

Security Analysis of Hybrid Security Mechanisms for Heterogeneous Sensor Networks

Mangal Sain, Ki Hwan Kim and Hoon Jae Lee

Department of Information Engineering, Dongseo University, Busan, South Korea

mangalsain1@gmail.com, gend90@gmail.com, hjlee@dongseo.ac.kr

Abstract— An efficient hybrid security mechanism (called LIGER) for heterogeneous sensor networks (HSNs) was presented by Traynor et al. LIGER is composed of two subschemes: LION and TIGER. However, this paper presents the LIGER has some designs and security flaws, and cannot be advocated for the real-time critical applications. We have identified that LION and TIGER are vulnerable to the impersonation attacks, gateway-bypass attacks, denial-of-service attacks, and man-in-the-middle attacks. Moreover, both schemes do not care about the dynamic session key which is an indispensable security requirement for the real-time heterogeneous sensor networks.

Keywords— Authentication, session-key establishment, gateway bypass attack, man-in-the-middle attack

I. INTRODUCTION

Heterogeneous sensor networks (HSNs) are rapidly growing in 21st century human life's, and are deploying everywhere in many real-time applications (e.g., homeland security, hydro-plant, agriculture and structural monitoring, etc.) [1][2]. HSNs consist of L1 nodes and L2 nodes [3]. An L1 node has low battery power, less computation power, low bandwidth and less memory; whereas, an L2 node (gateway) has high power, high computational, high bandwidth, and high memory. L1 node senses the environment data and forwards to the L2 node [3]. The ubiquitous nature of wireless sensor networks (WSNs) makes sensor data available to users anytime and anywhere, as needed. However, security is always a prime concern for a real WSN application success. With the fact of resource-constraint (less memory, low bandwidth, power) nature of L1 nodes, traditional security mechanisms are not applicable to the sensor nodes. Moreover, the wireless communication nature of sensor nodes makes application more vulnerable to security attacks. Thus security becomes a challenging task for the real-time WSNs application.

However, for providing security to a real WSN application, a key management plays an important role that can support the efficient security functionalities (e.g., authentication, integrity, confidentiality, dynamic session key establishment, etc.) to the applications.

In this regard, Traynor et al [3] proposed an efficient hybrid security mechanisms (called LIGER) for HSNs.

LIGER is composed of two sub-schemes: LION and TIGER. In both schemes, numbers of randomly selected keys (k) are preloaded into the L1 node (i.e., 30 keys), and into the L2 node (711 keys) before the nodes deployment into the terrain. LION scheme is a standalone key management which establishes a session key in two different scenarios: (i) an L1 has a direct shared key with the L2 node; and (ii) when the two L1 nodes are not having a shared key, in this case, both the L1 nodes establish a session-key through L2 node. Whereas, TIGER is based on key distribution center (KDC) and performs probabilistic authentication (between L1-to-L1 and L2-to-L1) and provides session keys with the least privilege.

In this paper, we have identified that LIGER is not a secure mechanism for the real-time HSN application and susceptible to impersonation attack, gateway-bypass attack, denial-of-service attack and man-in-the-middle attack. We will show that the security of LIGER suite is completely broken. In addition, LIGER does not care about the key revocation service (if, an L1 node is compromised), and no new node addition is allowed, so that it can maintain the node (L1) scalability for the applications.

The rest of the paper is organized as: Section-II reviews the LIGER. In Section-III, we present the impersonation attack, gateway-bypass attack, denial-of-service attack and man-in-the-middle attack on LIGER; and Section-IV discuss other security issues of LIGER. Finally, in Section-V concluding comments are given.

II. REVIEW OF LIGER

This section reviews the LIGER which is composed of two sub-schemes: LION and TIGER. Note due to the limited page restrictions we skip the key distribution processes, for details reader may refer to [3]. The outlines of the protocol executions are as follows: L1 and L2 nodes are preloaded with a random set of keys from an offline common key pool. The mapping of keys (i.e., sensor nodes keys) is kept in the KDC server. Upon distributing keys to L1 nodes, it is assumed that the L1 nodes are randomly dropped using an airplane into the terrain. However, LION operated in a standalone mode (without KDC) and establishes a shared session key. TIGER leverages on the presence of a KDC (with KDC). The details of LION and TIGER are shown below. The notations used in this paper are given in Table I, which is directly adopted from [3].

