Smart Computing Review

ZigBee Security for Residential Sensor Networks

Lee-Chun Ko[1] and Jin-Shyan Lee[2]

[1] Information & Communication Research Labs, Industrial Technology Research Institute / Hsinchu, Taiwan – ROC / brentko@itri.org.tw

[2] Department of Electrical Engineering, National Taipei University of Technology / Taipei, Taiwan – ROC / jslee@mail.ntut.edu.tw

*Corresponding author: Jin-Shyan Lee

Received October 8, 2011; revised November 5, 2011; accepted December 1, 2011; published December 31, 2011

Abstract: ZigBee standard is a low-complexity and energy-efficient wireless protocol for sensor network applications. It has been promoted by ZigBee alliance and widely developed by many companies for a variety of applications. In this paper, we introduce the residential mode of security mechanism in ZigBee networks for smart home networking applications, and we then point out the security problems when using this mode. Specifically, we show that the ZigBee network is subject to replay attacks, and thus packets information could be revealed to attackers. Moreover, the encryption keys are also possible to be revealed during the key update operation within ZigBee networks. To overcome the latter issues, in this paper, we provide some recommendations to improve the security of ZigBee sensor networks for residential applications. It is believed that the suggestions presented in this paper would benefit application engineers in designing the sensor networking security.

Keywords: Security, ZigBee, IEEE 802.15.4, wireless sensor networks

Introduction

Recently, internet of things (IoT) has become a very attractive research area worldwide. One of the topics in IoT is the wireless sensor networks (WSNs) with self-organization capabilities to cope with sensor failures, changing

A preliminary version of this paper appeared in International Conference on Security of Information and Networks, 2007, May 8-10, Gazimagusa (TRNC), North Cyprus. This version includes an updated revision, ZigBee r18 over IEEE 802.15.4-2006, and the comparison with previous specifications. This research was supported by the National Science Council, Taiwan, under Grant NSC 100-2221-E-027-010 and by the Ministry of Economic Affairs, Taiwan, under Grant OP22-9352-SN-2300.

DOI: 10.6029/smartcr.2011.02.001

environmental conditions, and different environmental sensing applications [1-6]. WSNs consist of many small sensing devices featuring low-power, low-cost, and self-configuration. These sensing devices are usually deployed in a large area and equipped with different types of sensors for sensing physical environment components such as light, temperature, and humidity. The sensed data are often collected and converted into digital information, and then conveyed to the back-end host via a network routing mechanism for further analysis. Applications of WSNs include battle field, medical, security, disaster monitoring, and automation. In order to achieve low-power, low-cost, and large amount of deployments, battery-powered sensing devices (usually with an 8-bit microcontroller and small-size memory) are the common hardware architecture.

In WSNs, the packets transmitted over the air should be protected via a cryptographic mechanism to prevent wireless attacks, such as eavesdropping, injection, and modification. The attacker could mount various attacks and the network would not function normally, or even entirely breakdown. Security issues in WSNs include not only protecting packets from eavesdropping but also key distribution, key management, device authentication, and secure routing [7-12]. Imagine that a WSN applied to a burglar alarm system without using a security policy, the attacker could inject packets to induce a false alarm or even to suppress the real alarm. Moreover, in battle fields, the attacker could sniff packets and obtain critical information from the network, or send bogus information to disturb intelligence information. Therefore, in real applications, WSNs should provide security policies in order for the network to work securely and normally.

For WSNs, some popular protocols have been developed, such as ZigBee [13-14], Z-Wave, Insteon, and Ultra-Low-Power Bluetooth (previously named Wibree). Among these protocols, the ZigBee protocol is considered the most popular on the market [15]. The ZigBee protocol provides two security modes when initializing the network, i.e., commercial mode and residential mode. In the residential mode, every device uses only network keys to encrypt and decrypt packets in the network layer, also in the medium access control (MAC) sub-layer if MAC security is enabled. In commercial mode, in addition to network keys, link keys are used in the application support layer to provide end-to-end security.

In this paper, the procedure of residential mode and key update on ZigBee network are introduced, and we then point out several security flaws when operating in this mode. We further describe that the ZigBee network remains subject to MAC commands replay attacks even if MAC security is enabled. If MAC security is configured to access control list mode or unsecured mode, then the attacker could infer some information of network payload of packets. More seriously, the ZigBee network may reveal a new network key when updating the current used key. Finally, we recommend several enhancements to ZigBee to improve the security of residential applications.

