically publishes these data. Besides, these devices can join 6LoWPAN networks when in their range; in contrast, an SBC device (which uses the 802.11 radio) requires a previous wireless network configuration to transmit data. If there is a 6LoWPAN network in the industry, at the transportation, and the destination, it is possible to track the vaccine box temperature end-to-end because the device joins the respective network and keeps sending the data. In addition, making the measured temperature publicly available to the citizens and the scientific community increases the transparency of the information, especially regarding the Covid-19 pandemic.

Healthcare for the elderly is another domain of smart healthcare that can be benefited from the ultra-low-power devices using LWPubSub. Even collecting data at a higher poll frequency, ultra-low-power devices run months without requiring battery replacement.

High-capacity devices are helpful to send a large amount of data, for instance, a security camera that detects an anomaly and sends the images to the cloud so that authorities can analyze them.

These applications, using LWPubSub, transmit context-aware and end-to-end secure messages between the IoT devices and the cloud platform, and the Broker (or any intermediary subscriber on DDS networks) is unable to read the message.

Conclusion

This paper proposes and evaluates an energy-efficient, context-aware, and end-to-end secure message structure for publish-subscribe systems. Both ultra-low-power and high-capacity devices benefit from the optimized topic and payload proposed.

The context-aware topic guarantees the unique device identification on its domain. The context-aware payload ensures the standardization of each device's sensors, including end-to-end security between the devices and the cloud platform.

We highlight the importance of the reduced number of bytes at the topic and the payload of messages to reduce energy consumption because carbon footprint and power effectiveness metrics are based on the number of bytes transmitted.

The proposed message system does not require customizing the protocols (MQTT, AMQP, or DDS) or the Broker. Thus, it is possible to use any Broker available to forward the secure messages to the cloud platform.

Experimental evaluation shows that our message system can run for months on ultra-low-power devices. We also present IoT applications that can benefit

(a) ultra-low-power

(b) SBC

Figure 9: Daily energy consumption of IoT devices

Figure 10: Ultra-low-power devices battery lifetime on the evaluated poll frequencies

from the message system. To achieve this contribution, we execute several experiments applying different poll frequencies to evaluate the diverse characteristics of IoT applications.

Limitations and Future works Although the results obtained prove the low energy consumption and the message size without resorting to fragmentation for ultra-low-power devices, the evaluation comprises one measurement from one sensor at a time. An extension of this work could be evaluating the message size to verify if sending measurements from more than one sensor in the same message requires fragmentation, its impact on energy consumption, and other implications this may cause. Additionally, for ultra-low-power devices, explore other encryption models that can be adopted when considering security, using key derivation functions and features to deliver the static keys, requiring analysis of the message size and energy consumption. Regarding total energy consumption, future works may include evaluating another TSCH Schedule.

Competing interests

The authors declare that they have no competing interests.

Author's contributions

All authors took part in the discussion and the proposal of the work described in this paper. Norisvaldo Ferraz Junior conceived of the initial idea, wrote the system, and carried out the experiments as a Doctoral student under the supervision of Sergio T. Kofuji and Anderson A. A. Silva. Norisvaldo wrote the paper with the support of Anderson A. A. Silva and Adilson E. Guelfi. Sergio T. Kofuji supervision and review this paper. All authors read and approved the final manuscript.

Author's information

Norisvaldo Ferraz Junior received his B.S. degree in information systems in 2003 from USC his Master's degree in Computer Engineering from IPT in 2016. He is currently pursuing a Ph.D. degree at the Polytechnic School of Universidade de São Paulo (USP). His research interests focus on security in the Internet of Things, smart cities, wireless sensor networks, and machine learning.

Anderson A. A. Silva received his Ph.D. Degree in Computer Engineering from USP in 2016, his Master's degree in Computer Engineering from IPT in 2010, his MBA in Systems Analysis from FECAP in 1993, and his Data Processing degree from FIEO in 1991. Anderson has over 25 experience in the IT field. Currently, he is a professor in several undergraduate and graduate courses in Sao Paulo and develops his second postdoctoral project in machine learning security at USP.

Adilson E. Guelfi is a Doctor in Electric Engineering at the Electronic Engineering Department, EPUSP (Escola Politecnica, University of Sao Paulo), Sao Paulo. He has over 18 years of experience in training, education, research, project development, and innovation in the information security area. Currently, Adilson has the Dean of Research and Graduate Studies position at UNOESTE (University of Western Sao Paulo) and is also a regular professor of the Master's program in Computer Engineering at the FIPT (Institute for Technological Research Foundation – Sao Paulo). **Sergio T. Kofuji** holds a Bachelor's Degree in Physics from the University of Sao Paulo (1985), a Master's Degree in Electrical Engineering from the University of Sao Paulo (1988), and a Ph.D. in Electrical Engineering from the University of Sao Paulo (1995). He is currently Professor Dr. of the Polytechnic School of the University of Sao Paulo. Currently, he has been focusing on projects related to IoT, 5G communication, Smart Objects, Machine Learning, and Artificial Intelligence, applied to Smart Cities and Smart Farms.

Acknowledgements

This work is supported by the partnership between the Huawei Company and Universidade de São Paulo (USP).

Availability of data and materials

The test cases are generated by an algorithm, which will be made available under an appropriate open source license upon accepting the article.

Funding

No funds have been received from any agency for this research.

Author details

¹Laboratório de Sistemas Integráveis, Polytechnic School of Universidade de São Paulo (USP), São Paulo/SP, BR. ² Instituto de Pesquisas Tecnológicas do Estado de São Paulo, São Paulo/SP, BR. ³ Centro Universitário SENAC, São Paulo/SP, BR. ⁴ Universidade Paulista (UNIP), São Paulo/SP, BR. ⁵ Universidade do Oeste Paulista, São Paulo/SP, BR.

