Lightweight and Secure Publish-Subscribe System for Cloud-Connected Ultra Low Power IoT Devices

Norisvaldo Ferraz Junior, Anderson A. A. Silva, Adilson E. Guelfi, Marcelo Teixeira de Azevedo and Sergio Takeo Kofuji

***Abstract*—The Internet of Things (IoT) enables the development of innovative applications in various domains such as healthcare, transportation, and Industry 4.0. The integration of the cloud platform's large processing and storage capacity with the ubiquitous sensing and actuation provided by the devices creates an IoT architecture that provides vast raw data. The IoT devices send the data to the cloud platform with the developed IoT applications, which can use publish-subscribe systems. However, messages with sensitive content require end-to-end security. Besides that, IoT devices may present processing, memory, payload, and energy restrictions. In this sense, messages in an IoT architecture need to achieve both energy-efficiency and secure message delivery. Thus, this article's main contribution refers to a system that standardizes the publish-subscribe topic and payload used by the cloud platform and the IoT devices. Our system also provides end-to-end security while surpassing the energy-efficiency to send data than the main related works in the literature regarding the use of publish-subscribe systems in IoT.**

***Index Terms*—IoT, WSN, Cloud Computing, Energy Efficiency, Security, Publish-Subscribe.**

I. INTRODUCTION

The Internet of Things (IoT) enables the development of applications in various domains, such as healthcare, transportation, and Industry 4.0 [1–3], providing services of sensing, monitoring, and automation of activities [4]. For this, IoT applications require the union of the resources provided by devices (such as environmental sensing) and the processing power of cloud computing [5–7].

The IoT is ubiquitous because of the many devices connected via the Internet [1; 8], which contain sensors and actuators to interact with their environment [9; 10]. Thus, there are data transmission standards in several layers [11] aiming to connect IoT devices to the Internet [1; 8] such as 5G, Narrow Band IoT (NBIoT) or Bluetooth, and not all protocols are IP-based. In turn, IPv6 over Low energy Wireless Personal Area Network (6LoWPAN) is an IETF standard that provides native integration to the Internet, with end-to-end message forwarding [7] for 6LoWPAN devices on Wireless Sensor Networks (WSN). 6LoWPAN devices, from now on, referred to as Constrained Wireless IoT devices (CWIoT), present restrictions in processing, memory, payload, and energy [12], a fact that poses challenges about the communication with the cloud.

With the integration of the constrained devices with the cloud computing the CWIoT respond to requests coming only from the cloud [5], as well as enabling aggregation and use of sensing data by cloud applications.

For this integration to be successful, CWIoT and the cloud platform need to exchange standardized messages. The cloud platform is IP-based; therefore, the integration of CWIoT to the cloud platform is only possible if the upper layers of the IP layer use the same protocols, which poses a challenge considering the restricted nature of CWIoT.

In this scenario, for the exchange of messages between CWIoT and the cloud platform, the Message Queuing Telemetry Transport (MQTT) and the Constrained Application Protocol (CoAP) are the *de facto* standards [13–16]. The MQTT uses messages in the publish/subscribe format, which contains two parts: the MQTT topic (a hierarchical list indicating which context the data sent refers to) and the MQTT payload. The CoAP messages use the request/response format using RESTful standard. Also, the LightWeight Machine-to-Machine (LWM2M) protocol [17] provides standardized and context-aware messages on top of CoAP.

A. Problem fundamentals and contribution

MQTT lacks standardization of its topic and payload. Because of that, we observe in the works of [14; 18–20], proposals for topics and payloads without concern for standardized messages - nor data (devices measurements), nor metadata (identification of devices and their sensors, deployment location). This lack of standardization results in MQTT topics and payloads, ranging from tens to hundreds of bytes. For instance, [19] defines an MQTT topic requiring 33 bytes and an MQTT payload with at least 64 bytes, while [20] defines a topic with at least 70 bytes.

For this reason, the works of [14; 18–20] are not directly concerned with the energy efficiency of the topic or the payload, mainly because they use devices of higher capacity than the CWIoT. The use of CWIoT poses challenges in communicating with the cloud via the Internet, given their restricted characteristics. However, even in the middle of these

Norisvaldo Ferraz Junior, Marcelo Teixeira de Azevedo and Sergio Takeo Kofuji are with Integrated Systems Laboratory at Polytechnic School - University of São Paulo (USP), Sao Paulo-SP, Brazil. E-mails: norisjunior@usp.br; marcelo.azevedo@pad.lsi.usp.br; kofuji@usp.br.

Anderson A. A. Silva is with Integrated Systems Laboratory at Polytechnic School of University of São Paulo (USP), Technological Research Institute of the State of São Paulo (IPT), University Centre Senac and University Paulista, São Paulo-SP, Brazil. E-mail: anderson.silva@pad.lsi.usp.br.

Adilson E. Guelfi is with University of Western Paulista (Unoeste), Presidente Prudente-SP, Brazil. E-mail: guelfi@unoeste.br.

The authors would like to thank the partnership between Huawei Company and Universidade de São Paulo (USP) for the support to this research

Digital Object Identifier: 10.14209/jcis.2021.11

challenges, there is the advantage that CWIoT is more energy-efficient than higher capacity devices [21].

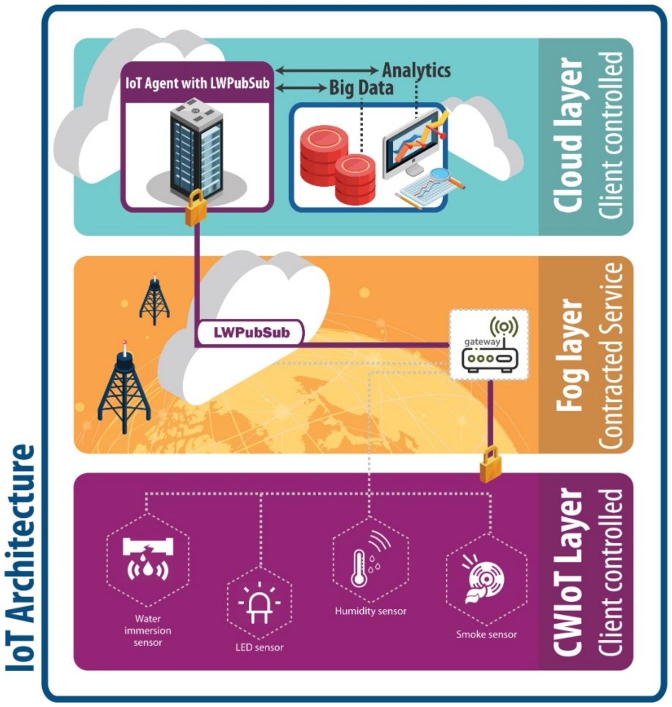


Fig. 1. Proposed lightweight, energy-efficient, and secure publish-subscribe system for IoT architectures

Considering the problems presented, the lack of standardization in publish-subscribe applications based on MQTT is a relevant gap. Standardization needs to be fully integrated into the MQTT topic and payload. On the cloud platform, the context-aware services [13; 15] allow embedding metadata (location, data type, device sensor used, and other characteristics) about the devices' data. Thus, both the MQTT topic and payload must be context-aware at the same time as being energy efficient.

Given the above, we observed that the WSN-cloud integration using MQTT lacks standardization and security. Also, CWIoT-cloud communication addresses other challenges, such as orchestrating possible billions of devices [1; 8] at the cloud platform.

Also, Transport Layer Security (TLS), as used by [22] on 802.11 devices, is not an option for WSN devices given its constrained nature [23; 24]. Because of that, a security mechanism must be used to overcome this issue.

Thus, our contribution consists of a LightWeight, energy-efficient, and end-to-end secure Publish-Subscribe system based on MQTT (LWPubSub) to interconnect WSN devices (CWIoT) to the cloud. Our system provides MQTT topic and payload standardization, which permits unique identification of a device (and each of its sensors) within the IoT architecture, and meets the CWIoT restrictions. The messages also accomplish the triad confidentiality, authenticity, and integrity system, providing end-to-end security between the CWIoT and the cloud platform. In this secure scenario, one can use any MQTT Broker to deliver the LWPubSub messages. At the cloud platform, the orchestration and provisioning of CWIoT devices occur in domains [1; 6; 7] (for instance, school, hospital, transportation, industry, among others) and their sites (each network of a specific domain). The IoT Agent, at the cloud platform, is the service responsible for receiving messages from CWIoT on the cloud platform [15; 25].

Fig. 1 presents the panorama of the system within the proposed IoT architecture.

The LWPubSub represents advances in interconnecting constrained devices to the cloud. The results obtained in comparison with the main related works show a 40% lower topic and a 48% lower payload (even when comparing works that do not implement security). The most important result is related to energy since the use of LWPubSub on the CWIoT in an IoT Architecture is 98% more energy efficient than the best work identified in the related works.

B. Article organization

The remainder of this article is organized as follows. In Section II, we define the main aspects of the interconnection of CWIoT-cloud and introduce the security concerns in IoT. In Section III, we present our lightweight, energy-efficient, and secure publish-subscribe system used in our proposed IoT architecture to interconnect the CWIoT with the cloud platform. The evaluation of the proposed system is in Section IV, and we present the results, discussion and comparison with the main related works in Section V. We highlight the main related works in Section VI. Finally, in Section VII, conclusions and future works are given.

