6TiSCH: Industrial Performance for IPv6 Internet-of-Things Networks

This paper focuses on the standardization activity of the IETF, to integrate IPv6 on industrial wireless networks, as a decisive step toward the implementation of IIoT systems.

By Xavier Vilajosana, *Senior Member IEEE*, Thomas Watteyne, *Senior Member IEEE*, Mališa Vučinić, *Member IEEE*, Tengfei Chang, and Kristofer S. J. Pister

ABSTRACT | The convergence of operational and information technologies in the industry requires a new generation of IP-compliant communication protocols that can meet the industrial performance requirements while facilitating the integration with novel web-based supervisory control and data acquisition (SCADA) systems. For more than a decade, the industry has relied on time-slotted channel hopping (TSCH) communication technology to meet these performance requirements through standards such as WirelessHART and ISA100.11a. TSCH-based networks have proven to yield over 99.999% end-to-end reliability, supporting flow isolation and QoS management while ensuring over a decade of battery lifetime. However, these technologies were designed to address the factory use cases of a decade ago, not considering IP compliance or standardized network management and resource orchestration as a must. The Internet Engineering Task Force (IETF) and the 6TiSCH working group (WG) have been actively working on this challenge by designing protocols to bridge the performance of industrial solutions with IP-compliant networks. The effort has resulted in 6TiSCH, a set of specifications that define the IPv6 control plane to manage and orchestrate a TSCH network. 6TiSCH provides the missing elements for zero-configuration TSCH network bootstrap, efficient network access authentication, and distributed and modular scheduling mechanisms. As a cross-layer effort, 6TiSCH leverages and integrates other IETF specifications and the WG has also driven the definition of novel specifications in other IETF WGs. An ultimate goal of this effort is the definition of a fully functional architecture where a combination of IETF protocols enables the envisioned convergence on top of the IEEE industrial standard. This paper introduces the work done by the 6TiSCH WG at IETF, evaluates the performance of the reference implementation, and discusses the 6TiSCH software ecosystem.

KEYWORDS | 6TiSCH; IEEE 802.15.4; IETF; industrial networks; IPv6; time-slotted channel hopping (TSCH)

I. INTRODUCTION

The term "automation," inspired by the word "automatic," was not in use until 1947, when general motors created an automation department to handle the rapid adoption of feedback controllers invented roughly a decade before. The goal was to replace and improve repetitive tasks, done manually at the time. The 20th century saw the

Manuscript received November 6, 2018; revised February 9, 2019 and March 12, 2019; accepted March 15, 2019. Date of publication April 11, 2019; date of current version May 28, 2019. This work was supported in part by the European Commission through the H2020 F-Interop Project, in part by the Spanish Ministry of Economy, and in part by the European Regional Development Fund (ERDF) through the SINERGIA Project under Grant TEC2015-71303-R. *(Corresponding author: Xavier Vilajosana.)*

X. Vilajosana is with Worldsensing S. L., 08034 Barcelona, Spain, and also with the Wireless Networks Laboratory, Universitat Oberta de Catalunya, Catalonia, 08018 Barcelona, Spain (e-mail: xvilajosana@uoc.edu).

T. Watteyne is with Analog Devices, Limerick, V94 T9PX Ireland, and also with the EVA Team, INRIA, 75012 Paris, France.

M. Vučinić and **T. Chang** are with the EVA Team, INRIA, 75012 Paris, France.

K. S. J. Pister is with the Department of Electrical Engineering and Computer Science (EECS), University of California at Berkeley, Berkeley, CA 94720-1770, USA.

Digital Object Identifier 10.1109/JPROC.2019.2906404

0018-9219 © 2019 IEEE. Personal use is permitted, but republication/redistribution requires IEEE permission.
See http://www.ieee.org/publications_standards/publications/rights/index.html for more information.

transformation of industrial automation into a combination of techniques, spanning mechanical, pneumatic, hydraulic, electrical, electronic, and computational expertise. Since then, the industry has relied on the use of control systems and information technologies for process and machinery management. Large process automation—typically, repetitive and high-precision work in large factories—has been taken over by industrial robots. Today, all modern factories automate large machinery and vehicles, but the idea of a fully automated industry, coined in the mid-20th century, remains a vision.

Strategic documents promote [1] a further development of technologies to reduce operational costs and improve efficiency through the digitization of the industrial processes. There is a clear need to retrofit and augment sensing and actuation capabilities of machines through the Internet-of-Things (IoT) devices and advanced data analytics. The machinery has traditionally been equipped with field buses—more recently with different flavors of Ethernet—to wire and internetwork the subsystems. Going wireless is indispensable not only to remove the significant installation costs due to wiring [2] but also to tackle the use cases involving rotational devices and mobility. Industrial users are not ready to give up on reliability provided by wired solutions in order to adopt wireless [3].

Yet, using wireless in an industrial context is challenging due to the harsh environmental conditions, where interference and fading cause any wireless connection to be inherently unreliable. Deploying devices whose batteries would need to be replaced often is inconvenient from the practical aspect, as well. To address these challenges, a set of industrial communication products and standards [4]–[9] was developed, all relying on the concept of time-slotted channel hopping (TSCH) [10]–[12]. TSCH was later adopted by the IEEE as a medium access control (MAC) technique through the IEEE 802.15.4e standard amendment and integrated in the 2015 release of the IEEE 802.15.4 standard.

In harsh industrial conditions, TSCH was demonstrated to provide wirelike reliability, greater than 99.999%, and ultralow power consumption, less than 50 μA on average [13]–[15]. With TSCH at its core, WirelessHART [4] standard introduced wireless to the field. The design of WirelessHART facilitated its adoption through backwards compatibility with the legacy (wired) highway addressable remote transducer (HART) protocol,[1] widely adopted at the time. Today, WirelessHART is used worldwide in industrial process automation, adding wireless connectivity to meters and sensors.

The traditional supervisory control and data acquisition (SCADA) systems are evolving toward cloud-based solutions [16]. WirelessHART is limited in that sense, as it forces application-level gateways, requiring application-specific software to relay the data to a cloud-based SCADA backend. With that, the need for a standardized integration with the Internet reached the industrial automation sector. The Internet Engineering Task Force (IETF) has promptly reacted: different standardization groups are designing solutions that enable each sensor and actuator to obtain an IPv6 address and seamlessly integrate with a cloud-based SCADA (e.g., 6TiSCH, 6LO, ROLL, CoRE, CBOR, among others).

The "IPv6 over the TSCH mode of IEEE 802.15.4e (6TiSCH)" standardization group at the IETF has been developing the management plane specifications to enable the underlying IEEE 802.15.4 network to bootstrap and be managed. This has resulted in a set of specifications that we refer to as 6TiSCH. Industrial IoT applications are the main target of the 6TiSCH design objectives [17].

Fig. 1 illustrates the architecture envisioned by the 6TiSCH working group (WG). 6TiSCH builds on IEEE 802.15.4 compliant hardware and its TSCH MAC layer. The IEEE 802.15.4 standard expects an "upper layer" to perform several critical management tasks that are needed for the network operation. The 6TiSCH WG specifies these solutions: interoperable and zero-configuration network bootstrap, network access authentication and parameter distribution, and the management of the wireless medium through scheduling. The WG, in its architecture [18], also profiles the rest of the protocol stack, leveraging other IoT standards produced by the IETF, such as for routing, compression of protocol headers, and application-layer transport and security.

This paper gives an overview of the 6TiSCH specifications and the work at the IETF that has been influenced by the 6TiSCH WG activities. This paper is organized as

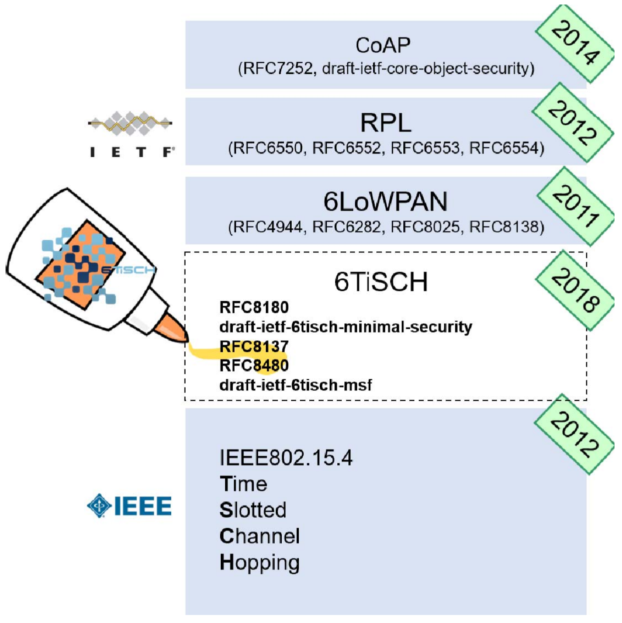

CoAP
(RFC7252, draft-ietf-core-object-security)
2014
RPL
(RFC6550, RFC6552, RFC6553, RFC6554)
2012
IETF
6LoWPAN
(RFC4944, RFC6282, RFC8025, RFC8138)
2011
6TiSCH
RFC8180
draft-ietf-6tisch-minimal-security
RFC8137
RFC8480
draft-ietf-6tisch-msf
2018
6TiSCH
IEEE
IEEE802.15.4
Time
Slotted
Channel
Hopping
2012

