Spring 3-20-2020

Attacking and securing beacon-enabled 802.15.4 networks

Sang Shin Jung
Georgia Institute of Technology

Marco Valero
Georgia State University

Anu G. Bourgeois
Georgia State University

Raheem Beyah
Georgia Institute of Technology

Recommended Citation

Jung, S.S., Valero, M., Bourgeois, A. et al. Attacking and securing beacon-enabled 802.15.4 networks. Wireless Netw 21, 1517–1535 (2015). https://doi.org/10.1007/s11276-014-0855-2

Wireless Netw (2015) 21:1517–1535
DOI 10.1007/s11276-014-0855-2

Attacking and securing beacon-enabled 802.15.4 networks

Sang Shin Jung · Marco Valero · Anu Bourgeois · Raheem Beyah

Published online: 2 December 2014

Abstract The IEEE 802.15.4 standard has attracted time-critical applications in wireless sensor networks because of its beacon-enabled mode and guaranteed timeslots (GTSs). However, the GTS management scheme's security mechanisms still leave the 802.15.4 medium access control vulnerable to attacks. Further, the existing techniques in the literature for securing 802.15.4 networks either focus on nonbeacon-enabled 802.15.4 networks or cannot defend against insider attacks for beacon-enabled 802.15.4 networks. In this paper, we illustrate this by demonstrating attacks on the availability and integrity of the beacon-enabled 802.15.4 network. To confirm the validity of the attacks, we implement the attacks using Tmote Sky motes for wireless sensor nodes, where the malicious node is deployed as an inside attacker. We show that the malicious node can freely exploit information retrieved from the beacon frames to compromise the integrity and availability of the network. To defend against these attacks, we present BCN-Sec, a protocol that ensures the integrity of data and control frames in beacon-enabled 802.15.4 networks. We implement BCN-Sec, and show its efficacy during various attacks.

Keywords Beacon-enabled 802.15.4 · Insider attacks · Media access control · MAC misbehavior · Time-critical applications · Wireless sensor networks

S. S. Jung (✉) · R. Beyah
CAP Research Group, School of Electrical and Computer Engineering, Georgia Institute of Technology, Atlanta, GA 30332, USA
e-mail: sangsin@gatech.edu

R. Beyah
e-mail: raheem.beyah@ece.gatech.edu

M. Valero · A. Bourgeois
Department of Computer Science, Georgia State University, Atlanta, GA 30303, USA
e-mail: mvalero@cs.gsu.edu

A. Bourgeois
e-mail: abourgeois@cs.gsu.edu

1 Introduction

Wireless sensor networks (WSNs) have emerged quickly and attracted a number of diverse applications. The use of these applications ranges from residential to government. For example, Lorincz et al. [25] have introduced an application (CodeBlue) and its infrastructure for a disaster situation or a medical emergency. Within the CodeBlue infrastructure, an emergency code from sensed data is used to alert responsible people for an immediate response. The military is also using WSNs to detect an adversary's behavior and location. For example, seismic sensors can be used to detect the movement of heavy artillery (e.g., tanks) in the battlefield. In either case, not receiving information about the environment in a time-sensitive manner can have significant consequences.

To provide support for time-sensitive communication, the 802.15.4 standard provides a beacon-enabled mode. Unlike nonbeacon-enabled mode, the beacon-enabled mode in an 802.15.4 network employs a few end device nodes and a centralized node [i.e., personal area network coordinator (PC)] that broadcasts beacons to synchronize the nodes in the network, manages guaranteed timeslots (GTS) (de)allocation requests from the nodes, and assigns dedicated slots for transmissions of the nodes through beacons. The beacon broadcast and GTS management scheme are the most critical parts of real-time delivery of

time-sensitive data during the contention free period (CFP) [5, 21, 22, 27, 28, 31].

However, many researchers have focused on improving the performance or energy efficiency of beacon-enabled 802.15.4 MAC and the use of its GTS scheme. For example, the IPP-HURRY research group has analyzed the delay bound of GTS allocations to maximize the throughput of each GTS allocation for real-time sensor networks [21, 22]. In addition, in [5] the authors present a case study of Siemens Industry Automation Division that requires real-time delivery of short alarms/messages. The case study evaluates GTS allocation to maximize low latency of its scheme. In many recent works on WSNs, reliable and real-time delivery are still primary problems to be solved. For example, Anisi et al. [3] present how nodes can deliver data to a base station with minimal packet loss. Shen et al. [35] point out the dynamics of industrial applications and the need for their real-time response. They present the implementation of a source-aware scheduling algorithm to achieve a minimum delay for the response.

Although there has been a significant emphasis on improving the performance of beacon-enabled 802.15.4 networks, there has been little work on securing them. This is significant, given that the GTS management scheme of the PC does not verify the identification (ID) of each node that requests GTSs. Further the nodes in the network do not validate the PC that broadcasts beacons. Therefore, an inside attacker can easily compromise the guaranteed data transmissions from the time-sensitive applications in the beacon-enabled 802.15.4 network by either impersonating a legitimate node (LN) (existing in the PAN or not) or the PC (e.g., implement a Sybil attack [9] at the MAC layer). This MAC misbehavior is a critical security problem not only in the 802.15.4 network itself but also in new emerging wireless technologies [e.g., wireless body area networks (WBANs)] that consider the use of the 802.15.4 network [13].

In this paper, we demonstrate six attacks that are possible by an inside attacker in a beacon-enabled 802.15.4 network. The inside attacker targets the vulnerabilities of the beacon broadcast and the GTS management scheme. The contributions of this paper include the discovery of vulnerable properties of the beacon-enabled mode in the 802.15.4 standard and the implementation and analysis of six potential insider attacks against the vulnerabilities. We also design and implement a lightweight security protocol for beacon-enabled 802.15.4 networks (BCN-Sec) and illustrate its costs and how it defends against these attacks.

The rest of this paper is organized as follows. We review related works, including security protocols for WSNs and attacks on beacon-enabled 802.15.4 networks, in Sect. 2. In Sect. 3, we briefly illustrate the beacon broadcast and the GTS management scheme and explain their vulnerabilities. In Sect. 4, we present the experiment design and show the hardware and software components. In Sect. 5, we first define an attack model and present an overview of the six attacks against the vulnerabilities. In Sect. 6, we describe the implementation of the attacks. We present BCN-Sec, which is used to defend against these attacks in Sect. 7. In Sect. 8, we show the experiment results in the execution of each attack and their defense with BCN-Sec. We conclude our work in Sect. 9.

2 Related work

In this section, we categorize current 802.15.4 defense mechanisms into nonbeacon-enabled mode and beacon-enabled mode according to the literature and highlight their limitations. We also discuss the difference between our attacks on beacon-enabled 802.15.4 networks and others previously demonstrated.

2.1 Defense mechanisms in beacon-less mode

The received signal strength indication (RSSI) has been proposed to identify nodes conducting a Sybil attack [7, 42]. The basic idea of RSSI-based methods is that sensor nodes at different locations can be differentiated by the different RSSIs. Demirbas et al. [7] calculate the ratio of RSSIs to improve the previous RSSI-based solutions. Yang et al. [42] propose a K-means cluster analysis that can be applied to RSSI readings. However, RSSI-based solutions can be evaded by malicious nodes (MNs) with mobility.

Another approach to securing beacon-less 802.15.4 networks focuses on the use of cryptography. Zhang et al. [43] propose lightweight identity certificates to distinguish between LNs and MNs using multiple stolen or forged IDs, while the authors of [10–12, 24] focus on key distribution and management algorithms to provide this protection. However, it is not practical for resource constrained sensor devices to use highly expensive key distribution methods.

Link layer security protocols constitute another category of defense mechanisms for beacon-less 802.15.4 networks. SPINS, TinySec, and MiniSec [19, 26, 32] fall in this category and are designed specifically for energy constrained sensor nodes and provide data authentication and confidentiality in the link layer. However, these protocols are susceptible to failures when an MN in the network (e.g., a compromised node or a malicious insider) acquires a shared pair-wise key or a network-wide secret key.

Moreover, even if their shortcomings are excluded, none of the aforementioned schemes can be directly applied to beacon-enabled 802.15.4 networks. This is because in addition to the data authentication provided by the aforementioned schemes, beacon and command messages must

also be secured. Perrig et al. [32] also state that traditional data authentication techniques cannot be used to provide broadcast beacon authentication.

2.2 Defense mechanisms in beacon-enabled mode

Few defense methods have been proposed for beacon-enabled mode. Amini et al. [2] propose an RSSI solution where they introduced the use of a disc number and a device ID. However, if an MN is close enough to an LN in the same PAN (i.e., an inside attacker), its RSSI will likely be similar to the RSSI of the LN.

The 802.15.4 standard [16] also has security mechanisms to provide data confidentiality and data authenticity. However, Sastry et al. [34] point out that these security mechanisms have vulnerabilities related to the initialization vector (IV) management, key management, and integrity protection. Moreover, the security mechanism only guarantees data authentication, not authentication for beacon broadcasts. Alim et al. [1] introduce EAP-Sens, which provides entity authentication and key management to validate each device ID with the extensible authentication protocol (EAP) [4] using EAP-generalized pre-shared keys (EAP-GPSKs) [6]. However, EAP-Sens has the same problems as the 802.15.4 standard because it is based on the security mechanisms of the 802.15.4 standard.

Overall, neither the aforementioned detection mechanisms nor secure link layer protocols for the beacon-enabled mode are effective in the case of inside attackers. Therefore, we present BCN-Sec, which has less overhead for data authentication than the security mechanisms in the 802.15.4 standard and provides authentication for beacon broadcasts.

2.3 Attacks on beacon-enabled 802.15.4 networks

Sokullu et al. [37] demonstrate GTS attacks on the 802.15.4, particularly in beacon-enabled mode, but they use ns-2 simulations. The GTS attacks in [37] were divided into four different scenarios: One Intelligent Attacker (OIA), One Random Attacker (ORA), Two Intelligent Attackers (TIAs), and Two Random Attackers (TRAs). Their GTS attacks cause collisions to GTSs during the CFP and target either the maximum number of GTSs to an LN or one randomly chosen GTS [37]. In contrast, the six attacks that we present seek to exploit GTSs legitimately (i.e., without causing MAC layer collisions), by sending GTS (de)allocation or broadcasting beacons at a synchronized timeslot during a contention access period (CAP) to illustrate the general vulnerabilities of the protocol.

In addition to presenting different types of attacks compared to those discussed in [37], we implemented our attacks on real devices (i.e., Tmote Sky motes) rather than in simulation. This latter point is extremely important for 802.15.4 MAC layer attacks because in addition to the challenge of accurately modeling physical layer interference, simulations do not account for constraints imposed by the hardware, operating system, and applications, which can lead to simplified attack scenarios.

This is especially pronounced in resource-constrained devices (e.g., Tmote Sky motes). For example, to implement the Sybil attack (at the MAC layer) in TinyOS, we modified the timer function of TinyOS (in TimerC.nc) to make it multithreaded so each fake node could use an instance. Each instance now has to compete internally (within TinyOS) to gain access to the node's resources (e.g., processor, transceiver), making this attack much more difficult to conduct. This small, but noticeable nuance is not present in simulation tools.

