EXCHANge: Securing IoT via Channel Anonymity

Savio Sciancalepore[1], Gabriele Oligeri[1], Giuseppe Piro[3,4], Gennaro Boggia[3,4], Roberto Di Pietro[1]

[1] *Hamad Bin Khalifa University (HBKU), Department of Computer Science and Engineering (CSE), Information and Computing Technology (ICT) Division, Doha, Qatar.*
e-mail: ssciancalepore@hbku.edu.qa, goligeri@hbku.edu.qa, rdipietro@hbku.edu.qa
[3] *Dep. of Electrical and Information Engineering (DEI), Politecnico di Bari, Bari (Italy); e-mail: {name.surname}@poliba.it.*
[4] *CNIT, Consorzio Nazionale Interuniversitario per le Telecomunicazioni.*

Abstract

Establishing confidentiality between communicating peers is still an issue in contexts where solutions based on asymmetric keys are not viable, such as in dynamic Internet of Things (IoT) systems made up of heterogeneous and resource constrained devices.
From the current literature, channel anonymity emerges as a promising methodology able to support key-establishment protocols. But, to the best of authors' knowledge, no works already demonstrated its practical adoption over a concrete communication technology. To bridge this gap, we experimentally show that a lightweight key-establishment protocol based on channel anonymity is viable.
The contributions of this work are mainfold. First, we introduce EXCHANge, a protocol that achieves key establishment exploiting channel anonymity, despite the presence of either a passive or active global-eavesdropper adversary. Second, we evaluate the performance of EXCHANge through an extensive experimental campaign involving real world IoT devices (OpenMote-CC2538). Our results demonstrate that the proposed solution introduces a limited over-

The final published version of this article is available online at `https://doi.org/10.1016/j.comcom.2018.11.003`.
Please cite this article as: S. Sciancalepore, G. Oligeri, G. Piro et al., EXCHANge: Securing IoT via channel anonymity, Computer Communications (2018), *https://doi.org/10.1016/j.comcom.2018.11.003*

head, thus being able to meet the requirements of resource constrained devices. Finally, we experimentally demonstrate the security of the EXCHANge protocol against passive and active adversaries. Overall, this paper proves that channel anonymity can be a powerful tool in the IoT setting, to achieve a secure, effective, and efficient key-establishment.

Keywords: Key Agreement; Channel Anonymity; IoT; IEEE 802.15.4, Experimentation.

1. Introduction

Internet of Things (IoT) is becoming an important part of our lives, where homes, offices, cars and even our bodies do share data [1], [2], [3]. IoT is currently growing at a fast pace mainly due to the wide deployment of short-range low-power communication technologies [4]. While on the one hand IoT is introducing new scenarios with new services and functionalities, on the other hand it is becoming an increasingly attractive target for cyber criminals. In fact, a smart adversary can easily exploit the vulnerabilities of the weakest element in the network to gain access to the overall IoT infrastructure [5].

Unfortunately, the current state of the art about IoT security is constituted by a plethora of heterogeneous solutions implemented at different levels of the architecture, starting from the network up to the application layer [6].

To guarantee the confidentiality of wireless communications between IoT devices, the vast majority of contributions in the literature widely uses pre-existing secrets stored in the device memory, while authentication and confidentiality are often split into separate problems [7, 8]. Authentication is performed through dedicated protocols via a one-time process, while confidentiality is achieved by periodically refreshing the encryption key [9]. Such a solution for key establishment is not secure: an adversary that is able to disclose all the device's secrets (e.g via cryptanalysis or just temporally-limited physical compromise) will have full access to all the subsequent communications.

A secure solution to key-establishment should not leverage any pre-existing secret stored in the device memory: although asymmetric encryption represents an effective solution to this problem, it is not the most efficient, because it introduces a significant CPU burden, usually unbearable in con-

texts where low-end devices are subject to tight CPU constraints. Moreover, there are even scenarios where the device architecture cannot simply support the asymmetric solution, e.g., in the low-end IoT devices [10]. An ideal solution should minimize the usage of the CPU, possibly exploiting the already existing communications and avoiding any bandwidth overhead.
This niche research problem is exactly where physical layer security can provide a significant improvement. In particular, it can be used to guarantee confidentiality among devices message exchange. In fact, key-establishment can be performed by simply using the radio interface and the standardized communication technology used by constrained devices, avoiding CPU expensive computations, battery drain, and bandwidth overhead [11], [12], [13]. Physical layer security is recently gaining momentum, due to the resulting efficiency and effectiveness of the proposed solutions for wireless communications. In fact, this is the perfect solution for IoT scenarios, where the devices already use their radio interface for exchanging radio messages along their operational life-time. However, these solutions usually leverage specific modulation or transmission techniques, e.g., OFDM or MIMO, requiring dedicated hardware to work as intended (see Sec. 2). In addition, only few of these techniques have been integrated in a real technology and evaluated through a real experimental campaign.

In this context, the work presented in [12] introduced COKE, i.e., a key-establishment protocol leveraging radio channel anonymity. It covers the following features: (i) it needs only one secure hash computation per key; (ii) it does not resort to any pre-existing secret in the device memory; and, (iii) it allows two peers to converge to a secret key by only exploiting the underlying radio communications among these two devices. COKE has been subsequently extended to be deployed on a wide network [13]. Unfortunately, the current contributions related to COKE, that are [12] and [13], still present some limitations. First, COKE is a theoretical solution, that has never been adapted to a real communication technology, yet. Second, because of the lack of any implementation on a real architecture, its real performance have never been measured in real deployment.
Contribution. In this paper, we introduce the EXCHANge protocol, a crypto-less over-the-air key establishment multi-round protocol based on sender/receiver anonymity, specifically conceived to secure IoT networks based on the IEEE 802.15.4 communication technology.

EXCHANge is inspired by the same principles of COKE, albeit it presents several differences, modifications and improvements in order to meet the con-

straints of technologies and real devices making up the IoT. To show its effectiveness and suitability for the IoT technology, the EXCHANge protocol has been implemented in the OpenWSN protocol stack [14] and tested on the OpenMote-CC2538 architecture [15]. By looking at the current literature, it emerges that EXCHANge is the first real-world implementation of a key-establishment protocol based on channel anonymity. We provide the results of an extensive measurement campaign in several configuration scenarios, considering both the security and the performance of the protocol. The performance achieved by EXCHANge represent a milestone for the key-establishment protocols working at the physical layer: EXCHANge can set-up a new key of 128 bits in less than 10 seconds (on average), while requiring only a single hash computation. We successfully tested the security of the EXCHANge protocol against both: i) a global eavesdropper adversary strategically deployed in the playground; and, ii) an active adversary trying to biasing the key generation and subsequently proxying the messages between the legitimate parties. Finally, we provide an experimental measurement of the energy consumption showing that EXCHANge is able to perform a key-establishment consuming just the 0.14% of the battery capacity of a node, in the worst case scenario.

We remark that the present contribution is the first that effectively demonstrates the feasibility of channel anonymity in the IoT context, and that experimentally shows the performance and robustness of such an approach in real IoT deployments, using state-of-the-art constrained devices. Since the security of EXCHANge is based on the hidden identity of the transmitter, its deployment is particularly suitable for indoor scenarios and wearable applications—major IoT use-cases—, where both the unpredictability of people movements and the environment heterogeneity make guessing the transmitting source a probabilistic problem.

Organization. This paper is organized as follows. Sec. 2 reviews the current state of the art as for key-establishment protocols exploiting the physical layer of the wireless communications, while Sec. 3 provides the necessary background on channel anonymity techniques. Sec. 4 introduces our solution, the core idea behind key-establishment protocols based on radio channel anonymity and the adversarial model. Sec. 5 shows the details of the protocol implementation, Sec. 6 discusses its performance, while Sec. 7 and Sec. 8 provide the security and the energy consumption analysis, respectively. The discussion of the main results achieved within this paper is carried out in

Sec. 9. Finally, some concluding remarks are reported in Sec. 10.

2. Related work

The majority of the solutions dealing with the key-establishment problem at the physical layer leverages either the channel state information (CSI) or the received signal strength (RSS). In the following, we provide a review of the most important contributions in the area.

RSS based key-establishment. Physical Layer security techniques based on the RSS leverage the intrinsic characteristics of the strength of the received signal to perform key agreement between two devices. To name a few of these solutions, authors in [16] proposed a secret bit generation algorithm based on multiple observations of the wireless channel, e.g., with multiple antennas, at different frequencies and times. Although the proposed solution mitigates the imperfect reciprocity of the radio channel, it has not been implemented in a real test-bed and its performance still depend on the channel model used for simulating the signal propagation.

A solution specifically designed for OFDM communication systems has been proposed by [17]. Authors designed a low pass filter to improve the channel reciprocity by suppressing the noise and they showed that the low pass filter improves the channel reciprocity, decreases the key disagreement, and effectively increases the success of the key generation. However, the success probability of the key agreement strictly depends on the noise level of the surrounding environment, requiring a pre-analysis stage that could be hard to achieve.

Authors in [18] proposed a secure key generation scheme based on the subcarriers' channel responses of OFDM systems, and subsequently in [19], they proved the practical aspects related to the security by implementing the key generation scheme on the Wireless open-Access Research Platform (WARP).

In [20], authors studied the key generation problem in the two-way relay channel, in which there is no direct channel between the key generating terminals. They proposed a key generation scheme that achieves a substantially larger key rate than that of a direct channel mimic approach.

Another solution to secret key generation exploiting wireless channels that exhibit sparse structure in the wideband regime has been proposed in [21]. Authors studied the impact of sparsity on the secret key capacity introducing two measures: (i) ergodic secret key capacity and (ii) outage probability. However, if the attacker could guess on the physical channel used for key

generation, the security of the whole scheme could be broken.
Another solution exploiting multiple antennas to avoid low secrecy rate in quasi-static channels is presented in [22]. Multiple-input-multiple-output (MIMO) based antennas recently turned out to be an effective solution to increase the entropy of keys generated by key-establishment protocols exploiting channel impulse response. By combining MIMO antennas and random beamforming, authors proved the security of the proposed protocol against an eavesdropper adversary. However, this solution requires a specific physical layer based on the use of MIMO techniques, as well as a dedicated hardware including multiple antennas both at the transmitter and the receiver. Thus, this approach cannot be used in a general setting, involving constrained devices equipped with only a single antenna for both transmission and reception. In addition, the above solution has been evaluated only via simulations, while a real experimental campaign is missing.
Authors in [23] investigated channel reciprocity based key-generation by implementing it on the 8 bit SoC CC2531 architecture. They provided a detailed performance and security analysis showing the code size, number of clock cycles, and power consumption. However, they do not discuss how to integrate such a solution within a protocol stack specifically designed for the IoT technology.
CSI based key-establishment. Approaches based on Channel State Information (CSI) represent a generalization of the RSSI-based approaches described in the previous section. The main idea is to leverage any kind of shared information from the physical characteristics of the communication channel to generate a secret key between communicating peers, e.g., delay spread.

To name a few examples, in [24], authors proposed a solution to deal with channels characterized by a long coherence time and therefore low secret bit generation due to the staticity and low entropy of the channel impulse response. The solution synthetically induces channel fluctuations by controlling amplitude and phase of each symbol in the training sequence of a multiple antenna transmitter. A thorough analysis supported by several simulations confirms that their solution can generate a secret bit rate between 2 and 20Kb/s as a function of the channel bandwidth.
Authors in [25] exploited the channel state information (CSI) from orthogonal frequency-division multiplexing (OFDM) to generate secret keys using off-the-shelf devices and the 802.11n protocol. They implemented a validation-recombination mechanism to extract keys using the combined information of

all the OFDM subcarriers.
Channel pulse response between two transceivers is considered as a source of common randomness in [26]. The authors presented an information theoretic and extensive study into secret-key agreement using the reciprocity of radio propagation channels, and also showed a number of numerical results bounding and simulating the secret-key lengths possible for indoor Ultra-WideBand (UWB) channels.
Experimental measurements have been included in [27] to assess the feasibility of a procedure for secret key extraction from the UWB channel physical parameters. Authors considered different parameters and observed low mismatch of the raw key between the devices and a high mismatch between the devices and the adversary.
Authors in [28] explored ray-tracing techniques to perform attacks and to evaluate the security of UWB secret key generation methods. They confirmed that it is difficult for a third party to obtain the exact channel responses and thus to retrieve the secret keys. However, the robustness of UWB key generation methods depends on the complexity for attackers to describe precisely the physical environment and on the post processing methods to agree on the same key.

Authors in [29] designed an authentication and key agreement protocol for mobile devices, called "The Dancing Signals" (TDS), being extremely fast and error-free. It guarantees that only devices in a close physical proximity can agree on a key and any device outside a certain distance gets nothing about the key. Note that the authentication proposed in this latter work can be easily coupled with the key agreement method proposed herein, providing also shorter times to agree on the key.

Resorting to channel anonymity despite CSI-based approaches has advantages and disadvantages [12][13]. Indeed, while CSI-based approaches require a certain amount of entropy to be present at the physical layer of the communication channel, approaches based on channel anonymity can be executed even on flat channels, since the randomness is provided by the two pairs participating in the key-establishment process. Moreover, CSI based approaches have been proven to be fragile to active adversaries [30], while in this paper, we prove how EXCHANge enables the two legitimate parties to detect malicious behaviours from an active adversary.

To sum up, all of above approaches are strictly dependent on channel conditions to find the right trade-off between high randomness and low number of errors; indeed, they become highly inefficient in harsh channel conditions.

Moreover, CSI-based approaches, as well as RSSI-based techniques, do not allow to detect the presence of active adversaries.