Several studies on the IEEE 802.15.4 security have been reported [16-18]. However, comparisons between the different versions of ZigBee and IEEE 802.15.4 are seldom discussed. This paper is an extension of our previous work [18], which only discussed the security issues of version ZigBee r07 [13] over IEEE 802.15.4-2003 [19]. In this paper, we not only extend to the issues of ZigBee r18 [14] over IEEE 802.15.4-2006 [20], but we also make a comparison between the two versions.

Security Mechanism in ZigBee Networks

■ ZigBee Protocol Stack

The ZigBee stack architecture is made up of a set of blocks called layers. Each layer performs a specific set of services for the layer above, including a data entity that provides a data transmission service and a management entity that provides all other services. Each service entity exposes an interface to the upper layer through a service access point (SAP), and each SAP supports a number of service primitives to achieve the required functionality. The ZigBee stack architecture, as shown in Figure 1, is based on the standard open systems interconnection seven-layer model, but defines only those layers relevant to achieving functionality in the intended market space. The IEEE 802.15.4 [19-20] defines specifications of the physical layer and MAC layer for supporting simple devices that consume minimal power and typically operate in a personal operating space. The ZigBee Alliance builds on this foundation by providing the network layer and the framework for the application layer, which includes the application support sub-layer (APS), the ZigBee device object (ZDO), and the manufacturer-defined application objects.

Wireless links under 802.15.4 can operate in three license free industrial scientific medical (ISM) frequency bands. These accommodate over air data rates of 250 kbps in the 2.4 GHz band, 40 kbps in the 915 MHz band, and 20 kbps in the 868 MHz. A total of 27 channels are allocated in 802.15.4, including 16 channels in the 2.4 GHz band, 10 channels in the 915 MHz band, and 1 channel in the 868 MHz band. The IEEE 802.15.4 MAC sub-layer controls access to the radio channel using the CSMA-CA mechanism. Its responsibilities also include transmitting beacon frames, synchronization and providing a reliable transmission mechanism.

The responsibilities of the ZigBee network layer include mechanisms used to join and leave a network, to apply security to the frames and route them to their intended destinations. The discovery and maintenance of routes between devices takes place in the network layer. Also, the discovery of one-hop neighbors and the storing of pertinent neighbor information are done at the network layer. In addition, the network layer of a ZigBee coordinator is responsible for starting a new network, when appropriate, and assigning addresses to newly associated devices.

The ZigBee application layer consists of the application support sub-layer, ZigBee device object, and manufacturer-defined application objects. The application support sub-layer is responsible for maintaining tables for binding, which is the ability to match two devices together based on their services and needs, and forwarding messages between bound devices. The responsibilities of the ZigBee device object include defining the role of the device within the network (e.g., ZigBee coordinator or end device), initiating and responding to binding requests, and establishing a secure relationship between network devices. Another responsibility of the ZigBee device object is discovery, which is the ability to determine which other devices are operating in the personal operating space and the devices associated. The manufacturer-defined application objects adhere to profiles defined within the ZigBee Alliance. They implement the actual applications according to the ZigBee-defined application descriptions. The device profile is a series of messages that permit the ZigBee devices to perform the functions forming the core capabilities of discovery, binding, and network management for the ZigBee devices.

Security services provided for ZigBee include methods for key establishment, key transport, frame protection, and device management. These services form the building blocks for implementing security policies within a ZigBee device. The architecture includes security mechanisms at three layers of the protocol stack. The MAC, network, and application support layers are responsible for the secure transport of their respective frames. The security mechanisms provided by the application support and network layers are for the processing the secure MAC frames. Furthermore, the application support sub-layer provides services for the establishment, and maintenance of security relationships. The ZigBee device object manages the security policies and the security configuration of devices.