II. WSN-CLOUD SECURE COMMUNICATION

Cloud environments support prediction and detection security controls [26; 27]. Often security is in the devices and systems used in the supported services. WSN devices (CWIoT) in this work are 6LoWPAN devices: embedded systems restricted in processing, memory, payload, and energy resources [12].

The connection of the CWIoT to the Internet (as stated in RFC 6282) requires the use of the 6LoWPAN Border Router (6LBR) [28]. 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.

For link-local communication, the 6LoWPAN network provides the devices with an energy-efficient link-layer protocol to exchange packets: the Time Slotted Channel Hopping (TSCH) protocol. TSCH is defined in the IEEE 802.15.4-2015 standard and is highly reliable and energy-efficient [29–31]. In a 6LoWPAN TSCH network, time is divided into slots, with the clock of all devices synchronized by the network coordinator (6LBR), with transmission and reception actions occurring at predefined times. There are mainly three types of slots: reception, transmission, and sleep (to save energy) [21; 29–31].

However, even with their limitations, 6LoWPAN devices can perform end-to-end communication and end-to-end security [7].

With cloud integration, CWIoT provides large volumes of raw data that need to be appropriately stored by the cloud (which, among other tasks, should analyze the data to remove outliers and misreadings) [32]. This performance of the cloud platform makes the data decision process more efficient.

For the decision to be more efficient, the raw data must contain some meaningful context [32–34]. Context-aware services at the cloud platform provide semantic meaning for the data (typically, the context information comprises a place and a device sensor, among others [17; 33; 35].

The MQTT publish-subscribe protocol use a server to deliver messages between the devices and the cloud: the MQTT Broker.

MQTT is a reliable protocol (by using TCP) for message delivery in IoT scenarios [36]. MQTT uses the publish/subscribe architecture and enables the interaction of multiple endpoints (broker, publisher, and subscriber) [18; 20], as presented in Fig. 2.

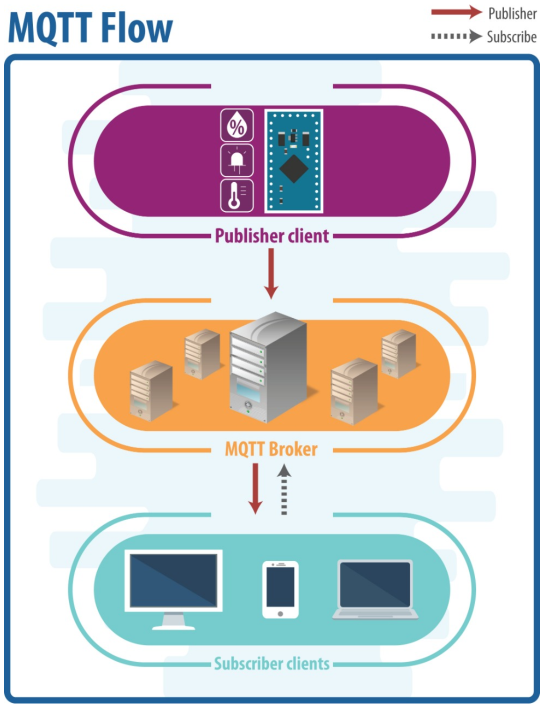


Fig. 2. MQTT protocol communication architecture

As seen in Fig. 2, the central entity in this communication is the MQTT Broker. This server allows communication between clients, which can operate as publishers (to send data), as subscribers (to receive data), or both. Interactions between clients are moderated by the broker, so clients do not need to know each other to exchange messages [20]. A client is a peer, an application, or a device that exchanges application messages over a given topic with another client. Client communication occurs at a request to the broker 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 [1; 16; 20].

The main MQTT messages are [16]: CONNECT (connect a client to the broker), SUBSCRIBE (subscribe/unsubscribe a client to/from a topic), PUBLISH (send a message from a publisher to the broker or from the broker to a client who previously subscribed to the topic).

The communication architecture of the MQTT protocol, shown in Fig. 2, exemplifies a communication with an IoT device (publisher) posting data from its sensors and actuators on a given topic. The other clients (subscribers) - laptop, desktop, smartphone - who previously subscribed to that topic, receive the message posted by the device through the MQTT broker.

Topics are hierarchically organized in a tree structure using the "/" separator. The payload (second field) is topic dependent.

As presented by [37; 38] it is common to use JSON in the payload of the MQTT message. In this way, an MQTT message (topic and payload) could be: "school/firstfloor/kitchen {"temp": "25.00"}".

We observe that the main disadvantage of MQTT is the lack of standardization in both fields of the MQTT message, causing messages to varying greatly in size, from tens to hundreds of Bytes [18–20], which may require incompatible resources of a CWIoT, and can lead to an unfeasible communication.

However, even with end-to-end message delivery using MQTT, end-to-end security in the CWIoT/WSN-Cloud union is a challenge (given the restricted characteristics of CWIoT) and a mandatory requirement, according to [7; 11; 28; 34; 39].

According to [40], it is urgent to research security issues in IoT. For instance, security is essential for the following IoT scenarios: patient data [34], node location [41], message traffic [2].

However, correctly apply end-to-end security is critical. The work of [10] presents an example of data leakage with the inappropriate use of encryption (mainly related to the repetition of the initialization vectors with the same key).

Another example of the necessity of secure messages is with smart homes. If the messages are unencrypted, a compromised smart lock may permit unrestricted access to the home [7].

III. LWPubSub System for WSN-Cloud Integration using Secure MQTT Messages

This section presents our LWPubSub system [1] based on MQTT, providing WSN devices-cloud integration with end-to-end security.

We understand the IoT as an architecture involving the union of devices, gateways, networks, and servers. Hence we use the term IoT architecture, also used by [1; 7; 10].

Because of that, we propose an IoT architecture able to manage the large amount of CWIoTs (and their sensors) attached to it on the cloud platform.

The IoT architecture presented in Fig. 3 comprises three layers, structured as follows:

- CWIoT: sends MQTT messages with end-to-end security using the LWPubSub system - each device has a deviceID, and each device's sensor has the respective

[1]The LWPubSub system (for cloud platform and CWIoT) is available at github.com/norisjunior/LWPubSub

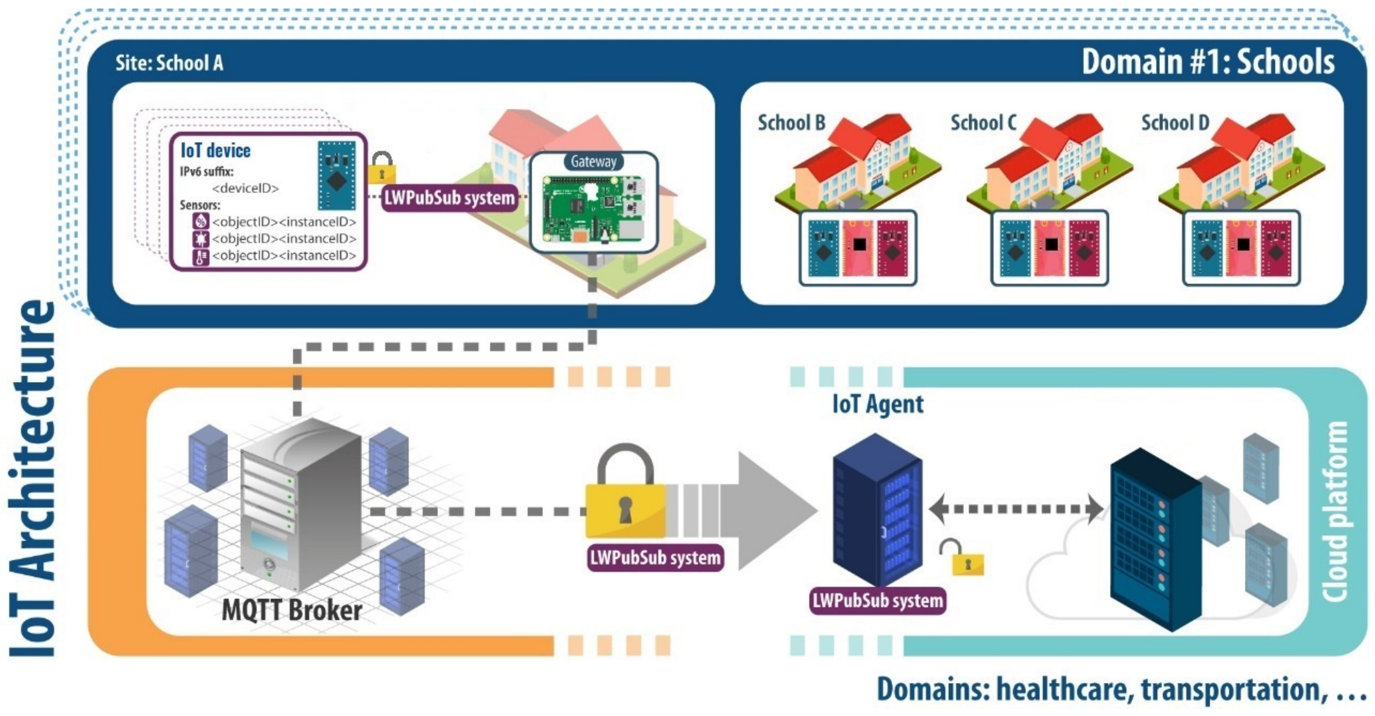


Fig. 3. Conceptual scheme of the proposed IoT architecture

objectID/instanceID metadata, where objectID is the type of the sensor and instanceID is the sensor itself. The gateway (6LBR) is an unconstrained device (although more compact than edge and core routers), usually connected to a continuous power source, and is the coordinator of the WSN.