Fig. 1. ***6TiSCH WG has produced a set of IP-compliant specifications that manage the underlying IEEE 802.15.4 TSCH MAC layer and enable the integration with other IETF solutions targeting IoT applications.***

[1]See https://www.fieldcommgroup.org/

follows. Section II discusses the 6TiSCH standardization process and related activities. Section III gives the background on the TSCH mode of the IEEE 802.15.4 standard. Section IV introduces the 6TiSCH bootstrap process. Section V focuses on 6TiSCH wireless medium management through scheduling. Section VI describes the lightweight security approach developed to minimize communication and implementation overhead. Section VII discusses how 6TiSCH is built with extensibility in mind. Section VIII lists the key performance indicators of the reference implementation. Section IX gives an overview of the ecosystem of tools developed by the 6TiSCH community and standardization contributors. Finally, Section X concludes this paper.

II. OPEN STANDARD

6TiSCH has been developed by the IETF, the standards body[2] behind most of the technical solutions used in the today's Internet. The IETF adopts an open standardization approach: participation to the standardization activities is open to all, contributions are judged on their technical merit only, and the resulting standards are available at no charge. Open standards are fundamental to ensure a widespread dissemination, transfer, and adoption of the technology [19]. They also facilitate knowledge sharing and collaboration among vendors, researchers, and end users.

Contrary to this, the industrial automation protocols have traditionally been developed by industrial consortia. The access to these standards and consortia often incurs significant charges, which may prevent new players entering the market. Motivated by the need to provide IP-compliant industrial communication technologies, the 6TiSCH WG was formed in the IETF. 6TiSCH WG proposed and promoted a set of specifications to fill in a technical gap: fully operational and compliant IP network over the TSCH mode of IEEE 802.15.4.

6TiSCH builds on IPv6 by design: industrial networks using 6TiSCH seamlessly integrate into the Internet architecture, without the need to bridge or handle protocol translation at the application layer. Therefore, 6TiSCH fully enables the vision of a "cloudified" industry, where sensor and actuator devices connect to cloud-based SCADA systems. Being IP-enabled not only facilitates network management through common IP-based tooling but also ensures proper future and backward compatibility of the technology, considering that machinery in the industry is built to last.

Yet, reaching that point has required not only the development of the management plane features to operate the underlying IEEE 802.15.4 TSCH network but also the joint work with the IETF 6LO and ROLL WGs to develop improved mechanisms for header compression. 6TiSCH security specifications leverage the effort in the IETF CoRE WG, partly driven by the 6TiSCH requirements. In addition, a close liaison is kept with the IEEE on future releases of the IEEE 802.15.4 standard.

[2]See https://www.ietf.org/

III. TIME-SLOTTED CHANNEL HOPPING

The TSCH mode was first standardized by IEEE in the IEEE 802.15.4e amendment. It was integrated with the IEEE 802.15.4 standard in its 2015 release. TSCH is designed for reliability and deterministic access while facilitating long radio sleep intervals that ensure low-power operation. It is essentially a combination of time-division and frequency-division multiple access (TDMA/FDMA).

TSCH splits time into timeslots. For a node to communicate in a timeslot, it needs to be tightly synchronized with the network [20], [21]. Multiple timeslots are grouped into a slotframe structure that repeats over time. A timeslot is long enough (typically 10 ms) for a node in the network to send a maximum sized 127 B frame to its radio neighbor and for that neighbor to send back a link-layer acknowledgment. The schedule defines to a node how each timeslot within a slotframe is used: transmit and to whom, receive, or sleep. The schedule orchestrates all communication in the network.

The IEEE 802.15.4 specification defines how the schedule is executed but it does not define how it is built or signaled between network entities. WirelessHART and ISA100.11a build the schedule in a centralized manner, where the central entity communicates it to each node in the network using application layer commands. 6TiSCH specifications build the schedule in a fully distributed manner, integrating the signaling into the management plane. Each node in the network monitors the resources needed to sustain its application requirements and updates the schedule through direct communication with its radio neighbors. The component in the 6TiSCH architecture that is in charge of dynamically adapting the schedule is called a scheduling function (SF).

The schedule can be represented as an $m \times n$ matrix, where m is the length of the slotframe in timeslots, and n is the number of available channels to hop. An element in the schedule matrix is called a cell, which is uniquely identified by the `(timeslot offset, channel offset)` pair. Timeslot offset serves as an index that maps the cell to the relative position of the corresponding timeslot in the current slotframe. Channel offset is used by the node to calculate the frequency it should tune its radio transceiver to. The channel offset is mapped to the transmission/reception physical channel by means of

$$f = (\mathrm{ASN} + \upsilon) \mod \eta \tag{1}$$

where υ is the channel offset of that cell and η is the total number of available channels to hop. ASN is the absolute sequence number, a global timeslot counter shared among all nodes in the network, and f is the calculated physical channel. Since ASN strictly increases, (1) loops through all available physical channels in η subsequent transmissions, resulting in "channel hopping." The resulting frequency diversity was demonstrated to combat multipath fading and interference, yielding wirelike reliability [11].

IV. ZERO-CONFIGURATION BOOTSTRAP

The IEEE 802.15.4 standard includes a large set of configuration options. Two nodes implementing the standard will not be interoperable if their implementations do not agree on certain parameters beforehand. For instance, what slot and channel offset should a recently booted node use to join a network? How long should a timeslot be? How do nodes efficiently maintain synchronization without creating synchronization loops in the network?

The 6TiSCH WG addresses such questions by relying on a "minimal profile" standardized in the Request for Comments (RFC)8180 [22]. To facilitate interoperable network formation, this profile defines a common schedule used for bootstrap and the control plane traffic. This minimal schedule consists of a single shared cell, used both for transmissions and receptions in a slotted Aloha manner. The cell carries the network advertisements announcing the common configuration, network access authentication, dynamic parameter distribution, and network formation traffic. Strategies to control the traffic on this cell are discussed by Vučinić *et al.* [23].

To maintain the synchronization, the IEEE 802.15.4 standard defines that each node has its "time-source neighbor" that acts as its time reference. The nodes' clocks are then realigned through periodic packet exchanges. How such a time source neighbor is selected is left out of the scope of the IEEE 802.15.4 standard. A challenge for the network is to build a loopless structure in dynamic wireless conditions, and so avoid clusters of nodes getting de-synchronized from the rest of the network. The 6TiSCH WG solves this problem by matching the routing topology built by the routing protocol for low-power and lossy networks (RPL) [24] to the timing topology. A routing parent of a node is also its time source neighbor, with one node in the network acting as the root.

Another challenge appears during the network formation process when a new node, referred to as pledge, attempts to join the network. A pledge may hear advertisements from multiple radio neighbors that are part of the network and needs to select the most preferable one in order to route its traffic toward the authentication, authorization, and accounting (AAA) server. To this end, RFC8180 defines that the advertisement frames carry limited routing information. These advertisement frames are not encrypted and can be accessed by pledges that are not yet part of the network. The pledge can then estimate the closest neighbor to the AAA server and use it for communication to join the network.

V. FLEXIBLE SCHEDULING

The 6TiSCH design targets dynamic scenarios supporting a variety of applications. It relies on a distributed scheduling approach, as opposed to centralized scheduling in WirelessHART and ISA100.11a. Building the schedule in a distributed manner reduces the end-to-end control plane traffic while ensuring adaptability of different parts of the network to changing conditions. Scheduling functionality in 6TiSCH is handled by two components: the 6TiSCH Operation Sublayer (6top) Protocol (6P) and the SF.

