Energy-Efficiency Protocol for Securing the Wireless Sensor Networks

Luu Long, Eunmi Choi

Energy-Efficiency Protocol for Securing the Wireless Sensor Networks

Luu Hoang Long[1], Eunmi Choi[2,*]

[1] Nexcel Solutions, Hochiminh, Vietnam
[2] Department of Information System, Kookmin University, Jeongneung-Dong, Seongbuk-Gu, Seoul, 136-702, Korea
longluu229@gmail.com, emchoi@kookmin.ac.kr

Abstract. Wireless Sensor Networks (WSNs) are used in many application areas because of its characteristics of easy installation and deployment of sensor nodes in any place and any form. In most of cases, WSNs are typically deployed in un-trusted environment so that security of data communication becomes the essential demand. Secure data transfer in sensor networks requires complicated consideration, compared to conventional desktop computers with the limited processing power, storage, bandwidth, and energy. In this paper, we provide a secure protocol for WSNs which is not only trying to achieve all major issues in security, but also satisfies the low-power consumption as well. We provide and develop secure protocols for data communication and algorithmic mechanisms which ensure energy-saving processing to maximize the performance of communication.

Keywords: Wireless Sensor Networks (WSN)1

1 Introduction

A wireless sensor network consists of spatially distributed autonomous sensors to cooperatively monitor physical or environmental conditions, such as temperature, sound, vibration, pressure, motion or pollutants. The development of wireless sensor networks was motivated by military applications such as battlefield surveillance. They are now used in many industrial and civilian application areas, including industrial process monitoring and control, machine health monitoring, environment and habitat monitoring, healthcare applications, home automation, and traffic control.
In wireless sensor networks, sensors usually communicate with each other using a multi hop approach. The biggest problem of sensor networks is power consumption, which is greatly affected by the communication between nodes. To solve this issue:

- *Aggregation points* are introduced to the network. This reduces the total

* Corresponding Author

* This research was supported by Basic Science Research Program through the National Research Foundation of Korea(NRF) funded by the Ministry of Education, Science and Technology(2012-0002774)

number of messages exchanged between nodes and saves some energy. Usually, aggregation points are regular nodes that receive data from neighboring nodes, perform some kind of processing, and then forward the filtered data to the next hop.

- Similar to aggregation points is *clustering*. Sensor nodes are organized into clusters, each cluster having a "cluster head" as the leader. The communication within a cluster must travel through the cluster head, which then is forwarded to a neighboring cluster head until it reaches its destination, the base station.
- Another method for saving energy is *setting the nodes to go idle* (into sleep mode) if they are not needed and wake up when required.

In this paper, we develop an energy-saving and practically efficient sensor-level secure protocol. We divide sensor nodes into a number of clusters and propose the usage of super sensor node as a control center in that cluster. In addition, we establish three types of keys for each sensor node according to each type of communication patterns: an individual key shared with the base station, a cluster key shared with multiple neighboring nodes, and a group key that is shared by all the nodes in the network.

We provide security for sensor networks by including a message authentication code (MAC) with each packet. A MAC can be viewed as a cryptographically secure checksum of a message. We use symmetric encryption algorithms for encrypting the message because the asymmetric encryption is impractical in such the constrained environments as sensor networks. The block cipher algorithm is chosen for encrypting the transmission message.

In our implementation, we try to achieve all security requirements in wireless sensor networks: data confidentiality, data authenticity, data integrity, data freshness, and semantic security. We achieve data confidentiality, data authenticity, data integrity primitives by using encryption mechanism and add MAC with each packet. We achieve data freshness and semantic security primitives through adding an IV into the MAC. For the whole protocol implementation, we use block cipher encryption as a default encryption algorithms due to its high-performance in resource constrained environments such as wireless sensor networks.

The rest of this paper is structured as follow. Section 2 introduces related works of research on security for WSNs. The system overall architecture will be presented in section 3. In section 4, we analyze our secure protocol. We conclude our paper and show future works in section 5.

