Security Threats in ZigBee-Enabled Systems: Vulnerability Evaluation, Practical Experiments, Countermeasures, and Lessons Learned

Niko Vidgren[1], Keijo Haataja[1], José Luis Patiño-Andres[2], Juan José Ramírez-Sanchis[2], Pekka Toivanen[1]
[1]University of Eastern Finland, School of Computing, Kuopio Campus
P.O. Box 1627, FI-70211 Kuopio, Finland
[2]Escola Tècnica Superior d'Enginyeria, University of Valencia, Spain
E-mail: nikovidgren@gmail.com, keijo.haataja@uef.fi, {jopan, juanra4}@alumni.uv.es, pekka.toivanen@uef.fi

Abstract

In this paper, two practical attacks against ZigBee security are proposed and the latter one is also carried out in our laboratory environment. The attack scenarios are based on utilizing several vulnerabilities found from the main security components of ZigBee technology. The first attack is based on sabotaging the ZigBee End-Device by sending a special signal that makes it wake-up constantly until the battery runs out. The second attack is based on exploiting the key exchange process in ZigBee when using the Standard Security level defined by the ZigBee specification: we also demonstrate with experimental figures that attacks against ZigBee-enabled devices become practical by using our attack scenario. In addition, countermeasures that render the proposed attacks impractical, although not totally eliminating their potential danger, are devised. Moreover, some new ideas that will be used in our future research work are proposed.

I. INTRODUCTION

In recent years, the use of wireless communication systems, especially Wireless Personal Area Networks (WPANs), and their interconnections via networks have grown rapidly. ZigBee is a relatively new emerging communication technology capable of forming WPANs between different kinds of wireless ad-hoc devices, such as laptops, PCs, mice, keyboards, printers, headsets, mobile phones, multimedia devices, and toys.

Since radio frequency (RF) waves can penetrate obstacles, wireless devices can communicate with no direct line-of-sight between them. Thus, RF communication is more flexible and easier to use than wired or infrared communication, but it also makes eavesdropping easier. Moreover, it is easier to disrupt and jam wireless RF communication than wired communication. Since wireless RF communication can suffer from these threats, additional countermeasures are needed to protect against them.

ZigBee [1] is a technology for short range wireless data transfer especially designed for wireless low-power devices providing battery lifetimes up-to several years with a single AA battery. ZigBee is based on IEEE 802.15.4-2003 standard [2]: it adds the network and application layers on top of the Physical layer (PHY) and Medium Access Control layer (MAC), which are defined by IEEE 802.15.4-2003. ZigBee also provides enhanced security control and support for mesh networks. [1][2]

The first version of ZigBee specification, simply called as *ZigBee* [1] (also referred to as *ZigBee 2004*), was released in 2004: nowadays this version can be considered obsolete and thus it is not supported anymore in new ZigBee devices. The second version of ZigBee specification, also called as *ZigBee* [1] (also referred to as *ZigBee 2006*), was released in 2006: this version is used when the ZigBee network should be as cheap as possible. The latest version of ZigBee specification, *ZigBee Pro* [1] (also referred to as *ZigBee 2007*), was released in 2007: this version is used when the size of the ZigBee network is very large and enhanced security features are needed to protect the network. Even though the IEEE 802.15.4-2003 [2] was updated to IEEE 802.15.4-2006 [3] in 2006, the current versions of ZigBee (i.e. ZigBee and ZigBee Pro) still use IEEE 802.15.4-2003 as their basis. [1][2][3]

ZigBee's main scope and purpose is ”home automation” including sensors and actuators, such as rain/light/smoke sensors, locks, and windows. ZigBee operates at 2.4 GHz frequency supporting theoretical data rate of 250 kb/s, at 915 MHz frequency supporting theoretical data rates of 40 kb/s and 250 kb/s, and at 868 MHz frequency supporting theoretical data rates of 20 kb/s, 100 kb/s, and 250 kb/s. ZigBee is based on Direct Sequence Spread Spectrum (DSSS) technique and Offset Quadrature Phase Shift Keying (O-QPSK) modulation. [1]

The results of this paper: In this paper, we propose two practical attacks against ZigBee security and the latter one is also carried out in our laboratory environment. The first attack is based on sabotaging the ZigBee End-Device (ZED) by sending a special signal that makes it wake-up constantly until the battery runs out. The second attack is based on exploiting the key exchange process in ZigBee when using the Standard Security level: we also demonstrate with experimental figures that attacks against ZigBee-enabled devices become practical by using our attack scenario. In addition, countermeasures that render the proposed attacks impractical, although not totally eliminating their potential danger, are devised. Moreover, some new ideas that will be used in our future research work are proposed.