TABLE 1. NOTATIONS AND DESCRIPTIONS

Notation	Descriptions
A, B	A and B are principles entities (i.e., L1 nodes)
$ID_{A0}, \dots, ID_{AK-1}$	Key identifiers corresponding to A's keys
K_A	Secret key known by node A
$K_{A,B}$	Shared session key between A and B
K_{A_AUTH}	An authentication key for node A
L1 and L2 A	sensor node(L1) and a gateway (L2)
$MAC(K_A, R S)$	Message authentication code of R and S using K_A
MAP_A	Bitmapping of A's keys
$E_K \{S\}$	S is encrypted using key K
N	A nonce

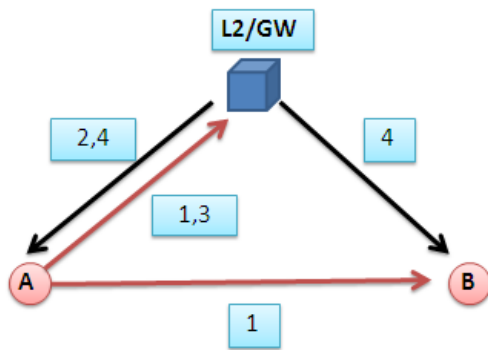
A. LION: Standalone Key Management: To establish a shared key, an L1 node (A) learns its neighbouring nodes through the *Hello* messages and then establishes shared keys with them by sending the message (i.e., A, N, $ID_{A0}, \dots, ID_{AK-1}$). Here, the $ID_{A0}, \dots, ID_{AK-1}$ are the keys identifiers. If a node (L2) has a shared key with the L1 node then they establish a shared key by following messages exchanges:

1. $A \rightarrow * : A, N, ID_{A0}, \dots, ID_{AK-1}$
2. $L2 \rightarrow A : A, L2, N, ID_{Ai}, E_{K_{Ai}}\{ID_{Ai}, L2, N\}$

L1 node maintains a list of its neighbouring nodes that do and do not have shared keys. Now, L1 establishes keys with its neighbouring nodes that do not have shared key with L1 (A). To accomplish this, it (L1) uses the neighbouring node (e.g., L2) that has already shared key, as follows:

3. $A \rightarrow L2 : A, B, N$
4. $L2 \rightarrow * : A, B, N, E_{K_{A,L2}}\{K_{A,B}, N\}, E_{K_{B,L2}}\{K_{A,B}, N\}$

Here $K_{A,B}$ is a session key between the A and B nodes, and will be deleted as soon as an end-to-end communication is setup. The flow of LION messages are depicted in Fig.1.

**Figure 1.** Flow of LION messages

B. TIGER: KDC-Based Key Management: This sub-scheme exploits the advantage of a KDC, which is available through the L2 node. In TIGER, L1 nodes perform probabilistic authentication using a temporary authenticator key (K_{A_Auth}). Here, K_{A_Auth} is generated using an XOR operation on some q

keys which are selected from its pre-loaded key-set. Since, the KDC knows all preloaded keys (k) of nodes, it can easily compute the K_{A_Auth} and authenticate to the L1 node. TIGER is divided into two scenarios: L1-L1 authentication and key establishment. **B.1) L1-L1 Authentication and Key Establishment:** An L1 (A) node requests to establish a secure session key with its neighboring node L1 (B). To accomplish this task, node A generates a token, forward to the node B, as follows:

1. $A \rightarrow B : A, B, N, MAP_A, MAC(K_{A_Auth}, A|B|N|MAP_A)$

Here MAP_A is a sorted bitmap of keys (q) that is used to generate an authenticator key (K_{A_Auth}).

Upon receiving A's token, B includes its own token to the A's token and sends to the L2, as follows:

2. $B \rightarrow L2 : A, B, N, MAP_B, MAC(K_{B_Auth}, A|B|N|MAP_B), MAP_A, MAC(K_{A_Auth}, A|B|N|MAP_A)$

Now, L2 forwards the B's message to the KDC:

3. $L2 \rightarrow KDC$: Send message 2 to the KDC.