2 Related Works

There are many protocols for securing data communication in wireless sensor networks. Among those protocols, SPINS [1], TinySec [2] and LEAP [3] emerge with more dominant features.

The SPINS is a suite of security building blocks proposed by Perig et all. It is optimized for resource constrained environments and wireless communication. SPINS has two secure building blocks: SNEP and μTESLA. SNEP provides data confidentiality, two-party data authentication, and data freshness. μTESLA provides authenticated broadcast for severely resource-constrained environments. All cryptographic primitives (i.e. encryption, message authentication code (MAC), hash, random number generator) are constructed out of a single block cipher for code reuse. This, along with the symmetric cryptographic primitives used reduces the overhead on the resource constrained sensor network. However, wireless sensor networks using SPINS can be the victim of DoS attack (Denial of Services) because SPINS rely on a shared counter between the sender and receiver for the block cipher in counter mode. Hence, each sensor node has to store the counter for each its senders.

The TinySec is the first fully-implemented link layer security architecture for wireless sensor networks. A link-layer's security architecture can detect unauthorized packets when they are first injected into the network. TinySec provides the basic security properties of message authentication and integrity (using MAC), message confidentiality (through encryption), semantic security (through an Initialization Vector) and replay protection. TinySec supports two different security options: authenticated encryption (TinySec-AE) and authentication only (TinySec-Auth). With authenticated encryption, TinySec encrypts the data payload and authenticates the packet with a MAC. The MAC is computed over the encrypted data and the packet header. In authentication only mode, TinySec authenticates the entire packet with a MAC, but the data payload is not encrypted.

LEAP (Localized Encryption and Authentication Protocol) is a key management protocol for sensor networks that is designed to support in-network processing, while restricting the security impact of a node compromise to the immediate network neighborhood of the compromised node. The design of the protocol is motivated by the observation that different types of messages exchanged between sensor nodes have different security requirements, and that a single keying mechanism is not suitable for meeting these different security requirements. Hence, LEAP establishes four types of keys for each sensor node – an individual key shared with the base station, a pair-wise key shared with another sensor node, a cluster key shared with multiple neighboring nodes, and a group key that is shared by all the nodes in the network. LEAP also includes an efficient protocol for inter-node traffic authentication based on the use of one-way key chains. The drawback of LEAP is it requires every node has space for storing up to hundreds of bytes of keying materials.

3 Architecture

In this section, we describe our secure protocol design and techniques/mechanisms that are used to build the secure protocol. The overall system architecture will be shown with our keying mechanisms.

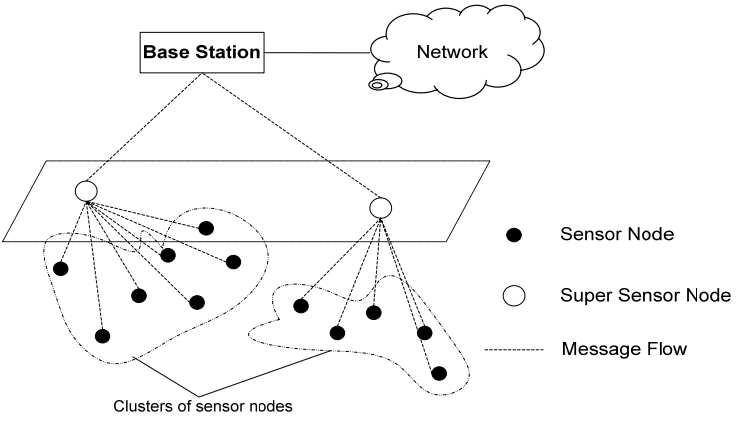

Fig. 1. Overall System Architecture

3.1 Overall System Architecture

Based on the prior research works of sensor secure protocols, we develop an energy-saving and practically efficient sensor-level secure protocol which achieves all the major issues of security in wireless sensor networks such as data confidentiality, data authenticity, data integrity, data freshness, and semantic security. As shown in Figure 1, there are three main actors in our protocols: the base station, the super sensor nodes, and the sensor nodes.