The rest of the paper is organized as follows. Section

1530-1605/12 $26.00 © 2012 IEEE
DOI 10.1109/HICSS.2013.475

II provides an overview of ZigBee security and describes the existing ZigBee security attacks. Our practical attacks against ZigBee security are proposed and the latter one is also demonstrated with experimental figures in Section III. Section IV devises countermeasures for these attacks. Finally, Section V concludes the paper and sketches future work.

II. ZigBee Security Basics and Existing Attacks

The main four concepts of ZigBee security are: [1][4][5]

1) *Security Level:* ZigBee supports two different security levels: *High Security* (also referred to as *Commercial Security*) and *Standard Security* (also referred to as *Residential Security*). Differences between these security levels are mainly in the key management and distribution.
2) *Trust Center (TC):* One of the devices in a ZigBee-enabled network, the TC, is responsible for the security management. The TC provides a safety mechanism using three types of security keys: the *network key*, the *master key*, and the *link key*. Moreover, the TC is responsible for selecting the suitable security level and for the key management. All ZigBee devices share the common network key, while the link key can be shared by any two ZigBee devices. The link key is derived from the master key, which is the basis for long-term security between two ZigBee devices.
3) *Authentication and Data Encryption:* Data is encrypted using 128-bit Advanced Encryption Standard (AES) with CCM (CCM = CBC-MAC = Counter with Cipher Block Chaining Message Authentication Code) mode allowing authentication and data encryption, thus forming a Federal Information Processing Standards (FIPS) compliant security mode called *AES-CCM*. The CCM mode is a mode of operation only for 128-bit cryptographic block ciphers. It combines the counter mode with the CBC-MAC authentication and uses the same encryption key for both modes. ZigBee uses a slightly modified version of CCM called CCM*, which gives more flexibility than the standard CCM: CCM* enables to use either authentication or encryption, while both are always required in CCM.
4) *Integrity and Freshness of Data:* Several different security keys and methods are used to ensure the integrity and freshness of data. The *Message Integrity Code (MIC)* can be used to make sure that the data has not been altered in transit (see Figure 1). ZigBee supports 16-, 32-, 64-, and 128-bit MIC lengths. The MIC is generated using the CCM* protocol.

Key management is an important part of any security specification. The security keys can be distributed in ZigBee-enabled networks either as over-the-air transmits or by pre-installing them onto the devices. Depending on the security level, there are differences in the key distribution:

- The network key is always transmitted encrypted over-the-air when using the High Security level, because its distribution is secured using the master key and thus the communicating devices can establish a trusted relationship between them.
- The network key is transmitted unencrypted over-the-air when using the Standard Security level and thus this is a serious vulnerability for the security of the ZigBee-enabled networks leading to the conclusion that the Standard Security level cannot be recommended for safety critical systems. However, it is possible to manually pre-install network keys onto each legitimate device of the ZigBee-enabled network, but this is clearly a trade-off between usability and security, at least when the size of the network is large: therefore, the network administrator is likely to opt for less secure but more usable options.

Replay attacks [6][7] can be prevented in ZigBee by using a 32-bit frame counter. The frame counter is a value that is constantly updated by the communicating devices every time a new frame has been received: thus, the frame will be discarded if a frame counter value is lower than the current value in the memory of the recipient. The frame counter is reset back to zero every time the network key is updated. In theory, this is an efficient way to prevent Replay attacks, but in practice there is a problem arising from the fact that ZigBee specification [1] does not define when the TC has to update the network key: thus, it is left to the discretion of the ZigBee-enabled network administrator to define it. If the network key is not updated in time, it will lead to a deadlock in the network as soon as the frame counter reaches its maximum value. [1][6][7]

There are some security threats against ZigBee-enabled systems, such as traffic sniffing (eavesdropping), packet decoding, and data manipulation/injection, which can be exploited by an attacker who uses a special hardware and software developed especially for attacking purposes. The attacker can use a cheap $40 device, the *AVR RZ Raven USB (Universal Serial Bus) Stick (RZUSB)* [8], for performing various attacks. The RZUSB consists of the AT90USB1287 microcontroller with integrated Full Speed USB interface, the AT86RF230 802.15.4 transceiver with miniature Printed Circuit Board (PCB) antenna, four Light-Emitting Diodes (LEDs), and an external memory interface. Moreover, an Integrated Development Environment (IDE) based on *GNU Compiler Collection* (GCC) is freely available for software development. [8]

The default firmware of RZUSB can be directly used for sniffing ZigBee traffic and acting as a ZigBee Personal Area Network (PAN) Coordinator or a ZED. If the attacker uses two RZUSB devices simultaneously, sniffing and packet modification/injection can be performed at the same time. This requires that the second RZUSB device uses a modified firmware called *KillerBee* that supports packet modification and injection. The *KillerBee software suite* [8] consists of this modified version of RZUSB firmware as well as some tools for performing attacks against ZigBee security. It is a freely available software suite developed using Python programming language [8]. Moreover, since KillerBee is an open framework, released under the Berkeley Software Distribution (BSD) license, it is expected that it will be expanded in the near future and thus ZigBee sniffing and hacking will soon become more effective and reveal many previously unknown vulnerabilities.

