Dependable interference-aware time-slotted channel hopping for wireless sensor networks

Dependable Interference-Aware Time-Slotted Channel Hopping for Wireless Sensor Networks

RASOOL TAVAKOLI, MAJID NABI, TWAN BASTEN, and KEES GOOSSENS, Eindhoven University of Technology

IEEE 802.15.4 Time-Slotted Channel Hopping (TSCH) aims to improve communication reliability in Wireless Sensor Networks (WSNs) by reducing the impact of the medium access contention, multipath fading, and blocking of wireless links. While TSCH outperforms single-channel communications, cross-technology interference on the license-free ISM bands may affect the performance of TSCH-based WSNs. For applications such as in-vehicle networks for which interference is dynamic over time, it leads to non-guaranteed reliability of the communications over time. This article proposes an Enhanced version of the TSCH protocol together with a Distributed Channel Sensing technique (ETSCH+DCS) that dynamically detects good quality channels to be used for communication. The quality of channels is extracted using a combination of a central and a distributed channel-quality estimation technique. The central technique uses Non-Intrusive Channel-quality Estimation (NICE) technique that proactively performs energy detections in the idle part of each timeslot at the coordinator of the network. NICE enables ETSCH to follow dynamic interference, while it does not reduce throughput of the network. The distributed channel quality estimation technique is executed by all the nodes in the network, based on their communication history, to detect interference sources that are hidden from the coordinator. We did two sets of lab experiments with controlled interferers and a number of simulations using real-world interference datasets to evaluate ETSCH. Experimental and simulation results show that ETSCH improves reliability of network communications, compared to basic TSCH and the state-of-the-art solution. In some experimental scenarios NICE itself has been able to increase the average packet reception ratio by 22% and shorten the length of burst packet losses by half, compared to the plain TSCH protocol. Further experiments show that DCS can reduce the effect of hidden interference (which is not detectable by NICE) on the packet reception ratio of the affected links by 50%.

CCS Concepts: • **Networks → Network protocol design**; **Network control algorithms**; **Network reliability**; **Wireless personal area networks**;

Additional Key Words and Phrases: IEEE 802.15.4, time-slotted channel hopping, TSCH, in-vehicle wireless sensor networks, interference, dependable, channel quality estimation, wireless co-existence

A preliminary version of this work was published in *Proceedings of the 12th International Conference on Mobile Ad Hoc and Sensor Systems (MASS'15)*.

This work was supported by the DEWI and SCOTT European projects, have received funding from the ARTEMIS Joint Undertaking under grant agreement no. 621353 and the ECSEL Joint Undertaking under grant agreement no. 737422, respectively.

Authors' addresses: R. Tavakoli (Corresponding author) and K. Goossens, Department of Electrical Engineering, Eindhoven University of Technology, P.O. Box 513, 5600 MB Eindhoven, the Netherlands; emails: {r.tavakoli, k.g.w.goossens}@tue.nl; M. Nabi, Department of Electrical Engineering, Eindhoven University of Technology, P.O. Box 513, 5600 MB Eindhoven, the Netherlands and Department of Electrical and Computer Engineering, Isfahan University of Technology, Isfahan 84156-83111, Iran; email: m.nabi@tue.nl; T. Basten, Department of Electrical Engineering, Eindhoven University of Technology, P.O. Box 513, 5600 MB Eindhoven, the Netherlands and TNO Embedded Systems Innovation, P.O. Box 6235, 5600 HE Eindhoven, the Netherlands; email: a.a.basten@tue.nl.

https://doi.org/10.1145/3158231

ACM Reference format:
Rasool Tavakoli, Majid Nabi, Twan Basten, and Kees Goossens. 2017. Dependable Interference-Aware Time-Slotted Channel Hopping for Wireless Sensor Networks. *ACM Trans. Sen. Netw.* 14, 1, Article 3 (January 2018), 35 pages.
https://doi.org/10.1145/3158231

1 INTRODUCTION

Low data rate Wireless Sensor Networks (WSNs) cover a wide range of applications from smart buildings to smart vehicles. This technology is also considered as a solution to be used in new generations of cars. It improves the flexibility and reconfigurability of In-Vehicle Networks (IVNs). Other than common WSN requirements that include reduced complexity of nodes and low power consumption, IVNs require dependable communication due to high Quality-of-Service (QoS) requirements. By definition, a dependable system should guarantee readiness for correct service (availability), continuity of correct service (reliability), and maintainability. Considering these requirements, the Time-Slotted Channel Hopping (TSCH) mode of the IEEE 802.15.4 standard [5] is a proper candidate to be used as the Medium Access Control (MAC) layer protocol for these networks.

TSCH divides time into fixed time periods called *timeslots*. Based on this, TSCH uses a TDMA-based mechanism together with a channel hopping scheme to assign different [timeslot, channel] pairs to each communication link in the network. Using this technique, TSCH provides availability of service by guaranteeing the access of wireless nodes to the medium. It also increases the reliability of the links against persistent multi-path fading and interference. There is a considerable work done on increasing performance and end-to-end reliability of low-power TSCH mesh networks. This includes works on configuring and improving performance of TSCH itself such as References [8, 25, 30, 36, 37], and also works that consider the layers on top of TSCH such as References [12, 13, 18, 23, 24, 34]. However, to meet the stringent requirements of industrial applications such as sub-second latency and reliable communications in in-vehicle networks, more work needs to be done [38].

IEEE 802.15.4 [5] defines 16 frequency channels in the license-free 2.4GHz ISM band. This band is also used by other standards including IEEE 802.11 Wi-Fi [2] and IEEE 802.15.1 Bluetooth [1]. The common usage of this band leads to cross-technology interference and packet losses, especially for the WSNs that use low power communication. The authors of Reference [32] categorize interference in in-vehicle environments into interference of in-car and out-of-car sources. They show that cross-technology interference of Wi-Fi and Bluetooth devices inside a car affects the performance of in-vehicle TSCH WSNs due to low distance of these interferers to the sensor nodes. It is also shown that out-of-car interference on channels has a highly dynamic behavior over time. This dynamic behavior leads to unstable performance of the TSCH protocol and thus, unstable and non-guaranteed dependability of communications over time.

We proposed an enhanced version of the TSCH protocol (ETSCH) in Reference Tavakoli et al. [31]. ETSCH uses dynamic channel whitelisting, based on the results of a Non-intrusive Channel-quality Estimation (NICE) technique. NICE performs frequent channel samplings without applying any change to the protocol. ETSCH also uses a secondary hopping sequence list to increase the transmission reliability of the configuration packets that are transmitted by the coordinator to setup the network. These techniques improve reliability of the TSCH protocol by adaptively selecting a subset of low-noise channels for hopping.

ETSCH is centralized and is mainly done by the coordinator of the network. Considering that interference conditions at each node only affect packet reception at that node, this technique is

mainly preventing interference for packet receptions at the coordinator. The interference conditions at other nodes of the network may be completely different on some channels. Accordingly, the contributions of the current article compared to our previous work [31] are as follows: (1) In addition to the lab experiments of Reference [31], performance of the ETSCH technique is evaluated under realistic in-vehicle scenarios, using real-world interference datasets. (2) A Distributed Channel Sensing (DCS) technique is proposed to estimate quality of the employed channels in all the nodes and collect these quality assessments at the coordinator. The results are combined with NICE results to select a list of low-noise channels over the network area. (3) The DCS technique requires transmission of dummy packets in the dedicated timeslots in which no data packet is provided by the upper layers to be transmitted. This may seem to impose energy consumption overhead, but we show that this technique leads to a lower average energy consumption for TSCH networks with default timeslot timings, because it allows to reduce idle listening. (4) The ETSCH mechanism together with DCS is implemented on real wireless nodes and a mesh WSN has been deployed in an anechoic chamber to simulate hidden interference at the coordinator. Several experiments with different setups have been performed to evaluate the DCS technique.

The results of the DCS technique is used by ETSCH for channel whitelisting. Because of that, in this article, we present DCS together with ETSCH and call the whole mechanism including all the techniques as ETSCH+DCS. To have a complete and coherent story, we also include the summarized results of the ETSCH evaluation of Reference [31]. The next section gives a short description and overview of the TSCH protocol and an application scenario. Section 3 gives an overview of related work in low-power multi-channel communication. The detailed description of the proposed ETSCH+DCS technique is presented in Section 4. Evaluation setup and performance analysis are given in Section 5. Section 6 concludes.

2 BACKGROUND

2.1 Time-Slotted Channel Hopping

TSCH is defined as one of the MAC operating modes of the IEEE 802.15.4 standard [5] to support industrial applications. It increases dependability of communications against external interference and persistent multi-path fading. TSCH divides time into fixed time periods called *timeslots*. One timeslot is defined as a period of time to transmit a single packet and its acknowledgement. Each link in the network is assigned to one of these timeslots to avoid collisions. A number of timeslots are grouped into a *slotframe*. Slotframes repeat over time to enable nodes to have periodic access to the medium. TSCH uses a global timeslot counter in the network that is called the *Absolute Sequence Number* (ASN). By use of the ASN and a global *Hopping Sequence List* (HSL), each node computes the operating channel of each timeslot using Equation (1),

$$Channel = HSL[(ASN + Channel\ Offset)\%|HSL|]. \tag{1}$$

$|HSL|$ is the number of channels in the HSL. Different *Channel Offsets* can be assigned to different links in the network to enable parallel communications in one timeslot on multiple channels. The HSL may include all or a subset of channels defined by the IEEE 802.15.4 PHY to be determined by the upper layers in the protocol stack. This protocol improves the availability of the network by providing guaranteed access to the medium for nodes and reducing intra-network collisions. It also improves reliability of the communications by hopping over multiple channels and eliminating blocking of wireless links (repeated dropping of packets due to interference on the operating channel).

TSCH defines the Enhanced Beacon (EB) that is an extension of the IEEE 802.15.4 beacon frame to construct application-specific beacon content. EB transmission is done by the coordinator(s) of the network and can be periodic and/or aperiodic. It provides a means for application-specific

information provided by higher layer protocols to be included in beacons. These data include the HSL and network timeslot schedule that is provided by the higher layers.

2.2 Application Scenario

In-vehicle wireless sensor networks potentially experience dynamic interference over time. The authors of Reference [32] show that wireless interference in a moving car has a dynamic behavior on each channel over time. This dynamic interference is due to vehicle movement that causes adjacency with different wireless devices that operate in different parts of the 2.4GHz ISM band. Due to non-guaranteed communications of WSNs in such conditions, these networks are mainly considered to be used for non-safety-critical applications in a car. Tire pressure monitoring, controlling windows, rear and front lights, monitoring the engine sensors, and monitoring seat belts are some of these applications. Though these applications are non-safety critical, each one still requires a level of QoS (e.g., latency) due to their real-time behavior. These QoS requirements demand the need for dependable wireless protocols and techniques.

Number of sensors and size of the network can be different from vehicle to vehicle. For a four-seat car, all the sensor nodes can be in range of each other and there is no need for multi-hop communication. A star or mesh network can cover the needs of such cars. For larger vehicles such as trucks, the number of sensors and the network area is larger than a normal passenger car. An important task of in-vehicle WSNs for these vehicles is to provide dependable multi-hop communication to meet the QoS requirements of different applications. Different sensors may experience different interference conditions in some frequency channels. For example, a tire pressure sensor that is placed on a front wheel may experience interference from another wireless sensor in a neighboring car. Due to the aluminium skin of the car, this interference may be invisible to a wireless sensor that is inside the dashboard of the car or another tire pressure sensor that is placed on an opposite wheel. As another example, a sensor at the back side of a truck may experience interference in some frequency channels from a car behind the truck, while the coordinator, which is placed in the front part of the truck, experiences interference on different channels from a car in front of the truck. These kinds of situations lead to a non-uniform distribution of interference over the wireless medium for different sensors.

3 RELATED WORK

The idea of channel hopping is used by a number of protocols and standards including IEEE 802.15.1 Bluetooth [1], WirelessHART [3], ISA100.11a [4], and TSCH which is one of the MAC operation modes of the IEEE 802.15.4 [5] standard. All of these protocols use a time-slotted approach to schedule network communications. At the start of each timeslot, which is synchronized in all network devices, each device hops to a new channel by use of a predefined hopping sequence pattern.

Exploiting a channel hopping technique reduces the probability of not being able to communicate; communication problems for a particular channel may be caused by narrow-band interference and multipath fading [26, 37]. Furthermore, channel hopping improves network performance through higher reliability and better network connectivity compared to a single-channel approach [37]. The TSCH protocol leaves a number of configuration tasks to the higher layers. This includes the scheduling task and selecting the list of channels for hopping. These two tasks have direct impact on the performance of the communications and reliability of the network. The IETF 6TiSCH [12] working group allocates the scheduling functions to the 6TiSCH Operation Sublayer (6top) [36]. Accordingly, several TSCH scheduling techniques such as in References [6, 9, 10, 22, 23, 25, 33] are proposed for 6TiSCH networks. While several articles address the scheduling task to reduce internal interference in the network, the channel selection for hopping sequence list is not studied

well. In this work, we focus on this task that has a high impact on the reliability of networks with external interference.

One of the enhancements to the channel hopping technique is to limit the used channels only to channels that are known to be of good quality [37]. This technique is known as whitelisting; similarly, blacklisting is the technique to skip using some poor channels. IEEE 802.15.1 Bluetooth [1] uses a technique, known as Adaptive Frequency Hopping (AFH) to reduce the impact of cross-technology interference by using good channels. Actually, the Bluetooth standard specification does not specify how to detect bad or good channels. Therefore, AFH developers should select the most appropriate quality estimation method for each particular solution. Normally two methods are used to perform channel assessment in AFH, namely Received Signal Strength Indication (RSSI) and Packet Error Rate (PER). Because the bandwidth and channel hopping rate of Bluetooth is considerably higher than IEEE 802.15.4 protocol, the quality estimation techniques of Bluetooth AFH do not perform well for IEEE 802.15.4. Thus, we skip those techniques and focus on those techniques that are developed on top of IEEE 802.15.4.

WirelessHART [3] and ISA100.11a [4] are two protocols designed for industrial applications and both use IEEE 802.15.4 radios in the 2.4GHz ISM band. These protocols also have the option to define a user configurable whitelisting feature at a global scope to control the channel hopping pattern. Channel blacklisting/whitelisting does not require cooperation and additional synchronization between interfering networks, and also there is no need for additional hardware. WirelessHART uses channel blacklisting and adds the channels that are affected by consistent interference to the blacklist. Using this technique, the network administrator can totally disable the use of blacklisted channels. ISA100.11a uses the history of communications on each link and based on these statistics, the devices will stop utilizing channels that are noisy for a particular period of time by adding them to a blacklist. Watteyne et al. [37] show that use of a static whitelist/blacklist can improve performance of a TSCH network in nearly static wireless conditions. They try different channel whitelist sizes for their trace-based simulations and find that a whitelist with size of 6 reduces the average ETX by 63% compared to blind channel hopping using all 16 channels. Also, the authors conclude that the IEEE 802.15.4 channels 11, 15, 20, and 26 are much less interfered with by IEEE 802.11 [2] (Wi-Fi). This is because Wi-Fi networks usually use the IEEE 802.11 channels 1, 6, and 11 that are not overlapped with those four channels of the IEEE 802.15.4 standard.

For environments such as vehicles that experience a high level of interference dynamism [19, 27, 29, 32], there should be an agile mechanism defined to perform channel quality estimation and HSL selection. Li et al. [21] propose an adaptive channel selection scheme based on the multi-arm bandit problem [16]. The selection of each channel is formulated as an independent process using packet transmission status (packet acknowledgement status) and Clear Channel Assessment (CCA) failures on that channel. In the proposed scheme, the channel selection is done on the transmitter side of each link. The channel list is transmitted to the coordinator by adding it to the information element of the TSCH packet. The coordinator broadcasts the new list of channels accordingly. Simulation results show that the algorithm is able to track existing interference on a channel in about 20 packet transmissions. Actually, this work does not specify the central channel selection method. Furthermore, packet transmission status is only available for ack-enabled transmissions, while real-time and multi-cast communications do not use acknowledgements, and it is the local interference at the receiver(s) side that affect the communications.

