Do we want the New Old Internet? Towards Seamless and Protocol-Independent IoT Application Interoperability

Vadim Safronov
University of Cambridge
vadim.safronov@cl.cam.ac.uk

Justas Brazauskas
University of Cambridge
jb2328@cam.ac.uk

Matthew Danish
University of Cambridge
mrd45@cam.ac.uk

Rohit Verma
University of Cambridge
rv355@cam.ac.uk

Ian Lewis
University of Cambridge
ijl20@cam.ac.uk

Richard Mortier
University of Cambridge
richard.mortier@cl.cam.ac.uk

ABSTRACT

IoT is developing rapidly with frequently appearing new wireless standards and applications. However, besides a large number of IoT benefits, its further development is now being slowed down due to the repetition of old Internet development flaws while dealing with IoT heterogeneity. The current misleading trend aims to solve all IoT interoperation problems by inserting IP Addresses into those wireless protocols where the IP stack clearly slows down application performance and drains the battery, e.g. LPWANs such as LoRaWAN and SigFox. This paper tackles IoT heterogeneity from a different perspective: it is the application interoperation which matters the most. The protocols beneath the application layer shall work for smooth upper-layer service provisioning where the IP shall be just one of the many underlying integration options instead of being the essential one. Inspired by previous proposals for a more flexible inter-networking architecture, this paper applies those theoretical concepts in practice by proposing a protocol-independent distributed interoperation model for smooth service provisioning over heterogeneous IoT wireless contexts. The arguments pro the IP-agnostic IoT application interoperation are supported by the model's prototype which showed 1.6-2 times faster MQTT application operation over LoRa and WiFi compared to the legacy IP-based MQTT provisioning over that protocols.

ACM Reference Format:

Vadim Safronov, Justas Brazauskas, Matthew Danish, Rohit Verma, Ian Lewis, and Richard Mortier. 2021. Do we want the New Old Internet? Towards Seamless and Protocol-Independent IoT Application Interoperability. In *The Twentieth ACM Workshop on Hot Topics in Networks (HotNets '21), November 10–12, 2021, Virtual Event, United Kingdom*. ACM, New York, NY, USA, 7 pages. <https://doi.org/10.1145/3484266.3487374>

1 INTRODUCTION

Smart spaces and cities have become a reality thanks to the IoT. We now have computer-controlled office climates, remote control of various smart appliances, asset automation and building management. To provide a wide range of IoT services, smart environments deploy sensors and actuators in order to react to any critical changes in a timely fashion. Diverse functional requirements and hardware constraints mean different IoT devices often use disparate communication protocols to interact with each other. E.g. LoRaWAN/Sigfox provides low-power, long-range and low-bandwidth sensors, while ZigBee/Z-Wave gives lower latency and higher bandwidth at shorter ranges.

Currently most IoT application protocols (e.g. MQTT [35]) assume they run over the commonly used Internet Protocol (IP). That assumption creates interoperation problems, especially for those IoT wireless protocols which are not meant to be layered on top of IP (e.g. LoRaWAN or other LPWANs) as bringing IP stack to them significantly decreases their performance. Therefore, IP-agnostic IoT interoperation, with its lower memory footprint compared to IP-based IoT networks [15], can save scarce hardware resources of constrained devices.

Speaking more generally, the Internet initially has appeared as a compound of scattered local networks (e.g. MILNET, ARPANET [13] etc.) where the interconnection design decisions were taken on the go and thus need to be reconsidered for our current reality. Modern IoT devices with

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

HotNets '21, November 10–12, 2021, Virtual Event, United Kingdom

© 2021 Copyright held by the owner/author(s).

ACM ISBN 978-1-4503-9087-3/21/11.

<https://doi.org/10.1145/3484266.3487374>

different functionalities and wireless technologies cannot be simply embraced by one generic network protocol like it was done in the early years of the Internet. Unlike classic power-plugged computers (which haven't changed so much from the network communication perspective), different IoT devices are designed for different purposes where each device performs a specific list of functions contributing to overall IoT service provisioning in a smart environment.