We previously introduced four attacks on the beacon-enabled 802.15.4 network [17]. This work extends the previous work in [17] with two new implemented attacks as well as the presentation of BCN-Sec to defend against the attacks.

3 Problem statement

In this section, we briefly explain the concept of beacon broadcasts and the GTS management scheme of the 802.15.4 standard. Additionally, we state the vulnerabilities of these schemes.

3.1 Beacon broadcasts

The 802.15.4 standard [16] defines the superframe (SF) that consists of a CAP, a CFP, and an inactive period for 802.15.4 beacon-enabled mode as shown in Fig. 1. The active period of the CAP and the CFP is divided into 16 timeslots. The timeslots can be synchronized through beacons that the PC periodically broadcasts at intervals defined by the *macBeaconOrder* value.

Upon receiving the beacons, the nodes take the beacon order (BO) and SF order (SO) from the SF specification field in Fig. 2(b) and re-calculates the timeslot interval, SF duration (SD), and beacon interval (BI) for synchronization to the SF of the PAN in Fig. 1.

3.2 Vulnerability of beacon broadcasts

3.2.1 Verification of the PC

The LNs in the PAN rely on the two important values, BO and SO to change their internal timers used for synchronization and transmitting messages. However, when processing the received beacons, the LNs do not authenticate

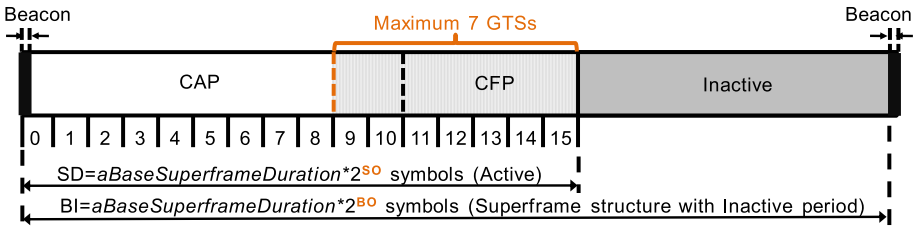
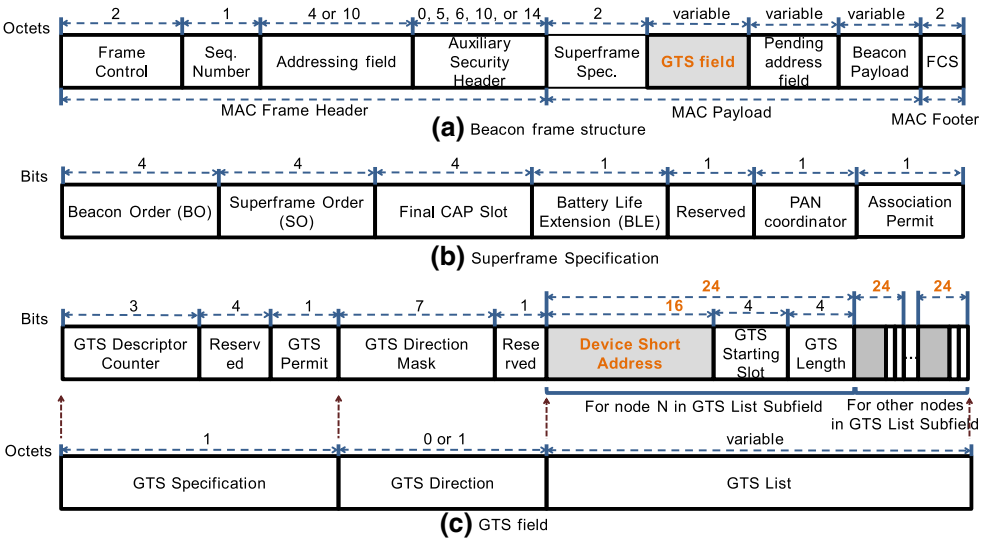


Fig. 1 The superframe structure of beacon-enabled 802.15.4



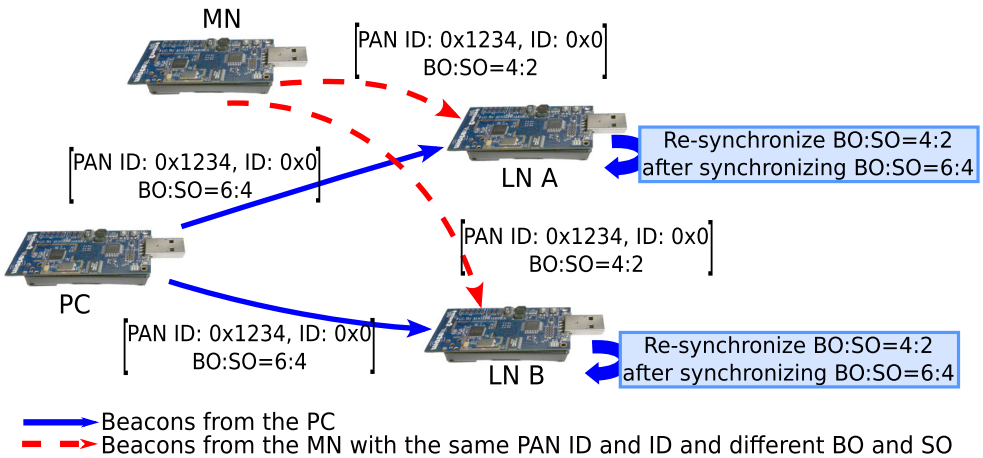


Fig. 3 An MN impersonating the PC and broadcasting false BO and SO with the same PAN ID and the PC's ID

the beacons and cannot tell whether they really come from the PC.

The LNs only confirm that the PAN ID in the packet is the same as the value used during bootstrapping of the network. Thus, if an MN broadcasts malicious beacons with the same PAN ID as that of the PC, the LNs process the beacons as shown in Fig. 3. As mentioned in Sect. 2, the data authentication of the standard does not apply to beacon broadcasts.

3.3 GTS management scheme

The PC in the PAN manages the GTSs during the CFP by adding the GTS field in the beacon frames as responses to the GTS (de)allocation requests as shown in Fig. 2. As shown in Fig. 1, the PC defines that each SF has a maximum of seven GTSs for the CFP other than *aMinCAPLength* in [16]. The PC assigns the LNs issuing GTS allocation requests at the specified GTSs. Then, the

PC releases the assigned slots only after receiving a GTS deallocation request from the same LNs. We briefly explain the normal GTS allocation and deallocation processes below.

3.3.1 GTS allocation

If an LN has data to transmit, it generates a GTS allocation request. The PC will allocate an available GTS to the LN, and all subsequent beacon frames will contain the GTS field defining the device address, GTS slot and direction. Upon receiving the beacon, the LN will schedule the pending packet to be transmitted at the allocated GTS. The GTS allocation process is shown in Fig. 4(a).

3.3.2 GTS deallocation

The GTS deallocation occurs in general when an LN using a specified GTS sends an explicit GTS deallocation request to the PC. The GTS deallocation process is shown in Fig. 4(b).

3.4 Vulnerabilities of GTS management scheme

The PC manages a list of GTSs to control the network access during the CFP. However, the GTS management scheme has the following vulnerabilities.

3.4.1 GTS expiration

According to the 802.15.4 standard, the PC can initiate GTS deallocation when it observes no data transmission from the LNs at the assigned GTSs. The PC deallocates unused GTSs within every $2 \times n$ SFs, where n is defined as either $2^{(8-macBeaconOrder)} (0 \leq macBeaconOrder \leq 8)$ or 1 $(9 \leq macBeaconOrder \leq 14)$. However, an MN can keep its assigned GTSs continuously by simply sending another GTS allocation request or transmitting data at the GTSs. This renders the GTS expiration ineffective.

3.4.2 Verification of sensor nodes' IDs

In the 802.15.4 GTS management scheme, the PC manages the IDs of LNs requesting GTS (de)allocations (i.e., the PC assigns or discards the GTSs by the LNs' IDs). By looking at the IDs, the PC also avoids duplicated GTS (de)allocation requests from the same LNs. However, the PC cannot avoid, for example, a GTS allocation (from an LN) and a deallocation requests (from an MN), where the MN impersonates the LN's ID as shown in Fig. 5. This is because the PC does not authenticate the LNs appropriately. Thus, the MN can easily subvert the verification process for the LNs by using new forged IDs or impersonating LNs in the network.

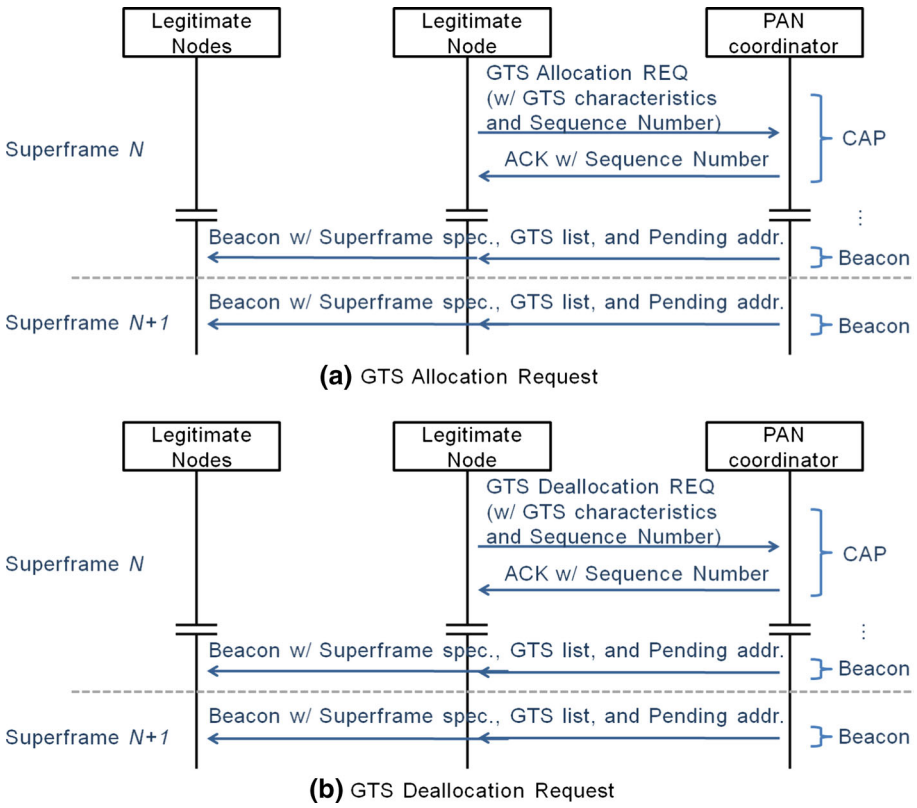


Fig. 4 GTS allocation and deallocation procedure

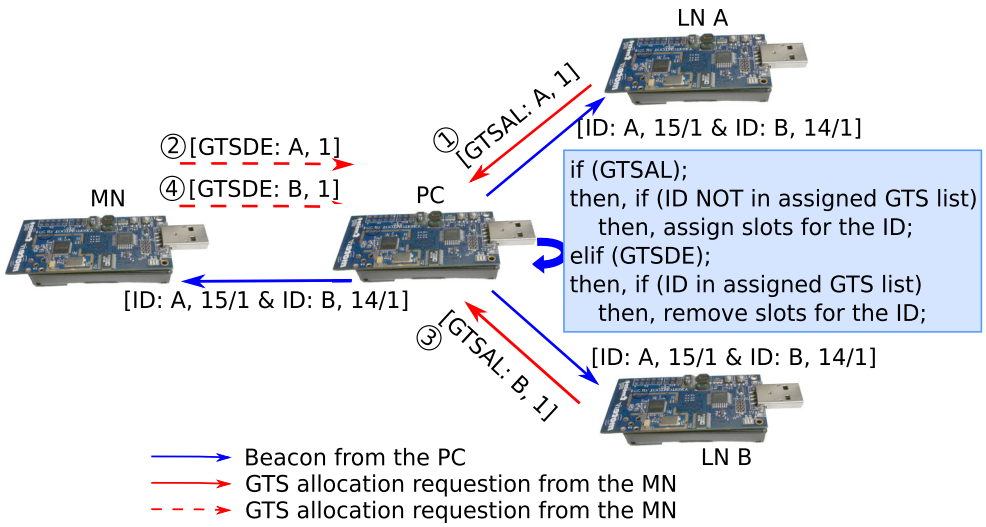


Fig. 5 An MN impersonating LNs A and B IDs

4 Experiment design

In this section, we present the network design and hardware and software components used for the experiments and implementation.