References

- Almutairi, J., Aldossary, M.: A novel approach for IoT tasks offloading in edge-cloud environments. *Journal of Cloud Computing* **10**(1), 28 (2021). doi:10.1186/s13677-021-00243-9
- Ni, J., Lin, X., Shen, X.S.: Toward Edge-Assisted Internet of Things: From Security and Efficiency Perspectives. *IEEE Network* **33**(2), 50–57 (2019). doi:10.1109/MNET.2019.1800229
- Al-Fuqaha, A., Guizani, M., Mohammadi, M., Aledhari, M., Ayyash, M.: Internet of Things: A Survey on Enabling Technologies, Protocols, and Applications. *IEEE Communications Surveys and Tutorials* **17**(4), 2347–2376 (2015). doi:10.1109/COMST.2015.2444095
- Asghari, P., Rahmani, A.M., Javadi, H.H.S.: Internet of Things applications: A systematic review. *Computer Networks* **148**, 241–261 (2019). doi:10.1016/j.comnet.2018.12.008
- Du, X., Zhou, Z., Zhang, Y., Rahman, T.: Energy-efficient sensory data gathering based on compressed sensing in IoT networks. *Journal of Cloud Computing* **9**(1) (2020). doi:10.1186/s13677-020-00166-x
- Junior, N.F., Silva, A., Guelfi, A., Kofuji, S.T.: Iot6sec: reliability model for internet of things security focused on anomalous measurements identification with energy analysis. *Wireless Networks* **25**(4), 1533–1556 (2019)
- Silva, A., Zhou, F., Pontes, E., Simplicio, M.A., Aguiar, R., Guelfi, A., Kofuji, S.: Energy-efficient node position identification through payoff matrix and variability analysis. *Telecommunication Systems* **65**(3), 459–477 (2017)
- Mohamed, N., Al-Jaroodi, J., Jawhar, I., Lazarova-Molnar, S., Mahmoud, S.: SmartCityWare: A service-oriented middleware for cloud and fog enabled smart city services. *IEEE Access* **5**(C), 17576–17588 (2017). doi:10.1109/ACCESS.2017.2731382
- Anagnostopoulos, T., Zaslavsky, A., Kolomvatsos, K., Medvedev, A., Amirian, P., Morley, J., Hadjieftymiades, S.: Challenges and

