6TiSCH: Industrial Performance for IPv6 Internet of Things Networks

Xavier Vilajosana, Thomas Watteyne, Malisa Vucinic, Tengfei Chang, Kristofer Pister

6TiSCH: Industrial Performance for IPv6 Internet of Things Networks

Xavier Vilajosana, *Senior Member, IEEE,* Thomas Watteyne, *Senior Member, IEEE,*
Mališa Vučinić, Tengfei Chang, 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, distributed and modular scheduling mechanisms. As a cross-layer effort, 6TiSCH leverages and integrates other IETF specifications and the working group has also driven the definition of novel specifications in other IETF working groups. 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 article introduces the work done by the 6TiSCH WG at IETF, evaluates the performance of the reference implementation and discusses the 6TiSCH software ecosystem.**

***Index Terms*—Industrial Networks, IETF, 6TiSCH, IEEE 802.15.4, Time-slotted Channel Hopping, IPv6**

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 20^{th} century saw the 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 20^{th} century, remains a vision.

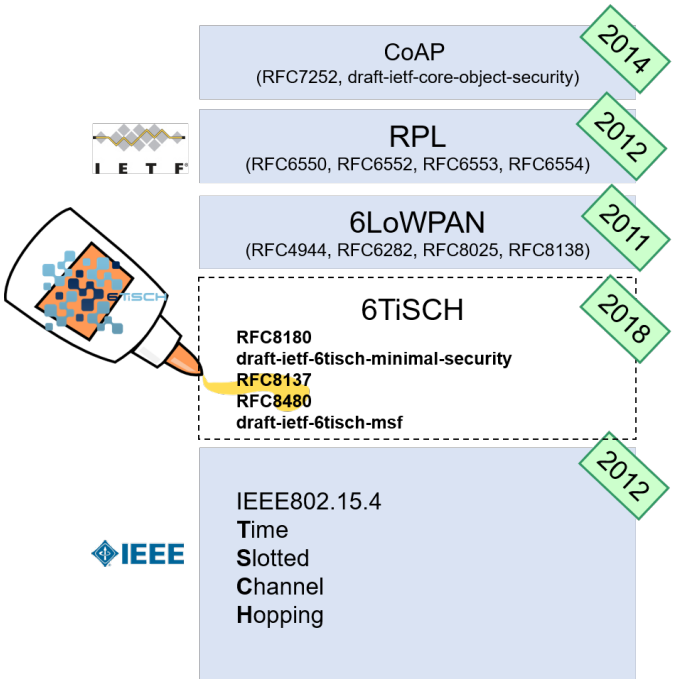


Fig. 1. The 6TiSCH WG has produced a set of IP-compliant specifications that manage the underlying IEEE 802.15.4 TSCH MAC layer and enable the integration with other IETF solutions targeting Internet-of-Things (IoT) applications.

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 inter-network the subsystems. Going wireless is indispensable 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

X. Vilajosana is with Worldsensing S.L and the Wireless Networks Lab, Universitat Oberta de Catalunya. Catalonia.
T. Watteyne, with Analog Devices and the EVA Team, INRIA, Paris, France
T. Chang, M. Vučinić are with the EVA Team, INRIA, Paris, France
K. Pister is with the University of California Berkeley, Berkeley, CA, USA

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], [5], [6], [7], [8], [9] was developed, all relying on the concept of *Time-slotted Channel Hopping (TSCH)* [10], [11], [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 wire-like reliability, greater than 99.999%, and ultra low power consumption, less than $50\mu A$ on average [13], [14], [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 world-wide in industrial process automation, adding wireless connectivity to meters and sensors.

The traditional Supervisory Control and Data Acquisition (SCADA) systems are evolving towards 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 back-end. 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: inter-operable 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 article gives an overview of the 6TiSCH specifications and the work at the IETF that has been influenced by the 6TiSCH WG activities. The article is organized as follows. Section II discusses the 6TiSCH standardization process and related activities. Section III gives background on the Time-slotted Channel Hopping 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 article.