4.1 Network design

In this paper, we deploy wireless sensor nodes supporting the 802.15.4 standard and its beacon-enabled mode. Normally, beacon-enabled 802.15.4 networks consist of few groups of clusters. One cluster can be composed of one PC and a few nodes. For the experiments, we arrange a small cluster that consists of one PC and three LNs including an MN.

The PC broadcasts beacons and receives sensed data from the LNs. The LNs sense the temperature and humidity around the experiment area and transmit the data to the PC during the CAP or the CFP. The LNs do not communicate with one another, but only with the PC (e.g., a unicast message transmission). Only four nodes were used because the open source implementation used became unstable with more than four nodes in the network. However, it is important to note that these attacks are *independent* of the number of nodes deployed in the network.

4.2 Hardware and software components

We used four Tmote Sky motes [29] based on the TelosB platform: one for the PC, two for LNs, and one for the MN. In addition, we used the Texas Instruments (TI) CC2420 Evaluation Board/Evaluation Module (EB/EM) [40] in conjunction with the TI Chipcon packet sniffer [39] to capture and analyze packet traffic in the network.

For the attack implementation, we used Open-ZB [30], an 802.15.4 open source software that supports a beacon-enabled mode. In particular, we used the open source of the ZigBee version in conjunction with TinyOS v2.x [41]. Figure 6 shows the Tmote Sky motes and CC2420 EB/EM.

5 Overview of attacks

In this section, we introduce the attack model and illustrate the overview of the attacks based on the model. We present a total of six attacks and categorize them according to the characteristics of the attacks.

5.1 Attack model

Similar to the threat models defined in [20] and [33], we assume that an MN is a mote-class, inside, and active

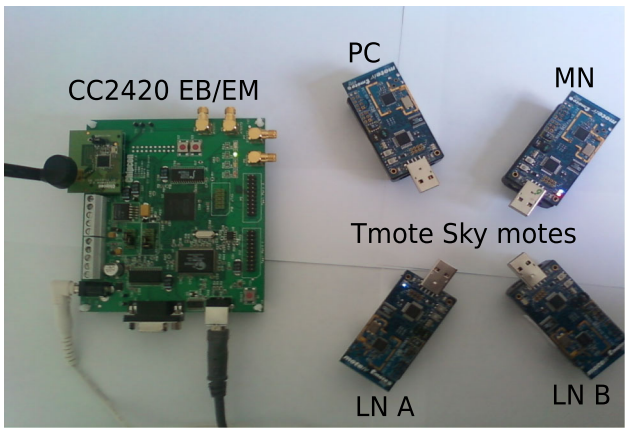


Fig. 6 Tmote Sky motes and CC2420 EB/EM

attacker. We also assume that the network uses the default configuration, that is there is no authentication present between the LNs and the PC and is operated in the beacon-enabled mode of the 802.15.4 standard [16]. The MN can eavesdrop the communication between the PC and the LNs within the same range of the beacon-enabled 802.15.4 network. Moreover, due to no authentication, the MN can easily impersonate both the PC and the LNs.

By eavesdropping and impersonation, the MN aims to achieve a few types of denial of service (DoS) attacks. In particular, the MN targets the beacon and CFP through the DoS attacks that are very critical for timeslot synchronization in the beacon-enabled network. For example, if the CodeBlue infrastructure [25] does not prevent the MN from impersonating the PC and the LNs, a DoS attack can prevent legitimate recipients from receiving time-sensitive alerts and medical information.

Given these assumptions, we deploy one Tmote Sky mote as the MN that has the same capabilities as the LNs. The MN is located near the LNs in the beacon-enabled 802.15.4 network. The MN listens to beacons from the PC to get synchronization information of the network and GTS (de)allocation requests from the LNs in the passive phase. In the active phase, the MN exploits vulnerabilities of the beacon broadcasts and the GTS management scheme by impersonating either the PC or the LNs.

Table 1 presents the summary of the attacks and the exploitation of the vulnerabilities in the beacon-enabled 802.15.4 network.

5.2 Impersonating existing identities in the PAN

In this category, we describe four attacks. The first attack presented is the synchronization attack. In this attack, the LNs are lead to synchronize their SF timeslots with the manipulated beacons from the MN. The next two attacks block data transmission from the LNs in the PAN that want to gain GTSs and transmit time-sensitive data in the slots. The fourth attack injects false sensed data into the traffic stream from an LN to the PC during the CFP.

5.2.1 *Synchronization attack*

This attack influences all the LNs in the network concurrently, whereas the other attacks in this category can affect only one or a few LNs. The MN first impersonates the PC's ID and uses the same PAN ID as that of the PC. The MN manipulates two important parameters: BO and SO as shown in Fig. 1. To compete with the beacons from the PC, the MN adjusts the BO and SO to have more SFs within one SF of the PC.

Figure 7 shows two different SF sequences. Figure 7(a) has the SFs configured with BO and SO (6 and 4) from the legitimate PC whereas Fig. 7 has the SFs with BO and SO (4 and 2) from the MN. Thus, the MN can compete with the PC by having four more SFs in one SF from the PC. This causes the LNs to constantly synchronize their SFs with the manipulated BO and SO (4 and 2) arriving immediately after those from the PC although they process the beacons from both the PC and the MN.

5.2.2 *DoS of data transmission*

Impersonating LNs: If an MN is in the transmission range of the PC, it can easily obtain the IDs of active LNs in the PAN. The MN also knows whether or not an LN tries to transmit its sensed data during the CFP by looking at the GTS allocation requests or the beacons. In this attack, the MN impersonates the active LNs in the PAN and sends GTS deallocation requests using the LNs' IDs to the PC.

Table 1 Attacks and the characteristics

Attacks		Unauthenticated Message (Direction)	Node impersonated	Vulnerabilities	Message transmission types
Synchronization attack		Beacon (PC → LN)	Existing PC	Beacon broadcasts	Broadcast
DoS of data transmission	Impersonating LNs	GTS allocation request (LN → PC)	Existing node IDs	GTS management scheme	Unicast
	Impersonating the PC	Beacon (PC → LN)	Existing PC	Beacon broadcasts	Broadcast
False data injection		Data (LN → PC)	Existing node IDs	GTS management scheme	Unicast
DoS of GTS requests		GTS allocation request (LN → PC)	Non-existing node IDs		
Stealing network bandwidth					

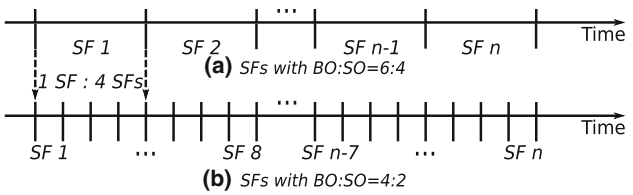


Fig. 7 The two different SF sequences, (**a**) and (**b**), from the PC and the MN respectively: (**a**) with BO:SO = 6:4 from the PC and (**b**) with BO:SO = 4:2 from the MN

Figure 8(a) shows an example of this attack. While two LNs request GTS allocation to transmit data in the next SF's CFP, the MN can terminate the data transmissions of the LNs by sending a GTS deallocation request with the LNs' IDs. Since the PC receives the GTS deallocation request while processing the GTS allocation from the LNs, it ignores the GTS allocation coming first and does not assign any GTS to the LNs. As a result, the LNs that do not have any assigned GTS cannot transmit its sensed data.

Impersonating the PC: The previous attack impersonates the LNs' IDs to cause the PC to deallocate the GTSs. In this attack, the MN impersonates the PC and broadcasts manipulated beacons with no GTS descriptor. When the LNs request GTS allocation to transmit its sensed data, the LNs wait for beacons coming with the GTS descriptors that tell them the assigned GTS information [e.g., address, slot, and length as shown in Fig. 2(c)].

If the beacons that the LNs receive do not have any GTS descriptor, the LNs assume that they cannot transmit the sensed data to the PC due to no GTS being allocated. Thus, the MN impersonating the PC keeps sending a manipulated beacon without GTS descriptors while the PC broadcasts a beacon with GTS descriptors. Since the LNs just process the last beacon, if there are more beacons received within the proper boundary of the timeslot, it receives the manipulated beacons from the MN and believes that no GTS is assigned to it by the PC. Thus, it does not transmit data to the PC as shown in Fig. 8(b).

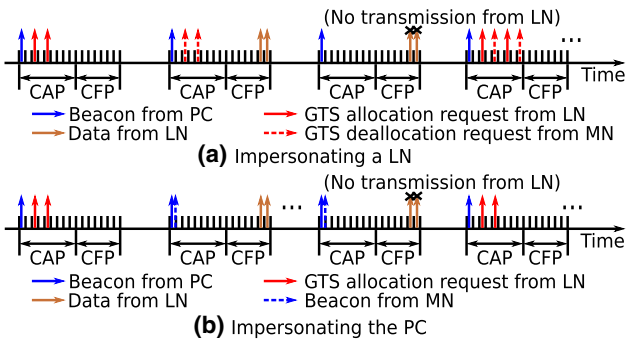


Fig. 8 An MN blocking an LN sending data during the CFP

5.2.3 False data injection

In this attack, the MN identifies that an LN has not requested GTS allocation by looking at the GTS descriptors in the beacons. Then, the MN chooses the LN's ID that does not have any GTS allocation requests and sends a GTS allocation request using that ID. After it confirms that a GTS is allocated by the PC, the MN sends false data with the ID to the PC during the CFP while the LN sends its sensed data during the CAP. The PC regards the false data as time-sensitive data from the LN due to it being sent with the same LN's ID during the CFP. The false data sent by the MN is passed to the application.

Figure 9 shows how this attack works; for instance, when an LN is transmitting current temperature data during the CAP, the MN sends a GTS allocation request with the spoofed ID, pretends to be another LN, and can inject false temperature data during the CFP.

5.3 Impersonating non-existing identities in the PAN

In this category, an MN forges up to 7 different IDs depending on the maximum number of available GTSs. The two attacks presented in this section occupy all 7 GTSs, which result in no GTS being available for the LNs.