Gomes et al. [17] propose the Multi-hop And Blacklist-based Optimized TSCH protocol (MABO-TSCH). This technique uses the multi-armed bandit optimization for channel quality estimation at each node, using packet delivery ratio experienced by that node. The normal data packets and their acknowledgement are used to disseminate local blacklists to neighbors for negotiation process. Experimental results, with a 40-node indoor network, show that MABO-TSCH outperforms the

default blind frequency hopping with a 23% higher throughput. This technique is suitable for large-scale TSCH networks in which each node has a few neighbors and negotiation on a local blacklist does not require a lot of communication between neighbors. Therefore, it is not efficient to be used for the small and dense networks that we target in this article.

Using solutions that only use the history of communications may not work well in dynamic environments. It is because they needs prior communications on each channel to gain enough knowledge about its condition. However, it would be impossible to detect interference condition changes on a channel after it is added to the blacklist. This will reduce the performance of these techniques when the interference conditions frequently changes on each channel. Elsts et al. [14] propose an adaptive channel selection technique that uses a combination of central whitelisting and distributed blacklisting. The authors define two types of nodes, i.e., upstream and downstream nodes. The upstream node, which is the coordinator of the network, performs frequent RSSI samplings and provides a channel whitelist without noisy channels. This HSL is broadcast to the network using EBs. Each downstream node extracts a blacklist based on the packet delivery ratio of its transmitted packets. Every time a downstream node wants to send data, it uses the channels that are in the central whitelist but not in the local blacklist. This technique uses the whole Tx offset of timeslots for RSSI samplings and therefore it may detect internal interference in the network as external interference. However, the RSSI procedure on different channels is not specified. This technique may also leads to an empty or very small channel hopping list at some nodes, if local blacklist at each node and the central whitelist share a lot of common channels. This technique also makes the number of available channel offsets unpredictable.

To cope with dynamic wireless medium conditions, Du et al. [11] propose the Adaptive Time-Slotted Channel Hopping (ATSCH), a dynamic whitelisting/blacklisting mechanism using hardware-based Energy Detections (EDs) introduced in the protocol. An ED is an estimate of the received signal power within the bandwidth of a channel over $128\mu s$. ATSCH works on top of the TSCH protocol and reserves two timeslots of each TSCH slotframe to perform energy samplings on the operating channel of these timeslots. There will be no communications in these timeslots; therefore, the gathered values of energy samplings can be considered as noise levels on those channels. These sampling results are used to assign a quality factor to each channel, thus each channel can be ranked according to its wireless conditions. A fixed-size subset of the best quality channels is selected periodically as the HSL for the TSCH protocol.

Our proposed ETSCH+DCS mechanism uses hardware-based EDs together with transmission logs to measure the quality of channels and select the best subset of channels as HSL. Although ATSCH and ETSCH both use the same philosophy of hardware-based channel sampling methods, there are several advances in ETSCH. (1) ATSCH reserves two timeslots of a TSCH slotframe, which results in a throughput cost to the network. ETSCH does not use transmission parts of timeslots and thus does not reduce the capacity of the network and requires no change to the TSCH schedule and protocol. (2) The rate of sampling in ATSCH is two samples per slotframe and is directly affected by the size of the slotframe. In contrast, ETSCH introduces the NICE technique to perform energy samplings at least two times per timeslot. It thus has a sampling rate that is at least $|Slotframe|$ times higher than ATSCH. This makes ETSCH perform better in highly dynamic wireless conditions. (3) ATSCH uses all 16 channels to broadcast EB periodically (containing the HSL), which may result in EB losses and synchronization loss between nodes. We propose a new method to broadcast EBs in a TSCH network using a secondary HSL. This technique uses a small and less dynamic hopping sequence list that contains the best-quality channels. By using this secondary hopping list for EB transmissions, the probability of EB losses is reduced. Furthermore, Jeon et al. [20] propose a technique to adaptively change the frequency of EDs in ETSCH based

on the interference dynamics. This reduces the energy consumption of ETSCH when power is a constraint at the coordinator node.

We also employ a distributed channel quality estimation technique called DCS, using packet reception and CCA logs, to detect interference at the position of non-coordinator nodes of the network. Thus, it considers noise at the point of other nodes of the network, which may be invisible at the coordinator node. This is while other blacklisting/whitelisting techniques that use history of communications at the coordinator (such as in Reference [21]) cannot detect and mitigate the existing interference at non-coordinator nodes of the network. Furthermore, ETSCH uses hardware-based samples to update the assigned quality to a blacklisted channel. In communication-based channel quality estimation techniques, it is impossible to detect channel condition changes after a channel is blacklisted.

In this article, we consider networks, such as wireless IVNs, that experience highly dynamic and non-uniform interference over time and channels [32]. This interference can be caused by in-range Wi-Fi networks along the road, as well as other wireless IVNs working in adjacent cars on the road. These different interference sources work on different channels and each one may be visible only to a subset of the wireless nodes for a short period of time. Therefore, an IVN experiences dynamic interference over time on different parts of the frequency band and network area. On the other hand, in these small networks, almost all nodes are in the communication range of each other. This allows the coordinator to broadcast synchronization packets directly to all the nodes. However, data communications in the network can be done through multi-hop links.

4 ENHANCED TIME-SLOTTED CHANNEL HOPPING WITH DISTRIBUTED CHANNEL SENSING

In this section, we describe all the components of ETSCH together with the DCS technique in detail. ETSCH components include NICE, HSL whitelisting, and EB hopping Sequence List (EBSL) whitelisting. We start with a brief overview of the functionality of all components and their relation, and then we describe each component in detail.

4.1 Overview

The basic idea of ETSCH is to adaptively select a subset of low-noise channels called the *whitelist* and use it as an input for the channel hopping algorithm. Centralized whitelisting performs well for networks in which all the nodes are in the communication range of the coordinator. Data links can be established between any couple of nodes following a mesh topology using the whitelisted channels. ETSCH adds three components to the basic TSCH protocol at the coordinator node. Figure 1 shows the placement of these techniques together with DCS technique within the protocol stack at the coordinator node, while Figure 2 shows their occurrence within the TSCH slotframe and timeslot structure. It should be considered that all of these components are very lightweight and run in the idle part of timeslots, and thus their execution has no impact on the protocol. Algorithm 1 shows the process of each of these components. In addition, each wireless node in the network has a DCS component to sense the channel conditions at that node and report them to the coordinator.

As Figure 1 shows, NICE runs in parallel with TSCH on the MAC layer of the coordinator to extract the quality of all available channels. NICE uses the EDs introduced in the protocol to measure the quality of each frequency channel. An ED is an estimate of the received signal power within the bandwidth of a channel and takes eight symbol periods (i.e., 128μs). This value is bounded with a minimum ($ED^{Min} = 0$) and a maximum (ED^{Max}) value (platform dependent) that is linearly mapped to the received power in dB, as described in the IEEE 802.15.4 [5] standard document. The first part of Algorithm 1 shows the process of the NICE technique (lines 1 to 9). NICE uses

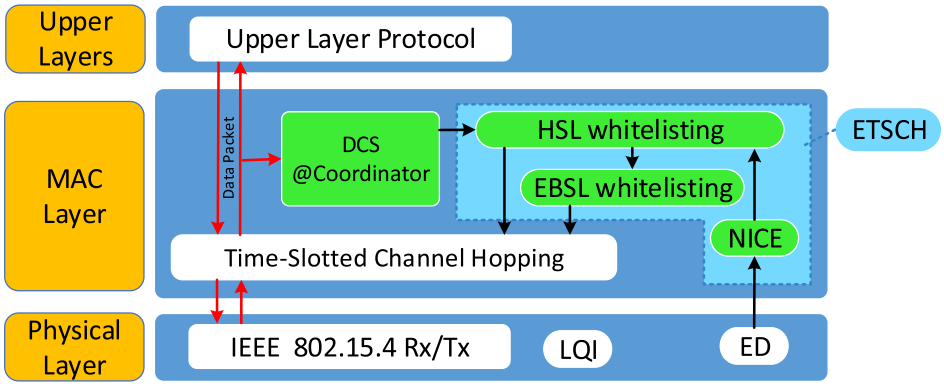


Fig. 1. ETSCH+DCS components in the coordinator node.

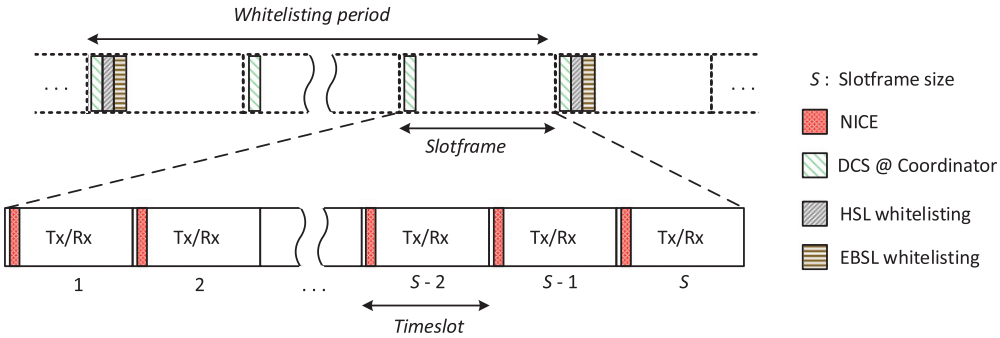


Fig. 2. Occurrence of ETSCH components within the TSCH slotframe and timeslot structure in the coordinator node.

the *silent period* in every coordinator timeslot to perform as many EDs on different channels as possible. Based on the timeslot diagram of the TSCH protocol and considering the fact that the coordinator is the time source of the network, this silent period is only available at the coordinator of the network. EDs are performed on successive channels and after 16 EDs, all the channels are sampled. The result of each ED is used to update the assigned Channel-Quality Estimation (CQE) to that channel. Figure 2 shows the occurrence of NICE in the idle part of each timeslot.

NICE provides centralized interference detection for ETSCH. It does not address the challenge that an interferer that is hidden from the coordinator may cause interference for some of the nodes in the network on some channels. To address this problem, it is necessary to employ a distributed channel quality estimation technique together with NICE. As we show below that the mentioned silent periods are only available at the coordinator of the network, NICE cannot be used in other nodes to perform EDs and extract the quality of channels. We use CCA and packet reception status as two channel quality estimators in other nodes of the network. By using these techniques, each node declares each channel as faulty or non-faulty and includes the results in the data packets that it sends to the coordinator. Because the status of each channel is a binary value, transmitting the status of all 16 channels leads only to a 2Byte overhead. At the coordinator, the channels' status field is extracted from all the incoming packets and is collected by the "Distributed Sensing @Coordinator" component (shown in Figure 1). As Figure 2 depicts, at the beginning of each slotframe the collected data will be analyzed. Accordingly, the assigned CQE to each channel

ALGORITHM 1: ETSCH+DCS Components

Data:
CQE []: an array to store Channel Quality Estimation results of all channels
HSL []: an array to store the main Hopping Sequence List, to be used by TSCH
EBSL []: an array to store Enhanced Beacon hopping Sequence List, to be used by TSCH

```
1  NICE (CQE [])
2      every timeslot do
3          while it is the silent period do
4              ch ← (ch + 1)%16;
5              energy_ level ← ED (ch);
6              CQE [ch] ← EWMAFilter(energy_ level); /* see EWMA Filter in Equation (6) */
7          end
8      end
9  end
```

```
10 DCS@Coordinator(HSL, CQE [])
       Input: PKT(node_id): Data packet received from device node_id
       Data: CC []: an array to store Channel Conditions that are received from different nodes
11     foreach PKT(node_id) received by coordinator do
12         CC [node_id] ← extract channels_condition field from PKT(node_id);
13     end
14     every Slotframe period do
15         foreach channel ch in HSL do
16             CC_avrg [ch] ← average of all channel conditions recorded in CC [] for channel ch;
17             CQE [ch] ← EWMAFilter(CC_avrg [ch]); /* see EWMA Filter in Equation (14) */
18         end
19         Clear CC[];
20     end
21 end
```

```
22 HSL_whitelisting(CQE []), |HSL|)
       Input: |HSL|: size of Hopping Sequence List
       Output: HSL []
23     every whitelisting period do
24         HSL_sorted ← Ascending sort of channels base on CQE [];
25         HSL ← HSL_sorted [1 to |HSL|];
26     end
27 end
```

```
28 EBSL_whitelisting(EBSL, HSL, k )
       Input: k: the EBSL entry which was used for the last EB transmission
       Output: EBSL []
29     every whitelisting period do
30         if EBSL [k] ∉ HSL [0 to 3] ∧ EBSL [k] ≠ 26 then
31             m = min{h | 0 ≤ h ≤ 3 ∧ HSL [h] ∉ EBSL};
32             EBSL [k] ← HSL [m];
33         end
34         k ← the EBSL entry which is used for EB transmission in this timeslot;
35     end
36 end
```

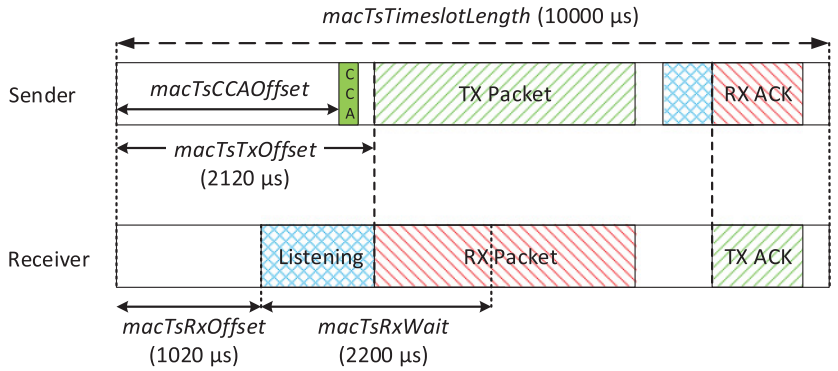


Fig. 3. The structure of transmit and receive timeslots in IEEE 802.15.4 TSCH mode.

is updated. The process of this technique at the coordinator node is shown in lines 10 to 21 of Algorithm 1. This DCS technique is presented in detail in Section 4.3.

The output of NICE and the DCS technique, as a single CQE array, is used periodically by the HSL whitelisting component that runs in the MAC layer at the coordinator node (lines 22 to 36 of Algorithm 1). The HSL whitelisting component selects a subset of best-quality channels for TSCH, based on the observed wireless conditions. According to the results of HSL whitelisting, EBSL whitelisting selects a subset of best channels, in a less dynamic way compared to the HSL whitelisting, to be used for EB transmissions. HSL and EBSL whitelisting components are performed periodically once every a couple of slotframes, called the whitelisting period. Figure 2 shows that these two components are executed at the beginning of the first slotframe of each whitelisting period, after execution of the DCS component. The results of HSL and EBSL whitelisting is included into the EB packets and broadcast to the other nodes in the first slot of each slotframe. In the following, we present all these components in details.

4.2 Non-Intrusive Channel-quality Estimation

To perform an ED in a frequency channel to estimate its noise level, there should be no transmissions in the network itself during that measurement. We propose NICE to perform the EDs on different frequency channels at coordinator and without any bandwidth cost to the protocol in Ref. [31]. Here we use the same examples given in Ref. [31] to describe the functionality of this technique. In the following, we first look at the TSCH communication diagram and then explain our NICE technique.