Inspired by general internetworking architectures [3, 12, 37], this paper argues that further IoT development shall move from current IP-dependent homogeneous interoperation towards a decentralised protocol-neutral heterogeneous model in order to seamlessly provide IoT services and applications over any set of underlying wireless technologies/protocols. It is the application interoperation which matters the most in the heterogeneous IoT reality where the IP protocol is just one of the tools for delivering upper services. Based on the concepts introduced in the Plutarch architecture [12] we propose a protocol-independent distributed model for seamless application interoperability in IoT environments. The aim of this paper is to make the initial move towards the correction of IoT interoperation paradigm on its early development stage until the technology is beyond the point of no return for escaping IP-dependency with its subsequent limitations.

Our IP-neutral IoT interoperation prototype highlights IP-free application interoperation benefits through testing seamless MQTT over WiFi and LoRa: two heterogeneous protocols where the first is high-bandwidth and works with IP while the latter comes without an IP stack and designed for low-power low-bandwidth communications. The prototype was evaluated against the standard LoRaWAN IP-based MQTT provisioning over that contexts showing 1.6-2 times faster application interoperability on the majority of links without power consumption overhead for battery-powered LoRa sensors.

2 RELATED WORK

2.1 Improving IoT interoperation

IP stack implementation in non-IP-based wireless protocols. Ongoing work seeks to improve IoT interoperability by providing an IP stack for non-IP protocols so that all IoT devices can be interconnected through IPv4/v6. SCHC [30] provides generic compression mechanism which can be applied to compress IPv6/UDP headers in low-power wide-area networks (LPWAN). Gimenez et al. [20] introduced an RFC draft with SCHC adaptation for LoRaWAN. The SCHC mechanism has also an open-source Python implementation called OpenSCHC [2].

Another example is ZigBeeIP [19] which is an IP-based version of a standard ZigBee specification [4]. ZigBeeIP is based

on the same IEEE 802.15.4 PHY/MAC standard and integrates with 6LoWPAN [31] in order to provide IP, TCP/UDP and upper application functionality. IETF 6Lo working group [1] is focused on bringing of 6LoWPAN to 6Lo [21], i.e. implementing IPv6 in other IoT wireless low-power standards such as LoRaWAN [25] and SigFox [28].

Porting of IP-dependent application protocols to non-IP contexts. Stanford-Clark et al. presented MQTT-SN [36], a modified MQTT version for non-IP sensor networks which provides integration with the standard MQTT [35]. The mapping between MQTT-SN and MQTT messages is done through an aggregation gateway which receives MQTT-SN packets from all connected clients. To not overload a broker by a large number of MQTT sessions, the aggregation gateway can establish just one general MQTT connection to the broker and decide which MQTT-SN packets will go further to the server from the sensors. Although there are more similar application integration solutions such as zigbee2mqtt [23] or LoRaWAN-CoAP integration [10], they are not aimed at general application interoperation approach for neither of that protocols and work for a limited set of specific intercommunication use cases.

2.2 General internetworking architectures

Crowcroft et al. [12] propose Plutarch to solve the network heterogeneity problem by an explicit mapping of names, addresses and routing rules on the border of adjacent networks. The proposed concept of *contexts* and *interstitial functions* allows heterogeneous communications through the definition of query models and mapping rules between any two distinct contexts, i.e. networks with different addressing schemes, packet formats or protocol used. Inspired by the Plutarch concept and by the role-based communication architecture suggested by Braden et al. [8], Dunkels et al. propose an adaptive wireless communication infrastructure called Chameleon [14]. Chameleon architecture is protocol-neutral and provides a group of communication primitives and abstractions for seamless interconnection of heterogeneous wireless sensor networks.