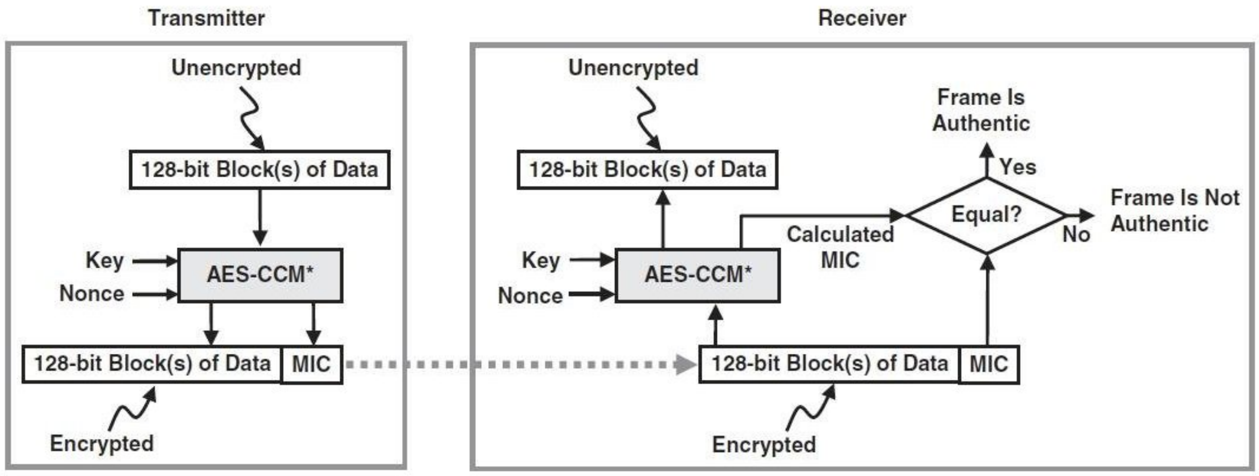


Fig. 1: Ensuring the integrity and freshness of data using the MIC. [5]

ZEDs, such as sensors and actuators, often run on batteries and have a very low duty cycle (i.e. ratio of active radio time compared to the silent period). Such networks have a predefined wake-up intervals for saving battery life, but it also opens new doors for practical Denial-of-Service (DoS) attacks. In a DoS attack, an attacker repeatedly jams the medium during both the Contention Access Period (CAP) and the Contention Free Period (CFP). In this way, a victim device can be put on endless retransmission loop, which ultimately leads to a battery exhaustion of a victim device or at least greatly reduces its battery life. [9][10]

A ZigBee network is vulnerable to the DoS attack if the message integrity is not verified, even if the communicated messages are encrypted. The DoS attack works in the following way: [5]

1) The attacker composes a message that includes a random content as the encrypted payload (without knowing the security key) and sets the frame counter to the maximum value. It is worth noting that the frame counter is used to set a *high-water mark* for the frame counter associated with the authorized originator of the frame.
2) The attacker sends the message to the victim device.
3) The victim device decrypts the payload to a random plaintext, which have no meaning to the next upper protocol layer.
4) Since the attacker has already set the *high-water mark* of the frame counter to the maximum value, any legitimate frame that arrives after the DoS attack will automatically be rejected by the victim device, because the frame counter value of the received message will be less than the *high-water mark* established during the DoS attack. [5][11]

ZigBee specification allows to apply security (authentication and encryption) for all ZigBee frames including the use of MIC. In this way, the DoS attack can be prevented. In addition, ZEDs can operate in non-beacon mode where the ZED actively polls the network coordinator to check if there is data available. ZEDs can still go to ”deep sleep” during inactivity periods and wake-up only occasionally to transfer data, thus saving the battery life. This wake-up action should occur at irregular intervals, because then it is more difficult for attackers to guess the exact time when data transfers take place. Moreover, the feature *wake-on-radio*, available from certain chip vendors, should be used when possible, since it makes more difficult for an attacker to guess the activity period of the network.

When a *nonce (number used once)* is used as a part of the AES-CCM*, encrypting the same plaintext twice will result in two different ciphertexts, because the nonce will be different even if the same security key is reused. This property is known as *semantic security*.