The TSCH technique of IEEE 802.15.4 uses synchronized timeslots to establish communication between pairs of nodes. A receiver node should be aware of the start of the sender's timeslot to turn on its radio and listen to the medium before transmission starts. Because of clock drift between the nodes, the synchronization process needs to be continuously performed to keep the nodes synchronized. To compensate an amount of timeslot phase differences caused by clock drifts, TSCH defines a diagram for timeslots shown in Figure 3. The timeslot duration, *macTsTimeslotLength*, is long enough for transmission of a maximum size packet and its ACK. There is an offset (*macTsRxOffset*) at the beginning of a receiver's timeslot before it starts listening to the medium. This *Rx* offset prevents interference from other nodes in the network that are behind for a maximum *macTsRxOffset* and still are transmitting in the previous timeslot. The packet *Tx* offset (*macTsTxOffset*) in a sender is defined with a value greater than the *Rx* offset to make the communication possible when the sender is ahead of the receiver for a *macTsTxOffset*−*macTsRxOffset* period of time. A

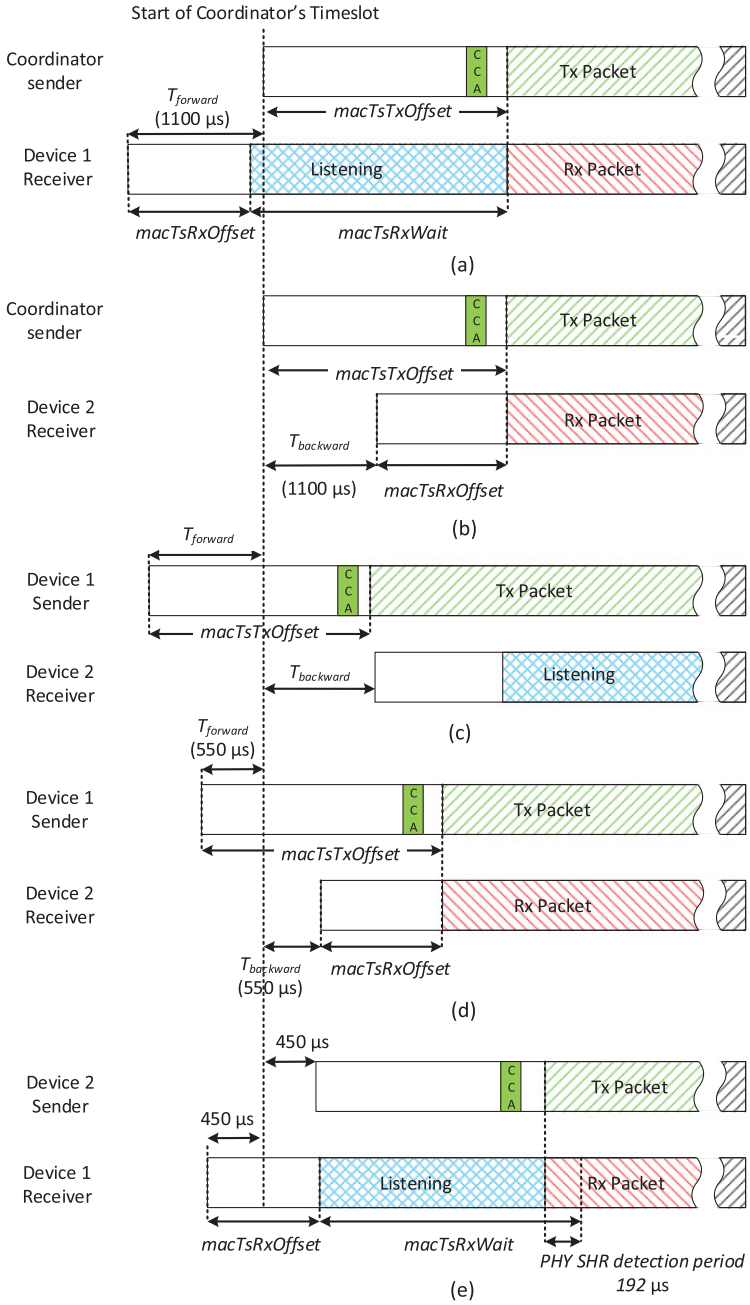


Fig. 4. Pairwise communications in the case of timeslot phase difference; (a) Device 1 starts $T_{forward}$ = 1,100μs ahead the coordinator, (b) Device 2 starts $T_{backward}$ = 1,100μs later than the coordinator, (c) Communication of device 1 with $T_{forward}$ = 1,100μs and device 2 with $T_{backward}$ = 1,100μs fails, (d) successful communication of devices 1 and 2 with $T_{forward} = T_{backward}$ = 550μs, and (e) successful communication of devices 1 and 2 with $T_{forward} = T_{backward}$ = 450μs, considering the required period for PHY Synchronization Header (SHR) detection at the receiver. This figure is the updated version of Figure 4 of Reference [31] with new default timeslot timings defined in the IEEE 802.15.4 standard document.

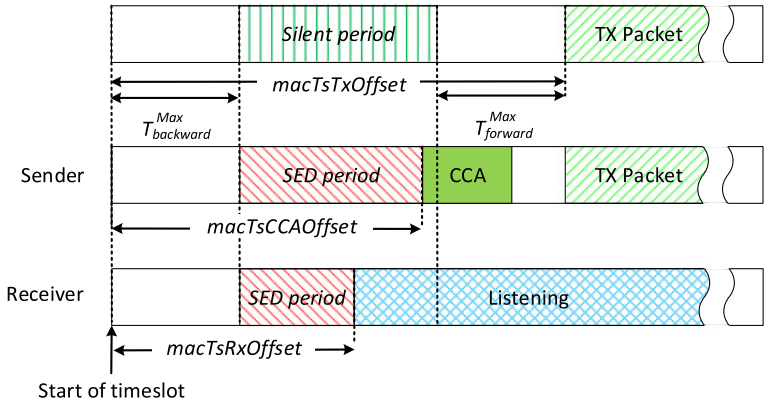


Fig. 5. Available time for Silent Energy Detection (SED) when coordinator is a sender or a receiver.

macTsCCAOffset offset is defined for a sender to perform a Clear Channel Assessment before each *Tx* and prevent packet transmission if the channel is busy. When a receiver starts to listen to the medium for a packet reception in a timeslot, it waits for a *macTsRxWait* period of time to receive the packet. If the transceiver cannot detect any packet preamble in this period, then the receiver considers this situation as a packet transmission failure and stops listening. The values of these parameters are defined in such a way that *macTsRxOffset* + *macTsRxWait* is greater than *macTsTxOffset*. Thus, the communication can be successful if the receiver is ahead of the sender for maximally the difference of these two values. Some other timings such as *Tx/Rx* durations and *ACK* transmission timings are defined in the protocol but not shown in Figure 3.

To extract the maximum allowed phase difference for default values of the mentioned offsets, we investigate different cases. As illustrated in Figure 4(a), if a receiver starts its timeslot $T_{forward}$ = 1,100μs earlier than the coordinator, it still can receive the packet from the coordinator. Also if a receiver starts its timeslot $T_{backward}$ = 1,100μs later than the coordinator, the two nodes can still communicate (Figure 4(b)). Considering the fact that the coordinator of a wireless network is the main source of synchronization, there is no chance for a device that starts $T_{forward}$ = 1,100μs before the coordinator to communicate with a device that starts $T_{backward}$ = 1,100μs after the coordinator (Figure 4(c)). To enable bidirectional transmission between each pair of nodes in the network, as shown in Figure 4(d) and (e), the forward and backward timeslot phase differences should be less than $T^{Max}_{backward} = T^{Max}_{forward}$ = 450μs. This value is extracted by considering the required time for PHY SHR detection at receiver, which should happen before the end of the listening period at receiver (Figure 4(e)). It means that the employed timeslot synchronization method should guarantee the synchronization loss to be less than these values to have a connected mesh network.

Each device may start its timeslot at maximum $T^{Max}_{forward}$ earlier or $T^{Max}_{backward}$ later than the coordinator. Therefore, from the coordinator perspective, for a $T^{Max}_{backward}$ time period at the start of each timeslot, there is a possibility of packet transmissions by some nodes in the previous timeslot. Also for a $T^{Max}_{forward}$ time period before *macTsTxOffset*, there is the possibility that some nodes start packet transmissions ahead of the coordinator. Considering these possibilities, there will be no packet transmission in the network for a T_{silent} period (Figure 5) that can be computed by Equation (2),

$$T_{silent} = macTsTxOffset - T^{Max}_{backward} - T^{Max}_{forward}. \quad (2)$$

For the timing defaults of the TSCH protocol, this value is T_{silent} = 1,220μs.

The reference time to declare the silent period is the start of the coordinator's timeslot. Thus the coordinator of the network knows the exact start and end point of this period over time. On the other hand, because of the allowed timeslot phase difference between network nodes and the coordinator, non-coordinator nodes cannot have an exact estimation about the start of the coordinator's timeslot. It makes it impossible to determine the silent period at those nodes. To perform an ED in a network and have an estimation about the noise level of the channels, there should be no transmissions in the network during these measurements. Therefore, we use this silent period in each timeslot to perform EDs but only on the coordinator device.

A wireless device can be receiver, transmitter, or an idle node during a timeslot. According to this, EDs during the silent period in the coordinator device can be divided into three types. If the coordinator is a receiver, then it should finish the ED process within the *macTsRxOffset* period. The overlap of this period and the silent period can be used for the ED process. When the coordinator is a transmitter, this period will be the overlap of *macTsCCAOffset* and the silent period. The whole silent period can be used for performing EDs when the coordinator has no *Rx/Tx* task. Figure 5 shows these Silent Energy Detection (SED) periods. The available ED duration for each type of timeslot transmission can be computed as Equations (3), (4), and (5),

$$T_{SED}^{Rx} = \min\left(macTsRxOffset, macTsTxOffset - T_{forward}^{Max}\right) - T_{backward}^{Max}, \tag{3}$$

$$T_{SED}^{Tx} = \min\left(macTsCCAOffset, macTsTxOffset - T_{forward}^{Max}\right) - T_{backward}^{Max}, \tag{4}$$

$$T_{SED}^{Idle} = T_{silent}. \tag{5}$$

According to the TSCH protocol defaults, these ED periods will be $T_{SED}^{Rx} = 570\mu$s, $T_{SED}^{Tx} = T_{SED}^{Idle} = 1220\mu$s. As mentioned before, each ED takes eight symbol periods and is the mean of 128 consequent measurements of the signal strength, each lasting for 1μs. To hop to the desired channel for performing EDs and also get the ED measurements from hardware and performing the quality estimation evaluations, we assume this time to be more than twice as high, namely $T_{ED} = 280\mu$s (the value observed in our experiments is less). Therefore, a coordinator can perform two EDs in receiving timeslots and three to four EDs in transmitting and idle timeslots. Each ED will be done in the channel next to the channel for which the prior ED was done.

Considering the default length of a timeslot as 10ms and the least number of possible EDs per timeslot to be two, the minimum sampling rate will be 200 samples per second. Considering the 16 available channels of IEEE 802.15.4 at the 2.4GHz band, each channel will be sampled about 12 times per second. This sampling rate is independent of the slotframe size. Furthermore NICE imposes no throughput cost to the protocol, i.e., it is non-intrusive.

Results of the EDs (higher values shows higher noise on the channel) are used to assign CQE values to each of the channels. To compute a stable estimate of the channel quality, as the ED measurements may fluctuate, we use an exponential smoothing technique [15]. This technique is also used by ATSCH to compute CQE values. Every time a new ED is done in a channel, a new CQE is calculated for it using Equation (6),

$$CQE_\tau(ch) = \alpha(ED^{max} - ED_\tau(ch)) + (1-\alpha)CQE_{\tau-1}(ch), \tag{6}$$

where $ED_\tau(ch)$ is the new ED sample of channel ch and $CQE_{\tau-1}(ch)$ is the last computed CQE for that channel. Higher values of $ED_\tau(ch)$ shows higher noise on a channel. Thus, we subtract this value from the maximum ED value (ED^{max}) so that low noise on a channel (i.e., low $ED_\tau(ch)$ values) leads to high $CQE_\tau(ch)$ values. Coefficient α, with $0 \le \alpha \le 1$, is the smoothing coefficient that controls the effect of new ED samples on the CQE_τ. By selecting small values of α, we obtain

stable estimation of channel quality values. However, very small values impose a delay in detection of changes in the quality of a channel.

4.3 Distributed Channel Sensing

Due to synchronization loss caused by clock drifts, it is impossible to determine the silent period at non-coordinator nodes. Thus, we cannot use NICE to extract quality of channels in non-coordinator nodes. However a third hop interferer, which is hidden from the coordinator, may generate interference for some of the other nodes. Therefore, we employ a distributed channel quality estimation technique called DCS to work together with NICE as the interference detection block of ETSCH.

The overhead of such a sensing technique should be taken into consideration. Using channel EDs at non-coordinator nodes (as ATSCH [11]) leads to extra power consumption that is a negative point in battery-powered wireless nodes. We take advantage of CCA and packet reception status, which are already available and defined in the protocol, as two parameters representing conditions of each communicating channel. All of these parameters give an estimation about the channel quality at the point of the measuring/receiving node, which provides enough information about the existence of interference at each node. The only limitation of using these parameters is that they only give information about the condition of the channels that are being used for communications. In other words, the condition of blacklisted channels is not extractable using these parameters. However, this is not an issue in ETSCH, since when a channel is detected as a bad-quality channel and is blacklisted, NICE still measures its quality and updates the assigned CQE to it. Therefore, if the coordinator realizes that the quality of a blacklisted channel is better than a used channel, there is a chance for that channel to be used for communications again.

The CCA is already defined in the IEEE 802.15.4 standard [5] as a part of the transmission diagram in transmitter nodes. Basically, CCA can be done by detecting energy above a threshold (using ED to detect interference from wireless devices from the network) or carrier sense detection (detect interference from sensor nodes in the network or neighbor networks, with the same modulation) or both. Normally, for a WSN that experiences interference from different sources, those modes with detecting energy above a threshold work better. This is because carrier sense detection may miss most of the interference, due to use of different modulations by other networks, while energy detection can detect interference caused by any type of modulation. By using these operation modes, for most of the IEEE 802.15.4-based sensor nodes, it is also possible to read the ED value from the radio after a CCA. In this work, we only consider the CCA result, which is a *Channel_idle* or *Channel_busy*, but the attached ED value can also be taken into account to calculate the quality of the channel. Every time a new CCA is done in node i (before transmitting a packet) in a dedicated slot, a new Channel Quality $CQ_{i,\tau}$ is calculated for that channel. We skip CCA in Tx slots with link option *shared* enabled. In these slots, several nodes may try to transmit, and CCA is used mainly to detect and avoid internal collisions. As we want to detect the presence of external interferences, we only use the results of CCA performed in the dedicated slots in which only one node may transmit.