- Opportunities of Waste Management in IoT-Enabled Smart Cities: A Survey. *IEEE Transactions on Sustainable Computing* **2**(3), 275–289 (2017). doi:10.1109/tsusc.2017.2691049
10. Uslu, B.Ç., Okay, E., Dursun, E.: Analysis of factors affecting IoT-based smart hospital design. *Journal of Cloud Computing* **9**(1) (2020). doi:10.1186/s13677-020-00215-5
 11. Al-Masri, E., Kalyanam, K.R., Batts, J., Kim, J., Singh, S., Vo, T., Yan, C.: Investigating Messaging Protocols for the Internet of Things (IoT). *IEEE Access* **8**, 94880–94911 (2020). doi:10.1109/ACCESS.2020.2993363
 12. Bacco, M., Boero, L., Cassara, P., Colucci, M., Gotta, A., Marchese, M., Patrone, F.: IoT Applications and Services in Space Information Networks. *IEEE Wireless Communications* **26**(2), 31–37 (2019). doi:10.1109/MWC.2019.1800297
 13. Araujo, V., Mitra, K., Saguna, S., Åhlund, C.: Performance evaluation of FIWARE: A cloud-based IoT platform for smart cities. *Journal of Parallel and Distributed Computing* **132**, 250–261 (2019). doi:10.1016/j.jpdc.2018.12.010
 14. Glaroudis, D., Iossifides, A., Chatzimisios, P.: Survey, comparison and research challenges of IoT application protocols for smart farming. *Computer Networks* **168**, 107037 (2020). doi:10.1016/j.comnet.2019.107037
 15. Lin, J., Yu, W., Zhang, N., Yang, X., Zhang, H., Zhao, W.: A Survey on Internet of Things: Architecture, Enabling Technologies, Security and Privacy, and Applications. *IEEE Internet of Things Journal* **4**(5), 1125–1142 (2017). doi:10.1109/JIOT.2017.2683200
 16. de Matos, E., Tiburski, R.T., Moratelli, C.R., Johann Filho, S., Amaral, L.A., Ramachandran, G., Krishnamachari, B., Hessel, F.: Context information sharing for the Internet of Things: A survey. *Computer Networks* **166**, 106988 (2020). doi:10.1016/j.comnet.2019.106988
 17. Vilela, P.H., Rodrigues, J.J.P.C., Solic, P., Saleem, K., Furtado, V.: Performance evaluation of a Fog-assisted IoT solution for e-Health applications. *Future Generation Computer Systems* **97**, 379–386 (2019). doi:10.1016/j.future.2019.02.055
 18. Bittencourt, L., Immich, R., Sakellariou, R., Fonseca, N., Madeira, E., Curado, M., Villas, L., DaSilva, L., Lee, C., Rana, O.: The Internet of Things, Fog and Cloud continuum: Integration and challenges. *Internet of Things* **3-4**, 134–155 (2018). doi:10.1016/j.iot.2018.09.005. 1809.09972
 19. Khaled, A.E., Helal, A., Lindquist, W., Lee, C.: IoT-DDL-Device Description Language for the 'T' in IoT. *IEEE Access* **6**, 24048–24063 (2018). doi:10.1109/ACCESS.2018.2825295
 20. Khaled, A.E., Helal, S.: Interoperable communication framework for bridging RESTful and topic-based communication in IoT. *Future Generation Computer Systems* **92**, 628–643 (2019). doi:10.1016/j.future.2017.12.042
 21. Kim, G., Kang, S., Park, J., Chung, K.: An MQTT-Based Context-Aware Autonomous System in oneM2M Architecture. *IEEE Internet of Things Journal* **6**(5), 8519–8528 (2019). doi:10.1109/JIOT.2019.2919971
 22. Silva, A.A., Ferraz Jr, N., Guelfi, A.E., Barboza, S., Kofuji, S.T.: Grouping detection and forecasting security controls using unrestricted cooperative bargains. *Computer Communications* **146**, 155–173 (2019)
 23. Anthi, E., Ahmad, S., Rana, O., Theodorakopoulos, G., Burnap, P.: EclipseIoT: A secure and adaptive hub for the Internet of Things. *Computers and Security* **78**, 477–490 (2018). doi:10.1016/j.cose.2018.07.016
 24. Malina, L., Srivastava, G., Dzurenda, P., Hajny, J., Fujdiak, R.: A Secure Publish/Subscribe Protocol for Internet of Things. In: *Proceedings of the 14th International Conference on Availability, Reliability and Security - ARES '19*, pp. 1–10. ACM Press, New York, New York, USA (2019). doi:10.1145/3339252.3340503
 25. Diro, A., Reda, H., Chilamkurti, N., Mahmood, A., Zaman, N., Nam, Y.: Lightweight Authenticated-Encryption Scheme for Internet of Things Based on Publish-Subscribe Communication. *IEEE Access* **8**, 60539–60551 (2020). doi:10.1109/ACCESS.2020.2983117
 26. Morin, É., Maman, M., Guizzetti, R., Duda, A.: Comparison of the Device Lifetime in Wireless Networks for the Internet of Things. *IEEE Access* **5**, 7097–7117 (2017). doi:10.1109/ACCESS.2017.2688279
 27. Mekki, K., Bajic, E., Chaxel, F., Meyer, F.: A comparative study of LPWAN technologies for large-scale IoT deployment. *ICT Express* **5**(1), 1–7 (2019). doi:10.1016/j.ict.2017.12.005
 28. Miorandi, D., Sicari, S., De Pellegrini, F., Chlamtac, I.: Internet of things: Vision, applications and research challenges. *Ad Hoc Networks* **10**(7), 1497–1516 (2012). doi:10.1016/j.adhoc.2012.02.016. 1207.0203
 29. Čolaković, A., Hadžialić, M.: Internet of Things (IoT): A review of enabling technologies, challenges, and open research issues (2018). doi:10.1016/j.comnet.2018.07.017
 30. Khan, M.A., Salah, K.: IoT security: Review, blockchain solutions, and open challenges. *Future Generation Computer Systems* **82**, 395–411 (2018). doi:10.1016/j.future.2017.11.022
 31. Suresh, V.M., Sidhu, R., Karkare, P., Patil, A., Lei, Z., Basu, A.: Powering the IoT through embedded machine learning and LoRa. *IEEE World Forum on Internet of Things, WF-IoT 2018 - Proceedings 2018-Janua*, 349–354 (2018). doi:10.1109/WF-IoT.2018.8355177
 32. Vrettos, G., Logaras, E., Kalligeros, E.: Towards Standardization of MQTT-Alert-based Sensor Networks: Protocol Structures Formalization and Low-End Node Security. *2018 IEEE 13th International Symposium on Industrial Embedded Systems, SIES 2018 - Proceedings*, 1–4 (2018). doi:10.1109/SIES.2018.8442109
 33. Tantitharanukul, N., Osathanunkul, K., Hantrakul, K., Pramokchon, P., Khoenkaw, P.: MQTT-Topic Naming Criteria of Open Data for Smart Cities. In: *2016 International Computer Science and Engineering Conference (ICSEC)*, pp. 1–6. IEEE, ??? (2016). doi:10.1109/ICSEC.2016.7859892
 34. Chaudhary, A., Peddoju, S.K., Kadarla, K.: Study of Internet-of-Things Messaging Protocols Used for Exchanging Data with External Sources. *Proceedings - 14th IEEE International Conference on Mobile Ad Hoc and Sensor Systems, MASS 2017*, 666–671 (2017). doi:10.1109/MASS.2017.85
 35. Fischer, M., Kumper, D., Tonjes, R.: Towards improving the privacy in the MQTT protocol. *Global IoT Summit, GloTS 2019 - Proceedings (2019)*. doi:10.1109/GIOTS.2019.8766366
 36. Bellavista, P., Berrocal, J., Corradi, A., Das, S.K., Foschini, L., Zanni, A.: A survey on fog computing for the Internet of Things. *Pervasive and Mobile Computing* **52**, 71–99 (2019). doi:10.1016/j.pmcj.2018.12.007
 37. Habibzadeh, H., Soyata, T., Kantarci, B., Boukerche, A., Kaptan, C.: Sensing, communication and security planes: A new challenge for a smart city system design. *Elsevier B.V.* (2018). doi:10.1016/j.comnet.2018.08.001
 38. Gomez, C., Arcia-Moret, A., Crowcroft, J.: TCP in the Internet of Things: From Ostracism to Prominence. *IEEE Internet Computing* **22**(1), 29–41 (2018). doi:10.1109/MIC.2018.112102200
 39. Dinculeană, D., Cheng, X.: Vulnerabilities and Limitations of MQTT Protocol Used between IoT Devices. *Applied Sciences* **9**(5), 848 (2019). doi:10.3390/app9050848
 40. da Cruz, M.A.A., Rodrigues, J.J.P.C., Lorenz, P., Solic, P., Al-Muhtadi, J., Albuquerque, V.H.C.: A proposal for bridging application layer protocols to HTTP on IoT solutions. *Future Generation Computer Systems* **97**(2019), 145–152 (2019). doi:10.1016/j.future.2019.02.009
 41. Object Management Group: DDS-XRCE: DDS for eXtremely Resource Constrained Environments. *OMG standard November* (2019)
 42. Amiri-Zarandi, M., Dara, R.A., Fraser, E.: A survey of machine learning-based solutions to protect privacy in the Internet of Things. *Computers and Security* **96**, 101921 (2020). doi:10.1016/j.cose.2020.101921
 43. Riekstin, A.C., Rodrigues, B.B., Nguyen, K.K., de Brito Carvalho, T.C.M., Meirosu, C., Stiller, B., Cheriet, M.: A Survey on Metrics and Measurement Tools for Sustainable Distributed Cloud Networks. *IEEE Communications Surveys & Tutorials* **20**(2), 1244–1270 (2018). doi:10.1109/COMST.2017.2784803
 44. Mytton, D.: Assessing the suitability of the Greenhouse Gas Protocol for calculation of emissions from public cloud computing workloads. *Journal of Cloud Computing* **9**(1), 45 (2020). doi:10.1186/s13677-020-00185-8
 45. Bekaroo, G., Santokhee, A.: Power consumption of the Raspberry Pi: A comparative analysis. In: *2016 IEEE International Conference on Emerging Technologies and Innovative Business Practices for the Transformation of Societies, EmergiTech 2016*, pp. 361–366. IEEE, ???