Other approaches. Apart from the classification given at the beginning of this chapter, it is worth mentioning another emerging class of approaches, specifically tailored for Wireless Body Area Networks (WBAN). They aim at using the gait of a person to perform key agreement between sensors attached to the body of an individual. Contributions as [31], [32], [33], [34] and [35] fall into this category. To perform device authentication and fast secret key extraction at the same time, these approaches exploit the heterogeneous channel characteristics among the collection of on-body channels during body motion. Specifically, with simple body movements, channel variations between line-of-sight on-body devices are relatively stable while those for non-line-of-sight devices are unstable. However, the whole security of these schemes leverages on their constant (and correlated) movement. If nodes are static, the communications become insecure. Finally, it is worth mentioning a solution that, while being rooted on theoretical analysis, for the first time do provide a viable encoding scheme to guarantee both perfect secrecy (i.e., no information leakage) and reliable communication over the generalized Ozarow-Wyners wire-tap channel [36].

The main features of the most representative contributions we found in the literature are summarized in Tab. 2, along with their relationship with the requirements discussed above.

3. Background on Channel Anonymity

Channel Anonymity has been recently proposed as a method to establish secret keys between devices [37, 38]. Specifically, this property refers to the impossibility for a passive eavesdropper to identify with 100% success rate the source of a message.

Channel anonymity is achieved by combining the sanitization of any identification information carried within the transmitted packet and the random modulation of the transmission power. While the former hinders any identification through the reading of the content of the message, the latter mitigates the analysis of the power of the transmitted messages.

In practice, a passive eavesdropper, namely $\mathcal{E}$, is able to guess a few bits of the key by changing its position with respect to the parties involved in the key exchange. In fact, by getting closer to one of the two peers, it increases its chances to distinguishing the transmitting source by analyzing the power

Feature	[16]	[17]	[18]	[20]	[21]	[22]	[23]	[24]	[25]	[26]	[27]	[28]	[29]	**EXCHANge**
Independence from PHY-layer technique	×	×	×	✓	×	×	✓	×	×	×	×	×	×	✓
Independence from channel model	×	×	×	×	×	×	×	×	×	×	✓	×	✓	✓
Robustness to non Flat channels	✓	×	✓	✓	×	×	×	✓	✓	×	×	×	✓	✓
No Dedicated Hardware	×	×	×	✓	✓	×	✓	×	×	×	×	×	×	✓
MAC layer Integration	×	×	✓	×	×	×	×	×	✓	×	×	×	✓	✓
Real Implementation and Performance Evaluation	×	×	✓	×	×	×	✓	×	✓	×	✓	✓	✓	✓

Table 1: Main features of the major contributions in the area and their relationship with the identified requirements.

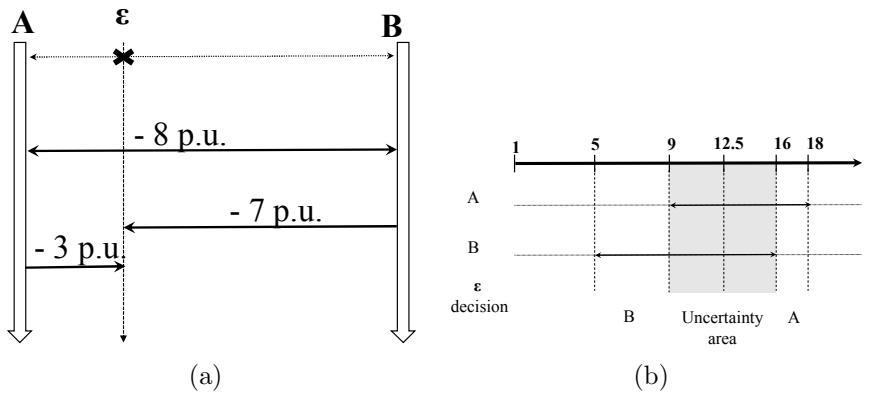


Figure 1: A practical example of the adversarial strategy.

of the involved signals [39]. However, it is worth noting that protocols based on channel anonymity such as [12] have been proved to be (probabilistically) resilient to this class of attacks.

To clarify these two aspects of the protocols, Figure 1(a) shows a simple scenario in which $\mathcal{E}$ tries to identify the secret key being generated by $\mathcal{A}$ and $\mathcal{B}$.

Eve's position with respect to $\mathcal{A}$ and $\mathcal{B}$ is such that it experiences an attenuation of 7 power units (p.u.) when $\mathcal{B}$ is transmitting and an attenuation of 3 p.u. when $\mathcal{A}$ is transmitting.

$\mathcal{A}$ and $\mathcal{B}$ can choose a random transmission power in the interval $[12;:::;21]$ p.u. ($\mathcal{E}$ is closer to $\mathcal{A}$ than $\mathcal{B}$). $\mathcal{A}$ and $\mathcal{B}$ can transmit at any time slot, experiencing a signal attenuation of 8 p.u.. As shown in Figure 1(b), $\mathcal{E}$ experiences a signal strength in the interval $5-16$ and $9-18$ p.u. when $\mathcal{B}$ and $\mathcal{A}$ are transmitting, respectively. By looking at the received signal strengths by $\mathcal{E}$'s side, three different areas can be identified:

- when the received signal strength is less than 9, $\mathcal{E}$ can be sure that the transmitting source is $\mathcal{B}$ (given that it is farther from $\mathcal{B}$);
- when the received signal strength is greater than 16, $\mathcal{E}$ can be sure that the transmitting source is $\mathcal{A}$ (given that it is closer to $\mathcal{A}$);
- when the received signal strength is in the uncertainty area $9-16$, $\mathcal{E}$ cannot deterministically establish the source of the exchanged bit.

In summary, while in the first two areas $\mathcal{E}$ can leverage the knowledge of the deployment scenario to deduce the actual transmitting source, in the

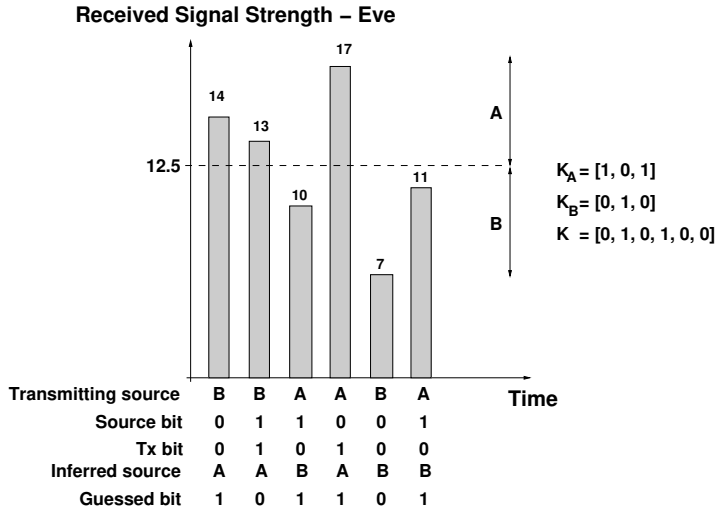


Figure 2: A practical example showing how EXCHANge achieves source anonymity.

third case, $\mathcal{E}$ does not gain any advantage. Indeed, $\mathcal{E}$ can perform a decision by establishing a threshold, e.g., the mean value of the uncertainty area (12.5 p.u.), and deciding for the source to be $\mathcal{B}$ in case the received signal strength is less than the threshold and deciding for $\mathcal{A}$ in case the received signal strength is higher than the threshold. Figure 2 shows the details about this adversarial strategy, assuming that the two initial secrets for $\mathcal{A}$ and $\mathcal{B}$ are $K_A = [1; 0; 1]$ and $K_B = [0; 1; 0]$, respectively. The threshold is fixed on the value of 12.5 p.u., i.e. exactly in the middle of the uncertainty area. When the strength of the received signal is under the threshold, the transmitting source is assumed to be $\mathcal{B}$ (and thus the eavesdropped bit is not complemented), while $\mathcal{A}$ is selected (and the corresponding bit is complemented) in case the value of the received signal strength is above the threshold.

As a result of the whole procedure, the eavesdropper $\mathcal{E}$ is able to disclose only 2 out of the 6 bits in the shared secret, i.e. the forth and the fifth, while the remaining bits are uncertain. Thus, only a trial-and-error brute force approach would ideally guarantee the disclosure of the whole secret. However, this could be made computationally hard by properly increasing the key length and the number of transmissions according to the desired security level.

As a final step, a generic cryptographically secure hashing function is applied over the shared bit-string K_S to compute the final key K.

3.1. Previous Work on Channel Anonymity and Main Limitations

Early solutions based on channel anonymity have been proposed by [40] and [41], and subsequently extended by [12] and [13] with a theoretical analysis in scenarios constituted by only two peers and large networks, respectively. The idea is mainly rooted in the concept of exploiting anonymous transmitters despite secret information.

To sum up, it emerges that no contribution in the literature shows how to apply channel anonymity in IoT networks based on the IEEE 802.15.4 communication technology. Furthermore, the results of a real implementation, as well as a detailed analysis of the security and energy trade-off, are not available, yet. Finally, as it will emerge clearly in the following, contrary to other approaches, such as either RSSI or CSI-based key-establishments, the entropy of the key generated by the proposed approach does not strictly depend on multipath fading conditions neither require the nodes to be dynamic, thus clearly representing a novel solution.

4. Protocol overview and adversarial model

The scenario considered in this paper includes two generic devices, namely $\mathcal{A}$ and $\mathcal{B}$, whose aim is to generate shared bit-string K_S of length $2N$ bits by exploiting $2N$ transmissions and resorting to two random secret seeds K_A and K_B, not disclosed to other parties. The two peers do not share any information and they do not resort to Public Key Infrastructure (PKI)-based solutions, and not even to Diffie-Hellman key exchange protocol [42].

To perform such a task, we introduce EXCHANge a crypto-less over-the-air key establishment multi-round protocol based on peers anonymity. Our solution requires the involved parties to feature only a radio and minimal CPU and storage space.

4.1. EXCHANge in a nutshell

The EXCHANge protocol requires the two involved parties, i.e., $\mathcal{A}$ and $\mathcal{B}$, to properly anonymize their radio messages, by removing the information that could led a third-party to the identification of the transmitting entity (e.g., the source and destination MAC addresses).

EXCHANge requires multiple rounds to generate a shared secret key. Each round of the protocol allows the two peers to establish up to one secure bit. The secrecy of this bit is probabilistic; thus, $2N$ transmissions have to be performed to generate a shared bit-string K_S of $2N$ bits.

At the beginning of each round, the transmitter is randomly selected using a channel contention algorithm, e.g., like that one implemented by the 802.11 family protocols [43]. To simplify the presentation, we model the output of the contention protocol by using a random variable $\mathsf{c}[i]$, taking the value 1 when the winner of the contention algorithm is $\mathcal{A}$, while taking the value 0 when the winner is $\mathcal{B}$. It is worth noting that c is not a shared information between $\mathcal{A}$ and $\mathcal{B}$, but it is indeed a mathematical concept used to simplify the subsequent discussion.

More in detail, the sender election process resorts to a vector c (of size $2N$) that controls the interleaving procedure between $\mathcal{A}$ and $\mathcal{B}$ in accessing the wireless channel. When $\mathsf{c}[i] = 1$ the transmitting source is $\mathcal{A}$, while when $\mathsf{c}[i] = 0$, the transmitting source is $\mathcal{B}$, with $i \in [0; 2N - 1]$. As it will be clear in the following, the protocol does not require the vector c to be a shared information between the two peers.

Considering the toy example shown in Figure 3, $\mathcal{A}$ and $\mathcal{B}$ choose two secret random seeds, $K_A = [0; 0]$ and $K_B = [1; 1]$, respectively. Now, for each round of the EXCHANge protocol, assuming that $\mathsf{c} = [1; 0; 1; 0]$ (i.e., the transmission sequence is [$\mathcal{A}$, $\mathcal{B}$, $\mathcal{A}$, $\mathcal{B}$]), if $\mathcal{A}$ transmits, then the bit b extracted from K_A is transmitted as complemented, while b is transmitted as it is when $\mathcal{B}$ is the transmitter. We observe that a passive observer can be also aware of the above rule, but she would not able to discriminate the transmitting source and therefore to decide if to complement or not the eavesdropped bit. Therefore, in the toy scenario depicted in Figure 3 both $\mathcal{A}$ and $\mathcal{B}$ transmit $[1; 1]$, and the final shared bit-string will be $K = [0; 1; 0; 1]$.

Bit complement. The security of our solution is rooted on the fact that an adversary, namely $\mathcal{E}$, cannot decide if the current eavesdropped bit has to be complemented or not. Indeed, assuming that $\mathcal{E}$ cannot infer on the current transmitting source (channel anonymity), she cannot decide if the current bit has to be complemented or not.

Even though $\mathcal{E}$ eavesdrops all the messages exchanged by the two communicating parties, she does not gain any significant advantage if complementing or not a particular eavesdropped bit. Therefore, the security of the EXCHANge protocol is not based on the value of the transmitted bit, but on the *hidden identity of the source*. Ideally, the best possible educated guess would be to randomly choosing the transmitter.

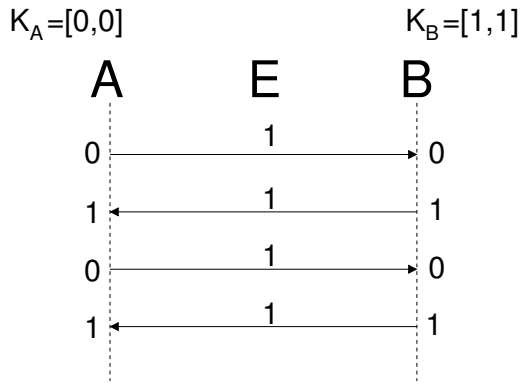


Figure 3: Toy example of the EXCHANge protocol.