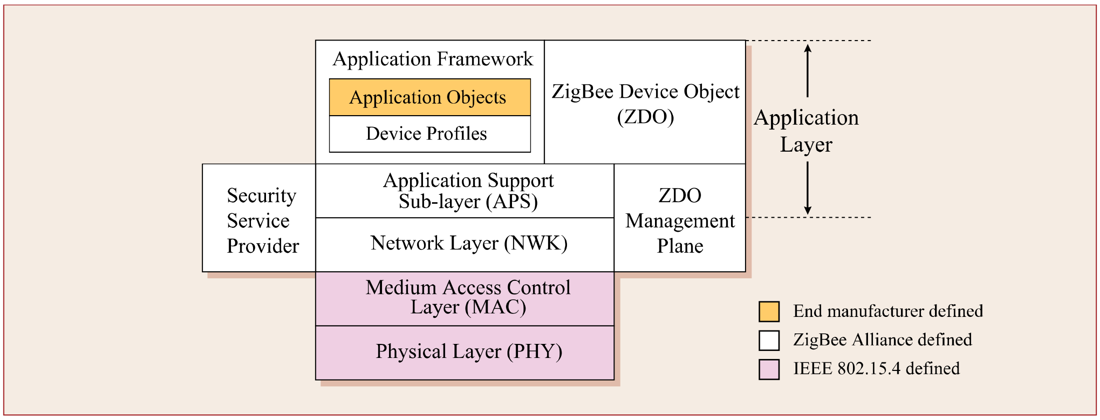


Figure 1. The ZigBee/IEEE 802.15.4 stack architecture

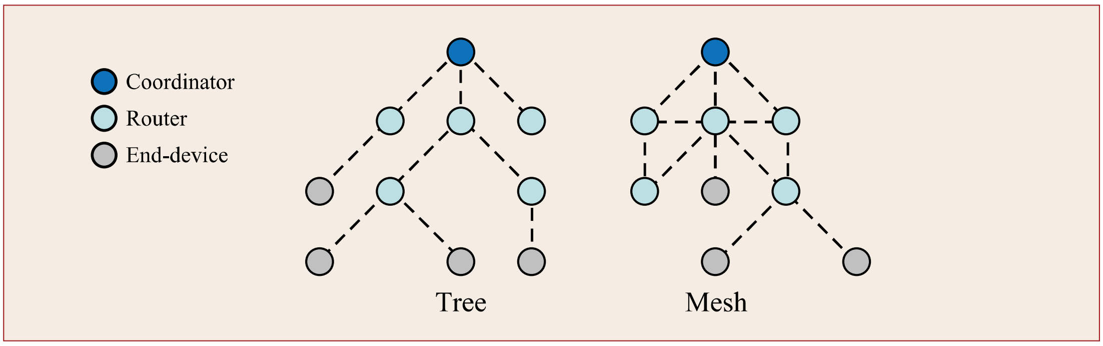


Figure 2. The topology supported by ZigBee network

■ Network Topology

The ZigBee network supports mesh and tree network topologies. As illustrated in Figure 2, there are three different types of ZigBee devices defined in the ZigBee network.

- Coordinator is the most capable device and there is only one coordinator in each network. Typically, the coordinator is connected to the network gateway and responsible for network formation and maintenance. In terms of security, the coordinator acts as a trust center for key establishment, key management, and authentication with each device.
- Router acts as an intermediate device for forwarding packets to other devices.
- End-device is the sensing element with less computing capability. End-devices can only communicate with their parent such as router or coordinator.

Encryption & Decryption

ZigBee adopts the advanced encryption standard (AES) block cipher with counter mode (CTR) and cipher block chaining message authentication code (CBC-MAC), also known as CTR with CBC-MAC (CCM) with 16-bit cyclic redundancy check (CRC). More specifically, ZigBee uses CCM* to encrypt and decrypt outgoing and incoming packets with a minor modification of CCM [19]. CCM* includes all features of CCM and additionally provides encryption-only and authentication-only capabilities. Currently, CCM* can only be used with a 128-bit block cipher, such as AES-128. Table 1 shows eight different security levels defined by the ZigBee protocol when executing the procedure of CCM*. The security attributes of MIC-32, MIC-64, and MIC-128 indicate that the encrypted packet contains only message integrity code (MIC) but has no encryption of the packet payload. The packet payload is encrypted when the security attributes contain ENC.

Table 1. Security level used in CCM*

Security level identifier	Security attributes	Data encryption	Length of MIC (bytes)
0x00	None	No	0
0x01	MIC-32	No	4
0x02	MIC-64	No	8
0x03	MIC-128	No	16
0x04	ENC	Yes	0
0x05	ENC-MIC-32	Yes	4
0x06	ENC-MIC-64	Yes	8
0x07	ENC-MIC-128	Yes	16

CCM* uses nonce to ensure that encrypting two identical plaintexts will not have two identical cipher texts. Generally, nonce is appended to outgoing packets without encryption. The nonce used in CCM* is composed of 8-byte sender's IEEE address, 4-byte frame counter, and 1-byte security control. Once the nonce and key are available, the procedure of CCM* in the network layer for outgoing packets can be executed. The original packet consists of a network header and payload. The network payload may be generated by the network layer itself or the application support layer. The procedure of CCM* is executed as follows. First, add an auxiliary (AUX) header that consists of nonce and a key sequence number between the network header and the payload. Then, the network header, AUX header, and network payload are included to compute MIC if the security attribute contains MIC; the network payload and MIC are encrypted if the security attribute contains ENC. Finally, the packet is then sent to the MAC layer if no error is occurred during the execution of the CCM* procedure. Encrypting the packet payload in CCM* can be performed by dividing the packet payload into 16-byte size blocks $M_1||\ldots||M_n$, and padding zero if the length of packet payload is not divisible by 16-byte. Then, each ciphertext block $C_1||\ldots||C_n$ is computed by