As for NICE, we use exponential smoothing to calculate the new channel quality $CQ_{i,\tau}$ based on its previous value $CQ_{i,\tau-1}$, but because CCA results in a Boolean value, we use maximum and minimum values of $CQ_{i,\tau}$ as the input data. Equation (7) shows the used formula,

$$CQ_{i,\tau}(ch) = \begin{cases} \alpha' CQ^{Max} + (1-\alpha')CQ_{i,\tau-1}(ch) & \text{if CCA returns } \textit{Channel\_idle} \\ \beta' CQ^{Min} + (1-\beta')CQ_{i,\tau-1}(ch) & \text{if CCA returns } \textit{Channel\_busy} \end{cases}. \tag{7}$$

Here, CQ^{Max} and CQ^{Min} are two constants that are set to upper and lower values of $CQ_{i,\tau}$, i.e., 255 and 0, respectively. Coefficient α' and β', with $0 \leq \alpha', \beta' \leq 1$, controls the effect of a *Channel_idle*

or *Channel_busy* on the computed channel quality, respectively. We use two different values to be able to give different weights to positive or negative CCA results. Lower values of each coefficient reduce the effect of a new CCA status on the $CQ_{i,\tau}$ and thus makes the quality estimation more stable over time. On the other hand, higher values makes the $CQ_{i,\tau}$ more adaptive to new conditions. The relation between these two coefficients shows the expected reliability of communications on each channel. When $\alpha' < \beta'$, $CQ_{i,\tau}$ is more sensitive to *channel_busy* samples compared to *channel_idle* samples. Thus, to have higher $CQ_{i,\tau}$ values for a channel, the probability of *channel_busy* occurrence should be lower than *channel_idle* occurrence. This guarantees a high communication reliability for channels with a high $CQ_{i,\tau}$. On the other hand, when $\alpha' > \beta'$, even channels with higher occurrence of *channel_busy* compared to *channel_idle* may get high $CQ_{i,\tau}$ values. This leads to possible use of un-reliable channels for communications when channels with high $CQ_{i,\tau}$ values are used for communications.

Packet reception status shows the success or failure of an expected packet transmission at the receiver side, without considering acknowledgement of that packet. It is different from the packet delivery status, which can only be extracted if acknowledgement is enabled. To use the packet reception status in our quality estimation technique, we should be sure that in each dedicated timeslot, there is at least one packet transmission. Otherwise, a failure in packet reception may mean a failure due to interference or skipping transmission by the defined transmitter. We refer a dedicated timeslot to a timeslot with link option *Tx* enabled and link option *Shared* disabled. We assume that even if there is no data to be transmitted by a source node on a dedicated Tx timeslot, it should send a keep alive or dummy packet to the assigned destination. As we exclude the Tx slots with the *shared* option enabled (these slot can be used for overprovision, association commands and other sudden traffic), this technique does not affect the functionality of shared timeslots. It should be considered that in a real schedule the number of dedicated slots should be in line with the required bandwidth by the application. Thus, if a mature slot scheduling mechanism is in place, all dedicated timeslots contain real data transmission in most slotframes. Therefore, transmission of the dummy packets are rarely required.

Considering the default timeslot timings that are specified in the standard, we claim that if dummy packets contain no payload, and also use the shortest possible MAC header, they apply no power overhead to the network. Based on the IEEE 802.15.4 standard [5], the PHY header is 6Bytes and the minimum MAC header which contains frame control, PAN identifier, destination address, source address, and frame check sum fields is 9Bytes. As defined in the IEEE 802.15.4 standard [5], the receiver nodes in a dedicated timeslot starts listening at $macTsRxOffset = 1{,}020\mu s$ and waits for $macTsRxWait = 2{,}200\mu s$ to receive the frame preamble. Considering the transmission offset which is $macTsTxOffset = 2{,}120\mu s$, the receiver continues listening for extra $(macTsRxOffset + macTsRxWait) - macTsTxOffset = 1{,}100\mu s$ from the expected Rx start time, to receive the packet (Figure 3). This is due to the possibility of synchronization loss between the transmitter and receiver. Therefore, if a transmitter has no data to transmit in a dedicated timeslot and skips the transmission, the intended receivers waste a lot of power in the listening phase.

Here we estimate the energy consumption in both cases of transmitting or not transmitting a dummy packet. If T_{Rx} represents the duration of the listening period at the receiver and T_{Tx} represents the duration of a packet transmission, then the overall energy consumption (E) of each transmission is

$$E = \Big((I_{Rx}T_{Rx}) + (I_{Tx}T_{Tx})\Big) \times V_{cc}, \tag{8}$$

where I_{Rx} and I_{Tx} stand for the radio transceiver current in receive and transmit modes, respectively. Based on the symbol timing definitions in the standard, transmitting each Byte takes $32\mu s$. Thus transmitting a dummy packet with PHY packet size of 15Bytes takes $15 \times 32 = 480\mu s$. By

considering the average synchronization loss between transmitter and receiver as 0s, the receiver keeps the radio ON for only T_{Tx} time after start of transmission. Considering the fact that for low-power wireless sensor nodes, the radio current in Rx mode is almost equal or even higher than the radio current in Tx mode [35] (e.g., Atmel ATmega256RFR2 wireless microcontroller [7]), the overall energy that is consumed for transmission of a dummy packet can be computed as

$$\begin{aligned} E_{dummy} &= \Big((I_{Rx} \times (480[\mu s] + (macTsTxOffset - macTsRxOffset)) + (I_{Tx} \times 480[\mu s])\Big) \times V_{cc} \\ &\approx \Big(I_{Rx} \times (480 + 1100 + 480)[\mu s]\Big) \times V_{cc} \\ &= 2060[\mu s] \times I_{Rx} \times V_{cc}. \end{aligned} \tag{9}$$

If the transmitter skips packet transmission at a dedicated timeslot, then the overall energy consumption will be

$$\begin{aligned} E_{No\text{-}Tx} &= \Big((I_{Rx} \times macTsRxWait) + (I_{Tx} \times 0)\Big) \times V_{cc} \\ &= 2200[\mu s] \times I_{Rx} \times V_{cc}. \end{aligned} \tag{10}$$

The results of Equations (9) and (10) show that the overall energy consumption (sum of energy consumed at transmitter and receiver) when we transfer a short dummy packet is less than the case that no transmission takes place. This energy consumption overhead is even more visible when there are multiple receivers listening to receive a packet from a single transmitter (multicast or broadcast transmissions) that transmits nothing.

These energy consumption calculations are for default timeslot timings defined in the IEEE 802.15.4 [5] standard document. For other timeslot timings, transmitting a dummy packet saves energy if $E_{dummy} < E_{No\text{-}Tx}$. The relation between timeslot offsets is defined in the protocol document as $macTsRxOffset + macTsRxWait/2 = macTsTxOffset$. Having this and Equations (9) and (10), transmitting a dummy packet saves energy when $(macTsTxOffset - macTsRxOffset) > 960$. Actually, this technique also improves the channel quality estimation accuracy of ETSCH and helps prevent packet failures caused by using noisy channels. This can lead to energy saving gain even for shorter timeslot offsets. We suggest transmitting dummy packets when timeslot offsets meet the mentioned condition. Otherwise, it is the user choice to skip packet reception status to be used in DCS technique.

Based on the above discussion, when a timeslot is dedicated for a node to transmit data and there is no data to transmit, we assume that the node transmits a small dummy packet instead. Thus, if a receiver node does not receive a packet in a dedicated timeslot, or there is a check-sum error when it receives a packet, this condition can be considered a *transmission_failure*. Otherwise, the packet reception status will be a *transmission_success*.

After each (expected) packet reception, we use the packet reception status to update the assigned quality to the channel that is used for packet reception. It is also possible to use the Link Quality Indicator (LQI) that is attached to the packet. Actually, LQI shows the communication quality between two nodes on a channel, not only the quality of the channel. Considering this, we only take the packet reception status into consideration. Every time a new packet reception is done in a channel at node i, a new Channel Quality $CQ_{i,\tau}$ is calculated for that channel using Equation (11),

$$CQ_{i,\tau}(ch) = \begin{cases} \alpha' CQ^{Max} + (1-\alpha') CQ_{i,\tau-1}(ch) & \text{if } \textit{successful packet reception} \\ \beta' CQ^{Min} + (1-\beta') CQ_{i,\tau-1}(ch) & \text{if } \textit{packet reception fails} \end{cases}. \tag{11}$$

Here we use the same technique and coefficients that are used for $CQ_{i,\tau}$ calculation in Equation (7). This is because both CCA and packet reception status show the success or failure in using the channel and thus they should affect the $CQ_{i,\tau}$ in the same way. It is also possible to use different

coefficients for Equation (7) and Equation (11), as one of them may show the channel condition better than the other according to the type of interference.

Using the described technique, when a channel is not used in the HSL and there is no transmission on it, its $CQ_{i,\tau}$ does not change at all. When this channel is added again to the HSL, its channel quality value on each node is referred to past and may affect new channel quality calculations. Therefore, on every node i we reset the $CQ_{i,\tau}$ value of each newly used channel in HSL to a predefined constant value CQ^{init}.

Because the coordinator is responsible of defining the HSL and distributing it, the results of the DCS technique at all the nodes should be gathered at the PAN coordinator. We attach this data to the normal data packets that are transmitted from end nodes to the PAN coordinator (unicast and broadcast packets). Transmitting the calculated channel quality of all 16 channels of each node to the PAN coordinator results in a high-throughput overhead for the network. To reduce this overhead, in each node i we use a threshold-based whitelisting and apply it on the $CQ_{i,\tau}$. By using Equation (12), the channels that have a quality greater than a predefined threshold θ, are selected as good channels to be reported to the coordinator,

$$CC_i(ch) = \begin{cases} 0 & \text{if } CQ_{i,\tau}(ch) < \theta \\ 1 & \text{if } CQ_{i,\tau}(ch) > \theta \end{cases}. \tag{12}$$

Here, $CC_i(ch)$ is the whitelisting result (channel condition) of channel ch at node i. Threshold θ should be greater or equal to CQ^{init}. This is because we initially want to consider a newly added channel ch to the HSL, with $CQ_{i,\tau}(ch) = CQ^{init}$, as a good channel.

To transfer channel qualities from end nodes to the coordinator, we can use an array of 16 bits in which each bit refers to the condition of the corresponding channel. This leads to 2Bytes of communication overhead per packet to transmit channel conditions sensed by each node to the coordinator. This 2Bytes can be added to the payload of any layer in the protocol stack, from application to MAC. As we consider one-hop communications in this work, we use 2Bytes of the MAC payload to attach this channel quality data to packets towards the coordinator. Note that this technique is only applicable for the nodes that have traffic towards the coordinator. Moreover, using MAC payload only works if all the data communications towards the coordinator are single-hop. Otherwise, for multi-hop networks, these data should be attached in the upper layers such as network layer. However, the exact attachment mechanism is not specified here and is an implementation decision. By each packet reception at the coordinator from node i, the CC_i report that is attached to the MAC payload is extracted and is added to a report list. Node ID i is also attached to the CC_i value in this list, and there can be only one CC_i connected to a node ID. Thus, old CC_i values of a node will be replaced by new values. All the reports in the report list are periodically used to update the CQE of each channel, and then the list will be cleared. Considering a slotframe as a period in which all the transmissions repeat in the network, we do this process at the beginning of each slotframe. At first, we average the CC_i values of different nodes for each channel to have a global notion of channel conditions in the network area,

$$CC_{avg}(ch) = \frac{\sum_{i=1}^{n} CC_i(ch)}{n}. \tag{13}$$

Here, n refers to the number of CC_i reports in the report list that is the number of nodes that sent a channel condition report to the coordinator during the last slotframe period. After computing the CC_{avrg} of each channel, we map it into the range of EDs and apply it to the CQE of that channel (that is continuously updated by NICE), using a predefined weight γ and its previous value as follows:

$$CQE_{\tau}(ch) = \gamma CC_{avrg}(ch)\,(ED^{Max} - ED^{Min}) + (1-\gamma)CQE_{\tau-1}(ch). \tag{14}$$

This CQE_{τ} value is used as the input of the whitelisting algorithm. Accordingly, DCS results are combined with NICE results. Thus, when a channel is detected as a bad-quality channel (due to results of DCS technique) and is blacklisted, NICE still measures its quality and updates the CQE assigned to it. Therefore, after some time the blacklisted channels get a chance to again be used for communications to check if the observed interference by DCS is gone. This only happens if NICE realizes that the quality of a blacklisted channel is better than a channel under use. Especially, this is crucial for the networks with dynamic interference conditions to re-introducing channels to the whitelist. In the following, we discuss about the central whitelisting technique that is used at the coordinator node.

4.4 Channel Whitelisting

Whitelisting is performed periodically by the coordinator of the network to select a subset of good-quality channels as the HSL for the TSCH protocol (lines 22 to 27 of Algorithm 1). The results of NICE and DCS, which are combined as a unique CQE array, is used as the input of this algorithm.

There are different approaches to do whitelisting, from threshold-based to ranking-based. To select a proper technique, a few constraints should be taken into account to do whitelisting for a TSCH network. First, if the whitelist size (|HSL|) is not prime to the slotframe size, each timeslot only touches a subset of the channels in the HSL, not all of them. This will cause a non-uniform chance of failures for different timeslots over time, because they may use different subsets of channels with different qualities. Thus, using a variable size whitelist with all possible whitelist sizes may cause problems. Furthermore, if an allocated link to a timeslot experiences persistent multi-path fading on a channel, due to the use of a small subset of channels for communications of that link, its packet error rate will be increased. We use a fixed size whitelist in this work, but, based on the user requirements, it is also possible to use a variable size whitelist with sizes that are prime to the slotframe size.

It should be considered that smaller whitelist sizes reduce the maximum number of channel offsets that can be used in the schedule. If more channel offset is used, then two different links with different channel offsets but same timeslot offset use the same channel for communications. This cause internal interference and packet losses. Accordingly, smaller whitelist sizes reduce the number of parallel communications that can be established at one timeslot and also the overall throughput of a TSCH network. In threshold-based approaches, the number of channels with a better quality than a specified threshold may be very low. Thus, it may not provide enough channels to meet a specific whitelist size. On the other hand, by using the NICE and DCS techniques, we assigned qualities to all of the channels and we can use these values to sort them. Therefore, we use a ranking technique to select a fixed number of best quality channels as the HSL for ETSCH. As IEEE 802.15.4 [5] standard suggests to use a pseudo-randomly shuffled set of all of the available channels for HSL, techniques such as the one presented by Shih et al. [28] can be used to shuffle the list of channels without regeneration overhead. The resulting shuffled HSL is used by the TSCH protocol for the hopping procedure.

4.5 EB Whitelisting

The coordinator device of the ETSCH network periodically uses whitelisting to extract the best HSL. The HSL and other information of the network such as link allocations and ASN are disseminated via the EBs defined in the TSCH protocol, in line with, e.g., IEEE 802.15.4 [5]. We set up the coordinator to broadcast EBs periodically in the first timeslot of the slotframe with highest priority. Periodic transmission of EBs helps all devices in the network to synchronize with their coordinator at the start of each slotframe and to be aware of changes in the network setups.

When a coordinator broadcasts an EB with an updated HSL, there is the possibility of missing this EB in one or more devices. Using unicast and ACK-enabled communications for transmitting EBs comes at a throughput cost to the network. The work in Reference [37] shows that some of the IEEE 802.15.4 channels are affected less than the others by coexisting Wi-Fi networks which are the main source of interference on IEEE 802.15.4 channels (i.e., channels 11, 15, 20, and 26). Thus we decide to use a second, less dynamic, hopping sequence list consisting of a small subset of best quality channels to disseminate EBs in ETSCH. The EBSL is defined by the coordinator and has a fixed size of 4. This size is in line with the fact that four 802.15.4 channels typically are less affected by coexisting Wi-Fi networks, compared to others [37]. We do not limit the EBSL to only those four channels, though, to ensure that always the best channels are selected. Therefore, the operating channel to transmit an EB for a given ASN and size of the slotframe (S) can be computed as

$$Channel(EB) = EBSL[\lfloor ASN/S \rfloor \% 4]. \tag{15}$$

We update this EBSL in a one-channel-per-period manner every time the main HSL is updated. In this method, every time the coordinator wants to broadcast an EB containing an update of the main HSL, it only updates the EBSL entry that was used for the last EB transmission (k). The process of updating this list is described in lines 28 to 36 of Algorithm 1. This algorithm finds the channel with the best quality, which is not in the EBSL, and then puts this channel in the last used entry of the EBSL. This updating method reduces the possibility of burst EB losses in a joined device by only using best-quality channels. Hence, when a device misses an EB that contains an updated HSL, it has a high chance to receive it in the later slotframes and synchronize its HSL to the network.