4.2. Adversarial model

The most important features of the adversary envisioned in this contribution can be summarized as follows:

- Global eavesdropping capabilities: it is able to eavesdrop all the communications in the network.
- Passive player capabilities: It does not interfere with the network by sending or altering messages — its goal is to keep staying stealthy.
- Network deployment aware: it knows the network physical topology (i.e., the deployment of the nodes).
- Active player capabilities: during the execution of the EXCHANge protocol, it can either modify the messages sent by one of the legitimate parties or even impersonating one party and preceding its transmission every time.

More in detail, the passive eavesdropper ($\mathcal{E}$) can be potentially constituted by a single entity or by a variety of eavesdropping stations, and it has the capability of eavesdropping all the communications in the network. In line with other closely related works [44], [45], [46], in this case $\mathcal{E}$ takes on a passive role since it does not tamper with or subvert any node in the network, while avoiding radio communications in order not to be recognized.

Moreover, $\mathcal{E}$ is aware of the physical deployment of the system, as in [47]. Thus, it knows the coordinates of the nodes in the network and it is able to adjust its position in order to achieve the maximum effectiveness

in performing attacks based on the analysis of the received signal strength, as previously detailed. Conversely, legitimate nodes are not aware of the adversary's position.

Being equipped with omni-directional antennas, $\mathcal{E}$ is able to retrieve any packet transmitted throughout the network. However, this is not enough to determine the position and the distance of any signal source with respect to its position, e.g., by using the received signal power [39]. In the following, we adopt the conservative approach used also in other related works such as [12] and [13], where the distance between a transmitter and a receiver can be approximated through the Friis transmission equation [39], with an inverse quadratic relation.

Although real environments can be approximated by more refined models, this is a conservative approach from the network perspective, since real deployments are characterized by random fast power fluctuations that hinder $\mathcal{E}$ in performing such an estimation. In summary, by assuming the Friis model as valid, we enable $\mathcal{E}$ to perform a more reliable estimation of the received signal strength, thus improving its effective chances to retrieve the shared secret.

Moreover, the assumption for $\mathcal{E}$ to only feature "omni-directional" antennas is a practical consideration rather than a limitation. In fact, even if a directional antenna can theoretically and easily improve the adversarial chances to guess the secret key, i.e., by focusing two different directive antennas on each of the communicating peers, such an attack does not scale out with the number of nodes in the network, thus becoming practically unfeasible [13]. Moreover, depending on the scenario, there are other major issues related to the directive-antenna attack. As an example, in the following we consider three different scenarios: wearable devices, indoor and outdoor device pairing.

- *Wearable devices.* Wearable devices are usually close each others during the pairing process and precisely discriminating one device from the other one might be a challenging task even with a directive antenna. Indeed, directive antennas working at 2.4 GHz commercially available nowadays are characterized by a main lobe aperture in the range between 30 and 40 degrees.

- *Indoor.* Indoor environments are characterized by walls, doors, and cupboard. If the two pairing devices are not in line-of-sight, the adversary has to deploy at least two different directive antennas and tracking

the two devices during their movements. The previous behaviour might be very difficult to deploy while being very easy to spot.

- *Outdoor.* Outdoor scenarios are giving more chances to the adversary. Indeed, assuming the presence of the line-of-sight between the adversary and the two devices, an open-field with no obstructions might give to the adversary the opportunity to infer on the transmitting source. Nevertheless, the above scenario is very unlikely. Indeed, even when outdoor, the link between the adversary and the two devices will be characterized by several obstructions, e.g., people moving, cars, buildings, etc. Therefore, the adversary will not be able to easily collect clean samples coming from the two devices.

Furthermore, our adversary could theoretically exploit radio fingerprinting techniques. Although there have been several attempts to uniquely identify a transmitter source by just looking at the received signal [48], realistic operating conditions involve propagation channels characterized by multipath fading and channel impairments that significantly affect the performance of the above techniques [49, 50, 51].

In addition, we consider an active adversary able to inject new packets during the key-establishment process. We will show in Sec. 7.2 how the EXCHANge protocol enables the two legitimate parties to detect the above behaviour.

Finally, note that any key establishment protocol must be coupled with an authentication mechanism. This is indeed necessary to assure that the protocol is not compromised nor hijacked by a malicious party. The authentication problem is commonly decoupled from the key establishment problem [7][8] [52][53]. We remark that our focus is on the feasibility of key establishment through channel anonymity, thus we assume the authentication problem is addressed using an existing method. This is a common approach, also adopted in [54, 55, 56, 57, 58, 59, 60, 61, 62, 63, 64].

5. The EXCHANge key-establishment protocol

In this section we provide a detailed description of the EXCHANge protocol, its implementation details, and a discussion on how to address run-time effects like packets losses and packets collisions.

5.1. Definitions and assumptions

We assume $\mathcal{A}$ and $\mathcal{B}$ to be two generic IoT devices that are able to communicate according through the IEEE 802.15.4 communication technology.

Let c be a vector of $2N$ random bits such that $c \xleftarrow[\$]{2N} [0;1]$. We use the notation $c \xleftarrow[\$]{2N} [0;1]$ to define the generation of $2N$ pseudo-random distinct elements from $[0;1]$ and assign them to c. Therefore, $\mathcal{A}$ and $\mathcal{B}$ interleave their transmission in a mutually exclusive fashion, i.e., if $c[i] = 1$ than $\mathcal{A}$ is the transmitting source, otherwise ($c[i] = 0$) $\mathcal{B}$ will transmit the message. Nevertheless, due to the distributed nature of the transmitting source selection, there might be a few cases for which $\mathcal{A}$ and $\mathcal{B}$ transmissions collide: we will deal with this case in Section 5.5.
We stress that c is not a shared bit-string between $\mathcal{A}$ and $\mathcal{B}$ but it is a mathematical structure to simplify the notation used throughout this paper. Indeed, our solution does not require any pre-shared secret but it only requests the two peers ($\mathcal{A}$ and $\mathcal{B}$) to generate two random bit strings (seeds) as it is depicted in the remaining of this section.

Let $K_A \xleftarrow[\$]{N} [0;1]$ and $K_B \xleftarrow[\$]{N} [0;1]$ be the secret seeds generated by $\mathcal{A}$ and $\mathcal{B}$, respectively, and being constituted by a sequence N of pseudo-random bits.

Definition 5.1. *Let $K_A[a(i)]$ ($K_B[b(i)]$) be the bit to be transmitted by $\mathcal{A}$ ($\mathcal{B}$) at time slot i, with:*

$$a(i) = \sum_{j=0}^{i} c[i] - 1; \quad b(i) = \sum_{j=0}^{i} \overline{c}[i] - 1; \quad \text{with } i \in [0; 2N-1] \tag{1}$$

where we use notation $\overline{x}$ to denote the binary complement of x.

Definition 5.2. *Let $e[i]$ be the* encrypted bit *transmitted during the time slot i, yielding:*

$$e[i] = \begin{cases} \overline{K}_A[a(i)] & \text{if } \mathcal{A} \text{ is transmitting} \\ K_B[b(i)] & \text{if } \mathcal{B} \text{ is transmitting} \end{cases} \tag{2}$$

Definition 5.3. *Let $K_S \in [0;1]^{2N}$ be the shared bit-string generated by the EXCHANge protocol, and $K_S[i]$ be the generated* shared *bit at time slot i, yielding:*

$$K_S[i] = c[i] \oplus e[i] \tag{3}$$

Therefore, each transmitted bit is encrypted with the identity of the source. As an example, recalling Figure 2, $c = [1; 1; 0; 0; 1; 0]$, $K_A = [0; 1; 0]$, $K_B = [1; 0; 1]$, $e = [1; 0; 1; 0; 1; 1]$, and $K_S = [0; 1; 1; 0; 0; 1]$. From Eq. (3), we observe that the value of the current transmitted bit $e[i]$ does not leak any useful information if the anonymity of the transmitting source is secret, i.e., if $c[i]$ is unknown to $\mathcal{E}$. It is worth noticing the difference among the following terms: $K_A[a(i)]$ ($K_B[b(i)]$) is the bit pseudo-randomly generated by $\mathcal{A}$ ($\mathcal{B}$), $e[i]$ is the encrypted bit transmitted at time slot i, and finally $K_S[i]$ is the shared bit generated at time slot i.
In order to deploy the EXCHANge protocol in a real IoT scenario, we need to modify three fields of the IEEE 802.15.4 radio message [65]. Firstly, as previously introduced, both the source and the destination addresses should be sanitized: we chose to set them to *00:00:00:00:00:00*. Finally, the encrypted bit $e[i]$ is stored in the least significant bit (LSB) of the MAC payload of the message. Figure 4 shows the details about the information injected by the EXCHANge protocol in a IEEE 802.15.4 radio message.

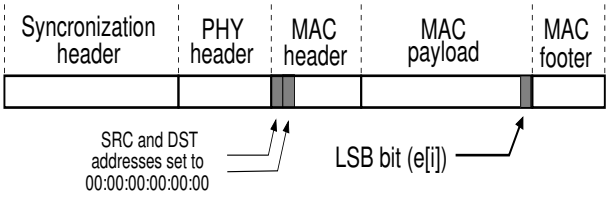


Figure 4: EXCHANge protocol: packet format for the IEEE 802.15.4 standard.

5.2. Finite state machine model

In this section we provide the details of a finite state machine (FSM) model of the EXCHANge protocol. We designed the FSM model with the aim of integrating the EXCHANge protocol within the TSCH mode of the IEEE 802.15.4 standard, that is currently recognized as one of the most successful enabling technologies at the basis of the IoT [66].
The protocol is characterized by three radio operative modes: *TX*, *RX*, and *SLEEP*. During the *SLEEP* mode the radio is inactive, while during the *TX* and the *RX* modes the radio is on, transmitting and receiving, respectively. Each status identified within the FSM performs a specific task. In detail:

- **PREP_TX**. A new message m is set up according to the IEEE 802.15.4 standard. Assuming the current slot as i, the LSB value of the MAC payload is set to the value of the encrypted bit $e[i]$, the message m is

loaded into the radio buffer, the frequency is set and a random transmission power is chosen.

- **RAND_GEN**. A random number $\stackrel{1}{\underset{\$}{\leftarrow}} [0; \Delta]$ is generated, Δ being a system parameter.
- **RAND_WAIT**. The node sets up a timer and waits a portion of time related with before initiating the transmission. While waiting, the node sets up its radio in the RX status. If no message is received before the expiring of the timer, then the node proceeds with the next status.
- **TX_GO**. The radio is switched between the RX and the TX status for initiating a new transmission.
- **TX_DATA**. The message m is transmitted to the peer.
- **RX_DATA**. The message m is received from the peer.
- **POST_TX**. The message m is further elaborated after the transmission (i.e., the LSB bit of the message payload is recorded and complemented (if needed), and local registers are cleared).
- **PARSE_RX**. The receiving node has now completed the reception of the message m. Therefore, the LSB of the message m, i.e., the encrypted bit $e[i]$, is extracted from the transmitted message m, decrypted and added to the shared secret key K.
- **PREP_ACK**. A new ack message is set up with a sequence number consistent with the last received message, the frequency is set and a random transmission power is chosen.
- **WAIT_ACK**. The node that sent the message waits for the peer to setting up the ack message.
- **TX_ACK** (**RX_ACK**) . The transmitting (receiving) node transmits (receives) the ack related to the last received message.
- **PARSE_ACK**. Processing of the received ack message.
- **RX_PROC**. Post-processing of the ack message just delivered.

5.3. The EXCHANge algorithm

The details of the EXCHANge protocol are depicted in Algorithm 1, by considering the statuses previously introduced. The *send* calls are non-blocking, while the *receive* calls are blocking and can be resumed by an external event when the associated boolean variable becomes true, i.e., depicted as $\downarrow$ *variable* in Algorithm 1. The end of the slot clears and resets the protocol parameters and variables, independently of its current status (line 24). Each of the node starts the protocol by choosing a random bit string, i.e., K_A for $\mathcal{A}$ and K_B for $\mathcal{B}$. At each instance i of the protocol, a random transmission power is selected (line 6) and a random waiting time—for a new message reception—is scheduled (lines 7–8). When the waiting time expires, the node initiates a new transmission of a message carrying a new secret bit: the bit is transmitted as complemented if $\mathcal{A}$ is the source, while it is transmitted as it is if the source is $\mathcal{B}$ (lines 9–13). Conversely, the peer receives, complements and stores the bit accordingly (lines 9-14). Finally, the receiving node acknowledges the last received bit by sending back to the transmitter the *ack* message with the same sequence number $sn[i]$ of the transmitted message (lines 16–17), by using a random transmission power. The peer receives the *ack* message and adds the secret bit to its own shared secret bit-string (lines 19–22). The sequence number is verified and if consistent a new bit is considered for transmission (lines 23–25), otherwise the current bit is re-transmitted. Finally, the shared bit-string K_S is hashed by a cryptographically secure hash function $H(\cdot)$ (lines 26-28). Note that the hash function is necessary to translate the shared bit-string (of variable length) in a fixed-length shared key.