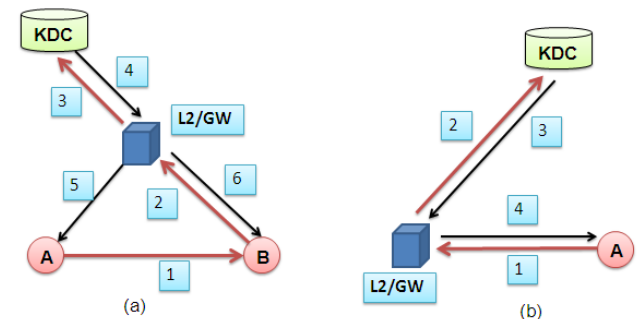
After receiving the message 3 (from L2), KDC verifies the validity of node A and B by generating the authenticator keys (i.e. K_{A_Auth} and K_{B_Auth}) using the sorted bitmaps of identifiers corresponding to keys that are used to create each token. If both the tokens are verified, then KDC computes a message that having the following: (i) a session key which is encrypted with newly chosen authenticator key for both the A and B; and (ii) the message also contains a new bitmap (corresponding to each authenticator keys), which is also used to sign the session keys. Now, KDC responds to the L2 node, as follows:

4. $KDC \rightarrow L2 : A, B, N, MAP_{A'}, E_{K_{A'_Auth}}\{K_{A,B}, N\}, MAP_{B'}, E_{K_{B'_Auth}}\{K_{A,B}, N\}, MAC(K_{A,B}, A|B|N|K_{A,B}|MAP_{A'}|MAP_{B'})$

Upon receiving the message 4, L2 verifies the MAC and forward the session keys to the node A and B, as follows:

5. $L2 \rightarrow A : A, B, N, MAP_{A'}, E_{K_{A'_Auth}}\{K_{A,B}, N\}, MAC(K_{A,B}, A|B|N|K_{A,B}|MAP_{A'})$

6. $L2 \rightarrow B : A, B, N, MAP_{B'}, E_{K_{B'_Auth}}\{K_{A,B}, N\}, MAC(K_{A,B}, A|B|N|K_{A,B}|MAP_{B'})$
- The flow of L1-L1 is shown in Fig. 2(a).

**Figure 2.** Flow of TIGER messages ((a) L1-L1 and (b) L2-L1)

B.2) L2-L1 Authentication and Key Establishment: L2 broadcasts a *Hello* message and shows its presence to the L1 nodes. Upon receiving the *Hello* message, L1 (A) node generates a token as in L1-L1 (above) and sends to L2/GW node, as follows:

1. $A \rightarrow L2: A, L2, N, MAP_A, MAC(KA_{Auth}, A|L2|N|MAP_A)$

After receiving A'_{s} message, L2 forwards the message to the KDC:

2. $L2 \rightarrow KDC$: Send message 1 to the KDC.

Now, KDC verifies the MAC, if verified, and then KDC generates a message and sends to the L2 node, as follows:

3. $KDC \rightarrow L2: A, N, MAP_A', EKDC, L2\{KA_{L2}, N, EK_{A_{Auth}}\{KA_{L2}, N\}\}$

Next, L2 forwards the session key to the node A by generating a message:

4. $L2 \rightarrow A: A, N, MAP_A', EA'_{Auth}\{KA_{L2}, N\}, MAC(KA_{L2}, A|N|MAP_A', EA'_{Auth}\{KA_{L2}, N\})$. The flow of L2-L1 is shown in Fig.2 (b).

III. SECURITY ANALYSIS OF LIGER

In this section, we will demonstrate that LION is vulnerable to impersonation attack, gateway-bypass attack, and denial-of-service attack. TIGER is vulnerable to impersonation, man-in-the-middle attack. Furthermore, we will show that LIGER is lacking from other indispensable security services.