The attack that works against both ZigBee and Ultra-Wideband (UWB) security is called the *Same-Nonce attack*. If, for any reason, the Access Control List (ACL) provides the same nonce and the same security key for two consecutive messages, an eavesdropper can recover partial information regarding the plaintexts. In the attack, the eavesdropper makes sure that the nonce and the security key used for generating two ciphertexts are the same. After that, the eavesdropper knows that the XOR of these two ciphertexts will be the same as the XOR of their corresponding plaintexts. [5]

The Same-Nonce attack can be successfully performed, for example, by causing a power failure that results in a clear ACL. If the last nonce states are unknown after the power failure, the system resets the nonce states to the default value. This action increases the chance of reusing the same nonce with a security key that has been used before the power failure. Thus, the system becomes vulnerable to the Same-Nonce attack. [5][12]

Since the system becomes vulnerable to the Same-Nonce attack after a power failure that results in a clear ACL, a simple and practical countermeasure is to store the nonce states in a Non-Volatile Memory (NVM) and recover them after each power failure. This is exactly what the ZigBee specification proposes and thus the attack can be easily prevented in ZigBee-enabled systems. [1]

All threats against ZigBee security are not only focused on just interfering, sabotaging, or manipulating the data itself: indeed, a physical attack is possible against ZigBee-enabled networks and has to be taken into account when forming and managing a network. Since ZigBee is often used in increasingly important applications, such as controlling the infrastructure of critical systems in a commercial building, industrial plant, utility grid, or a home security system, it is very important to design the ZigBee-enabled network in such a way that the devices are also protected from physical attacks. This means, for example, placing the devices to such locations that are hard to reach and protecting them with some kind of surveillance (e.g. by using an intrusion detection and prevention system).

If an attacker is capable of stealing a ZigBee device, it is possible to extract its data and even the stored security keys: this has already been demonstrated by Travis Goodspeed [13] in his GoodFET project [14]. However, the attack works only with ZigBee chips of certain vendors. It is important that either a some kind of automatic system exists for noticing missing devices or that periodical checks are being made. In a case of a missing device, the security keys must be updated immediately to prevent unauthorized use of the whole ZigBee-enabled network.

III. Novel Attacks

This section proposes two practical attacks against ZigBee security: a *ZigBee End-Device Sabotage attack* (see Section III-A) and a *ZigBee Network Key Sniffing attack* (see Section III-B). Both attacks are based on utilizing several vulnerabilities found from the main security components of ZigBee technology. The second attack is also carried out in our laboratory environment to demonstrate with experimental figures that attacks against ZigBee-enabled devices become practical by using our attack scenario.

A. ZigBee End-Device Sabotage Attack

ZigBee uses different kinds of sensors/actuators, such as locks and light sensors, as ZEDs. Such devices are not connected to the electrical network, since they spend most of their time in a *sleep mode*, which greatly reduces the power consumption.

The maximum power consumption during the *active mode* and the amount of power leakage during the sleep mode are good examples of hardware-level performances that directly affect on battery life. It is not only a function of hardware-level performance of each node, since it also depends on the operation efficiency of the network. For example, the number of operations a device needs to perform when establishing communication links with other nodes in the network can be reduced by simplifying the networking protocol itself. The selection of the routing mechanism and link cost calculation method can greatly impact the operation efficiency of a network.

The ZigBee specification allows the network installers to select any link cost calculation method they find most suitable for their applications. When a device tries to transmit data to another device, the total energy required for successful transmission of data must be taken into account instead of instantaneous power consumption. The *goodput (application level throughput)* is defined as the number of useful bits transmitted per second, excluding the packet overhead and retransmissions.

Depending on the application scenario, a ZED can spend most of its time in sleep mode or it can wake-up frequently when needed. The power consumption during the warm-up period can be low, but if the circuit start-up time is very long, the energy consumption during the wake-up period can noticeably decrease the battery life of a device. Figure 2 illustrates an average power consumption during the wake-up part of the duty cycle when 24 bytes of data is received.

Battery lifetimes up to several years are possible in ZigBee-enabled systems due to power-saving modes and battery-optimized network parameters, such as a selection of beacon intervals, guaranteed time slots, and various enable/disable options. Let us consider a typical ZigBee security application, a magnetic reed switch door sensor: the sensor itself consumes almost no power at all. The sensor is configured to have a ”heartbeat” at one-minute intervals for checking whether an event has occurred or not: in case of an event, the sensor immediately sends a message. If we assume dozens of events per day, the sensor can still function the whole shelf life of an alkaline AAA battery. Thus, such a configuration easily allows various remote tasks, such as sensor parameter updates and reporting interval changes, while still maintaining the whole shelf life of a battery.

