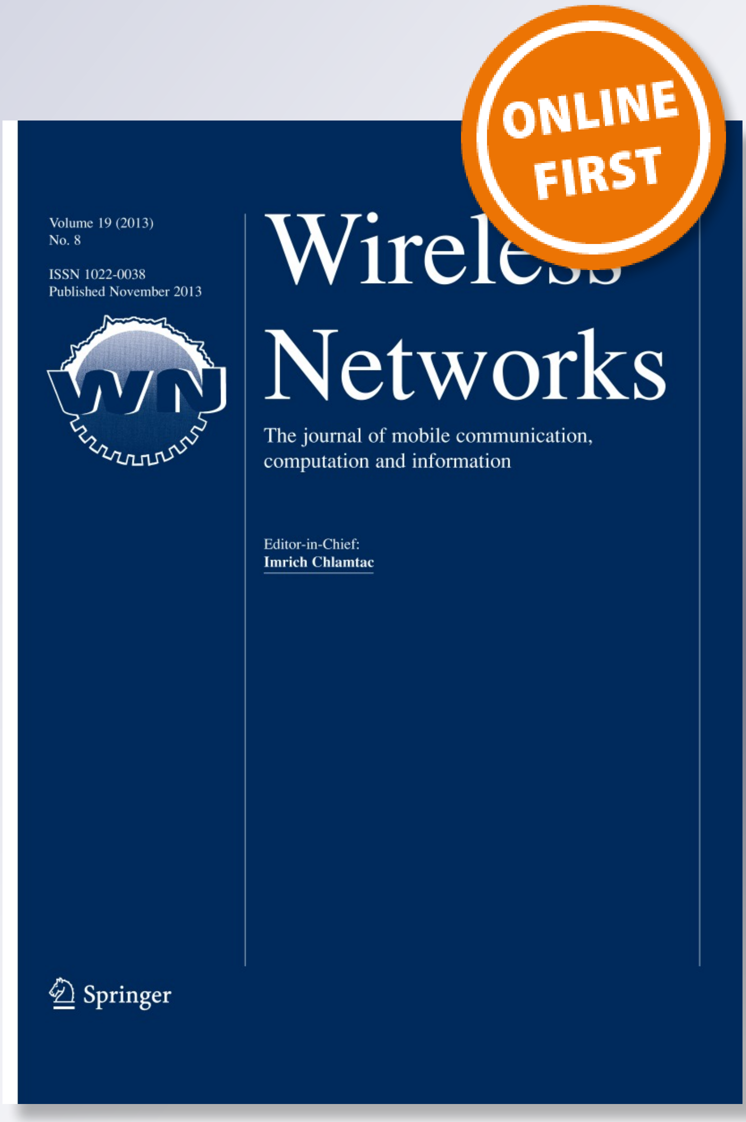
A solution to the GTS-based selective jamming attack on IEEE 802.15.4 networks

Roberta Daidone, Gianluca Dini & Marco Tiloca

Wireless Networks
The Journal of Mobile Communication, Computation and Information

ISSN 1022-0038

Wireless Netw
DOI 10.1007/s11276-013-0673-y



Springer

Your article is protected by copyright and all rights are held exclusively by Springer Science +Business Media New York. This e-offprint is for personal use only and shall not be self-archived in electronic repositories. If you wish to self-archive your article, please use the accepted manuscript version for posting on your own website. You may further deposit the accepted manuscript version in any repository, provided it is only made publicly available 12 months after official publication or later and provided acknowledgement is given to the original source of publication and a link is inserted to the published article on Springer's website. The link must be accompanied by the following text: "The final publication is available at link.springer.com”.

Springer

Author's personal copy

Wireless Netw
DOI 10.1007/s11276-013-0673-y

A solution to the GTS-based selective jamming attack on IEEE 802.15.4 networks

Roberta Daidone · Gianluca Dini · Marco Tiloca

© Springer Science+Business Media New York 2013

Abstract The IEEE 802.15.4 standard allows devices to access the medium not only in contention mode but also in a contention-free way, in order to support quality of service (QoS). In contention-free mode, devices access the medium according to the guaranteed time slot (GTS) mechanism, which is vulnerable to the selective jamming attack. This is a particularly insidious form of denial of service that allows an attacker to thwart QoS while limiting her own exposure at the minimum. In this paper, we present selective jamming resistant GTS , a solution against the GTS-based selective jamming. We also show that our solution is standard compliant and affordable for resource-scarce devices like Tmote Sky motes.

Keywords IEEE 802.15.4 · GTS · Security · Denial of service · Jamming · Wireless sensor networks

1 Introduction

IEEE 802.15.4 is a widely adopted communication standard for personal area networks (PANs) composed of low-cost, low-power, resource-scarce devices [15]. Wireless sensor networks (WSNs) are a relevant example of this kind of networks. Given the importance of communication efficiency in PANs, IEEE 802.15.4 gives support to quality of service (QoS) through the guaranteed time slot (GTS) mechanism, which allows devices to access the medium without contention [6].

R. Daidone · G. Dini · M. Tiloca (✉)
Dipartimento di Ingegneria della Informazione, Elettronica, Informatica, Telecomunicazioni, University of Pisa, Via Diotisalvi 2, 56122 Pisa, Italy
e-mail: marco.tiloca@iet.unipi.it

In a PAN, the medium access temporization is divided into consecutive *superframes*. In its turn, each superframe is divided into a contention access period (CAP) and a contention free period (CFP). In the CFP, nodes access the medium during pre-assigned slots, namely GTS slots (hereafter *slots*). At the beginning of each superframe, network devices require slots to the *PAN Coordinator*, a particular network node, which allocates available slots to requiring nodes, and returns them the *GTS List* specifying its allocation decision.

Unfortunately, the PAN Coordinator transmits the GTS List in the clear. Therefore, an adversary simply equipped with a radio receiver/transmitter can easily eavesdrop the allocation decision, select a slot, and jam it. We call this kind of denial of service (DoS) attack the GTS-based selective jamming attack.

With respect to a classical wide-band jamming where the adversary jams the whole channel, a GTS-based selective jamming attack defines a different trade-off between attack severity and attack detectability. A wide-band jamming attack has the highest severity but it is the simplest to detect. In contrast, a selective jamming is much more difficult to detect because the adversary limits its exposure to a slot. However, it may cause severe QoS degradation to specific traffic segments.

Sokullu et al. [26] have first identified this type of attack and illustrated two possible incarnations, namely the *random attack* and the *intelligent attack* . In the random attack, the adversary selects the Slot to jam at random. In contrast, in the intelligent attack, the adversary exploits the knowledge of the allocation decision to select the longest Slot. Furthermore, Sokullu *et al.* have evaluated that an intelligent attacker can achieve a corruption strength of 50.48 % [25], which means that only half of the available bandwidth would be actually available for communication during the

Published online: 29 November 2013

Author's personal copy

CFP. It turns out that an intelligent attack makes it possible to compromise the QoS of the whole network. Notwithstanding the efficiency and the severity of this type of attack, no countermeasure has been devised so far.

In this paper we fill this gap by a twofold contribution. First of all, we complete the range of possible GTS-based selective jamming incarnations with the *sniper attack*. In this attack, the adversary selects a victim node and then, exploiting the knowledge of the allocation decision, jams the Slot allocated to that node. It follows that this attack may compromise the QoS of a specific node, or even thwart its communication capability altogether, with the minimum chances of being detected.