- MQTT Broker: the destination of the encrypted message generated by the LWPubSub system (on the CWIoT or the cloud platform).
- Cloud: sends commands to and receives measurements from the CWIoT (through the IoT Agent), storing and providing services for the IoT data.

The IoT architecture can act in several domains, managing, orchestrating, and provisioning the CWIoT on the respective sites, together with their respective sensors. The cloud platform aggregates and manipulates the CWIoT sensors data using the provided metadata.

Our LWPubSub system receives the following requirements and parameters from the IoT architecture for its operation presents in Fig. 4. The **domains** are the vertical markets [1] that build the IoT architecture. A given IoT architecture, for example, 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). The **sites of each domain** specify the sites of a specific domain (the sites are the places to deploy and use the CWIoT) and its respective **IPv6 prefix**. The **IPv6 suffix of the CWIoT** (deviceID) ensures the unique identification of each CWIoT at a domain. We utilize the *objectID/instanceID* metadata provided by the IPSO registry [42] to identify the CWIoT sensors.

The system requirements presented are essential and mandatory in an IoT Architecture. These requirements guide the required number of bytes of the MQTT topic and payload used in the publish-subscribe actions. The number of transmitted bytes directly impacts the energy efficiency of the CWIoT and, consequently, the architecture as a whole. Fig. 4 shows the LWPubSub structure.

According to Fig. 4, the LWPubSub contains three structures for the execution of its tasks: Cross-platform publish-subscribe message, Cloud platform layer, and CWIoT layer.

LWPubSub constructs the MQTT message (topic and payload) as shown in Fig. 4. We define two topics in the system: one for sending messages from CWIoT to the cloud platform (`/<# of domain>/<CWIoT IPv6 suffix>`) and another for sending from the cloud to CWIoT (`/<# of domain>/<CWIoT IPv6 suffix>/cmd`) - the first part contains the domain number, and the second the IPv6 suffix of the CWIoT.

Still in the sense of standardization, the MQTT payload contains the *objectID* and *instanceID* metadata, and the data to be sent (`<objectID><instanceID>|<data>`).

For the LWPubSub system use any MQTT Broker, the system sends the topic in plaintext, which guarantees the end-to-end delivery of messages between MQTT clients without the need for an intermediate proxy.

Furthermore, to guarantee end-to-end security, the LWPubSub system encrypts the MQTT payload using symmetric encryption. Considering the restrictions of CWIoT, the Advanced Encryption Standard (AES) is safe and suitable for this purpose [43]. LWPubSub permit the use of AES Cipher-Block-Chaining (AES-CBC) or Counter (AES-CTR), which require a 16-byte Initialization Vector (IV), that must never be repeated together with the key for the same message. Thus,

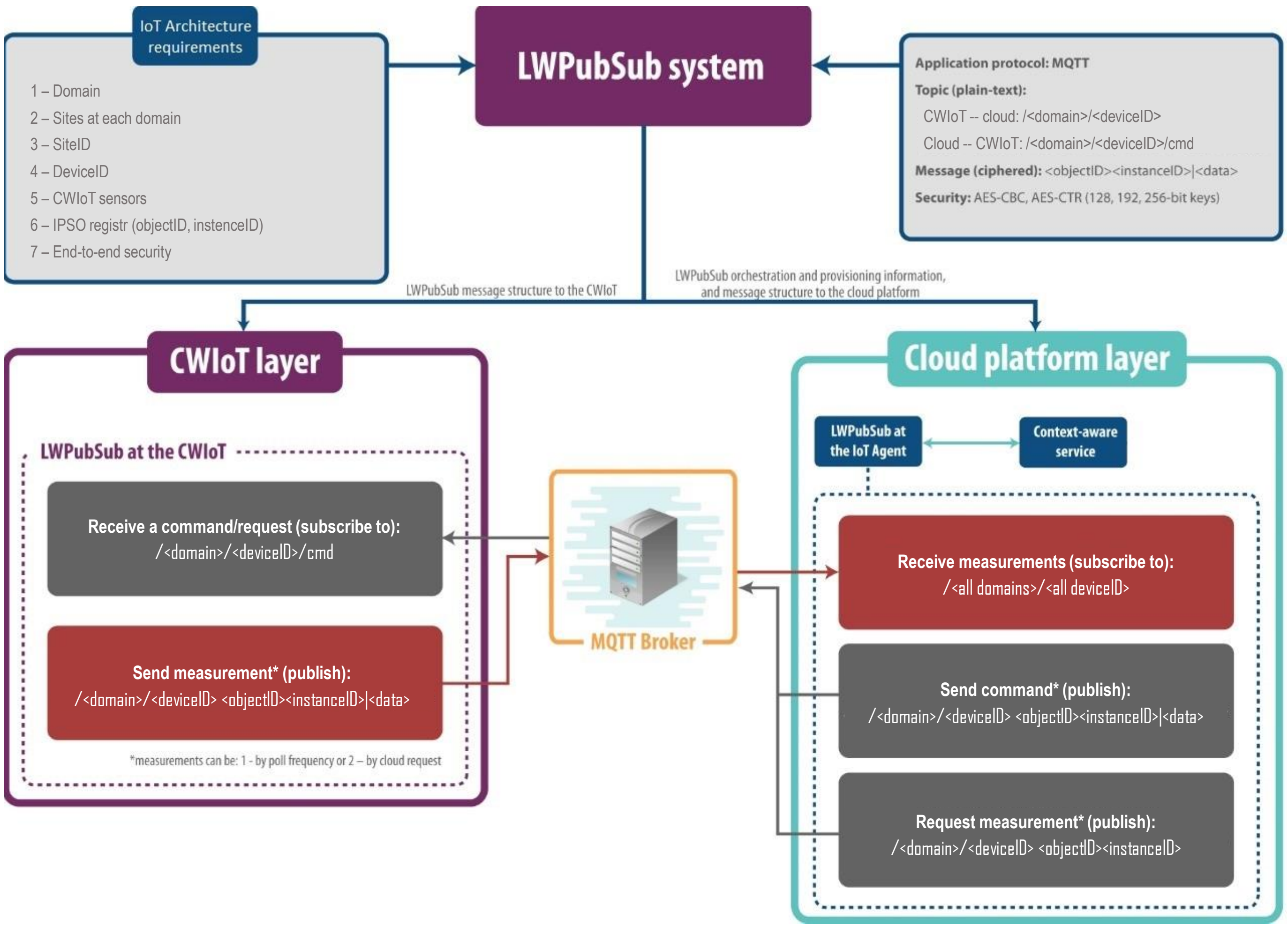

IoT Architecture requirements
1 – Domain
2 – Sites at each domain
3 – SiteID
4 – DeviceID
5 – CWIoT sensors
6 – IPSO registr (objectID, instenceID)
7 – End-to-end security
LWPubSub system
Application protocol: MQTT
Topic (plain-text):
CWIoT -- cloud: /<domain>/<deviceID>
Cloud -- CWIoT: /<domain>/<deviceID>/cmd
Message (ciphered): <objectID><instanceID>|<data>
Security: AES-CBC, AES-CTR (128, 192, 256-bit keys)
LWPubSub message structure to the CWIoT
LWPubSub orchestration and provisioning information, and message structure to the cloud platform
CWIoT layer
LWPubSub at the CWIoT
Receive a command/request (subscribe to): /<domain>/<deviceID>/cmd
Send measurement* (publish): /<domain>/<deviceID> <objectID><instanceID>|<data>
*measurements can be: 1 - by poll frequency or 2 – by cloud request
MQTT Broker
Cloud platform layer
LWPubSub at the IoT Agent
Context-aware service
Receive measurements (subscribe to): /<all domains>/<all deviceID>
Send command* (publish): /<domain>/<deviceID> <objectID><instanceID>|<data>
Request measurement* (publish): /<domain>/<deviceID> <objectID><instanceID>

Fig. 4. Structure of the LWPubSub system for IoT architectures

before sending each message, the CWIoT and the IoT Agent must generate a random 16-byte number to be used as the IV.

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

1) The first byte is an identifier of the encryption algorithm used.
2) The next 16 bytes refer to IV.
3) The next bytes (at least 16) refer to the encrypted message.

We consider end-to-end secure the messages exchanged between the CWIoT and the IoT Agent.

Regarding the CWIoT communication, radio uptime is the device resource that requires more energy [21; 44; 45]. Therefore, two resources require attention: the link-layer protocol must have low energy consumption, and messages should avoid fragmentation since the MTU on 6LoWPAN networks is limited to 127 bytes. Thus, we use the TSCH protocol, which provides reliability in the delivery of messages [29; 30] while presenting low energy consumption compared to other link-layer protocols used in 6LoWPAN networks [29].

At the cloud platform (as shown in Fig. 4) occurs the CWIoT provisioning according to the metadata of the devices sensors (*objectID*/*instanceID*). After provisioning, the cloud can receive contextualized data from each CWIoT sensor.