ZEDs rely on a parent (a router or a coordinator) to remain awake and receive data packets. When a ZED wakes-up from a sleep mode, it sends a message (poll request) to its parent asking if there is any data available: upon receipt of the poll request, the parent sends a response to the poll request as well as the buffered data (if any) of the corresponding ZED. In case there is no buffered data, the ZED can return to a sleep mode.

We call our first attack as *ZigBee End-Device Sabotage attack*. Our attack works in the following way:

1) An attacker impersonates the ZigBee Router (ZR) or the ZigBee Coordinator (ZC) of the ZigBee-enabled system in order to abuse the poll requests of legitimate ZEDs.
2) Since the default polling rate of ZEDs is 100 ms, they will send poll requests every 100 ms to their parent (the attacking device). The attacking device sends broadcast or multicast replies to all poll requests of legitimate ZEDs, thus keeping them awake all the time.
3) Since ZEDs remain awake all the time, they will continue to send poll requests every 100 ms to the attacking device for checking the buffered data.
4) Based on the default polling rate (100 ms) and the power leakage during each wake-up (see Figure 2), the attacker can cause power failures to ZigBee sensors/actuators and thus use our attack for unauthorized openings of, for example, ZigBee-enabled house door locks, windows, car door locks, and motion sensors.

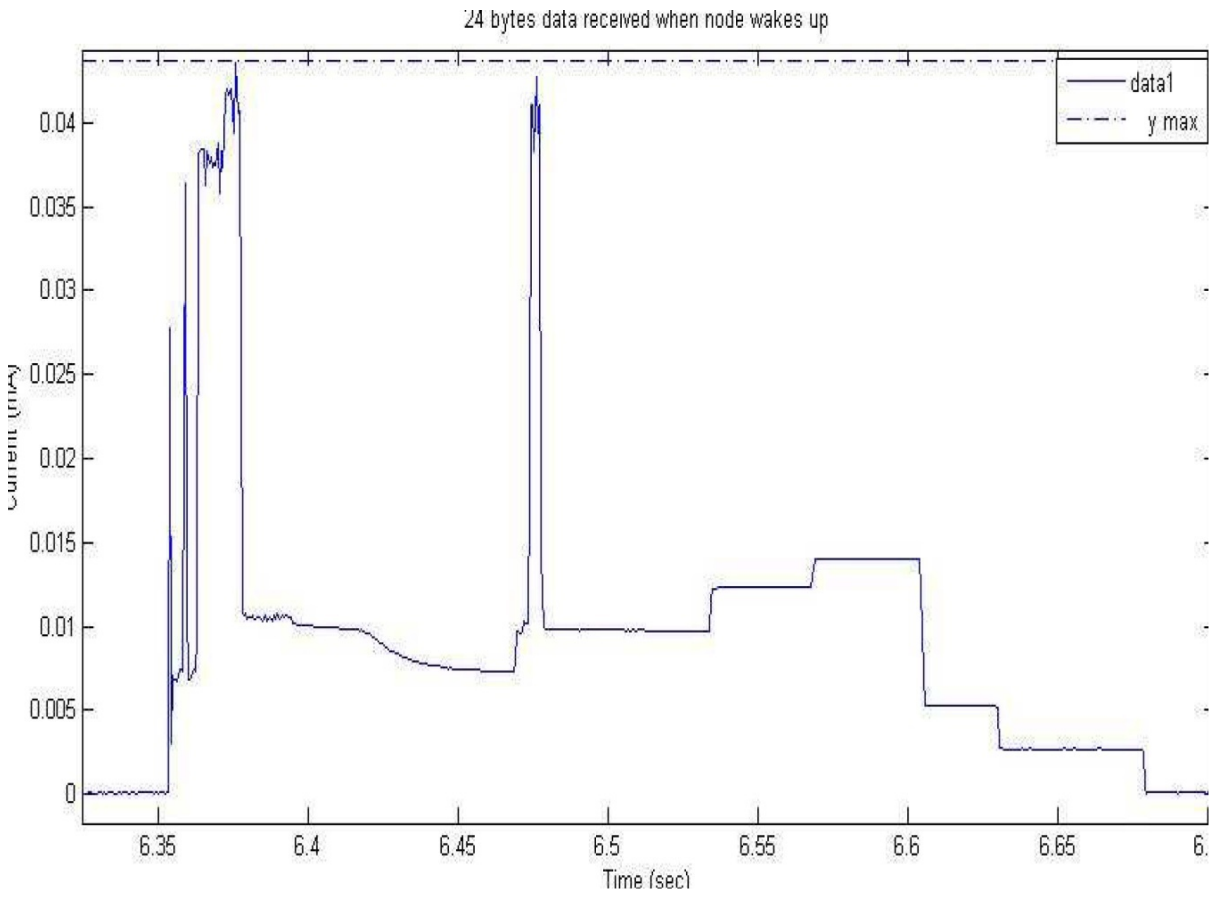


Fig. 2: An average power consumption during the wake-up part of the duty cycle when 24 bytes of data is received. [15]

B. ZigBee Network Key Sniffing Attack

There is a serious vulnerability in the network key transport of ZigBee when using the Standard Security level: if the network key is not pre-installed to the legitimate devices of the ZigBee-enabled network that is using the Standard Security level, the TC sends the current network key unencrypted over-the-air to the devices when they want to join the network. Thus, an attacker can intercept the network key and use it for eavesdropping and attacking purposes against the compromised ZigBee-enabled network.

