Available online at www.sciencedirect.com

ScienceDirect

Procedia Computer Science 37 (2014) 376 – 381

Procedia Computer Science

The 6th International Symposium on Applications of Ad hoc and Sensor Networks (AASNET'14)

New security approach for ZigBee Weaknesses

Wissam Razouk[a,b,*], Garth V. Crosby[b], Abderrahim Sekkaki[a]

[a]*Hassan II University, Faculty of science, Dept of mathematics and computer science, 5366, Casablanca, Morrocco*
[b]*Southern Illinois University, Dept of technology, Engineering building, 62901, Illinois, USA*

Abstract

The latest version of ZigBee offers improvements over many aspects like low power consumption, flexibility and inexpensive deployment. However problems persist, as the enhanced protocol still has many security weaknesses due to the fact that constrained wireless sensor network devises cannot uses standard security protocols such as public key cryptography mechanisms. In this paper, we highlight relevant security concerns related to ZigBee security features, then we propose a new approach suitable for ZigBee enabled wireless sensor networks. The proposed solution decreases considerably the likelihood of successful attacks,and reduces security impact in the event of a node compromise. Lastly we discuss the security and performance related to the proposed scheme.

Keywords: Wireless sensor networks; ZigBee protocol; Security; Internet of Things.

1. INTRODUCTION

The Internet of things (IoT) is becoming quickly a reality and wireless sensor networks are going to be certainly widely deployed in the near future. Indeed, wireless sensor networks play an important role in IoT and are considered as an emerging technology with a wide range of applications in many areas. Their importance increases especially with the fast progress of their implementation. Therefore, security becomes a pressing need.

ZigBee is known to have many appealing advantages like low-cost, high reliability and low-complexity[6], and has a wide application range, whether in industrial automation, intelligent control, or medical health etc. But ZigBee still faces many challenges, such as computing restrictions of nodes, limited memory space, and constrained energy consumption and communication capability. These restrictions make it unpractical to apply classical security mechanisms like public key cryptography. That is why it is important to further study this topic, and focus more research activities on this specific area.

Our contribution: we summarize our contribution as follows:

[*] Corresponding author. Tel.: +212-677-051377.
E-mail address: wissam.razouk@etude.univcasa.ma

doi:10.1016/j.procs.2014.08.056

- In the original version of ZigBee, the protocol have nine exchange messages. This type of protocols require a number of operations that is linear with the number of nodes in the system, which might be impractical since wireless sensor networks can consist of a great number of nodes. Our approach needs only four message exchanges to fully complete the process of a secure communication.
- ZigBee protocol suffers from serious weaknesses related to key distribution, as keys are transmitted either over-the-air or preinstalled onto the devices in an insecure way. Furthermore, all nodes share the same Master key. Thus, the compromise of one single node jeopardizes the entire network. In our solution, each node has a private key to limit successful attacks. Moreover, a one-time-use session key is used to secure the communication between two nodes, and cannot be further used in future communications. Thus, our approach reduces the security impact in the event of a node compromise.
- ZigBee protocol uses a frame counter to provide freshness protection. In other words ordered sequence of inputs are used to reject frames that have been replayed. We show that this approach is not efficient. In our solution, we suggest to use a timestamp or a random values as a nonce to prevent replay attacks.
- The proposed approach relies on simple operations and does not involve computationally expensive cryptographic operations to provide protection against several attacks.

The rest of this paper is organized as follows: Section 2 is an overview of ZigBee security architecture. In Section 3 we highlight the most relevant security issues related to ZigBee. Our solution is presented in Section 4. We also discuss the security and performance of our solution in Section 5. Finally, we conclude the paper in Section 6.

2. ZigBee Security Architecture

We present the key points that are needed for a clear understanding ZigBee security architecture, and we omit the details that are not directly relevant to our work. However, a detailed survey on ZigBee security can be found in[1 9]. We also refer the reader to the related works in this extended version of this paper.
Three types of keys are available in ZigBee technology: Master, Link and Network keys.