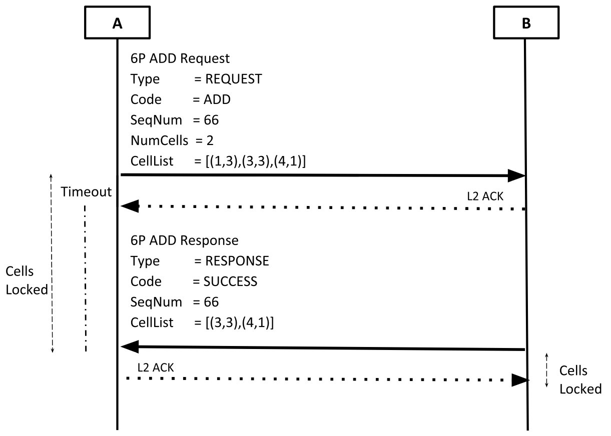


Fig. 2. ***Two-step 6P transaction which results in two additional cells from A to neighbor B.***

A. 6top Protocol

6P is a pairwise negotiation protocol that enables two radio neighbor nodes to allocate cells in their schedules for communication. The protocol defines seven commands to `ADD`, `DELETE`, `RELOCATE`, `COUNT`, `LIST`, `SIGNAL`, or `CLEAR` cells in the neighbor's schedule. This set of commands enables the management of a neighbor's schedule. Transactions occur either in a two-step exchange or a three-step message exchange between nodes. Each transaction ends with nodes either committing the transaction or aborting it.

A two-step transaction enables the requesting node to propose specific cells for use in the schedule of both nodes. In Fig. 2, node A triggers a two-step transaction requesting the allocation of two cells with node B. Node A indicates three candidate cells that may fit its current schedule so B can select two of them. Node B, in this example, confirms the allocation by indicating the selected cells in the response. Timeout events are used to protect the resources from being infinitely locked in case of drops. Sequence numbers are used to match a request with the response, as well as to detect any possible inconsistency in the schedule of the two nodes. A three-step transaction enables the receiver of the request to propose specific cells to use. In this case, interaction is handled through `REQUEST`, `RESPONSE`, and `CONFIRMATION` messages, as illustrated in Fig. 3.

Even with link-layer acknowledgments in place, schedule inconsistencies may happen, for example, due to the internal loss of state. 6P detects possible inconsistencies

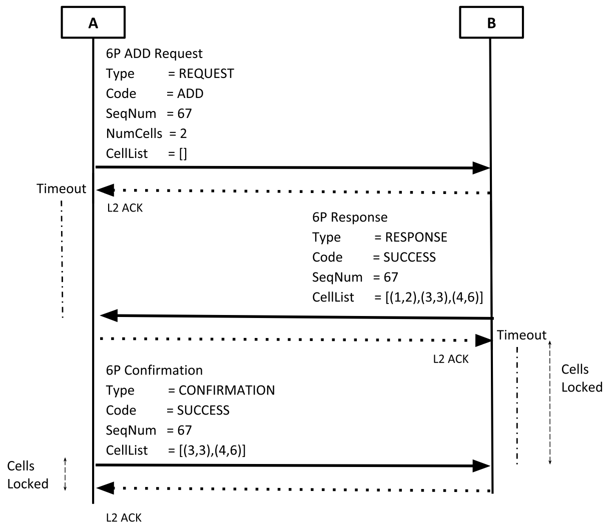


Fig. 3. ***Three-step 6P transaction which results in two additional cells from A to neighbor B.***

based on the sequence numbers and a set of timeouts. In case such an inconsistency is detected, nodes can rollback the individual transaction or eventually clear the entire schedule. This decision is left to the particular SF.

6P messages are carried directly over IEEE 802.15.4 frames. They are encapsulated in a generic structure called information element (IE), specified in IEEE 802.15.4 for the transport of MAC-layer commands and parameters. IEEE specifies many different types of IEs, including those that are open for customization by vendors. 6P uses a generic IE allocated and specified for the IETF [25] and defines how 6P content is encapsulated within. The development of this IE container for the IETF was triggered by the 6TiSCH WG and is a proof of the close liaison between the IETF and the IEEE. The 6P protocol is standardized in the RFC8480 [26].

B. Minimal Scheduling Function

6P provides the signaling for the two nodes to agree on a set of cells to use in their communication schedule. The 6TiSCH vision is to support different types of traffic and application needs by making the cell allocation policy a separate module in the communication stack. This module is called an SF and it is in charge of driving 6P according to the application needs.

The SF design has sparked the interest of the research community. There are numerous SF proposals in the literature aiming at addressing different traffic patterns or optimizations. Examples include traffic requiring low latency [27], bursty traffic [28], scalability [29], end-to-end flow isolation [30], quick setup [31], autonomous operation [32], and content-centric operation [33].

To provide a common base for interoperability purposes, 6TiSCH WG standardizes the minimal SF (MSF) [34]. MSF defines two types of cells to be installed: autonomous cells and managed cells. The autonomous cells are used by nodes to provide the minimal bandwidth with their neighbors. The managed cells are used to dynamically respond to the traffic in the network.

The main idea of autonomous cells comes from Duquennoy *et al.* [32], where nodes calculate the cell's indexes[3] in the schedule only as a function of MAC addresses. A node computes the autonomous cell to receive as a hash of its MAC address. For a neighbor in the IPv6 neighbor cache, the node can compute the autonomous cell to transmit as a hash of the neighbor's MAC address. The technique allows each node to have a minimum amount of bandwidth with its radio neighbors, useful for both control and application traffic. Because the only input to the hash function is a MAC address, the installation of these cells does not require any 6P transactions.

More dynamic traffic needs are handled through the allocation of managed cells using 6P. The interaction of MSF and 6P is handled through a programmatic interface not defined by the IETF as it depends on the particular stack implementation. The interface enables the SF to trigger the transmission of 6P commands.

MSF monitors the utilization of cells with a particular neighbor to determine if more bandwidth should be allocated through additional cells. The utilization of cells with the neighbors is monitored through a running window, a simple technique from the implementation point of view. MSF defines the tunable thresholds that trigger the addition or removal of cells with the neighbor. Fig. 4 illustrates the schedule of two nodes when MSF is used.

MSF also addresses traffic bursts. IEEE 802.15.4 standard defines a special flag called pending bit that is carried in the MAC header. Any node running MSF can set the pending bit to signal to its neighbor that it intends to

[3]As a reminder, a cell in the schedule is uniquely identified by the (`timeslot offset, channel offset`) pair.

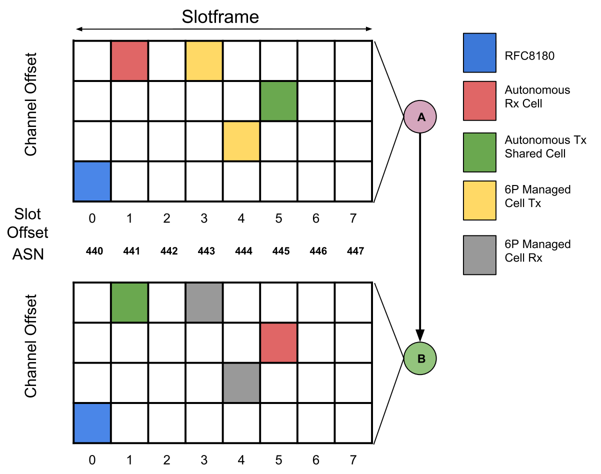


Fig. 4. ***Example schedule produced by the MSF between nodes A and B.***

use the next timeslot for communication, even when the timeslot has not been previously scheduled.

VI. OBJECT-BASED LIGHTWEIGHT SECURITY

When it comes to communication security, existing industrial automation solutions such as WirelessHART and ISA100.11a build on top of the MAC-layer security features provided by the IEEE 802.15.4 standard. The assumption behind the design of the IEEE 802.15.4 security module is that cryptographic keys are already in place for all the nodes in the network. The challenges of network access authentication, i.e., the node that attempts to join the network who it claims to be, authorization, i.e., should this particular node be admitted into the network, and key distribution, i.e., here is the key K that should be used for MAC-layer encryption, are all left out of the scope. In the traditional Internet, these problems are solved by a plethora of different specifications. For example, Extensible Authentication Protocol (EAP) [35] serves as a generic authentication framework. EAP encapsulates EAP methods, such as EAP-pre-shared key (PSK) [36], EAP-Authentication and Key Agreement [37], and EAP-Transport Layer Security (TLS) [38], as the actual implementations of different cryptographic authentication protocols. Then, Protocol for Carrying Authentication for Network Access [39] or 802.1x [40] are necessary to transport EAP messages in different types of networks. In addition, RADIUS [41] or DIAMETER [42] are used to enable centralized AAA decisions in the backend. With this, a new node gets admitted to the network and configured. To communicate securely, TLS [43] is typically used as an umbrella solution for authenticated key agreement and to provide confidentiality, authenticity, and replay protection of the secure channel.

These specifications have evolved over different periods of time and were designed with very different applications in mind. As a result, they:

1) use different data formats;
2) incur significant message overhead to signal information that can be easily inferred;
3) incur significant communication overhead, e.g., to pass the control from one protocol to the other, defined in a different specification, and so the software library;
4) partly overlap in the provided functionality, as they evolved independently.