5.3.1 DoS of GTS requests

The goal of this attack is *not* for the MN to use the bandwidth requested, rather it is to prevent the LNs from transmitting data during the CFP. To perform this attack, an MN continuously monitors the available GTS slots with the intent of completely occupying them. Then, the MN sends several GTS allocation requests to fill up all the available GTSs in the SF. The advantage of this attack is that the MN can reduce its energy consumption, because once it occupies all 7 GTSs, it does not need to send out further any GTS allocation requests.

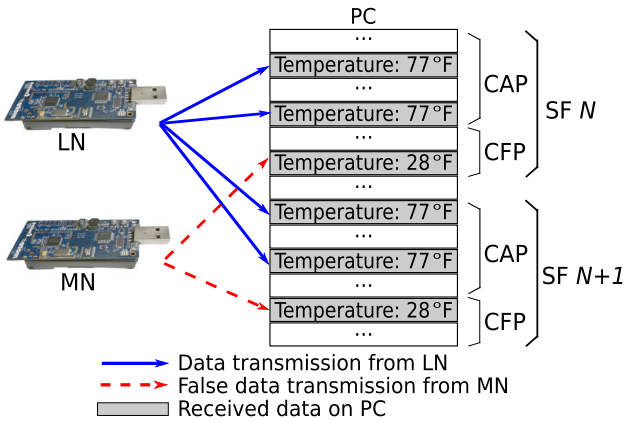


Fig. 9 An MN sending false temperature to the PC

If the PC performs the GTS expiration, the MN starts sending GTS allocation requests until it occupies all 7 GTSs. The MN repeats the aforementioned steps to ensure that the LNs do not have access to available GTSs. Figure 10 shows how the MN takes two GTSs of the LNs (while currently occupying ① five GTS allocation requests) as follows: (1) the MN monitors ② and ③ GTS deallocation requests from the LNs A and B. (2) Then, the MN completely fills all 7 GTSs with ④ two additional GTS allocation requests.

5.3.2 *Stealing network bandwidth*

Similar to the DoS of GTS requests, the MN observes the GTS list in order to eventually occupy the available GTSs. However, in this attack, the MN transmits data at the assigned timeslots. The purpose of data transmission is to prevent the PC from dropping the assigned GTSs.

As shown in Fig. 11, the second CFP has data transmitted from both LNs and an MN. However, since the LNs send GTS deallocation requests during the third CAP, the MN sends a GTS allocation request to occupy the new free GTS. Eventually, only the MN sends data during the fourth CFP. The timeslots will never be vacant during the CFP of every SF.

6 Implementation of attacks

In this section, we introduce the application layer and MAC layer modules that were implemented to execute the six attacks described in the previous section. We explain in detail how the MN runs the attacks in the PAN.

6.1 Attack modules for implementation

We have implemented our attacks based on the existing modules provided by Open-ZB. Given the modules in the

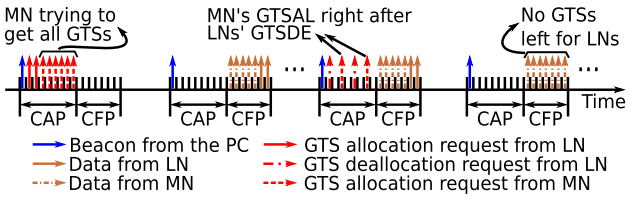


Fig. 11 An MN stealing network bandwidth in all 7 GTSs during CFP

MAC layer of 802.15.4, we mainly modified the source code of the MAC layer for our attacks and added a malicious application (MAC misbehavior app) as shown in Fig. 12. The modified MAC layer and the malicious application target the vulnerabilities of the GTS management scheme described in Sect. 3.

We have two options for implementing the MAC misbehavior attacks: the first option is to implement a module in the application layer, while the second option relies on the implementation of modules in the application and MAC layers. We present the discussion of the most efficient approach, implementation in the MAC layer, and point the reader to the technical report [18] for the discussion on the application layer implementation.

In the MAC layer, we implement the MAC misbehavior attacks by adding *Mal-PD_DATA management* as shown in Fig. 13. *Mal-PD_DATA management* intercepts a function, *PD_DATA.indication()*, which indicates all packets (e.g., beacon, command (GTS request), data) of the communication in the PAN. Then, it directly executes the attacks in the MAC layer. For instance, if the packet is a beacon from the PC, *Mal-PD_DATA management* looks at available GTSs and directly calls *Mal-GTS management* to fill all remaining GTSs. If the packet is a GTS allocation request from an LN, *Mal-PD_DATA management* informs *Mal-GTS management* of the LN's ID. Then, *Mal-GTS management* sends a GTS deallocation request with the ID. This allows our attacks to be more efficiently executed with

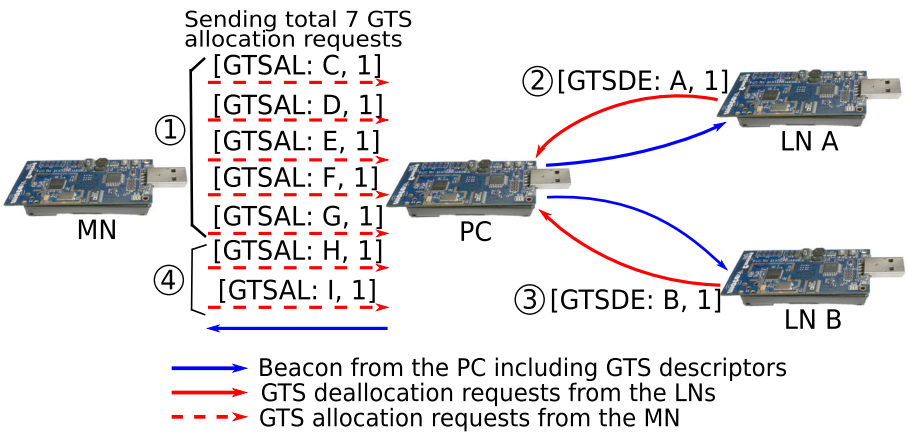


Fig. 10 An MN filling up all 7 GTSs

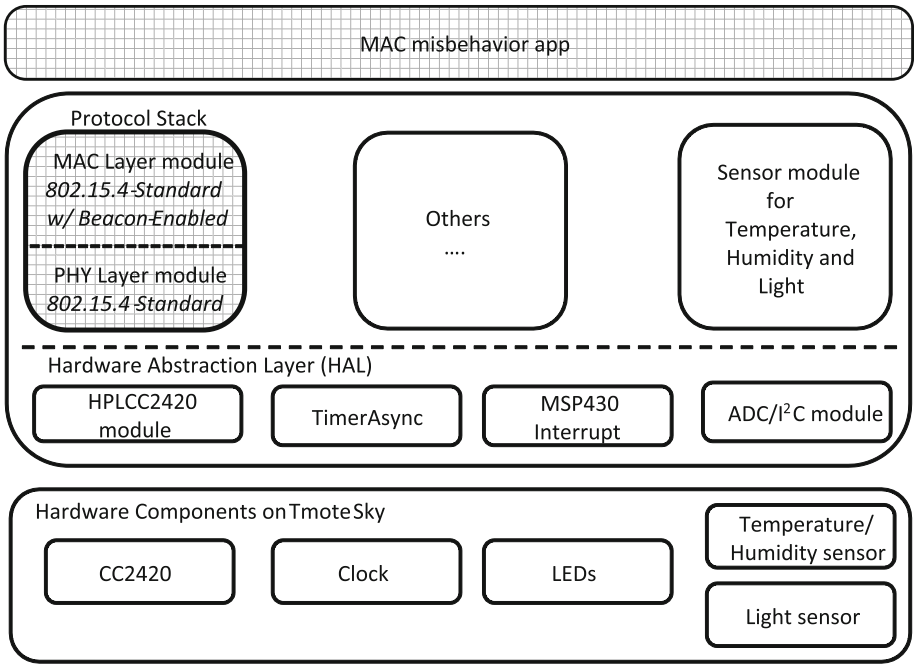


Fig. 12 The software and hardware modules of the Tmote Sky mote. The software modules consist of the general modules of TinyOS 2.x and the protocol stack of Open-ZB. The region with the *gray perpendicular lines* represents the modified modules for the MN

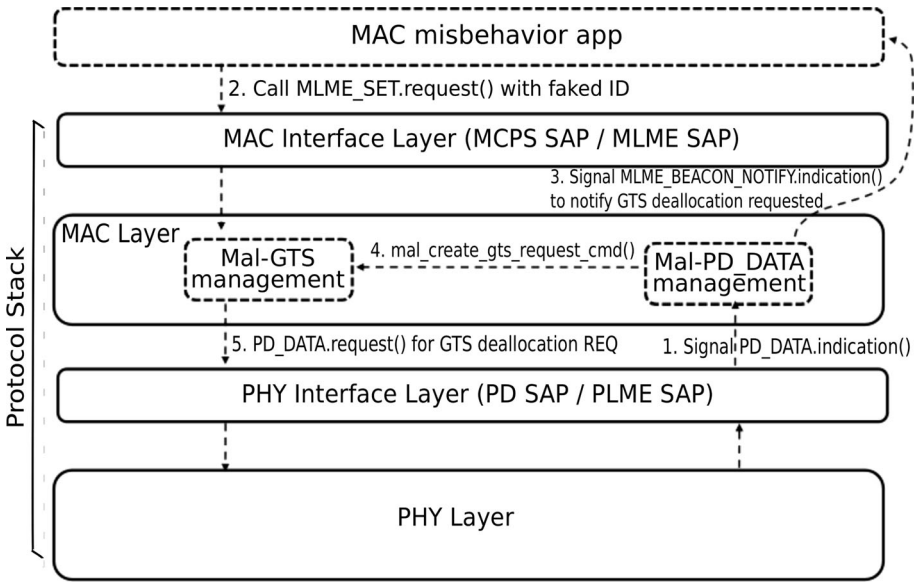


Fig. 13 The attacks improved in the MAC layer with the *Mal-PD_DATA* management module

lower communication overhead. The reader is pointed to the technical report for the detailed algorithms for implementing each attack.

7 Securing beacon-enabled 802.15.4

In this section, we propose an applied security protocol *BCN-Sec* dedicated to the beacon-enabled 802.15.4 network under the assumptions in Sect. 5.1 and present an overview and the implementation of *BCN-Sec*.

BCN-Sec aims to provide both unicast authentication and broadcast authentication that can defend against the aforementioned attacks. Table 2 shows that the attack/countermeasure relationships for the attacks considered in this paper. Unicast authentication can defend against the following attacks: DoS of data transmission by impersonating LNs, False data injection, DoS of GTS requests, and Stealing network bandwidth. In addition, broadcast authentication can defend against the following attacks: Synchronization attack and DoS of data transmission by impersonating the PC.

Table 2 Security requirements of BCN-Sec

Attacks	Security requirements	
	Unicast authentication	Broadcast authentication
Synchronization attack	N	Y
DoS of data transmission		
Impersonating LNs	Y	N
Impersonating the PC	N	Y
False data injection	Y	N
DoS of GTS requests	Y	N
Stealing network bandwidth	Y	N

We have implemented BCN-Sec which is based on a cipher block chaining message authentication code (CBC-MAC) (message integrity code is substituted for MAC henceforward to avoid confusion with MAC) 128-bit with the advanced encryption standard (AES) (CBC-MIC henceforward) and one-way key chains. The CBC-MIC is sufficient for message authentication since it can process arbitrary length messages due to the block chaining method and has reasonable performance for authentication with AES (due to a minimum number of cipher function calls [14]).