- (2016). doi:10.1109/EmergiTech.2016.7737367
46. Elsts, A., Fafoutis, X., Oikonomou, G., Piechocki, R., Craddock, I.: TSCH Networks for Health IoT. *ACM Transactions on Internet of Things* **1**(2), 1–27 (2020). doi:10.1145/3366617
 47. Martinez, B., Montón, M., Vilajosana, I., Prades, J.D.: The Power of Models: Modeling Power Consumption for IoT Devices. *IEEE Sensors Journal* **15**(10), 5777–5789 (2015). doi:10.1109/JSEN.2015.2445094
 48. OMA LightweightM2M (LWM2M): OMA (Open Mobile Alliance) Specification (2017)
 49. Kovacs, E., Bauer, M., Kim, J., Yun, J., Le Gall, F., Zhao, M.: Standards-Based Worldwide Semantic Interoperability for IoT. *IEEE Communications Magazine* **54**(11), 40–46 (2016). doi:10.1109/MCOM.2016.1600460CM
 50. Cabrini, F.H., de Barros Castro Filho, A., Filippo Filho, V., Kofuji, S.T., Moura, A.R.L.P.: Helix sandbox: An open platform to fast prototype smart environments applications. In: 2019 IEEE 1st Sustainable Cities Latin America Conference (SCLA), pp. 1–6 (2019). IEEE
 51. Duquennoy, S., Elsts, A., Nahas, B.A., Oikonomo, G.: TSCH and 6TiSCH for Contiki: Challenges, Design and Evaluation. In: 2017 13th International Conference on Distributed Computing in Sensor Systems (DCOSS), pp. 11–18. IEEE, Ottawa, Canada (2017). doi:10.1109/DCOSS.2017.29
 52. Dunkels, A., Osterlind, F., Tsiftes, N., He, Z.: Software-based on-line energy estimation for sensor nodes. *Proceedings of the 4th Workshop on Embedded Networked Sensors, EmNets 2007*, 28–32 (2007). doi:10.1145/1278972.1278979