In order to evaluate the security of LIGER, we are assuming that an attacker has following capabilities: (i) since sensors (L1) are not tamper-proofed, an attacker can easily capture few nodes from the terrain and dig out all the preloaded keys and cryptographic systems [4] [5]. In addition, Becher et al have shown real-world physical attacks on WSNs [6]. Moreover, authors claimed that node capture attacks can have major impact on WSN application [6]. (ii) An attacker can monitor the wireless channels and have full control over them. Traynor et al assumed that the L1 nodes are deployed randomly into the terrain using an airplane, and each L1 is preloaded with 30 keys with its keys identifier.

A. LION analysis

Since LION scheme does not care about the key revocation for a compromised node; it is practical that compromised nodes can join the network. Assuming that if an L1 node is compromised and some later time it can do the followings: (i) it (compromised node) can easily impersonate as a legitimate L1 node; and (ii) it can bypass the gateway (i.e., L2) node and create its own sub-network. Moreover, LION is vulnerable to the denial-of-service attack.

L1 impersonation attack: The compromised node ($L1_c$) can impersonate as a legitimate L1, as follows. $L1_c$ node creates a fake identity (A_c), nonce (N_c), and sends a message and establishes a secret key with the L2 node, directly

1. $A_c \rightarrow *: A_c, N_c, ID_{A0}, \dots, ID_{AK-1}$

2. $L2 \rightarrow A_c: A_c, L2, N_c, ID_{Ai}, EK_{Ai}\{ID_{Ai}, L2, N_c\}$

We have seen that the A_c has established a secure communication easily with the L2. Now, we will see that how an A_c can establish a session key (using an L2) with other

legitimate neighbors. To accomplish this task, $L1_c$ node creates a fake identity (A_c), nonce (N_c), and sends a message to the L2 (GW) node, and establish a session key, as follows:

3. $A_c \rightarrow L2: A_c, B, N_c$

4. $L2 \rightarrow *: A_c, B, N_c, EK_{Ac, L2}\{KA_{Ac, B}, N_c\}, EK_{B, L2}\{KA_{Ac, B}, N_c\}$

By doing so compromised nodes (A_c) can easily impersonate a legitimate L1 node.

Gateway-bypass attack: In this attack, a malicious node can establish its own sub-network without involving the legitimate gateway (known as gateway-bypass). Now we will see how LION is vulnerable to the gateway-bypass attack. As we know, A_c (compromised) node has 30 keys, so it possible that many neighbouring nodes having shared keys with A_c . Thus, to perform this attack, A_c can broadcast a *Hello* message to its neighbouring nodes and show own presence as a legitimate $L2_c$ node (GW). Thereafter, L1 sensor nodes respond, as follows:

1. $A \rightarrow *: A, N, ID_{A0}, \dots, ID_{AK-1}$

2. $L2_c \rightarrow A: A, L2_c, N, ID_{Ai}, EK_{Ai}\{ID_{Ai}, L2_c, N\}$

Using message 1, $L2_c$ and L1 nodes can maintains their list of its neighboring nodes that do and do not have shared keys with L1 nodes. Now, if A ($L1$) node wants to communicate to the B ($L1$) node which are apart but they are having shared keys with $L2_c$. In this case, $L2_c$ can easily comprise its own sub-network, as follows:

3. $A \rightarrow L2_c: A, B, N$

4. $L2_c \rightarrow *: A, B, N, EK_{Ac, L2_c}\{KA_{Ac, B}, N\}, EK_{B, L2_c}\{KA_{Ac, B}, N\}$

By doing this, a compromised (A_c) node can bypass the legitimate gateway and setup its own sub-network and control the real-time application.

Denial-of-service attack: In LION, many of message parameters (such as identities, nonce) are open in the air (refer section-II, A. LION standalone; and messages are 1, 2, 3, 4). Therefore, any attacker easily intercepts the wireless message and alters them (i.e., fake identities and nonce), accordingly; and then easily forwards again and again the altered message to the L2 node. In this manner, L2 node will always keep busy with the altered messages, and will not perform its tasks, as expected. Thus, LION is vulnerable to the denial-of-service attack.