After cloud provisioning, CWIoTs can initialize. The LWPubSub system (as shown in Fig. 4) defines three types of messages, according to the following:

- CWIoT-cloud (most common case):
  - Send measurement: the CWIoT obtains the sensor measurement, and composes the MQTT payload by inserting the metadata (*objectID* and *instanceID*), followed by the data. In this case, the CWIoT sends messages after waiting for predetermined time intervals or after a request from the cloud platform.
- Cloud-CWIoT:
  - Request measurement: requires the CWIoT to respond to the requested sensor measurement immediately, sending in the MQTT payload only the *objectID* and *instanceID* metadata that represent from which sensor of the CWIoT it wants to receive the measurement.
  - Command execution: similarly to “Request measurement”, in this message the cloud platform sends, in addition to metadata, the command to be executed in the “data” field.

Regardless of the originator, the MQTT payload is encrypted and linked to the respective MQTT topic, thus composing the LWPubSub message.

As shown in Fig. 4, the cloud platform subscribe to all topics of its orchestrated and provisioned CWIoT (excluding, therefore, the topics that end with "/cmd"). CWIoT subscribes exclusively on the topic of its deployment domain (and with its IPv6 suffix) - the third part of the topic ends with "/cmd". Upon receiving the message, the actions performed are:

- Cloud platform: the IoT Agent interprets the MQTT topic to define the orchestration and provisioning location (domain, site, CWIoT triad). After that, the context-aware service store, along with previous information, the sensor measurements sent in the MQTT payload.
- CWIoT: execute a command or send a measurement from a sensor based on the cloud platform's message.

IV. LWPubSub Evaluation and Results

To analyze the efficiency and performance of the proposed LWPubSub system, we conducted several experiments. In this section, we also present the results.

We observe and evaluate, with the execution of the experiments, the following metrics:

1) Energy consumption: intends to observe the behavior of CWIoT in different operating conditions, obtaining measurements with variations in poll frequencies and encryption modes (both further detailed).
2) MQTT topic and payload: evaluates the LWPubSub topic and message sizes.
3) Footprint: firmware size required in CWIoT memory (ROM).

For the validation of the LWPubSub system and to collect the results, we use the experimental scenario presented in Fig 5.

In Table I, we summarize the characteristics of the equipment, devices, sensors, and the parameters of the LWPubSub system.

For the validation of the IoT Architecture, and considering as a premise the use of open standards, the FIWARE platform [15; 46] provides the needed infrastructure 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 [46]. We developed an IoT Agent for MQTT (based on the existing FIWARE) that interprets LWPubSub messages (an additional contribution of this work), and we integrated it in the FIWARE-based cloud platform.

The cloud platform is the first layer of an IoT Architecture that needs to be structured and initialized. In the cloud platform, the domains, sites, and devices are orchestrated and provisioned first (in the case of this experiment, we provision the domain "99", prefix "fd00::/64", and the CWIoT suffix "00124b05257a"). Based on this triad, we create the following structure using *objectID* and *instanceID*: 33030 (HDC temperature sensor), 33110 (red LED), 33111 (green LED), and 33380 (alarm). We use the pattern defined in Section III with the use of IPSO Objects concatenated with the object's existing instance (sensor).

The MQTT Broker, based on *mosquitto* [47] runs at the Huawei Cloud. The IoT gateway runs on a Raspberry Pi model 3B, which contains an IEEE 802.11 interface connected to the Internet and an IEEE 802.15.4 interface (via Launchpad - the 6LBR) for communication with the 6LoWPAN network.

Next, we start the CWIoT and its network, since the cloud platform is structured to receive the lightweight, context-aware, and secure LWPubSub messages.

The devices of the validation scenario are the Texas Instruments Sensortag (CWIoT) and Launchpad (6LBR), both with CC2650 (IEEE 802.15.4) radio, 128 KB ROM, 20 KB RAM. To comply with the energy-efficient goal of the LWPubSub, we use the Time Slotted Channel Hopping (TSCH) as the link-layer protocol as it presents the lowest energy consumption relative to this layer on 6LoWPAN networks [30] (we use TSCH minimal schedule). Above TSCH, the 6LoWPAN protocol uses RPL, followed by the TCP protocol that loads the MQTT payload with the LWPubSub application.

We use Contiki-NG, an opensource Operating System (OS) for CWIoT, which supports the constrained features of devices while allowing the development of firmware for specific applications such as MQTT.

Thus, as can be seen in Fig 5, the 6LBR (Launchpad) receives the LWPubSub messages from the CWIoT (Sensortag) and forwards them to MQTT Broker (Raspberry Pi).

For end-to-end security, we implement symmetric encryption with the system being able to use AES-CBC or AES-CTR with 128, 192, and 256-bit keys.

In possession of the metadata (*objectID* and *instanceID*) and the data, the LWPubSub system follows these steps:

1) IoT Agent: Handle received data using context-aware service.
2) CWIoT: Sends requested measurement or executes a command.

Fig. 6 presents a measurement sample taken from the experiments. In this example, the 16 bytes of the topic (not ciphered) - uniquely identifies the CWIoT on the domain. The first 5 bytes of the payload uniquely identifies the HDC temperature sensor of the Sensortag (the remaining bytes comprise the separator between metadata and data - the measurement sent). Only the payload (11 bytes) is encrypted. We note the importance of end-to-end security, since if the LWPubSub system does not encrypt the MQTT payload, the device structure and data may be maliciously observed in the MQTT Broker.

IoT applications may require different time intervals when collecting sensor measurements, which can range from use as wearable (which requires many measurements per minute) to use as measuring environmental conditions (which requires 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:

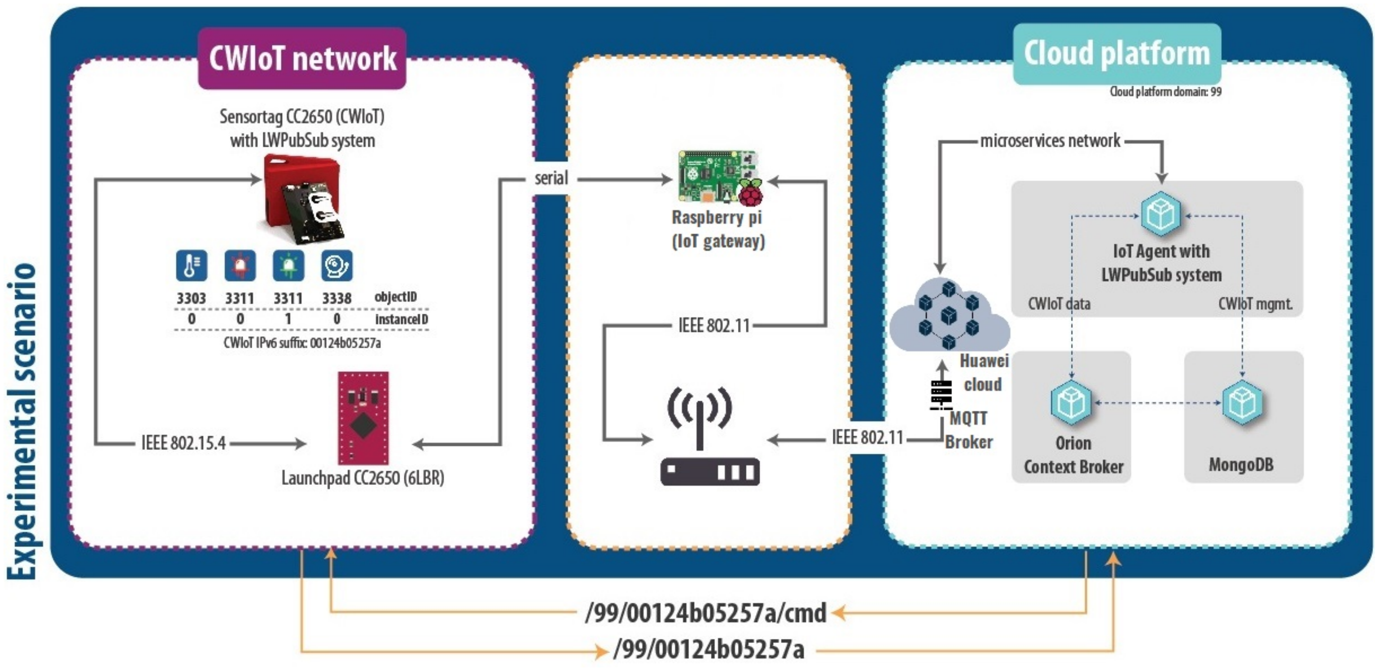


Fig. 5. Experimental scenario

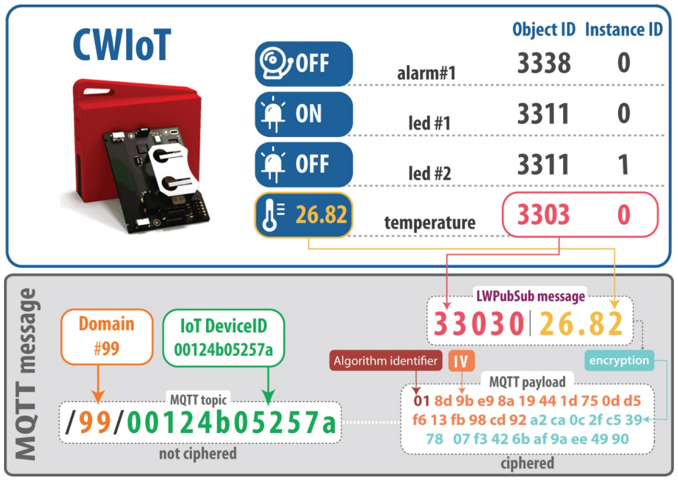


Fig. 6. Example internal message construction with sample measurement obtained from the LWPubSub system in the experiments

- 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.