5.4. Implementation of the FSM

The protocol Finite State Machine (FSM) illustrated in Section 5.2 has been implemented in a real IoT experimental board, the OpenMote-CC2538 [15]. It is a System on Chip (SoC) architecture, featuring a 32-bit Cortex-M3 microcontroller and a CC2520-like radio transceiver. The microcontroller runs up to 32 MHz and includes 32 Kbytes of RAM and 512 Kbytes of Flash. The radio operates at the 2.4 GHz band and is fully compatible with the IEEE 802.15.4 standard and later modifications (IEEE 802.15.4-2015).
Each slot of the IEEE 802.15.4 communication protocol lasts for 10ms, equivalent to 329 32-KHz radio tics. Therefore, we tuned the duration of each status in the FSM in order both to meet this high-level requirements about the duration of the whole slot, and carefully considering the speed of execution

Algorithm 1: EXCHANge protocol

let $K_A \xleftarrow[\$]{K} [0,1]$ and $K_B \xleftarrow[\$]{K} [0,1]$.
let p_m and p_M be the minimum and maximum transmission powers, respectively.
let $sn[i]$ be the sequence number of the current message;
let $2N$ be the size of the generated shared bit-string and the number of transmissions;
let $H(\,)$ be a cryptographic secure hash function;

/* **PREP_TX** */
/* The peers randomly choose the transmission power for the message and the ack. */
$T \xleftarrow[\$]{1} [p_m, p_M]$;
/* Select a random waiting time: ELAPSED is True when such time is elapsed, False otherwise. */
/* **RAND_GEN - RAND_WAIT - RX_DATA** */
ELAPSED = FALSE;
receive($e[i]$, $sn[i]$); // # ELAPSED
/* **TX_DATA - POST_TX** */
if *ELAPSED* **then**
    /* Send the i^{th} bit with seq. number $sn[i]$ */
    $e[i] = \overline{K}_A[a(i)]$; /* if $\mathcal{B}$: $e[i] = K_B[b(i)]$; */
    $sn = sn[i]$;
    send($e[i]$, $sn[i]$);
end
else
    /* **PARSE_RX** */
    /* Extract the secret bit from the received message */
    $K_S[i] = \overline{e}[i]$; /* if $\mathcal{B}$: $K_S[i] = e[i]$; */
    /* **PREP_ACK - TX_ACK** */
    /* Transmit the ack */
    send($sn[i]$);
end
if *ELAPSED* **then**
    /* **WAIT_ACK - RX_ACK** */
    /* Receive the ack */
    receive($sn[i]$);
    /* Add a new bit to the key */
    $K_S[i] = \overline{e}[i]$; /* if $\mathcal{B}$: $K_S[i] = e[i]$; */
end
/* **PARSE_ACK - RX_PROC** */
if $sn[i] == sn$ **then**
    $i = i + 1$;
end
if $i \quad 2N$ **then**
    /* Generate a shared key. */
    $K = H(K_S)$;
end
/* Wait for the current slot to expire. */
RESET;

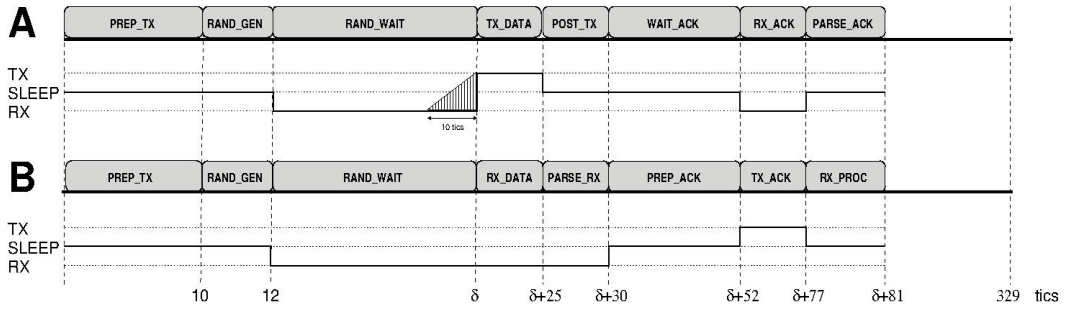


Figure 5: EXCHANge timeline.

Status name	Duration (tics)
PREP_TX	10
RAND_GEN	12
RAND_WAIT	$[0; \Delta]$
TX_DATA	25
RX_DATA	25
POST_TX	5
PARSE_RX	5
WAIT_ACK	22
PREP_ACK	22
RX_ACK	25
TX_ACK	25
PARSE_ACK	4
RX_PROC	4

Table 2: EXCHANge: Statuses of the finite state machine.

of the CC2538 SoC. The result of this tuning operation is depicted in Figure 5. Both $\mathcal{A}$ and $\mathcal{B}$ start in *sleep* mode, and switch to *RX* mode after 12 tics. Subsequently, both $\mathcal{A}$ and $\mathcal{B}$ schedule a random wait period: the first one ($\mathcal{A}$) reaching the period's end, it switches to the *TX* status initiating the transmission. Conversely, the second one ($\mathcal{B}$) remaining in the *RX* status, it receives the message. Eventually, the receiver ($\mathcal{B}$) acknowledges the packet reception with an *ack* message. The duration of each status of the FSM is included in Table 5.4.

As for the random number extraction process and the *RAND_WAIT* status, it is worth noting that only natural integers numbers can be extracted, and that the upper bound of the random wait needs to be carefully set in order not to exceed the time-slot duration of 10ms. Moreover, a further re-

striction is imposed by the time necessary to switch the radio between the RX an the TX mode, that is 10 tics. Given all these considerations, 23 possible choices are allowed.

5.5. Collisions and Lost packets

In this section we detail the features that make the EXCHANge protocol robust to collisions and packet loss.

Collisions. A collision event might happen when both $\mathcal{A}$ and $\mathcal{B}$ choose the same random wait time. Moreover, recalling Figure 5, we observe that during the random wait, a node switching from the *RX* to the *TX* status needs $_s = 10$ tics: this period of time is also a source for potential collisions, e.g., the nodes might switch concurrently and eventually collide. The probability of collision can be computed by observing that the **RAND_WAIT** period is a uniformly distributed random variable in the interval $[0; \Delta]$, with $\Delta = 236$, yielding:

$$P_C = \frac{1}{\lfloor \Delta = \ _s \rfloor} = \frac{1}{23} \approx 0{:}04 \tag{4}$$

When $\mathcal{A}$ and $\mathcal{B}$ collide for the transmission at time slot i^{th}, the *ack* message is not generated and eventually the protocol is instantiated again for the establishment of the same bit. If, instead, only one of the parties detects the collision, the presence of the sequence number in the exchanged packets allows for the detection of the collision event during the following exchange, and the recover of the agreement during the same round of the protocol.

Packet loss. Wireless communications are prone to packet losses due to multipath fading and low signal-to-noise ratio. Our protocol is robust to packet loss: the *ack* transmitted for each of the received bit guaranteeing that a new bit is added to the shared secret key only when both the peers have agreed on it. If either the message or the *ack* are lost during the i^{th} run of the protocol, no bit is added to the shared secret key and the protocol is instantiated again for the establishment of the i^{th} bit.

6. Performance Assessment

In this section we provide the results of a measurement campaign characterized by a benign scenario, i.e., without the presence of any adversary. Indeed, this section provides details about the performance of the EXCHANge protocol in terms of robustness to packet loss and required time to exchange a bit-string of a given length. At the same time we introduce several concepts

that will be used later on (Sec. 7) when testing the security of EXCHANge against a global eavesdropper adversary.

We remark that we adopted the OpenWSN protocol stack, an open-source implementation of a complete protocol stack for IoT architectures [14]. The protocol stack includes CoAP, RPL, and IEEE 802.15.4 in TSCH mode, thus being compatible with the majority of the commercial IoT devices. OpenWSN mainly consists of two modules: the firmware that runs on the devices, comprising the implementation of all the standards mentioned above, and OpenVisualizer, running on the host PC. The coordinator of the IoT network—working both as the gateway of the IoT network and as a debugging and monitoring tool—is connected to the host PC as well.

Firstly, we recall from Sec. 5.2 that each run of the EXCHANge protocol allows to generate (at most) one shared secret bit. In addition, it is executed within a single slot of the IEEE 802.15.4 protocol, lasting for 10ms. This means that, in order to generate 128 secret bits, we need at least 128×10 ms equal to 1.28s. Moreover, we also have to take into account possible collision events, introducing (on average) a 4% overhead, thus yielding to 1.33s (see Section 5.5). Therefore, the EXCHANge protocol requires 1.33s to generate a shared key of 128 bits. We will investigate the overhead related to energy and delay later on in the paper.

Figure 6 shows the time to generate a secret key of 128 bits as a function of

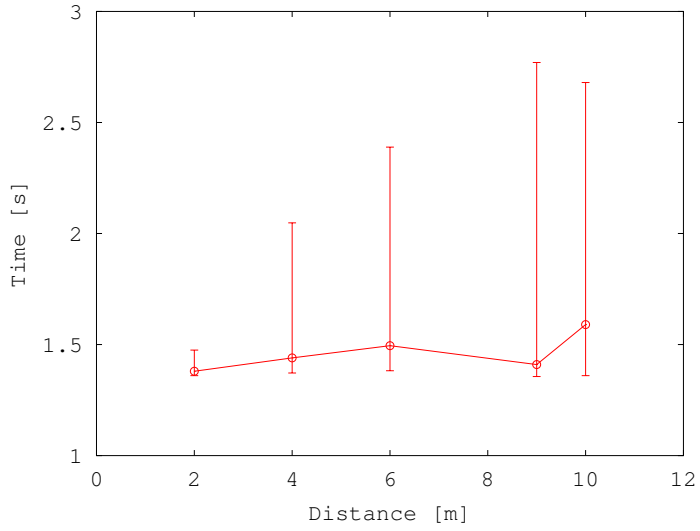


Figure 6: Time to generate a 128-bits shared bit-string (by performing 128 transmissions) varying the distance between the devices.

the distance between $\mathcal{A}$ and $\mathcal{B}$. The errorbars show the quantile 5, 50, and 95 related to 20 runs of the protocol. Increasing the distance between $\mathcal{A}$ and $\mathcal{B}$ introduces a significant packet loss, that in turn affects the time to establish the secret key. In particular, for long distances ($\geq$ 8m) we observe a high variance in the key-establishment process, due to the path loss and moving objects in the peers' surroundings. Finally, we highlight that on average the protocol needs about 140 slots (about 1.4 seconds) to exchange 128 bits.

The slot-frame size. The performance of the EXCHANge protocol also depend on the configuration parameters of the IEEE 802.15.4 protocol. In fact, the slots of the IEEE 802.15.4 protocol (that can be actually used to run the EXCHANge protocol) are grouped into slot-frames. The number of slots belonging to a slot-frame can be varied freely, according to the duty-cycle required for a given application and throughput considerations. In the tests that we conducted we chose a slot-frame size varying between 11 and 101, where 11 is the default value of the slot-frame size provided by the OpenWSN protocol stack that runs on the OpenMote-CC2538, while 101 is the slot-frame size recommended by the reference document [67] and widely used in literature [68], [69], [70], [71]. Therefore, recalling from the above section (i) that an average time of $t_p = 1.4$ seconds is needed to generate a shared bit-string of 128 bits, (ii) assuming the slot-frame as constituted by $N_{SF} \in [11, \ldots, 101]$ slots, and (iii) that the time t_P is needed by the EXCHANge protocol to generate a bit-string when N_P out of N_{SF} frames are dedicated to it, yields:

$$t_P = \left\lceil t_p \cdot \frac{N_{SF}}{N_P} \right\rceil$$

The above equation is depicted by Figure 7, showing the time (in seconds) necessary to generate a bit-string of 128 bits, assuming a IEEE 802.15.4 slot-frame of a given size, while varying the number of slots dedicated to the key establishment protocol.

In general, the more the number of slots dedicated to the key agreement protocol, the faster the protocol generates a key. We also observe that, given a slot-frame of size N_{SF}, the higher number of slots that can be dedicated to the protocol is $N_{SF} - 1$, because at least one slot is needed for guaranteeing standardized communication and tight synchronization between neighboring nodes. As anticipated before, the network administrator can choose the slotframe size and the number of active slots starting from considerations on the required duty-cycle and network throughput. As an example, we

consider an IEEE 802.15.4 network consisting of a number $N_N \geq 2$ of nodes in the same transmission range and a number N_S of slots in a slot-frame. At least N_N slots should be left free by the protocol, in order to allow the tight synchronization between the neighboring devices; the remaining $N_S - N_N$ slots could be used by the EXCHANge protocol. In addition, note that the running of the EXCHANge protocol does not affect the normal operation of the other devices in the network that are not involved in the key negotiation procedure. In fact, even assuming that these nodes have their radio on during the performing of the key agreement protocol, they simply discard a message with sanitized MAC addresses if they are not directly involved in a key negotiation procedure.

Finally, we observe from Fig. 7 that choosing a slot-frame size of $N_{SF} = 101$ slots with $N_P = 100$ slots dedicated to the EXCHANge protocol allows to generate a shared bit-string in about 1:42 seconds—that is pretty close to the optimal time that can be reached when considering a shared key of 128 bits.