B. TIGER analysis

The same attack model (i.e., compromised L1 node) is also applicable to the TIGER which is based on KDC. TIGER scheme is entirely depend on the availability of KDC, since it contains information of all the preloaded keys (i.e., keys with their identifiers) of sensor nodes. **B.1) Analysis of L1-L1 authentication and key establishment:**

L1 impersonation attack: In TIGER, a compromised node (A_c) can impersonate a legal node and can easily join the network. Moreover, it can establish a secure and authenticated session with other non-compromised node (B). To achieve this attack, A_c can compute a fake MAC_c , and map some of keys (MAP_c) from its own key-pool. Now A_c can generate a token, send to the B node. Then B adds its own token into the A_c 's token and forwards to the KDC. Since KDC has all the keys, it can easily verify both the A_c and B by computing the corresponding tokens. Thereafter, KDC sends session keys to A_c and B through $L2$ node. The flow of impersonation attack is shown below:

1. $A_c \rightarrow B: A_c, B, N_c, MAP_{A_c}, MAC_c(K_{A_c, Auth}, A_c | B | N_c | MAP_{A_c})$
2. $B \rightarrow L2: A_c, B, N_c, MAP_B, MAC(K_{B, Auth}, A_c | B | N_c | MAP_B), MAP_{A_c}, MAC(K_{A_c, Auth}, A_c | B | N_c | MAP_{A_c})$.
3. $L2 \rightarrow KDC$: forward message 2 to the KDC.
4. $KDC \rightarrow L2: A_c, B, N_c, MAP_{A_c'}, EK_{A_c'}_{Auth}\{K_{A_c, B}, N_c\}, MAP_{B'}, EK_{B'}_{Auth}\{K_{A_c, B}, N_c\}, MAC(K_{A_c, B}, A_c | B | N_c | K_{A_c, B} | MAP_{A_c'} | MAP_{B'})$.
5. $L2 \rightarrow A_c: A_c, B, N_c, MAP_{A_c'}, EK_{A_c'}_{Auth}\{K_{A_c, B}, N_c\}, MAC(K_{A_c, B}, A_c | B | N_c | K_{A_c, B} | MAP_{A_c'})$.
6. $L2 \rightarrow B: A_c, B, N_c, MAP_{B'}, EK_{B'}_{Auth}\{K_{A_c, B}, N_c\}, MAC(K_{A_c, B}, A_c | B | N_c | K_{A_c, B} | MAP_{B'})$.

By performing such methods, an attacker can easily impersonate an $L1$ node and join the network.

Man-in-the-middle attack (MITM): Since TIGER does not allow the mutual authentication between all entities (i.e., $L1$, $L2$, and KDC), therefore, a MITM can easily involve (as an $L2_c$) and control the sensor network. Suppose an attacker captures few $L1$ nodes (e.g., 10 nodes) and extract all the relevant material, as he needed. He can make use of own gateway ($L2_c$) and load the all compromised keys (i.e. $30 \times 10 = 300$) and other relevant material, such cryptosystems, nonce generator, etc. Now, $L2_c$ has enough keys and can take part as a MITM, as follow.

1. $A \rightarrow B: A, B, N, MAP_A, MAC(K_{A, Auth}, A | B | N | MAP_A)$
2. $B \rightarrow L2_c: A, B, N, MAP_B, MAC(K_{B, Auth}, A | B | N | MAP_B), MAP_A, MAC(K_{A, Auth}, A | B | N | MAP_A)$.
3. $L2_c \rightarrow KDC$: Send message 2 to the KDC.
4. $KDC \rightarrow L2_c: A, B, N, MAP_{A'}, EK_{A'}_{Auth}\{K_{A, B}, N\}, MAP_{B'}, EK_{B'}_{Auth}\{K_{A, B}, N\}, MAC(K_{A, B}, A | B | N | K_{A, B} | MAP_{A'} | MAP_{B'})$.

Upon receiving message 4, $L2_c$ can drop many original message parameters, such as $MAP_{A'}, EK_{A'}_{Auth}\{K_{A, B}, N\}, EK_{B'}_{Auth}\{K_{A, B}, N\}, MAP_{B'}, MAC(K_{A, B}, A | B | N | K_{A, B} | MAP_{A'} | MAP_{B'})$. And it computes its fake messages and forwards to the node A and B , as follows:

5. $L2_c \rightarrow A: A, B, N, MAP_{A'}, EK_{A'}_{Auth}\{K_{A, B}, N\}, MAC_c(K_{A, B}, A | B | N | K_{A, B} | MAP_{A'})$.
6. $L2_c \rightarrow B: A, B, N, MAP_{B'}, EK_{B'}_{Auth}\{K_{A, B}, N\}, MAC_c(K_{A, B}, A | B | N | K_{A, B} | MAP_{B'})$.

