Survey on LEACH-based Security Protocols

Triana Mugia Rahayu, Sang-Gon Lee*, Hoon-Jae Lee

Department of Ubiquitous IT, Division of Computer and Information Engineering

Dongseo University

Busan, Korea

gia.sutriadi@gmail.com, nok60@dongseo.ac.kr, hjlee@ dongseo.ac.kr

*Corresponding author

***Abstract*— Energy efficiency is one of the major concerns in designing protocols for WSNs. One of the energy-efficient communication protocols for this network is LEACH that works on cluster-based homogeneous WSNs. Though LEACH is energy-efficient but it does not take security into account. Because WSNs are usually deployed in remote and hostile areas, security becomes a concern in designing a protocol. In this paper we present our security analysis of five security protocols that have been proposed to strengthen LEACH protocols. Those protocols are SLEACH, SecLEACH, SC-LEACH, Armor LEACH and MS-LEACH.**

***Keywords*— Security analysis, WSN, LEACH, SLEACH, SecLEACH, SC-LEACH, Armor-LEACH, MS-LEACH.**

I. INTRODUCTION

Wireless sensor network (WSN) is widely used in many applications, from military until civil applications. This network can contains hundreds even thousands of small-size sensor devices that are deployed in remote or hostile area. These sensors are resource-constrained devices which are limited in computation, power and memory.

Because WSNs employ many constraint-based devices, preserving network lifetime is one of the major concern in designing proper communication protocol for WSN. Heinzelman et. al. [1] proposed an energy-efficient communication protocol for wireless microsensor networks called Low-Energy Adaptive Clustering Hierarchy (LEACH). Using simulation they showed LEACH can achieve as much as a factor of 8 reduction in energy dissipation compared with conventional routing protocols. LEACH is a clustering-based protocol that rotates randomly the role of cluster heads (CHs) in order to evenly distribute the energy load among the sensors in the network. It incorporates data aggregation scheme into routing protocol to reduce the amount of information that CHs should transmit to base station (BS).

LEACH divides network operation processes into rounds. Every round consists of 2 phases, *setup phase* and *steady-state phase*. Cluster formation is done in setup phase. This phase is comprised of 3 steps. At first, the self-elected CH candidates advertise their intentions to become CHs for that round. In the next step, based on the signal strength of received advertisement message, nodes send request to a CH in which they want to join. In the last step CHs send the schedule messages containing TDMA time slots to their cluster members. After clusters formation, LEACH enters the steady-state phase. In the steady-state phase the actual communication takes places. At first cluster members send their sensor readings to CH. Then CH aggregates the readings and sends the aggregated data to BS. Figure 1 depicts the summary of each step of LEACH.

Setup phase
1. $H \Rightarrow \mathcal{G}$: id_H, **adv**
2. $A_i \rightarrow H$: id_{A_i}, id_H, **join_req**
3. $H \Rightarrow \mathcal{G}$: $id_H, (\dots, \langle id_{A_i}, t_{A_i} \rangle, \dots)$, **sched**

Steady-state phase
4. $A_i \rightarrow H$: id_{A_i}, id_H, d_{A_i}
5. $H \rightarrow BS$: $id_H, id_{BS}, \mathcal{F}(\dots, d_{A_i}, \dots)$

Figure 1. LEACH protocol

The various symbols denote:

H, A_i, BS : A CH, ordinary node and base station, respectively
$\mathcal{G}$: The set of all nodes in the network
$\Rightarrow, \rightarrow$: Broadcast and unicast transmissions, respectively
id_x : Id of node x
adv, **join_req**, **sched** : String identifier for message type
$\langle id_x, t_x \rangle$: A node id x and its time slot t_x in its cluster's TDMA schedule
d_x : Sensed data from node x
$\mathcal{F}()$: Data aggregation function

LEACH concerns only about energy efficiency and does not take security aspect into consideration. Thus there have been some works proposed in order to secure LEACH.

In the next section security requirements of WSN is presented and followed by protocol description and security analysis of SLEACH [2], SecLEACH [3], SC-LEACH [4], Armor LEACH [5] and MS-LEACH [6]. Section IV concludes our paper.

II. SECURITY REQUIREMENTS OF WSN

The security requirements of WSN are similar to those of traditional wireless network since they share some properties [7]. But due to hostile environment and resources-constrained sensors, it is more challenging to devise communication protocols that satisfy these requirements for WSN. In this section the required security properties are presented.