Figures

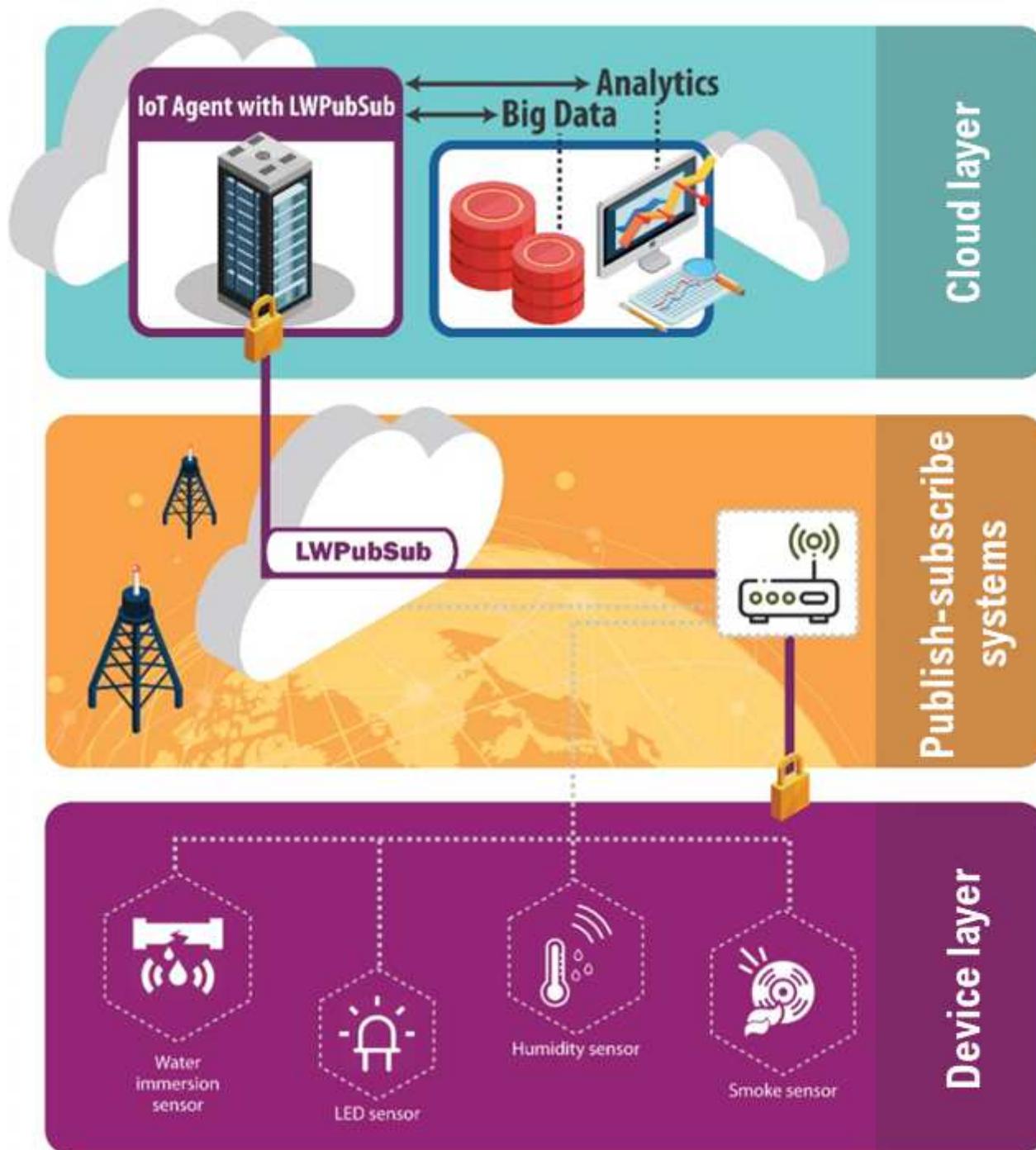


Figure 1

LWPubSub context-aware and secure message using publish-subscribe systems

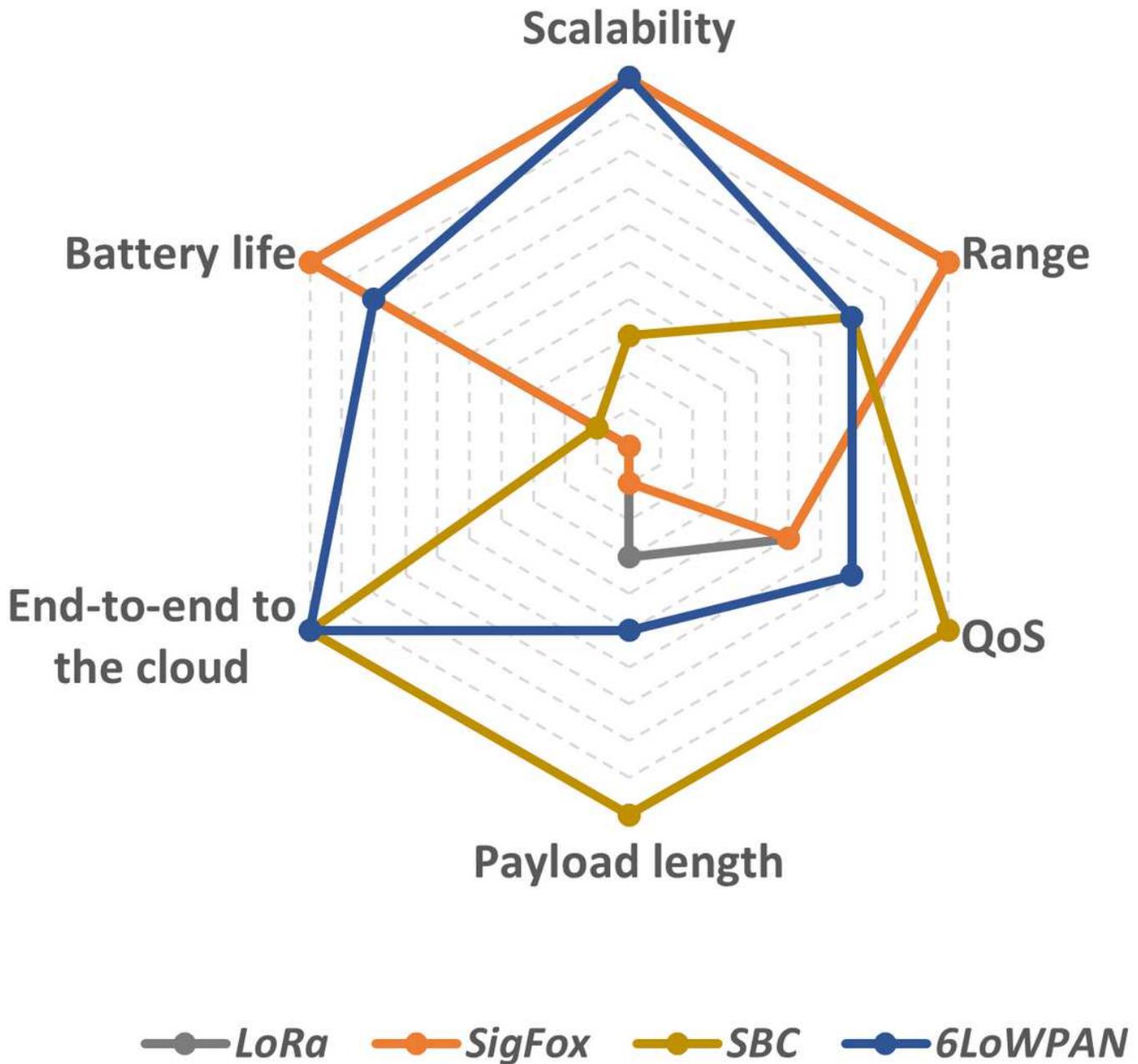


Figure 2

IoT device's platform comparison regarding battery life, end-to-end data transmission to the cloud, payload length, QoS, range, and scalability