After verifying the $L2_c$ messages, A and B can easily believe that they are connecting with a real $L2$, whereas in reality, it is a fake $L2_c$.

B.1) Analysis of $L2$ - $L1$ authentication and key establishment:

L1 impersonation attack: The $L1$ node can be impersonated as we have demonstrated above in the $L1$ - $L1$ authentication and key establishment. In this attack, an A_c forwards its message to the $L2$ node, and then $L2$ sends A_c 's message to the KDC. Since, KDC has all the keys information; it can be easily verified to the A_c and responds with the new session key. The procedure of $L1$ impersonation attack is as follows:

1. $A_c \rightarrow L2: A_c, L2, N_c, MAP_{A_c}, MAC(K_{A_c, Auth}, A_c | L2 | N_c | MAP_{A_c})$
2. $L2 \rightarrow KDC$: Send message 1 to the KDC.
3. $KDC \rightarrow L2: A_c, N_c, MAP_{A_c'}, EK_{KDC, L2}\{K_{A_c, L2}, N_c, EK_{A_c, Auth}\{K_{A_c, L2}, N_c\}\}$
4. $L2 \rightarrow A_c: A_c, N_c, MAP_{A_c'}, E_{A_c}_{Auth}\{K_{A_c, L2}, N_c\}, MAC(K_{A_c, L2}, A_c | N_c | MAP_{A_c'}, E_{A_c}_{Auth}\{K_{A_c, L2}, N_c\})$.

Man-in-the-middle attack: Likewise to the above MITM attack model (i.e., $L1$ - $L1$), the $L2$ - $L1$ scheme is also susceptible to the MITM. To accomplish this attack, the fake $L2_c$ broadcasts *Hello* message and shows own presence to the $L1$ node (A). Then, node A sends its message to the $L2_c$, it forwards the A 's message to the KDC. Upon receiving the $L2_c$'s message, KDC verifies the message and responds to the $L2_c$ that contains the session key for the node A . The flow of MITM is as follows:

1. $A \rightarrow L2_c: A, L2_c, N, MAP_A, MAC(K_{A, Auth}, A | L2_c | N | MAP_A)$
2. $L2_c \rightarrow KDC$: Send message 1 to the KDC.
3. $KDC \rightarrow L2_c: A, N, MAP_{A'}, EK_{KDC, L2_c}\{K_{A, L2_c}, N, EK_{A, Auth}\{K_{A, L2_c}, N\}\}$

Upon receiving the message from KDC, $L2_c$ can simply drop few message parameters, such as $EK_{KDC, L2_c}\{K_{A, L2_c}, N, EK_{A, Auth}\{K_{A, L2_c}, N\}\}$ and can compute its own message, as follows:

4. $L2_c \rightarrow A: A, N, MAP_{A'}, E_{A'}_{Auth}\{K_{A'}, L2_c, N\}, MAC(K_{A'}, L2_c, A | N | MAP_{A'}, E_{A'}_{Auth}\{K_{A'}, L2_c, N\})$.

By doing so, a MITM can easily involve and control the network, accordingly.

Moreover in TIGER ($L2$ - $L2$ communication link), during our analysis we have found that, since $L1$ has direct key sharing with the $L2$ node then both the $L1$ and $L2$ can establish a session key without involving the KDC. However, performing of additional steps (2 and 3) increases the computation and communication cost.

IV. OTHER SECURITY SERVICES

This section discusses the other security services issues of Traynor et al. Since $L1$ nodes are deployed in an open

environment, it is practical that any adversary can physically capture the nodes and controls the critical applications (as we have demonstrated above). However, authors have suggested that the intrusion detection mechanisms are available for ad-hoc networks and can be easily applied on LIGER, as well. Indeed, we are agreed with authors, but key revocation mechanism is still required for non-compromised nodes, so that a compromised node cannot impersonate them. For the sake of simplicity, supposed that L1 nodes are deployed in a forest for the fire monitoring if key revocation mechanism is not provided then any compromised node can control the application and generate the fake fire alarms and warnings to the rescue teams. However authors did not care about the key revocation which is an important requirement for HSNs. Another issue is that, authors did not care about the new node addition mechanism, resultant, LIGER is not a scalable suite for the real-time applications.