Timeslot phase difference caused by clock drift between a device and the coordinator can lead to disconnection of the link between them. This leads to burst EB losses even when the EBSL is the same at both. To solve this problem, we take channel 26, which is a non-overlapping channel with Wi-Fi, as a permanent member of the EBSL. This channel is considered to be the least-noisy channel in urban environments. Every time a joined device experiences a burst EB loss equal to a predefined number N_{BL}, it considers this situation as a synchronization loss caused by timeslot phase difference and starts a passive scan on channel 26 to be synchronized again with its coordinator. Because the size of the EBSL is 4, it is possible for a device to receive an EB on channel 26 after a maximum time of 4 beacon periods, assuming no packet losses. If packet failures happen on EB reception on channel 26, then the joining device should wait for another 4 bacon periods to again have the chance to receive an EB and join the network.

5 PERFORMANCE EVALUATION

In this section, we investigate the performance of ETSCH through various experiments and simulations. We define two evaluation sets to evaluate ETSCH and ETSCH with DCS technique (ETSCH+DCS). In the following, we describe our evaluation setup and continue with the performance evaluation of each setup. We introduced a number of acronyms during description of the proposed techniques in this article. To enhance the understanding of the analysis, Table 1 gives a list of defined acronyms and their descriptions.

5.1 Setup

To evaluate the performance of ETSCH using experiments, we implemented the TSCH protocol on top of the Atmel implementation of IEEE 802.15.4 MAC for ATmega256RFR2 Xplained Pro kit [7]. This kit includes an ATmega256RFR2 chip that integrates an AVR microcontroller and a 2.4GHz RF transceiver. Our implementation follows default TSCH timings, defined in the standard. We also

Table 1. List of Acronyms

Acronym	Description
ETSCH	Enhanced Time-Slotted Channel Hopping
NICE	Non-Intrusive Channel-quality Estimation
HSL	Hopping Sequence List
EBSL	Enhanced Beacon hopping Sequence List
DCS	Distributed Channel Sensing
ETSCH+DCS	ETSCH plus DCS
ETSCH–EBSL	ETSCH minus (without) EBSL
ED	Energy Detection
NG	Noise Generator
PRP	Packet Reception Probability which is extracted by simulations
PRR	Packet Reception Ratio which is extracted by experiments

exploit some controlled noise generators using the same Atmel motes. Because of different channel setups in different wireless standards, in reality an IEEE 802.15.4 network may observe interference on multiple adjacent channels from a coexisting wireless technology (e.g., Wi-Fi). To mimic this real situation in our setup, each noise generator provides controlled interference by transmitting dummy packets on a pair of adjacent IEEE 802.15.4 channels. To implement this mechanism, a noise generator transmits a short packet on a channel and immediately hops to the paired adjacent channel. This process is done continuously to generate interference on both the paired adjacent channels only by one noise generator. Furthermore, each noise generator can be programmed to hop to different pairs of adjacent channels within predefined periods and using a predefined sequence.

The DCS technique is proposed for situations where there is an interference source around the network that is hidden from the coordinator. Actually, when there is no hidden interference source, the DCS technique has no effect on the channel whitelist, and whitelisting only follows the output of NICE. Based on this, we define two evaluation sets to study the performance of (1) the ETSCH technique without any interference hidden from the coordinator and (2) the ETSCH+DCS technique under existence of hidden noise. In the first evaluation set, we compare the performance of ETSCH with ATSCH and basic TSCH using experiments and simulations. The experiments are a summary of the results of Reference [31] in which we used different levels of interference dynamism to evaluate the agility of the available channel sensing techniques. In addition to the lab experiments of Reference [31], performance of the ETSCH technique is evaluated under realistic in-vehicle scenarios, using simulations and real-world interference datasets. For the second evaluation set, we use experiments and focus on the performance evaluation of the DCS technique in the presence of hidden interference that is not detectable by NICE. Because DCS is an additional technique on top of ETSCH, we also perform the same experiments for ETSCH as well as TSCH to compare the results.

5.2 ETSCH Performance Evaluation

This subsection evaluates the performance of ETSCH (skipping the DCS technique) in comparison with other channel quality estimation technique called ATSCH and also the TSCH protocol. As co-channel wireless interference is the main source of packet errors in urban networks, whitelisting can reduce this negative effect by using good-quality channels. The level and also dynamism of interference can affect the performance of the whitelisting technique. To test the performance of ETSCH, we use experiments with controlled noise generators as well as simulations with

real-world in-vehicle interference on IEEE 802.15.4 channels. We try different interference scenarios with different levels and dynamism of interference for the network under test.

We investigated various metrics to evaluate the performance of the proposed techniques. Packet Reception Probability (PRP) is the probability of one successful packet transmission for given noise power during transmission of each bit of that packet in simulations. PRP is a probabilistic value that is extracted based on the signal to noise ratio. On the other hand, Packet Reception Ratio (PRR) is the number of packets that are successfully received at the receiver node over the total number of packets transmitted by the sender node, extracted from the experiments. Both metrics reflect the quality of the links. The length of burst packet losses is the number of consecutive packets losses over a link and shows the time duration of link level disconnections. This metric is important for many applications to avoid long disconnections, as they required continuity of correct service over time. The maximum of this metric shows the worst-case latency of a link in a TSCH network if the number of retransmissions is infinite.

5.2.1 Lab Experiments, Setup, and Analysis. We use a mesh TSCH network with seven devices and one PAN coordinator. Motes are distributed in random places in a 10 m $\times$ 10 m office workspace. We use a wider area for our experiments, compared to an in-vehicle network in a truck. This is because due to obstacles (such as body of the car and passengers) in an in-vehicle testbed, quality of links are normally lower than an office workspace. Instead of obstacles, we use longer distances between motes to reduce the link qualities in the experiments. We ran the experiments using a complete ETSCH with NICE and EBSL, and also a reduced version of it without the EBSL, called ETSCH–EBSL (ETSCH minus EBSL) in this section for ease of reference. ETSCH–EBSL uses the basic hopping sequence list (all 16 channels) to transmit EBs. This allows us to investigate the impact of the EBSL on the performance of the network. We also implemented ATSCH [11] on top of our TSCH implementation to use it for our performance evaluations.

Slotframes of size $S = 11$ are used in the experiments to be prime to the hopping sequence list size of 8. The first timeslot is allocated to EB transmission by the coordinator, 7 other timeslots each is allocated to one of the devices to transmit a packet of 100Bytes, and the three last timeslots are idle. As ATSCH needs two more timeslots per slotframe to perform EDs, we use two of the three idle timeslots for it. Each experiment lasts 6,000 slotframes, and thus each mote broadcasts 6,000 packets in an experiment. All motes listen to all the timeslots for packet reception from other motes. Doing this, we can extract the quality of all available links in the network, as we build a full mesh network. We run a number of experiments with different α values to find a proper value. A value of α around 0.1 was found to have the best results. We use values of a power of 2 for all the used coefficients, so that calculations can be simplified using bitwise shift; this minimizes the processing overhead on the sensor nodes. Thus, $\alpha = 1/8$ is used for experiments. The value of other parameters are used in this experiment is shown in Table 2.

We consider four interference scenarios in our experiments as follows: high, medium, low, and no interference. In the no interference scenario, we run the experiments without any controlled noise generator to see the cost of periodic HSL changes on the performance of our mechanism. Table 3 provides a short description of each scenario.

Considering wireless IVNs as a case study, there are some interference sources along urban roads [32]. An in-vehicle network in a moving car constantly experiences interference from different sources (e.g., Wi-Fi networks). Assuming that each interference source is visible over a range of up to 50m, and this car moves with a speed of 36km/h, each noise source would be visible for 5s. Our high-interference scenario models this kind of interference when there are three interference sources visible at any time, which each generate interference on two IEEE 802.15.4 channels. This scenario simulates the situation that a car moves into the range of a new interference sources (e.g.,

Table 2. Setup for Basic ETSCH Evaluation

Parameter	Value	Description
α	1/8	Exponential smoothing exponent for NICE
Tx power	0dBm	Transmission power of sensor nodes
Packet size	100Bytes	Size of the data packets at MAC layer
$\|HSL\|$	8	Hopping sequence list size
$\|EBSL\|$	4	Enhanced beacon sequence list size
$\|Slotframe\|$	11	slotframe size (number of timeslots per slotframe)
Whitelisting period	10	Whitelisting period in terms of $\|Slotframe\|$
N_{BL}	5	Number of burst EB losses to do re-synchronization

Table 3. Interference Scenarios

Scenario	Behavior of NG(s)
No interference	no controlled NG
Low interference	noise on 2 channels (1 NG), hop every 20s to new channels
Medium interference	noise on 6 channels (3 NGs), hop every 20s to new channels
High interference	noise on 6 channels (3 NGs), hop every 5s to new channels

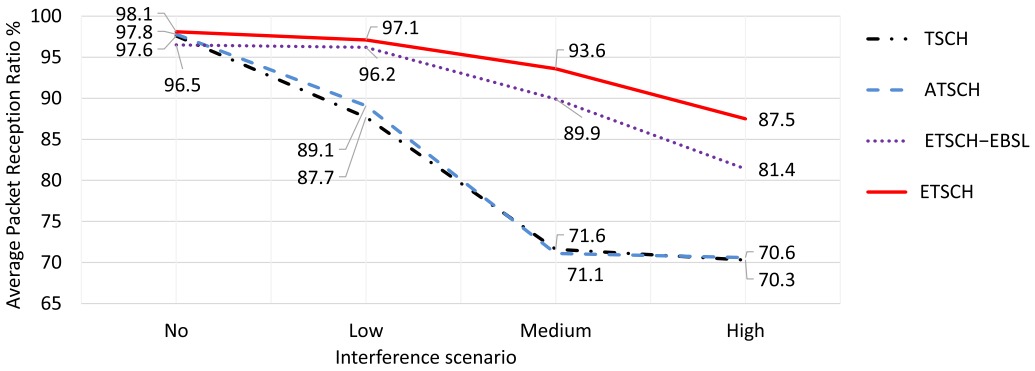


Fig. 6. Average achieved PRR of different mechanisms for different interference scenarios.

a Wi-Fi network) every 5se. For the medium interference scenario, we consider lower mobility of IVNs, which leads to increasing the visibility duration of each interference source. Thus, we have lower dynamism of interference in this scenario. In the following, we analyze the results of this experiment set for different interference scenarios.

Figure 6 shows the average of achieved PRR of all links in the network for different mechanisms and interference scenarios. Both versions of ETSCH provide better PRR on average in comparison to TSCH and ATSCH, when the network experiences dynamic interference. This shows the effect of highly adaptive channel quality estimation that is realized by NICE, which selects the best quality channels for hopping. As depicted in Figure 6, ATSCH performs almost the same as basic TSCH. There are two reasons for these results: (1) The rate of channel samplings by ATSCH is much lower than for ETSCH. Therefore, it can only deal with very low dynamic interference, and it cannot detect and follow the highly dynamic interference (that exists in in-vehicle networks). This leads to increasing packet losses when noisy channels are also selected to be used in the HSL.

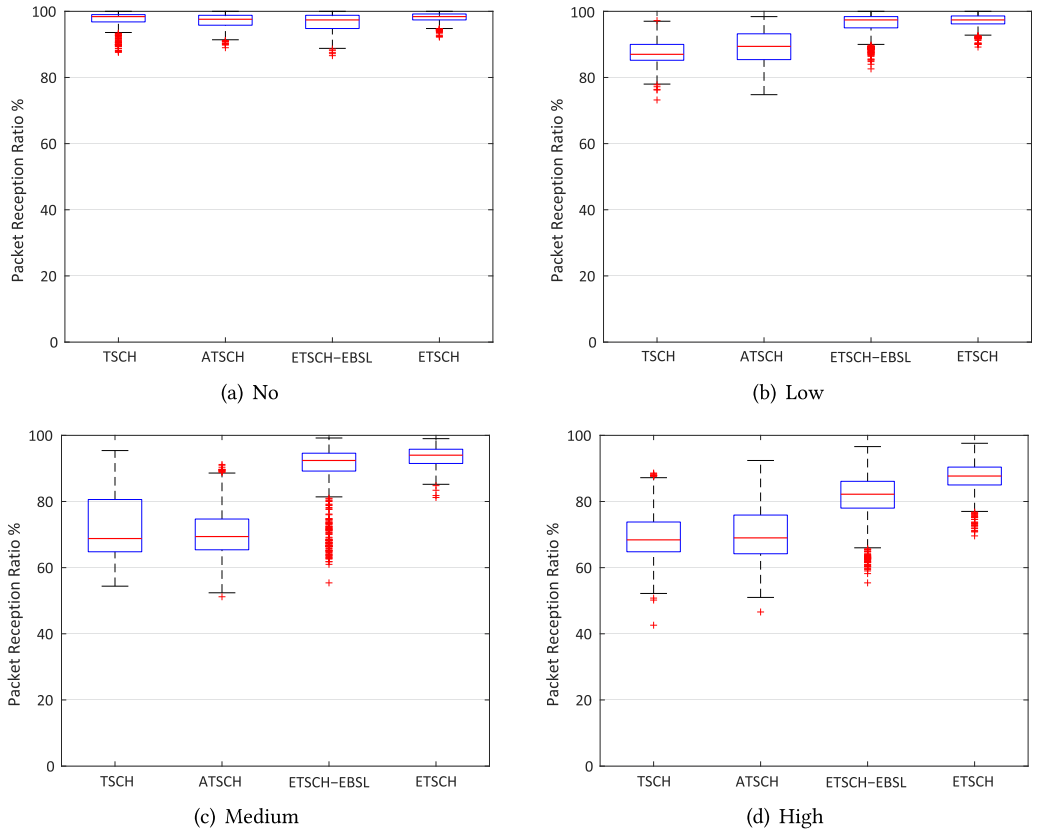


Fig. 7. PRR distribution of all network links, over a window of 500 transmissions, for different mechanisms and different interference scenarios.

(2) ATSCH does all the samplings in one timeslot every slotframe. Our NICE technique spreads channel samplings over a slotframe, and therefore it can detect noisy channels better.

Figure 6 also shows that using an EBSL to disseminate EBs improves the PRR of ETSCH compared to ETSCH–EBSL in all the interference scenarios. This is because it reduces the possibility of EB losses, and, accordingly, HSL mismatches between the coordinator and nodes.