II. An Open Standard

6TiSCH has been developed by the Internet Engineering Task Force (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 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.

III. Time-slotted Channel Hopping

The Time-slotted Channel Hopping (TSCH) mode was first standardized by IEEE in the IEEE 802.15.4e amendment. It

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

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

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 which 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 which 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 acknowledgement. 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 (1):

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

In (1), 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, Eq. (1) loops through all available physical channels in η subsequent transmissions, resulting in "channel hopping". The resulting frequency diversity was demonstrated to combat multi-path fading and interference, yielding wire-like 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 RFC8180 [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, 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 scope of the IEEE 802.15.4 standard. A challenge for the network is to build a loop-less 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 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 towards 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 Scheduling Function (SF).

A. 6top Protocol (6P)

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 7 commands to `ADD`, `DELETE`, `RELOCATE`, `COUNT`, `LIST`, `SIGNAL`, or

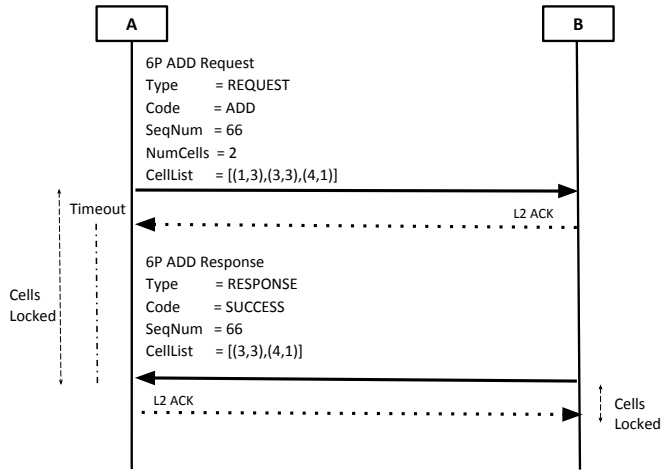


Fig. 2. A 2-step 6P transaction which results in 2 additional cells from A to neighbor B.

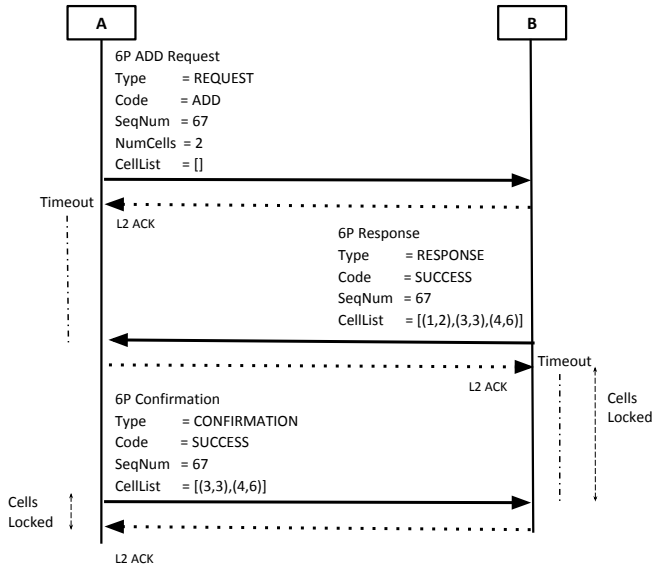


Fig. 3. A 3-step 6P transaction which results in 2 additional cells from A to neighbor B.

`CLEAR` cells in the neighbor's schedule. This set of commands enables the management of a neighbor's schedule. Transactions occur either in a 2-step or a 3-step message exchanges between nodes. Each transaction ends with nodes either committing the transaction or aborting it.

A 2-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 2-step transaction requesting the allocation of 2 cells with Node B. Node A indicates 3 candidate cells that may fit its current schedule so B can select 2 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 3-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 based on the sequence numbers and a set of timeouts. In case such an inconsistency is detected, nodes can roll-back the individual transaction or eventually clear the entire schedule. This decision is left to the particular Scheduling Function.