Like the 802.15.4 standard [16], BCN-Sec uses the CBC-MIC for the PC to authenticate data and control messages from the LNs. This enables BCN-Sec to prevent an MN from transmitting forged data messages (e.g., false data injection). BCN-Sec can also defend against the exploitation of forged GTS (de)allocation requests (e.g., DoS of data transmission by impersonating LNs, DoS of GTS requests, and stealing network bandwidth).

Further, in BCN-Sec, broadcast authentication using one-way key chains is used for the LNs to authenticate beacons from the PC whereas the 802.15.4 standard relies on a network-wise shared key. This enables BCN-Sec to prevent an MN from broadcasting forged beacons (e.g., synchronization attack and DoS of data transmission by impersonating the PC).

Figure 14 illustrates the comparison between the frames without security "No security" and the frames for BCN-Sec's unicast and broadcast authentication.

The following sub-sections explain BCN-Sec's unicast and broadcast authentication operations and implementation in detail.

7.1 Unicast message authentication

In several of our attacks, an MN is able to send GTS (de)allocation requests with forged IDs. This is possible because the PC does not sufficiently authenticate control messages (GTS requests). To prevent these attacks, *unicast* command and data frames are authenticated with a MIC using a keyed cryptographic hash function by the PC.

7.1.1 Implementation

We use a hardware (HW) CBC-MIC provided by the CC2420 chip on the Tmote Sky mote (HW-MIC) [38]. The HW-MIC is one of CC2420's MIC security operations. It uses a 128-bit key that can be set on demand and generates a 128-bit MIC for the LNs from any size unicast messages.

Compared to the "No security" unicast message in Fig. 14(b) and (c), BCN-Sec adds a 4-byte *security header* (*SH*) as shown in Fig. 15 and a 16-byte *MIC* for unicast authentication in Fig. 14(b′) and (c′). Each field in the *SH* is used to configure the HW-MIC except the *key counter* (*KCT*) field. The 2 bytes of *KCT* increase the length of the *sequence number* (*SN*) to a total of 3 bytes. BCN-Sec also uses an internal 4-byte *secret counter* (*SCT*) for each LN as a nonce that is included in the *MIC*.

7.1.2 Authentication procedure

The following illustrates how BCN-Sec applies the HW-MIC to unicast messages with a control message (a command frame):

7.1.3 Key setup

In BCN-Sec, each LN has pre-shared keys and *SCTs* with the PC (K_{LNi} and SCT_{LNi} respectively, where i is $\{1, 2, \ldots, n\}$, the index of each LN in the PAN) before the LNs are deployed.

7.1.4 Transmitting control messages

When *LNi* transmits a unicast control message [CMD_{LNi} as shown in Fig. 14(b)] to the PC, *LNi* first generates a *MIC* (MIC_{LNi}) for CMD_{LNi}. MIC_{LNi} can guarantee freshness since it includes the *SN* (1-byte) in the frame control field as shown in Fig. 14, the *KCT* (2-bytes) as shown in Fig. 15, and the *SCT*[1] (4-bytes) pre-shared with the PC. The following shows a BCN-Sec unicast control message $BCNSECCMD_{LNi}$ corresponding to Fig. 14(b′)

[1] Including the SCT inside of the MIC as opposed to transmitting an encrypted nonce in the frame can lead to synchronization problems if packets are lost. However, since beacon-enabled 802.15.4 uses link-layer acknowledgments and retransmissions, the chance of losing synchronization is minimal.

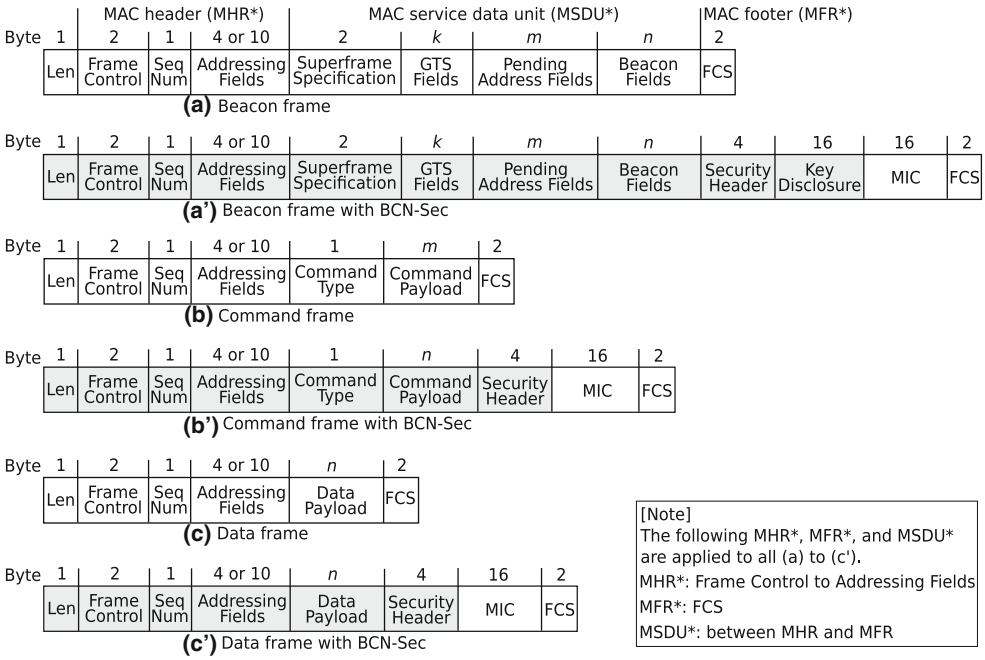


Fig. 14 The frames of the 802.15.4 MAC with "No security" in (**a**), (**b**), and (**c**) and BCN-Sec in (**a**′), (**b**′), and (**c**′). In (**a**′), *Key Disclosure* is a disclosed key in one-way key chains. The *shaded regions* are authenticated fields

$$LNi \xrightarrow[BCNSECCMD_{LNi}]{} PC$$
$$= Len||MHR||MSDU||SH||$$
$$MIC_{LNi}(K_{LNi}, Len|MHR|MSDU|SH|SCT_{LNi})$$
$$||MFR$$

where *Len* is the length of the frame and *MHR*, *MSDU*, and *MFR* represent a MAC header (MHR), a MAC service data unit (MSDU), and a MAC footer (MFR) for the CMD_{LNi}. BCN-Sec adds *SH* and MIC_{LNi} to *MSDU* for $BCNSECCMD_{LNi}$. MIC_{LNi} covers the *SH SH* and all the fields of CMD_{LNi} except *MFR* to be authenticated.

7.1.5 Verifying control messages

The PC takes *Len*, *MHR*, *MSDU*, and *SH* from $BCNSECCMD_{LNi}$ and SCT_{LNi} pre-shared. Then, it generates a MIC from them with K_{LNi}. If the MIC and MIC_{LNi} match, the PC confirms that $BCNSECCMD_{LNi}$ came from *LNi*.

7.2 Broadcast authentication

While unicast authentication provides the PC with the trust of command and data messages that are transmitted from the LNs, broadcast authentication provides an efficient way to ensure the authenticity of the PC's beacon broadcasts. Unlike unicast command and data messages sent by multiple LNs to a single PC, beacon messages are *broadcast* to all the LNs in the PAN.

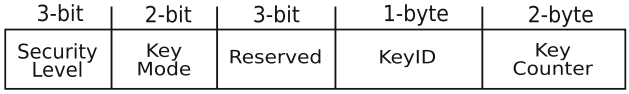


Fig. 15 The *security header* (4-bytes) of BCN-Sec

During *broadcasts*, the LNs share the same key as the PC in a similar fashion to unicast authentication. However, if this key is not dynamic and is known to every LN, any LN can masquerade as the PC. For this reason, BCN-Sec uses one-way key chains [23] that use a one-way function to generate a sequence of keys. Many techniques similar to one-way key chains have been proposed for various cryptographic applications and particularly for broadcast authentication [32, 44].

7.2.1 Implementation

For the one-way key chain, we have ported an MD5 implementation in C [8] to TinyOS. Then, we coupled its hash function (*one-way*) with the HW-MIC. In BCN-Sec, the MD5 hash function is used to generate multiple sets of one-way key chains for the PC[2] (e.g., $K_1 = \{k_{(1,1)}, k_{(1,2)}, \ldots, k_{(1,j)}\}$, …, $K_i = \{k_{(i,1)}, k_{(i,2)}, \ldots, k_{(i,j)}\}$, where K is a set of multiple key

[2] We assume that the PC is more powerful and has storage requirements for the key chain (e.g., shimmer [36]) since the PC is required to store the entire key chains. The storage requirements for the LNs is minimal (e.g., only one key for unicast authentication and the last key(s) of one-way key chains).

chains and k is a key in the key chain). Only the PC knows all the keys in the key chains, and it pre-shares the last keys (e.g., $k_{(1,j)}, k_{(2,j)}, \ldots, k_{(i,j)}$) in the key chains with the LNs during bootstrapping.

The LNs cannot get any previous key from the last keys since the MD5 hash function is *one-way*. Thus, the PC discloses the previous key (e.g., $k_{(i,j-1)}$) from the key chains in the *key disclosure* (*KD*) field as shown in Fig. 14(a′) whenever it broadcasts beacons (i.e., one key per broadcast). The PC also discloses two references for the previous key $k_{(i,j-1)}$: (1) the key ID (i) in the *KeyID* field and (2) the key index ($j-1$) in the *KCT* field of the *SH*. The key ID (i) refers to a set of the key chains, and the key index ($j-1$) refers to an index in the key chain.

The LNs can then verify the key $k_{(i,j-1)}$ through the pre-disclosed key with the MD5 hash function. The key disclosure in BCN-Sec considers two special cases: (1) the disclosed key is out-of-sync. (2) the PC desires to change the key chain (possibly because it suspects that a key chain has been compromised).

If an LN has missed one or more beacons and the key index is out-of-sync, it can verify that the key (e.g., $k_{(i,j-1)}$) is at its ID (i) and index ($j-1$) in the key chains by using the last key $k_{(i,j)}$. For instance, if the key index is 20 (out of 50 keys) on the first key chain (the key $k_{(1,20)}$ and the last key $k_{(1,50)}$), BCN-Sec executes the MD5 hash function with the key n times ($(F(k_{(1,20)})^n = k_{(1,50)}$) to get the last key, where n is 50 (total keys) -20 (key index).

If the PC wants to change the key chain (e.g., from K_1 to K_2), it turns on the reset flag (1-bit) in the *Reserved* field in the *SH* in Fig. 15. Then, the PC discloses the next to last key in the new key chain ($k_{(2,j-1)}$) in the *KD* field (because the LNs are loaded with several initial keys for various key chains pre-deployment). The obvious weakness with this approach is that after all of the key chains have been exhausted, the LNs would need to be provided a new set of keys offline. However, as we plan to do in our future work, BCN-Sec[3] can be implemented such that new initial keys from new key chains can be disclosed in beacon frames. This will increase the size of the beacon frame during this disclosure period, but will obviate the need for getting new keys offline.