More importantly, mutual authentication and dynamic session keys are essential security requirements for the applications. Whereas LIGER performs the probabilistic authentication and establishes static session keys with the least privilege, thus not suitable to the real-time HSNs.

V. CONCLUSION

In this paper, we have pointed out that LIGER is insecure under the impersonation attack, gateway-bypass attack, denial of service attack, and man-in-the-middle attack. In addition, many of important security services are not considered, such as, key revocation, new node addition, and mutual authentication and dynamic session keys. Resultant, LIGER is not suitable for the real-time critical applications.

ACKNOWLEDGMENT

This research was financially supported by the Ministry of Trade, Industry and Energy(MOTIE) of the Republic of Korea and Korea Institute for Advancement of Technology(KIAT) through the Regional Specialized Industry Development Program(Grant Number : R0005598).

This work is supported by Basic Science Research Program through the National Research Foundation of Korea (NRF) funded by the Ministry of Education, Science and Technology. (Grant number: NRF-2011-0023076).

REFERENCES

- [1] G. Zhao, "Wireless Sensor Networks for Industrial Process Monitoring and Control: A Survey," *Network Protocols and Algorithms*, Vol. 3, No. 1, 2011, pp. 46-63.
- [2] A. Koubaa, M. Alves, "A Two-Tiered Architecture for Real-Time Communication in Large-Scale Wireless Sensor Networks: Research Challenges," Published in WIP proceeding of the 17th Euromicro Conference on Real-Time System, Palma de Mallorca, Spain, 2005.
- [3] P. Traynor, R. Kumar, H. Choi, G. Cao, S. Zhu, and T. L. Porta, "Efficient Hybrid Security Mechanisms for Heterogeneous Sensor Networks," *IEEE Transactions on Mobile Computing*, Vol.6, No. 6, June 2007, pp. 663-677.
- [4] W. Wang, L. Hu, and Y. Li, "Security Analysis of a Dynamic Program Update Protocol for Wireless Sensor Networks," *IEEE Communication Letters*, Vol. 14, No. 8, August 2010, pp.782-784.
- [5] X. Chen, K. Makki, K. Yen, and N. Pissinou, "Sensor Network Security: A Survey," *IEEE Communication Surveys & Tutorials*, vol.11, No. 2, second quarter 2009, pp. 52-73.

- [6] A. Becher, Z. Benenson, and M. Dornseif, "Tampering with Motes: Real-World Physical Attacks on Wireless Sensor Networks," *Proceeding of the 3rd International Conference on Security in Pervasive Computing (SPC'06)*, pp.104-118.



and a TPC member in several international conferences

Mangal Sain received the M.Sc. degree in computer application from India in 2003 and the Ph.D. degree in computer science in 2011. Since 2012, he has been an Assistant Professor with the Department of Computer Engineering, Dongseo University, South Korea. His research interest includes wireless sensor network, cloud computing, Internet of Things, embedded systems, and middleware. He has authored over 30 international publications including journals and international conferences. He is a member of TIIS



Ki Hwan Kim

2013: BS at Dongseo University, Republic of Korea
2015: MS at Dongseo University, Republic of Korea
2015 ~ current: doctor's course Dongseo University, Republic of Korea
Research Interests: Wireless Sensor Networks, Cryptography and Network Security, Side Channel Analysis



ubiquitous sensor network/radio frequency identification security.

Hoon-Jae Lee received his BS, MS, and PhD. degrees in Electrical Engineering from Kyungpook National University, Daegu, South Korea, in 1985, 1987, and 1998, respectively. He is currently a professor in the Department of Information and Communication Engineering at Dongseo University. From 1987 to 1998, he was a research associate at the Agency for Defense Development (ADD). His current research interests include developing secure communication system, side-channel attack, and