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 (MSF)

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 a *Scheduling Function (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 Scheduling Function (MSF) [34]. MSF defines 2 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 both for control and application traffic. Because the only input to the hash function is a MAC

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

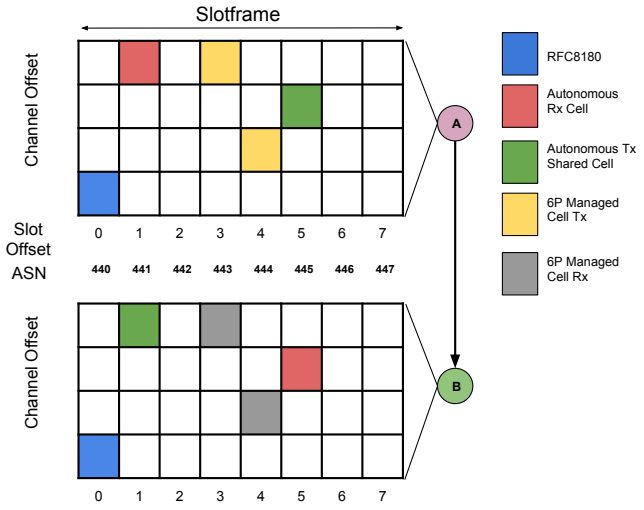


Fig. 4. Example schedule produced by the Minimal Scheduling Function (MSF) between nodes A and B.

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 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. is 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-scope. In the traditional Internet, these problems are solved by a plethora of different specifications. For example, EAP [35] serves as a generic authentication framework. EAP encapsulates EAP methods, such as EAP-PSK [36], EAP-AKA [37], EAP-TLS [38], as the actual implementations of different cryptographic authentication protocols. Then, PANA [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 Authentication, Authorization and Accounting (AAA) decisions in the back-end. With that, 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; and *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 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.

6TiSCH 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.

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

A. Constrained Join Protocol (CoJP)

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 is 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. re-key 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 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]. CoJP hence 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 *pre-shared key (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. As a consequence, 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.

[5] 11 messages for the core DTLS 1.2 authentication with pre-shared keys and 5 messages for EAP signaling [38].

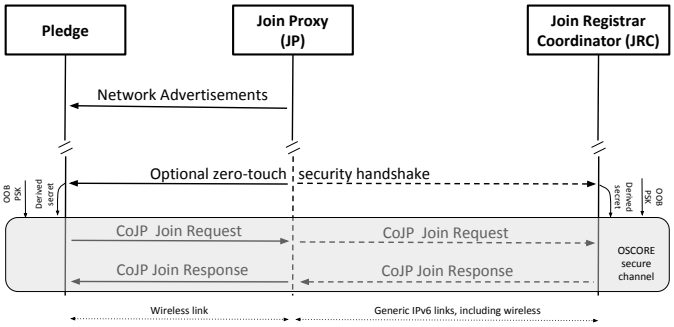


Fig. 5. Exchanges taking place during the join process. Physical links are depicted with solid, logical connections with dashed lines. OOB stands for out-of-band.

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 (OSCORE)

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, that 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.

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 scheduling function, with the adopted candidate being the Minimal Scheduling Function (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 scheduling functions. This facilitates the support for different types of traffic that may be simultaneously present in a network. The numerous proposed scheduling functions 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.

Additionally, 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 Constrained Join Protocol. 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`.

[6] http://www.openwsn.org/

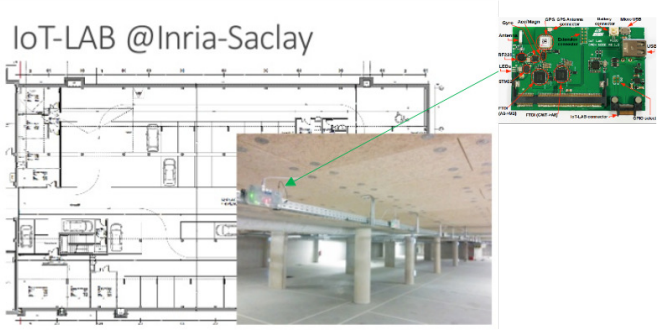


Fig. 6. Measurements are done on a network of 37 "A8" nodes on the Saclay IoT-Lab testbed, deployed in a parking structure.