- Data confidentiality – To ensure that the content of the message should not be revealed to the unauthorized receiver. The standard approach to achieve this is by encrypting the message with a secret key that only the intended receivers possess.
- Data integrity and freshness – Data integrity guarantees that the message has not been altered during the propagation. Data freshness protects network from reply attack.
- Source authentication – Enables a sensor node to ensure the identity of the peer node it is communicating with.
- Availability – To guarantee the survivability of network services against Denial-of-Service attacks.

III. LEACH-BASED SECURITY PROTOCOLS

A. *SLEACH*

SLEACH is the first protocol that attempt to add security to LEACH. Figure 2 summarizes SLEACH protocol.

Setup phase

1. 1 $H \Rightarrow \mathcal{G}$: $id_H, mac_{k_H}(id_H||c_H||\text{sec}\_adv)$, **sec_adv**
   A_i : store(id_H)
   BS : if $mac_{k_H}(id_H||c_H||\text{sec}\_adv)$ is valid, add (id_H, V)
1. 2 $BS \Rightarrow \mathcal{G}$: $V, mac_{k_j}(V)$
1. 3 $BS \Rightarrow \mathcal{G}$: k_j
   A_i : if $\left(f(k_j) = k_{j+1}\right) and\ (id_H \in V)$ id_H is authentic
2. $A_i \rightarrow H$: id_{A_i}, id_H, **join_req**
3. $H \Rightarrow \mathcal{G}$: $id_H, (\dots, \langle id_{A_i}, t_{A_i}\rangle, \dots)$, **sched**

Steady-state phase

4. $A_i \rightarrow H$: $id_{A_i}, d_{A_i}, mac_{k_{A_i}}(id_{A_i}||c_{A_i})$
5. 1. $H \rightarrow BS$: $id_H, \mathcal{F}(\dots, d_{A_i}, \dots)$, $mac_{k_H}(id_H||\mathcal{F}(\dots, d_{A_i}, \dots)||c_H)$
5. 2. $H \rightarrow BS$: $id_H, \left(\dots, id_{A_i}, mac_{k_{A_i}}(id_{A_i}||c_{A_i}) \dots\right)$, $mac_{k_H}(id_H||c_H)$, **mac_array**
6. $BS \rightarrow H$: intruder ids

Figure 2. SLEACH protocol

Additional symbols denote:

k_X : Symmetric key shared by node X and BS
$mac_k()$: MAC calculated using k key
$||$: concatenation operator
sec_adv,
mac_array : String identifier for message type
V : An array of node ids
c_x : Counter shared by node x and BS

Prior to deployment each sensor node X is loaded with two keys: X_x, a master symmetric key that node X shares with BS; and k_j, a group key that is shared by all members in the network. From X_x the key holders derive k_X for message authentication code (MAC) computation and verification. k_j is the last key of a key sequence S generated by applying successively a one-way hash function f to an initial key k_0 ($S = k_0, k_1, \dots, k_{j-1}, k_j$ where $f(k_j) = k_{j+1}$). The BS keeps S secret, but shares the last element k_j with the rest of the network. SLEACH uses μTESLA to do broadcast authentication.

In the first step of the setup up phase, CH candidates broadcast modified advertisement message, the **sec_adv**, to the network. This message contains CH's id, the MAC of the CH's id and **sec_adv**. This protocol includes counter of CH into that MAC computation. CH calculates the MAC using the key it shares with BS, k_H. Upon receiving advertisement messages, each node stores the ids of the CH. When BS receives any advertisement message, it verifies the MAC of the message. If the message is valid, BS put the CH's id into the list of legitimate CHs, V.

After compiling the V list, BS broadcasts the V along with the MAC of V that is computed using k_j. The key k_j is the last key chain that has not been revealed to the network while now whole network hold the key k_{j+1}. After some time BS reveals k_j. Every node then checks the validity of k_j using its k_{j+1} and check if the id of the chosen CH candidate is in the list of the legitimate CHs.

Then in the second step each node sends its join request **join_req** to a CH in which it wants to join.

After that, in the last step of setup phase, CHs broadcast schedule message **sched** containing TDMA time slots to their cluster members.