Based on the ideas of layered and separated network naming [7, 11, 22] Ahglren et al. [3] propose a node identity (NID) internetworking architecture where NID is decoupled from a network protocol and can be used to bridge across heterogeneous network contexts. Unlike completely decentralised approach proposed by Plutarch, Ahglren et al. claims that a core network is still needed (either based on IP or any other addressing protocol) in order to manage the distribution of NIDs across different contexts and escape potential storing of a large number of NIDs on each communicating node.

The ideas proposed by previously mentioned internetworking architectures were covered in later works on ILNP

by Atkinson et.al [5, 6] and then by Yanagida et al. [37]. ILNP employs "Locator" and "Identifier" namespaces in order to separate a node unique identifier from transport and network protocols. The discovery and binding of NIDs is also done by ILNP. The concept is IP-neutral and can potentially be employed in other wireless sensor networks.

3 IOT INTEROPERATION MODEL

Of the internetworking architectures reviewed in Section 2.2, Plutarch has one of the highest levels of abstraction and applies to a wide range of intercommunication scenarios. Its flexible, IP-neutral, decentralised approach makes Plutarch a suitable basis for making distributed seamless application interoperation over heterogeneous IoT networks.

3.1 Main concepts

Figure 1 represents our proposed distributed and protocol-neutral interoperation architecture. The model is based on the Plutarch abstractions of *contexts* and *interstitial functions* (IF) adjusted for IoT scenarios, and on the *node identity* (NID) concept proposed by Ahlgren et al. [3]:

- **Context:** an area of homogeneity in message formats, transport protocols and naming services. For a smart environment, a context is a group of IoT devices which use same network protocols, packet formats and addressing schemes. E.g. a group of smart bulbs operating through the same WiFi network forms a single context. Similarly, several ZigBee smartplugs connected into a mesh network are considered as the same context.
- **Interstitial Function:** a software function which is implemented within a network gateway and is responsible for correct traffic transmission between different contexts. Besides being a relay which maps messages from one protocol to another, an interstitial function is also in charge of enforcing control policies on a context border as well as within a context: e.g. ensures a certain level of QoS, filters sensitive or redundant data and automates context workflows.
- **Node Identity:** a protocol-independent device unique identifier which is decoupled from the address provided by a context and its protocols. For example a NID can be derived from a public key of a public/private key pair.

IoT sensors and actuators of the same context are managed by a single *smart gateway* (SGW) with an IF which performs the following functions:

- **interconnection of adjacent IoT device groups:** provision of local IoT interoperation within a single device group, communication and traffic control between adjacent contexts.

Figure 1: IoT interoperation model.

- **enforcement of context policies** defined by a user or delegated by a general control platform such as a Building Management System (BMS). Some lower-layer context policies can also be in place aiming for correct traffic transmission between the contexts and making sure that Quality of Service (QoS) remains at the same level (e.g. while switching from high-bandwidth contexts to low-power networks with much lower transmission rate and higher latency).
- **IoT device local or external discovery** through requesting other SGWs. To provide robust end-to-end (E2E) connectivity, an SGW is also responsible for device discovery in other contexts through interaction with adjacent SGWs. Each SGW has a dynamic address table that maps a device NID to its appropriate in-context address (e.g. its L2 or L3 address), or to the address of the next SGW on the path to the destination device.

3.2 Address resolution and routing

Interoperation between non-adjacent IoT contexts separated by several heterogeneous networks require additional address resolution and routing functions discussed in [3, 12, 37]. While the model concepts allow for support of routing functionality without sticking to IP addresses, it is crucial to mention, that in a large number of IoT applications, such as smart space automation and management, address resolution can be less tricky in fact. In smart environments the majority of IoT devices, such as sensors and actuators, stays at the same place (e.g. door open/close or temperature sensors), so the mappings can be stored statically on SGWs without requiring dynamic address resolution most of the time. Implementation and functional complexity of IP-free switching between different contexts on an SGW can be negligible, as most such IoT scenarios realistically require a few protocol types for seamless service provisioning. E.g. a smart building can employ WiFi for power-hungry and/or high-bandwidth IoT devices such as surveillance cameras and power-connected household appliances (fridges, microwaves, etc.), while using

ZigBee/Z-Wave and LoRaWAN/SigFox standards for indoor and outdoor battery-powered low-bandwidth sensors/actuators respectively.