Please 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 Scheduling Function 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.

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]. As a consequence, 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 towards 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 in

[7] https://www.iot-lab.info/hardware/a8/
[8] https://www.iot-lab.info/

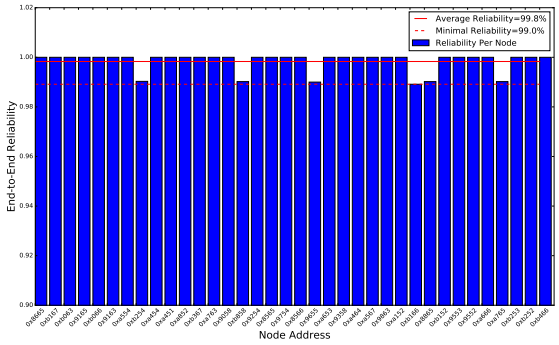


Fig. 7. The average end-to-end reliability of the 36 nodes in the experimentation testbed is 99.8%; any single node has an end-to-end reliability higher than 99%.

Fig. 7 for each of the 36 nodes in the network. The average for the network 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 (2), where H_{num} is the number of hops from a source node to its destination, $offset_{cells}$ 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.

$$L = \sum_{n=1}^{H_{num}} (offset_{rx_tx} + \frac{L_{frame}}{PDR})_n \quad (2)$$

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 I shows 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 5 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 Eq. 2. In this case, we consider that the node can handle the maximum number of retransmissions to a given neighbor within the same slotframe.

TABLE I
LATENCY IN A 5-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

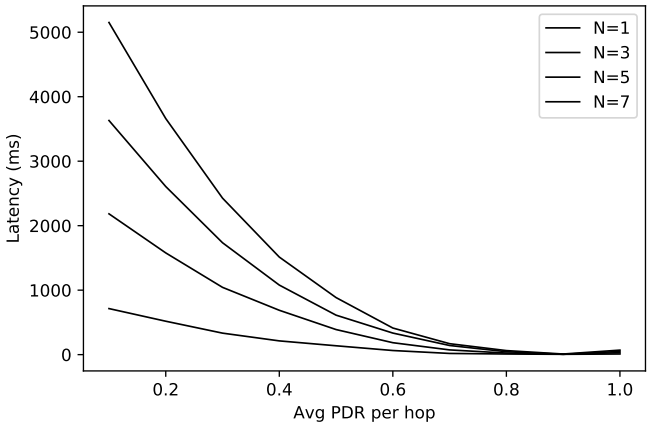


Fig. 8. Computed latency with latency-aware scheduling. Latency is a function of the number of hops N, and the average PDR per hop. These results assume a slotframe length of 101 timeslots, a timeslot of 10 ms, and a maximum of 3 link-layer retransmissions. We consider each node has at least 3 transmit cells and 3 receive cells chained (as per LLSF) from its previous and next hop.

In this case, a node has at least 3 transmit cells to its parent and 3 receive cells from its children. As a consequence, most of the eventual retransmissions can be handled within the same slotframe.

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 towards 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 network-wise energy imbalance and the nodes closer to the root typically consume more energy.

TABLE II
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

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 will be different and dependent on hardware characteristics and implementation optimizations out of the scope of this article. Table II 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 from 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 4 years, and address the main lessons learnt [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 datasets 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 analyse 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 article is an overview of the 6TiSCH standards, developed in an open standardization context by the Internet Engineering Task Force (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 an hyper-connected industry.

ACKNOWLEDGMENT

This work is partially supported by the European Commission through the H2020 F-Interop project, and by the Spanish

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

Ministry of Economy and the FEDER regional development fund through the SINERGIA project (TEC2015-71303-R).

REFERENCES

[1] J. Higgins, "Digital transformation of European industry and enterprises, Strategic Policy Forum on Digital Entrepreneurship," in *European Commission. Ref. Ares(2015)1304723*, March 2015.