Secondly, we present a standard-compliant countermeasure against the GTS-based selective jamming attack, i.e. selective jamming resistant GTS (SJRG). To the best of our knowledge, SJRG is the first countermeasure against this attack. When SJRG is active, an attacker can do no better than a random attack (of course this attack, as well as the wide-band jamming, is inevitable). SJRG is based on two basic mechanisms: (1) protection of secrecy and integrity of beacon and GTS request frames by means of encryption; and (2) protection from network traffic analysis by means of intra-slot randomization. While several solutions can be devised, the challenge is to devise one that is compliant with the IEEE 802.15.4 standard. We took up the challenge and conceived the aforementioned mechanisms in such a way that SJRG is fully compliant to IEEE 802.15.4. An implementation of SJRG has been integrated in the open source implementation of IEEE 802.15.4 [20].

The remainder of this paper is organized as follows. In Sect. 2, we discuss about jamming, a particular DoS attack in wireless networks, and refer some significant related works. Section 3 provides a brief overview of the IEEE 802.15.4 standard, with particular attention to the GTS mechanism. Section 4 describes the GTS-based DoS attack, and how it can be performed by means of selective jamming. In Sect. 5, we present SJRG, and discuss how it copes with selective jamming against GTS. Section 6 describes our implementation of SJRG for the TinyOS platform on Tmote Sky motes, while in Section 7 we show and discuss our implementation, and evaluate its performance from several points of view, including network and energy overhead. Finally, in Sect. 8 we draw our conclusive remarks.

2 Wireless denial of service: the jamming attack

Denial of service attacks are a threat which is vital to take into account while securing wireless communications. DoS attacks can be referred to any event that diminishes or eliminates a network's capability to perform its expected functions [9]. In other words, DoS attacks target availability by preventing communication between network devices or by preventing a single device from sending traffic [13].

Jamming consists of corrupting messages transmitted by legitimate users, by interfering in the network's operational frequencies. Jamming is one of the most common DoS attacks, and is definitely considered a severe issue in wireless communications [14, 17, 28, 29, 30]. In the rest of this section, we consider (1) techniques typically adopted to detect and defeat jamming; (2) the effectiveness of jamming against wireless communication; and (3) how different kinds of jamming attacks have been classified.

Typical techniques to detect jamming attacks consist of analyzing: (1) the received signal strength indicator; (2) the average time required to sense an idle channel; and (3) the packet delivery ratio [30].

On the other hand, the most adopted defense against the jamming attack relies on spread-spectrum communication among network devices [13, 24, 27]. This countermeasure requires the attacker either to follow the adopted hopping sequence, or to interfere with a wide section of the band. Another solution relies on legitimate network nodes, which collaboratively identify the jammed region in order to route traffic around it [9]. This requires to adopt a proper routing protocol, such as the TinyOS Destination-Sequenced Distance-Vector Routing [8], which determines high-quality links according to associated link quality estimators.

It has been proven that link layer jamming particularly affects wireless networks performance. In [7], the authors discuss a selective jamming attack, according to which the adversary disturbs the transmission of specific and particularly important kinds of packets. Also, they show the effectiveness of selective jamming on the TCP protocol and its performance. Finally, they propose some methods based on cryptographic primitives, aimed at mitigating its effects.

In [31], the authors define an *intelligent jammer* from the energy consumption point of view. Since in a WSN it is reasonable to assume that the attacker is a sensor node, energy is an issue even for the attacker. Since the intelligent jammer knows MAC protocol specifications, she can preserve energy by attacking at a specific time. On the contrary, a blind jammer wastes energy emitting a continuous signal without any knowledge of the medium access criteria.

An adversary can perform different kinds of jamming, and different classifications of such an attack have been proposed so far. In Xu et al. [30], focus on WSNs, and classify jamming attacks as *constant*, *deceptive*, *random*, and *reactive*. A constant jammer aims at corrupting all network packets by continuously transmitting random signals. Note that such an "always-on" jamming strategy is

Author's personal copy

easier to detect, since it is based on the continuous presence of a high interference level [14, 29, 30]. The deceptive attack consists of injecting a constant stream of bytes into the network, making it look as legitimate traffic. Instead, a random jammer performs the attack by alternating a sleep phase with a jamming phase, thus reducing energy consumption. Finally, a reactive jammer, such as the sniper attacker, performs jamming only when she detects a transmission by other nodes. Of course, reactive jamming results to be much more difficult to be detected, since it is likely to be confused with regular collisions.

O'Flynn considers IEEE 802.15.4 networks, and refers to a different classification of jamming attacks [12], i.e. (1) a wide-band jamming of all available channels; (2) a specific jamming performed upon detecting transmissions of IEEE 802.15.4 messages; and (3) a much more precise jamming against specific messages or network nodes. The sniper attacker performs a jamming of the third kind, and she is particularly dangerous since she perfectly knows when her target transmits. So, she does not need to read the first several bytes of IEEE 802.15.4 MAC headers.

Wood et al. [10] take into account IEEE 802.15.4 networks, and present DEEJAM, a protocol which provides a number of defences against energy-efficient jamming attacks in IEEE 802.15.4 networks. The authors classify jamming attacks into four categories, namely *interrupt jamming*, *activity jamming*, *scan jamming*, and *pulse jamming*. DEEJAM aims at hiding messages from a jammer node, evading its search, and reducing the impact of messages that are corrupted anyway. In particular, it makes use of channel hopping, or uses a pseudo-random sequence as Start of Frame Delimiter at the physical-layer. As a result, the DEEJAM protocol maintains a packet delivery ratio of up to 88 %. However, several WSNs applications are supposed to rely on approved standard, as IEEE 802.15.4. Therefore, DEEJAM can not be considered an official protocol and is not likely to be widely adopted.

3 IEEE 802.15.4

In this Section, we briefly summarize the main features of IEEE 802.15.4, with particular reference to the GTS mechanism and available security services. Table 1 provides the reader with a list of acronyms we use throughout the paper.

As described by the IEEE 802.15.4 standard [15], PANs can be composed of two types of devices, namely full-function devices (FFDs) and reduced-function devices (RFDs). FFDs can communicate with both FFDs and RFDs, while RFDs can communicate only with other FFDs. Also, one specific FFD is elected as the *PAN Coordinator*, and is responsible for network management. Two possible topologies are admitted: *Star* and *Peer-to-peer*. In the Star topology each RFD communicates directly with the PAN Coordinator, whereas in the Peer-to-peer topology each device can communicate with any other FFD in its range. In the rest of this paper, we consider the Star topology, and simply refer to an RFD as a *node*.

As specified by the standard, PANs can work in two possible ways, namely *nonbeacon-enabled* or *beacon-enabled* mode. In particular, in nonbeacon-enabled mode, frames are transmitted according to an *unslotted* carrier sense multiple access with collision avoidance (CSMA-CA) algorithm. In case the medium is sensed idle, the transmission starts immediately. Otherwise, a device

Table 1 List of acronyms

Acronym	Term
AES	Advanced encryption standard
ASH	Auxiliary security header
CAP	Contention access period
CBC_MAC	Cipher block chaining Message authentication code
CCM	Counter with CBC_MAC (mode of operation)
CFP	Contention-free period
CRC	Cyclic redundancy check
CSMA-CA	Carrier sense multiple access with collision avoidance
CTR	Counter mode
FCFS	First come first served
FCS	Frame check sequence
FFD	Full-function device
GTS	Guaranteed time slot
MAC	Medium access control
MFR	MAC footer
MHR	MAC header
MIC	Message integrity code
PAN	Personal area network
RFD	Reduced-function device

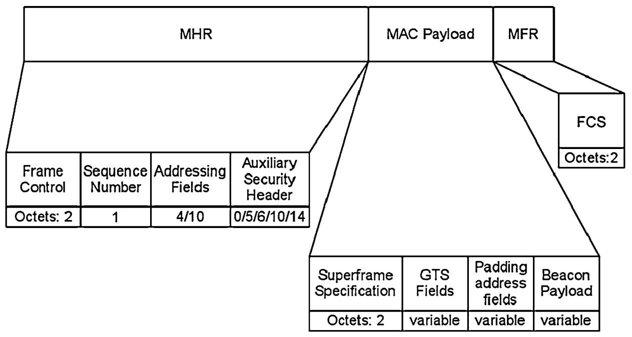


Fig. 1 Beacon frame format

Author's personal copy

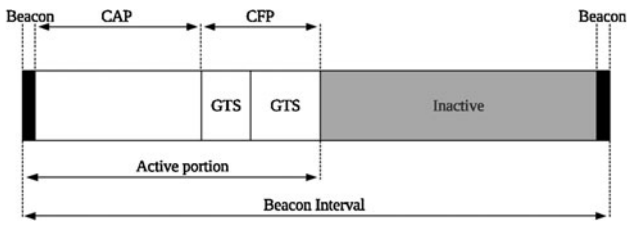


Fig. 2 IEEE 802.15.4 superframe structure

delays the transmission for an exponential random backoff time. In beacon-enabled networks, the *PAN Coordinator* periodically broadcasts *beacon frames* in order to synchronize devices. Each device transmits frames according to a *slotted* CSMA-CA algorithm. The structure of a *beacon* frame is shown in Fig. 1.