3.3 Smart policies, QoS guarantees and dealing with redundant data

An IF is also responsible for enforcing administrative and user-defined policies within a controlled context. An illustrative example of such IF functionality for dealing with sensitive and redundant IoT data is given in our Smart Coffee Pot project [24]. The IoT appliance consists of a weight sensor for the coffee pot and two smart plugs that monitor power consumption from a bean grinder and a coffee brewer. All devices are connected to the same local WiFi network shared by a Raspberry Pi (RPI) model 3B+. Instead of sending raw and potentially sensitive sensor readings from a local WiFi to a remote server for data processing, the RPI side implements an IF with a local event-recognition mechanism that looks for correlations between the pot's weight and the power consumption of both the grinder and the coffee machine. Thus, nobody's sensitive nor any redundant raw sensor data is sent outside of the local WiFi. Only discrete events such as 'coffee has been made' or 'the pot is empty' are passed along to the remote web platform displaying the coffee pot's status to the public. Although the described IF implementation is WiFi-to-WiFi, the core of its smart logic can be ported to other inter-context scenarios, e.g. WiFi-LoRaWAN or ZigBee-LoRaWAN where filtering redundant traffic is crucial for keeping QoS at the same level as saves scarce resources of low-power low-bandwidth networks.

4 PROTOTYPING AND TESTING

4.1 IP-agnostic MQTT interoperation

Figure 2 depicts the main components of our IP-neutral application interoperation model's prototype, which demonstrates seamless MQTT over WiFi and LoRa wireless standards. The main prototype aims are: (1) testing the effectiveness of IP-agnostic IoT application interoperation against the legacy IP-dependent one, (2) showing the model's potential for dealing with high level of IoT heterogeneity in an IP-free manner: WiFi provides a high-bandwidth power-hungry and short-range IP network, while LoRa has no IP support and is designed for a wide propagation range at low bandwidth rates (see its official documentation [25]). Besides E2E MQTT provision over LoRa and WiFi, other main artefacts of the deployment include a LoRa MQTT client and a LoRa-WiFi IF both written in micropython [29]. The components of our interoperation model are briefly described below.

LoRa context. Both Sensor A and Sensor C run MQTT clients on top of a Pycom LoPy4 controller [33] connected to a Pysense shield [34]. Sensor A imitates the behaviour

Figure 2: IP-agnostic MQTT interoperation prototype.

of a Class A LoRaWAN device (battery-powered sensor, e.g. a temperature or a humidity sensor) regularly publishing sensor readings to a given MQTT topic. Sensor A is able to receive incoming messages only for a short bit of time just after publishing an uplink MQTT message while sleeping all other time in order to save its 500 mAh Li-Pol battery (connected to a Pysense shield). Sensor C has an external power supply and imitates the behaviour of a Class C LoRaWAN device (e.g. a smart plug or a light switch) always listening for incoming MQTT messages and regularly publishing readings to a given MQTT topic. The LoRa IoT Controller is a power-plugged LoPy4 device which is subscribed to MQTT topics of sensors A, B and C. The controller manages both sensors A and C by publishing LoRa MQTT commands to their control topics.

Smart Gateway (SGW). The gateway consists of a LoPy4 board with a power-plugged Pysense shield and implements an IF with WiFi and LoRa MQTT clients written in micropython. The SGW performs all necessary WiFi-LoRa mapping and control functions for seamless MQTT provision over the given contexts. The gateway functions also include context policies enforcement such as MQTT topic filtering and inter-context QoS control.