- Master keys: Considered as the most important keys among communicating nodes. These keys are used in the Key Establishment Procedure called SKKE (Symmetric Key Key Exchange) to keep the exchange of link keys between two nodes confidential. They are pre-installed in each node during the manufacturing of devices, or may be set up over-the-air in the ZigBee network. These are usually shared among all nodes. New nodes also use the master key through SKKE protocol to set up the link keys with the other nodes.
- Link keys: these keys are used to encrypt all the information exchanged between two nodes. They are managed by the application level, and are unique between each pair of nodes.
- Network key: Initially generated by the Trust Center, these keys aim to protect against outsider attacks with little resources needed. They might also be regenerated at different intervals and are necessary for the new nodes to join the network. They are 128b keys and are shared among all devices in the network.

ZigBee network uses a Trust Center[8] to decide whether or not new nodes are authorized to join the WLAN. Usually there is only one Trust Center that broadcasts messages using the network key which all members can distinguish. The old network key is used in case a new one needs to be spread through the network. The frame counter is used to reject replayed frames and is also updated in this situation. Both network and link keys can be installed on each pair of devices. However for increased security the link key is always used, although more memory space is required.
ZigBee key establishment protocol has two different cases according to the configuration of TC. In Case 1, TC creates the LK itself and sends it to each principal. Therefore, the initiator and the responder have no role in the creation of LK. In Case 2, TC creates the Master Key MK and sends it to each principal. Using this MK, A and B initiate an SKKE procedure to establish LK. In this case the two principals create LK mutually using SKKE protocol. At the end of a successful run the two principals will be able to establish secure communication using the encryption key LK.The related scenario is visualized in Fig. 1. We describe the procedure in details as follows:

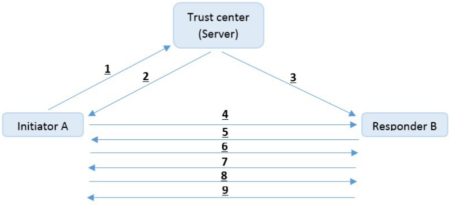

Fig. 1. ZigBee key establishment scenario

- **Message 1:** The initiator A begins the communication with the responder B by sending the first request message to the Trust Center.
- **Message 2 and 3**: TC sends the Master Key MK to each principal.
- **Message 4:** A sends B his request to start SKKE.
- **Message 5:** B receives the SKKE request from A. The messages (4) and (5) are encrypted by MK, which was received in the previous two messages. The remaining four messages represent the SKKE protocol itself.
- **Messages 6 and 7:** These steps include the challenges (NA, NB) of the principals encrypted by MK.
- **Messages 8 and 9:** Include a complex messages which can be computed by both parties to verify each other[5].

3. Security Concerns

Relevant security weaknesses related to ZigBee protocol are presented as follows:

1. Key distribution: The first vulnerability of ZigBee network is key distribution, as security keys are transmitted in ZigBee networks either over-the-air, or preinstalled onto the devices in an unsecure way[2]. There are different key distribution approaches depending on the security level:

   - Using the High Security level, the network key is encrypted and transmitted over the air using the Master key, which is shared among all nodes. Thus, the compromise of one single node leads to an untrusted relationship between all communicating devices in the network.
   - Using the Standard Security level, the safety of the system becomes even more critical as the network key is transmitted unencrypted over-the-air. Therefore, the Standard Security level has serious vulnerabilities and cannot be recommended for security purposes.
   - Using preinstalled network keys onto each legitimate device of the ZigBee network. This is done manually, and is not practical specially when the network size is large.