We performed a 24-hour experiment for each poll frequency, totaling five days of the experiment, with periodic sending of temperature sensor measurements.

V. Results and Discussion

The experiments performed have the purpose of observing the footprint, size in bytes of the topic and payload, and the energy required from CWIoT for two situations:

1) Connection and subscribe actions with MQTT Broker.
2) Exchange of messages considering the execution of the activities outlined in Fig. 4:
   - CWIoT: send measurement at specified time interval, send sensor measurements after receiving a request, execute action after receiving a command.
   - Cloud platform: receive sensor measurement, send command execution, send measurement request.

In the first round of experiments, we performed each of the previous CWIoT activities 100 times (we waited 10 seconds between executing each event) at each of the six encryption modes (AES-CBC or AES-CTR with 128, 192, and 256-bit keys). These experiments allow us to obtain the mean and standard deviation of the energy involved in these operations, making a total of 1800 messages.

With the conclusion of this first round of experiments, we summarize the main results of LWPubSub in Table II.

The second round of experiments aims to evaluate the CWIoT in a scenario of periodic sending of measurements. Thus, it will consider the wide variety of IoT applications presented in Section IV, which require different time intervals when sending measurements. Therefore, the experiments based on poll frequencies shown in Table I were executed (each one) for 24 hours, so we evaluate the real conditions of an IoT device with the use of LWPubSub. The main objective is to evaluate the daily energy consumption of a CWIoT.

TABLE I
EXPERIMENT PARAMETERS

Equipment and Devices	
Cloud platform	Microsservices/docker containers
MQTT Broker	Mosquitto on Raspberry Pi Model 3B
Border Router	Launchpad CC2650 coupled with R.Pi
CWIoT (IoT end-device)	Sensortag CC2650
CWIoT O.S.	Contiki-NG v.4.4
Cloud platform parameters	
Platform	FIWARE-based
Context-Broker	Orion 2.3.0
Database	Mongo DB 3.6
CWIoT parameters and current consumption	
Link-layer	TSCH-minimal schedule
TX	7.9 mA
RX	6 mA
Low Power Mode (LPM)	0.55 mA
CPU	3.48 mA
LWPubSub system parameters	
Domain	99
IPv6 prefix	fd00::/64
CWIoT (Sensortag) suffix	00124b05257a
LWPubSub IoT Agent	version 0.12
Topic for commands	/99/00124b05257a/cmd
Topic for measurements	/99/00124b05257a
objectID	3303, 3311, and 3338
Encryption modes	AES-CBC and AES-CTR
Encryption key sizes	128, 192, and 256
Polling frequency	very high: 5 per sec high: quarter of a minute medium: quarter of an hour low: quarter of a day very low: 1 per day

TABLE II
LWPUBSUB MAIN RESULTS

Footprint (Bytes)	
LWPubSub system application	9507
Other Contiki-NG functions	98088
LWPubSub message size (Bytes)	
Topic (measurement*)	16
Topic (command**)	20
Encrypted payload (measurement*)	33
Encrypted payload (command**)	33
Total IPv6 packet (measurement*)	114
Total IPv6 packet (command**)	122
**measurement: from CWIoT to Cloud*	
**command: from Cloud to CWIoT*	
Energy (mJ)	
Connect to MQTT Broker	19.800
Subscribe to MQTT Broker	6.367
CWIoT - send measurement	0.940(±0.114)
CWIoT - execute a command	0.985(±0.070)
CWIoT - receive a request and send a response	1.986(±0.142)

A. Standardization and interoperability of MQTT messages

As stated by [48], standardization decreases the gaps and reduce system complexity. Our LWPubSub system goes in this direction once we provide a unique MQTT topic to each device and standardized the metadata.

Regarding the **MQTT topic**, our system design proposes a short and lightweight, yet complete, MQTT topic. The MQTT topic generated by our proposed system retain the unique identification of devices in the respective domain.

Our proposal is opposed to the various messages exchanged between devices - a resource used by [18; 19], which generates longer messages. The smallest possible MQTT topic observed in the work of [18] is 27 Bytes, while in [19] it is 33 Bytes.

The work of [20] covers the interconnection of IoT devices to the cloud platform. However, it does not act in standardizing the MQTT messages and presents different topics when exchanging control and data messages. For data messages, the topic is at least 70 bytes.

Fig. 7 presents our MQTT topic results, comparing it with the results of the main related works (the smaller, the better).

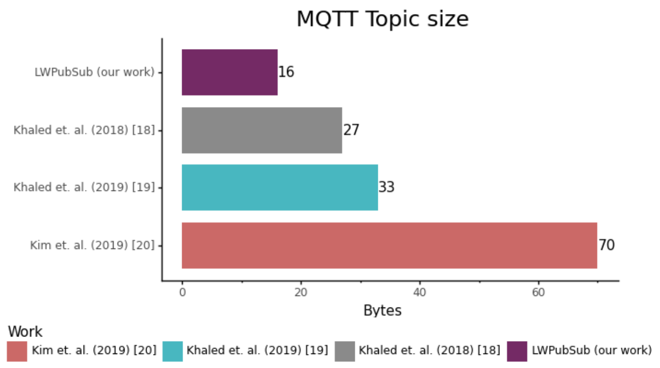


Fig. 7. MQTT topic size results and comparison with the main related works

Regarding the **MQTT payload**, the experiments demonstrated that an humidity or temperature messages using LWPubSub (MQTT payload with confidentiality, authenticity, and integrity) is 32 Bytes long. The total IPv6 packet is 113 (CWIoT-cloud) or 117 (cloud-CWIoT) Bytes, thus, there is no fragmentation of messages sent/received by the CWIoT. The lowest MQTT payload observed in the works of [18; 19] is 64 Bytes, and the total IP packet is even greater.

Fig. 8 presents our results, comparing it with the main related works (the smaller the message, the better). Our proposal still includes end-to-end security, while only [18] applies security.

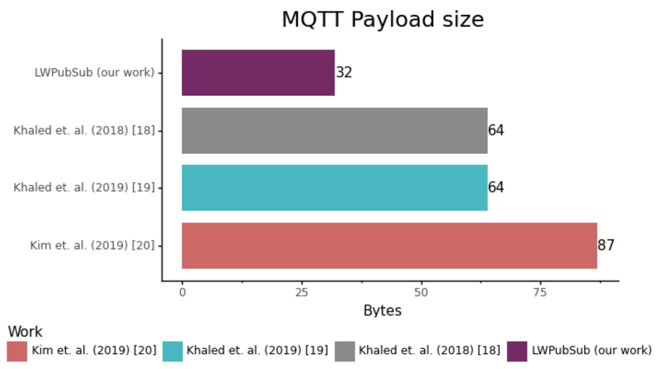


Fig. 8. Payload results and comparison with the main related works

Considering the MQTT messages evaluated, and the MTU of 6LoWPAN messages (127 Bytes), our LWPubSub messages does not resort to fragmentation.

B. Footprint

The result of the footprint shown in Table II comprises the LWPubSub system developed in Contiki-NG using the Sensortag CWIoT. We segregate the LWPubSub system from the rest of the operating system's (OS) functions to identify the footprint required in ROM by the application. The other OS functions, such as TSCH, RPL, 6LoWPAN, and TCP, among others, require 98088 Bytes. The LWPubSub system (considering the application itself that runs on the MQTT plus the encryption functions) corresponds to 9507 Bytes (the CBC and CTR modes and the different AES-key sizes do not influence the space required in the memory).

The use of LWPubSub system in CWIoT brings advantages relative to the footprint. Our 9.5KB footprint covers the entire MQTT standard stack plus the LWPubSub system inside the CWIoT and is smaller than [19]. The work of [18] presents a 13MB footprint but includes not only the MQTT in the footprint calculation (even though, extracting, for example, half of the footprint requires many more bytes than the LWPubSub footprint results). Fig. 9 presents the footprint comparison (the smaller the footprint, the better).

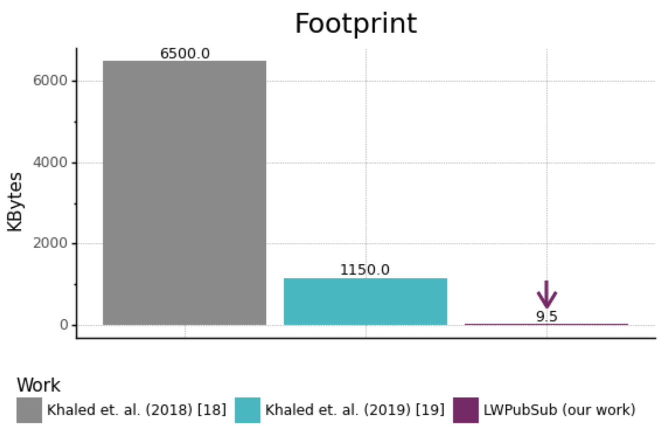


Fig. 9. Footprint results and comparison with the related works

C. Energy consumption

For energy consumption calculation, we proceed according to Eq. 1, using the Contiki-NG Energest [49] module. We calculate the energy (in millijoules) from the time spent (in seconds) of each CWIoT 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 I.

$$Energy_{CPU,LPM,TX,RX} = time * V * I \tag{1}$$

The CWIoT connection with the MQTT Broker requires 19.800 mJ, while MQTT subscribe action requires 6.367 mJ - these actions do not depend on the encryption algorithm since they are not encrypted.