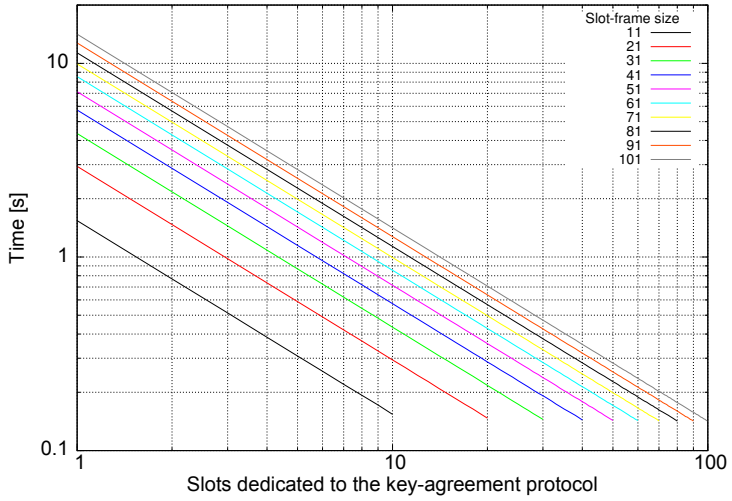


Figure 7: Time to generate a 128-bit shared key varying the number of slots dedicated to the EXCHANge protocol in the slot-frame.

Depending on the security requirements of the application, the EXCHANge protocol allows the two peers to generate a shared secret bit-string of arbitrary size. Figure 8 shows the average time needed to generate a bit-string of size spanning between 80 up to 1000 bits while varying the number of dedicated slots N_P of the EXCHANge protocol in a slot-frame of $N_{SF} = 11$ slots (default value for the OpenWSN protocol stack).

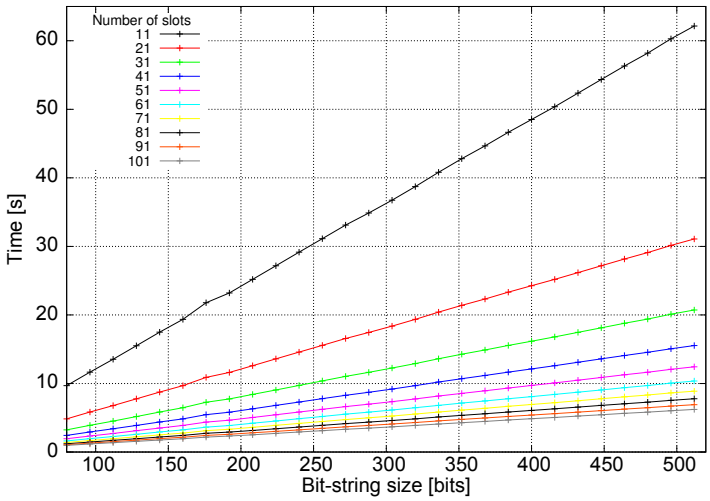


Figure 8: Time to transmit a bit-string of different sizes varying the number of dedicated slots in a slot-frame of 11 slots.

7. Security Analysis

The adversary considered in this section is fully passive: $\mathcal{E}$ can eavesdrop and log all the communications happening on all the frequencies of the radio spectrum and throughout the playground. Moreover, $\mathcal{E}$'s goal is to disclose the secret key that $\mathcal{A}$ and $\mathcal{B}$ generate during the consecutive runs of the EXCHANge protocol. To achieve her goal, the best strategy for $\mathcal{E}$ is to eavesdrop the transmitted messages and trying to infer on the transmitting source. In fact, recalling Eq. (3) we observe that the transmitted bit $e[i]$ is encrypted with the identity of the source $c[i]$, and therefore the value of the encrypted bit $e[i]$ does not leak any information about the current value of the transmitted bit $K_S[i]$.

Assuming $2N$ successful and secret transmissions yielding to a key length of $2N$ bits, the key space is given by:

$$P_D = 2^{2N} \tag{5}$$

To successfully disclose the secret key with 50% probability by random guessing, $\mathcal{E}$ has to perform on average $2^{2N\ -\ 1}$ trials, that is assumed to be computationally unfeasible when $2N \geq 80$ [72]. However, there can be cleverer attacks that can be played by $\mathcal{E}$ with the most dreadful one described in the following.

RSSI-based source identification. $\mathcal{E}$ can infer on the current source

identity by measuring the received signal strength (RSSI), and performing a proximity estimation, knowing the devices' deployment.

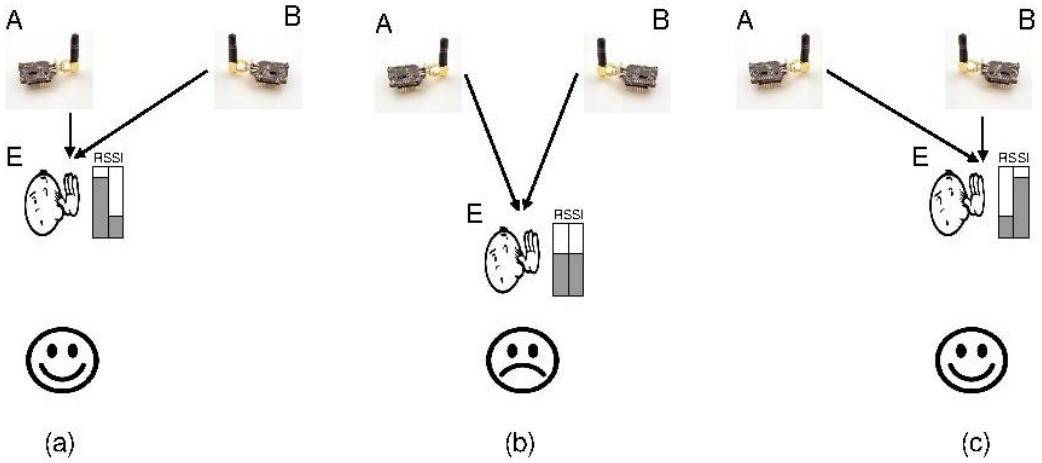
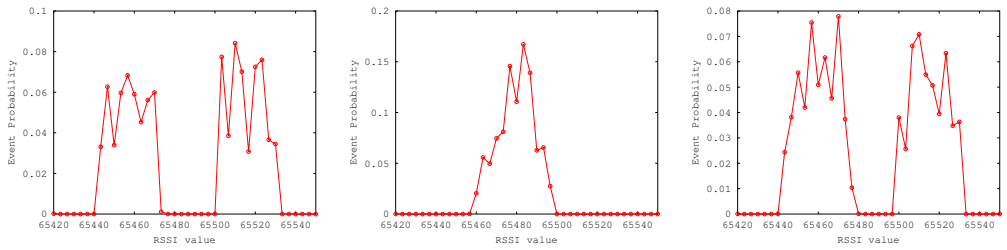


Figure 9: RSSI-based source identification: $\mathcal{E}$ measures the received signal strengths $R_{\mathcal{A}}$ and $R_{\mathcal{B}}$ from $\mathcal{A}$ and $\mathcal{B}$, respectively, and infers on the current transmitting source by looking at the different levels of the received power : (a) $R_{\mathcal{A}} \geq R_{\mathcal{B}}$, (b) $R_{\mathcal{A}} \approx R_{\mathcal{B}}$, and (c) $R_{\mathcal{A}} \leq R_{\mathcal{B}}$.

Figure 9 shows the RSSI-based attack implemented by the adversary. Assuming $\mathcal{E}$ is aware of the devices' layout, it can move closer to one of the two device (Figure 9(a) and (c)), hence being able to identify the transmitting source by performing an educated guess based on the received signal strength. Figure 10 shows the probability mass function associated to the received signal strength measured by $\mathcal{E}$ at the three different positions of Figure 9. $\mathcal{E}$ experiences higher RSSI values from the closer device while she measures lower signal strengths from the farther device: it is also worth noticing that RSSI values in Figure 10(a) and Figure 10(c) can be parted into two distinct sets corresponding to the transmitted messages by $\mathcal{A}$ and $\mathcal{B}$. Conversely, $\mathcal{E}$'s guessing capabilities are significantly reduced when the received signal levels from $\mathcal{A}$ and $\mathcal{B}$ have similar values (Figure 9(b) and Figure 10(b)). The toy example of Figure 9 assumes both $\mathcal{A}$ and $\mathcal{B}$ are transmitting with the same power. That is why we use random transmission power in EXCHANge. In particular, recalling Algorithm 1 (line 7), we highlight that the EXCHANge protocol mitigates the RSSI-based source identification by *randomizing the transmitting power*. Under such an assumption, i.e., random modulation of the transmission power, $\mathcal{A}$ and $\mathcal{B}$ can mimic different distances to $\mathcal{E}$, and

therefore, by shuffling their respective "fake" positions, they regain the source anonymity.



(a) (b) (c)

Figure 10: Probability mass function associated to the received signal strengths experienced by $\mathcal{E}$ during real measurements featuring the scenarios of Fig. 9.

7.1. Mitigating the RSSI-based source identification

Source identification can be mitigated by exploiting random transmission power, as in the EXCHANge protocol: since $\mathcal{E}$'s guessing is only based on the received signal strength, $\mathcal{A}$ and $\mathcal{B}$ might pretend to switch their positions at each round of the EXCHANge protocol by enforcing random transmission powers. Moreover, moving objects in the line-of-sight between $\mathcal{E}$ and the peers can drastically change the received signal strength experienced by $\mathcal{E}$ and therefore making her guessing even more challenging.
In this section, we evaluate the performance of the EXCHANge protocol in a real world scenario where $\mathcal{E}$ can get closer to a target node to increase her chances to discriminate the transmitting source by looking at the received signal strength and therefore guessing the values of the generated bits. We assume $\mathcal{A}$ and $\mathcal{B}$ can randomly change their transmission power and the playground is affected by moving objects such as people moving around and cutting the line-of-sight between $\mathcal{A}$ and $\mathcal{B}$.
Figure 11 shows $\mathcal{A}$ and $\mathcal{B}$ deployed at 6 m of distance from each other and 0.8 m of height from the ground, in an open area clear from obstructions and radio interferences. Then, we considered 4 different positions for $\mathcal{E}$ sitting at 3, 2, 1, and 0.5 m to $\mathcal{A}$, respectively. For each deployment, we transmitted 20 bit-strings of 128 bits for a total of about 2560 exchanged bits.

The best strategy to identify the actual transmitting source S relies on an educated guess that maps the current received signal strength R to the

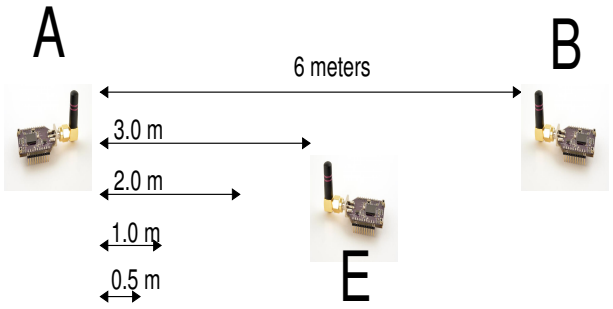


Figure 11: The Experimental set-up: $\mathcal{A}$ and $\mathcal{B}$ are placed at 6 meters far away while $\mathcal{E}$ is moved from 3m up to 0.5m close to $\mathcal{A}$.

identity of the transmitter (either $\mathcal{A}$ or $\mathcal{B}$). In particular, it is well known that the received signal strength becomes weaker when the distance between the transmitter and the receiver becomes larger [39]. Therefore, $\mathcal{E}$'s best strategy is to define a received signal strength threshold and assuming that the signals received with a strength exceeding the threshold have been transmitted from the closer device ($\mathcal{A}$), while those ones experiencing a received signal strength below the threshold have been transmitted by the farther device, yielding:

$$S = \begin{cases} \mathcal{A} & \text{if} \quad R \geq \\ \mathcal{B} & \text{if} \quad R < \end{cases}$$

The value of the threshold can be empirically evaluated by considering the average value of all the possible received signal strengths for a specific scenario [12], yielding:

$$= R_{min} + \frac{R_{max} - R_{min}}{2}$$

where R_{max} and R_{min} are the maximum and minimum RSS values experienced by $\mathcal{E}$ for a specific deployment scenario.
As an example of the above procedure, let us consider Figure 12 referring to the scenario of Figure 11 where $\mathcal{E}$ is at a distance of 1.5m from $\mathcal{A}$. Black lines shows the probability mass function (p.m.f.) associated to the received signal strengths experienced by $\mathcal{E}$ while the red line shows the threshold (= 65459) computed by the adversary to discriminate between the sources.

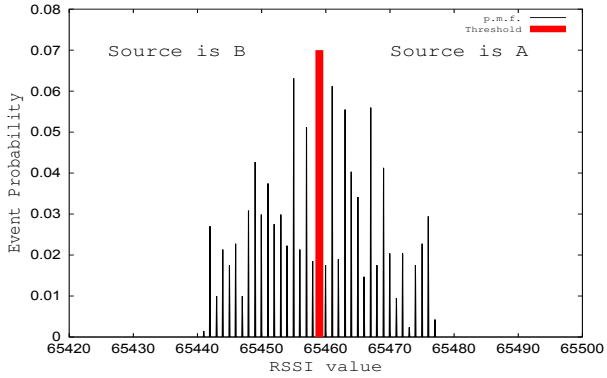


Figure 12: An example of RSSI-based source identification: $\mathcal{E}$ splits the set of experienced RSS values into two subsets. Then, she decides that the smaller values (on the left of the red bar) are from the farther device ($\mathcal{B}$) while the bigger ones (on the right of the red bar) are from the closer node ($\mathcal{A}$).