In beacon-enabled mode, the PAN Coordinator bounds the medium access temporization as a sequence of *superframes*. As shown in Fig. 2, each superframe is bounded by two consecutive beacon frames, periodically transmitted by the PAN Coordinator.

A superframe can have an *active portion* and an *inactive portion*. During the inactive portion, the PAN Coordinator may switch to low-power mode to save energy. The active portion consists of 16 equally sized *superframe slots*. The active portion includes a CAP and an optional CFP.

During the CAP, nodes access the medium on a contention basis, according to a slotted CSMA-CA algorithm. On the other hand, devices may ask the PAN Coordinator for dedicated portions of the CFP, *Slots* in our parlance, in order to access the medium without contention. This mechanism is known as GTS (see Sect. 3.1), and is particularly useful for applications with QoS constraints and requirements, such as low latency or particular bandwidth requirements.

3.1 Guaranteed time slot (GTS)

Contention free period allows nodes to access the medium during pre-assigned superframe slots, according to the GTS mechanism of the IEEE 802.15.4 standard [15]. Slots are allocated within the CFP by the PAN Coordinator, and are composed by one or more superframe slots.

Figure 3 shows the sequence of operations which take place during a GTS allocation/deallocation process. A node can ask the PAN Coordinator for one Slot, specifying the amount of superframe slots needed and the traffic direction, i.e. transmission or reception. If the request is accepted, one Slot is reserved to that specific user for its own transmissions/receptions. When the assigned Slot is no more needed, a node requests the PAN Coordinator to deallocate it. Both allocations and deallocations are requested by MAC command frames, transmitted anytime nodes successfully access the medium during the CAP.

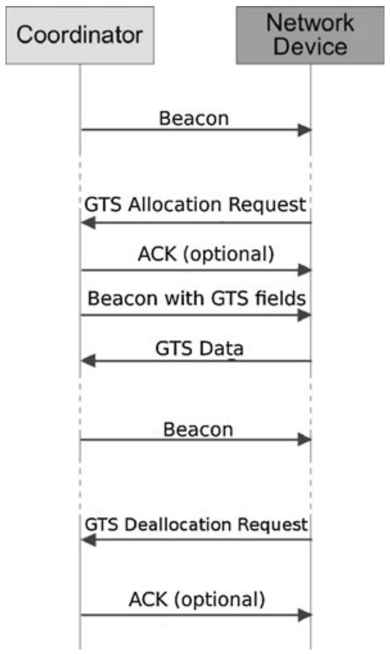


Fig. 3 GTS allocation and deallocation requests

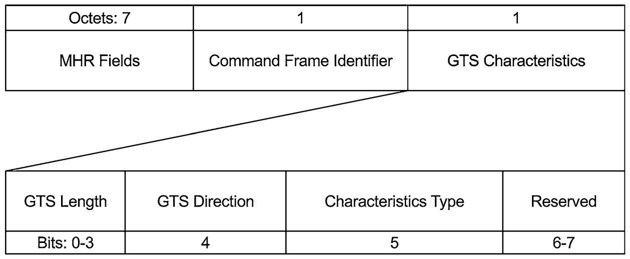


Fig. 4 GTS request command format

On the other hand, the PAN Coordinator manages a pool of seven Slots in a first come first served (FCFS) fashion. Until there are still superframe slots available in the CFP, the PAN Coordinator provides requesting nodes with one slot each. Each slot size is the amount of superframe slots specified in the associated GTS request. The information carried within the GTS fields of each beacon frame specifies (1) whether the node's request has been accepted or not; and (2) when the node is supposed to exclusively access the medium during the CFP. By doing so, a number of users can access the medium without colliding with each other.

Both allocation and deallocation requests are issued by means of a MAC Command frame, namely *GTS Request Command*, whose structure is shown in Fig. 4. The *GTS Characteristics* field is the most significant field in GTS allocation requests, and is composed of the following subfields: (1) the *GTS Length*, which specifies the number of superframe slots requested for the Slot; (2) the *GTS Direction*, which specifies the direction of the data frame

Author's personal copy

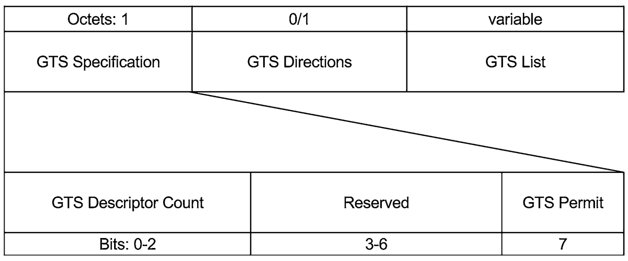
transmission; and, finally, (3) the *Characteristics Type*, which distinguishes between GTS allocation and GTS deallocation.

Upon receiving a GTS Request Command, the PAN Coordinator may send back an optional ACK. Then, the PAN Coordinator decides whether to allocate a slot, considering the GTS requests specifications and the current available capacity in the superframe. GTS requests are considered in a FCFS fashion, and assigned slot s are placed contiguously starting from the end of the CAP. Finally, GTS related information is carried in the *GTS fields* of beacon frames (see Fig. 1).

Figure 5 shows the format of such GTS fields, which consist of: (1) the *GTS Specification*, which specifies if GTS is allowed or not and defines the size of the GTS List field; (2) the *GTS Directions*, which identify the directions of Slots in the superframe; and, finally, (3) the *GTS List*, which includes *GTS descriptors* representing the satisfied GTS requests.

Figure 6 shows the format of a GTS descriptor. Each one of them is 3 bytes long, and is composed of the following fields: (1) the *Device Short Address*, which contains the short address of the device for which the GTS descriptor is intended; (2) the *GTS Starting Slot*, which contains the superframe slot at which the Slot begins; and, finally, (3) the *GTS Length*, which contains the number of contiguous superframe slots over which GTS is active.

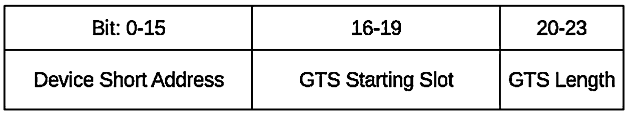
Every GTS requesting node continues to track beacon frames for at most *aGTSDescPersistenceTime* superframes, in order to check if its request has been accepted [15]. If this is the case, the requesting node extracts the GTS Starting slot from the right GTS descriptor, thus gaining knowledge of its own transmission/reception time. Thanks to the GTS Starting slot, each satisfied node knows when it can transmit or receive frames without any contention to access the medium. Otherwise, if a GTS allocation request can not be satisfied, the unsatisfied node notifies a failure to the next upper layer.