WiFi context. All WiFi devices run power-plugged Raspberry Pis (RPis) with the Raspberry Pi OS [18] on top. Wireless connection is provided by the WiFi Access Point which runs hostapd [26], a mosquitto MQTT broker [16] and a local NTP server [27]. Sensor B employs a paho-mqtt client [17] and imitates a WiFi MQTT sensor. The WiFi IoT Controller also employs the same paho-mqtt lib and is subscribed to all sensor A, B and C topics controlling only the local Sensor B by publishing MQTT commands to its control topic.

LoRaWAN MQTT broadcasting. Using a standard MQTT TCP-based broker in LoRa would result in significant bandwidth consumption greatly decreasing the application performance and, thus, resulting in disruptions of message delivery to LoRa MQTT clients. Thus, a broker-free MQTT concept

(similar to MQTT/UDP approach [38]) was employed in the LoRa context. Every LoRa MQTT subscriber regularly listens to broadcast messages from other LoRa MQTT clients. If a broadcast message (an MQTT topic with some payload) is received, the subscriber accepts the message in case of a topic match. All messages are encrypted by AES-128 securing message payload from being overheard by non-authorised LoRa MQTT clients.

4.2 Testing the prototype

4.2.1 Evaluation use case. To model a general interoperation use case, all sensors were programmed to generate random values from 0 to 255 on their given MQTT topics. The SGW with IF on the WiFi-LoRa border makes MQTT messages visible for both IoT controllers subscribed to the topics of sensors A, B and C. The controllers can manage only those sensors which lie in their own context (e.g. the LoRa IoT controller can control the LoRa sensors only). Both controllers publish control MQTT messages to managed sensors when a high reading value comes from another local or external sensor (modelling sense-actuate automation activities in a smart space).

4.2.2 The baseline. The current well-known practice of managing WiFi and LoRa devices through MQTT assumes the employment of a full LoRaWAN stack [25]. Within the LoRaWAN paradigm sensors are not able to exchange data directly. Instead, all communications happen through a LoRaWAN Gateway (LoRaWAN GW) which makes sensors visible to an IP network by encapsulating LoRa messages into IP packets and passing them further to the LoRaWAN Server which speaks to a central MQTT broker. To implement that evaluation scenario all MQTT-specific code was removed from the LoRa sensors so they could behave like standard LoRaWAN Class A/C devices managed by the WiFi IoT Controller along with the Sensor B. To enable that legacy IP-dependent MQTT interoperation (Figure 3) the SGW was reprogrammed into a simple LoRaWAN GW [32] which transmits LoRa sensor data to the local ChirpStack LoRaWAN Server [9] with installed mosquitto MQTT broker [16].

4.2.3 E2E latency measurements. Figures 4 and 5 show the empirical CDFs (ECDFs) with distributions of E2E latencies of command reaction measured on Sensor B and C for both MQTT interoperation variants. The latency was calculated as the difference between the timestamps when a sensor reading was generated and when a control command came to the target sensor as a reaction to that reading. In order to minimise time offsets between the nodes and E2E latency errors, all nodes were synced with the local NTP server. The E2E latency was measured around 1000 times for both decentralised and standard approaches. The command reaction

Figure 3: IP-dependent MQTT interoperation over LoRaWAN.

Figure 4: ECDFs with E2E latencies from sensors A and C measured on Sensor B in both evaluation scenarios.

time was deliberately not measured on Sensor A due to its power-saving nature which assumes much less incoming control traffic compared to more power-hungry sensors B and C (the command reaction to the sensor A readings was still measured on both of them).

The latency differences between the IP-agnostic and IP-dependent MQTT interoperation approaches are seen quite clearly, notwithstanding any minor shifts that might arise from NTP offset adjustments. The IP-neutral approach has lower latency on most of the communication links, especially seen for the sensor's C command reaction to the readings from the sensors A and B (Figure 5), which is 1.7-2 times

Figure 5: ECDFs with E2E latencies from sensors A and B measured on Sensor C in both evaluation scenarios.

faster compared to its IP-based MQTT variant. For the WiFi Sensor B (Figure 4) there are less latency differences which are around the same for A-B interoperation, however the IP-agnostic approach still has 1.6 faster reaction to the readings from the Sensor C.