When implemented together as part of a communication stack, it should, therefore, come as no surprise that few lines of code can be reused, that the resulting messages are lengthy and require fragmentation when transported within IEEE 802.15.4 frames, and that many protocol messages are exchanged unnecessarily. For these reasons, WirelessHART and ISA100.11a defined custom security protocols and data formats to not only provide more efficient solutions but also to enable secure end-to-end communication between a network node and, e.g., the network gateway. While admittedly very efficient, these industrial solutions do not integrate with the existing Internet infrastructure.

TiSCH WG effort also tackles the communication security challenges, with the goal of obtaining the best of both worlds: efficiency of the industrial stacks and security of open Internet solutions. Recently, the security-related effort in the IETF for IoT applications has been focused on the concept of object security [44], [45]. This effort is split among many WGs in the IETF, such as CoRE, ACE, 6TiSCH, and COSE. 6TiSCH adopts object security[4] as a basic primitive to design a secure and efficient communication stack.

The use of a single primitive to carry all security-related components of the communication stack enables major savings in terms of the code footprint: common parsing and decoding routines, common cryptographic algorithm implementations. When information can be implicitly inferred from the context it is transported within, the objects can also be efficiently compressed, resulting in small message overhead. The design of new security mechanisms and protocols, wrapped within these secure objects, allows us to optimize the communication overhead by tailoring the solution(s) to IoT constraints.

A. Constrained Join Protocol

In order to efficiently solve the problems of network access authentication and key distribution, the 6TiSCH WG produced the "minimal security framework for 6TiSCH" specification [46]. The specification defines the process by which a new node, called a pledge, joins the network. During this join process, the pledge communicates with the join registrar/coordinator (JRC), who plays the role of the AAA server.

The specification [46] defines the Constrained Join Protocol (CoJP) that is used by the pledge to request admission into the network and, if the request is granted, for the JRC to configure the pledge with the network parameters including runtime cryptographic keys. Pledge uses one of its radio neighbors that are already part of the network, called a join proxy (JP), to reach the JRC that may reside outside of the local network. As an outcome of the successful CoJP exchange, the pledge is configured with cryptographic keys and other parameters that are used at the MAC layer (e.g., short addresses). The use of these parameters enables the pledge to join the network and start generating application traffic. CoJP also supports the asynchronous parameter update during the network lifetime. This update is initiated by the JRC, and can be used to, e.g., rekey the network or repudiate a misbehaving node.

The pledge joins the network using CoJP in a single round-trip exchange with no fragmentation required. This comes with an added benefit of common code across

[4]Object security refers to a generic data object that is cryptographically protected and carries application data or simply different protocol messages.

the communication stack and, therefore, minimal effort for a new vendor to develop a 6TiSCH-based product. In comparison, it takes EAP-TLS 16 messages to complete the network access exchange.[5] Hence, CoJP results in significant time savings during the network formation phase when only a minimal amount of bootstrap bandwidth is available for communication [23].

CoJP can be used in two deployments options that differ in the provisioning and the type of the security credentials used for network access authentication.

1) One-Touch Credential Provisioning: In the minimal setting, the pledge and the JRC are assumed to share a secret symmetric key, called a PSK. The PSK is typically provisioned out of band, during the installation phase, and is required to be unique for each pledge. This deployment option is the most efficient in terms of communication overhead but results in an additional effort to provision unique PSKs on each device.

2) Zero-Touch Credential Provisioning: In some scenarios, provisioning unique PSKs at deployment time can be quite cumbersome and may increase the installation cost. To facilitate the deployment in such cases, 6TiSCH provides an optional "zero-touch" option where the exchange of security credentials occurs during the manufacture time of the hardware device [47]. It is assumed that the device manufacturer provisions a digital certificate to the pledge and maintains an online Internet service to attest of pledge's identity to the JRC at the join time. Consequently, there is no overhead for credential provisioning during the installation phase, but the option requires the pledge to perform the full authenticated key agreement handshake with the JRC, involving the exchange of certificates and tokens.

With both one-touch and zero-touch credential provisioning options, CoJP exchanges are secured by a mechanism called object security for constrained RESTful environments (OSCORE) [45].

B. Object Security for Constrained RESTful Environments

The OSCORE mechanism is standardized in the CoRE WG of the IETF. It provides end-to-end communication security between two endpoints at the application layer, which potentially communicate over an application-layer proxy. OSCORE encapsulates the application message and certain fields of the protocol header in a secure object, providing confidentiality, authenticity, and replay protection. Based on a secret shared between the communicating endpoints, OSCORE derives a security context, a set of parameters needed to keep the secure channel alive.

6TiSCH leverages OSCORE to secure the CoJP exchanges. In the case of a one-touch deployment option, the OSCORE shared secret is the PSK provisioned to the pledge, enabling the CoJP exchange to complete in a single round trip while providing mutual authentication. In the case of a zero-touch deployment, the CoJP join exchange is preceded by an authenticated key agreement protocol, such that the pledge and the JRC mutually authenticate and derive a shared secret. This secret is then used for OSCORE to protect the subsequent CoJP exchange. Fig. 5 depicts the exchanges taking place during the join.

[5]Eleven messages for the core DTLS 1.2 authentication with PSKs and five messages for EAP signaling [38].

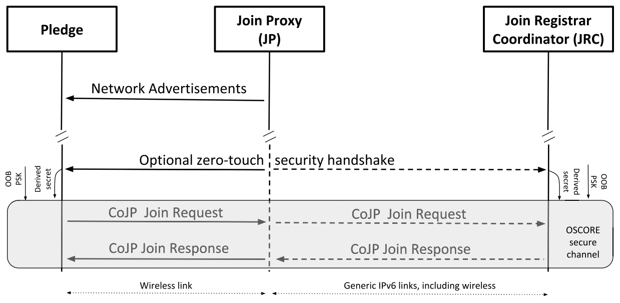


Fig. 5. ***Exchanges taking place during the join process. Solid lines: physical links. Dashed lines: logical connections. OOB stands for out of band.***

Since OSCORE is already a mandatory component of CoJP, the application developers can reuse the same implementation for communication security, resulting in lower effort and code footprint savings. OSCORE is also used as a secure channel for the transport of authorization tokens [48], as standardized in ACE WG. These tokens carry dynamic keying material allowing IoT devices to establish secure channels with generic hosts in the Internet.

VII. EXTENSIBILITY

Industrial control and automation applications may require extended network services such as synchronization or end-to-end flow isolation to support different requirements, for example, alarms. Even though the standardization efforts at the IETF have consolidated the protocol stack through the 6TiSCH architecture, the modular design facilitates extensibility.

While the 6TiSCH WG is chartered to produce a single SF, with the adopted candidate being the MSF, the architecture encourages the development of additional SFs. 6P, for example, has been designed as a generic pairwise negotiation mechanism that can support multiple concurrent SFs. This facilitates the support for different types of traffic that may be simultaneously present in a network. The numerous proposed SFs listed in Section V-B are an indication of the attractiveness of this extensibility feature. We find this modular aspect a key to the successful adoption of 6TiSCH.

In addition, precise synchronization and global time awareness extend network functionality to applications that require the correlation of data from different sources. The network time protocol (NTP) [49] and IEEE 1588 [50] have not been designed to be transported in small frames, making them unsuitable for IEEE 802.15.4. 6TiSCH takes advantage of the synchronized nature of the underlying TSCH and enables nodes to share a common base of time. Through a simple protocol extension [51], a 6TiSCH network can be augmented with the global time information at a relatively small cost. Essentially, as the nodes are synchronized, a global time reference can be transported to any node in the network leveraging the CoJP. This global time information is then mapped to the local timekeeping and periodically refreshed.

It is common in industrial automation systems that certain application events have higher priority than others. In such situations, the corresponding packets need to reach the back-end SCADA system with minimal latency. Taking advantage of the IPv6 features, traffic can be tagged with a traffic class field—referred as DiffServ code point (DSCP) [52]—and cross the network with higher priority. This functionality has been defined as part of the 6TiSCH architecture [18] and becomes a relevant advantage when comparing 6TiSCH to WirelessHART. Mainly due to the IPv6 nature of the 6TiSCH architecture, bridging or application layer packet handling is not needed at the routers, while flow isolation is natively supported by the IP infrastructure through the DSCP IPv6 header field.

VIII. PERFORMANCE OF TODAY'S MINIMAL 6TiSCH

OpenWSN[6] open-source project [53] implements the entire set of "minimal" 6TiSCH specifications. OpenWSN is considered the reference implementation during interoperability testing events organized by ETSI [54]. We report here the real-world performance numbers of the OpenWSN implementation, in its `Release 1.14.0`.