- **Base Station:** communicates with Super Sensor Nodes and Sensor Nodes.
- **Super Sensor Node:** A super sensor node communicates with other super sensor nodes. With this communication, a cluster communicates with other clusters. Also, a super sensor node communicates with sensor nodes in its cluster and the base station.
- **Sensor Node:** A sensor node communicates with other sensor nodes in a cluster through its super sensor node. Particularly, the super sensor node acts as an intermediate node for transmitting the packet sent from a sensor node to another in its cluster. A sensor node communicates with its super sensor node and the base station. A sensor node may send an alert to the base station if it observes any abnormal or unexpected behavior by a neighboring node.

3.2 Keying Mechanisms

In this section, we describe for each kind of nodes in our secure protocol. We assume that only the base station can generate the keys for the whole secure protocol. It means any key setup process for any kind of node in the network has to pass through

the base station. In succession, we will consider the keys setup for the super sensor node and the sensor node.

In case of adding a new super sensor node S into a sensor network, the global key and individual key are pre-loaded into to S before its deployment. S is a new super sensor node, so that S does not need to care about the list of keys/counters shared with each sensor node in the cluster at this moment.

The things that S should do now are establishing the cluster key and list of keys shared with other super sensor nodes. These keys can be established through 3 steps:

- Step 1: Neighbors discovery. In this step, S broadcasts a message to discover all super sensor nodes near by it. The broadcasting message contains the ID of S. After receiving the message from S, S's neighbors response with their ID. S lists out its neighbors' ID in a list called LNID.
- Step 2: Generate Keys. S sends LNID to the base station and asks the base station for generating keys. The packet that is sent in this step is encrypted using the individual key shared between S and the base station.
- Step 3: Deliver Keys. After generating keys, the base station sends appropriate keys to S and S's neighbors. The transmission packets in this step are encrypted using the individual key shared between the base station and each node.

In case of adding a new sensor node N into a sensor network, the global key and individual key are pre-loaded into to S before its deployment. The cluster key can be transmitted after N successfully established a secret key with its super sensor node. The secret key establishing process gets through 3 steps which are described as follow:

- Step 1: Super Sensor Nodes Discovery. In this step, N broadcasts a message to discover all super sensor nodes near by it. The broadcasting message contains the ID of N. After receiving the message from N, only the super sensor nodes (which are near by N) response with their ID.
- Step 2: Choose Cluster to Join. From the super sensor nodes' IDs from the previous step, N chooses one cluster to join (in this case, a cluster is represented by n super sensor node's ID).
- Step 3: Secret Key Generation. N sends the chosen cluster's ID to the base station and asks for the secret key generation. The packet that is sent in this step is encrypted using the individual key shared between S and the base station.
- Step 4: Keys Delivery. After generating keys, the base station sends appropriate keys to N and N's super sensor node. The transmission packets in this step are encrypted using the individual key shared between the base station and each node.

3.3 Packet Format

Based on above consideration, we create the packet format for our secure protocol as follows.

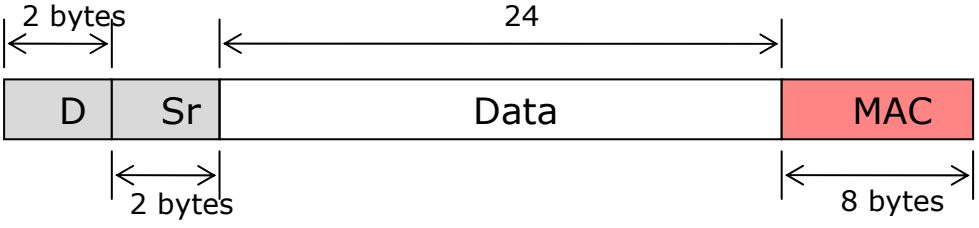

Fig. 2. Packet Format in Secure Protocol

As you can see in Fig. 2, the transmission packet in our secure protocol includes four parts: Des||Src||Data||MAC. Where:

- Des is the ID of destination node and it will not be encrypted. The size of Des is 2 bytes.
- Src is the ID of node that sends the packet and it will not be encrypted. The size of Src is 2 bytes.
- Data is the information that be sent. In our secure protocol, this information will be encrypted by key K_{enr} before sending. The output's size of skipjack algorithm is 8 bytes so that we chose the size of data is 24 bytes which is the multiple of 8 bytes.
- MAC is security checksum for the whole packet. A key K_{mac} is used to create MAC. We chose the size of MAC is 8 bytes to decrease the size of whole packet as much as possible.

For example, A and B are principals, such as communicating nodes. The complete message that A sends to B is:

$$A \rightarrow B: (B\_ID \parallel A\_ID \parallel \{Data\}K_{enr} \parallel \{M\}K_{mac})$$

Now we will describe what are included in M (in previous state) before it is encrypted by the Kmac. Assume that in the hexa-mode:

- Two bytes of A's ID are A_1 and A_2,
- Two bytes of B's ID are B_1 and B_2.
- Two bytes of the counter C shared between A and B are C_1 and C_2,
- The first eight bytes of $\{Data\}K_{enr}$ are $D_1, D_2, \ldots, D_7, D_8$.

As MAC acts as the checksum value for the transferring packet, MAC should contain all parts of the packet's information. Particularly, MAC contains A_1& A_2, B_1& B_2, and C_1& C_2. There are two more bytes for representing the data; and the last

two bytes of data D_7&D_8 are chosen. Then, the content of M before it is encrypted by key K_{mac} should be:

Fig. 3. The structure of MAC

Finally, we describe the mechanism that is used to create K_{enr} and K_{mac}. As discussed above, each node shares with the base station a key called individual key. Moreover, each sensor node shares with its super sensor node a secret key. These kinds of keys are used for securing the node-to-node transmission packet. In this section, we call them "symmetric key" so that it is easier to describe our mechanism.

We surveyed of cipher block chaining algorithms for software implementation on embedded microcontrollers and found that RC5 and Skipjack are most appropriate. RC5 is slightly faster Skipjack. However, for good performance, RC5 requires the key schedule to be pre-computed, which uses 104 extra bytes of RAM per key [5]. Because of these drawbacks, the default block cipher in our protocol is Skipjack.

Assume that node A shares with node B a symmetric key KS. The size of K_S is 10 bytes. The mechanism of creating K_{enr} and K_{mac} from Ks is shown in the figure below:

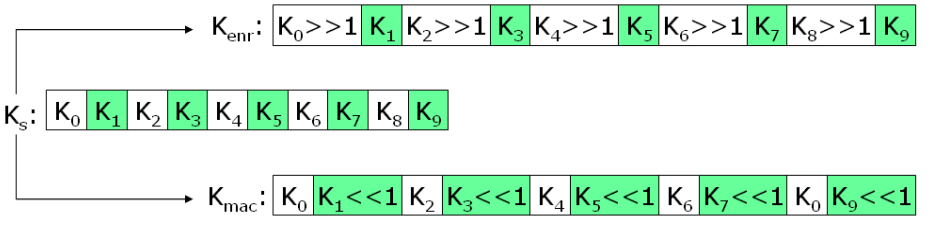

Fig. 4. The mechanism of generating K_{enr} and K_{mac}

As you can see in Fig. 4, all even bytes in 9 bytes of symmetric key will be right-shift one step to create the K_{enr} and all odd bytes in 9 bytes of symmetric key will be left-shift one step in order to create the K_{mac}. The reason that we used shifting mechanism is this approach is very fast so that we can save the sensor's energy.