Octets: 1	0/1	variable
GTS Specification	GTS Directions	GTS List

GTS Descriptor Count	Reserved	GTS Permit
Bits: 0-2	3-6	7

Fig. 5 Format of the GTS fields



Bit: 0-15	16-19	20-23
Device Short Address	GTS Starting Slot	GTS Length

Fig. 6 Format of the GTS descriptor

If a Slot is no longer required, it can be deallocated at any time, at the discretion of the PAN Coordinator or the devices that originally issued the request.

3.2 Security services

IEEE 802.15.4 provides also a number of security services, and makes them available to the higher layers. The standard provides data confidentiality, data authenticity, and replay protection on a per-frame basis. If communications are secured, senders build an auxiliary security header (ASH), insert it next to the standard MAC header, and secure frames before transmitting them. According to the information carried within the ASH, recipients retrieve the right cryptographic key and correctly unsecure MAC frames.

The standard includes a security suite based on the advanced encryption standard (AES) 128 bits symmetric-key cryptography [19]. Besides, three different security modes are available, i.e. encryption only (*CTR*); authentication only (*CBC_MAC*); and, finally, both encryption and authentication (*CCM*). Both CBC_MAC and CCM rely on a message integrity code (MIC), whose size can be either 4, 8, or 16 bytes. By means of these security tools, *security data structures* and *security procedures* provided by the standard, it is possible to secure/unsecure MAC frames and contrast the GTS-based selective jamming attack.

4 GTS-based selective jamming attack

As described in Sect. 3.1, the PAN Coordinator manages GTS allocation requests, and notifies the accepted ones by broadcasting unencrypted beacon frames. However, as defined by Sokullu et al. [25, 26], the intelligent attacker exploits the knowledge of the allocation decision in order to find out the longest slot, and selectively jam it.

Instead, the sniper attacker we defined exploits the transmission of unsecured beacon frames in order to know which users have been granted a collision-free slot. Then, the attacker creates collisions only during the slot of a specific user, resulting in a DoS attack against it.

Figure 7 shows an example of GTS-based selective jamming attack. By eavesdropping the medium, an adversary is able to extract the GTS List from the GTS fields of the beacon frame (see Fig. 5). Thus, she can gain knowledge of how Slots have been scheduled within the superframe. In other words, the adversary becomes aware of which specific users are going to access the medium during the CFP of the current superframe, and during which specific Slot each one of them will access the medium.

Author's personal copy

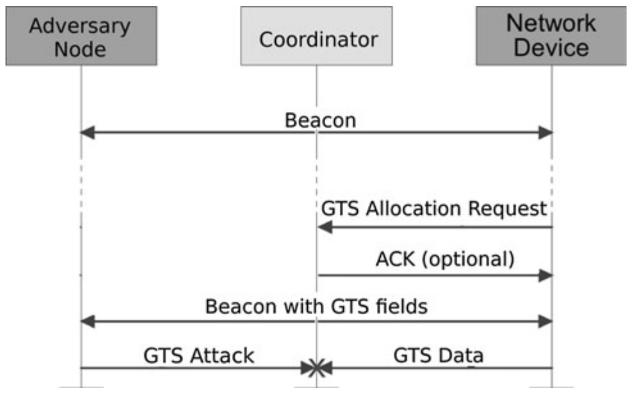


Fig. 7 Example of GTS-based selective jamming attack

This means it is very easy for the adversary to *selectively* interfere with transmissions, causing collisions and corruptions of data frames between the legitimate GTS clients and the PAN Coordinator. The sniper attacker performs the GTS-based selective jamming attack as follows.

1. She selects her victim, that is, she picks a specific node among network devices.
2. She collects beacon frames and parses their MAC headers. By doing so, she figures whether her victim has been assigned a Slot in the CFP of the current superframe. If this is the case, she jams the Slot assigned to her victim, selectively interfering with her transmission/reception.

5 Selective jamming resistant GTS (SJRG)

As explained in Sect. 4, an adversary can intercept broadcast beacon frames, retrieve the GTS descriptors from them, and perform the attack discussed above. The actual vulnerability in the GTS mechanism consists of the adversary having free access to the information carried within the GTS descriptors. A solution to this consists of *encrypting* and *authenticating* GTS-related information within beacon frames.

Encryption makes it possible to prevent an adversary from knowing slots allocation. Also, authentication assures that beacon frames have been actually built by the PAN Coordinator. Of course, beacons can be secured by means of the security services provided by the IEEE 802.15.4 standard. Unfortunately, the standard allows for encrypting just the beacon payload portion of the beacon MAC payload (see Fig. 1). In contrast, the information we need to protect is carried within the GTS fields.

In this section, we describe SJRG, our solution to the GTS-based selective jamming attack. It is compliant with the IEEE 802.15.4 standard, and effective against both the intelligent attacker [25, 26] and the sniper attacker.

Our countermeasure consists of the following steps: (1) MAC frames smart encryption and authentication; and (2) random Slots allocation. The main objective of our solution is to prevent an attacker from gaining access to the information carried within the GTS Fields of beacon frames (see Sect. 3.1). SJRG manages such fields as follows.

The *GTS Specification* field contains the GTS Descriptor Count subfield, and specifies the number of GTS descriptors contained in the *GTS List*. The GTS List is a list of *GTS descriptors*, and is moved to the beacon payload portion of the beacon frame, so making it possible to encrypt and authenticate it. Finally, the *GTS Directions* subfield is moved to the beacon payload portion, so making it possible to secure it.

According to our countermeasure, the information required to successfully perform the attack is moved inside the beacon payload. As a consequence, such information can now be encrypted, so that the attacker is not able to correctly retrieve and analyze it. The PAN Coordinator and the nodes which make use of this countermeasure must be able to mutually recognize each other. In order to do that, it is sufficient to rely on a proper SJRG flag in beacon frames and GTS Request commands. As to the beacon frames, we use one bit of the Reserved subfield of the GTS Specification field (see Fig. 5). As to the GTS Request commands, we use one bit of the Reserved subfield of the GTS Characteristics field (see Fig. 4). We believe these two bits are the only part of SJRG which may result in a modification of the IEEE 802.15.4 standard.

It is useful to encrypt and authenticate also MAC command frames. By doing so, the adversary would not be able to analyze network traffic and recognize GTS Request Commands. If MAC command frames are encrypted, the adversary can still recognize them from their MAC header, but cannot either recognize a GTS Request Command nor distinguish between GTS allocation and deallocation requests. Authentication is needed as well, otherwise the adversary would be able to spread fake GTS Request Commands. Encryption and authentication rely on a fresh *nonce* value, i.e. a randomly generated number used to prevent replay attacks.

The standard states that the PAN Coordinator manages the Slots assignment in a static and thus predictable way. That is, if no deallocations occur, assigned Slots are not meant to change their position in the CFP, even for a considerable amount of consecutive superframes. Thus, even if the adversary cannot analyze the encrypted GTS List, she is still able to infer this information by analyzing the network traffic pattern, and observe the sequence of transmissions during the CFP. This analysis can be made

Author's personal copy

pointless by unpredictably changing the position of Slots on a per superframe basis.

By doing so, it is possible to practically preclude an intelligent attacker or a sniper attacker from purposely causing collisions during a specific Slot. Also, since only the order of Slots in the CFP is changed, our solution is not in conflict with the IEEE 802.15.4 FCFS scheduling policy. In fact, GTS requests are still served in the same way by the PAN Coordinator, thus SJRG does not affect the amount of accepted GTS requests.

Thanks to SJRG, the sniper attacker is not able to purposely interfere during a specific Slot. As a consequence, she can attempt a collision only on a randomly picked Slot. SJRG reduces the probability of success of the sniper attacker to x/n, where x and n are the size in superframe slots of the target Slot and the CFP, respectively. The worst case for the sniper attacker is when all Slots have been assigned and have size equal to 1 superframe slot. In such a case, since the maximum allowed GTS allocation is 7 Slots [15], the sniper attacker has a statistical success rate of 1/7.