Note that the presented performance numbers should not be considered as representative for 6TiSCH, in general, for two main reasons. First, the quality of the hardware and the implementation directly influence the performance. For instance, with a proper hardware acceleration, a solution will consume less compared to an entirely software-based solution, such as the OpenWSN project. Second, as discussed in Section VII, 6TiSCH has been designed with extensibility in mind. A SF specifically designed to provide low latency will outperform the numbers shown here. We do believe, however, that the presented results give insight into the performance of a "vanilla" implementation of 6TiSCH.

The measurements presented in this section are obtained by running the OpenWSN implementation in a network of 37 "A8" nodes,[7] deployed in the Saclay site of the IoT-Lab testbed.[8] The nodes are deployed in an underground parking facility, as shown in Fig. 6. The nodes run OpenWSN release 1.14.0.

[6]http://www.openwsn.org/
[7]https://www.iot-lab.info/hardware/a8/
[8]https://www.iot-lab.info/

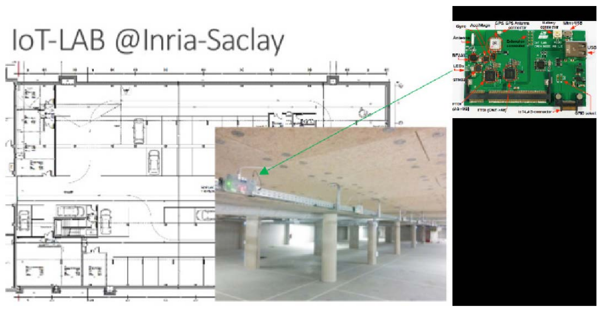


Fig. 6. ***Measurements are done on a network of 37 "A8" nodes on the Saclay IoT-Lab testbed, deployed in a parking structure.***

A. Reliability

We define end-to-end reliability as the ratio between the number of received packets and the number of packets sent over the duration of the experiment. Note the difference between the end-to-end reliability and the reliability of individual links in the network, where MAC-layer retransmissions can take place. Because of the channel hopping nature of TSCH, each MAC-layer retransmission happens on a different channel, enhancing the reliability of the network, thanks to the uncorrelated impact of multipath fading per channel [11]. Consequently, a dropped frame influences the end-to-end reliability only if it has been retransmitted the maximum number of times. A default value of this parameter specifying how many times a node should retransmit a frame before giving up is suggested by the RFC8180.

To measure end-to-end reliability in this experiment, we use routing packets that travel from individual nodes in the network toward the root. Therefore, these packets traverse multiple hops and are subject to MAC-layer retransmissions. Over the course of the experiment, a total of 3544 packets were exchanged. We present the measured end-to-end reliability shown in Fig. 7 for each of the 36 nodes in the network. The average for the network

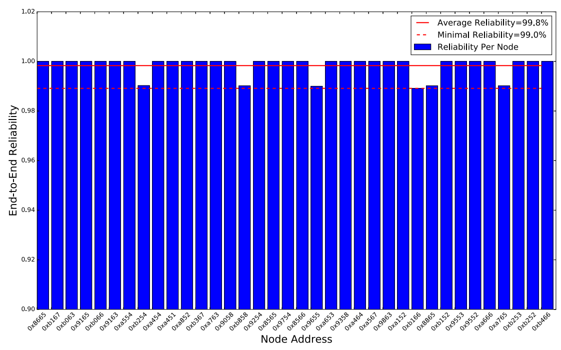


Fig. 7. ***Average end-to-end reliability of the 36 nodes in the experimentation testbed is 99.8%; any single node has an end-to-end reliability higher than 99%.***

is 99.8%, with each node achieving ≥99% end-to-end reliability.

Improved reliability can be achieved by exploiting other forms of redundancy in the network scheduling. To that end, path diversity or replication through disjoint paths can be exploited [55].

B. Latency

Latency represents the time interval between the instant a packet is generated at the sender and the instant it is received at the destination, potentially multiple hops away. The latency is influenced by multiple parameters, such as the number of hops to the destination, the slotframe length, the relative position in the schedule of the receive/transmit cells, and the PDR at each hop. End-to-end latency can be described using

$$L = \sum_{n=1}^{H_{\text{num}}} \left(\text{offset}_{rxtx} + \frac{L_{\text{frame}}}{\text{PDR}} \right)_n \tag{2}$$

where H_{num} is the number of hops from a source node to its destination, $\text{offset}_{\text{cells}}$ is the offset in the node's schedule from the *Rx* cell from its previous hop to the *Tx* cell to its next hop, L_{frame} is the length of the slotframe, and PDR indicates the packet delivery ratio to its next hop.

Minimal latency will typically be achieved using scheduling techniques that "daisy chain" transmit and receive cells along multiple hops in the network [27]. Other techniques yield a higher latency as transmit and receive cells are placed at different offsets within the schedule. Table 1 gives the calculated bad case latency, for a 10-ms timeslot length, several slotframe lengths (31, 67, 101), and several average PDR values per link, in a five-hop linear network. We could observe that the latency is higher when the receive cell from the previous hop in a node's schedule is preceded by the transmit cell to its routing parent. The node then needs to buffer the packet for the duration of the complete slotframe, before sending it to the next hop.

Fig. 8 shows the latency in the network with latency-aware scheduling, computed using (2). In this case, we consider that the node can handle the maximum number of retransmissions to a given neighbor within the same slotframe. In this case, a node has at least three transmit cells to its parent and three receive cells from its children. Consequently, most of the eventual retransmissions can be handled within the same slotframe.

Table 1 Latency in a Five-Hop Linear Network With Suboptimal Scheduling

PDR	31 cells	67 cells	101 cells
100%	3,050 ms	6,650 ms	10,050 ms
80%	3,438 ms	7,488 ms	11,313 ms
60%	4,083 ms	8,883 ms	13,417 ms
40%	5,375 ms	11,675 ms	17,625 ms

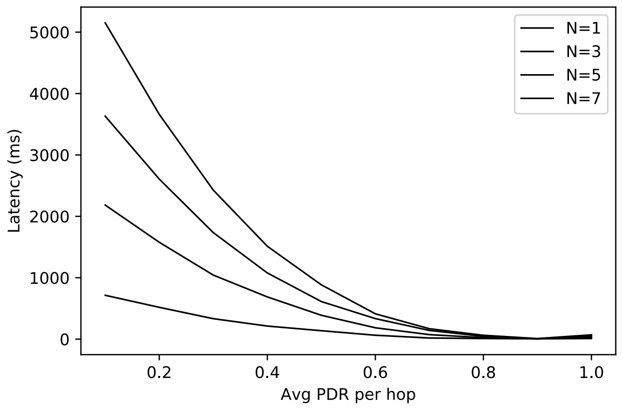


Fig. 8. ***Computed latency with latency-aware scheduling. Latency is a function of the number of hops N, and the average PDR per hop. These results assume a slotframe length of 101 timeslots, a timeslot of 10 ms, and a maximum of three link-layer retransmissions. We consider that each node has at least three transmit cells and three receive cells chained (as per LLSF) from its previous and next hops.***

C. Scalability

The capacity of the root node is in practice the limiting factor for the size of the network. This limitation typically comes from the maximum number of cells the root can have with its immediate children, as well as their reporting rate.

Traffic from the root of the network toward the nodes, i.e., the downstream traffic, is typically routed using the source routing mechanism. The source route is added as an extension header to IPv6 and contains the list of compressed addresses that the packet needs to traverse to reach the destination. The 6TiSCH WG has promoted and adopted the 6LoRH compression format standardized in RFC8138 [56] to efficiently compress the source routes.

In the best case, when the MAC addresses of the nodes along a path differ in a single byte, source routing overhead can be compressed to one byte per address. 6TiSCH can then support nodes that are up to 42 hops deep, before needing fragmentation.

The deeper is the network, the higher is the traffic load that needs to be forwarded by the nodes closer to the root. This causes networkwise energy imbalance and the nodes closer to the root typically consume more energy.

D. Energy Consumption

The energy consumption is tightly coupled to the hardware used and the network topology. We physically measure the energy consumption of the I3Mote [57] running the OpenWSN implementation and transmitting sensor data. The consumption measured on other platforms

Table 2 Charge Consumed for Different Networking Actions, Measured on the I3Mote Platform. TX Stands for Transmit, RX for Receive

Action	Charge
TX advertisement (84 bytes)	32.92 μC
RX advertisement (84 bytes)	34.62 μC
TX (127 bytes, 25 bytes ACK)	57.91 μC
TX (no data)	2.26 μC
RX (127 bytes, 25 bytes ACK)	60.21 μC
RX (no packet reception)	23.98 μC

will be different and dependent on hardware characteristics and implementation optimizations out of the scope of this paper. Table 2 presents the charge consumed by a node to perform the main actions that occur during the network operation. These numbers can be used to feed energy consumption models and accurately predict battery lifetime [58].