4 Experimental Results

We made the simulation for assessing the performance of our protocol comparing to Typical Protocol and LEAP. In this simulation, we sequentially added sensors to

create sensor networks. The number of sensors varies from 100 to 600 sensors and compute:

- The time for creating a sensor network and establishing keys.
- The number of packets for sending and receiving of each sensor

We implemented the simulation using C++ programming language. The resources we chose to perform comparison tests are: Windows XP; Intel® Core™ 2 Duo T7300 2.00 GHz, 200 GHz; RAM 2.00 GB. The experimental results are visually shown below.

In Figure 5, you can see that when the number of sensors increases, the time for creating the sensor network and establishing keys of our secure protocol is much better than in case of typical protocol. When the sensor's density increases, the number of neighbors of each sensor will be increased. In case of the typical protocol, the more neighbors each sensor has the more key generating step and delivering key step must be done at that sensor. That is the reason why the times are significantly increased in case of the typical protocol. In our protocol, the sensors just need to establish the key with its directly super sensor and the key generation process is done at the base station so that our secure protocol's times are slightly increased.

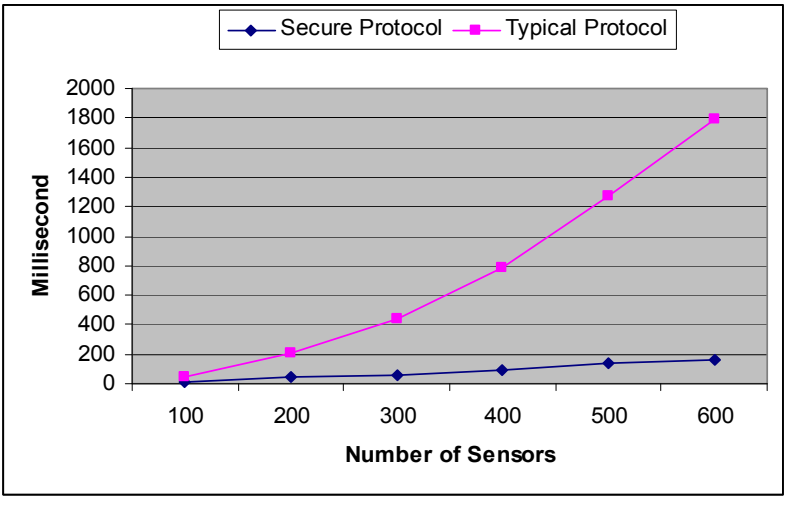

Fig. 5. Time for creating a sensor network and establishing keys

We run the simulation and collect the number of packets each sensor sent and received to make comparison between our secure protocol and LEAP. The results are shown in Figure 6.

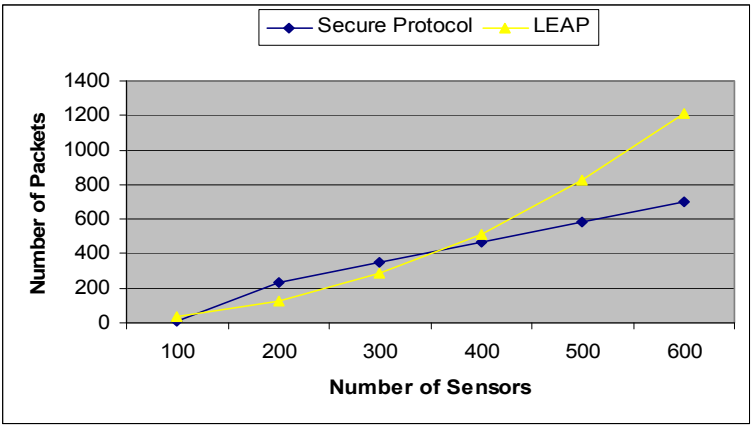

Fig. 6. The number of packets received to create sensor networks

The above figure shows that with the low sensor density, the number of packets each sensor received in LEAP is lower than in case of our secure protocol. However, with the high sensor density (this case is commonly in wireless sensor networks), our secure protocol is much better. In our secure protocol, we divided the sensing area into number of clusters. Each cluster is controlled by one super sensor. When adding a new sensor, this sensor just needs to communicate with its super sensor. This approach helps to reduce the number of packets received to create sensor networks.