High latency values and slow E2E feedback seen in the IP-based MQTT approach are caused by a higher number of intermediate hops between the sensors. All LoRa traffic has to pass through the LoRaWAN Server, and thus the nodes cannot talk directly to each other like they do in the IP-agnostic MQTT variant. It can be argued that the problem of IoT application interoperability lies not only in IP, but also in the MQTT implementation choices of upper-layer protocols, e.g. reliance on TCP instead of faster UDP. However, the same version of TCP-based MQTT broker was used on the WiFi side in both IP-agnostic and IP-dependent evaluation scenarios. Thus, the observed performance improvements of the IP-agnostic approach are mainly caused by the model’s architectural features described in Section 4.1.

4.2.4 Power consumption comparison. Figure 6 shows the Sensor’s A battery discharge time in both MQTT approaches. The battery voltage was measured every time when the sensor was waking up and transmitting data. To decrease the experiment duration (LoRaWAN battery can serve up to 10 years assuming that each sensor generates just a few readings per day) the sensor was programmed to produce a reading every 5 seconds within a 45-minute interval.

Figure 6: Sensor A battery discharge speed in both evaluation scenarios.

The results clearly show around the same battery discharge speed for both MQTT variants. Thus, no power-consumption overhead is brought by running an MQTT client on top of a LoRa sensor.

5 CONCLUSIONS

We presented the design and implementation of the proposed protocol-independent interoperation model for running IoT applications seamlessly. The prototype evaluation shows promising results highlighting the potential to improve IoT interoperation by moving from legacy IP-dependent communication paradigms while focusing on protocol-neutral application provision over heterogeneous IoT contexts. The benefits of escaping from the IP or any other underlying bottleneck protocols look more clear now as provide a more flexible organisation of IoT applications with decreased interoperation latency and no power-consumption overheads for battery-powered devices. This paper is a starting point and gives direction towards the development of IP-agnostic application-focused IoT architectures in order to prevent the IPv4/v6 becoming the new (old) normal with its all subsequent flaws existing since the creation of the Internet.

To better demonstrate the model’s efficacy, the next step includes generalisation to a larger set of wireless IoT protocols and services. The work also opens avenues towards distributed control of smart spaces where management functions can be delegated from a central platform (e.g. a BMS) to SGWs providing more user-oriented control over IoT applications and data in a secure, resilient and privacy-preserving manner.

ACKNOWLEDGMENTS

This research forms part of Centre for Digital Built Britain’s work within the Construction Innovation Hub. The funding was provided through the Government’s modern industrial strategy by Innovate UK, part of UK Research and Innovation.