After the LNs obtain the disclosed key, the LNs need to authenticate the PC's beacon by using the HW-MIC and the key.

7.2.2 Authentication procedure

The following shows how BCN-Sec applies the HW-MIC and the one-way key chain to broadcast messages (i.e., a beacon message):

Key setup: The PC generates the key chains ($K_i = \{k_{(i,0)}, k_{(i,1)}, k_{(i,2)}, \ldots, k_{(i,j-1)}, k_{(i,j)}\}$) from $k_{(i,0)}$ by using MD5 hash function F. Thus, $k_{(i,j)} = F(k_{(i,j-1)})$, where j is the maximum number of keys (new keying material would be exchanged prior to $j=1$ to continue the secure communications). Then, the last keys $k_{(i,j)}$ are pre-shared between the PC and the LNs before they are deployed.

Broadcast beacons: The PC sends the first message using the next to last key in the key chain (K_i). The pattern continues and the PC traverses the chain backwards, using the previous key in the chain for the next transmission. The key expires after each transmission and the function that generates the key is *one-way* so that even LNs cannot masquerade as the PC.

When the *PC* broadcasts a beacon message [$BCN_{(i,j-1)}$ as shown in Fig. 14(a)] to the LNs, the *PC* enables the HW-MIC to add the *KD* $k_{(i,j-1)}$ and generates a *MIC* ($MIC_{(i,j-1)}$) for $BCN_{(i,j-1)}$.

The following shows a BCN-Sec beacon message $BCNSECBCN_{(i,j-1)}$:

$$PC \xrightarrow[BCNSECBCN_{(i,j-1)}]{} LNs$$
$$= Len||MHR||MSDU||SH||k_{(i,j-1)}||$$
$$MIC_{(i,j-1)}(k_{(i,j-1)}, Len|MHR|MSDU|SH|k_{(i,j-1)})$$
$$||MFR$$

where $(i, j-1)$ is a sequence of the key chains, i is changed once the PC desires to change to a new key chain, and j is decremented by 1 after each transmission while $j>0$. BCN-Sec adds *SH*, $k_{(i,j-1)}$, and $MIC_{(i,j-1)}$ to *MSDU* for $BCNSECBCN_{(i,j-1)}$. $MIC_{(i,j-1)}$ covers the *SH SH*, the *KD* $k_{(i,j-1)}$, and all the fields of the beacon $BCN_{(i,j-1)}$ to be authenticated.

Verifying beacon: The LNs first take $k_{(i,j-1)}$ from $BCNSECBCN_{(i,j-1)}$ and verify it with the MD5 hash function F. $k_{(i,j)} = F(k_{(i,j-1)})$, where $k_{(i,j)}$ is already verified in the previous beacon or trusted in the case of the initial key disclosed pre-deployment. If $k_{(i,j-1)}$ is verified, the LNs generate a MIC from the fields: *Len*, *MHR*, *MSDU*, *SH*, and $k_{(i,j-1)}$ from $BCNSECBCN_{(i,j-1)}$. If the MIC and $MIC_{(i,j-1)}$ match, the LNs confirm that $BCNSECBCN_{(i,j-1)}$ came from the PC. That is, the LNs receive the previous key in the chain and assume that it is valid if its hash is the current key. Thus, using one-way key chains enables the PC to efficiently provide authentication to its beacon broadcasts.

7.3 BCN-Sec communication costs

In this section, we present the communication costs of BCN-Sec. Figure 14(a–c) shows beacon, command, and

[3] It is also important to note that the current implementation can be extended to defend against a birthday attack as in [15].

data frames with "No security". Figure 14(a′–c′) presents the additional fields in the frames using BCN-Sec compared to "No security" and the length. In addition to the *SH* (4-bytes), BCN-Sec adds the *KD and MIC* (16-bytes each) to regular beacons for broadcast authentication and the *MIC* (16-bytes) to regular command and data frames for unicast authentication.

As a comparison to BCN-Sec, we estimated the security mechanisms of the 802.15.4 standard and chose one of the security mechanisms, MIC-128 [16], which is similar to the HW-MIC. We use the label "Security" for MIC-128 henceforward. Enabling "Security" adds a 6-byte *auxiliary security header* and a 16-byte *MIC*.

Table 3 shows a summary of the communication costs of BCN-Sec with its viability during different attacks. In the table, we present the packet sizes (*MHR*, *MSDU*, and *MFR*) from our experiments (beacon fra-mes with all 7 GTS descriptors, command frames with GTS allocation request, and data frames with 2-byte data). The packet size in total shows the increased overhead (communication cost) and viability of "Security" and BCN-Sec from "No security."

In beacon-enabled 802.15.4 networks, BCN-Sec can prevent the synchronization attack and DoS of data transmission by impersonating the PC with 94.7 % overhead whereas "Security" is not a viable solution due to the network-wise shared key. Moreover, BCN-Sec adds 166 % overhead to secure unicast control messages (ST'S (de)allocation requests) whereas "Security" has 183 % overhead (17 % more).

For unicast data message authentication, BCN-Sec has 125 % overhead whereas "Security" provides the same authentication as BCN-Sec with 137.5 % overhead (12.5 % more). Thus, although BCN-Sec has the same level of authentication as MIC-128 for unicast messages, it has less communication overhead than "Security" of the standard. In addition, BCN-Sec provides broadcast authentication.

8 Attack analysis

We have verified our implementation with the TI packet sniffer [39] during various attacks. We show three phases for each attack experiment: *No attack*, *Attack*, and *BCN-Sec Enabled During Attack*.

The throughput given in Figs. 16, 17, and 19 are based on the total number of data in bits per second. The data is counted only during the CFP and does not include the *SH* and the *MIC* (when BCN-Sec is enabled). As a result, the data throughput when BCN-Sec is enabled is lower than "No Security." For each of the six attacks, we measured the packet transmission for 70–230 s depending on the complexity of each attack.

8.1 Synchronization attack

In this attack, the MN impersonates the PC and broadcasts manipulated beacons (i.e., for a shorter SF) to LN2. Then,

Table 3 The viability and communication costs of BCN-Sec

Attacks	Security	Viability	Communication cost (Bytes)					Increase over no security
			MHR	MSDU	MFR	Security overhead[a]	Total	
• Synchronization attack • DoS of data transmission -. Impersonating the PC	No security (beacon)	N	10	26	2	-	38	-
	Security	N	16	42	2	22	60	57.8%
	BCN-Sec	Y	10	62	2	36	74	94.7%
• DoS of data transmission -. Impersonating LNs • DoS of GTS requests • Stealing network bandwidth	No security (GTS request)	N	8	2	2	-	12	-
	Security	Y	14	18	2	22	34	183%
	BCN-Sec	Y	8	22	2	20	32	166%
False data injection	No security (data)	N	12	2	2	-	16	-
	Security	Y	18	18	2	22	38	137.5%
	BCN-Sec	Y	12	22	2	20	36	125%

Security overhead[a] is the size of security materials: "Security" with *auxiliary security header* and *MIC* and BCN-Sec with *SH*, *MIC*, and *KD* (only for beacon frame)

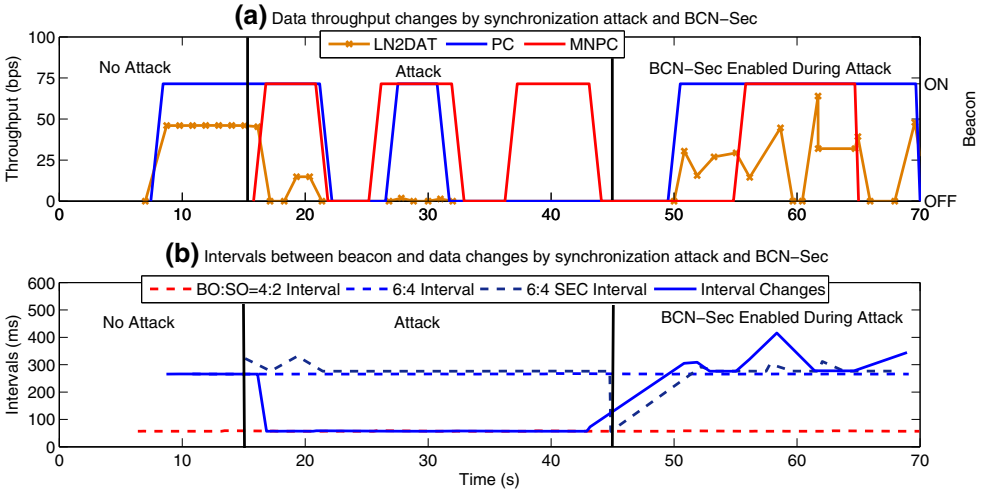


Fig. 16 LN2 data throughput changes after synchronizing with beacons from MN and after BCN-Sec enabled. The intervals are between the first beacon and the first data packet in SFs

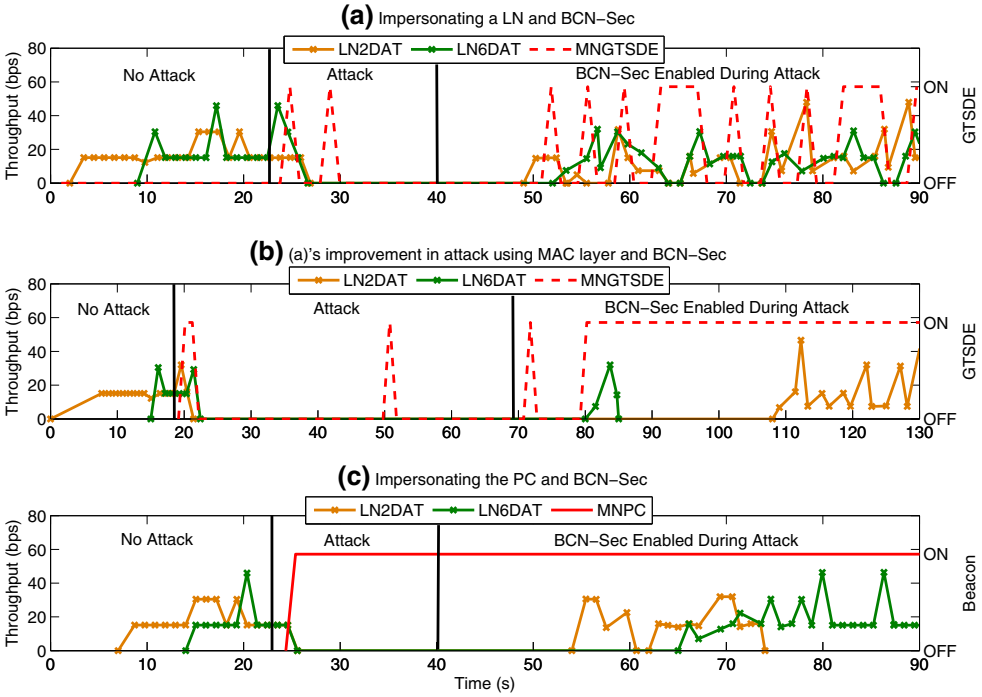


Fig. 17 LN2 and LN6 data throughput during CFP blocked by MN and BCN-Sec enabled. LN2DAT and LN6DAT: Data from LN2 and LN6, MNGTSDE: GTS deallocation requests from MN, and MNPC: beacons from MN

LN2 synchronizes with the manipulated beacons as shown in the *Attack* phase. Figure 16(a) shows LN2's data throughput and (b) the intervals between the beacon and the first data packet in the SF in three phases of the experiments.