After cluster formation has been done, the network enters the steady-state phase. In the first step every cluster member sends its sensor reading to its CH according to the time slot assigned for it. The message is shown in step 4 of figure 2. It contains id of the sender, sensor reading, and the MAC of the sender's id and counter. That cluster member uses the key it shares with BS to calculate MAC. Counter is used here to provide data freshness.

After CH collects all the sensor readings from its members, it aggregates them. Then it creates a message containing its id, aggregated data and MAC of them. It also includes its counter shared with BS into the MAC computation. Then it sends this message to BS. This is shown in step 5.1 in figure 2.

Another message a CH sends to BS is the MAC array of its cluster members **mac_array**. Step 5.2 in figure 2 shows the message containing CH's id, string of cluster members' ids and their MACs pairs (taken from message in step 4) and the MAC of CH's id and the counter.

In the final step BS verifies every MAC of sensor nodes. If the BS finds that any MAC is invalid, it will drop the whole packet and sends back the list of illegitimate sensors to corresponding CH. So for the next round CH can blacklist them.

This approach has some drawbacks that root in the lack of secret keys shared among nodes in the network. Those drawbacks are:

- This protocol does not prevent intruder joining into the network. In the cluster formation stage, anyone can send join request message to any available CH in the network without authentication. This happens because

there are no secret key shared among nodes. CHs are incapable of verifying nodes joining their clusters. This condition compels BS at last to verify each clusters members before it accepts the aggregated data sent by their CHs. This scheme can result in wasting network resources because BS will drop the whole aggregated data packet when any cluster member found to be illegal.

- The schedule message is not authenticated and sent in plain text. This condition opens up an opportunity for intruders to disrupt the communications. For example intruders can send the false schedule, or send a packet at the same time a certain node is scheduled to send its data. Another problem is that the intruders may send advanced false schedule message while member nodes are not equipped with protection schemes that may help them not to receive this message. If some of the cluster members use the fake schedule, then data transmission collision may happen because two or more nodes may send data at the same time. Therefore authentication and integrity are needed to secure the schedule message.
- The sensed data sent by each member to the CH is not protected, sent in the plain text. In the WSN applications such as for battlefield and healthcare, the data are sensitive information. The confidentiality is needed in order to protect the data from unwanted eavesdropper. Besides it is not encrypted, the sensed data is not included in the MAC computation of the message as well. So the BS only receives the already compressed data from a CH. The purpose of the MAC here is for the (member) node authentication by BS. There is no data authentication to check whether the data have been tampered or not.
- Another drawback is that there is no key update provisioning for key k_j. If the key chain stored in the BS is all used, then BS does not have secret group key to secure the broadcast authentication mechanism.
- After detecting the originators of failed MACs, BS sends the list of those intruders' ids to their CHs. Later on CHs will drop the message from these nodes for the remaining of the round. This means that BS depends on the CHs to provide aggregated data from legitimate nodes. But the problem may rise for the next new CHs. Because the new CHs do not know the previous blacklisted nodes, the new CHs will include these intruders in the data gathering process up until another list of intruders' ids is revealed again by BS. This condition wastes network resource. Because before the intruders being blacklisted, the BS already dropped the message containing their data.

B. SecLEACH

SecLEACH was proposed to enhance SLEACH. Figure 3 summarizes SecLEACH protocol.

Setup phase

1. $H \Rightarrow \mathcal{G}$: $id_H, nonce$, **adv**
   A_i : choose r such that $r \in (\mathcal{R}_H \cap \mathcal{R}_{A_i})$
2. $A_i \rightarrow H$: id_{A_i}, id_H, r, **join_req**, $mac_{k_{[r]}}(id_{A_i}||id_H||r||nonce)$
3. $H \Rightarrow \mathcal{G}$: $id_H, (\dots, \langle id_{A_i}, t_{A_i}\rangle, \dots)$, **sched**

Steady-state phase

4. $A_i \rightarrow H$: id_{A_i}, id_H, d_{A_i}, $mac_{k_{[r]}}(id_{A_i}||id_H||d_{A_i}||nonce+j)$
5. $H \rightarrow BS$: $id_H, id_{BS}, \mathcal{F}(\dots, d_{A_i}, \dots)$, $mac_{k_H}(\mathcal{F}(\dots, d_{A_i}, \dots)||c_H)$

Figure 3. SecLEACH protocol

Additional symbols denote:
r : Key index in the key rings
R_x : The node X's key ring
$k_{[r]}$: Symmetric key associated with r
j : Reporting cycle within current round

SecLEACH uses random key predistribution proposed by Eschenauer and Gligor [8]. Prior to deployment each node is given a secret key that it shares with BS and a set of key rings drawn from a large key pool. Those key rings contain pairs of key id and the key.