To better investigate the behavior of the different mechanisms, Figure 7 shows the distribution of average PRR of all links in the network over a window of 500 transmissions for the different mechanisms and scenarios. The results show that using an EBSL generally leads to a lower standard deviation in the PRR results. This guarantees a higher reliability level for the links of the network. Figure 7(a) shows that for the scenario with no controlled noise generator, the standard deviation of ETSCH–EBSL results is higher than for TSCH. This is because TSCH always uses the same HSL for hopping, while ETSCH–EBSL may use different sets of channels as the HSL due to detecting low noise on some channels. In the case of missing one EB by a node, which can be due to transient interference or multipath fading, there is a possibility of HSL mismatch between that node and other nodes in the network. This situation leads to packet drops on incoming and outgoing links on this node, which is due to using different channels by the source and destination on each link. It continues until the node detects this synchronization loss and synchronizes its HSL with the coordinator. This synchronization may take a few slotframes and reduces the PRR of those links

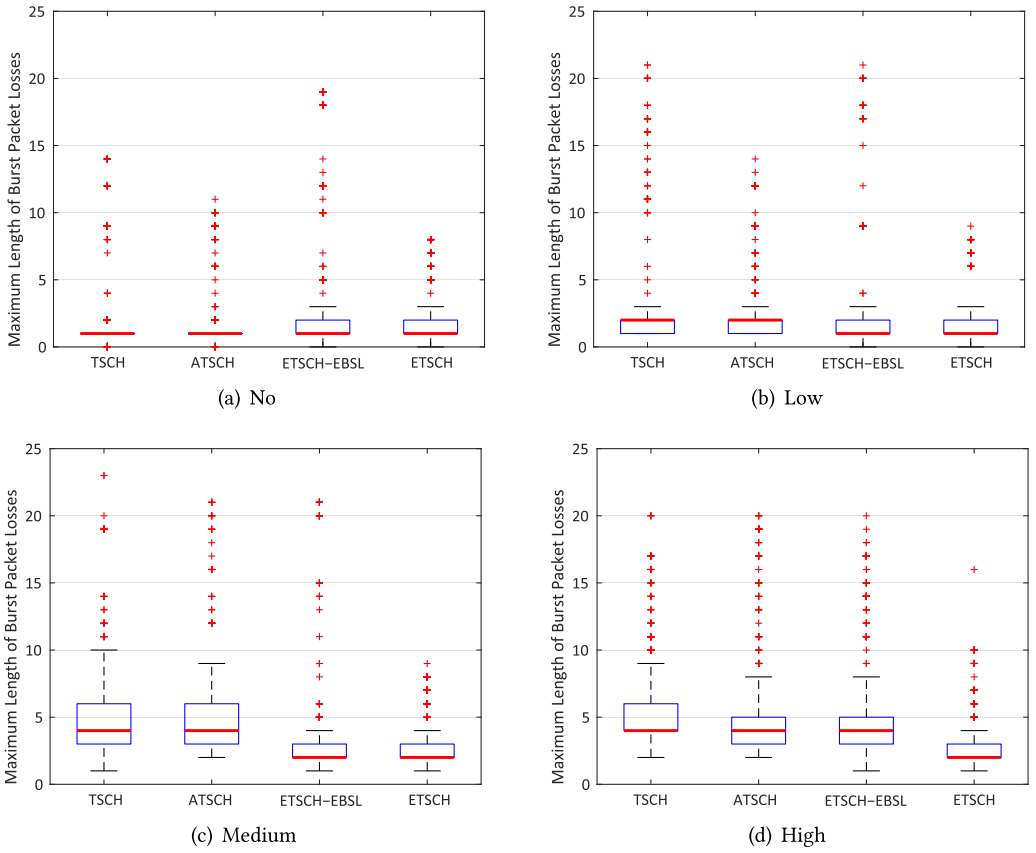


Fig. 8. Maximum length of burst packet losses of all links over a window of 500 transmissions for different mechanisms and different interference scenarios.

for that time window. By using EBSL in ETSCH, after missing one EB by a node, it still has the chance to receive the EB on the next three slotframes and synchronize its HSL with the coordinator. This reduces continuous packet losses on each link and thus provides a higher PRR as well as a lower standard deviation in the results of ETSCH for scenarios with interference.

Figure 8 illustrates the distribution of maximum length of burst packet losses over a window of 500 transmissions, considering all links of the network in each mechanism. This shows the continuity of correct service of all links in the network, in which any link may be a candidate to be used for dedicated communication in the real-world networks. Furthermore, we extract the maximum length of burst packet losses in windows of 500 transmissions. Thus, for each link, we have 12 values of maximum length of burst packet losses (6,000 packet transmissions for each node). This shows the occurrence of communication problems during time on each link and helps to better perceive the performance of communications.

Figure 8(a) shows the measured results for the no-interference scenario. Compared to basic TSCH and ATSCH, for both versions of ETSCH there is a slight increase in the maximum length of the burst packet losses in absence of interference. Due to the higher number of channel samplings by ETSCH, it experiences fast changes of the assigned quality to channels caused by small quality variations. This leads to more frequent HSL changes for ETSCH that lead to a higher chance of

HSL mismatch between each device and coordinator. By considering the outliers in this scenario, which are the bottlenecks for maximum length of burst packet losses, ETSCH still outperforms TSCH and ATSCH.

For the second interference scenario (Figure 8(b)), plain TSCH has maximum burst packet losses with a median length of two. This is because TSCH hops over all channels and there are only two adjacent channels with interference which cause packet losses. ATSCH has no gain compared to the results of TSCH. The reason is the slow channel quality estimation process, which causes some delay in detecting noisy channels and some packets to be lost. Both versions of ETSCH follow the dynamism of interference faster and thus result in smaller median value of burst packet losses. Considering the outliers, ETSCH–EBSL experiences occasional burst packet losses with higher values compared to ATSCH. This is due to more frequent HSL changes that cause HSL mismatches between devices and the coordinator when there is an EB loss. However, ETSCH can reduce these kinds of burst packet losses by using the EBSL synchronizing the HSL in a more reliable way.

Figure 8(c) shows that when the number of noise generators increases to three (six noisy channels), TSCH and ATSCH experience higher median value and a wider distribution of the maximum length of burst packet losses. For TSCH that always uses the same HSL for hopping, having more noisy channels increases the possibility of using consecutive noisy channels for transmissions on a link. For ATSCH that has a slow interference detection process, using noisy channels in the HSL and HSL mismatch between nodes and the coordinator, due to EB losses, is causing bursts. ETSCH–EBSL decreases burst packet losses by performing more channel samplings and thus detecting noisy channels faster than ATSCH. ETSCH has even lower maximum length of burst packet losses with less deviation, because it reduces EB losses that are the source of HSL mismatch between nodes.

For the high-interference scenario (Figure 8(d)), all techniques except ETSCH perform almost the same as plain TSCH. This is because of highly dynamic interference that causes EB losses and problems for HSL synchronization between coordinator and devices. Even in this scenario, ETSCH decreases the maximum length of burst packet losses compared to other mechanisms (for median value, normal distribution, and outliers). This is done by keeping the network nodes synchronized, using best channels to transmit EBs.

5.2.2 Simulations with Real-World Interference Data. We implemented the functionality of ETSCH and ATSCH on top of the Matlab simulation framework that is presented in Reference [32] for the TSCH protocol. The authors of Reference [32] also perform a number of experiments to measure the real-world in-vehicle interference of different scenarios. We use the reported datasets of different scenarios as the input for the Matlab simulator to evaluate the performance of our technique under real-world interferences. Because these datasets show the interference at one point in a car, simulations are only valid for the communications toward one node. Therefore, we consider a star TSCH network with a number of devices and one PAN coordinator where each timeslot is allocated for links from a device to the coordinator. We consider the same distance of 3m between all the devices and the coordinator, path-loss exponent $\gamma = 3.5$, and transmission power $P_{tx} = 0$dBm.

As for the lab experiments, slotframes of size $S = 11$ are used in the simulations. The first timeslot is assumed to be used for EB transmission by the coordinator to broadcast HSL (and EBSL). The other timeslots in each slotframe are dedicated for transmission of packets (with length 100Bytes) by the network nodes to the coordinator. In ATSCH, the last two timeslots of each slotframe are used for performing EDs. Based on the length of available datasets, each simulation lasts for 300s; thus there are 30,000 timeslots in one simulation. Other parameters are configured as in the previous experiments (Table 2).

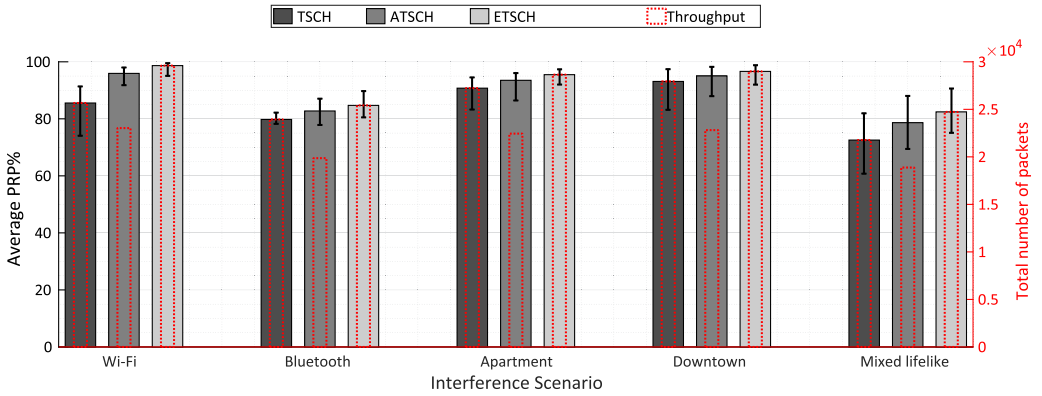


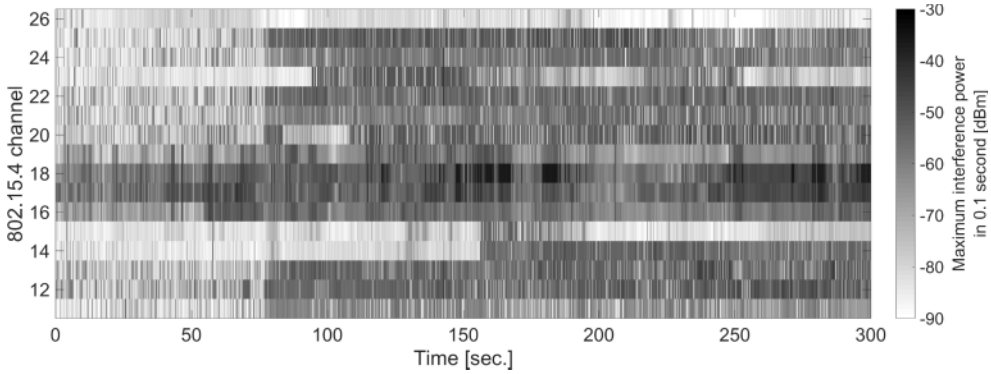
Fig. 9. Average and distribution of PRP over the 300s of simulation together with achieved throughput in term of the total number of timeslots with successful packet transmissions for different mechanisms and different interference scenarios.

The channel offset, which is used for a TSCH slotframe, can affect the PRP of that slotframe. This is due to the possible periodic behavior of a noise signal on a channel. Because a network can define multiple slotframes with different channel offsets, we consider all possible parallel transmissions on different available channels to compute the PRP of each timeslot. Accordingly, for each timeslot, we compute the PRP of transmissions on all available channels in the HSL and then use the average of them as PRP of that timeslot.

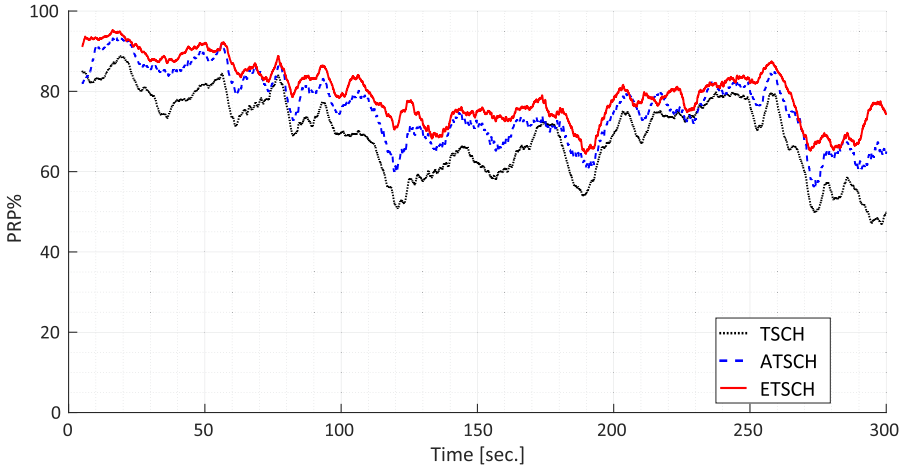
Figure 9 shows the simulation results for different mechanisms and different interference scenarios. Five interference scenarios are considered; i.e., Wi-Fi and Bluetooth file transfer between two devices in a car, driving along a road near some apartments and an office area downtown, and a lifelike scenario that is a mix of the Wi-Fi, Bluetooth, and downtown scenarios. As the results show, ETSCH outperforms ATSCH and TSCH in terms of PRP in all scenarios. For a scenario like Bluetooth, where interference affects almost all the channels [32], using a whitelist is not very effective. This is because interference is distributed almost uniformly over all channels and whitelisted channels perform almost the same as blacklisted channels. For other scenarios, in which the distribution of interference over channels is not uniform, both ATSCH and ETSCH techniques perform better than the plain TSCH. While ETSCH outperforms ATSCH in terms of the average PRP in these scenarios, it also has a lower standard deviation, which shows its continuity of correct service over time. For the Bluetooth scenario, both ETSCH and ATSCH have a higher standard deviation of PRP compared to TSCH. This is because Bluetooth also uses a channel hopping scheme, together with an adaptive channel whitelisting. This causes the coexisting ETSCH network to perform good when whitelists of Bluetooth and ETSCH are not overlapping and to perform bad when whitelists are overlapping.

Figure 9 also depicts the total number of timeslots with successful packet transmissions to give an estimation of the achievable throughput. This value is calculated based on the number of available timeslots for communications and the computed average PRP for each scenario. ATSCH has a lower throughput than the plain TSCH in all scenarios. This is due to the overhead of idle timeslots that are used by ATSCH for channel estimation. On the other hand, ETSCH increases the throughput of the network by increasing the PRP, while NICE has no throughput overhead.

To study the PRP over time, Figure 10(a) shows interference of the lifelike scenario on the IEEE 802.15.4 channels [32]. Figure 10(b) presents the interference impact on the PRP of different mechanisms, extracted by simulations. This figure confirms the results of Figure 9 and also shows that,



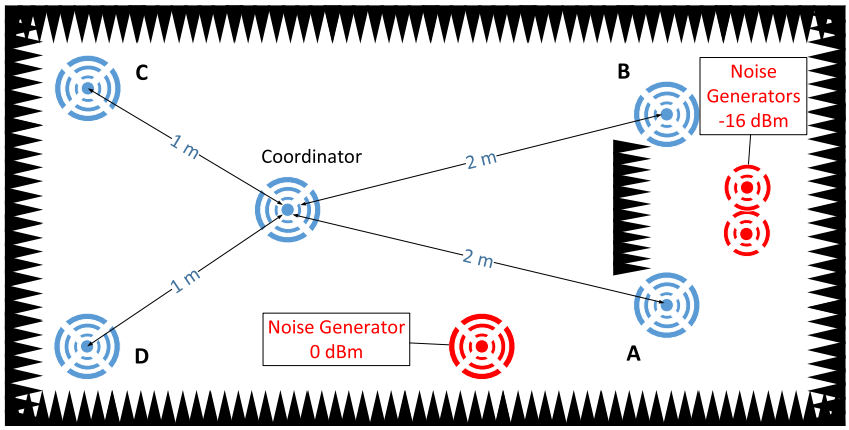
(a) Interference behavior over 300 s using HeatMap plot.



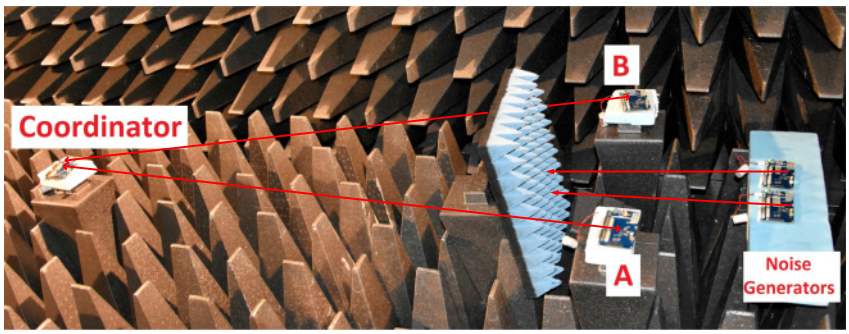
(b) PRP moving average over a window of 500 timeslots, for different mechanisms.