For example, we can derive the energy consumption of node B in Fig. 4. Let us assume a 101-slotframe length, 10-ms timeslots, a network topology as depicted in Fig. 4, network advertisements periodically sent every 30 s, node A reporting 127 B of data every 10 s, and node B being powered through a AA 1500-mAh LiPo battery. Under these assumptions reflecting a common configuration, node B consumes 80.8 μA on average, which leads to the battery lifetime of 1.5 years.

IX. 6TiSCH ECOSYSTEM

6TiSCH specifications have been standardized in parallel with a relevant evolution of the software ecosystem. This includes different tools and full protocol stack implementations, instantiating the architecture defined by the 6TiSCH WG [18].

The major IoT open-source projects implement the 6TiSCH architecture and specifications: OpenWSN [53], Contiki-NG [59], and RIOT [60]. "Pre-6TiSCH" commercial products are already on the market, including Analog Devices' SmartMesh IP product lines [13].

The community has also developed tools to facilitate the conformance and interoperability testing of the implementations, in parallel with the standardization process. The F-Interop project [61] has developed an online testing platform that includes a 6TiSCH specification test suite [62]. The tests have been derived from the different ETSI Plugtests events occurred in the last four years and address the main lessons learned [54].

The 6TiSCH simulator [63] has been designed to support the evaluation of large networks quickly through high-level, discrete-event abstractions. It allows the experimentation with controlled topologies and facilitates the development and evaluation of different 6TiSCH specifications configurations at network scale.

The ecosystem is also complemented with different experimentation facilities and tools. For example, IoT-Lab [64] supports the deployment of networks executing the 6TiSCH specifications. The OpenTestbed[9] is an open-source testbed solution developed by the OpenWSN community, building upon the previous Rover [65] project.

6TiSCH open data action (SODA) project develops the tools that automate the performance benchmarking of 6TiSCH implementations on testbeds [66] and facilitate reproducible and repeatable research. SODA also aims at providing reference performance data sets of 6TiSCH in industry relevant test scenarios that can be used by industry stakeholders to assess whether 6TiSCH meets their requirements, by the researchers to evaluate and compare new proposals, and by the 6TiSCH WG to identify performance bottlenecks and evolve the next generation of standards.

Different hardware platforms have been designed with the 6TiSCH requirements in mind. OpenMote [67] is an open hardware initiative designed for industrial low-power network prototyping. I3Mote [57] is designed to interface industrial buses and bridge them to an industrial network using the 6TiSCH specifications. The Beamlogic[10] is a 16-channel packet sniffer that has been designed to facilitate the debugging of channel hopping protocol stacks such as the TSCH mode of IEEE 802.15.4 that 6TiSCH is based on. Beamlogic is complemented with the Argus software[11] to allow multiple users to access the data stream sniffed on a single device. Argus publishes the sniffed data to a broker where different subscribers, e.g., a remote Wireshark instance, are able to receive and analyze it independently. Wireshark[12] is a well-known open-source network analyzer used to analyze and validate the standard compliance of network protocols.

X. CONCLUSION

The evolution of industrial automation technologies is demanding a new generation of wireless communication stacks that can seamlessly connect industrial equipment to the Internet while meeting the reliability and security constraints imposed by the industrial scenarios. This paper is an overview of the 6TiSCH standards, developed in an open standardization context by the IETF. 6TiSCH specifications permit the development of fully Internet-compatible communication technologies that enables machines to interconnect to the new generation of cloud-hosted SCADA systems while ensuring traffic guarantees and not requiring adapters or application layer bridges. The 6TiSCH architecture, therefore, can be seen as an industrial plug-and-play technology enabler, designed to augment legacy equipment, and facilitate the vision of a hyperconnected industry. ■

[9]https://github.com/openwsn-berkeley/opentestbed
[10]http://www.beamlogic.com
[11]https://github.com/openwsn-berkeley/argus
[12]https://www.wireshark.org

REFERENCES

[1] J. Higgins, "Digital transformation of European industry and enterprises, strategic policy forum on digital entrepreneurship," in *Proc. Eur. Commission Ref. Ares*, Mar. 2015.

[2] V. C. Gungor and G. P. Hancke, "Industrial wireless sensor networks: Challenges, design principles, and technical approaches," *IEEE Trans. Ind. Electron.*, vol. 56, no. 10, pp. 4258–4265, Oct. 2009.

[3] B. Martínez, C. Cano, and X. Vilajosana. (2018). "A square peg in a round hole: The complex path for wireless in the manufacturing industry." [Online]. Available: https://arxiv.org/abs/1808.03065

[4] D. Chen, M. Nixon, and A. Mok, *WirelessHART: Real-Time Mesh Network for Industrial Automation*, 1st ed. Springer, 2010.

[5] *Wireless Systems for Industrial Automation: Process Control and Related Applications*, Standard ANSI/ISA-100.11a-2011, 2009. [Online]. Available: https://isa100wci.org/

[6] *Industrial Communication Networks—Fieldbus Specifications—WIA-PA Communication Network and Communication Profile*, Standard IEC 62601:2015, Nov. 2011.

[7] *Industrial Networks—Wireless Communication Network and Communication Profiles—WIA-FA*, Standard IEC 62948:2017, Jul. 2017.

[8] M. Zheng, W. Liang, H. Yu, and Y. Xiao, "Performance Analysis of the Industrial Wireless Networks Standard: WIA-PA," *Mobile Netw. Appl.*, vol. 22, no. 1, pp. 139–150, Feb. 2017.

[9] W. Liang, X. Zhang, Y. Xiao, F. Wang, P. Zeng, and H. Yu, "Survey and experiments of WIA-PA specification of industrial wireless network," *Wireless Commun. Mobile Comput.*, vol. 11, no. 8, pp. 1197–1212, Aug. 2011.

[10] K. S. Pister and L. Doherty, "TSMP: Time synchronized mesh protocol," in *Proc. IASTED Int. Symp. Distrib. Sensor Netw. (DSN)*, 2008.

[11] T. Watteyne, A. Mehta, and K. S. Pister, "Reliability through frequency diversity: Why channel hopping makes sense," in *Proc. ACM Symp. Perform. Eval. Wireless Ad Hoc, Sensor, Ubiquitous Netw. (PE-WASUN)*, 2009, pp. 116–123.

[12] L. Doherty, W. Lindsay, and J. Simon, "Channel-specific wireless sensor network path data," in *Proc. Conf. Comput. Commun. Netw. (CCN)*, Aug. 2007, pp. 89–94.

[13] T. Watteyne, L. Doherty, J. Simon, and K. Pister, "Technical overview of smartmesh IP," in *Proc. 7th Int. Conf. Innov. Mobile Internet Services Ubiquitous Comput.*, Jul. 2013, pp. 547–551.

[14] X. Vilajosana *et al.*, "A realistic energy consumption model for TSCH networks," *IEEE Sensors J.*, vol. 14, no. 2, pp. 482–489, Feb. 2014.

[15] T. Watteyne, J. Weiss, L. Doherty, and J. Simon, "Industrial ieee802.15.4e networks: Performance and trade-offs," in *Proc. IEEE ICC*, Jun. 2015, pp. 604–609.

[16] O. Givehchi, H. Trsek, and J. Jasperneite, "Cloud computing for industrial automation systems—A comprehensive overview," in *Proc. IEEE 18th Conf. Emerg. Technol. Factory Autom. (ETFA)*, Sep. 2013, pp. 1–4.

[17] T. Watteyne *et al.*, "Industrial wireless IP-based cyber–physical systems," *Proc. IEEE*, vol. 104, no. 5, pp. 1025–1038, May 2016.

[18] P. Thubert, *An Architecture for IPv6 Over the TSCH Mode of IEEE 802.15.4*, document draft-ietf-6tisch-architecture-11, Apr. 2018.

[19] B. M. Leiner *et al.*, "A brief history of the Internet," *ACM SIGCOMM Comput. Commun. Rev.*, vol. 39, no. 5, pp. 22–31, 2009.

[20] T. Watteyne, M. R. Palattella, and L. A. Grieco, *Using IEEE 802.15.4e Time-Slotted Channel Hopping (TSCH) in the Internet of Things (IoT): Problem Statement*, document RFC7554, Internet Engineering Task Force, May 2015.

[21] X. Vilajosana, P. Tuset-Peiro, F. Vazquez-Gallego, J. Alonso-Zarate, and L. Alonso, "Standardized low-power wireless communication technologies for distributed sensing applications," *Sensors*, vol. 14, no. 2, pp. 2663–2682, 2014.