What follows is the sequence of actions that take place at the beginning of each superframe in the presence of SJRG.

1. Every node interested in requesting a Slot prepares a GTS Request command frame. Then, these nodes specify the number of required superframe slots, and set the SJRG flag in the Reserved subfield of the GTS Characteristics field (see Fig. 4). Finally, they authenticate and encrypt the GTS Request command frame, and send it to the PAN Coordinator.
2. The PAN Coordinator verifies the authenticity of GTS Request commands, and decrypts them. Then, for each one of them, it verifies that the SJRG flag in the Reserved subfield of the GTS Characteristics field is set.
3. The PAN Coordinator serves GTS requests in an FCFS fashion, according to IEEE 802.15.4 standard specifications. Once GTS requests have been served, the Slots allocation is randomly altered.
4. The PAN Coordinator builds the beacon frame for the current superframe as follows.
    (a) The GTS Directions and GTS List fields are filled, according to the output from step 3. These fields are placed into the Beacon Payload field, instead of the GTS fields.
    (b) The GTS Descriptor Count subfield within the GTS Specification field is set to 0.
    (c) The SJRG flag in the Reserved subfield of the GTS Specification field is set.
    (d) The beacon frame is authenticated. Then, the PAN Coordinator encrypts the Beacon Payload field, and broadcasts the beacon frame to network nodes.
5. Upon reception of a beacon frame, each node verifies its authenticity. Then, each node which has issued a GTS request performs the following actions.
    (a) It verifies that the SJRG flag in the Reserved subfield of the GTS Specification field is set.
    (b) It decrypts the Beacon Payload field, and verifies the authenticity of the beacon frame. Then, it retrieves the GTS List from the Beacon Payload, and checks if its GTS request has been accepted, i.e. if it has been granted a Slot for the current superframe.

It is worth clarifying that SJRG is not a countermeasure against the wide-band jamming attack. In fact, it is not effective in case an adversary interferes with all nodes' transmissions during the CFP, by continuously jamming all available channels. Still, we believe that, in a WSN, it is very likely that the attacker relies on sensor nodes and aims at limiting energy consumption, thus avoiding performing wide-band jamming.

5.1 Discussion on dictionary attack

Since SJRG requires the encryption of a small amount of bytes to protect the allocation of Slots, there might still be a chance for an adversary to find a breach in SJRG by performing a *dictionary attack*.

In order to understand what a dictionary attack consists of, consider an attacker who eavesdrops the medium, records all possible encrypted GTS Lists, and builds a dictionary. Such a dictionary allows the adversary to associate the behavior of GTS devices to the corresponding GTS List, even if it is encrypted. Each GTS List includes the Slot during which devices are supposed to transmit. If the adversary succeeds in building the dictionary, she can (1) listen to a beacon frame and retrieve the encrypted GTS List of the current superframe; (2) look for the corresponding entry in the dictionary; and (3) find the exact Slot to jam in order to hit her victim. Of course, this would make SJRG useless against selective jamming.

In the following, we show how such a dictionary attack is not practically feasible. We recall that the cryptographic key used to authenticate and encrypt MAC frames has to be renewed when the Frame Counter field value is 0xffffffff, as specified by the IEEE 802.15.4 standard [15]. Also, we assume that all MAC data frames are sent by network devices to the PAN Coordinator, thus the GTS Directions content remains constant over time. This is reasonable, because in many applications the PAN Coordinator just collects information transmitted by sensor nodes.

According to the IEEE 802.15.4 standard, we can have up to 7 Slots per superframe. Thus, by randomly altering Slots allocation, we can have up to $7! = 5040$ possible

Author's personal copy

GTS List configurations, carried within beacon payloads. Also, the IEEE 802.15.4 standard requires to encrypt each frame considering a 13 bytes nonce. Even if the nonce has a predictable structure, it includes a 32 bits long frame counter, which varies at each frame. Then, a complete dictionary consists of $5040 \cdot 2^{32}$ entries. Each entry includes one possible GTS List configuration with 7 GTS descriptors, whose size is 3 bytes each. Since we have assumed that the GTS Directions content remains constant over time, it is not necessary to include such a field in the dictionary, so each entry is 21 bytes in size.

Thus, building a complete dictionary would be practically unfeasible, since it requires more than 413 TB to be stored, which is not affordable for a wide range of adversaries. Also, if we considered a security mode which includes both encryption and authentication, we would have some extra unpredictability due to the extra bytes of the MIC. As a consequence, the dictionary would get even larger in terms of both number of entries and their size.

Moreover, in order to have a complete dictionary, the adversary would have to build all the possible $5040 \cdot 2^{32}$ entries. This is impossible because the Frame Counter field value reaches 0xffffffff after 2^{32} beacons, which is less than the amount of dictionary entries. Thus, the adversary cannot complete the dictionary before key renewal takes place.

6 SJRG implementation

We implemented SJRG referring to the implementation of IEEE 802.15.4 for the TinyOS platform [3] available at [4]. This open source implementation is written in the *nesC* language [1], and is maintained by the Working Group [2]. We extended the above mentioned release, and implemented the IEEE 802.15.4 security services and procedures [20]. We defined modules that implement security data structures and security procedures described by the IEEE 802.15.4 standard [16], with reference to the Tmote Sky platform [18] and the CC2420 chipset [5].

As to SJRG, we extended the MAC frames parsing process, in order to properly manage the ASH in the presence of secured MAC beacon and command frames [20]. In this section, we consider the CCM_16 security mode, which authenticates frames using a 16 bytes MIC, and then encrypts both the MIC and the payload. Nevertheless, SJRG works properly also with CCM_4 and CCM_8 modes (see Sect. 3.2 for an overview about security modes). In case the PAN Coordinator and network devices rely on SJRG, they mutually recognize each other by means of the SJRG flag carried within beacon frames and GTS Request commands (see Sect. 5).

As already discussed, it is vital also to exchange slots position in an unpredictable way. Otherwise, the adversary would be able to predict the new allocation scheme, and successfully perform the GTS attack. Therefore, the random reorganization of slots within the MAC frame must rely on a secure pseudo-random number generator (SPRNG).

We implemented a SPRNG based on the CC2420 chipset [5]. Specifically, we took inspiration from the ANSI X9.17 pseudo-random bit generator [11], and considered a practical variation in order to build our own one. Specifically, our randomization function can be expressed as follows.

$$x_i = \begin{cases} E_k(s) & \text{if } i = 1 \\ E_k(x_{i-1}) & \text{if } i \neq 1 \end{cases}$$

The input of the encryption function E is a key k and a quantity to be encrypted. The seed s must be a fresh quantity, i.e. a nonce value, and can be initialized during the PAN Coordinator startup. Since the adversary does not know the key k, the output x_i of the above function can be considered a random variable. Note that if we assume that the key k is a secret, then the seed s can be public.

After the PAN Coordinator has performed the regular Slots allocation, the order of slots within the CFP is altered by adding all GTS starting Slots with a pseudo-random quantity generated with the SPRNG. As a result, the adversary cannot obtain information about Slots allocation by either analyzing the network traffic or observing the order of transmissions during the CFP.

7 SJRG evaluation

We evaluate SJRG by considering four different aspects: (1) *effectiveness*, i.e. the gain in terms of delivery ratio achieved with respect to an unprotected setup; (2) *memory footprint*, i.e. the extra amount of memory required by our implementation; (3) *network performance*, i.e. the delay due to additional processing and transmissions; and, finally, (4) the additional *per packet energy consumption*.

We performed our evaluation by taking into account a real application on a realistic scenario. We considered an IEEE 802.15.4 star topology consisting of one PAN Coordinator and 7 RFD nodes acting as GTS senders. Also, we considered the presence of one sniper attacker. All nodes were Tmote Sky motes, which feature a CC2420 chipset and are provided with a 48 kB ROM [18].