2. Frame counter: In ZigBee specification the notion of frame counter is offered as a security service and emphasized as the freshness protection. It uses an ordered sequence of inputs to reject frames that have been replayed. The counter will normally reset if a new key is created. The sequential freshness is used in this context to prevent malicious attacks. But it is not a strong approach for many reasons; for example an adversary can choose superior values to avoid the rejection of specific frames, as the frame counter uses incrementing values rather than random values. It is also easy to overflow the frame counters; as pointed out in previous published studies[6], an attacker might cause rejection of further frames and produce a denial of service simply by forging a frame with the maximum value 0xFFFFFFFF.
3. Forward security: Another weakness that can be found in the ZigBee security model is that the forward security requirement is not addressed properly (despite high security mode). Upon leaving the network, a node is still able to access the communication, because it still possesses the master and link keying material, due to the fact that a proper revocation has not been done. Indeed, if we take as an example a company or an institution using ZigBee for opening doors or improving energy efficiency etc, such a place needs a good approach to manage thousands of ZigBee devices. And a very possible situation is where one or many of the ZigBee devices are lost, misused or stolen. Using practical experiment, studies show that the extraction of security keys is possible[3 4].

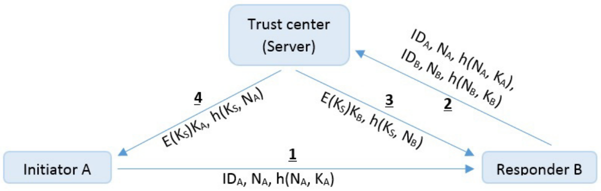
Trust center
(Server)
ID_A, N_A, $h(N_A, K_A)$,
ID_B, N_B, $h(N_B, K_B)$
2
4
$E(K_S)K_A$, $h(K_S, N_A)$
3
$E(K_S)K_B$, $h(K_S, N_B)$
1
Initiator A
ID_A, N_A, $h(N_A, K_A)$
Responder B

Fig. 2. Proposed security solution for enabled ZigBee wireless sensor networks

Thus, the extraction of data become very feasible, and this has been already demonstrated in published research projects like[3,4]. For that reason, if the keys stored on the devices are not properly revoked an adversary might take advantage of the situation, and exploit this weakness. As such attacking the network and application level becomes a very feasible task. Therefore, this type of attacks should not be underestimated and must be taken seriously.

4. Eavesdropping and data manipulation: More security vulnerabilities related to ZigBee enabled systems were demonstrated. For example, special software and hardware can be used for attacking purposes such as eavesdropping (traffic sniffing), data injection/manipulation or packet decoding. Cheap $40 devices like AVR RZ Raven USB (Universal Serial Bus) Stick (RZUSB) can be used to exploit many security vulnerabilities related to ZigBee enabled systems[7]. Furthermore, an Integrated Development Environment (IDE) based on GNU Compiler Collection (GCC) is a freely available for software development[7]. Another example is the KillerBee software suite[7], which is a modified version of RZUSB firmware. This tool is also capable of exploiting ZigBee vulnerabilities and is freely available software suite[7]. It is expected that these tools will be enhanced to be even more efficient in the near future, exposing more currently unknown ZigBee weaknesses.

We propose in the next section a new approach to fix the discussed weaknesses.

4. Proposed Solution

Three entities are involved in this proposal: the Trust Center, the initiator node A, and the responder node B. Each node i stores its identifier $\mathbf{ID_i}$ and its secret key $\mathbf{K_i}$. The Trust Center has access to a database where information related to the network is stored (in this case we are interested in the IDs and secret keys related to the nodes). No keys are shared permanently among the nodes, which decreases considerably the ability to compromise the network upon the exposure of one single node. Finally, the temporary session key Ks, is a one-time-use key shared between both the initiator and responder nodes during a given communication. In our approach, we propose to use random numbers as nonces to ensure the freshness of the messages containing the session keys. The nonce $\mathbf{N_i}$ is updated after each communication to prevent replay attacks. We use the following notations to describe our solution throughout the paper:

TC	Trust Center
A	Initiator node
B	Responder node
$\mathbf{ID_i}$	Identifier of a node i
$\mathbf{N_i}$	A random number generated by a node i
$\mathbf{K_i}$	The secret key of a given node i
$\mathbf{H_i}$	The output of the hash function $\mathbf{h}(\mathbf{N_i}, \mathbf{K_i})$
$\mathbf{E_K(M)}$	An encrypted message M with a key K
$\mathbf{K_S}$	One time use session key.