$\mathcal{E}$ will decide that the signals with received signal strength over the threshold have been transmitted by $\mathcal{A}$ while the others have been transmitted by $\mathcal{B}$; she will change the values of the previously eavesdropped bit accordingly—we recall that the bits transmitted by $\mathcal{A}$ have to be complemented.
Now, for each of the $\mathcal{E}$'s positions depicted in Figure 11, we counted the number of secret bits, i.e., not-guessed by $\mathcal{E}$ applying the above procedure. Figure 13 shows the quantile 5, 50, and 95 computed over the number of bits not-guessed by the adversary in a sequence of 20 runs of the protocol assuming 128 transmissions at each round, as a function of the position of $\mathcal{E}$: 0.5, 1, 2, and 3m to $\mathcal{A}$. We observe that the number of secret bits that are kept effectively secret increases when $\mathcal{E}$ moves farther away from $\mathcal{A}$: indeed, $\mathcal{E}$ decreases her chances to guess the correct values of the bits when she moves away from the target node ($\mathcal{A}$) up to her worst case positioning (in the middle of the scene), 3m away from each device; in this location her guessing capabilities are just slightly better than random guess, i.e., about 52 bits on average (the rightmost point of the curve in Figure 13) out of 128 bits. Moreover, we observe that the variance of the secret bits tends to reduce for greater distances: this phenomenon can be explained by resorting to environmental factors. Indeed, when $\mathcal{E}$ is very close to $\mathcal{A}$ (0.5 meters), people moving around contribute either constructively or destructively to the guessing process introducing a significant variance in the guessing process. Conversely, when $\mathcal{E}$ moves farther away from $\mathcal{A}$, the received signal strength is more stable due the longer distances between the nodes and the moving

objects and therefore the variance is significantly tighter.
It is worth noting that the information implied from Figure 13 can have just a statistical significance for $\mathcal{E}$—however, their importance is not negligible, as reported in the following. Indeed, assuming a conservative stance, $\mathcal{E}$ is able to derive the same data reported in Figure 13 and she is at 1m from $\mathcal{A}$. From Figure 13, she knows that (on the average) 48 bits of the keys are kept secret, while 80 have been correctly guessed: thus, the entropy of this key is 48. However, she is not aware of which bits are the correct ones and which ones have to be flipped to derive the correct key—the information derived from Figure 13 is purely a statistical one. All she knows is that, on average, she has to make 2^{47} decryption attempts over a single eavesdropped encrypted bit-string to recover the secret key with 50% probability.

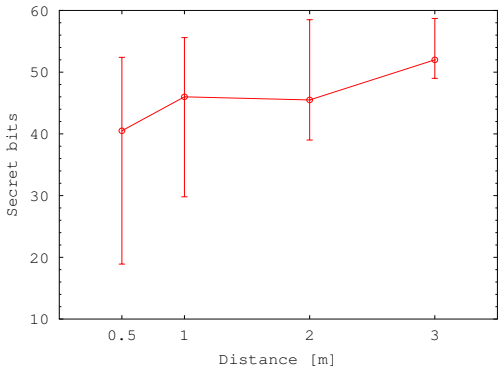


Figure 13: $\mathcal{E}$ guessing capabilities: quantile 5, 50, and 95 associated to the number of bits not-guessed by $\mathcal{E}$ (secret bits) during the RSSI-based source identification attack on a shared key of 128 bits varying the distance between $\mathcal{E}$ and $\mathcal{A}$ as depicted in Figure 11.

Starting from the above results, we computed the number of transmissions in order to guarantee at least 80 secrets bits, i.e., an entropy of 80, the number of secret bits needed to meet the security level in today's wireless communications [72]. We consider the lower bounds (best case for $\mathcal{E}$) from Figure 13 and we evaluate the number of requested transmissions in order to obtain 80 secret bits. Figure 14 shows the number of transmissions requested to generate an entropy of at least 80 (at least 80 secret bits) as a function of the distance between $\mathcal{A}$ and $\mathcal{E}$. We observe that the EXCHANge protocol requires about 209 transmissions when $\mathcal{E}$ is far away from the devices, i.e., when the distance is greater than 3 meters, while it requires an increasing

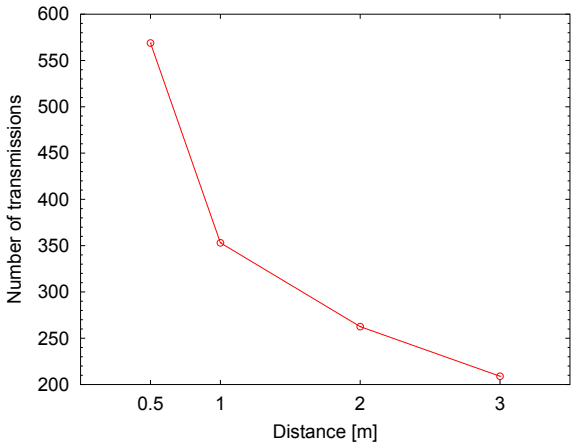


Figure 14: Maximum number of transmissions requested to establish at least 80 secret bits as a function of the distance between $\mathcal{A}$ and $\mathcal{E}$.

amount of transmissions when $\mathcal{E}$ moves closer to the target node: we measured about 569 transmissions (on average) to establish (at least) 80 secret bits in a shared key of 128 bits, when $\mathcal{E}$ is only 50 cm far away from the target node ($\mathcal{A}$).

Finally, we emphasize that the transmission of the acknowledgement does not allow the attacker to improve its guessing chances. In fact, the transmission power of the ack message is randomized, too, thus not providing any direct information about its sender (indeed, same considerations as per standard messages of the EXCHANge protocol applies for the ack, too).

7.2. Detecting Active Attacks

In this section, we consider an *active adversary* that is able to transmit messages during the execution of the EXCHANge protocol. We stress that device authentication is out of the scope of EXCHANge (and this work); nevertheless, we show that EXCHANge enables the two parties to detect a third malicious entity ($\mathcal{E}$) willing to bias the secret key generation process. Indeed, hereby we consider an adversary willing to participate to the key-establishment process—in a stealthy manner—with the goal to bias it in a way to increase his chances to guess the final shared secret key. Given the nature of the radio channel, this is a very challenging behaviour to detect. In the following, we prove that EXCHANge enables the two participating parties to statistically detect such an attack.

In the following, we assume $\mathcal{A}$ and $\mathcal{B}$ want to perform a secret key estab-

lishment in the presence of $\mathcal{E}$, an active adversary able to eavesdrop all the radio communications and to transmit radio messages. Indeed, we observe that an active adversary might participate to the key-establishment process biasing a certain number of bits and increasing her chances to guess the final secret key.

Therefore, the adversary might implement a two-stage attack: first, biasing the secret bits during the first part, and later proxying the messages between $\mathcal{A}$ and $\mathcal{B}$ during the second part—that is, realizing a typical man-in-the-middle attack. Assuming the above attack, in order to maximize its effectiveness, $\mathcal{E}$ might want to tamper with all the bits of the key. To do so, the adversary has to win all the contentions of the EXCHANge protocol: this is indeed possible by setting the waiting time to zero.

Now, we have the following attack scenario: $\mathcal{E}$ is deciding all the bits of the protocol, while $\mathcal{A}$ and $\mathcal{B}$ participate to the protocol as receivers only. As stated before, at the end of the protocol, $\mathcal{A}$ and $\mathcal{B}$ keys differ while $\mathcal{E}$ is aware of both keys and therefore can proxy the messages between the nodes, which in turn have their secrecy compromised. In the following, we show how $\mathcal{A}$ and $\mathcal{B}$ can easily detect the above attack by measuring the *fairness of the contentions* experienced during the run of the EXCHANge protocol.

Let x_i be a random variable that takes on the value 0 (1) when $\mathcal{A}$ ($\mathcal{B}$) wins the contention algorithm for the transmission of the next bit. Moreover, let $X = \sum_N x_i$ be the random variable counting the number of times $\mathcal{B}$ wins the contention over a batch of N transmissions. Since each round of the EXCHANge protocol is independent, we can assume x_i to be an i.i.d random variable.

Hence, the random variable X obeys to a binomial distribution. Assuming N consecutive rounds of the EXCHANge protocol, the Chernoff inequality for the binomial distribution [73] yields:

$$P\left(|X \leq (1 - \quad) \quad |\right) \leq e^{\quad \frac{\cdot^2}{2}}; \tag{6}$$

where is the mean of the distribution and is a deviation factor from the mean ($1 \leq \quad \leq 0$). For a given value of , it is possible to compute the Chernoff bound representing the upper bound probability to experience a given number of wins in the contention algorithm. Assuming a fair contention, and therefore a winning probability of $p = 0{:}5$, Fig. 15 shows the occurrence probability of a given , with different bit-string lengths.

Our results demonstrate that high values of , i.e., $> 0{:}5$ (i.e. the

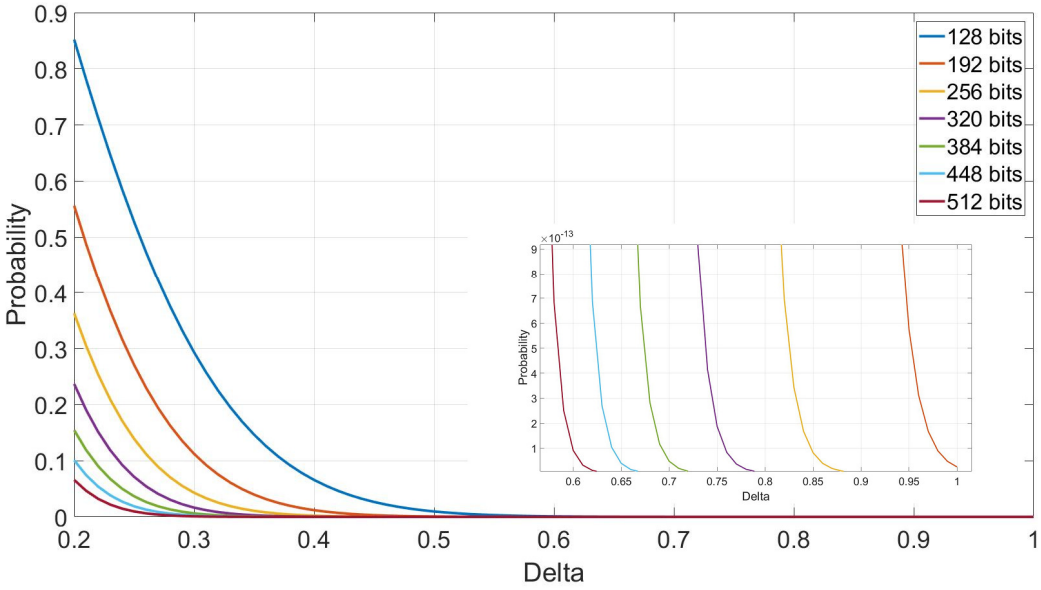


Figure 15: Chernoff bound on win probabilities by a party in the EXCHANge protocol, assuming different lenghts of the shared bit-string.

number of contention actually won by $\mathcal{B}$ is less than half of the expected ones), are very unlikely—and this probability further lessen as N increases.

Therefore, the legitimate parties ($\mathcal{A}$ and $\mathcal{B}$) can use this bound by setting a (very low) threshold in the probability of occurrence of a given number of wins by the other party in the contention algorithm. Indeed, if one of the two parties exceeds the average number ( ) of won contention by more than a fraction, the run is considered insecure and a new key establishment will be initiated. The threshold value should be decided as a trade-off between discarding too many keys (false positive) and detecting on-going attacks (true positive).

For instance, consider a run in which the two parties perform a total of 128 transmissions and one of the two parties wins the contention algorithm 100 times out of the total 128. The value of for this event is = 0:56. By looking at the magnification in Fig. 15, we observe that the above event happens with a probability equal to $2{:}5 * 10^{-6}$. Therefore, assuming a threshold set to $T = 0{:}00001$, since this probability is lower than the threshold, the run will be considered suspicious, its output will be discarded, and a new run of the EXCHANge protocol will be initiated.

7.3. Robustness to Denial of Service attacks

Being performed over a wireless channel, EXCHANge is vulnerable to Denial of Service (DoS) attacks, aiming at disrupting over-the-air messages. The most common and effective DoS attack is *jamming*, where the adversary injects random noise into the wireless channel to prevent message exchange. Over the years, several solutions have been developed to mitigate jamming, with the vast majority of them resorting to a previously established secret between the communicating parties [74], [75].

However, there is a small set of solutions that does not require pre-shared secrets while exploiting either uncoordinated frequency hopping [76] or probabilistic pairing [77], hence thwarting the DoS threat.

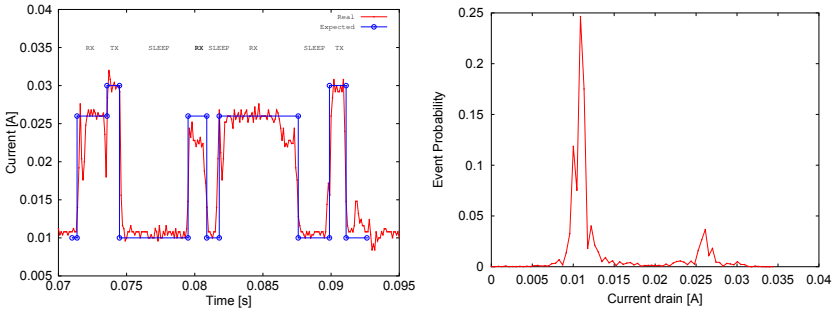
The adversary might also be interested to run multiple consecutive runs of the EXCHANge protocol, and therefore accomplishing a *battery depletion attack*. However, this situation could be easily mitigated on the device side, e.g., by monitoring the number of key establishment requests and bounding them to a certain number per given period of time. The computation of the upper bound can be based, for instance, on the number of neighboring devices, or based on the maximum energy consumption allowed for the EXCHANge protocol (see Sec. 8 for more details).

8. Energy analysis

This section introduces the energy consumption analysis of the EXCHANge protocol.