The energy consumption of the LWPubSub presented in Fig. 10 refers to the CWIoT messages "Send measurement", "Execute a command", and "Receive a request and send response". These messages require respectively, on average, 0.985, 0.940, and 1.986 mJ (considering all measurements covering all encryption modes and key sizes). In Fig. 10, we present the detailed energy consumption of these messages.

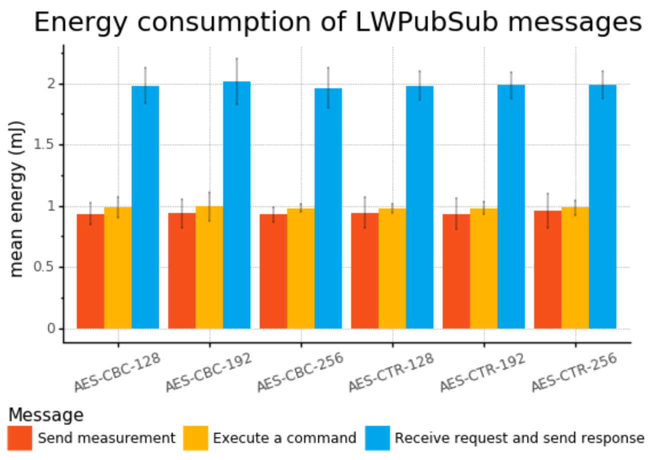


Fig. 10. CWIoT messages energy consumption

The encryption modes have similar power consumption, with an overall average (CBC or CTR, using 128, 192, and 256-bit keys) of 0.123 mJ (± 0.005).Therefore, for the second round of experiments, we use the AES-CTR-256 mode because it contains the largest key size and provides the most robust security.

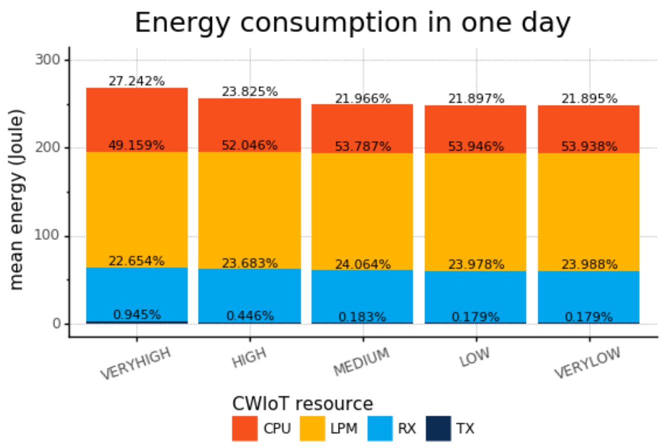


Fig. 11. CWIoT energy consumption in one day

We summarize the results of daily energy consumption in Fig. 11. Observing the results presented in Fig 11, regardless of the poll frequency, the commands sent by the cloud platform consume significantly lower energy compared to a day of CWIoT usage (which sends measurements periodically). Therefore, we conclude that one can intersperse the messages sent from the cloud to the CWIoT with the messages previously configured in the poll frequencies once what guides the energy consumption of the CWIoT are: sleep time (LPM) and listening (RX). CPU and TX consumption barely rises and is practically the same for medium, low, and very low poll frequencies. Thus, sending 96, 4, or 1 message per day (medium, low, and very low poll frequencies) consumes almost the same amount of energy.

Furthermore, the "very high" poll frequency, which demands more energy from the CWIoT, ends up not differing from the other poll frequencies' energy consumption, even those that wait longer between measurements such as "low" and "very low".

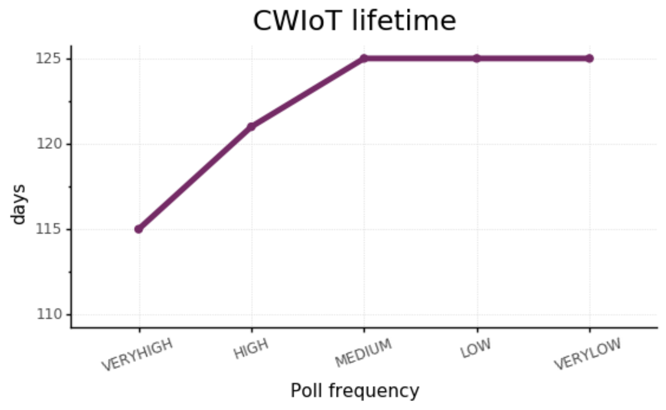


Fig. 12. CWIoT lifetime on the evaluated poll frequencies

We can observe the previous comparison of energy consumption when looking at the battery life of the CWIoT at each poll frequency. Two AA batteries provide 30780 J ($2.85Ah * 1.5V * 3600secs. * 2$) of energy. We present in Fig. 12 the battery life in each experiment scenario, given the poll frequencies presented and the daily energy consumption in Fig. 11.

The use of CWIoT on an IoT Architecture was the better decision once the constrained device can perform the same tasks as an unconstrained device like Raspberry Pi, relative to MQTT messages.

Our system requires 0.940 mJ of energy of a CWIoT to send a message (details in Table II and Fig. 10). This energy consumption requires 41 times less energy than the best result of [19] - 39.47 mJ (their work does not apply security) and 320 times less energy than the best result of [18] - 301 mJ.

Regarding receiving messages and executing commands, our LWPubSub system also demonstrates an advantage over the works of [18; 19]. Our proposal requires 1.986 mJ of energy, while the work of [18] requires 561 mJ and [19] 39.9 mJ.

The LWPubSub system is also advantageous for connecting with the MQTT Broker, which requires 19.800 mJ, while [18] requires 222 mJ, and [19] 298.5 mJ. The work of [20] does not present results regarding the energy consumption of IoT devices.

The comparative results are consolidated and presented in Fig. 13.

D. Security

The LWPubSub system applies confidentiality to the MQTT payload allowing the use of six encryption modes. As a comparison, [18] has only one mode, which contains a vulnerability: the authors use the same generated session key

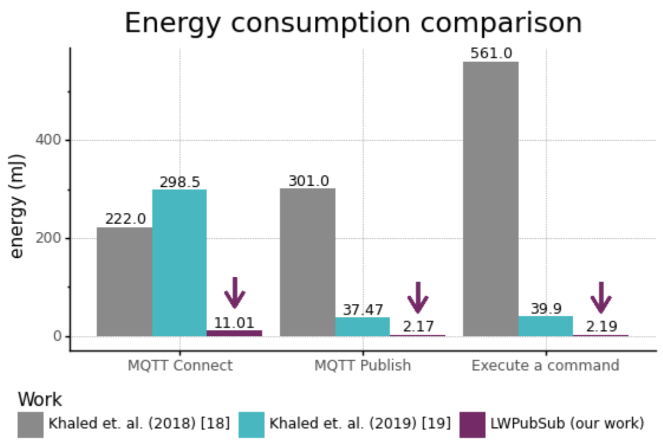


Fig. 13. Energy results compared with the related works

to encrypt the same message (for instance, on/off a sensor). Thereby, new encrypted on/off messages transmitted later will be identical. By contrast, our system sends a byte to identify the encryption algorithm and a new IV at each message to not reuse the triad: key, IV, message.

Moreover, we achieve end-to-end security between MQTT clients (CWIoT and IoT Agent), thus allowing the use of any MQTT Broker without customization.

VI. Related works

Regarding cloud-to-devices messaging using MQTT, the works of [18–20] aim to exchange MQTT messages using unrestricted IoT devices. The work of [20] proposes integration to the cloud platform, while [18; 19] uses a framework developed by the authors for communication between IoT devices. Regardless of the approach, the previously cited works use robust equipment like the Raspberry Pi as an IoT device, which we use as a gateway between the network of IoT devices and the cloud. Conversely, our architecture proposes the use of CWIoT to perform sensing and actuation tasks in the environment.

Also, the platform proposed by [20] does not send periodic messages from the IoT device to the cloud. Instead, the cloud platform sends several rules to be followed by the devices that, in turn, perform the scheduled actions. Their platform also requests the device's sensors' status, which can generate messages of at least 87 Bytes. Furthermore, the works of [19; 20] do not use security to protect the MQTT payload.

About context-aware messages on WSN, the work of [50] proposes the integration of legacy embedded systems software applications, which uses TinyOS. The authors present in the conclusion as a future work add new features to provide context-awareness capabilities. We use context-aware messages in both the topic and the payload. These messages enable the provisioning of constrained devices and their sensors on the cloud platform.

In the sense of MQTT security for IoT, the work of [23] comprises three security levels for publish/subscribe MQTT messages. Although the authors cite constrained devices, they do not evaluate energy consumption nor message size. Also, the work of [23] does not present the structure of the MQTT topics. An implementation is essential to verify the suitability for constrained devices, such as footprint, message size (to evaluate fragmentation), and energy consumption. If a security mechanism requires an amount of energy that does not fit the constrained device's requirements, the solution is not an option.

Diro *et al.* [24] propose a security scheme in the interconnection of the IoT-device with the MQTT Broker. The proposed security mechanism requires a trusted and customized MQTT Broker to deliver modified MQTT messages between publishers and subscribers, breaking end-to-end security. The message is only secure because the MQTT Broker is responsible for ensuring the MQTT header's new security proposed fields. The proposed security mechanism contains only the proposal's schematics, and the authors do not implement the solution in real IoT devices.