REFERENCES

- [1] IETF 6Lo Working Group. 2021. IPv6 over Networks of Resource-constrained Nodes (6lo). (2021). <https://datatracker.ietf.org/wg/6lo/about/>
- [2] C. Adjih, S. A. Romero, D. Barthel, A. Bernard, D. O. D. Beltran, O. Gimenez, A. Kandasamy, L. Toutain, and S. Sakane et al. 2021. OpenSCHC. (2021). <https://github.com/openschc/openschc>
- [3] B. Ahlgren, J. Arkko, L. Eggert, and J. Rajahalme. 2006. A Node Identity Internetworking Architecture. In *Proc. IEEE INFOCOM*. 1–6. <https://doi.org/10.1109/INFOCOM.2006.51>
- [4] ZigBee Alliance. 2015. *ZigBee Document 05-3474-21*. Technical Report. <https://zigbeealliance.org/wp-content/uploads/2019/11/docs-05-3474-21-0csg-zigbee-specification.pdf>
- [5] R.J. Atkinson and S.N. Bhatti. 2012. *Identifier-Locator Network Protocol (ILNP) Architectural Description*. RFC 6740. RFC Editor. <https://www.rfc-editor.org/info/rfc6740>
- [6] Randall Atkinson, Saleem Bhatti, and Stephen Hailes. 2009. ILNP: Mobility, multi-homing, localised addressing and security through naming. *Telecommunication Systems* 42 (12 2009), 273–291. <https://doi.org/10.1007/s11235-009-9186-5>
- [7] H. Balakrishnan, K. Lakshminarayanan, S. Ratnasamy, S. Shenker, I. Stoica, and M. Walfish. 2004. A Layered Naming Architecture for the Internet. In *Proc. ACM SIGCOMM*. 343–352. <https://doi.org/10.1145/1015467.1015505>
- [8] Robert Braden, Ted Faber, and Mark Handley. 2003. From Protocol Stack to Protocol Heap: Role-Based Architecture. *SIGCOMM Comput. Commun. Rev.* 33, 1 (Jan. 2003), 17–22. <https://doi.org/10.1145/774763.774765>
- [9] ChirpStack. 2021. ChirpStack, open-source LoRaWAN Network Server stack. (2021). <https://www.chirpstack.io/>
- [10] Antonio Cilfone, Luca Davoli, and Gianluigi Ferrari. 2019. Virtualizing LoRaWAN Nodes: a CoAP-based Approach. In *2019 International Symposium on Advanced Electrical and Communication Technologies (ISAECT)*. 1–6. <https://doi.org/10.1109/ISAECT47714.2019.9069691>
- [11] David Clark, Robert Braden, Aaron Falk, and Venkata Pingali. 2003. FARA: Reorganizing the Addressing Architecture. *SIGCOMM Comput. Commun. Rev.* 33, 4 (Aug. 2003), 313–321. <https://doi.org/10.1145/972426.944770>
- [12] J. Crowcroft, S. Hand, R. Mortier, T. Roscoe, and A. Warfield. 2003. Plutarch: An Argument for Network Pluralism. In *Proc. ACM SIGCOMM FDNA '03*. 258–266. <https://doi.org/10.1145/944759.944763>
- [13] Stephen C. Dennett, E. Feinler, and Francine Perillo. 1985. ARPANET Information Brochure. <https://apps.dtic.mil/sti/pdfs/ADA164353.pdf>
- [14] A. Dunkels, F. Österlind, and Z. He. 2007. An Adaptive Communication Architecture for Wireless Sensor Networks. In *Proc. ACM SenSys*. 335–349. <https://doi.org/10.1145/1322263.1322295>
- [15] Orestis Evangelatos, Kasun Samarasinghe, and Jose Rolim. 2012. Evaluating design approaches for smart building systems. In *2012 IEEE 9th International Conference on Mobile Ad-Hoc and Sensor Systems (MASS 2012)*, Vol. Supplement. 1–7. <https://doi.org/10.1109/MASS.2012.6708524>
- [16] Eclipse Foundation. 2021. Eclipse Mosquitto™. An open source MQTT broker. (2021). <https://mosquitto.org/>
- [17] Eclipse Foundation. 2021. Eclipse Paho™ MQTT Python Client. (2021). <https://github.com/eclipse/paho.mqtt.python>
- [18] Raspberry Pi Foundation. 2021. Raspberry Pi OS. (2021). <https://www.raspberrypi.org/software/>