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

[3] B. Martínez, C. Cano, and X. Vilajosana, "A square peg in a round hole: The complex path for wireless in the manufacturing industry," *CoRR*, vol. abs/1808.03065, 2018.

[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*, International Society of Automation (ISA) Std. ISA100.11a, 2009. [Online]. Available: https://isa100wci.org/

[6] *Industrial communication networks - Fieldbus specifications - WIA-PA communication network and communication profile*, International Electrotechnical Commission International Standard IEC 62 061, November 29th 2011.

[7] *Industrial networks - Wireless communication network and communication profiles - WIA-FA*, International Electrotechnical Commission International Standard IEC 62 948, July 27 2017.

[8] M. Zheng, W. Liang, H. Yu, and Y. Xiao, "Performance Analysis of the Industrial Wireless Networks Standard: WIA-PA," *Mobile Networks and Applications*, vol. 22, no. 1, pp. 139–150, February 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 Communications & Mobile Computing*, vol. 11, no. 8, pp. 1197–1212, August 2011.

[10] K. S. Pister and L. Doherty, "TSMP: Time Synchronized Mesh Protocol," in *IASTED International Symposium on Distributed Sensor Networks (DSN)*, 2008.

[11] T. Watteyne, A. Mehta, and K. S. Pister, "Reliability Through Frequency Diversity: Why Channel Hopping Makes Sense," in *ACM Symposium on Performance Evaluation of Wireless Ad Hoc, Sensor, and Ubiquitous Networks (PE-WASUN)*. ACM, 2009, pp. 116–123.

[12] L. Doherty, W. Lindsay, and J. Simon, "Channel-Specific Wireless Sensor Network Path Data," in *Conference on Computer Communications and Networks (CCN)*, August 2007.

[13] T. Watteyne, L. Doherty, J. Simon, and K. Pister, "Technical overview of smartmesh ip," in *2013 Seventh International Conference on Innovative Mobile and Internet Services in Ubiquitous Computing*, July 2013, pp. 547–551.

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

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

[16] O. Givehchi, H. Trsek, and J. Jasperneite, "Cloud computing for industrial automation systemsâĂŤa comprehensive overview," in *Emerging Technologies & Factory Automation (ETFA), 2013 IEEE 18th Conference on*. IEEE, 2013, pp. 1–4.

[17] T. Watteyne, V. Handziski, X. Vilajosana, S. Duquennoy, O. Hahm, E. Baccelli, and A. Wolisz, "Industrial Wireless IP-Based Cyber-Physical Systems," *Proceedings of the 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*, Internet Engineering Task Force Std. draft-ietf-6tisch-architecture-14 [work-in-progress], April 2018.

[19] B. M. Leiner, V. G. Cerf, D. D. Clark, R. E. Kahn, L. Kleinrock, D. C. Lynch, J. Postel, L. G. Roberts, and S. Wolff, "A brief history of the Internet," *ACM SIGCOMM Computer Communication Review*, 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*, Internet Engineering Task Force Std. RFC7554, 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*, Internet Engineering Task Force Std. RFC8180, May 2017.

[23] M. Vučinić, T. Watteyne, and X. Vilajosana, "Broadcasting Strategies in 6TiSCH Networks," *Wiley Internet Technology Letters*, November 2017.

[24] T. Winter, P. Thubert, A. Brandt, J. Hui, R. Kelsey, P. Levis, K. S. Pister, R. Struik, J. Vasseur, and R. Alexander, *RPL: IPv6 Routing Protocol for Low-Power and Lossy Networks*, Internet Engineering Task Force Std. RFC6550, March 2012.

[25] T. Kivinen and P. Kinney, *IEEE 802.15.4 Information Element for the IETF*, Internet Engineering Task Force Std. RFC8137, May 2017.

[26] Q. Wang, X. Vilajosana, and T. Watteyne, *6TiSCH Operation Sublayer (6top) Protocol (6P)*, Internet Engineering Task Force Std. RFC8480, November 2018.