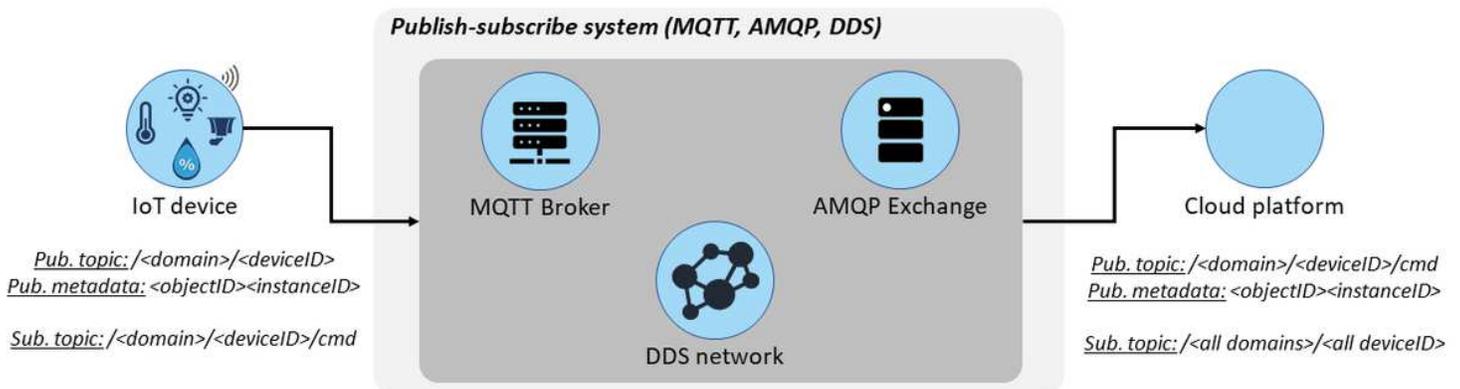


Figure 3

Structure of the LWPubSub message structure for publish-subscribe messages

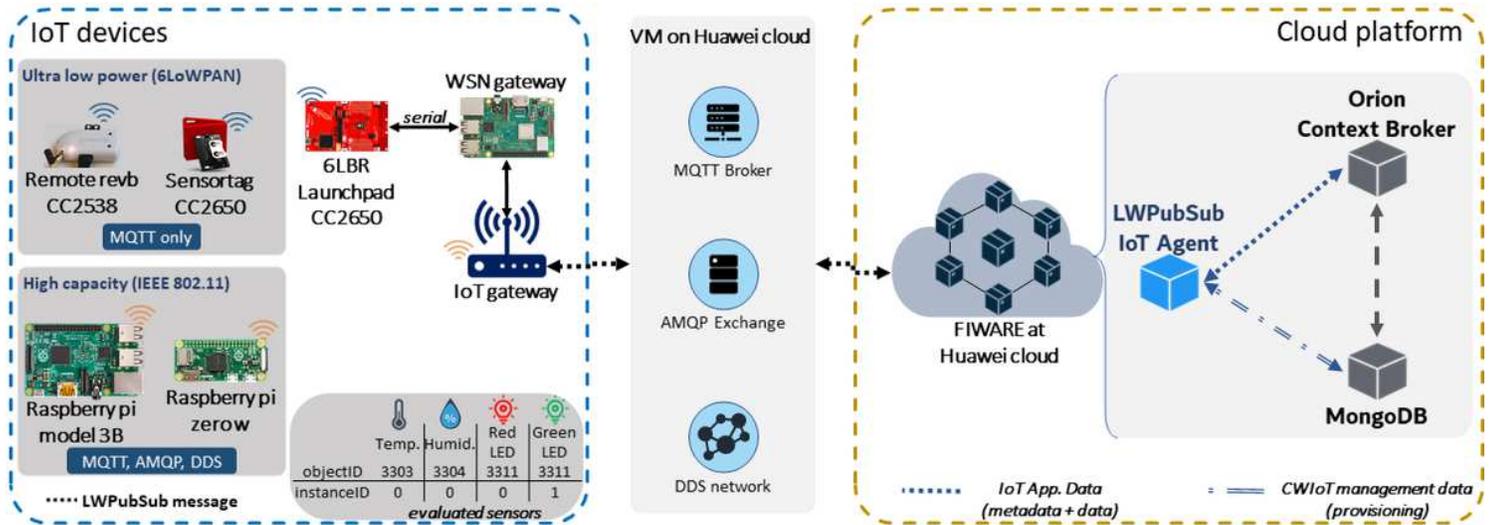


Figure 4

Experimental scenario

Publish-Suscribe systems overhead

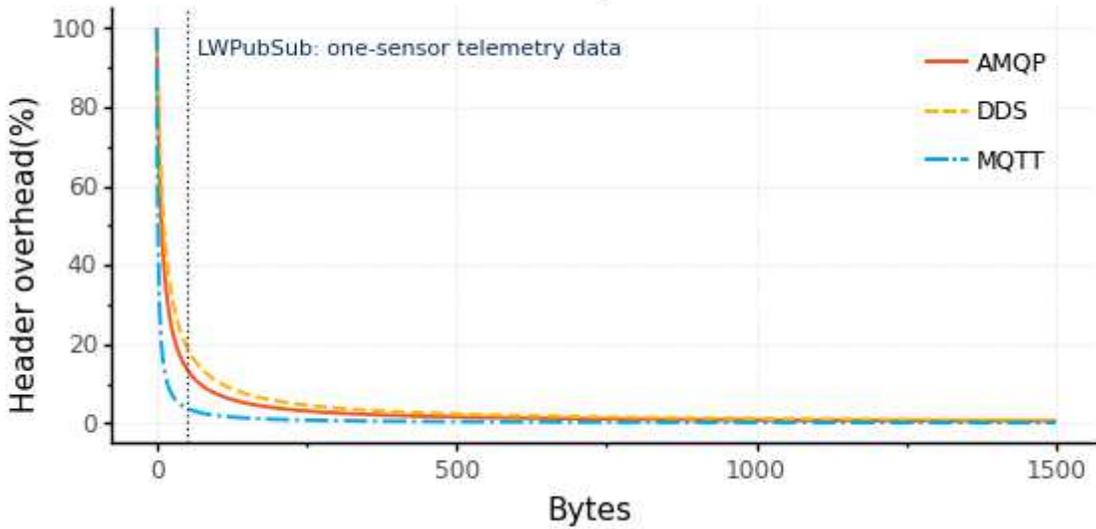


Figure 5