$$C_i = E(Key, A_i) \oplus M_i \qquad (1)$$

where $E(K,X)$ means AES-128 encrypting block X with key K. $A_i = flags \mathbin{||} nonce \mathbin{||} counter\ i$ for i=1, 2, …, n, of which *flags* is 1-byte constant, 2-byte *counter* is initialized to 1 and increased by 1 after encrypting each block. The MIC is also encrypted using a similar way with *counter* set to 0. $\oplus$ is the XOR operation that is a connective in logic known as the "exclusive or". It yields true if exactly one (but not both) of two conditions is true.

Once each device in the routing path and the intended destination device receive an encrypted packet, they retrieve nonce from the AUX header and use a key to compute the key stream. The decrypted payload and MIC are obtained by

XORing the key stream and encrypted packet. Then, MIC is verified by comparing the received MIC with the computed MIC to determine whether or not they are the same. If the verification of MIC is correct, then the packet is parsed for further analysis. Otherwise, the packet is dropped and reported to the upper layer. Due to the space limitation of this paper, the readers may refer to the ZigBee specification for more detailed descriptions on the procedure of CCM*.

■ Authentication in Residential Mode

When a device successfully joins a secure network, this device must be authenticated by the coordinator before sending packets into the network. The residential mode in the ZigBee network relies on maintaining the network keys in each device for encrypting and decrypting packets. There are two different authentication procedures in residential mode: with or without pre-configured (or pre-installed) network key. Operating without a pre-configured network key is not secure because the network key is sent to a new joiner without any encryption and the network key could possibly be revealed in this transfer. In this paper, we do not offer details on the authentication procedure of non-pre-configured network keys. We here focus on the authentication procedure of pre-configured network keys, i.e., each sensor device is pre-configured with the network key before deployment into the network. A detailed description on this procedure is briefly introduced as follows.

The security services provided for ZigBee include methods for key establishment, key transport, frame protection, and device management. These services form the building blocks for implementing the security policies within a ZigBee device. The architecture includes security mechanisms at three layers of the protocol stack. The MAC, network, and application support layers are responsible for the secure transport of their respective frames. The security mechanisms provided by the application support and network layers are for the processing of secure MAC frames. Furthermore, the application support sub-layer provides services for the establishment and maintenance of security relationships. The ZigBee device object manages the security policies and the security configuration of devices.

After a new joiner successfully joins a router with a pre-configured network key (by performing the secure join procedure), the router will send an encrypted *Update-Device* command to the coordinator to inform that a new device has joined the network. If the coordinator accepts this new joiner, then it will directly send an encrypted *Transport-Key* command (as shown in Figure 3) with 16-byte *Key* field containing a dummy key (i.e., 16 bytes 0x00) to the new joiner. Then, this new joiner is considered to be authenticated. More detailed descriptions on secure join and residential mode authentication can be found in the ZigBee specification.

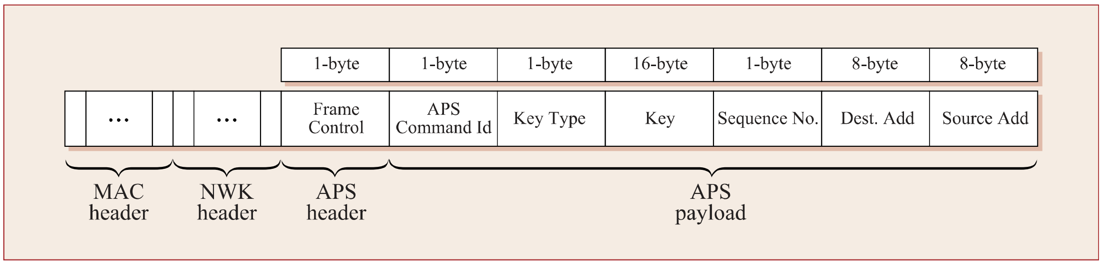


Figure 3. The format of *Transport-Key* command [13]

Security Issues in ZigBee Residential Mode

In this Section, we pointed out some security flaws both in the old and new versions of ZigBee networks when operating in the residential mode. Note that the ZigBee r07 [13] and ZigBee r18 [14] versions adopt IEEE 802.15.4-2003 [19] and IEEE 802.15.4-2006 [20] as their MAC layer specifications, respectively.