5 Conclusion

In summary, this paper describes an energy-saving and practically efficient sensor-level secure protocol in wireless sensor networks. We establish three types of keys for each sensor node according to each type of communication patterns: an individual key shared with the base station, a cluster key shared with multiple neighboring nodes, and a group key that is shared by all the nodes in the network. Each key for each type of communication patterns provides more security for our protocol.

We also come up with a simulation to analysis the performance of our secure protocol. Through the simulation, we experience our protocol by comparing the time for creating a sensor network and establishing keys to a typical protocol; and comparing the number of packets sent and received to LEAP. The experimental results show that our secure protocol performance is potential especially when the sensor density of the sensor network is high.

References

1. Adrian Perrig, Robert Szewczyk, Victor Wen, David Culler, J. D. Tygar. SPINS: Security Protocols for Sensor Networks. In The Seventh Annual International Conference on Mobile Computing and Networking (MobiCom 2001), 2001.
2. Chris Karlof, Naveen Sastry, David Wagner. TinySec: A Link Layer Security Architecture for Wireless Sensor Networks. ACM SenSys 2004, November 3-5, 2004.
3. Sencun Zhu, Sanjeev Setia, Sushil Jajodia. LEAP: Efficient Security Mechanisms for Large-Scale Distributed Sensor Networks. In The Proceedings of the 10th ACM conference on Computer and communications security, 2003.
4. Mayank Saraogi . Security in Wireless Sensor Networks. In ACM SenSys, 2004.
5. Lander Casado, Philippas Tsigas. ContikiSec: A Secure Network Layer for Wireless Sensor Networks under the Contiki Operating System. 'Identity and Privacy in the Internet Age" book, Wednesday, September 30, 2009.
6. Yee Wei Law, Jeroen Doumen, and Pieter Hartel. Survey and Benchmark of Block Ciphers forWireless Sensor Networks. ACM Transactions on Sensor Networks (TOSN), Volume 2 , Issue 1, Pages: 65 - 93 (February 2006)
7. Wang, S. Liu, K.Z. Hu, F.P. Simulation of Wireless Sensor Networks Localization with OMNeT. Mobile Technology, Applications and Systems, 2005 2nd International Conference, Pages: 1-6 (15-17 Nov. 2005).
8. R. L. Rivest, M.J.B. Robshaw, R. Sidney, and Y.L. Yin, "The RC6 Block Cipher", AES submission, Jun 1998.
9. Adrian Perrig, Ran Canetti, Dawn Song, and J. D. Tygar. Efficient and secure source authentication for multicast. In Network and Distributed System Security Symposium, NDSS '01, February 2001.
10. Mihir Bellare, Joe Kilian, and Phillip Rogaway. The security of the cipher blocks chaining message authentication code. Journal of Computer and System Sciences, 61(3):362-399, December 2000.
11. D.W. Carman, P.S. Kruus and B.J.Matt, Constraints and approaches for distributed sensor network security, NAI Labs Technical Report No. 00010 (2002).
12. R.L. Rivest. The RC5 encryption algorithm, in: Workshop on Fast Software Encryption (1995) pp. 86.96.
13. F. Stajano and R. Anderson, The resurrecting duckling: Security issues for ad-hoc wireless networks, in: International Workshop on Security Protocols (1999).
14. J.Douceur. The Sybil Attack. In First International Workshop on Peer-to-Peer Systems (IPTPS'02), 2002.
15. C.Karlof and D.Wagner. Secure Routing in Sensor Networks: Attacks and Counter measures. To appear in Proc. of First IEEE Workshop on Sensor Network Protocols and Applications, May 2003.