VII. Conclusion

In this article, we presented and evaluated our LWPubSub system within a cloud-connected IoT architecture. Our system provides standardization and end-to-end secure messages between constrained wireless IoT devices and the cloud platform through any MQTT Broker.

Our LWPubSub system includes standardization to the MQTT messages, meeting the requirements of IoT architectures. Besides, the MQTT topic generated by the LWPubSub system guarantees the unique device identification on its domain and site. In the MQTT payload, we guarantee end-to-end security and the unique identification of each device's sensors. We include all these metadata (along with the data) in our LWPubSub system's message.

The demonstrated LWPubSub system standardization provides the lowest energy consumption among the main related works, the smallest footprint, and the smallest MQTT topic and payload size, as shown in our results.

The strategic decision of using constrained wireless IoT devices is critical for the success of the IoT architecture - given the ubiquity and proliferation of these devices and low energy consumption. Also, some sensors (temperature, humidity, led, light) are natively present in CWIoT, such as Sensortag.

Furthermore, compared to Raspberry Pi, CWIoTs are cheaper, restricted in size to be coupled with other equipment, and (even being constrained) can perform the same sensing and actuation functions in IoT scenarios using MQTT. We use the Raspberry Pi as a more robust device: the gateway, not the IoT end-device.

a) Limitations and Future works: The confidentiality applied to the MQTT payload ensures that an intermediary like the MQTT Broker cannot read the message, ensuring end-to-end security. However, one of the limitations of this paper is not to explore the use of authentication and integrity and the energy consumption needed by it. It should be addressed in future works, for example, using AES_CCM_8.

Although the results obtained prove the low energy consumption and message size without resorting to fragmentation, the LWPubSub system only sends one measurement from one sensor at a time. An extension to this work could be evaluating the message size to verify if sending measurements from more than one sensor in the same message requires fragmentation and what energy consumption and other implications this may cause. The authenticity and integrity of the MQTT payload are important features for future works. Additionally, other encryption models can take place when considering security, using key derivation functions and features to deliver the static keys, which also require analysis of the message size and energy consumption. Regarding total energy consumption, future work may include evaluating another TSCH Schedule.

References

[1] A. Al-Fuqaha, M. Guizani, M. Mohammadi, M. Aledhari, and M. Ayyash, "Internet of Things: A Survey on Enabling Technologies, Protocols, and Applications," *IEEE Communications Surveys and Tutorials*, vol. 17, no. 4, pp. 2347–2376, 2015. doi: 10.1109/COMST.2015.2444095

[2] J. Ni, X. Lin, and X. S. Shen, "Toward Edge-Assisted Internet of Things: From Security and Efficiency Perspectives," *IEEE Network*, vol. 33, no. 2, pp. 50–57, 2019. doi: 10.1109/MNET.2019.1800229

[3] H. Habibzadeh, T. Soyata, B. Kantarci, A. Boukerche, and C. Kaptan, "Sensing, communication and security planes: A new challenge for a smart city system design," *Computer Networks*, vol. 144, pp. 163–200, 2018. doi: 10.1016/j.comnet.2018.08.001

[4] P. Asghari, A. M. Rahmani, and H. H. S. Javadi, "Internet of Things applications: A systematic review," *Computer Networks*, vol. 148, pp. 241–261, jan 2019. doi: 10.1016/j.comnet.2018.12.008

[5] H. Elazhary, "Internet of Things (IoT), mobile cloud, cloudlet, mobile IoT, IoT cloud, fog, mobile edge, and edge emerging computing paradigms: Disambiguation and research directions," *Journal of Network and Computer Applications*, vol. 128, no. October 2018, pp. 105–140, 2019. doi: 10.1016/j.jnca.2018.10.021

[6] M. G. Samaila, M. Neto, D. A. B. Fernandes, M. M. Freire, and P. R. M. Inácio, "Challenges of securing Internet of Things devices: A survey," *Security and Privacy*, vol. 1, no. 2, p. e20, mar 2018. doi: 10.1002/spy2.20

[7] K. Sha, W. Wei, T. Andrew Yang, Z. Wang, and W. Shi, "On security challenges and open issues in Internet of Things," *Future Generation Computer Systems*, vol. 83, pp. 326–337, 2018. doi: 10.1016/j.future.2018.01.059

[8] Z. Shelby and C. Bormann, *6LoWPAN: the wireless embedded internet*. Wiley, 2011. ISBN 9780470747995

[9] D. Miorandi, S. Sicari, F. De Pellegrini, and I. Chlamtac, "Internet of things: Vision, applications and research challenges," *Ad Hoc Networks*, vol. 10, no. 7, pp. 1497–1516, sep 2012. doi: 10.1016/j.adhoc.2012.02.016

[10] N. Neshenko, E. Bou-Harb, J. Crichigno, G. Kaddoum, and N. Ghani, "Demystifying IoT Security: An Exhaustive Survey on IoT Vulnerabilities and a First Empirical Look on Internet-Scale IoT Exploitations," *IEEE Communications Surveys & Tutorials*, vol. 21, no. 3, pp. 2702–2733, 2019. doi: 10.1109/COMST.2019.2910750

[11] F. Montori, L. Bedogni, M. Di Felice, and L. Bononi, "Machine-to-machine wireless communication technologies for the Internet of Things: Taxonomy, comparison and open issues," *Pervasive and Mobile Computing*, vol. 50, pp. 56–81, oct 2018. doi: 10.1016/j.pmcj.2018.08.002

[12] A. Čolaković and M. Hadžialić, "Internet of Things (IoT): A review of enabling technologies, challenges, and open research issues," *Computer Networks*, vol. 144, pp. 17–39, oct 2018. doi: 10.1016/j.comnet.2018.07.017

[13] B. Cheng, S. Zhao, J. Qian, Z. Zhai, and J. Chen, "Lightweight Service Mashup Middleware with REST Style Architecture for IoT Applications," *IEEE Transactions on Network and Service Management*, vol. 15, no. 3, pp. 1063–1075, 2018. doi: 10.1109/TNSM.2018.2827933

[14] M. Bacco, L. Boero, P. Cassara, M. Colucci, A. Gotta, M. Marchese, and F. Patrone, "IoT Applications and Services in Space Information Networks," *IEEE Wireless Communications*, vol. 26, no. 2, pp. 31–37, apr 2019. doi: 10.1109/MWC.2019.1800297

[15] V. Araujo, K. Mitra, S. Saguna, and C. Åhlund, "Performance evaluation of FIWARE: A cloud-based IoT platform for smart cities," *Journal of Parallel and Distributed Computing*, vol. 132, pp. 250–261, 2019. doi: 10.1016/j.jpdc.2018.12.010

[16] D. Glaroudis, A. Iossifides, and P. Chatzimisios, "Survey, comparison and research challenges of IoT application protocols for smart farming," *Computer Networks*, vol. 168, p. 107037, 2020. doi: 10.1016/j.comnet.2019.107037

[17] D. Tracey and C. Sreenan, "OMA LWM2M in a holistic architecture for the Internet of Things," in *2017 IEEE 14th International Conference on Networking, Sensing and Control (ICNSC)*. IEEE, may 2017. doi: 10.1109/ICNSC.2017.8000091. ISBN 978-1-5090-4429-0 pp. 198–203.

[18] A. E. Khaled, A. Helal, W. Lindquist, and C. Lee, "IoT-DDL-Device Description Language for the 'T' in IoT," *IEEE Access*, vol. 6, pp. 24 048–24 063, 2018. doi: 10.1109/ACCESS.2018.2825295

[19] A. E. Khaled and S. Helal, "Interoperable communication framework for bridging RESTful and topic-based communication in IoT," *Future Generation Computer Systems*, vol. 92, pp. 628–643, mar 2019. doi: 10.1016/j.future.2017.12.042

[20] G. Kim, S. Kang, J. Park, and K. Chung, "An MQTT-Based Context-Aware Autonomous System in oneM2M Architecture," *IEEE Internet of Things Journal*, vol. 6, no. 5, pp. 8519–8528, 2019. doi: 10.1109/JIOT.2019.2919971

[21] É. Morin, M. Maman, R. Guizzetti, and A. Duda, "Comparison of the Device Lifetime in Wireless Networks for the Internet of Things," *IEEE Access*, vol. 5, pp. 7097–7117, 2017. doi: 10.1109/ACCESS.2017.2688279

[22] E. Anthi, S. Ahmad, O. Rana, G. Theodorakopoulos, and P. Burnap, "EclipseIoT: A secure and adaptive hub for the Internet of Things," *Computers and Security*, vol. 78, pp. 477–490, 2018. doi: 10.1016/j.cose.2018.07.016

[23] L. Malina, G. Srivastava, P. Dzurenda, J. Hajny, and R. Fujdiak, "A Secure Publish/Subscribe Protocol for Internet of Things," in *Proceedings of the 14th International Conference on Availability, Reliability and Security - ARES '19*. New York, New York, USA: ACM Press, 2019. doi: 10.1145/3339252.3340503. ISBN 9781450371643 pp. 1–10.