In our testing application, the PAN Coordinator broadcasts a beacon frame, so that sender nodes can associate to the PAN. Then, all sender nodes ask for a Slot to the PAN Coordinator, specifying that they support SJRG. Since the PAN Coordinator provides SJRG, it broadcasts a SJRG beacon frame. Once they have received the SJRG beacon frame, sender nodes gain knowledge of their reserved Slot. From then on, each one of them transmits its data frames to the PAN Coordinator during the assigned Slot. The attacker

Author's personal copy

node aims at disrupting data frames transmission of a specific sender, but SJRG forces it to pick a Slot as target in a random way.

7.1 Effectiveness

The effectiveness of our countermeasure has been evaluated considering the probability of success of the attacker to jam communications of her target node, both in the presence and in the absence of SJRG. The presence of 7 senders represents the worst case from the attacker point of view, because the PAN Coordinator always allocates 7 Slots in the CFP. Also, we fixed the size of Slots to one superframe slot each. That is, the smaller the target is, the harder it is for the attacker to hit it.

In order to estimate how many successful transmissions have been made, we observed the amount of acknowledgment frames received by each node that transmits during its own Slot, in the presence of the attacker node. In order to increase the accuracy of our results, we performed 10 repetitions of 100 transmissions for each experiment. The results we present here are averaged over all the different repetitions. We also report the standard deviation we derived from the independent replication method.

Our experimental evaluation considered the following two scenarios:

- *No SJRG* the 88.4 % of transmissions from the target node were corrupted by the sniper attacker, while no transmissions from other sender nodes were corrupted. The 11.6 % of successful transmissions from the target node are due to imperfect clock sychronization of Tmote Sky motes. That is, because of the clock drift effect, a short frame might be transmitted before the adversary starts jamming the target Slot.
- *SJRG* the 13.8 % of transmissions from the target node were corrupted by the sniper attacker, and the 13.7 % from the other sender nodes were corrupted as well. This is due to the fact that the attacker cannot recognize her target anymore, and jams one Slot, by picking it at random. Thus, in the presence of SJRG, all sender nodes have a certain probability of being jammed. Nevertheless, the percentage of failure is affordable, and can be handled by means of retransmissions.

The amount of data frames sent by the target node and correctly received by the PAN Coordinator confirms our theoretical assumption, i.e. the attacker has a statistical success rate of 1/7. So, no DoS occurs.

7.2 Memory footprint

The amount of memory required by our implementation has been evaluated by comparing the TinyOS image size

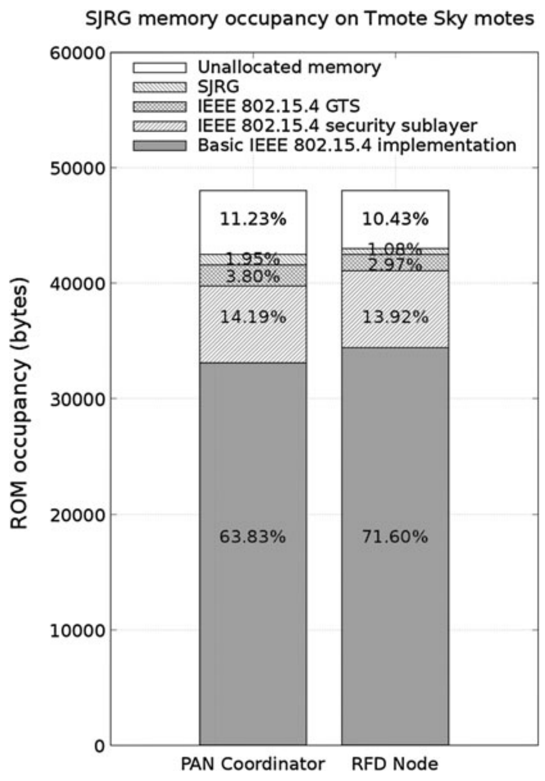


Fig. 8 SJRG memory occupancy on Tmote Sky motes

for Tmote Sky motes, both in the presence and in the absence of SJRG. We evaluated memory consumption considering the most complex and complete standard security configuration (i.e. KeyIdMode3 key retrieval and CCM_16 security mode), as a worst case.

As shown in Fig. 8, the extra amount of memory is mostly due to the implementation of the IEEE 802.15.4 security sublayer. Specifically, we highlight the following five contributions: (1) basic IEEE 802.15.4 implementation; (2) IEEE 802.15.4 security sublayer; (3) IEEE 802.15.4 GTS; (4) SJRG; and (5) unallocated memory.

Our implementation of SJRG occupies the 1.95 % of the whole Tmote Sky memory for the PAN Coordinator, and the 1.08 % for sender devices. We believe that such an extra memory consumption is reasonable and affordable. Finally, even in the presence of SJRG, there is still a not negligible amount of unallocated memory, both on the PAN Coordinator and sender devices. This memory can be used for additional features or more complex applications.

7.3 Network performance

As to network performance, SJRG is operating on top of the 2.4 GHz physical layer, with a 250 Kb/s bit rate [5]. We modeled the impact of SJRG on network performance by considering two main aspects, namely *processing*

Author's personal copy

Table 2 SJRG processing overhead contributions

Contribution	Processing overhead (μs)	Standard deviation (μs)
HW IEEE 802.15.4 security	254.47	1.34
SW IEEE 802.15.4 security	1752.26	5.46
SJRG	125.83	2.12

overhead and *transmission overhead* experienced by the PAN Coordinator.

The processing overhead consists of two contributions: (1) the hardware encryption and authentication contribution, due to the CC2420 chipset; and (2) the software contribution, due to the MSP430 microcontroller. In order to evaluate the software contribution, we considered the IEEE 802.15.4 security processing and the SJRG processing separately.

In Table 2, the first line reports the hardware encryption and authentication contribution. The second line shows the processing overhead due to software security procedures. The third line reports the processing overhead introduced by SJRG. Note that only the third line regards SJRG, while the first and second ones show contributions due to the IEEE 802.15.4 security sublayer. These values are averaged over all the different repetitions. We also include the standard deviation we derived from the independent replication method.

The transmission overhead has been evaluated analytically, considering a bit rate equal to 250 Kb/s [5]. We have considered the time required to transmit the additional bytes added by SJRG, according to the standard security policy. The size of the original GTS beacon packet to manage 7 Slot s is 34 bytes, including the standard frame header and the cyclic redundancy ceck (CRC). Since we consider SJRG relying on the most complete IEEE 802.15.4 security services (i.e. KeyIdMode3 key retrieval and CCM_16 security mode), the beacon frame size is increased of 31 bytes, of which 30 are due to the IEEE 802.15.4 security sublayer.

More in details: (1) 14 bytes are due to the presence of the ASH; (2) 16 bytes are required to provide frame authentication; and (3) 1 byte is due to the fact that we split the GTS fields, duplicate the GTS Direction subfield, and place it into the Beacon Payload field. We keep a fake copy of the GTS Direction subfield in its original position, in order to stay compliant with the standard.

We computed the transmission time d_{tx} of a SJRG beacon frame as the ratio between the frame size in bits and the bit-rate:

$$d_{tx} = \frac{65 \times 8}{0.250} = 2080\,\mu s$$

The original beacon frame is 34 bytes in size, so it can be transmitted in

Table 3 SJRG transmission overhead contributions

Key retrieval policy	ASH size (bytes)	Transmission overhead (μs)
KeyIdMode3	14	448
KeyIdMode2	10	320
KeyIdMode1	6	192
KeyIdMode0	5	160
Security mode	**MIC size (bytes)**	**Transmission overhead (μs)**
CBC_MAC_16/CCM_16	16	512
CBC_MAC_8/CCM_8	8	256
CBC_MAC_4/CCM_4	4	128

$$d_{tx} = \frac{34 \times 8}{0.250} = 1088\,\mu s$$

so the transmission overhead is

$$2080\,\mu s - 1088\,\mu s = 992\,\mu s$$

Note that SJRG adds only one byte to the beacon payload, while the 14 bytes of the ASH and the 16 bytes of the MIC are due to the IEEE 802.15.4 security policies.