Header overhead using LWPubSub messages

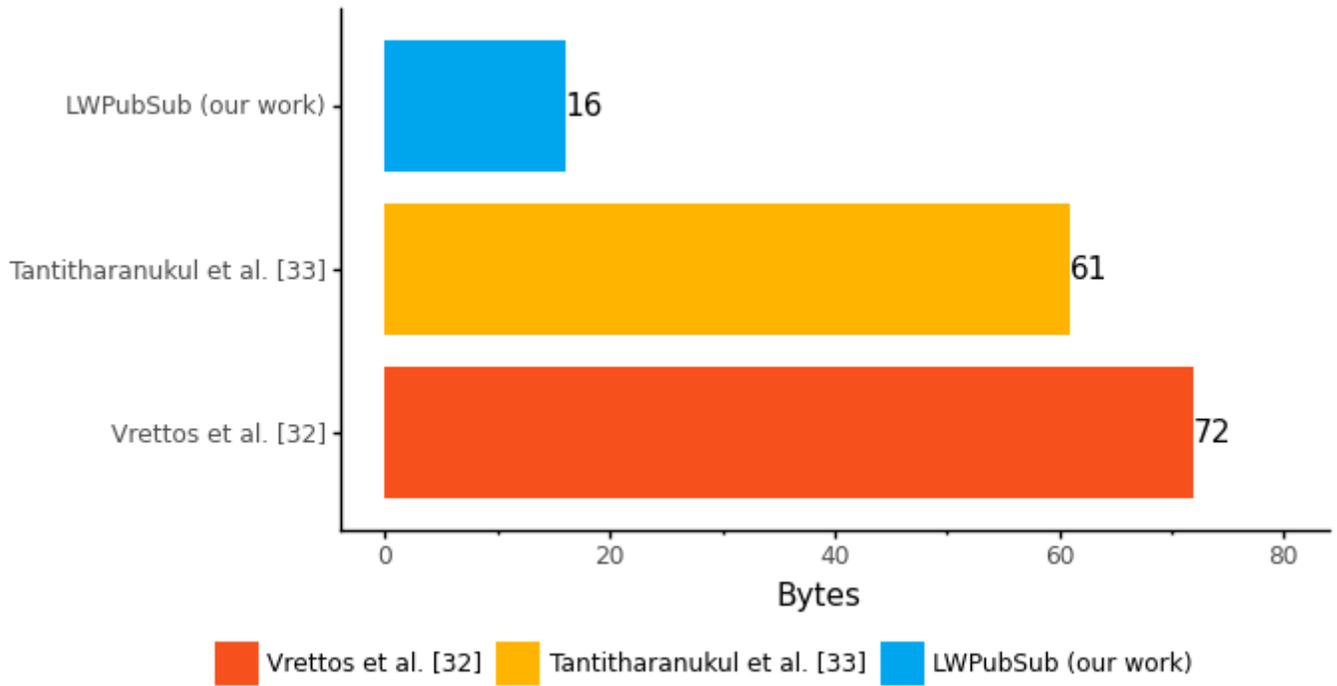


Figure 6

Topic results comparison with the main related works

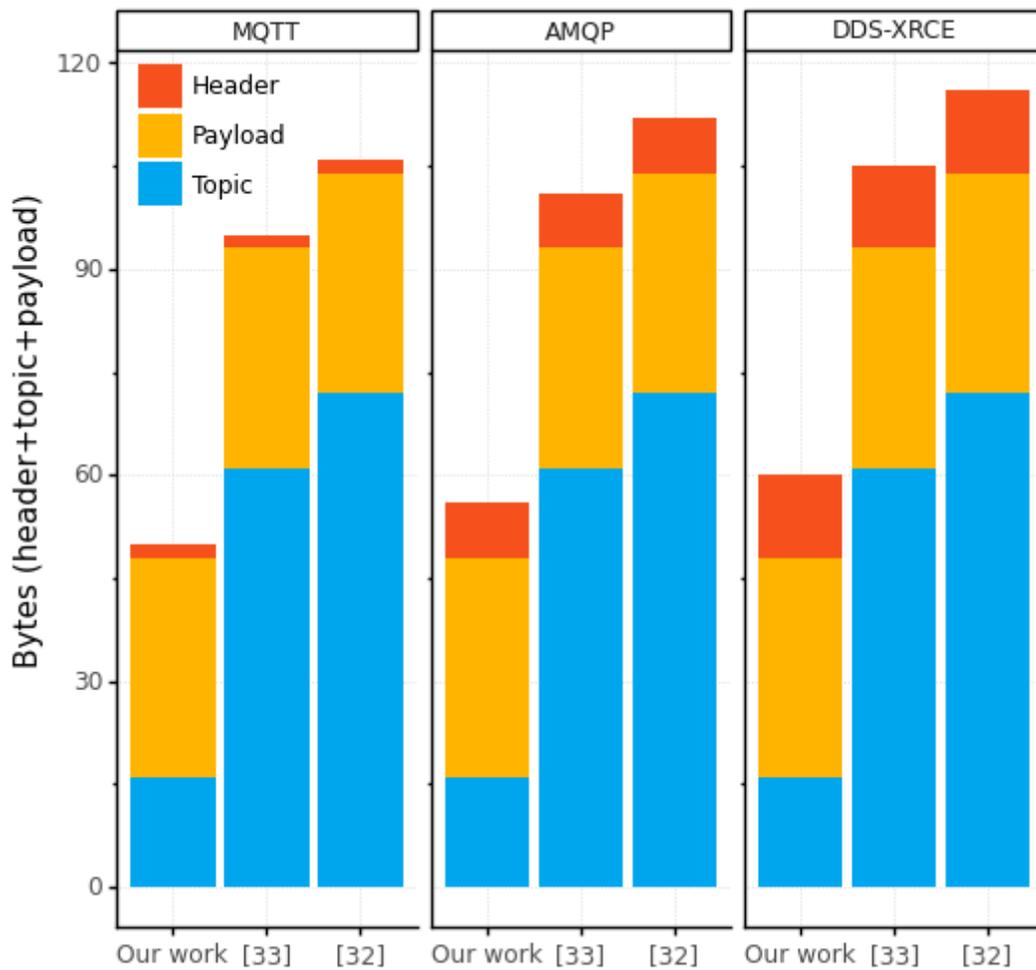


Figure 7

Message size comparison with the main related works

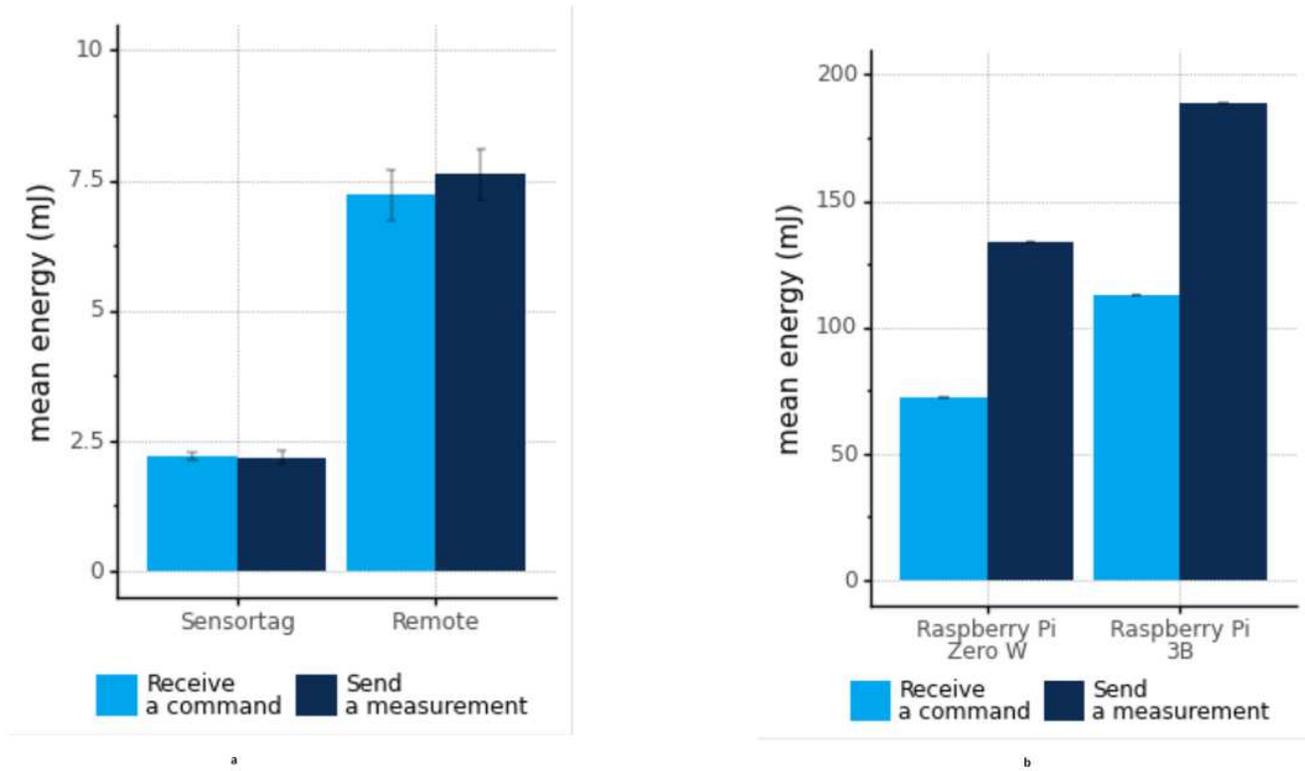
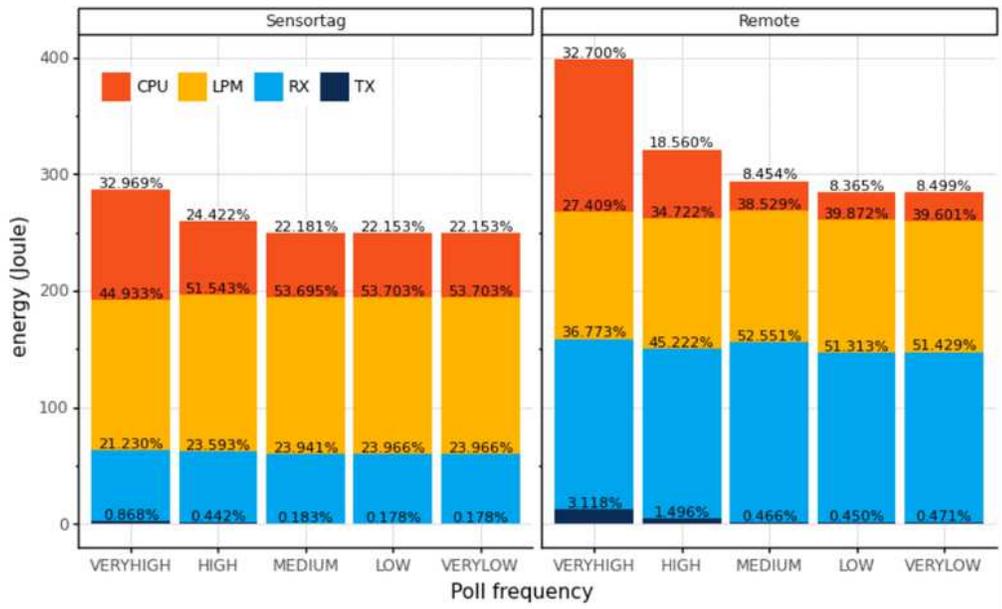