■ Replay Attacks

Since the residential mode does not provide end-to-end security, ZigBee defined that the MAC layer uses network keys as the default keys for packets encryption and decryption. When using the default keys, the ZigBee specification specified that the MAC layer security should not use the optional external frame counter (see Section 3.3.2 of [13]), i.e., no checking incoming frame counter. Therefore, the replay attacks can be mounted through this vulnerability. The network layer will drop the replayed packets through checking the incoming frame counter of network packets. However, the MAC command packets are parsed only in the MAC layer. Consequently, the replaying MAC command packets would be possible. IEEE

802.15.4 defined nine different MAC commands such as association request, association response, disassociation notification, data request, etc. (see Section 7.3 of [19]).

All MAC commands could be replayed. If the association request and data request commands are replayed, then the attacker is able to use an alternative device to join to the network after the original device leaves the network. For example, a device *D*1 joins router *R*1 by sending an association request and then a data request to *R*1. After *D*1 leaves the network for various reasons (e.g., *R*1 forces *D*1 to leave), the attacker replays these two commands to *R*1, and *R*1 still accepts this association. This attack will cause the network existing the dummy device that do not actually present. If the disassociation notification command is replayed, then the attacker is able to force the target device to leave the network after the target device rejoins the network. For example, a device *D*1 is notified to leave the network by receiving a disassociation notification command from its parent router *R*1. Once *D*1 rejoins *R*1, the attacker is able to force *D*1 to leave the network by replaying the previous disassociation notification command sent by *R*1. We do not offer details of the complete results of replaying each different MAC command. Interested readers can easily infer that replaying other commands in some circumstances will cause different types of attacks, and we believe that such attacks cause serious damages to the network.

If someone changes the implementation of ZigBee specification by checking the incoming frame counter for preventing replay attacks, then another problem will occur. IEEE 802.15.4-2003 has a problem in checking the incoming frame counter when using the default security, i.e., default key [16]. When using the default security, each device maintains exactly one incoming frame counter for all neighbor devices. Therefore, suppose a device *D1* receives 10 packets with frame counter 0 to 9 from another device; the packets sent from other devices to *D1* with a frame counter lower than 10 will be rejected since *D1* maintains only one largest incoming frame counter for all neighbor devices.

Revealing Network Keys

In this paper, we give a concrete procedure of how to obtain the network key, which is used to secure the entire ZigBee network. Note that this attack is applicable to both versions of ZigBee specifications when operating in the residential mode. Here, we emphasize that once the network key is revealed, the entire network is no longer secure even if the coordinator sends new key, which is encrypted using the old key, to all nodes.

- Noncе Reuse: As mentioned before, the procedure of CCM* is similar to stream cipher; it uses nonce and key to generate key stream and produces ciphertext by XORing plaintext and key stream. Hence, it is obvious that the same nonce and key will generate the same key stream. If the attacker finds out two encrypted packets use the same nonce (this can be found by observing network AUX header), then XORing these two encrypted packets will obtain the XORed result of decrypting these two packets. For example, suppose two encrypted packets c_1 and c_2 use the same nonce, XORing these two encrypted packets will have

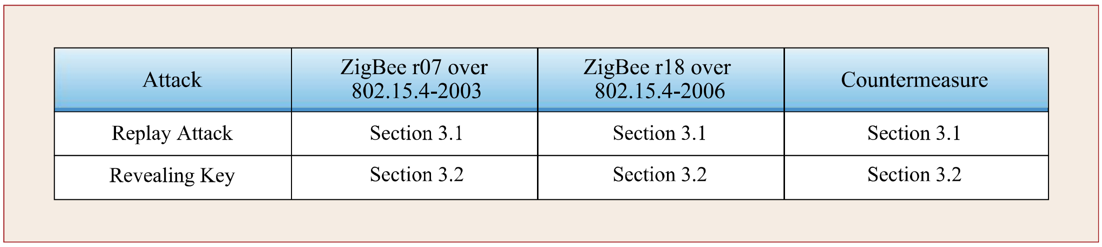
  $$c_1 \oplus c_2 = [m_1 \oplus E(key, nonce)] \oplus [m_2 \oplus E(key, nonce)] = m_1 \oplus m_2 \qquad (2)$$

  Once some fields of m_1 are known, the corresponding fields of m_2 are also revealed, and vice versa. The only one situation using the same nonce in CCM* is that the same device sends two encrypted packets using the same frame counter. Normally, this situation would not occur in ZigBee networks since the frame counter is initialized at 0 and increased by 1 each time after sending one packet until 2^{32}-1 packets have been sent. However, there still are some possibilities that one device reuses the frame counter, and hence reuses the nonce [16]. Two situations will reset the frame counter to 0, that is 1) *Power Failure*: When a device exhausts the energy or the attacker temporarily disrupts the power source, and 2) *Reforming Networks*: Some application or maintenance requirements that need parts of the devices to rejoin to the network, or re-construct the entire network. The following attack shows the network key is possibly revealed by using this security vulnerability.