[27] T. Chang, T. Watteyne, Q. Wang, and X. Vilajosana, "LLSF: Low Latency Scheduling Function for 6TiSCH Networks," in *International Conference on Distributed Computing in Sensor Systems (DCOSS)*. IEEE, May 2016, pp. 93–95.

[28] M. Domingo-Prieto, T. Chang, X. Vilajosana, and T. Watteyne, "Distributed PID-based Scheduling for 6TiSCH Networks," *IEEE Communications Letters*, March 2016.

[29] E. Municio and S. Latré, "Decentralized Broadcast-based Scheduling for Dense Multi-hop TSCH Networks," in *Workshop on Mobility in the Evolving Internet Architecture (MobiArch)*. New York, NY, USA: ACM, 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 *ACM International Conference on Modeling, Analysis and Simulation of Wireless and Mobile Systems (MSWiM)*. New York, NY, USA: ACM, 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," *International Journal of Distributed Sensor Networks*, 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 *Conference on Embedded Networked Sensor Systems (SenSys)*. Seoul, South Korea: ACM, 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)*, Internet Engineering Task Force Std. draft-ietf-6tisch-msf-01 [work-in-progress], Oct 2018.

[35] J. Vollbrecht, J. D. Carlson, L. Blunk, B. D. Aboba, and H. Levkowetz, *Extensible Authentication Protocol (EAP)*, Internet Engineering Task Force Std. RFC3748, June 2004.

[36] F. Bersani and H. Tschofenig, *The EAP-PSK Protocol: A Pre-Shared Key Extensible Authentication Protocol (EAP) Method*, Internet Engineering Task Force Std. RFC4764, January 2007.

[37] J. Arkko and H. Haverinen, *Extensible Authentication Protocol Method for 3rd Generation Authentication and Key Agreement (EAP-AKA)*, Internet Engineering Task Force Std. RFC4187, Janunary 2006.

[38] D. Simon, R. Hurst, and B. D. Aboba, *The EAP-TLS Authentication Protocol*, Internet Engineering Task Force Std. RFC5216, March 2008.

[39] D. Forsberg, B. Patil, A. E. Yegin, Y. Ohba, and H. Tschofenig, *Protocol for Carrying Authentication for Network Access (PANA)*, Internet Engineering Task Force Std. RFC5191, May 2008.

[40] *IEEE Standard for Local and metropolitan area networks–Port-Based Network Access Control*, IEEE Std. IEEE Std 802.1X-2010, 2010.

[41] A. Rubens, C. Rigney, S. Willens, and W. A. Simpson, *Remote Authentication Dial In User Service (RADIUS)*, Internet Engineering Task Force Std. RFC2865, June 2000.

[42] V. Fajardo, J. Arkko, J. Loughney, and G. Zorn, *Diameter Base Protocol*, Internet Engineering Task Force Std. RFC6733, Oct 2012.

[43] E. Rescorla and T. Dierks, *The Transport Layer Security (TLS) Protocol Version 1.2*, Internet Engineering Task Force Std. RFC5246, August 2008.

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

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

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

[47] M. Richardson, *6tisch Zero-Touch Secure Join protocol*, Internet Engineering Task Force Std. draft-ietf-6tisch-dtsecurity-zerotouch-join-03 [work-in-progress], October 2018.

[48] L. Seitz, F. Palombini, M. Gunnarsson, and G. Selander, *OSCORE profile of the Authentication and Authorization for Constrained Environments Framework*, Internet Engineering Task Force Std. draft-ietf-ace-oscore-profile-02 [work-in-progress], June 2018.

[49] J. Martin, J. Burbank, W. Kasch, and P. D. L. Mills, "Network Time Protocol Version 4: Protocol and Algorithms Specification," 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 *In 34 th Annual Precise Time and 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*, Internet Engineering Task Force Std. draft-vilajosana-6tisch-globaltime-02 [work-in-progress], July 2018.

[52] J. Polk, F. Baker, and M. Dolly, *A Differentiated Services Code Point (DSCP) for Capacity-Admitted Traffic*, Internet Engineering Task Force Std. RFC5865, May 2010.