- **Step 1:** The initiator A sends a request to establish communication with the node B. The message (1) contains the node's identifier $\mathbf{ID_A}$, a nonce $\mathbf{N_A}$ generated by A and $\mathbf{H_A}$ which is a hash of $\mathbf{N_A}$ along with the private key $\mathbf{K_A}$. We added the nonce to this step to provide freshness and ensure that when receiving the next message, A will be sure that the communication has not been replayed as $\mathbf{N_A}$ is random and different for each session. Only the Trust Center has access to $\mathbf{K_A}$ and can rebuild $\mathbf{H_A}$ to verify if A is a legitimate node. Note that $\mathbf{H_A}$ is a one way function, as hash functions are required to be irreversible. Therefore, even if this message is disclosed, the attack cannot be successful, as only legitimate parties possess the secret key $\mathbf{K_A}$ to recover the plain message of the next step.

- **Step 2:** The responder B build its own message in the same way, and sends the received information related to A along with its information to the Trust Center as a request for authentication and also to obtain a new temporary session key.
- **Step 3:** The Trust Center (TC) receives the message (2) from B, and verifies at first whether the forwarded message is valid or not by rebuilding $\mathbf{H'_A}$ and $\mathbf{H'_B}$ using $\mathbf{K_A}$ and $\mathbf{K_B}$ for the stored $\mathbf{ID_A}$ and $\mathbf{ID_B}$ respectively. Comparing $\mathbf{H'_A}$ with $\mathbf{H_A}$ and $\mathbf{H'_B}$ with $\mathbf{H_B}$ proves the message is legitimate as only A and B possess the secret keys $\mathbf{K_A}$ and $\mathbf{K_B}$ and are able to build a valid message.
- **Step 4:** The trust center generate a the session key $\mathbf{K_S}$, and sends it in an encrypted form using $\mathbf{K_A}$ and $\mathbf{K_B}$ to both A and B respectively. The nonces $\mathbf{N_A}$ and $\mathbf{N_B}$ provide protection against replay attacks. Note that in our solution, both nodes A and B authenticate as legitimate nodes, and can verify the freshness of the received messages from the Trust Center.
- **Step 5:** Finally the nodes A and B receive the encrypted information, and retrieve the secret session key using their private keys $\mathbf{K_A}$ and $\mathbf{K_B}$ respectively. A and B are sure that the received message is fresh as it contains the nonces $\mathbf{N_A}$ and $\mathbf{N_B}$. At this point, both the initiator A and the responder B can communicate in a secure way using the session key $\mathbf{K_S}$. The proposed solution can be easily understood by looking at Fig. 2.

The Trust Center can make a periodical verification, and verify if all nodes are still in the WSN. If not, TC can revoke the access from a specific node simply by deleting or disabling its related information in the data base. This technique prevents from exploiting secret information by an adversary.

5. Security And Performance Analysis

5.1. Security Analysis

In our scheme the session key is not distributed using a Master key. Therefore it is unlikely that the network gets compromised upon a disclose of a specific key (which the case of the Master key in the original protocol). This is mainly due to the fact that the secrete key of each node is used to encrypt the message containing the temporary session key. Secondly, we use a random number to provide freshness of the exchange message, while in the original ZigBee protocol the notion of frame counter is adopted, and an ordered sequence of input is used to reject replayed frames, which is not a strong approach as an adversary can choose specific values to avoid the rejection of the frames. Our solution has also the following properties:

Data Secrecy: The information transmitted between the Trust Center and the nodes A and B is not understandable by a potential attacker as different private keys are used to communicate with each node. While in the original security protocol used by ZigBee devices the Link key is either transmitted in clear or by using the Master key that is shared among all nodes increasing the possibility of the compromise of the entire network. In our approach, each node stores an identifier and a private key to secure the communication. The hash function is irreversible and is also required to be a one-way function.

Resist replay attacks: Our solution is designed to counter replay attacks. In each session different random numbers are included in the message exchanges to prevent this type of vulnerability. For example, an eavesdropper could try to impersonate the Trust Center and replay one of its previous responses; however, the message would not be validated by the nodes, as the nonce included in the message would not be fresh, and would not match the nonce in the request message. Thus, the message would not match the verification and the attack would fail. Therefore, our approach resists replay attacks.