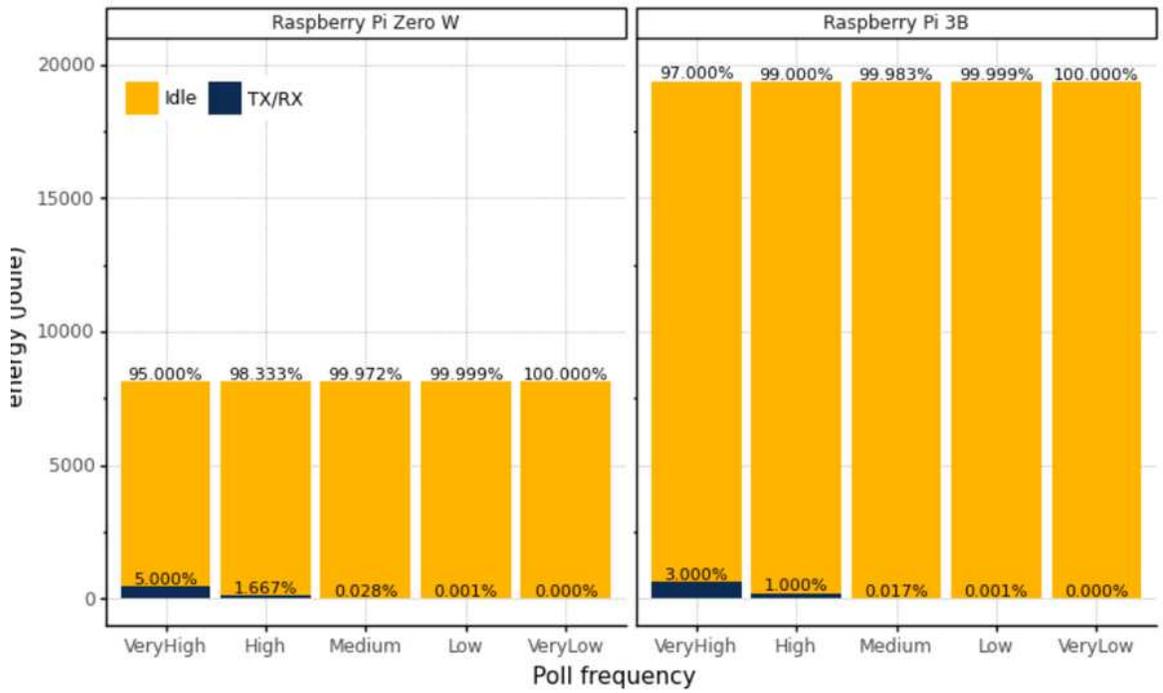


Figure 8

Energy consumption to send measurements and receive commands



a



b

Figure 9

Daily energy consumption of IoT devices

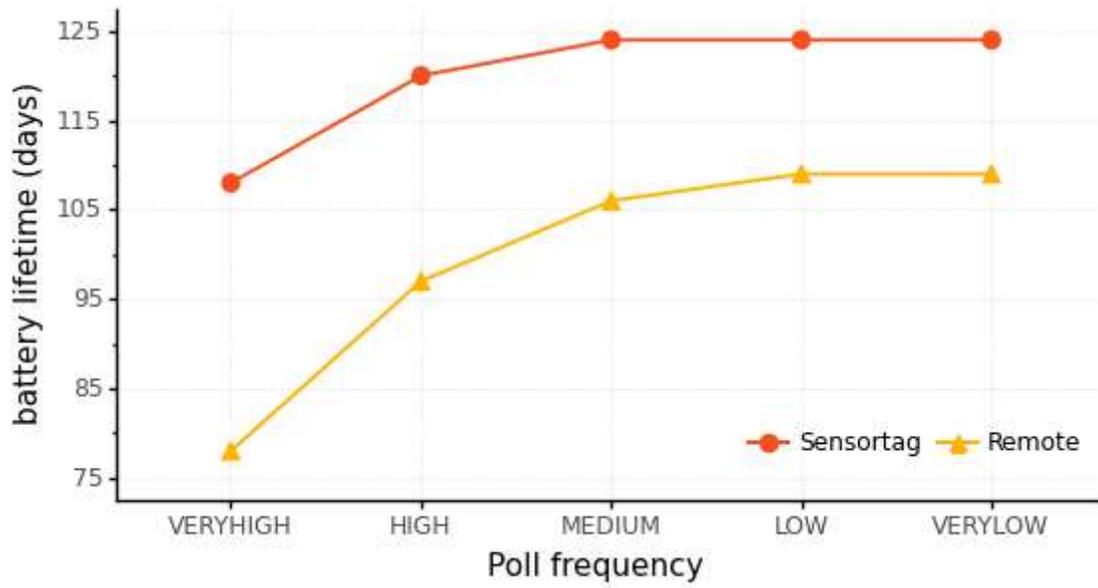


Figure 10

Ultra-low-power devices battery lifetime on the evaluated poll frequencies