Fig. 10. Effect of the Mixed lifelike interference on the IEEE 802.15.4 channels [32] and performance of different mechanisms under this interference, extracted by simulations.

depending on the interference conditions, PRP may changes a lot over time. As Figure 10(b) shows, ETSCH detects noisy channels at the beginning of the simulation very fast (about 1 to 2s), while it takes about 10s for ATSCH to detect the noisy channels and follow interference. Compared to the plain TSCH, the whitelisting techniques used by ATSCH and ETSCH may provide a considerable increase in average PRP in some periods of time (e.g., time 100s to 160s). This increase in PRP may also be very low in some periods (e.g., time 200s to 250s). This behavior highly depends on the number of noisy channels at each timeslot, which makes whitelisting very effective when the distribution of noise over different channels is not uniform. It is also possible that the used whitelisting technique performs worse than the basic TSCH for short periods of time (e.g., ATSCH at time 230s). This happens when the set of noisy channels changes quickly and overlaps with the selected whitelist. Thus, there will be packet errors until the new noisy channels are detected and a new whitelist is picked. We may conclude that real-world interference has a high impact on the TSCH protocol and techniques such as ETSCH can improve the reliability of this protocol over time.



(a) Node deployment map



(b) Picture taken from part of the actual setup

Fig. 11. Node deployment to evaluates the performance of DCS technique on top of ETSCH.

5.3 ETSCH+DCS Performance Evaluation

In this section, we evaluate the performance of ETSCH when DCS is enabled in the presence of hidden interference from the coordinator. The interference datasets that are provided by Reference [32] are only for a single point in a car. However, for our evaluations in this section, we need interference at multiple points to simulate the hidden interference problem. Therefore, we only use experiments for our evaluations and skip simulations in this section.

5.3.1 Evaluation Setup. We use a mesh TSCH network with four devices and one PAN coordinator for this set of experimental evaluation. Motes are deployed in an anechoic chamber as depicted in Figure 11(a). These experiments are performed in an anechoic chamber to implement a scenario with hidden interference. To do this, we used two noise generators with a low transmission power of −16dBm. As depicted in Figure 11(a), these noise generators are placed close to each other at a distance of 0.5m of two end nodes. A signal absorber wall is placed between the coordinator and these noise generators to prevent a line of sight radiation path between them. Because there is no reflection in the anechoic chamber, the interference will be completely hidden from the coordinator. This placement causes a level of interference for those two nodes that are close to the noise generators, while this interference at the coordinator would be almost invisible. Figure 11(b) shows a part of the actual setup, including the coordinator, nodes A and B, and two

Table 4. Setup for ETSCH Evaluation

Parameter	Value	Description
α'	1/8	Exponent used for increasing CQ
β'	1/4	Exponent used for decreasing CQ
γ	1/8	Weight used to apply CC_{avrg} into CQE
θ	128	CQ threshold used for DCS
CQ^{init}	180	Initial value of CQ for DCS
CQ^{Max}	255	Maximum value of CQ for DCS
CQ^{Min}	0	Minimum value of CQ for DCS
$\|Slotframe\|$	5	slotframe size (number of timeslots per slotframe)

hidden noise generators. Using this scenario, we can study the performance of the DCS technique, while NICE cannot detect this interference at the coordinator point. To study the performance of NICE when DCS is on, we use one noise generator at a transmit power of 0dBm, which is visible to all of the network nodes. All of these noise generators use a period of 20s to hop to new channels to generate dynamic interference.

We ran the experiments for ETSCH with and without DCS. This allows us to investigate the impact of the DCS technique on the performance channel quality estimation while there is noise hidden from the coordinator in the network. To have a complete performance study, we also ran the experiments using the basic TSCH protocol. As in the first experiment set, each experiment lasts for 6,000 slotframes, and thus each node broadcasts 6,000 packets in an experiment. Each node also listens to all other timeslots for packet reception from other nodes.

5.3.2 Parameters Setup. Table 4 shows the used values for all the parameters in this scenario. Like what we did for choosing the value of α in Table 2, we run a number of experiments with different combinations of α' and β' values to find a suitable value for them. Because the optimum values for these parameters depend on the dynamism of the interference, we tried different interference scenarios and picked the values that have the best results for all scenarios. As shown in the table, we pick two different exponents for CQ calculations at each node for the DCS mechanism (α' and β'). The exponent that is used for positive samples (α') has a lower value than the exponent used for negative samples (β'). This is because we want to make negative or channel_busy samples more effective in CQ calculation to detect the noisy channels faster. It prevents long burst transmission failures on a noisy channel by reducing its CQ below the threshold θ after a few subsequent negative channel assessments and reports it to the coordinator as a noisy channel. We consider the value of threshold θ to be the mean of CQ^{Min} and CQ^{Max}. By defining the CQ^{init} to be greater than θ, an initial chance is given to a newly used channel to be considered as a good channel, even if a few transmission failures happen. This is because a newly added channel to the HSL is assumed to be a good (low-noise) channel.

The coordinator determines the quality of each frequency channel (CQE) based on the output of NICE as well as the result of the DCS technique. As stated in Equation (14), a weighted average is applied with γ as the weight for the DCS result. Considering that the results of DCS are mapped to the range of ED, and observing that in our ETSCH experiments the CQE value of each channel was normally in the lowest 1/8 part of the ED range, we pick value of 1/8 for γ. Using this value, the effect of NICE and the DCS technique on the computed CQE of each channel is expected to be almost the same. Other parameters are the same as for the first experiment set (Table 2).

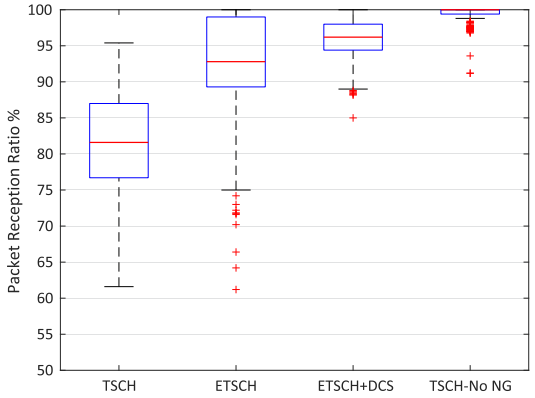


Fig. 12. PRR distribution (of all links in the network) over a window of 500 transmissions for different mechanisms in a setup with hidden interference; TSCH without interference (TSCH-No NG) is given as reference.

5.3.3 Performance Analysis. We investigate PRR to evaluate the performance of the DCS technique. Figure 12 shows the distribution of PRR of all links in the network for different mechanisms. We included one experiment with the TSCH protocol without any Noise Generator (NG). This experiment is done to check if the deployed network works fine and the placement of nodes does not negatively affect the performance of communications between them. A window of 500 transmissions is used to plot the results. As Figure 12 shows, when there is no noise generator in the network area (TSCH-No NG), the median value for PRR is 100%, and all links in the network are perfect. As depicted in this figure, on average ETSCH provides better PRR in comparison with TSCH. This is because ETSCH can detect the visible noise generator and skip using those channels that are noisy because of it (as confirmed in the ETSCH evaluation of the previous section). The high variation in the results of TSCH and ETSCH is due to the different interference conditions at the point of different nodes. For example, nodes A and B experience interference from all three noise generators, while nodes C and D only are affected by one noise generator. This leads to a considerable variation in the quality of incoming links to each node and thus different PRR values for them. The next observation of Figure 12 is that on average, ETSCH+DS provides higher PRR than ETSCH. This is because the noise that is generated by the two hidden noise generators on four random channels, only affects the incoming links of nodes A and B which leads to higher packet errors for these links. The DCS technique reports these noisy channels to the coordinator to be skipped for the channel hopping process. This leads to less packet errors and thus higher PRR for the mentioned links which leads to less variation in the results. Even by using the DCS technique, there is still some packet loss. These packet losses are due to the detection process of this technique which uses communication status to see if there is noise on a channel or not. This leads to some delay in detecting noisy channels.

To better investigate the behavior of different mechanisms, Figure 13 shows the PRR distribution of all incoming links to each node in the network for different mechanisms. The figure shows that ETSCH provides better PRR than TSCH for all the device nodes in the network. This PRR increase is less for nodes A and B compared to nodes C and D. This is because nodes A and B are affected by the two hidden noise generators and ETSCH cannot prevent their interference, while for nodes C and D it is only the visible noise generator that affects the packet reception, which is handled by ETSCH. The difference between the PRRs of TSCH at nodes A and B, which have the same setup, might be due to the orientation of the noise generators' antennas, which leads

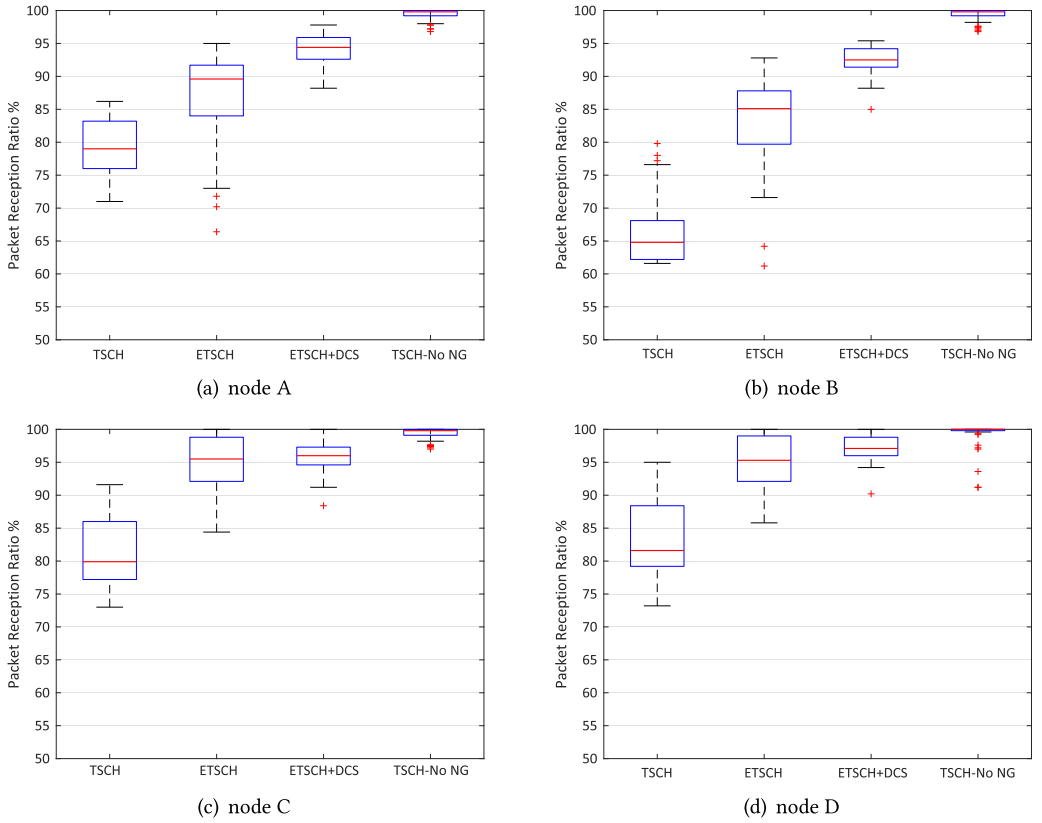


Fig. 13. PRR distribution of incoming links to different nodes over windows of 500 transmissions for different mechanisms in hidden interference setup.

to different interference effects on each of the nodes. This is because of the non-perfect omni-directional antennas of the sensor nodes. The high variation in PRR of ETSCH at each node is due to different distance between each couple of nodes. Because the transmission power of all the nodes is the same, different distances cause different signal-to-noise ratios at each receiver for different links. This leads to different PRR for incoming links to a node. For example, due to the short distance between A and B, there is a high chance that packets transmitted by B are correctly received at A, even in presence of interference. This is while for packets that are transmitted by node C, the greater distance leads to lower signal power at node A and thus lower signal-to-noise ratio and higher rate of packet errors.

In Figure 13, for ETSCH+DCS, the PRR of incoming links to nodes A and B increased compared to ETSCH, and its variation is also reduced. This shows that the DCS technique detects the interference hidden from the coordinator at the point of other nodes. As mentioned, this technique cannot completely resolve the negative effect of those hidden interferers because it uses normal communications to detect if there is noise on a channel or not.

As Figure 13 depicts, ETSCH+DCS outperforms the ETSCH also for incoming links to nodes C and D that are not affected by the hidden noise generators. The reason is the CCA mechanism that is defined in the TSCH protocol. CCA is done before each packet transmission by each node to skip transmissions if the used channel is busy. In this case, the outgoing links from both nodes

A and B will skip transmission due to the busy_channel result of CCA on the channels that are affected by the hidden noise generators. Thus, the PRR of links from nodes A and B to nodes C and D will be decreased for ETSCH. This is while the DCS technique reduces the CCA failures at nodes A and B and thus increases the number of received packets at nodes C and D. In conclusion, DCS is a useful technique to be used together with ETSCH to improve reliability of the network and reduce variation in PRR of the different links.

5.4 Energy Consumption Analysis

EDs that are performed by NICE and the 2Bytes field that is added to each packet for the DCS technique are the energy consumption overhead of ETSCH+DCS compared to the plain TSCH. Because NICE is used only by the coordinator of the network and energy may not be a stringent constraint for the coordinator, energy consumption is not a crucial metric in our work. However, to have a comprehensive comparison between ETSCH+DCS and other mechanisms, we analyze energy consumption. The energy consumption for a given number of packet communications (E_{comm}) can be extracted by Equation (16),

$$E_{comm}[J] = \Big((I_{Rx}N_{Rx}T_{listen}) + (I_{Tx}N_{Tx}T_{Tx})\Big) \times V_{cc}, \tag{16}$$

where I_{Rx} and I_{Tx} stand for the radio transceiver current in receive and transmit modes, respectively. N_{Rx} and N_{Tx} reflect the number of occurrences of each operation in the experiment duration, and V_{cc} represents the operation voltage of the transceiver. T_{Tx} represents the duration of a full packet transmission and T_{listen} is the duration that the receiver should listen to the medium to receive a packet which on average is

$$T_{listen} = T_{Tx} + (macTsTxOffset - macTsRxOffset) = T_{Tx} + 1.1\text{ms}. \tag{17}$$

This is the packet transmission/reception duration plus the duration that a receiver should wake up before the transmitter starts packet transmission. As the physical layer of the IEEE 802.15.4 standard defines, transmission of each Byte takes 32μs. Thus, transmission duration of a packet with length L_{packet} can be represented as

$$T_{Tx}[\mu s] = L_{packet}[Bytes] \times 32[\mu s/Bytes]. \tag{18}$$

Since ETSCH+DCS and ATSCH perform a number of EDs per slotframe for channel quality estimations, they impose energy overhead as well. The energy consumption of EDs (E_{CQE}) can be calculated as

$$E_{CQE}[J] = \Big(I_{ED}N_{ED}T_{ED}\Big) \times V_{cc}, \tag{19}$$

where I_{ED} stands for the radio transceiver current in energy detection mode and T_{ED} represents the duration of one energy detection. N_{ED} is the number of EDs performed while the packets are being transmitted.