However, the user can trade off security and efficiency, thus reducing the transmission overhead due to the presence of the ASH and the MIC. IEEE 802.15.4 allows for choosing among (1) different key retrieval methods, which influences the ASH size, and (2) different sizes of the MIC field, in case authentication or encryption and authentication are required. Table 3 provides an overview of different ASH and MIC sizes, together with the associated transmission overhead. It is evident that by properly matching the ASH and MIC sizes, it is possible to reduce the transmission overhead.

Our results show that the transmission overhead introduced by SJRG is mostly due to the IEEE 802.15.4 security sublayer. Even in case all Slots are in use, SJRG adds only one byte to the beacon frame, which means 32 μs of additional transmission delay. If some Slots are not allocated, SJRG assumes that all Slots are present, as explained in Sect. 5. Of course, this increases the beacon frame size. However, beacon frames are transmitted once per superframe, which means that one beacon is broadcast every 0.983 seconds (considering BeaconOrder = 6 and SuperframeOrder = 6 [15]). Thus, we believe that the increase of beacon frames size is affordable [21–23].

7.4 Energy consumption

As to energy consumption, we considered processing and transmission contributions separately. Each contribution has the form $\mathcal{E}_i = \mathcal{P}_i \times d_i$. We define d_i as the delay contribution due to operation *i*, and refer to delay values

Author's personal copy

Table 4 SJRG energy consumption contributions

Transmission	$\mathcal{P}_{\mathrm{tx}}$ (mW)	d_{tx} (μs)	$\mathcal{E}_{\mathrm{tx}}$ (nJ)
	31.32	992	31069.44
Processing	$\mathcal{P}_{\mathrm{HWsec}}$ (mW)	$\mathrm{d}_{\mathrm{HWsec}}$ (μs)	$\mathcal{E}_{\mathrm{HWsec}}$ (nJ)
	31.32	254.47	7970
	$\mathcal{P}_{\mathrm{SWsec}}$ (mW)	$\mathrm{d}_{\mathrm{SW\ sec}}$ (μs)	$\mathcal{E}_{\mathrm{SWsec}}$ (nJ)
	1.08	1752.26	1892.44
	$\mathcal{P}_{\mathrm{SJRG}}$ (mW)	$\mathrm{d}_{\mathrm{SJRG}}$ (μs)	$\mathcal{E}_{\mathrm{SJRG}}$ (nJ)
	1.08	125.83	135.90

reported in Sect. 7.3. $\mathcal{P}_{\mathrm{i}} = \mathrm{V}_{\mathrm{i}} \times \mathrm{I}_{\mathrm{i}}$ is the single power contribution, expressed as the product between voltage and current of the MSP430 microcontroller and the CC2420 chipset, responsible for processing and transmission, respectively [18]. Table 4 provides an overview of such contributions.

Considerable increases in per packet energy consumption are due to the transmission overhead of the extra bytes required by the IEEE 802.15.4 security sublayer. Also the processing overhead of standard encryption and authentication algorithms has a considerable impact on energy consumption. However, these contributions can be reduced by changing the size of the ASH and the MIC to be transmitted, as discussed in Sect. 7.3.

The actual SJRG contribution to energy consumption is the one reported in the last entry of Table 4, that is the energy consumed to add the SJRG field to the beacon frame after having computed the Slot s order. We believe that this additional energy consumption is affordable, if compared to the contributions introduced by standard security mechanisms, including their transmission overhead.

7.5 On scalability

In this section, we argue that scalability of SJRG with respect to the number of users and the number of attackers is not an issue.

As to the number of users, the IEEE 802.15.4 standard admits up to 7 GTS users during the CFP, thus practically limiting the amount of GTS users to be considered in the performance analysis of SJRG. At the same time, GTS users which have been granted a Slot, and thus potential victims of the attacker, do not contend with each other to access the medium. As a consequence, although the total number of users in the system may impact the GTS mechanism, such a number does not affect SJRG.

As to the number of attackers, we argue that considering two or more attackers is not so interesting, besides being beyond the threat model we consider in this paper. In fact, consider the borderline case in which several attackers, independently of each other, perform a random attack, as SJRG prevents them from performing the intelligent and the sniper attack. In such a case, the greater the number of attackers, the greater the likelihood that they jam all Slots, so causing a total jamming of the collision-free portion of the channel. However, this attack would be easily detectable. As a consequence, several independent attackers would hamper each other.

8 Conclusion

We have presented and discussed SJRG, our standard compliant solution to the GTS-based selective jamming attack, able to cope with both the intelligent and the sniper attack. Such attacks aim at disrupting communications by selectively jamming contention free Slots.

SJRG relies on IEEE 802.15.4 security services, and provides a two steps countermeasure against selective jamming. First, SJRG makes sure that beacon frames and MAC Command frames are secured, i.e. both encrypted and authenticated. Secondly, it unpredictably changes the position of assigned Slots, forcing the attacker to pick the target Slot in a random way and reducing the attack success rate.

SJRG is not a countermeasure against the wide-band jamming attack, i.e. it is not effective in case an adversary interferes with all nodes' transmissions by continuously jamming all available channels. However, we believe that, in a WSN, it is very likely that the attacker uses sensor nodes, and avoids performing wide-band jamming in order to limit energy consumption, i.e. SJRG is still very effective.

We have implemented SJRG for the TinyOS platform on Tmote Sky motes, and evaluated it on a realistic application scenario. In particular, we have shown that (1) the attack success rate can be reduced to 13.8 %; (2) SJRG results in an additional memory consumption which is definitely affordable; (3) the network performance degradation due to SJRG is practically negligible; and (4) the per packet energy consumption is affordable and mostly due to the IEEE 802.15.4 security sublayer contributions.

Acknowledgements This work has been supported by the EU FP7 Network of Excellence CONET (Grant Agreement FP7-224053); the EU FP7 Integrated Project PLANET (Grant agreement FP7-257649); and the TENACE PRIN (No. 20103P34XC) funded by the Italian Ministry of Education, University and Research. We would like to thank Mário Alves and Ricardo Severino from ISEP, for the useful feedback received during the early stages of this work and the standard GTS implementation for TinyOS. A special thank goes to Alessandro Pischedda, for his help in the final stages of this work. Finally, we thank the anonymous reviewers for having helped us to improve our work with their precious comments and advices.