[22] X. Vilajosana, K. S. Pister, and T. Watteyne, *Minimal IPv6 Over the TSCH Mode of IEEE 802.15.4e (6TiSCH) Configuration*, document RFC 8180, Internet Engineering Task Force, May 2017.

[23] M. Vučinić, T. Watteyne, and X. Vilajosana, "Broadcasting strategies in 6TiSCH networks," *Internet Technol. Lett.*, vol. 1, no. 1, p. e15, Jan./Feb. 2017.

[24] T. Winter *et al.*, *RPL: IPv6 Routing Protocol for Low-Power and Lossy Networks*, document RFC 6550, Internet Engineering Task Force, Mar. 2012.

[25] T. Kivinen and P. Kinney, *IEEE 802.15.4 Information Element for the IETF*, document RFC 8137, Internet Engineering Task Force, May 2017.

[26] Q. Wang, X. Vilajosana, and T. Watteyne, *6TiSCH Operation Sublayer (6top) Protocol (6P)*, document RFC 8480, Internet Engineering Task Force, Nov. 2018.

[27] T. Chang, T. Watteyne, Q. Wang, and X. Vilajosana, "LLSF: Low latency scheduling function for 6TiSCH networks," in *Proc. IEEE Int. Conf. Distrib. Comput. Sensor Syst. (DCOSS)*, May 2016, pp. 93–95.

[28] M. Domingo-Prieto, T. Chang, X. Vilajosana, and T. Watteyne, "Distributed PID-based scheduling for 6TiSCH networks," *IEEE Commun. Lett.*, vol. 20, no. 5, pp. 1006–1009, May 2016.

[29] E. Municio and S. Latré, "Decentralized broadcast-based scheduling for dense multi-hop TSCH networks," in *Proc. ACM Workshop Mobility Evolving Internet Archit. (MobiArch)*. New York, NY, USA, 2016, pp. 19–24.

[30] F. Theoleyre and G. Z. Papadopoulos, "Experimental validation of a distributed self-configured 6TiSCH with traffic isolation in low power lossy networks," in *Proc. ACM Int. Conf. Modeling, Anal. Simulation Wireless Mobile Syst. (MSWiM)*, New York, NY, USA, 2016, pp. 102–110.

[31] K. Choi and S.-H. Chung, "Enhanced time-slotted channel hopping scheduling with quick setup time for industrial Internet of Things networks," *Int. J. Distrib. Sensor Netw.*, vol. 13, no. 6, pp. 1–20, 2017.

[32] S. Duquennoy, B. Al Nahas, O. Landsiedel, and T. Watteyne, "Orchestra: Robust mesh networks through autonomously scheduled TSCH," in *Proc. ACM Conf. Embedded Netw. Sensor Syst. (SenSys)*, Seoul, South Korea, 2015, pp. 337–350.

[33] Y. Jin, U. Raza, A. Aijaz, M. Sooriyabandara, and S. Gormus, "Content centric cross-layer scheduling for industrial IoT applications using 6TiSCH," *IEEE Access*, vol. 6, pp. 234–244, 2018.

[34] T. Chang, M. Vučinić, X. Vilajosana, S. Duquennoy, and D. Dujovne, *6TiSCH Minimal Scheduling Function (MSF)*, document draft-ietf-6tisch-msf-01, Internet Engineering Task Force, Oct. 2018.

[35] J. Vollbrecht, J. D. Carlson, L. Blunk, B. D. Aboba, and H. Levkowetz, *Extensible Authentication Protocol (EAP)*, document RFC 3748, Internet Engineering Task Force, Jun. 2004.

[36] F. Bersani and H. Tschofenig, *The EAP-PSK Protocol: A Pre-Shared Key Extensible Authentication Protocol (EAP) Method*, document RFC 4764, Internet Engineering Task Force, Jan. 2007.

[37] J. Arkko and H. Haverinen, *Extensible Authentication Protocol Method for 3rd Generation Authentication and Key Agreement (EAP-AKA)*, document RFC 4187, Internet Engineering Task Force, Jan. 2006.

[38] D. Simon, R. Hurst, and B. D. Aboba, *The EAP-TLS Authentication Protocol*, document RFC 5216, Internet Engineering Task Force, Mar. 2008.

[39] D. Forsberg, B. Patil, A. E. Yegin, Y. Ohba, and H. Tschofenig, *Protocol for Carrying Authentication for Network Access (PANA)*, document RFC 5191, Internet Engineering Task Force, May 2008.

[40] *IEEE Standard for Local and Metropolitan Area Networks—Port-Based Network Access Control*, IEEE Standard 802.1X-2010, 2010.

[41] A. Rubens, C. Rigney, S. Willens, and W. A. Simpson, *Remote Authentication Dial-in User Service (RADIUS)*, document RFC 2865, Internet Engineering Task Force, Jun. 2000.

[42] V. Fajardo, J. Arkko, J. Loughney, and G. Zorn, *Diameter Base Protocol*, document RFC 6733, Internet Engineering Task Force, Oct. 2012.

[43] E. Rescorla and T. Dierks, *The Transport Layer Security (TLS) Protocol Version 1.2*, document RFC 5246 Internet Engineering Task Force, Aug. 2008.

[44] M. Vučinić, B. Tourancheau, F. Rousseau, A. Duda, L. Damon, and R. Guizzetti, "OSCAR: Object security architecture for the Internet of Things," *Ad Hoc Netw.*, vol. 32, pp. 3–16, Sep. 2015.

[45] G. Selander, J. Mattsson, F. Palombini, and L. Seitz, *Object Security for Constrained RESTful Environments (OSCORE)*, document draft-ietf-core-object- security-13, Internet Engineering Task Force, Jun. 2018.

[46] M. Vučinić, J. Simon, K. S. Pister, and M. Richardson, *Minimal Security Framework for 6TiSCH*, document draft-ietf-6tisch-minimal-security-09, Internet Engineering Task Force, Nov. 2018.

[47] M. Richardson, *6tisch Zero-Touch Secure Join protocol*, document draft-ietf-6tisch-dtsecurity-zerotouch-join-03, Internet Engineering Task Force, Oct. 2018.

[48] L. Seitz, F. Palombini, M. Gunnarsson, and G. Selander, *OSCORE profile Authentication Authorization for Constrained Environments Framework*, document draft-ietf-ace-oscore-profile-02, Internet Engineering Task Force, Jun. 2018.

[49] J. Martin, J. Burbank, W. Kasch, and P. D. L. Mills, *Network Time Protocol Version 4: Protocol and Algorithms Specification*, document RFC 5905, Jun. 2010. [Online]. Available: https://rfc-editor.org/rfc/rfc5905.txt

[50] K. Lee and J. Eidson, "IEEE 1588 standard for a precision clock synchronization protocol for networked measurement and control systems," in *Proc. 34th Annu. Precise Time Time Interval (PTTI) Meeting*, 2002, pp. 98–105.

[51] X. Vilajosana, P. Tuset-Peiro, B. Martinez, and J. Munoz, *Global Time Distribution in 6TiSCH Networks*, document draft-vilajosana-6tisch-globaltime-02, Internet Engineering Task Force, Jul. 2018.

[52] J. Polk, F. Baker, and M. Dolly, *A Differentiated Services Code Point (DSCP) for Capacity-Admitted Traffic*, document RFC 5865, Internet Engineering Task Force, May 2010.

[53] T. Watteyne *et al.*, "OpenWSN: A standards-based low-power wireless development environment," *Trans. Emerg. Telecommun. Technol.*, vol. 23, no. 5, pp. 480–493, 2012.

[54] M. R. Palattella, X. Vilajosana, T. Chang, M. A. R. Ortega, and T. Watteyne, "Lessons learned from the 6TiSCH plugtests," in *Internet of Things. IoT Infrastructures*. Cham, Switzerland: Springer, 2016, pp. 415–426.

[55] J. de Armas, P. Tuset, T. Chang, F. Adelantado, T. Watteyne, and X. Vilajosana, "Determinism through path diversity: Why packet replication makes sense," in *Proc. Int. Conf. Intell. Netw. Collaborative Syst. (INCoS)*, Sep. 2016, pp. 150–154.

[56] P. Thubert, C. Bormann, L. Toutain, and R. Cragie, *IPv6 over Low-Power Wireless Personal Area Network (6LoWPAN) Routing Header*, document RFC 8138, Internet Engineering Task Force, Apr. 2017.

[57] B. Martinez *et al.*, "I3Mote: An open development platform for the intelligent industrial Internet," *Sensors*, vol. 17, no. 5, p. 986, Apr. 2017.