- The Attacks: As described before, in order not to reveal the network key, it is suggested to use a pre-configured network key when joining the network. Using a pre-configured network key causes the trust center to send a *Transport-Key* command that contains a dummy key to new joiner. The following attacks infer a new network key when performing the network key update procedure by using the *Transport-Key* commands and nonce reuse problem. The steps of the attack are described as follows. 1) The attacker collects the *Transport-Key* commands sent or forwarded by the target devices and constructs the attack tables (for example, Table 2) for each target device, these attack tables record which device sends or forwards the *Transport-Key* commands using what frame counter value, and also the detailed contents of this packet. 2) The attacker temporarily disrupts the power source by any means to induce the target devices reboot. This will cause the target devices to rejoin the network and reset the frame counter to 0. 3) Once the attacker observes that the target devices send another packet *p* using the same frame counter that is recorded in the attack table, the attacker is then able to obtain some information from packet *p* by simply XORing *p* and the *Transport-Key* command that is recorded in the attack tables. Most fields in the *Transport-Key* commands are known, such as *APS Command Id*, *Key Type*, *Key*, and *Source Address*; these fields are totally 26 bytes. Therefore, the corresponding 26-byte fields of the network payload of packet *p* will be revealed.

4) Another situation may reveal the new network key. If the coordinator observes that there are too many devices rejoining the network due to step 2 or other policies that need to perform the network key update procedure, then the new network key is broadcast to all devices via the coordinator using the a *Transport-Key* command. However, once any target device re-broadcasts the *Transport-Key* command using the same frame counter that is recorded in the attack tables, the new network key is revealed by XORing two *Transport-Key* commands (one for sending a dummy key and one for re-broadcasting a new network key) that use the same frame counter.

Table 2. Summery of Attacks and Countermeasures



Attack	ZigBee r07 over 802.15.4-2003	ZigBee r18 over 802.15.4-2006	Countermeasure
Replay Attack	Section 3.1	Section 3.1	Section 3.1
Revealing Key	Section 3.2	Section 3.2	Section 3.2

One problem is how to recognize the *Transport-Key* commands. We know that the network payload consists of the application support data packets or application support command packets. Unfortunately, the *Transport-Key* command is 36-byte length, which is different from all other application support commands. The length of MAC header can be deduced from the contents of MAC header, length of network header is fixed to 8 bytes, length of AUX header is fixed to 14 bytes, length of MIC can be deduced from the security level, and the length of CRC checksum is 2 bytes. Accordingly, if the attacker finds a packet that has a network payload length of 36 bytes, then this packet has a very high probability of being a *Transport-Key* command. The only one exception is that the application support data packets have an exact length of 36 bytes. However, a 2-bit *Frame Type* in the application support header indicates that the packet is a data packet or command packet. This can be used to further confirm that two XORed packets are both command packets. Another method can also confirm that two XORed packets are both *Transport-Key* commands by observing whether or not the XORed results of *APS Command Id* and *Key Type* fields are zero. The security level can be known by sending a beacon request to the coordinator or any router, and the *Stack Profile* field contained in the response packet will then show which security level is currently used in this network.

Security Improvement for Residential Mode

As mentioned above, both versions of ZigBee networks have several security flaws when operating in the residential mode. These attacks could be easily carried out to cause serious damage to the ZigBee network or even reveal the network key. We would like to give several changes to the specification that we believe will improve the security of ZigBee users. The previous section identified three clusters for QoS classes and features to build up classification rules through unsupervised learning. In this section, the accuracy of the classification rules is experimentally evaluated. For classification, we chose the K-nearest neighbor (KNN) algorithm. Our experimental results are compared with the minimum mean distance (MMD) classifier.

■ Security Suggestions for ZigBee r07 Version

Basically, the access control list mode and unsecured mode in IEEE 802.15.4-2003 should never be used since two modes do not perform any cryptographic operations on the packets. Using only the network layer security is not strong enough to prevent all attacks. The cryptographic operation in the MAC layer is necessary. We suggest not using the default security material. Each device should maintain different incoming frame counters for all neighbor devices when operating in the residential mode. This can be done by using an access control list entry [16] to maintain security information about neighbor devices. Fortunately, ZigBee r18 version does not have such a problem.