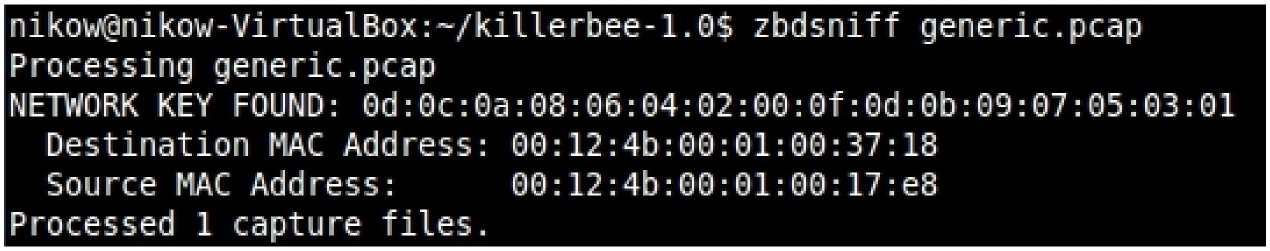
We call our second attack as *ZigBee Network Key Sniffing attack*. It abuses the abovementioned serious vulnerability in the network key transport of ZigBee when the target network is using the Standard Security level. The attack was carried out in our laboratory environment to demonstrate with experimental figures that attacks against ZigBee-enabled devices become practical by using our attack scenario. Moreover, we want to raise a discussion about this serious vulnerability by proposing that *the Standard Security level should be removed altogether from the ZigBee specification.*

Our attack can be conducted without any vast knowledge of the ZigBee technology using free and readily available software tools and cheap ZigBee hardware, thus making our attack very practical and dangerous. Our practical laboratory experiment was conducted using a small ZigBee-enabled network consisting of only two devices: the ZED and the ZC. The only purpose for these two devices was to communicate in the ZigBee-enabled network using the Standard Security level.

Our attack works in the following way:

1) The over-the-air traffic of the network was captured with Texas Instruments CC2531 USB dongle [16] using Texas Instruments SmartRF Protocol Packet Sniffer software [17] that is readily available from Texas Instruments' website [17].
2) The captured data was converted from Texas Instruments' PSD-format (Packet Sniffer Data) to PCAP-format (Packet Capture) so that KillerBee [8] can understand it.
3) The network key was parsed from the PCAP-file using KillerBee's zbdsniff tool [8]: it is basically a parser that searches for the Application Support Sublayer (APS) key transport command frames from the captured data, checks that the key type is 0x01 (i.e. the network key), and finds the network key.

Figure 3 illustrates the result of a successful attack: indeed, our practical experiment showed that the Standard Security level provides insufficient level of security and thus it should not be used in any safety-critical ZigBee-enabled systems.



```
nikow@nikow-VirtualBox:~/killerbee-1.0$ zbdsniff generic.pcap
Processing generic.pcap
NETWORK KEY FOUND: 0d:0c:0a:08:06:04:02:00:0f:0d:0b:09:07:05:03:01
  Destination MAC Address: 00:12:4b:00:01:00:37:18
  Source MAC Address:      00:12:4b:00:01:00:17:e8
Processed 1 capture files.
```

Fig. 3: The result of a successful attack.

IV. Countermeasures

ZigBee security specifications have gone through a series of reviews by experts, and the released versions generally do good work in improving the security of ZigBee-enabled systems. However, some attacks against ZigBee security are still possible as our paper clearly shows.