- [19] M. Franceschinis, C. Pastrone, M. A. Spirito, and C. Borean. 2013. On the performance of ZigBee Pro and ZigBee IP in IEEE 802.15.4 networks. In *Proc. IEEE WiMob*. 83–88.
- [20] O. Gimenez and I. Petrov. 2020. *Static Context Header Compression (SCHC) over LoRaWAN*. RFC-internet-draft. RFC Editor. <https://www.rfc-editor.org/info/rfc9011>
- [21] C. Gomez, J. Paradells, C. Bormann, and J. Crowcroft. 2017. From 6LoWPAN to 6Lo: Expanding the Universe of IPv6-Supported Technologies for the Internet of Things. *IEEE Communications Magazine* 55 (12 2017). <https://doi.org/10.1109/MCOM.2017.1600534>
- [22] A. Jonsson and B. Ahlgren. 2003. The Split Naming/Forwarding Network Architecture. In *Proc. Swedish National Computer Networking Workshop (SNCNW)*.
- [23] Koen Kanter. 2021. Zigbee2MQTT. (2021). <https://github.com/Koenkk/zigbee2mqtt>
- [24] Ian Lewis. 2020. Cambridge Coffee Pot project. (2020). https://github.com/ijl20/cambridge_coffee_pot
- [25] LoRa Alliance Technical Committee. 2017. LoRaWAN™ 1.1 Specification. https://lora-alliance.org/wp-content/uploads/2020/11/lorawan_specification_v1.1.pdf. (2017).
- [26] Debian Manpages. 2021. Hostapd manual page. (2021). <https://manpages.debian.org/bullseye/hostapd/hostapd.8.en.html>
- [27] Debian Manpages. 2021. Ntpd manual page. (2021). <https://manpages.debian.org/bullseye/ntpsec/ntpsec.8.en.html>
- [28] Kais Mekki, Eddy Bajic, Frederic Chaxel, and Fernand Meyer. 2018. Overview of Cellular LPWAN Technologies for IoT Deployment: Sigfox, LoRaWAN, and NB-IoT. In *2018 IEEE International Conference on Pervasive Computing and Communications Workshops (PerCom Workshops)*. 197–202. <https://doi.org/10.1109/PERCOMW.2018.8480255>
- [29] MicroPython. 2021. Project GitHub page. (2021). <https://github.com/micropython/micropython>
- [30] A. Minaburo, L. Toutain, C. Gomez, D. Barthel, and J.C. Zuniga. 2020. SCHC: Generic Framework for Static Context Header Compression and Fragmentation. RFC 8724. RFC Editor. <https://www.rfc-editor.org/info/rfc8724>
- [31] Geoff Mulligan. 2007. The 6LoWPAN Architecture. In *Proceedings of the 4th Workshop on Embedded Networked Sensors (EmNets '07)*. Association for Computing Machinery, New York, NY, USA, 78–82. <https://doi.org/10.1145/1278972.1278992>
- [32] Pycom. 2019. LoRaWAN nano-gateway source code. (2019). <https://github.com/pycom/pycom-libraries/blob/master/examples/lorawan-nano-gateway/nanogateway.py>
- [33] Pycom. 2020. LoPy4 Datasheet. (2020). https://docs.pycom.io/gitbook/assets/specsheets/Pycom_002_Specsheets_LoPy4_v2.pdf
- [34] Pycom. 2020. Pysense Datasheet. (2020). <https://docs.pycom.io/gitbook/assets/pysense-specsheet.pdf>
- [35] OASIS Standard. 2019. MQTT Version 5.0. (2019). <https://docs.oasis-open.org/mqtt/mqtt/v5.0/os/mqtt-v5.0-os.pdf>
- [36] A. Stanford-Clark and H. L. Truong. 2013. MQTT For Sensor Networks (MQTT-SN) Protocol Specification. (2013). https://www.oasis-open.org/committees/download.php/66091/MQTT-SN_spec_v1.2.pdf
- [37] R. Yanagida and S. Bhatti. 2019. Seamless Internet Connectivity for Ubiquitous Communication. In *Proc. ACM UbiComp/ISWC*. 1022–1033. <https://doi.org/10.1145/3341162.3349315>
- [38] Dmitry Zavalishin. 2019. MQTT/UDP. (2019). https://github.com/dzavalishin/mqtt_udp/wiki