To measure the current drawn by the EXCHANge protocol we used a RIGOL DS1052E digital oscilloscope, by sampling the voltage to the terminals of a 1Ω probe resistor connected to the current sense port of the OpenMote OpenBase. The RIGOL DS1052E has a vertical resolution of 8 bits, the vertical range has been set to 10mV/div while the horizontal range has been set to 2ms/div. We sampled several runs of the protocol and subsequently exported the data to GNU Octave for the analysis. Firstly, we identified and isolated one run of the EXCHANge protocol as depicted in Figure 16(a). By comparing Figure 16(a) with Figure 5, it can be observed as the device initially behaves as $\mathcal{A}$, transmitting one bit of the bit-string, and subsequently it behaves as $\mathcal{B}$, receiving another bit. The final transmission represents the *ack* message transmitted to acknowledge the last received bit. Figure 16(a) also depicts the current drain of the protocol during the three different statuses. We identified three different current drain levels, each one corresponding to

a different device status, i.e., *SLEEP*, *RX*, and finally *TX*. In order to precisely estimate the current drains, we performed a statistical analysis about the current drain over all the collected measurements. Figure 16(b) shows the probability mass function associated to the current drain values sampled during extensive runs of the EXCHANge protocol. We observe three major peaks: about 10mA when the device is in the *SLEEP* status, about 26mA when the device is in the *RX* status, and finally, about 30mA when the device is in the *TX* status. The above values are also confirmed by the OpenMote-CC2538 data-sheet [78].



(a) (b)

Figure 16: (a) Current drain of one run of the EXCHANge protocol, (b) Probability mass function associated to the current drain sampled during multiple runs of the EXCHANge protocol.

Finally, we are able to estimate the energy consumption of the EXCHANge protocol by considering the time spent by the device in each of the statuses. We also notice that during our measurements, we did not observe any significant differences in the current drain when changing the transmission power, and therefore we assume as constant and equal to 30mA the current consumption when the device is in the *TX* status.
The energy consumption can be computed by integrating the instantaneous current drain $i(t)$ over the time and multiplying it by 3.3V, i.e., the voltage of the OpenMote OpenBase board [78], yielding:

$$E[\text{mJ}] = 3.3\text{V} \int_{1}^{1} i(t)dt \tag{7}$$

Evaluating Eq. (7) over the values extracted from Figure 16(b) yields to an average power consumption of about 45 mJ. This value is particularly significant if compared with the consumption of the radio during the *SLEEP*,

TX and RX modes, i.e., about 33, 99 and 86mJ by recalling Figure 16(a). Therefore, recalling the results from Figure 14, and assuming a consumption of 45mJ per exchanged bit, Table 8 resumes the major trade-offs of the EXCHANge protocol, assuming at least 80 secret bits in a shared key of 128 bits and a typical battery storage capacity of 9000J (AA-type)[79].

% of battery lifetime	0.14	0.09	0.07	0.06
$\mathcal{A}$-$\mathcal{E}$ distance [m]	0.5	1	2	3

Table 3: Protocol consumption assuming a 128 bits secret key with at least 80 secret bits and estimation of the battery percentage consumed by one run of the protocol, assuming the mote is powered by two AA batteries.

9. Discussion and Remarks

Key entropy. Contrary to other approaches, such as either RSSI or CSI-based key-establishments, the entropy of the key generated by the EXCHANge protocol does not strictly depend on multipath fading conditions. Other solutions such as [16], [17],[25] exploit temporal fluctuations of either the received signal strength or the channel state to generate secret bits: unfortunately, the wireless channel can be static for long periods, hence significantly delaying the key-establishment process—it has to wait for channel conditions with sufficient entropy. EXCHANge exploits the randomness of the internal pseudo-random number generator. Hence, it does not rely on any external source of entropy, achieving performance that are independent of the channel conditions.

Resource constraints. It has been shown as the EXCHANge protocol is particularly lightweight in terms of both memory footprint, CPU burden, and bandwidth. In particular, considering the CC2538 architecture, it requires just 4356 Bytes and 44 Bytes of ROM and RAM, respectively, and only one hash computation. As for the bandwidth, we showed that EXCHANge can be run by using only one out of the 11 slots in the IEEE 802.15.4 protocol, while the required energy to establish a secret key of 128 bits with at least 80 secret bits requires the 0.14% of the battery capacity in the worst case scenario, i.e., when $\mathcal{E}$ is only 50 cm away from the target node ($\mathcal{A}$).

Robustness to device tampering. The peculiarities of the EXCHANge protocol, such as high key entropy and low resources request, allow frequent re-generation of the secret key between the two peers. This makes the EXCHANge protocol particularly suitable for scenarios where the adversary $\mathcal{E}$ is allowed to compromise the device and disclose all the secrets in its memory. The adoption of our solution makes device compromising and secrets' disclosure useless from the attacker point of view: in fact the EXCHANge protocol does not need any long-term secret to be stored in memory.

Hardware portability. While the results provided in Sec. 6 have been obtained by using the OpenMote-CC2538 hardware platform, they are indeed valid for any other hardware platform featuring the IEEE 802.15.4 protocol, such as the new CC2650[1]. In addition, although EXCHANge has been implemented and evaluated assuming the use of the IEEE 802.15.4-2015 standard in the Time Synchronized Channel Hopping (TSCH) mode, its design is very general and it can be integrated also in the legacy mode of the IEEE 802.15.4 technology. In fact, as already stated before, EXCHANge does not resort to specific PHY and MAC layer peculiarities, but only to a generic synchronization protocol that allows neighboring devices to be synchronized on a time-slot basis. Our solution paves the way to other implementations in different communication technologies, such as Bluetooth Low Energy (BLE), given the presence of a bidirectional communication channel that can be used for the messages exchange. Finally, EXCHANge can be also implemented in the new Long Range (LoRa) networks, given the presence of a bidirectional communication channel. As a general comment, we recommend the adoption of EXCHANge to all the scenarios characterized by high mobility and fast context changes, since *signal fading* significantly affects (negatively) the adversary chances to guess the secret key.

Security. We proved the robustness of the EXCHANge protocol to the RSSI-based source identification attack by an extensive measurement campaign and we provided a lower bound to the number of transmissions (about 569) to generate a secret key of 128 bits with at least 80 secure bits in very harsh scenario conditions, i.e., $\mathcal{E}$ was placed very close (50 cm) to $\mathcal{A}$, while $\mathcal{B}$ was 6 meters far away from $\mathcal{A}$. In addition, we proved the capability

[1] `http://www.ti.com/product/CC2650`

of the EXCHANge protocol to detect and effectively reject active attacks, by leveraging statistical considerations about the fairness of the contention algorithm in each run of the protocol.

10. Conclusion

This paper proves, for the first time, that channel anonymity can be effectively used to generate a shared secret key and that this technique can be successfully used to provide key agreement in real IoT deployments. The proposed solution relies neither on any pre-shared information among devices, nor on any pre-loaded information (per device), nor on any static, long-term information stored on the device itself. These stateless properties of the protocol reduce the incentive for an adversary to physical compromise the device. Further, we have also shown how this protocol can detect an active external adversary attempting to tamper with the key establishment, even if the channel is not authenticated—that is, providing an effective solution to the Men in the Middle Attack in this specific setting. What is more, the EXCHANge protocol has been shown, via real-world implementation, to be a truly viable solution to secure the current and coming IoT devices. For instance, it enables the exchange of an 80-bits shared secret (under adverse system conditions) with as little of 569 one-bit messages, and just the 0.14% of the battery lifetime, while requiring a minimal memory footprint. Under less harsh conditions, the same objectives can be achieved with just 280 one-bit messages and requiring only the 0.06% of the battery lifetime.
All the above features pave the way for further research on channel anonymity in general, and to their specific applications to the IoT setting.

Acknowledgements

This work was framed in the context of the project SymbIoTe, which receives funding from the European Union's Horizon 2020 research and innovation programme under grant agreement 688156. The findings achieved herein are solely responsibility of the authors.

References

[1] L. Grieco, A. Rizzo, S. Colucci, S. Sicari, G. Piro, D. Di Paola, and G. Boggia, "IoT-aided robotics applications: technological implications,

target domains and open issues," *Elsevier Computer Communications*, vol. 54, Dec. 2014.

[2] O. Hersent, D. Boswarthick, and O. Elloumi, *The Internet of Things: Key Applications and Protocols*. Wiley, 2012.

[3] R. Di Pietro, S. Guarino, N. V. Verde, and J. Domingo-Ferrer, "Security in wireless ad-hoc networks - A survey," *Elsevier Computer Communications*, vol. 51, pp. 1–20, 2014.

[4] D. Evans, "The Internet of Things, How the Next Evolution of the Internet Is Changing Everything," Cisco Internet Business Solutions Group (IBSG). White paper, Apr. 2011.

[5] S. Sicari, A. Rizzardi, L. Grieco, and A. Coen-Porisini, "Security, privacy and trust in Internet of Things: The road ahead ," *Computer Networks*, vol. 76, pp. 146 – 164, 2015.

[6] J. Granjal, E. Monteiro, and J. S. Silva, "Security for the Internet of Things: A Survey of Existing Protocols and Open Research Issues," *IEEE Communications Surveys Tutorials*, vol. 17, no. 3, pp. 1294–1312, 3rd Quarter 2015.

[7] M. Wilhelm, I. Martinovic, and J. B. Schmitt, "Secret keys from entangled sensor motes: implementation and analysis," in *ACM Conference on Security and Privacy in Wireless and Mobile Networks (WiSec)*. ACM, 2010, pp. 139–144.

[8] C. Castelluccia and P. Mutaf, "Shake them up!: a movement-based pairing protocol for cpu-constrained devices," in *International Conference on Mobile Systems, Applications, and Services (MobiSys)*. ACM, 2005, pp. 51–64.

[9] S. Mirzadeh, H. Cruickshank, and R. Tafazolli, "Secure device pairing: A survey," *IEEE Communications Surveys and Tutorials*, vol. 16, no. 1, pp. 17–40, 2014.

[10] C. Bormann, M. Ersue, and A. Keranen, "Terminology for constrained-node networks," RFC 7228, May 2014.

[11] P. Barsocchi, G. Oligeri, and C. Soriente, "SHAKE: Single HAsh key establishment for resource constrained devices," *Ad Hoc Networks*, pp. 288 – 297, 2013.

[12] R. Di Pietro and G. Oligeri, "COKE: Crypto-less over-the-air key establishment," *IEEE Transactions on Information Forensics and Security*, vol. 8, no. 1, pp. 163 –173, Jan. 2013.

[13] R. Di Pietro and G. Oligeri, "ESC: An efficient, scalable, and crypto-less solution to secure wireless networks," *Elsevier Computer Networks*, vol. 84, pp. 46–63, Jun. 2015.

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

[15] X. Vilajosana, P. Tuset, T. Watteyne, and K. Pister, "OpenMote: Open-Source Prototyping Platform for the Industrial IoT," in *Proc. of 7th International Conference on Ad Hoc Networks (AdHocHets 2015)*, San Remo (Italy), Sep. 2015.

[16] K. Liu, S. Primak, and X. Wang, "On secret key generation from multiple observations of wireless channels," in *IEEE International Conference on Communication Systems (ICCS)*, Nov. 2014, pp. 147–151.

[17] J. Zhang, R. Woods, A. Marshall, and T. Q. Duong, "An effective key generation system using improved channel reciprocity," in *IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, Apr. 2015, pp. 1727–1731.

[18] J. Zhang, A. Marshall, R. Woods, and T. Q. Duong, "Secure key generation from OFDM subcarriers' channel responses," in *IEEE Globecom Workshops (GC Wkshps)*, Dec. 2014, pp. 1302–1307.

[19] J. Zhang, R. Woods, A. Marshall, and T. Q. Duong, "Verification of key generation from individual OFDM subcarrier's channel response," in *IEEE Globecom Workshops (GC Wkshps)*, Dec. 2015, pp. 1–6.

[20] H. Zhou, L. M. Huie, and L. Lai, "Secret key generation in the two-way relay channel with active attackers," *IEEE Transactions on Information Forensics and Security (TIFS)*, vol. 9, no. 3, pp. 476–488, Mar. 2014.

[21] T. H. Chou, S. C. Draper, and A. M. Sayeed, "Secret Key Generation from Sparse Wireless Channels: Ergodic Capacity and Secrecy Outage," *IEEE Journal on Selected Areas in Communications*, vol. 31, no. 9, pp. 1751–1764, Sep. 2013.

[22] M. G. Madiseh, S. W. Neville, and M. L. McGuire, "Applying beamforming to address temporal correlation in wireless channel characterization-based secret key generation," *IEEE Transactions on Information Forensics and Security (TIFS)*, vol. 7, no. 4, pp. 1278–1287, Aug. 2012.

[23] C. T. Zenger, M. Pietersz, and C. Paar, "Preventing relay attacks and providing perfect forward secrecy using physec on 8-bit c," in *IEEE International Conference on Communications Workshops (ICC)*, May 2016, pp. 110–115.

[24] P. Huang and X. Wang, "Fast secret key generation in static wireless networks: A virtual channel approach," in *IEEE International Conference on Computer Communications (INFOCOM)*, Apr. 2013, pp. 2292–2300.

[25] W. Xi, X.-Y. Li, C. Qian, J. Han, S. Tang, J. Zhao, and K. Zhao, "KEEP: Fast secret key extraction protocol for D2D communication," in *IEEE International Symposium of Quality of Service (IWQoS)*, May 2014, pp. 350–359.

[26] R. Wilson, D. Tse, and R. A. Scholtz, "Channel Identification: Secret Sharing using Reciprocity in Ultrawideband Channels," in *IEEE International Conference on Ultra-Wideband (ICUWB)*, Sep. 2007, pp. 270–275.