This section devises countermeasures that render the proposed attacks impractical although not totally eliminating their potential danger:

- *Against ZigBee End-Device Sabotage attack:* It is worth noting that ZigBee End-Device Sabotage attack will last only until the network administrator notices the error and replaces the batteries. A practical countermeasure is to use a remote alerting system for warning about power failures of ZigBee devices, but this approach requires active role of network administrator. Another practical countermeasure, which does not require continuous maintenance and actions of the network administrator, is to configure the legitimate ZEDs in a cyclic sleep mode that allows modules to wake-up periodically for checking data. The legitimate ZEDs can be configured to sleep for a predetermined time by using the *Sleep Period* parameter. After the predetermined time expires, the ZEDs will wake-up for checking data. If data is received, the module will start the *Time Before Sleep* timer and remain awake until the timer expires. In this way, the network is protected from our ZigBee End-Device Sabotage attack, because the ZEDs will resume sleep when the *Time Before Sleep* timer expires.
- *Against ZigBee Network Key Sniffing attack:* The most obvious countermeasure against this attack is to pre-install the network key when using the Standard Security level, but unfortunately it is a clear trade-off between usability and security at least when the size of the network is large. Thus, the better option is to use the High Security level in safety-critical ZigBee-enabled systems, because then the network key is never transported unencrypted over-the-air.

V. Conclusion and Future Work

Two practical attacks against ZigBee security were proposed and the latter one was also carried out in our laboratory environment. The attack scenarios are based on utilizing several vulnerabilities found from the main security components of ZigBee technology. The first attack is based on sabotaging the ZigBee End-Device by sending a special signal that makes it wake-up constantly until the battery runs out. The second attack is based on exploiting the key exchange process in ZigBee when using the Standard Security level defined by the ZigBee specification: we also demonstrated with experimental figures that attacks against ZigBee-enabled devices become practical by using our attack scenario. Moreover, countermeasures that render the proposed attacks impractical, although not totally eliminating their potential danger, were devised. The proposed countermeasures are very simple and economical to implement: these attributes will also appeal to the ZigBee device manufacturers.

It is difficult to create a protocol which caters to all possible types of wireless devices, as the security of the protocol is likely to be limited by the capabilities of the least powerful or the least secure device type. Most attacks against ZigBee security are based exactly on this problem.

The problems we want to investigate in our future research work are concerned with the following issues:

1) A physical attack must be taken seriously, because there exist practical experiments on how to extract security keys from the ZigBee devices [13][14]. Let us assume a real life example of a hotel using ZigBee for monitoring safe deposit boxes, improving energy efficiency, or opening hotel room doors [18][19][20]. Such a hotel can contain thousands of ZigBee devices and thus needs a good plan to manage them all. We want to investigate how this can be done in practice, especially in a situation where one or several of the ZigBee devices are stolen either by a hotel customer or an attacker. Moreover, we want to further investigate ZigBee chips of different vendors for comparing how the security keys are physically stored.
2) ZigBee is a relatively new wireless technology and thus new attacks against it are likely to be found. We want to further investigate ZigBee security weaknesses and propose countermeasures against discovered attacks.
3) Issues related to ZigBee user experience (ease of use) have become more and more important in recent years. Thus, we want to investigate how enhanced user experience will affect ZigBee security in various scenarios, including social aspects

and user acceptance/habits in security management. Moreover, we want to devise best practices depending on the risk analysis within each scenario.

4) Since the ZigBee specification leaves a lot of freedom to the implementer in many cases, we want to investigate and compare ZigBee protocol stacks of different vendors to see how some of the aspects have been implemented in practice. The network key update process is a good example of this implementer responsibility: it is not defined in the ZigBee specification and if a proper network key update protocol is missing from the implementation, it will lead to a deadlock as soon as the frame counter reaches its maximum value.
5) We want to extend our research work to cover also RF-fingerprinting techniques [21], [22], [23], [24], because we feel that the use of RF fingerprints could be the future of secure ZigBee communications.

Acknowledgements

This research work was funded by the European Union Artemis project Design, Monitoring, and Operation of Adaptive Networked Embedded Systems (DEMANES).

REFERENCES

[1] ZigBee Alliance, ZigBee Specifications: ZigBee and ZigBee Pro. [Online]. Available: http://www.zigbee.org. [Accessed Sep. 6, 2012].

[2] IEEE, IEEE 802.15.4-2003 Specification. [Online]. Available http://standards.ieee.org/getieee802/download/802.15.4-2003.pdf. [Accessed Sep. 6, 2012].

[3] IEEE, IEEE 802.15.4-2006 Specification. [Online]. Available http://standards.ieee.org/getieee802/download/802.15.4-2006.pdf. [Accessed Sep. 6, 2012].

[4] I. Pérez-González, My Bandwidth Is Wider Than Yours: Ultra-Wideband, Wireless USB and WiNET in Linux. [Online]. Available: http://ols.fedoraproject.org/OLS/Reprints-2007/perez-gonzalez-Reprint.pdf. [Accessed Sep. 6, 2012].

[5] S. Farahani, *ZigBee Wireless Networks and Transceivers*, Newnes, Elsevier, Burlington, USA, 2008.