[24] A. Diro, H. Reda, N. Chilamkurti, A. Mahmood, N. Zaman, and Y. Nam, "Lightweight Authenticated-Encryption Scheme for Internet of Things Based on Publish-Subscribe Communication," *IEEE Access*, vol. 8, pp. 60 539–60 551, 2020. doi: 10.1109/ACCESS.2020.2983117

[25] I. Nakagawa and S. Shimojo, "IoT Agent Platform Mechanism with Transparent Cloud Computing Framework for Improving IoT Security," in *2017 IEEE 41st Annual Computer Software and Applications Conference (COMPSAC)*. IEEE, jul 2017. doi: 10.1109/COMPSAC.2017.156. ISBN 978-1-5386-0367-3 pp. 684–689.

[26] A. A. Silva, N. Ferraz Jr, A. E. Guelfi, S. Barboza, and S. T. Kofuji, "Grouping detection and forecasting security controls using unrestricted cooperative bargains," *Computer Communications*, vol. 146, pp. 155–173, 2019.

[27] N. F. Junior, A. Silva, A. Guelfi, and S. T. Kofuji, "IoT6Sec: reliability model for internet of things security focused on anomalous measurements identification with energy analysis," *Wireless Networks*, vol. 25, no. 4, pp. 1533–1556, 2019. doi: 10.1007/s11276-017-1610-2

[28] M. A. Khan and K. Salah, "IoT security: Review, blockchain solutions, and open challenges," *Future Generation Computer Systems*, vol. 82, pp. 395–411, may 2018. doi: 10.1016/j.future.2017.11.022

[29] A. Elsts, X. Fafoutis, G. Oikonomou, R. Piechocki, and I. Craddock, "TSCH Networks for Health IoT," *ACM Transactions on Internet of Things*, vol. 1, no. 2, pp. 1–27, apr 2020. doi: 10.1145/3366617

[30] S. Duquennoy, A. Elsts, B. A. Nahas, and G. Oikonomo, "TSCH and 6TiSCH for Contiki: Challenges, Design and Evaluation," in *2017 13th International Conference on Distributed Computing in Sensor Systems (DCOSS)*. Ottawa, Canada: IEEE, jun 2017. doi: 10.1109/DCOSS.2017.29. ISBN 978-1-5386-3991-7 pp. 11–18.

[31] B. Martinez, M. Montón, I. Vilajosana, and J. D. Prades, "The Power of Models: Modeling Power Consumption for IoT Devices," *IEEE Sensors Journal*, vol. 15, no. 10, pp. 5777–5789, 2015. doi: 10.1109/JSEN.2015.2445094

[32] L. Bittencourt, R. Immich, R. Sakellariou, N. Fonseca, E. Madeira, M. Curado, L. Villas, L. DaSilva, C. Lee, and O. Rana, "The Internet of Things, Fog and Cloud continuum: Integration and challenges," *Internet of Things*, vol. 3-4, pp. 134–155, oct 2018. doi: 10.1016/j.iot.2018.09.005

[33] E. de Matos, R. T. Tiburski, C. R. Moratelli, S. Johann Filho, L. A. Amaral, G. Ramachandran, B. Krishnamachari, and F. Hessel, "Context information sharing for the Internet of Things: A survey," *Com-*

puter Networks, vol. 166, p. 106988, 2020. doi: 10.1016/j.comnet.2019.106988

[34] P. H. Vilela, J. J. Rodrigues, P. Solic, K. Saleem, and V. Furtado, "Performance evaluation of a Fog-assisted IoT solution for e-Health applications," *Future Generation Computer Systems*, vol. 97, pp. 379–386, 2019. doi: 10.1016/j.future.2019.02.055

[35] C. Granell, D. Havlik, S. Schade, Z. Sabeur, C. Delaney, J. Pielorz, T. Usländer, P. Mazzetti, K. Schleidt, M. Kobernus, F. Havlik, N. R. Bodsberg, A. Berre, and J. L. Mon, "Future Internet technologies for environmental applications," *Environmental Modelling and Software*, vol. 78, pp. 1–15, 2016. doi: 10.1016/j.envsoft.2015.12.015

[36] C. Gomez, A. Arcia-Moret, and J. Crowcroft, "TCP in the Internet of Things: From Ostracism to Prominence," *IEEE Internet Computing*, vol. 22, no. 1, pp. 29–41, jan 2018. doi: 10.1109/MIC.2018.112102200

[37] M. Fischer, D. Kumper, and R. Tonjes, "Towards improving the privacy in the MQTT protocol," *Global IoT Summit, GIoTS 2019 - Proceedings*, 2019. doi: 10.1109/GIOTS.2019.8766366

[38] P. Bellavista, J. Berrocal, A. Corradi, S. K. Das, L. Foschini, and A. Zanni, "A survey on fog computing for the Internet of Things," *Pervasive and Mobile Computing*, vol. 52, pp. 71–99, 2019. doi: 10.1016/j.pmcj.2018.12.007

[39] D. Dinculeană and X. Cheng, "Vulnerabilities and Limitations of MQTT Protocol Used between IoT Devices," *Applied Sciences*, vol. 9, no. 5, p. 848, feb 2019. doi: 10.3390/app9050848

[40] S. Yu, G. Wang, X. Liu, and J. Niu, "Security and Privacy in the Age of the Smart Internet of Things: An Overview from a Networking Perspective," *IEEE Communications Magazine*, vol. 56, no. 9, pp. 14–18, 2018. doi: 10.1109/MCOM.2018.1701204

[41] G. Han, H. Wang, M. Guizani, S. Chan, and W. Zhang, "KCLP: A k-Means Cluster-Based Location Privacy Protection Scheme in WSNs for IoT," *IEEE Wireless Communications*, vol. 25, no. 6, pp. 84–90, 2018. doi: 10.1109/MWC.2017.1800061

[42] OMA LightweightM2M (LWM2M), "OMA (Open Mobile Alliance) Specification," 2017.

[43] C. W. Hung and W. T. Hsu, "Power consumption and calculation requirement analysis of AES for WSN IoT," *Sensors (Switzerland)*, vol. 18, no. 6, p. 1675, may 2018. doi: 10.3390/s18061675

[44] B. Martinez, X. Vilajosana, F. Chraim, I. Vilajosana, and K. S. Pister, "When Scavengers Meet Industrial Wireless," *IEEE Transactions on Industrial Electronics*, vol. 62, no. 5, pp. 2994–3003, 2015. doi: 10.1109/TIE.2014.2362891

[45] X. Vilajosana, Q. Wang, F. Chraim, T. Watteyne, T. Chang, and K. S. Pister, "A realistic energy consumption model for TSCH networks," *IEEE Sensors Journal*, vol. 14, no. 2, pp. 482–489, 2014. doi: 10.1109/JSEN.2013.2285411

[46] E. Kovacs, M. Bauer, J. Kim, J. Yun, F. Le Gall, and M. Zhao, "Standards-Based Worldwide Semantic Interoperability for IoT," *IEEE Communications Magazine*, vol. 54, no. 11, pp. 40–46, 2016. doi: 10.1109/MCOM.2016.1600460CM

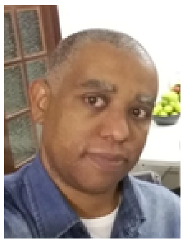
[47] R. A. Light, "Mosquitto: server and client implementation of the MQTT protocol," *The Journal of Open Source Software*, vol. 2, no. 13, p. 265, may 2017. doi: 10.21105/joss.00265

[48] M. Amiri-Zarandi, R. A. Dara, and E. Fraser, "A survey of machine learning-based solutions to protect privacy in the Internet of Things," *Computers and Security*, vol. 96, p. 101921, 2020. doi: 10.1016/j.cose.2020.101921

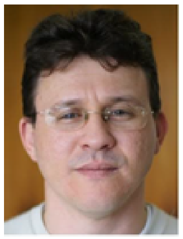
[49] A. Dunkels, F. Osterlind, N. Tsiftes, and Z. He, "Software-based on-line energy estimation for sensor nodes," *Proceedings of the 4th Workshop on Embedded Networked Sensors, EmNets 2007*, pp. 28–32, 2007. doi: 10.1145/1278972.1278979

[50] W. Tiberti, D. Cassioli, A. Di Marco, L. Pomante, and M. Santic, "A Model-Based Approach for Adaptable Middleware Evolution in WSN Platforms," *Journal of Sensor and Actuator Networks*, vol. 10, no. 1, p. 20, mar 2021. doi: 10.3390/jsan10010020

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 PhD. Degree in Computer Engineering from USP in 2016, his Masters 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 Eletric 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 information security area. Currently, Adilson has the position of Dean of Research and Graduate Studies at UNOESTE (University of Western Sao Paulo), and also he is a regular professor of the master's program in Computer Engineering at the FIPT (Institute for Technological Research Foundation – Sao Paulo).

Marcelo Teixeira de Azevedo is a Doctor in Eletric Engineering at the Electronic Engineering Department, EPUSP (Polytechnic School of the University of São Paulo), São Paulo. He is Prof. Dr. at University of Santa Barbara d'Oeste, Control and Automation Engineering Dept. His current research interests include topics such as Internet of Things, Cyber-security, Network Communication, Smart City and Industry 4.0.

Sergio Takeo Kofuji holds a Bachelor's Degree in Physics from the University of São Paulo (1985), a Master's Degree in Electrical Engineering from the University of São Paulo (1988) and a PhD in Electrical Engineering from the University of São Paulo (1995), He is currently Professor Dr. of the Polytechnic School of the University of São Paulo. Currently He has been focusing in Projects related to IoT, 5G communication, Smart Objects, Machine Learning and Artificial Intelligence, applied to Smart Cities and Smart Farms.