[27] F. Marino, E. Paolini, and M. Chiani, "Secret key extraction from a uwb channel: Analysis in a real environment," in *IEEE International Conference on Ultra-WideBand (ICUWB)*, Sep. 2014, pp. 80–85.

[28] S. T. B. Hamida, J. B. Pierrot, B. Denis, C. Castelluccia, and B. Uguen, "On the Security of UWB Secret Key Generation Methods against Deterministic Channel Prediction Attacks," in *IEEE Vehicular Technology Conference*, Sep. 2012, pp. 1–5.

[29] W. Xi, C. Qian, J. Han, K. Zhao, S. Zhong, X.-Y. Li, and J. Zhao, "Instant and Robust Authentication and Key Agreement Among Mobile Devices," in *Proc. of ACM Conference on Computer and Communications Security (SIGSAC)*, 2016, pp. 616–627.

[30] S. Eberz, M. Strohmeier, M. Wilhelm, and I. Martinovic, "A Practical Man-In-The-Middle Attack on Signal-Based Key Generation Protocols," in *17th European Symposium on Research in Computer Security (ESORICS)*, ser. Lecture Notes in Computer Science, vol. 7459. Springer, sep 2012, pp. 235–252.

[31] W. Xu, G. Revadigar, C. Luo, N. Bergmann, and W. Hu, "Walkie-Talkie: Motion-Assisted Automatic Key Generation for Secure On-Body Device Communication," in *ACM/IEEE Int. Conf. on Information Processing in Sensor Networks (IPSN)*, Apr. 2016, pp. 1–12.

[32] W. Xu, C. Javali, G. Revadigar, C. Luo, N. Bergmann, and W. Hu, "Gait-Key: A Gait-Based Shared Secret Key Generation Protocol for Wearable Devices," *ACM Transactions on Sensor Networks (TOSN)*, vol. 13, no. 1, pp. 1–27, Jan. 2017.

[33] G. Revadigar, C. Javali, W. Xu, A. V. Vasilakos, W. Hu, and S. Jha, "Accelerometer and Fuzzy Vault-Based Secure Group Key Generation and Sharing Protocol for Smart Wearables," *IEEE Transactions on Information Forensics and Security*, vol. 12, no. 10, pp. 2467–2482, Oct. 2017.

[34] D. Schrmann, A. Brsch, S. Sigg, and L. Wolf, "BANDANA: Body area network device-to-device authentication using natural gAit," in *IEEE Int. Conf. on Perv. Comput. and Commun. (PerCom)*, Mar. 2017, pp. 190–196.

[35] L. Shi, J. Yuan, S. Yu, and M. Li, "ASK-BAN: Authenticated Secret Key Extraction Utilizing Channel Characteristics for Body Area Networks," in *Proc. of ACM Conference on Security and Privacy in Wireless and Mobile Networks*, 2013, pp. 155–166.

[36] G. Aliberti, R. Di Pietro, and S. Guarino, "Reliable and perfectly secret communication over the generalized Ozarow-Wyners wire-tap channel,"

Computer Networks, vol. 109, Part 1, pp. 21 – 30, 2016, Special Issue on Recent Advances in Physical-Layer Security .

[37] A. Pfitzmann and M. Köhntopp, *Anonymity, Unobservability, and Pseudonymity | A Proposal for Terminology*, 2001, pp. 1–9.

[38] D. Chaum, "The Dining Cryptographers Problem: Unconditional Sender and Recipient Untraceability," *Journal of Cryptology*, vol. 1, no. 1, pp. 65–75, Mar. 1988.

[39] T. Rappaport, *Wireless Communications: Principles and Practice*. Upper Saddle River, NJ, USA: Prentice Hall PTR, 2001.

[40] M. Young, "A Secure and Usefull Keyless Cryptosystem," in *Information Processing Letter*, 1985, pp. 35–38.

[41] B. Alpern and F. B. Schneider, "Key exchange using keyless cryptography," Tech. Rep., 1982.

[42] W. Diffie and M. E. Hellman, "New Directions in Cryptography," *IEEE Transactions on Information Theory*, vol. 22, no. 6, pp. 644–654, Nov. 1976.

[43] Y. Huang, Y. Wang, R. Zhu, X. Chen, and Q. Meng, "Synchronized contention windows-based backoff algorithm in IEEE 802.11 wireless networks," in *International Conference on Computer, Information and Telecommunication Systems (CITS)*, Jul. 2016, pp. 1–5.

[44] R. Di Pietro, G. Oligeri, C. Soriente, and G. Tsudik, "United We Stand: Intrusion Resilience in Mobile Unattended WSNs," *IEEE Transactions on Mobile Computing*, vol. 12, no. 7, pp. 1456–1468, Jul. 2013.

[45] R. Di Pietro, G. Oligeri, C. Soriente, and G. Tsudik, "Intrusion-Resilience in Mobile Unattended WSNs," in *IEEE International Conference on Computer Communications (INFOCOM)*, Mar. 2010, pp. 1 –9.

[46] R. Di Pietro, G. Oligeri, C. Soriente, and G. Tsudik, "Securing Mobile Unattended WSNs against a Mobile Adversary," in *IEEE Symposium on Reliable Distributed Systems*, Nov. 2010, pp. 11–20.

[47] H. Chan, A. Perrig, and D. Song, "Random key predistribution schemes for sensor networks," in *Proc. of Symposium on Security and Privacy*, May 2003, pp. 197–213.

[48] K. B. Rasmussen and S. Capkun, "Implications of radio fingerprinting on the security of sensor networks," in *International Conference on Security and Privacy in Communications Networks and the Workshops - SecureComm 2007*, Sep. 2007, pp. 331–340.

[49] Q. Xu, R. Zheng, W. Saad, and Z. Han, "Device Fingerprinting in Wireless Networks: Challenges and Opportunities," *IEEE Communications Surveys Tutorials*, vol. 18, no. 1, pp. 94–104, 1st Quart. 2016.

[50] S. U. Rehman, K. W. Sowerby, S. Alam, and I. Ardekani, "Radio frequency fingerprinting and its challenges," in *IEEE Conference on Communications and Network Security*, Oct. 2014, pp. 496–497.

[51] G. Baldini and G. Steri, "A Survey of Techniques for the Identification of Mobile Phones Using the Physical Fingerprints of the Built-In Components," *IEEE Communications Surveys Tutorials*, vol. 19, no. 3, pp. 1761–1789, thirdquarter 2017.

[52] S. Mathur, N. M, C. Ye, and A. Reznik, "Radio-telepathy: extracting a secret key from an unauthenticated wireless channel," in *MobiCom*, 2008, pp. 128–139.

[53] J. Croft, N. Patwari, and S. K. Kasera, "Robust uncorrelated bit extraction methodologies for wireless sensors," in *Proc. of ACM/IEEE International Conference on Information Processing in Sensor Networks*. ACM, 2010, pp. 70–81.

[54] T. X. Zheng, H. M. Wang, J. Y. an, Z. Han, and M. H. Lee, "Physical Layer Security in Wireless Ad Hoc Networks Under A Hybrid Full-/Half-Duplex Receiver Deployment Strategy," *IEEE Transactions on Wireless Communications*, vol. 16, no. 6, pp. 3827–3839, Jun. 2017.

[55] J. Zhang, R. Woods, T. Q. Duong, A. Marshall, Y. Ding, Y. Huang, and Q. Xu, "Experimental Study on Key Generation for Physical Layer Security in Wireless Communications," *IEEE Access*, vol. 4, pp. 4464–4477, Aug. 2016.

[56] H. Taha and E. Alsusa, "Secret Key Exchange Using Private Random Precoding in MIMO FDD and TDD Systems," *IEEE Transactions on Vehicular Technology*, vol. 66, no. 6, pp. 4823–4833, Jun. 2017.

[57] Z. Li and H. Wang, "A key agreement method for wireless body area networks," in *IEEE Conference on Computer Communications Workshops (INFOCOM WKSHPS)*, Apr. 2016, pp. 690–695.

[58] S. Jain and J. Guajardo, "Physical Layer Group Key Agreement for Automotive Controller Area Networks," in *Proc. of International Conference on Cryptographic Hardware and Embedded Systems (CHES)*, 2016, pp. 85–105.

[59] G. Revadigar, C. Javali, W. Hu, and S. Jha, "DLINK: Dual link based radio frequency fingerprinting for wearable devices," in *IEEE Conference on Local Computer Networks (LCN)*, Oct. 2015, pp. 329–337.

[60] G. Revadigar, C. Javali, H. J. Asghar, K. B. Rasmussen, and S. Jha, "Mobility Independent Secret Key Generation for Wearable Health-care Devices," in *Proc. of EAI International Conference on Body Area Networks*, 2015, pp. 294–300.

[61] F. Renna, N. Laurenti, S. Tomasin, M. Baldi, N. Maturo, M. Bianchi, F. Chiaraluce, and M. Bloch, "Low-power Secret-key Agreement over OFDM," in *Proc. of ACM Workshop on Hot Topics on Wireless Network Security and Privacy*, 2013, pp. 43–48.

[62] T. Shimizu, H. Iwai, and H. Sasaoka, "Physical-Layer Secret Key Agreement in Two-Way Wireless Relaying Systems," *IEEE Transactions on Information Forensics and Security (TIFS)*, vol. 6, no. 3, pp. 650–660, Sept. 2011.

[63] G. R. Tsouri and J. Wilczewski, "Reliable Symmetric Key Generation for Body Area Networks Using Wireless Physical Layer Security in the Presence of an On-body Eavesdropper," in *Proc. of the 4th International Symposium on Applied Sciences in Biomedical and Communication Technologies*, 2011, pp. 153:1–153:6.

[64] G. Di Crescenzo, M. Striki, and J. S. Baras, "Modeling Key Agreement in Multi-hop Ad Hoc Networks," in *Proc. of the International Confer-*

ence on Wireless Communications and Mobile Computing (IWCMC), 2006, pp. 39–44.

[65] "IEEE Standard for Low-Rate Wireless Personal Area Networks (WPANs)," *IEEE Std 802.15.4-2015 (Revision of IEEE Std 802.15.4-2011)*, pp. 1–709, Apr. 2016.

[66] M. R. Palattella, N. Accettura, X. Vilajosana, T. Watteyne, L. A. Grieco, G. Boggia, and M. Dohler, "Standardized Protocol Stack for the Internet of (Important) Things," *IEEE Communications Surveys Tutorials*, vol. 15, no. 3, pp. 1389–1406, 3rd Quarter 2013.

[67] X. Villajosana, K. Pister, and T. Watteyne, "Minimal IPv6 over the TSCH Mode of IEEE 802.15.4e (6TiSCH) Configuration," IETF, RFC 8180, May 2017.

[68] D. Stanislowski, X. Vilajosana, Q. Wang, T. Watteyne, and K. S. J. Pister, "Adaptive Synchronization in IEEE802.15.4e Networks," *IEEE Transactions on Industrial Informatics*, vol. 10, no. 1, pp. 795–802, Feb. 2014.

[69] S. Sciancalepore, G. Piro, E. Vogli, G. Boggia, L. Grieco, and G. Cavone, "LICITUS: a lightweight and standard compatible framework for securing layer-2 communications in the IoT," *Computer Networks (COMNET), Elsevier*, vol. 108, pp. 66–77, Oct. 2016.

[70] S. Sciancalepore, G. Piro, G. Boggia, and G. Bianchi, "Public Key Authentication and Key Agreement in IoT Devices With Minimal Airtime Consumption," *IEEE Embedded Systems Letters*, vol. 9, no. 1, pp. 1–4, March 2017.

[71] S. Sciancalepore, M. Vucinic, G. Piro, G. Boggia, and T. Watteyne, "Link-layer security in TSCH networks: effect on slot duration," *Transactions on Emerging Telecommunications Technologies*, vol. 28, no. 1, 2017.

[72] NIST, "Recommendation for Key Management, Part 1: General," *NIST Special Publication (SP) 800-57, Part 1 Rev. 4*, Jan. 2016.

[73] M. Mitzenmacher and E. Upfal, *Probability and Computing: Randomized Algorithms and Probabilistic Analysis*. Cambridge University Press, 2005.

[74] K. Grover, A. Lim, and Q. Yang, "Jamming and Anti-jamming Techniques in Wireless Networks: A Survey," *Int. J. Ad Hoc Ubiquitous Comput.*, vol. 17, no. 4, pp. 197–215, Dec. 2014.

[75] A. Mpitziopoulos, D. Gavalas, C. Konstantopoulos, and G. Pantziou, "A Survey on Jamming Attacks and Countermeasures in WSNs," *Commun. Surveys Tuts.*, vol. 11, no. 4, Oct. 2009.

[76] M. Strasser, C. Ppper, S. Capkun, and M. Cagalj, "Jamming-resistant key establishment using uncoordinated frequency hopping," in *2008 IEEE Symposium on Security and Privacy (sp 2008)*, May 2008, pp. 64–78.

[77] R. Di Pietro and G. Oligeri, "Freedom of Speech: Thwarting Jammers via a Probabilistic Approach," in *Proceedings of the ACM Conference on Security & Privacy in Wireless and Mobile Networks*, ser. WiSec '15. New York, NY, USA: ACM, 2015, pp. 4:1–4:6. [Online]. Available: http://doi.acm.org/10.1145/2766498.2766515

[78] Texas Instruments, "CC2538 datasheet," Apr. 2015.

[79] "Primary batteries," in *Batteries for Portable Devices*, G. Pistoia, Ed. Elsevier Science B.V., 2005, pp. 33 – 76.