Figure 16(a) illustrates that while the data throughput of LN2 is increasing up to 50 bps with the interval at about 265 ms, the MN starts broadcasting malicious beacons with BO (4) and SO (2) around the 15-s mark whereas the PC broadcasts beacons with BO (6) and SO (4). At this moment, LN2 synchronizes the SFs with the MN's beacons and transmits data to the MN with the interval of about 56 ms [which corresponds to BO (4) and SO (2)]. As a result, LN2's data throughput (to the legitimate PC) starts decreasing. Figure 16(b) shows the patterns of the interval changes (265 ms (the *No Attack* phase) → 56 ms (the *Attack* phase → 277[4] ms (the *BCN-Sec Enabled During Attack* phase)) while the synchronization attack is executing and BCN-Sec is enabled.

Once BCN-Sec is enabled around the 45-s mark in both Fig. 16(a) and (b), LN2's data throughput increases to 50

[4] The difference of the interval during BCN-Sec enabled compared to 265 ms of the *No Attack* phase is 12 ms because of the processing overhead added by BCN-Sec

bps. In addition, the interval increases to 277 ms accordingly, which is a proper interval for BCN-Sec.

8.2 DoS of data transmission

As mentioned in Sect. 6.1, the attacks can be implemented at the Application or MAC layers. In this section, we illustrate why we chose to implement to implement the attacks at the MAC layer.

Impersonating LNs at the Application Layer: In this attack, the MN impersonates LN2 and LN6 and sends GTS deallocation requests to block LN2 and LN6's data transmission during the CFP. Figure 17(a) shows the decline of data throughput on LN2 and LN6 while the MN is sending GTS deallocation requests using LN2 and LN6's IDs.

Around the 25-s mark and 30-s mark, the MN sends GTS deallocation requests against LN2 and LN6's GTS descriptors in the beacon. This makes the PC drop GTS descriptors for LN2 and LN6. Therefore, the data throughput from LN2 and LN6 during the CFP drops to 0 bps. Even though LN2 and LN6 try to send GTS allocation requests, the requests cannot be processed due to the MN continuously sending GTS deallocation requests. However, the data throughput from LN2 and LN6 increase after BCN-Sec is enabled around the 40-s mark, even while the MN continues to send GTS deallocation requests.

Impersonating LNs at the MAC Layer: By modifying the MAC layer (as discussed in Sect. 6.1), the MN can send faster GTS deallocation requests; fast enough to be sent immediately following beacons and LN2 and LN6's GTS allocation requests. This leads to the same result of blocking data transmission, but allows the *Attack* phase to begin much sooner. Thus, Fig. 17(b) shows that the data throughput from LN2 and LN6 drops around the 20-s mark, which is earlier than when it occurred in Fig. 17(a) (around the 27-s mark).

In Fig. 17(b), in addition to the MN's first GTS deallocation requests around the 20-s mark as shown in Fig. 17(a), the MN also blocks LN2 and LN6's GTS allocation requests around the 50-s mark, which shows that the MN efficiently sends GTS deallocation requests only when LN2 and LN6 send GTS allocation requests. After BCN-Sec is enabled around the 70-s mark, the MN is still sending GTS deallocation request, and the PC is ignoring these request because BCN-Sec is enabled. However, the throughput from LN2 and LN6 only begins to increase after the 80-s mark. This is because the quick deallocation in response to the constant GTS allocation request caused collisions and an inadvertent brief DoS attack (which is not the focus of BCN-Sec).

Impersonating the PC: In this attack, the MN impersonates the PC and broadcasts manipulated beacons with no GTS descriptors. Figure 17(c) shows that the MN starts sending the same beacons without GTS descriptors (i.e., no GTS descriptors in the beacons) around the 25-s mark, which immediately cripples LN2 and LN6's data throughput. LN2 and LN6 process only the manipulated beacons from the MN after around the 25-s mark and assume that no GTS is available due to the manipulated beacons.

By impersonating the PC, we produce the same blocking of data transmission as that shown in Fig. 17(a), (b), and (c). However, after BCN-Sec is enabled at around the 40-s mark, LN2 and LN6 begin to synchronize with the PC, not with the MN. Then, their data throughput increases since the PC allocates the GTSs for LN2 and LN6 properly.

8.3 False data injection

In this attack, the MN impersonates LN2 to send false data in GTSs. Figure 18 shows the changes of humidity and temperature values from LN2. We tested this attack inside a building, where the humidity and temperature conditions were approximately 48 % and 83 °F respectively.

However, since the MN sends false data readings of 90 % for the humidity and 28 °F for the temperature during the CFP, this results in fluctuations of the sensed data reported for 10 s around the 23–31-s mark. Since 28 °F is below the freezing point, the false temperature data might lead to a warning sign in a critical situation.

After BCN-Sec is enabled, the false data readings are ignored from the PC because data from the MN are not authenticated.

8.4 DoS of GTS requests

In this attack, the MN sends GTS allocation requests with 7 different IDs to take all 7 GTSs. Figure 19(a) shows the normal data throughput patterns with the first two GTS allocation requests from LN2 until the 80-s mark. the MN starts occupying GTSs from the 80-s mark by sending GTS allocation requests and takes 6 GTSs; all GTSs except the one GTS already assigned to LN2. LN2 continues to send data using its one assigned GTS until around the 110-s mark.

Around the 115-s, LN2 sends a GTS deallocation request and the MN sends one GTS allocation request immediately and quickly occupies the remaining one GTS. This causes the halt of data transmission from the 120-s mark to the 150-s mark.

After BCN-Sec is (enabled and all GTSs are deallocated), LN2's data throughput returns to normal with the GTS allocation requests while the MN continuously sends GTS allocation requests.

8.5 Stealing network bandwidth

In this attack, the MN takes all 7 GTSs and transmits data to make the bandwidth unavailable to LN2. Figure

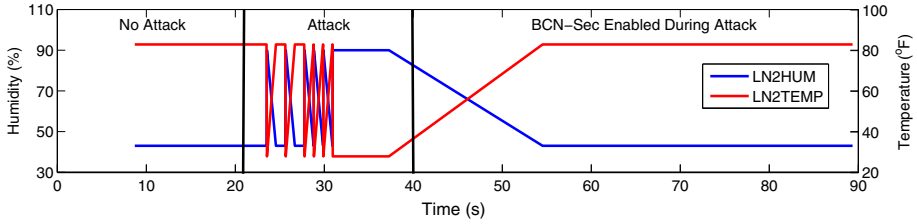


Fig. 18 Fluctuation of humidity and temperature and No fluctuation after BCN-Sec enabled

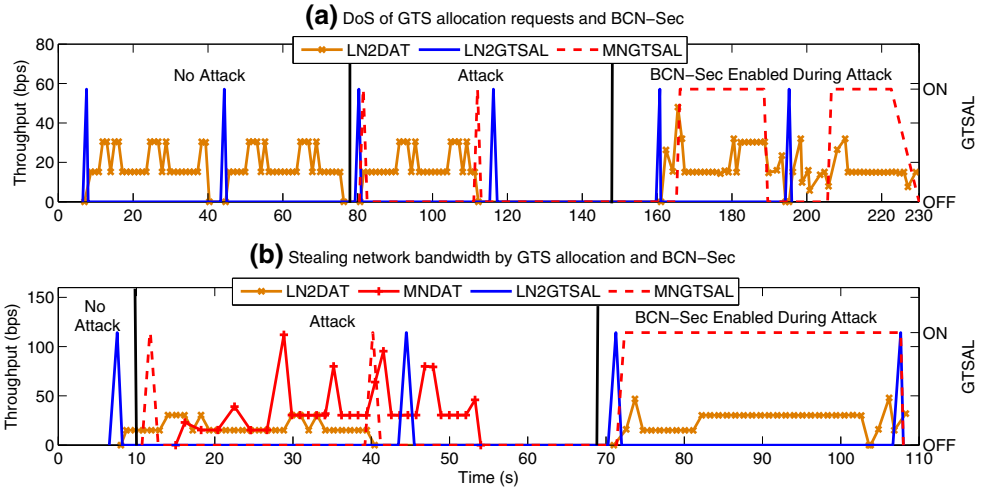


Fig. 19 MN filling up all 7 GTSs. LN2 DAT: LN2 Data, LN2 GTS AL: LN2 GTS allocation request, LN2 GTS DE: LN2 GTS deallocation request, and MN GTS AL: GTS allocation requests from MN. An MN (MN) stealing GTSs during CFP. LN2 DAT and MN DAT: Data from LN2 and MN respectively and MN GTS: GTS allocation requests from MN

19(b) shows the data throughput from LN2 and the MN with the GTS allocation requests from both. While LN2 has one GTS and transmits data during the CFP, the MN starts sending GTS allocation requests with 7 forged IDs around the 10-s mark and transmits data at the assigned GTSs. One of 7 GTS allocation requests of the MN is discarded at the first attempt because one GTS is already assigned to LN2.

However, as soon as LN2 releases its GTS around the 40-s mark, the MN immediately occupies the last GTS (and has all 7 GTSs). This is why only the MN is able to transmit data from the 40-s mark to the 55-s mark. Although LN2 sends one GTS allocation request around the 45-s mark, it is ignored by the PC because all 7 slots have been allocated to the MN.

After BCN-Sec is (enabled and all GTSs are deallocated) around the 70-s mark, even though the MN tries to get GTSs by generating multiple GTS allocation requests, they are ignored and there is only LN2's data throughput from the 75-s mark.

9 Conclusion and future work

In this paper, we first described the existing vulnerabilities of the beacon broadcast and the GTS management scheme in the 802.15.4 standard. We also investigated security protocols proposed in recent years and security mechanisms adopted in the standard. However, to date, no method comprehensively addresses the weakness of the beacon-enabled 802.15.4 MAC. To demonstrate the vulnerabilities in the 802.15.4 MAC, we implemented six attacks: (1) Synchronization attack, (2) DoS of data transmission by impersonating LNs, (3) DoS of data transmission by impersonating the PC, (4) False data injection, (5) DoS of GTS requests, and (6) Stealing network bandwidth. We also presented and implemented a countermeasure, BCN-Sec, and demonstrated how it defended against the attacks. We analyzed the results for each attack and their defense with BCN-Sec. Future work will provide a dynamic-key disclosure for newly generated key chains and a detailed energy measurement of our attacks and BCN-Sec.

Acknowledgments This work was partly supported by NSF Grant No. CAREER-CNS-844144.

References

1. Alim, M. A.,& Sarikaya, B. (2008). EAP-Sens: A security architecture for wireless sensor networks. In *Proceedings of the 4th annual international conference on wireless internet (WICON '08)*. ICST, ICST, Brussels, Belgium.
2. Amini, F., Misic, J., & Pourreza, H. (2008). Detection of sybil attack in beacon enabled IEEE 802.15.4 networks. In *Proceedings of the international wireless communications and mobile computing conference*.