The expected number of packet transmissions for a successful packet delivery over a link is equal to 1/PRP. We use this metric to consider the average energy consumption for packet (re)transmissions to deliver all packets to their destinations. Thus, the total energy consumption of each mechanism is

$$E_{mechanism} = \frac{1}{PRP} \times (E_{comm} + E_{CQE}). \tag{20}$$

Based on the ATmega256RFR2 datasheet, $I_{ED} = I_{Rx} = 12.5$mA, $I_{Tx} = 10$mA, $V_{cc} = 3.3$V, and T_{ED} is 128μs. Here we use the PRP values extracted from simulations of the lifelike interference scenario (Figure 9). Considering that each slotframe consists of eight timeslots in our experiments (skipping the three timeslots left idle to be used by ATSCH), the coordinator sends an EB with length $L_{packet} =$ 100Bytes and receives seven packets with the same size. With an average of 2.5 EDs per timeslot

in ETSCH and PRP = 82.5%, the consumed energy by ETSCH+DCS in a slotframe is $E_{ETSCH+DCS}$ = 1.761mJ at the coordinator. TSCH does not perform any EDs and does not have the 2Bytes overhead of the DCS technique for each packet (packets with size of 98Bytes). Because it has a lower PRP of 72.5% compared to ETSCH+DCS, its energy consumption is E_{TSCH} = 1.830mJ which is still more than the consumed energy by ETSCH+DCS. ATSCH uses two extra timeslots in each slotframe for EDs and has PRP of 79%. Thus, its energy consumption is E_{ATSCH} = 1.692mJ. This is 4% lower than the energy consumed by ETSCH+DCS in a slotframe, but in this example it comes with a bandwidth overhead of 20% for the network due to use of idle timeslots for channel quality estimation.

Again, we emphasize that the energy consumption overhead of the channel quality estimation is only for the coordinator node, which has less energy limitation compared to the other nodes in the network. Instead, higher PRP provided by ETSCH+DCS leads to lower number of required packet retransmissions by the end nodes. It means that the end nodes, which usually have stringent energy constraints, consume less energy when ETSCH+DCS is applied compared to TSCH and ATSCH.

6 CONCLUSIONS

This article proposes ETSCH+DCS, a mechanism on top of the IEEE 802.15.4 [5] TSCH protocol that uses a combination of a central and a distributed channel-quality estimation technique. This mechanism extracts the quality of different wireless channels to select the channels with the lowest interference as the hopping sequence list to improve the performance of the TSCH protocol. The central channel measurement technique, called NICE, operates at the coordinator of the network and proactively measures the spectrum energy in the idle part of each timeslot, when all the nodes in the network are silent. The energy sampling results are used to assign qualities to wireless channels. The distributed technique is performed by all the nodes in the network and uses the CCA results together with packet reception status to estimate the noise level on each channel. To use the packet reception status as a sign of interference on the communication channel, it is proposed to send a dummy packet with the shortest possible MAC header, when there is no packet available from the application layer. Based on the TSCH communication diagram, it can be shown that transmitting a small packet consumes less overall energy than not sending a packet, because of reduced idle listening. The results of the centralized and distributed channel quality estimation techniques are used to assign a quality factor to each channel. Using these qualities, channels with better qualities are periodically selected as the hopping sequence list of TSCH. ETSCH+DCS also uses a small secondary hopping sequence list (EBSL), that consists of the best quality channels, to disseminate periodic EBs. These EBs contain control information of the network such as the HSL. Only one field of the EBSL is updated per period, and thus the rate of EB losses in the network is reduced compared to using the regular HSL for broadcasting EBs. Experimental and simulation results show that ETSCH with NICE and EBSL provides higher packet reception ratios and lower length of burst packet losses compared to the plain TSCH protocol and another related work called ATSCH. Experiments also show that the DCS technique can detect existing interference in parts of the network that is not detectable by the centralized NICE technique, and thus it increases the PRR in those scenarios.

REFERENCES

[1] IEEE. 2005. IEEE standard for information technology—local and metropolitan area networks–specific requirements—part 15.1a: Wireless medium access control (MAC) and physical layer (PHY) specifications for wireless personal area networks (WPAN). *IEEE Std 802.15.1-2005 (Revision of IEEE Std 802.15.1-2002)* (June 2005), 1–700. DOI: http://dx.doi.org/10.1109/IEEESTD.2005.96290

[2] IEEE. 2007. IEEE standard for information technology—telecommunications and information exchange between systems—local and metropolitan area networks—specific requirements—part 11: wireless LAN medium access

control (MAC) and physical layer (PHY) specifications. *IEEE Std 802.11-2007 (Revision of IEEE Std 802.11-1999)* (June 2007), 1–1076. DOI: http://dx.doi.org/10.1109/IEEESTD.2007.373646

[3] HART. 2008. *HART Field Communication Protocol Specifications, Revision 7.1, HART Communication Foundation Std.* (2008).

[4] ANSI. 2011. *ANSI/ISA-100.11a-2011 Wireless Systems for Industrial Automation: Process Control and Related Applications, International Society of Automation Std.* (2011).

[5] IEEE. 2016. IEEE standard for low-rate wireless networks. *IEEE Std 802.15.4-2015 (Revision of IEEE Std 802.15.4-2011)* (April 2016), 1–709. DOI: http://dx.doi.org/10.1109/IEEESTD.2016.7460875

[6] N. Accettura, E. Vogli, M. R. Palattella, L. A. Grieco, G. Boggia, and M. Dohler. 2015. Decentralized traffic aware scheduling in 6TiSCH networks: Design and experimental evaluation. *IEEE IoT J.* 2, 6 (Dec. 2015), 455–470. DOI: http://dx.doi.org/10.1109/JIOT.2015.2476915

[7] Atmel Corporation. 2017. ATmega256RFR2 Xplained Pro Evaluation Kit. (Sept. 2017). http://www.atmel.com/tools/ATMEGA256RFR2-XPRO.aspx.

[8] Tengfei Chang, Thomas Watteyne, Kris Pister, and Qin Wang. 2015. Adaptive synchronization in multi-hop TSCH networks. *Comput. Netw.* 76 (2015), 165–176. DOI: http://dx.doi.org/10.1016/j.comnet.2014.11.003

[9] T. Chang, T. Watteyne, Q. Wang, and X. Vilajosana. 2016. LLSF: Low latency scheduling function for 6TiSCH networks. In *Proceedings of the 2016 International Conference on Distributed Computing in Sensor Systems (DCOSS'16)*. 93–95. DOI: http://dx.doi.org/10.1109/DCOSS.2016.10

[10] M. Domingo-Prieto, T. Chang, X. Vilajosana, and T. Watteyne. 2016. Distributed PID-based scheduling for 6TiSCH networks. *IEEE Commun. Lett.* 20, 5 (May 2016), 1006–1009. DOI: http://dx.doi.org/10.1109/LCOMM.2016.2546880

[11] P. Du and G. Roussos. 2012. Adaptive time slotted channel hopping for wireless sensor networks. In *Proceeedings of the Computer Science and Electronic Engineering Conference (CEEC'12)*. 29–34. DOI: http://dx.doi.org/10.1109/CEEC.2012.6375374

[12] D. Dujovne, T. Watteyne, X. Vilajosana, and P. Thubert. 2014. 6TiSCH: Deterministic IP-enabled industrial internet (of things). *IEEE Commun. Mag.* 52, 12 (Dec. 2014), 36–41. DOI: http://dx.doi.org/10.1109/MCOM.2014.6979984

[13] Simon Duquennoy, Beshr Al Nahas, Olaf Landsiedel, and Thomas Watteyne. 2015. Orchestra: Robust mesh networks through autonomously scheduled TSCH. In *Proceedings of the 13th ACM Conference on Embedded Networked Sensor Systems (SenSys'15)*. ACM, New York, NY, 337–350. DOI: http://dx.doi.org/10.1145/2809695.2809714

[14] A. Elsts, X. Fafoutis, R. Piechocki, and I. Craddock. 2017. Adaptive channel selection in IEEE 802.15.4 TSCH networks. In *Proceedings of the 2017 Global Internet of Things Summit (GIoTS'17)*. 1–6. DOI: http://dx.doi.org/10.1109/GIOTS.2017.8016246

[15] Everette S. Gardner and David G. Dannenbring. 1980. Forecasting with exponential smoothing: Some guidelines for model selection. *Decision Sciences* 11, 2 (1980), 370–383. DOI: http://dx.doi.org/10.1111/j.1540-5915.1980.tb01145.x

[16] J. C. Gittins. 1979. Bandit processes and dynamic allocation indices. *J. Roy. Stat. Soc. Ser. B* 41, 2 (1979), 148–177. http://www.jstor.org/stable/2985029

[17] Pedro Henrique Gomes, Thomas Watteyne, and Bhaskar Krishnamachari. 2017. MABO-TSCH: Multihop and blacklist-based optimized time synchronized channel hopping. *Trans. Emerg. Telecommun. Technol.* (2017), e3223–n/a.DOI: http://dx.doi.org/10.1002/ett.3223

[18] D. De Guglielmo, B. Al Nahas, S. Duquennoy, T. Voigt, and G. Anastasi. 2017. Analysis and experimental evaluation of IEEE 802.15.4e TSCH CSMA-CA Algorithm. *IEEE Trans. Vehic. Technol.* 66, 2 (Feb 2017), 1573–1588. DOI: http://dx.doi.org/10.1109/TVT.2016.2553176

[19] S. A. Hanna and J. Sydor. 2012. Distributed sensing of spectrum occupancy and interference in outdoor 2.4 GHz Wi-Fi networks. In *Proceedings of the Global Communications Conference (GLOBECOM'12)*. 1453–1459. DOI: http://dx.doi.org/10.1109/GLOCOM.2012.6503318

[20] K. Jeon and S. Chung. 2017. Adaptive channel quality estimation method for enhanced time slotted channel hopping on wireless sensor networks. In *Proceedings of the 2017 Ninth International Conference on Ubiquitous and Future Networks (ICUFN'17)*. 438–443. DOI: http://dx.doi.org/10.1109/ICUFN.2017.7993823

[21] Peishuo Li, T. Vermeulen, H. Liy, and S. Pollin. 2015. An adaptive channel selection scheme for reliable TSCH-based communication. In *Proceedings of the 2015 International Symposium on Wireless Communication Systems (ISWCS'15)*. 511–515. DOI: http://dx.doi.org/10.1109/ISWCS.2015.7454397

[22] K. Muraoka, T. Watteyne, N. Accettura, X. Vilajosana, and K. S. J. Pister. 2016. Simple distributed scheduling with collision detection in TSCH networks. *IEEE Sens. J.* 16, 15 (Aug 2016), 5848–5849. DOI: http://dx.doi.org/10.1109/JSEN.2016.2572961

[23] M. R. Palattella, N. Accettura, L. A. Grieco, G. Boggia, M. Dohler, and T. Engel. 2013. On optimal scheduling in duty-cycled industrial IoT applications using IEEE802.15.4e TSCH. *IEEE Sens. J.* 13, 10 (Oct 2013), 3655–3666. DOI: http://dx.doi.org/10.1109/JSEN.2013.2266417

[24] M. R. Palattella, N. Accettura, X. Vilajosana, T. Watteyne, L. A. Grieco, G. Boggia, and M. Dohler. 2013. Standardized protocol stack for the internet of (important) things. *IEEE Commun. Surv. Tutor.* 15, 3 (2013), 1389–1406. DOI: http://dx.doi.org/10.1109/SURV.2012.111412.00158

[25] M. R. Palattella, T. Watteyne, Q. Wang, K. Muraoka, N. Accettura, D. Dujovne, L. A. Grieco, and T. Engel. 2016. On-the-fly bandwidth reservation for 6TiSCH wireless industrial networks. *IEEE Sens. J.* 16, 2 (Jan. 2016), 550–560. DOI: http://dx.doi.org/10.1109/JSEN.2015.2480886

[26] K Pister and Lance Doherty. 2008. TSMP: Time synchronized mesh protocol. In *Proceedings of the International Symposium on Distributed Sensor Networks* (2008), 391–398.

[27] S. Pollin, M. Ergen, M. Timmers, A. Dejonghe, L. van der Perre, F. Catthoor, I. Moerman, and A. Bahai. 2006. Distributed cognitive coexistence of 802.15.4 with 802.11. In *Proceedings of the 2006 1st International Conference on Cognitive Radio Oriented Wireless Networks and Communications*. 1–5. DOI: http://dx.doi.org/10.1109/CROWNCOM.2006.363456

[28] C. F. Shih, A. E. Xhafa, and J. Zhou. 2015. Practical frequency hopping sequence design for interference avoidance in 802.15.4e TSCH networks. In *Proceedings of the 2015 IEEE International Conference on Communications (ICC'15)*. 6494–6499. DOI: http://dx.doi.org/10.1109/ICC.2015.7249359

[29] Kannan Srinivasan, Maria A. Kazandjieva, Saatvik Agarwal, and Philip Levis. 2008. The beta-factor-factor: Measuring wireless link burstiness. In *Proceedings of the 6th ACM Conference on Embedded Network Sensor Systems (SenSys'08)*. ACM, New York, NY, 29–42. DOI: http://dx.doi.org/10.1145/1460412.1460416

[30] D. Stanislowski, X. Vilajosana, Q. Wang, T. Watteyne, and K. S. J. Pister. 2014. Adaptive synchronization in IEEE802.15.4e Networks. *IEEE Trans. Industr. Inform.* 10, 1 (Feb 2014), 795–802. DOI: http://dx.doi.org/10.1109/TII.2013.2255062

[31] R. Tavakoli, M. Nabi, T. Basten, and K. Goossens. 2015. Enhanced time-slotted channel hopping in WSNs using non-intrusive channel-quality estimation. In *Proceedings of the 2015 IEEE 12th International Conference on Mobile Ad Hoc and Sensor Systems (MASS'15)*. 217–225. DOI: http://dx.doi.org/10.1109/MASS.2015.48

[32] Rasool Tavakoli, Majid Nabi, Twan Basten, and Kees Goossens. 2016. An experimental study of cross-technology interference in in-vehicle wireless sensor networks. In *Proceedings of the 19th ACM International Conference on Modeling, Analysis and Simulation of Wireless and Mobile Systems (MSWiM'16)*. ACM, New York, NY, 195–204. DOI: http://dx.doi.org/10.1145/2988287.2989141

[33] P. Thubert, M. R. Palattella, and T. Engel. 2015. 6TiSCH centralized scheduling: When SDN meet IoT. In *Proceedings of the 2015 IEEE Conference on Standards for Communications and Networking (CSCN'15)*. 42–47. DOI: http://dx.doi.org/10.1109/CSCN.2015.7390418

[34] P. Thubert, T. Watteyne, M. R. Palattella, X. Vilajosana, and Q. Wang. 2013. IETF 6TSCH: Combining IPv6 connectivity with industrial performance. In *Proceedings of the 2013 7th International Conference on Innovative Mobile and Internet Services in Ubiquitous Computing*. 541–546. DOI: http://dx.doi.org/10.1109/IMIS.2013.96

[35] X. Vilajosana, Q. Wang, F. Chraim, T. Watteyne, T. Chang, and K. S. J. Pister. 2014. A realistic energy consumption model for TSCH networks. *IEEE Sens. J.* 14, 2 (Feb. 2014), 482–489. DOI: http://dx.doi.org/10.1109/JSEN.2013.2285411

[36] Qin Wang, Xavier Vilajosana, and Thomas Watteyne. 2017. *6top Protocol (6P)*. Internet-Draft draft-ietf-6tisch-6top-protocol-07. Internet Engineering Task Force. Retrieved from https://datatracker.ietf.org/doc/html/draft-ietf-6tisch-6top-protocol-07.

[37] Thomas Watteyne, Ankur Mehta, and Kris Pister. 2009. Reliability through frequency diversity: Why channel hopping makes sense. In *Proceedings of the 6th ACM Symposium on Performance Evaluation of Wireless Ad Hoc, Sensor, and Ubiquitous Networks (PE-WASUN'09)*. ACM, New York, NY, 116–123. DOI: http://dx.doi.org/10.1145/1641876.1641898

[38] T. Watteyne, J. Weiss, L. Doherty, and J. Simon. 2015. Industrial IEEE802.15.4e networks: Performance and trade-offs. In *Proceedings of the 2015 IEEE International Conference on Communications (ICC'15)*. 604–609. DOI: http://dx.doi.org/10.1109/ICC.2015.7248388

Received December 2016; revised September 2017; accepted November 2017