■ Security Suggestions for ZigBee r18 Version

Fundamentally, the *macSecurityEnabled* of MAC attributes (see section 7.4.2 of [20]) in IEEE 802.15.4-2006 should be TRUE. In addition, the format of nonce in CCM* should be modified; an 8-byte source address in nonce can be modified to an 8-byte random number when sending each outgoing packet, or when the device is rebooted. This avoids the nonce reuse problem even if the device is rebooted and the frame counter is then reset to 0. The reason for changing the source address to a random number is that each router in the routing path will decrypt and then encrypt the forwarding packets. Therefore, after the destination device receives the encrypted packets, the 8-byte source address in the AUX header is the source address of the previous hop router instead of the original source device. The destination device is still able to know the source address of the original source device from the packet herder. Hence, modifying the source address to an 8-byte random number does not affect any security requirements.

The packet length of each application support command should be equal. If each application support command has a different packet length, then the attacker can easily identify the purpose of each application support command. This may cause some levels of security threats. Therefore, we suggest that each application support command related to security should have an equal packet length to prevent such threat. The simplest way of achieving this is to add some unrelated fields with random contents at the end of some shorter commands, making the length of all packets equal.

Conclusions

This paper has briefly introduced the security issues of ZigBee networks and pointed out several flaws when operating in the residential mode. The security flaws include the MAC commands replay attack and packet loss. We also suggest that both access control list mode and unsecured mode in the MAC layer should not be used. Otherwise, the packet contents and network key will be revealed due to the nonce reuse problem. Moreover, several recommended designs for preventing from these attacks are also presented to improve the security of ZigBee residential sensor networks. It is believed that the suggestions presented in this paper would benefit application engineers in designing sensor networking security. Future work will attempt to investigate the security issues of the KillerBee project [21] to provide a more comprehensive comparison of ZigBee networks.

References

[1] J. Zheng, P. Lorenz, P. Dini, “Guest editorial: Wireless sensor networking,” *IEEE Network*, vol. 20, no. 3, pp. 4-5, May-Jun. 2006. Article (CrossRef Link)

[2] T. Pfeifer, S. Olariu, Alois Ferscha, “Editorial: Special issue on wireless sensor networks and applications,” *Comput. Commun.*, vol. 28, no. 13, pp. 1481-1483, 2005. Article (CrossRef Link)

[3] D. Culler, D. Estrin, M. Srivastava, “Guest editors’ introduction: Overview of sensor networks,” *IEEE Computer*, vol. 37, no. 8, pp. 41-49, Aug. 2004. Article (CrossRef Link)

[4] F. Xia, A. Vinel, R. Gao, L. Wang, T. Qiu, “Evaluating IEEE 802.15.4 for cyber-physical systems,” *EURASIP J. Wireless Communications & Networking*, vol. 2011, 2011. Article (CrossRef Link)

[5] J. S. Lee, “Performance evaluation of IEEE 802.15.4 for low-rate wireless personal area networks,” *IEEE Trans. Consumer Electronics*, vol. 52, no. 3, pp. 742-749, Aug. 2006. Article (CrossRef Link)

[6] J. S. Lee, Y. C. Huang, “ITRI ZBnode: A ZigBee/IEEE 802.15.4 platform for wireless sensor networks,” in *Proc. of IEEE Int. Conf. Systems, Man & Cybernetics*, pp. 1462-1467, Oct. 2006. Article (CrossRef Link)

[7] H. Chan, A. Perrig, D. Song, “Random key predistribution schemes for sensor networks,” in *Proc. of IEEE Symp. Security & Privacy*, pp. 197-213, 2003. Article (CrossRef Link)

[8] W. Du, J. Deng, Y. S. Han, P. K. Varshney, “A pairwise key pre-distribution scheme for wireless sensor networks,” in *Proc. ACM Conf. Computer & Communications Security*, pp. 42-51, 2003. Article (CrossRef Link)

[9] L. Eschenauer, V. D. Gligor, “A key-management scheme for distributed sensor networks,” in *Proc. of ACM Conf. Computer & Communications Security*, pp. 41-47, 2002. Article (CrossRef Link)

[10] C. Karlof, N. Sastry, D. Wagner, “Tinysec: A link layer security architecture for wireless sensor networks,” in *Proc. of ACM Conf. Embedded Networked Sensor Systems*, pp. 162-175, 2004. Article (CrossRef Link)

[11] C. Karlof and D. Wagner, “Secure routing in wireless sensor networks: Attacks and countermeasures,” *Elsevier’s Ad Hoc Networks Journal*, pp. 293-315, 2003. Article (CrossRef Link)

[12] A. Perrig, R. Szewczyk, J. D. Tygar, V. Wen, D. E. Culler, “Spins: Security protocols for sensor networks,” in *Proc. ACM Conf. Mobile Computing & Networks*, pp. 189-199, 2001. Article (CrossRef Link)

[13] ZigBee Alliance, “ZigBee Document 053474r07: ZigBee Specification,” San Ramon, CA, USA, September 2005.