[53] T. Watteyne, X. Vilajosana, B. Kerkez, F. Chraim, K. Weekly, Q. Wang, S. Glaser, and K. S. Pister, "OpenWSN: a standards-based low-power wireless development environment," *Transactions on Emerging Telecommunications Technologies (ETT)*, vol. 23, no. 5, pp. 480–493, 2012.

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

[55] J. d. Armas, P. Tuset, T. Chang, F. Adelantado, T. Watteyne, and X. Vilajosana, "Determinism through path diversity: Why packet replication makes sense," in *2016 International Conference on Intelligent Networking and Collaborative Systems (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*, Internet Engineering Task Force Std. RFC8138, April 2017.

[57] B. Martinez, X. Vilajosana, I. Kim, J. Zhou, P. Tuset-Peiro, A. Xhafa, D. Poissonnier, and X. Lu, "I3Mote: An Open Development Platform for the Intelligent Industrial Internet," *MDPI Sensors*, vol. 17, no. 5, pp. 1–20, 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 Journal*, vol. 14, no. 2, pp. 482–489, February 2014.

[59] S. Duquennoy, A. Elsts, A. Nahas, and G. Oikonomou, "TSCH and 6TiSCH for Contiki: Challenges, Design and Evaluation," in *International Conference on Distributed Computing in Sensor Systems (DCOSS)*, Ottawa, Canada, June 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 *IEEE International Conference on Computer Communications (INFOCOM)*, Turin, Italy, April 2013.

[61] S. Ziegler, S. Fdida, T. Watteyne, and C. Viho, "F-Interop - Online Conformance, Interoperability and Performance Tests for the IoT," in *Conference on Interoperability in IoT (InterIoT)*, Paris, France, October 2016.

[62] M.-R. Palattella, F. Sismondi, T. Chang, L. Baron, M. Vucinic, P. Modernell, X. Vilajosana, and T. Watteyne, "F-Interop Platform and Tools: Validating IoT Implementations Faster," in *AdHoc-Now 2018 - 17th International Conference on Ad Hoc Networks and Wireless*, Saint Malo, France, Sep. 2018, pp. 1–12. [Online]. Available: https://hal.inria.fr/hal-01858004

[63] M. Esteban, D. Glenn, M. Vučinić, and T. Watteyne, "Simulating 6TiSCH Networks," *Transactions on Emerging Telecommunications Technologies (ETT)*, vol. 29, no. 7, pp. 1–20, 2018.

[64] C. Adjih, E. Baccelli, E. Fleury, G. Harter, N. Mitton, T. Noel, R. Pissard-Gibollet, F. Saint-Marcel, G. Schreiner, J. Vandaele, and T. Watteyne, "FIT IoT-LAB: A Large Scale Open Experimental IoT Testbed," in *World Forum on Internet of Things (WF-IoT)*, December 2015, pp. 459–464.

[65] Z. Brodard, H. Jiang, T. Chang, T. Watteyne, X. Vilajosana, P. Thubert, and G. Texier, "Rover: Poor (but Elegant) Man's Testbed," in *ACM International Symposium on Performance Evaluation of Wireless Ad Hoc, Sensor, and Ubiquitous Networks (PE-WASUN)*, Valletta, Malta, November 2016.

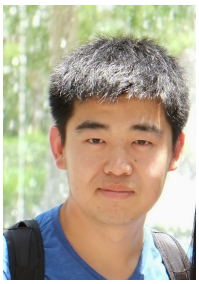
[66] M. Vučinić, M. Pejanović-Djurišić, and T. Watteyne, "SODA: 6TiSCH Open Data Action," in *IEEE Workshop on Benchmarking Cyber-Physical Networks and Systems (CPSBench)*. IEEE, 2018, pp. 42–46.

[67] X. Vilajosana, P. Tuset-Peiro, T. Watteyne, and K. Pister, "OpenMote: Open-Source Prototyping Platform for the Industrial IoT," in *International Conference on Ad Hoc Networks (AdHocNets)*. San Remo, Italy: EAI, September 2015.