3. Anisi, M., Abdullah, A., & Razak, S. (2013). Energy-efficient and reliable data delivery in wireless sensor networks. *Wireless Networks*, *19*(4), 495–505.
4. Aboba, B., Blunk, L., Vollbrecht, J., Carlson, J., & Levkowetz, H. (2004). *Extensible authentication protocol EAP*. http://tools.ietf.org/html/rfc3748
5. Chen, F., Talanis, T., German, R., & Dressler, F. (2009). Real-time enabled IEEE 802.15.4 sensor networks in industrial automation. In *IEEE international symposium on industrial embedded systems, 2009.SIES '09* (pp. 136–139).
6. Clancy, T., & Tschofenig, H. (2009). *Extensible authentication protocol: Generalized pre-shared key EAP-GPSK method*. http://tools.ietf.org/html/rfc5433
7. Demirbas, M., & Song, Y. (2006). An RSSI-based scheme for sybil attack detection in wireless sensor networks. In *Proceedings of the international symposium on world of wireless mobile and multimedia networks*.
8. Deutsch, L. P. (2011). http://sourceforge.net/projects/libmd5-rfc/files/latest/download?source=files
9. Douceur, J. R. (2002). The sybil attack. In *Proceedings of the 1st international workshop on peer-to-peer systems (IPTPS'02)*. New York, NY: IPTPS.
10. Du, W., Deng, J., Han, Y., Chen, S., & Varshney, P. (2004). A key management scheme for wireless sensor networks using deployment knowledge. In *Proceedings of the 23rd annual joint conference of the IEEE computer and communications societies (INFOCOM 2004)*.
11. Du, W., Deng, J., Han, Y. S., & Varshney, P. K. (2003) A pairwise key pre-distribution scheme for wireless sensor networks. In *Proceedings of the 10th ACM conference on computer and communications security*. New York, NY: ACM.
12. Eschenauer, L., & Gligor, V. D. (2002). A key-management scheme for distributed sensor networks. In *Proceedings of the 9th ACM conference on computer and communications security, CCS '02* (pp. 41–47). New York, NY: ACM.
13. Hayajneh, T., Almashaqbeh, G., Ullah, S., & Vasilakos, A. (2014). A survey of wireless technologies coexistence in WBAN: Analysis and open research issues. *Wireless Networks*, 1–35.
14. Herbert, H., & Frankel, S. (2003). The AES-XCBC-MAC-96 algorithm and its use with IPsec. http://www.faqs.org/rfcs/rfc3566.html
15. Hu, Y. C., Jakobsson, M., & Perrig, A. (2005). Efficient constructions for one-way hash chains. In*Proceedings of the 3rd international conference on applied cryptography and network security, ACNS'05* (pp. 423–441). Berlin: Springer.
16. IEEE P802.15 Working Group. (2006). *Wireless medium access control and physical layer specications for low-rate wireless personal area networks*. IEEE Standard, 802.15.4-2006. ISBN 0-7381-4997-7.
17. Jung, S. S., Valero, M., Bourgeois, A., & Beyah, R. (2010). Attacking beacon-enabled 802.15.4 networks. In *SecureComm '10: Proceedings of the 6th international ICST conference on security and privacy in communication networks*. Berlin: Springer.
18. Jung, S. S., Valero, M., Bourgeois, A., & Beyah, R. (2012). *Attacking and securing beacon-enabled 802.15.4 networks*. Technical Report GT-ECE-CAP-12-02, Georgia Institute of Technology.
19. Karlof, C., Sastry, N., & Wagner, D. (2004). TinySec: A link layer security architecture for wireless sensor networks. In *Proceedings of the 2nd international conference on embedded networked sensor systems (SenSys '04)*. New York, NY: ACM.
20. Karlof, C., & Wagner, D. (2003). Secure routing in wireless sensor networks: Attacks and countermeasures. In *Proceedings of the 1st IEEE. 2003 IEEE international workshop on sensor network protocols and applications*.
21. Koubaa, A., Alves, M., & Tovar, E. (2006). GTS allocation analysis in IEEE 802.15.4 for real-time wireless sensor networks. In *Parallel and distributed processing symposium, 2006. IPDPS 2006. 20th international* (p. 8).
22. Koubaa, A., Alves, M., & Tovar, E. (2006). i-GAME: An implicit GTS allocation mechanism in IEEE 802.15.4 for time-sensitive wireless sensor networks. In *18th Euromicro conference on real-time systems, 2006* (p. 10), pp. 192.
23. Lamport, L. (1981). Password authentication with insecure communication. *Communications of the ACM*, *24*, 770–772.
24. Liu, D., & Ning, P. (2003). Establishing pairwise keys in distributed sensor networks. In *Proceedings of the 10th ACM conference on computer and communications security*. New York, NY: ACM.
25. Lorincz, K., Malan, D. J., Fulford-Jones, T. R. F., Nawoj, A., Clavel, A., Shnayder, V., et al. (2004). Sensor networks for emergency response: Challenges and opportunities. *IEEE Pervasive Computing*, *3*(4), 16–23.
26. Luk, M., Mezzour, G., Perrig, A., & Gligor, V. (2007). MiniSec: A secure sensor network communication architecture. In *Proceedings of the 6th international conference on information processing in sensor networks (IPSN '07)*. New York, NY: ACM.
27. Mehta, A., Bhatti, G., Sahinoglu, Z., Viswanathan, R., & Zhang, J. (2009). Performance analysis of beacon-enabled IEEE 802.15.4 MAC for emergency response applications. In *2009 IEEE 3rd international symposium on advanced networks and telecommunication systems (ANTS)*, (pp. 1–3).
28. Mishra, A., Na, C., & Rosenburgh, D. (2007). On scheduling guaranteed time slots for time sensitive transactions in IEEE 802.15.4 networks. In *Proceedings of military communications conference, 2007. MILCOM 2007*. IEEE.
29. Moteiv.com: Tmote-Sky-datasheet. (2006). http://www.moteiv.com
30. Open-zb.net. (2011). http://www.open-zb.net/
31. Park, P., Fischione, C., & Johansson, K. (2009). Performance analysis of GTS allocation in beacon enabled IEEE 802.15.4. In *6th Annual IEEE communications society conference on sensor, mesh and ad hoc communications and networks, 2009, SECON '09* (pp. 1–9).
32. Perrig, A., Szewczyk, R., Wen, V., Culler, D., & Tygar, J. D. (2001) SPINS: Security protocols for sensor networks. In *Proceedings of the 7th annual international conference on Mobile computing and networking (MobiCom '01)*. New York, NY: ACM.
33. Roosta, T., Shieh, S., & Sastry, S. (2006). Taxonomy of security attacks in sensor networks and countermeasures. In *Proceedings of the 1st IEEE international conference on system integration and reliability improvements*.
34. Sastry, N., & Wagner, D. (2004). Security considerations for IEEE 802.15.4 networks. In *Proceedings of the 3rd ACM workshop on wireless security (WiSe '04)*. New York, NY: ACM.
35. Shen, W., Zhang, T., Gidlund, M., & Dobslaw, F. (2013). SAS-TDMA: A source aware scheduling algorithm for real-time communication in industrial wireless sensor networks. *Wireless Networks*, *19*(6), 1155–1170.
36. Shimmer Research.com. (2011). Shimmer research homepage. http://www.shimmer-research.com/
37. Sokullu, R., Dagdeviren, O., & Korkmaz, I. (2008). On the IEEE 802.15.4 MAC layer attacks: GTS attack. In *Proceedings of 2nd international conference on sensor technologies and applications (SENSORCOMM '08)*.
38. Texas Instruments Inc. (2011). *2.4 GHz IEEE 802.15.4 / ZigBee-ready RF Transceiver chipcon Products from Texas Instruments*. http://focus.ti.com/docs/toolsw/folders/print/packet-sniffer.html
39. Texas Instruments Inc. (2011). *SmartRF Packet Sniffer User Manual Rev. 1.9*. http://focus.ti.com/docs/toolsw/folders/print/packet-sniffer.html

40. Texas Instruments Inc. *User Manual Rev. 1.0 CC2420DK Development Kit*. http://focus.ti.com/lit/ug/swru045/swru045
41. TinyOS.net. (2011). http://www.tinyos.net/
42. Yang, J., Chen, Y., & Trappe, W. (2008). Detecting sybil attacks in wireless and sensor networks using cluster analysis. In *Proceedings of the 5th IEEE international conference on mobile ad hoc and sensor systems*.
43. Zhang, Q., Wang, P., Reeves, D., & Ning, P. (2005) . Defending against sybil attacks in sensor networks. In *Proceedings of the 25th IEEE international conference on distributed computing systems workshops*.
44. Zhu, S., Setia, S., & Jajodia, S. (2003). LEAP: Efficient security mechanisms for large-scale distributed sensor networks. In *CCS '03: Proceedings of the 10th ACM conference on computer and communications security*, (pp. 62–72). New York, NY: ACM.

Sang Shin Jung is a Ph.D. student in the School of Electrical and Computer Engineering at Georgia Institute of Technology. He is a member of the CAP Research Group. He received his B.S in Computer Science from Daejeon University in South Korea in 2002 and received his M.S. in Computer Science from Georgia State University in 2011. He also spent 5 years working as a software engineer in embedded system for Samsung Electronics during 2002–2007. Currently, He has been working on embedded systems security. He is a member of ACM and IEEE.

Marco Valero a native of Venezuela, is a Senior R&D Architect at Internap Network Services Corporation. He received his PhD. in Computer Science from Georgia State University in 2012, and has been working with the Communications Assurance and Performance Research Group (CAP) since the Fall of 2007. Marco also has a Masters Degree in Computer Science from Georgia State (2009), and Bachelor of Science in Computer Science from the Universidad Nacional Experimental del Tachira (2005). His research interests include network traffic optimization, Internet traffic analysis and characterization, MAC protocols optimization, and network security.

Anu G. Bourgeois received her B.S. and Ph.D. in electrical and computer engineering from Louisiana State University in 1991 and 2000, respectively. During 1996–2000 she held the Board of Regents Fellowship at LSU, and the Vincent A. Forte Fellowship during 1998–1999. Currently, she is an Associate Professor in the Department of Computer Science at Georgia State University. Dr. Bourgeois' research interests include parallel and distributed computing, reconfigurable networks, wireless networks, fault tolerant computing, and STEM education. She has served as a reviewer for several journals and as a program committee member for numerous workshops and conferences. She is a senior member of the IEEE.

Raheem Beyah is an Associate Professor in the School of Electrical and Computer Engineering at Georgia Tech where he leads the Communications Assurance and Performance Group (CAP) and is a member of the Communications Systems Center (CSC). He received his Bachelor of Science in Electrical Engineering from North Carolina A&T State University in 1998. He received his Masters and Ph.D. in Electrical and Computer Engineering from Georgia Tech in 1999 and 2003, respectively. Dr. Beyah has served as a Guest Editor for MONET and is currently an Associate Editor of the (Wiley) Wireless Communications and Mobile Computing Journal. His research interests include network security, wireless networks, network traffic characterization and performance, and critical infrastructure security. He received the National Science Foundation CAREER award in 2009 and was selected for DARPA's Computer Science Study Panel in 2010. He is a member of AAAS, ASEE, a lifetime member of NSBE, and a senior member of ACM and IEEE.