Authentication: This feature is important for many applications. In our approach, only legitimate parties that possess the secret keys $\mathbf{K_A}$ and $\mathbf{K_B}$ can create valid messages, and only genuine nodes can derive the session keys $\mathbf{K_S}$. Also the message exchange involves a hash function that allows data integrity to be checked.

Resist impersonation attack: For example, an adversary can try to be authenticated as a certain node, and gain access to the communication without being authorized to do so. In this case, the attack would not be successful, because the secret keys $\mathbf{K_A}$ and $\mathbf{K_B}$ are unknown; therefore, it is not possible to retrieve the session key $\mathbf{K_S}$.

5.2. Static Performance Assessment

In addition to providing many security properties against several possible attacks on the ZigBee systems, our solution has low computational cost, and fits constrained wireless sensor networks in terms of communication and storage requirements.

Computation cost: Standard cryptographic algorithms like public key systems have a very high computational cost, and need large memory space. Therefore these kinds of methods are not suitable for very constrained devices such as wireless sensor nodes. Our proposal requires only a hash function to be implemented and a random number generator. Both the nodes and the Trust Center have enough computational power to handle cryptographic operations based on symmetric key cryptosystem.

Communication Cost: Our solution is accomplished with only four messages between the Trust Center and the nodes, versus nine messages in the original ZigBee security protocol. Therefore, the proposed scheme reduces considerably the communication overheads and is considered practical and feasible.

Storage requirement: Each node needs only to store its private key, instead of three different keys in the original protocol (Master, link and network key). In addition to the implementation of the hash function and the random number generator. The nonce is stored in a rewritable memory because it needs updates. This is considered very low compared to the original ZigBee security protocol. Thus, the proposed solution is lightweight and practical.

6. Conclusion

We have presented the security model of ZigBee, this model has improved considerably through many expert reviews. However, the model presents deficiencies that may limit its application, and several attacks are still possible as mentioned in this paper. This work is an attempt to highlight the most relevant weaknesses and proposes a new approach to enhance security. Our solution is efficient and prevents many security attacks. Moreover, the computational cost and storage requirements are quite low compared to standard solutions.

Acknowledgment

I would like to thank the ”Fulbright program” for supporting this paper through funding my mobility in USA as a visiting scholar in Southern Illinois University.

References

1. ZigBee Alliance. Zigbee specifications: Zigbee and zigbee pro [online]. available http://www.zigbee.org.
2. Kyung Choi, Minjung Yun, Kijoon Chae, and Mihui Kim. An enhanced key management using zigbee pro for wireless sensor networks. In *Information Networking (ICOIN), 2012 International Conference on*, pages 399–403. IEEE, 2012.
3. Travis Goodspeed. Extracting keys from second generation zigbee chips. *Black Hat USA*, 2009.
4. GoodFET Project. [online]. available: http://goodfet.sourceforge. net.
5. FIPS Pub. 198, the keyed-hash message authentication code (hmac). *Federal Information Processing Standards Publication*, 198, 2002.
6. Naveen Sastry and David Wagner. Security considerations for ieee 802.15. 4 networks. In *Proceedings of the 3rd ACM workshop on Wireless security*, pages 32–42. ACM, 2004.
7. Joshua Wright. Killerbee: Practical zigbee exploitation framework or wireless hacking and the kinetic world available: http://www.willhackforsushi.com/presentations/ toorcon11-wright.pdf.
8. Yang Xiao, Venkata Krishna Rayi, Bo Sun, Xiaojiang Du, Fei Hu, and Michael Galloway. A survey of key management schemes in wireless sensor networks. *Computer communications*, 30(11):2314–2341, 2007.
9. Ender Yüksel, Hanne Riis Nielson, and Flemming Nielson. Zigbee-2007 security essentials. In *Proc. 13th Nordic Workshop on Secure IT-systems*, pages 65–82, 2008.