[58] X. Vilajosana, Q. Wang, F. Chraim, T. Watteyne, T. Chang, and K. S. Pister, "A realistic energy consumption model for TSCH networks," *IEEE Sensors J.*, vol. 14, no. 2, pp. 482–489, Feb. 2014.

[59] S. Duquennoy, A. Elsts, B. Al Nahas, and G. Oikonomo, "TSCH and 6TiSCH for contiki: Challenges, design and evaluation," in *Proc. Int. Conf. Distrib. Comput. Sensor Syst. (DCOSS)*, Ottawa, ON, Canada, Jun. 2017, pp. 1–8.

[60] E. Baccelli, O. Hahm, M. Günes, M. Wählisch, and T. Schmidt, "RIOT OS: Towards an OS for the Internet of Things," in *Proc. IEEE Int. Conf. Comput. Commun. (INFOCOM)*, Turin, Italy, Apr. 2013, pp. 79–80.

[61] S. Ziegler, S. Fdida, T. Watteyne, and C. Viho, "F-Interop—Online conformance, interoperability and performance tests for the IoT," in *Proc. Conf. Interoperability IoT (InterIoT)*, Paris, France, Oct. 2016.

[62] M.-R. Palattella *et al.*, "F-Interop platform and tools: Validating IoT implementations faster," in *Proc. 17th Int. Conf. Ad Hoc Netw. Wireless (AdHoc-Now)*, Saint Malo, France, Sep. 2018, pp. 1–12. [Online]. Available: https://hal.inria.fr/hal-01858004

[63] E. Municio *et al.*, "Simulating 6TiSCH networks," *Trans. Emerg. Telecommun. Technol.*, vol. 30, no. 3, p. e3494, 2019.

[64] C. Adjih *et al.*, "FIT IoT-LAB: A large scale open experimental IoT testbed," in *Proc. World Forum Internet of Things (WF-IoT)*, Dec. 2015, pp. 459–464.

[65] Z. Brodard *et al.*, "Rover: Poor (but Elegant) man's testbed," in *Proc. ACM Int. Symp. Perform. Eval. Wireless Ad Hoc, Sensor, Ubiquitous Netw. (PE-WASUN)*, Valletta, Malta, Nov. 2016, pp. 61–65.

[66] M. Vučinić, M. Pejanović-Djurisić, and T. Watteyne, "SODA: 6TiSCH open data action," in *Proc. IEEE Workshop Benchmarking Cyber-Phys. Netw. Syst. (CPSBench)*, Apr. 2018, pp. 42–46.

[67] X. Vilajosana, P. Tuset-Peiro, T. Watteyne, and K. Pister, "OpenMote: Open-source prototyping platform for the industrial IoT," in *Proc. Int. Conf. Ad Hoc Netw. (AdHocNets)*. Sanremo, Italy: EAI, Sep. 2015, pp. 211–222.

ABOUT THE AUTHORS

Xavier Vilajosana (Senior Member, IEEE) was a Visiting Researcher at France Telecom R&D Labs, Paris, France, in 2008. From 2012 to 2014, he was a Visiting Professor at the University of California at Berkeley (UC Berkeley), Berkeley, CA, USA, and held a prestigious Fulbright Fellowship. Until 2016, he was a Senior Researcher with the HP Research and Development Laboratories, Barcelona, Spain. He is currently the Principal Investigator with the WINE Research Group, Universitat Oberta de Catalunya, Barcelona, where he is also a Professor with the Computer Science, Telecommunications, and Multimedia Department. He is also a Co-Founder of Worldsensing, Barcelona, and OpenMote Technologies, Barcelona, two prominent startups developing cutting-edge Internet-of-Things-related technology. He has been one of the main promoters of low-power wireless technologies, co-leading the OpenWSN.org initiative at UC Berkeley, and promoting the use of low-power wireless standards for the emerging industrial Internet paradigm. He also contributed to the industrialization and introduction of low-power wide area networks to urban scenarios through Worldsensing. He has authored different Internet Drafts and RFCs as part of his standardization activities for low-power industrial networks. He actively contributes to the IETF 6TiSCH, 6Lo, and ROLL Working Groups. He has authored or co-authored more than 30 high-impact journal publications and has contributed several demos, tutorials, and courses in the field of low-power wireless networks. He holds more than 20 patents.

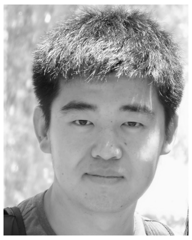
Mr. Vilajosana is a founding member and a member of the IEEE Sensors Council in Spain.

Thomas Watteyne (Senior Member, IEEE) received the M.Sc. degree in networking and the M.Eng. degree in telecommunications, and the Ph.D. degree in computer science from INSA Lyon, Villeurbanne, France, in 2005 and 2008, respectively.

Between 2005 and 2008, he was a Research Engineer at France Telecom, Orange Labs, Paris, France. He founded and co-leads Berkeley's OpenWSN Project, University of California at Berkeley, Berkeley, CA, USA, an open-source initiative to promote the use of fully standards-based protocol stacks for the Internet-of-Things (IoT). He was a Postdoctoral Research Lead with Prof. K. Pister's Team at the University of California at Berkeley. Since 2013, he has been a Co-Chair of the IETF 6TiSCH Working Group, which standardizes how to use IEEE802.15.4e TSCH in IPv6-enabled mesh networks and has been a member of the IETF Internet-of-Things Directorate. He is an insatiable enthusiast of low-power wireless mesh technologies. He currently holds an advanced research position with the EVA Research Team, INRIA, Paris, where he designs, models and builds networking solutions based on a variety of IoT standards. He is also a Senior Networking Design Engineer with the Dust Networks Product Group, Analog Devices, Limerick, Ireland, the undisputed leader in supplying low-power wireless mesh networks for demanding industrial process automation applications.

Mališa Vučinić (Member, IEEE) received the Electrical Engineering degree from the University of Montenegro, Podgorica, Montenegro, in 2010, the joint master's degree (Hons.) from the Politecnico di Torino, Turin, Italy, and the Grenoble Institute of Technology, Grenoble, France, in 2012, and the Ph.D. degree from Grenoble Alps University, Grenoble, in 2015.

From 2012 to 2015, he was a Research Engineer with STMicroelectronics, Crolles, France, and was a Visiting Scholar with the University of California at Berkeley, Berkeley, CA, USA, in 2015. He is currently a Researcher Scientist with the EVA Team, INRIA, Paris, France. He is active in the IETF and has co-authored multiple drafts in different working groups, including 6TiSCH. Main concepts from his Ph.D. dissertation have found their way to the Internet standards tackling object security and its use in the IoT. His current research interests include the intersection of network security and performance analysis, theory, and practice.



Tengfei Chang received the Ph.D. degree in computer systems architecture from the University of Science and Technology, Beijing, China, in 2017.

In 2014, he joined the University of California at Berkeley (UC Berkeley), Berkeley, CA, USA, as a Visiting Scholar. From 2015 to 2017, he was with the EVA Team, INRIA, Paris, France, as a Pre-Postdoctoral Research Engineer, where he led the project of OpenWSN, which is an open-source project founded by UC Berkeley. In 2017, he joined the F-Interop Project as a Postdoctoral Research Engineer at INRIA, which is an H2020 European research project. He is a Postdoctoral Research Engineer with the EVA Team, INRIA. He was a Technical Support for 6TiSCH interoperability plugtest. He is also one of the main implementers of the IETF 6TiSCH standard protocol stack. His current research interests include wireless sensor and actuator networks, swarm robotics, and any embedded systems design.

Kristofer S. J. Pister received the B.A. degree in applied physics from the University of California at San Diego, La Jolla, CA, USA, in 1986, and the M.S. and Ph.D. degrees in electrical engineering and computer science (EECS) from the University of California at Berkeley, Berkeley, CA, USA, in 1989 and 1992, respectively.

He taught in the Electrical Engineering Department, University of California at Los Angeles, Los Angeles, CA, USA. In 1996, he was with EECS, University of California at Berkeley. He developed Smart Dust, a project with the goal of putting a complete sensing/communication platform inside a cubic millimeter. For this project, he was awarded the second annual Alexander Schwarzkopf Prize for Technological Innovation, in 2006, from the I/UCRC Association, for developing and successfully commercializing Smart Dust. He has also focused his energies on synthetic insects, which he has characterized as basically Smart Dust with legs. He is the Co-Director of the Berkeley Sensor and Actuator Center (BSAC) and the Ubiquitious Swarm Laboratory, University of California at Berkeley.

Dr. Pister was a recipient of the Alfred F. Sperry Founder Award in 2009 for his contributions to the science and technology of instrumentation, systems, and automation.