Xavier Vilajosana is the principal investigator at the WINE research group at UOC and professor at the Computer Science, Telecommunications, and Multimedia department. In addition, Xavier is a co-founder of Worldsensing, and OpenMote Technologies, 2 prominent startups developing cutting-edge IoT related technology. Until March 2016 Xavier has been a senior researcher at the HP R&D Labs. From January 2012 to January 2014, Xavier was visiting Professor at the University of California Berkeley holding a prestigious Fulbright fellowship. In 2008, he was visiting researcher of France Telecom R&D Labs, Paris. Xavier 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. Xavier is the author of different Internet Drafts and RFCs, as part of his standardization activities for low power industrial networks. Xavier is contributing actively at the IETF 6TiSCH, 6Lo and ROLL Working Groups. Xavier holds more than 20 patents, more than 30 high impact journal publications and have contributed with several demos, tutorials, and courses in the field of low power wireless networks. Finally, Xavier is IEEE Senior Member and founding member and vocal of the IEEE Sensors Council in Spain.

Thomas Watteyne is an insatiable enthusiast of low-power wireless mesh technologies. He holds an advanced research position at Inria in Paris, in the EVA research team, where he designs, models and builds networking solutions based on a variety of Internet-of-Things (IoT) standards. He is Senior Networking Design Engineer at Analog Devices, in the Dust Networks product group, the undisputed leader in supplying low power wireless mesh networks for demanding industrial process automation applications. Since 2013, he co-chairs the IETF 6TiSCH working group, which standardizes how to use IEEE802.15.4e TSCH in IPv6-enabled mesh networks, and is a member of the IETF Internet-of-Things Directorate. Prior to that, Thomas was a postdoctoral research lead in Prof. Kristofer Pister's team at the University of California, Berkeley. He founded and co-leads Berkeley's OpenWSN project, an open-source initiative to promote the use of fully standards-based protocol stacks for the IoT. Between 2005 and 2008, he was a research engineer at France Telecom, Orange Labs. He holds a Ph.D. in Computer Science (2008), an MSc in Networking (2005) and a MEng in Telecommunications (2005) from INSA Lyon, France. He is a Senior member of IEEE. He is fluent in 4 languages.

Mališa Vučinić received the Engineering degree from the University of Montenegro, Podgorica, Montenegro, in 2010, the joint Master's (Hons.) degree 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. He is currently a Researcher Scientist with team EVA of Inria, Paris, France. From 2012 to 2015, he was a Research Engineer with STMicroelectronics, Crolles, and was a Visiting Scholar with the University of California at Berkeley, Berkeley, CA, USA, in 2015. Mališa is active in the IETF and co-authors multiple drafts in different working groups, including 6TiSCH. Main concepts from his PhD dissertation have found way towards 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 is a Postdoc Research Engineer at Inria-EVA, Paris. He obtained his Ph.D. degree in Computer System Architecture at 2017 from University of Science and Technology, Beijing. At 2014, he was visiting at the University of California, Berkeley as visiting scholar. From November 2015 to October 2017, he joined Inria-EVA team as a Pre-Postdoc Research Engineer, leading the project of OpenWSN, which is an Open Source project founded by UC Berkeley. At 2017, he joined the F-Interop project as a Postdoc Research Engineer, which is an H2020 European research project. He is also one of the main implementors of IETF 6TiSCH standard protocol stack. He has worked as technical support for 6TiSCH interoperability plugtest. He has huge interests on Wireless Sensor and Actuator Network, Swarm Robotic and any Embedded System design.

Kris Pister received a B.A. in Applied Physics from UC San Diego, 1986, and an M.S. and Ph.D. in EECS from UC Berkeley in 1989 and 1992. Prior to joining the faculty of EECS in 1996, he taught in the Electrical Engineering Department, UCLA. Professor Pister 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." Professor Pister was award the Alfred F. Sperry Founder Award in 2009 for his "contributions to the science and technology of instrumentation, systems, and automation." Kris is a co-Director of the Berkeley Sensor and Actuator Center (BSAC) and the Ubiquitious Swarm Lab.