[6] C. Alcaraz and J. Lopez, "A Security Analysis for Wireless Sensor Mesh Networks in Highly Critical Systems," *IEEE Transactions on Systems, Man, and Cybernetics, Part C – Applications and Reviews*, Vol. 40, No. 4, July 2010.

[7] C. Karlof and D. Wagner, "Secure Routing in Wireless Sensor Networks: Attacks and Countermeasures," *Elsevier Ad Hoc Networks*, Vol. 1, pp. 293–315, 2003.

[8] J. Wright, KillerBee: Practical ZigBee Exploitation Framework or "Wireless Hacking and the Kinetic World". [Online]. Available: http://www.willhackforsushi.com/presentations/toorcon11-wright.pdf. [Accessed Sep. 6, 2012].

[9] R. Rodrigues da Silva Severino, On the Use of IEEE 802.15.4/ZigBee for Time-Sensitive Wireless Sensor Network Applications. [Online]. Available: http://www.cooperating-objects.eu/fileadmin/dissemination/2009-thesis-award/severino.pdf. [Accessed Sep. 6, 2012].

[10] I. Ramachandran, A. Das, and S. Roy, Analysis of the Contention Access Period of IEEE 802.15.4 MAC. [Online]. Available: http://www.ee.washington.edu/research/funlab/Publications/2006/CAP_802_15_4_Analysis.pdf. [Accessed Sep. 6, 2012].

[11] N. Sastry and D. Wagner, "Security Considerations for IEEE 802.15.4 Networks," in *Proceedings of the 3rd ACM Workshop on Wireless Security*, Philadelphia, USA, 2004, pp. 32–42.

[12] R. Housley, RFC 5084 – Using AES-CCM and AES-GCM Authenticated Encryption in the Cryptographic Message Syntax (CMS). [Online]. Available: http://www.rfc-archive.org/getrfc.php?rfc=5084. [Accessed Sep. 6, 2012].

[13] T. Goodspeed, "Extracting Keys from Second Generation Zigbee Chips," *Black Hat USA 2009*, Las Vegas, USA, 2009. [Online]. Available http://www.blackhat.com/presentations/bh-usa-09/GOODSPEED/BHUSA09-Goodspeed-ZigbeeChips-PAPER.pdf. [Accessed Sep. 6, 2012].

[14] GoodFET Project. [Online]. Available: http://goodfet.sourceforge.net. [Accessed Sep. 6, 2012].

[15] C. Misal, *Analysis of Power Consuption of an End Device in a Zigbee Mesh Network*, Master Thesis, University of North Carolina, Charlotte, North Carolina, USA, 2007.

[16] Texas Instruments, CC2531 USB Hardware – User's Guide. [Online]. Available: http://www.ti.com/lit/ug/swru221a/swru221a.pdf. [Accessed Sep. 6, 2012].

[17] Texas Instruments, SmartRF Protocol Packet Sniffer. [Online]. Available: http://www.ti.com/tool/packet-sniffer. [Accessed Sep. 6, 2012].

[18] T. Buley, *My Man, ZigBee*. [Online]. Available: http://www.forbes.com/2009/01/11/zigbee-mirage-ces-tech-personal-cz_tb_0112zigbee.html. [Accessed Sep. 6, 2012].

[19] Telegesis Ltd., ZigBee Case Study, *Hotech Edge Ltd Safely Networks Safes with ZigBee!*. [Online]. Available: http://www.telegesis.com/downloads/general/Hotech%20Edge%20success%20story.pdf. [Accessed Sep. 6, 2012].

[20] Ember Corporation, ZigBee Slashes Hotel Energy Costs by up to 40 Percent. [Online]. Available: http://www.thefreelibrary.com/ZigBee+slashes+hotel+energy+costs+by+up+to+40+percent.-a0154056797. [Accessed Sep. 6, 2012].

[21] S. Pasanen, K. Haataja, N. Päivinen, and P. Toivanen, "New Efficient RF Fingerprint-Based Security Solution for Bluetooth Secure Simple Pairing," in *Proceedings of the IEEE 43rd Hawaii International Conference on System Sciences (HICSS)*, Koloa, Kauai, Hawaii, Jan. 5–8, 2010.

[22] K. Haataja, *Security Threats and Countermeasures in Bluetooth-Enabled Systems*, Ph.D. Diss., University of Kuopio, Department of Computer Science, Feb. 6, 2009.

[23] M. Barbeau, J. Hall, and E. Kranakis, "Detecting Impersonation Attacks in Future Wireless and Mobile Networks," *Lecture Notes in Computer Science*, vol. 4074, pp. 80–95, Springer-Verlag, 2006.

[24] O. Ureten and N. Serinken, "Wireless Security Through RF Fingerprinting," *Canadian Journal of Electrical and Computer Engineering*, vol. 32, no. 1, pp. 27–33, 2007.