Author's personal copy

References

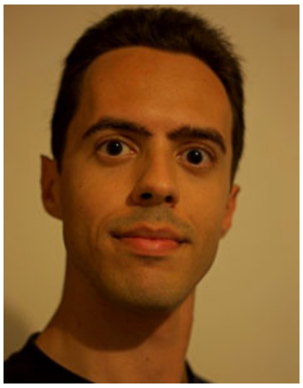
1. nesC. (2004). A programming language for deeply networked systems. URL http://nescc.sourceforge.net/.
2. TinyOS IEEE 802.15.4 Working Group. (2008). *TinyOS documentation Wiki*. http://www.tinyos.net/scoop/special/working_group_tinyos_154.
3. TinyOS Home Page. (2010). URL http://www.tinyos.net/.
4. Main tree of the TinyOS operating system for embedded, wireless devices. (2011). URL http://code.google.com/p/tinyos-main/.
5. Texas Instruments CC2420 2.4 GHz IEEE 802.15.4 / ZigBee-ready RF Transceiver. (2012). URL http://focus.ti.com/lit/ds/symlink/cc2420.pdf.
6. Koubaa, A., Alves, M., & Tovar, E. (2006). GTS allocation analysis in IEEE 802.15.4 for real-time wireless sensor networks. In: 20th International Parallel and Distributed Processing Symposium.
7. Proano, A., & Lazos, L. (2010). Selective jamming attacks in wireless networks. In: 2010 IEEE International Conference on Communications, pp. 1–6.
8. Woo, A., Tong, T., & Culler, D. (2003). Taming the underlying challenges of reliable multihop routing in sensor networks. In: Proceedings of the 1st International Conference on Embedded Networked Sensor Systems, SenSys '03, (pp. 14–27) New York, NY, USA: ACM.
9. Wood, A.D., & Stankovic, J.A. (2002). Denial of service in sensor networks. *Computer, 35*(10), 54–62.
10. Wood, A.D., Stankovic, J.A., & Zhou, G. (2007). DEEJAM: Defeating energy-efficient jamming in IEEE 802.15.4-based wireless networks. In: 4th Annual IEEE Communications Society Conference on Sensor, Mesh and Ad Hoc Communications and Networks, (pp. 60–69).
11. Menezes, A.J., van Oorschot, P.C., & Vanstone, S.A. (2001). *Handbook of applied cryptography*. Boca Raton, FL: CRC Press.
12. O'Flynn, C.P. (2011). Message Denial and Alteration on IEEE 802.15.4 Low-Power Radio Networks. In: 2011 4th IFIP International Conference on New Technologies, Mobility and Security (NTMS), (pp. 1–5)
13. Raymond, D.R., & Midkiff, S.F. (2008). Denial-of-service in wireless sensor networks: Attacks and defenses. *IEEE Pervasive Computing, 7*(1), 74–81.
14. Noubir, G., & Lin, G. (2003). Low-power DoS attacks in data wireless LANs and countermeasures. *SIGMOBILE Mobile Computing Communications Review, 7*, 29–30.
15. Institute of Electrical and Electronics Engineers, Inc. (2006). New York: IEEE Std. 802.15.4-2006, IEEE Standard for Information technology—Telecommunications and information exchange between systems—Local and metropolitan area networks—Specic requirements Part 15.4: Wireless Medium Access Control (MAC) and Physical Layer (PHY) Specifications for Low-Rate Wireless Personal Area Networks (WPANs).
16. Hauer, J.H. , Daidone, R., Severino, R., Büsch, J., Tiloca, M., & Tennina, S. (2011). An Open-Source IEEE 802.15.4 MAC Implementation for TinyOS 2.1. In: Proceedings of 8th European Conference on Wireless Sensor Networks (EWSN 2011), (pp. 1–2).
17. Lazos, L., Liu, S., & Krunz, M. (2009). Mitigating control-channel jamming attacks in multi-channel ad hoc networks. In: Proceedings of the second ACM conference on Wireless network security, WiSec '09. (pp. 169–180) New York, NY: ACM.
18. Moteiv Corporation:Tmote Sky: Datasheet (2006). URL http://www.snm.ethz.ch/snmwiki/pub/uploads/Projects/tmote_sky_datasheet.pdf.
19. National Institute of Standards and Technology. (2001). Federal Information Processing Standards Publication 197, Specification for the Advanced Encryption Standard (AES).
20. open-ZB.net:OpenSource Toolset for IEEE 802.15.4 and ZigBee (2010). URL http://www.open-zb.net/.
21. Daidone, R. (2011). Experimental Evaluations of Security Impact on IEEE 802.15.4 Networks. In: Ph.D. Forum at IEEE International Symposium on a World of Wireless, Mobile and Multimedia Networks (WoWMoM 2011).
22. Daidone, R., Dini, G., & Tiloca, M. (2011). On experimentally evaluating the impact of security on IEEE 802.15.4 networks. In: International Conference and Workshops on Distributed Computing in Sensor Systems (DCOSS 2011), (pp. 1–6).
23. Daidone, R., Dini, G., & Tiloca, M. (2012) On preventing GTS-based Denial of Service in IEEE 802.15.4. In: European Conference on Wireless Sensor Networks, EWSN 2012 (pp. 69–70).
24. Pickholtz, R., Schilling, D., & Milstein, L. (1982). Theory of Spread-Spectrum Communications—A Tutorial. *IEEE Transactions on Communications, 30*(5), 855–884.
25. Sokullu, R., Korkmaz, I., & Dagdeviren, O. (2009). GTS attack: An IEEE 802.15.4 MAC layer attack in wireless sensor networks. *International Journal On Advances in Internet Technologies, 2*(1), 104–114.
26. Sokullu, R., Dagdeviren, O., & Korkmaz, I. (2008). On the IEEE 802.15.4 MAC layer attacks: GTS attack. In: Proceedings of the Second International Conference on Sensor Technologies and Applications, SENSORCOMM '08, (pp. 673–678).
27. Anderson R.J. (2001) Security engineering: A guide to building dependable distributed systems. New York, NY: Wiley
28. Xu, W., Ma, K., Trappe, W., & Zhang, Y. (2006). Jamming sensor networks: Attack and defense strategies. *IEEE Network, 20*(3), 41–47.
29. Xu, W., Wood, T., Trappe, W., & Zhang, Y. (2004). Channel surfing and spatial retreats: Defenses against wireless denial of service. In: Proceedings of the 3rd ACM workshop on Wireless security, WiSe '04. (pp. 80–89) New York, NY: ACM.
30. Xu, W., Trappe, W., Zhang, Y., & Wood, T. (2005). The feasibility of launching and detecting jamming attacks in wireless networks. In: Proceedings of the 6th ACM International Symposium on Mobile ad hoc networking and computing, MobiHoc '05, (pp. 46–57) New York, NY, USA: ACM.
31. Law Y.W., Hartel P., den Hartog J., & Havinga P. (2005). Link-layer jamming attacks on S-MAC. In: Wireless Sensor Networks, 2005. Proceeedings of the Second European Workshop on, (pp. 217–225).

Author Biographies

Roberta Daidone is a Ph.D. candidate, advised by Gianluca Dini, in Computer Engineering, at the "Dipartimento di Ingegneria della Informazione, Elettronica, Informatica, Telecomunicazioni" of the University of Pisa. Her current research interests include secure communications in IEEE 802.15.4 networks, experimental and simulative evaluations of costs of security in wireless sensor networks (WSNs), development of security frameworks for WSNs. She received her Bachelor's Degree and Master's Degree in Computer Engineering from the University of Pisa, Pisa, Italy, in 2007 and 2010, respectively.

Author's personal copy

Gianluca Dini received his "Laurea" degree in Electronics Engineering from the University of Pisa, Pisa, Italy, in 1990, and his Ph.D. in Computer Engineering from Scuola Superiore "S. Anna", Pisa, Italy, in 1995. From 1993 to 1994 he was Research Fellow at the Department of Computer Science of the University of Twente, The Netherlands. From 1993 to 1999 he was Assistant Professor at the Dipartimento di Ingegneria della Informazione, Elettronica, Informatica, Telecomunicazioni of the University of Pisa, where he is now Associate Professor. His research interests are in the field of distributed computing systems, with particular reference to security. Currently he is working on security in networked embedded systems. He has published more than one hundred papers in international conferences, books and journals and has participated to many projects funded by the Commission of the European Community, the Italian Government and private companies.



Marco Tiloca received his Bachelor's Degree and Master's Degree in Computer Engineering from the University of Pisa, Pisa, Italy, in 2006 and 2009, respectively. He received his Ph.D. in Computer Engineering from the "Dipartimento di Ingegneria della Informazione, Elettronica, Informatica, Telecomunicazioni" of the University of Pisa, in 2013. Currently, he is a Postdoctoral Research Fellow at Swedish ICT SICS in Stockholm, Sweden. His research interests are in the field of network security with particular reference to constrained devices, and include key management and secure group communication. Currently, he is working on security in the Internet of Things.