[14] ZigBee Alliance, “ZigBee Document 053471r18: ZigBee Specification,” San Ramon, CA, USA, June 2009.

[15] J. S. Lee, Y. W. Su, C. C. Shen, “A comparative study of wireless protocols: Bluetooth, UWB, ZigBee, and Wi-Fi,” in *Proc. of IEEE Conf. Industrial Electronics (IECON)*, pp. 46-51, Nov. 2007. Article (CrossRef Link)

[16] N. Sastry and D. Wagner, “Security consideration for IEEE 802.15.4 networks,” in *Proc. of ACM Workshop Wireless Security*, pp. 32-42, 2004. Article (CrossRef Link)

[17] T. Shon, B. Koo, H. Choi, Y. Park, “Security architecture for IEEE 802.15.4-based wireless sensor network,” in *Proc. of the Int. Symp. Wireless & Pervasive Computing*, Feb. 2009. Article (CrossRef Link)

[18] L. C. Ko, “Security considerations for residential mode on ZigBee network,” in *Proc. of Int. Conf. Security of Information & Networks*, pp. 126-13 , May 2007.

[19] IEEE 802.15.4-2003, “Wireless Medium Access Control (MAC) and Physical Layer (PHY) Specifications for Low-Rate Wireless Personal Area Networks (LR-WPANs),” IEEE, Oct. 2003.

[20] IEEE 802.15.4-2006, “Wireless Medium Access Control (MAC) and Physical Layer (PHY) Specifications for Low-Rate Wireless Personal Area Networks (LR-WPANs),” IEEE, Sep. 2006.

[21] J. Wright, “Killerbee: Practical Zigbee Exploitation Framework,” in *Proc. of Computer Security Conference*, Oct. 2009.

Lee-Chun Ko received the B.S. degree in computer science from the National Taiwan University of Science and Technology, Taipei, Taiwan, R.O.C., in 2003, and the M.S. degree in computer science from National Central University, Jhongli, Taiwan, in 2005. During 2005-2011, he was a design engineer at the Information and Communications Research Laboratory, Industrial Technology Research Institute (ITRI), Hsinchu, Taiwan. He was a visiting scholar at the Department of Electrical and Computer Engineering, Carnegie Mellon University, Pittsburgh in 2008 and 2009. His current research interests include security issues of wireless sensor network, secure protocol design and wireless communication security.

Jin-Shyan Lee received the B.S. degree in mechanical engineering from the National Taiwan University of Science and Technology, Taipei, Taiwan, R.O.C., in 1997, and the M.S. and Ph.D. degrees in electrical and control engineering from National Chiao Tung University, Hsinchu, Taiwan, in 1999 and 2004, respectively. During 2003-2004, he was a Visiting Researcher at the Department of Electrical and Computer Engineering, New Jersey Institute of Technology, Newark. He was a Researcher at the Information and Communications Research Laboratory, Industrial Technology Research Institute (ITRI) during 2005-2009. Since August 2009, he has been an Assistant Professor of Department of Electrical Engineering at National Taipei University of Technology. His current research interests include Petri nets, wireless sensor networks, remote monitoring and control, supervisory control, and hybrid systems. His research work has led to a number of papers in journals and conference proceedings. He was invited to speak at North New Jersey IEEE Control Systems Chapter, Newark, and University of Rome “La Sapienza,” Rome, Italy. Dr. Lee is the recipient of 2010 “Early Career Award” from the IEEE Industrial Electronics Society (IES), 2008 “Youth Automatic Control Engineering Award” from the Chinese Automatic Control Society (CACS), 2004 “International Scholarship” from the Society of Instrument & Control Engineers (SICE), and a finalist in both the Annual International Award and Young Author’s Award at the 2004 SICE Annual Conference, Sapporo, Japan. He is a member of the Technical Committee on Discrete Event Systems of the IEEE Systems, Man, and Cybernetics (SMC) Society. He has served for various IEEE conferences as technical program committee (TPC) member. He is also an active reviewer in several journals focusing on Petri nets and sensor networks. He organized special sessions related to Petri nets and wireless sensor networks at the 2011, 2010, 2009 and 2007 IEEE Industrial Electronics Conference (IECON), 2010 SICE, and 2006 IEEE International Conference on SMC. He is an IEEE Senior Member since April 2011.