From figure 3, we can see in the setup phase that at first CH broadcasts its advertisement message **adv** containing its id and a nonce. Any node, after receiving all advertisement messages, will then choose the CH it wants to join based on the signal strength and choose the index of secret common keys it shares with chosen CH.

In the second step a node sends its join request message **join_req** to a CH containing its id, CH's id, the index of shared keys and also the MAC using the shared key. In order to prevent reply attack the node includes the nonce in MAC computation.

After that CH broadcasts the schedule message **sched** to its cluster members.

In steady-state phase, each cluster member sends its sensor readings to CH according to the time slots. As shown in step 4 in figure 3, SecLEACH includes sensor reading into the MAC computation. It uses the key it shares with CH to compute MAC. It also puts nonce and the reporting cycle within the current round into the MAC computation to provide freshness. This is different from SLEACH.

In the last step of steady-state phase, step 5 of figure 3, CH sends the aggregated data to the BS along with the MAC using the key shared with BS. It also puts counter shared with BS into the MAC computation.

As we can see, compared to SecLEACH, SLEACH only provides nodes-to-BS pairwise key solution and does not provide shared secret keys among nodes. By deploying

random key predistribution solution, SecLEACH improves upon SLEACH by providing a solution for node-to-CH authentication. It thus cuts down the steps where BS needs to verify each cluster members before accepting the aggregated data sent by their CH.

The join request message is now authenticated. Therefore it prevents intruder from joining the network.

Different from SLEACH that does not provide data authentication for sensed data sent by members to their CH, SecLEACH includes the data in the MAC of the message. This scheme helps verify the data whether has been tempered or not.

But SecLEACH still inherits some of SLEACH drawbacks. Those are:

- The data integrity of schedule message is not provided. It is still distributed in plain text makes the problems found in the SLEACH protocol apply in this protocol as well.
- The sensed data is not encrypted, this scheme lacks of confidentiality property in the case the data is sensitive.

We can see also that both protocols still trust CHs to do the data aggregation. Because BS is the only trusted party in the network, both schemes still lack of secure data aggregation protocol.

C. SC-LEACH

SC-LEACH offers two enhancements for LEACH. First one is the algorithm that maximizes the probability of producing the optimal number of CHs in every round. Second one is the enhancement from security aspect using the pre-shared key pair found in [9]. Figure 4 depicts the protocol without the security protection as explained in [4].

Setup phase
1. $H \Rightarrow \mathcal{G}$: $sequence||ID_H||ch(r)$
2. $A_i \rightarrow H$: $sequence||S_{A_i}$
3. $H \rightarrow \mathcal{G}$: $ID_H||$ $[sequence||CDMA\ code||TDMAschedule||ch(r)]$

Steady-state phase
4. $A_i \rightarrow H$: encoded form of id_{A_i}
5. $H \rightarrow BS$: $\mathcal{F}(\dots, d_{A_i}, \dots)$

Figure 4. SC-LEACH protocol

Additional symbols denote:
$sequence$: sequence of present round
ID_H : the index of every key in the CH's key ring
$ch(r)$: the number of all nodes which have ever been CHs
S_{A_i} : A_i's nonce used in the communication with CH

From figure 4, the setup phase starts with each CH candidate broadcasts its datagram. The datagram contains the information of present round sequence, the index of every key in its key ring and the amount of nodes which have been selected as CH recorded so far. Then if a node wants to join a cluster, it determines the nonce it will use in the communication with the chosen CH. After that that node sends that key and sequence information to CH. Every CH then allocates TDMA time slot according to the number of nodes registering in its cluster. In step 3 of figure 4, the message contains CH's key ring index, present round sequence number, CDMA code, TDMA time slot and the number of all nodes which have been CHs.

After the formation of the cluster, cluster members then send the encoded data to the CH based on their time slot. Once a frame is complete, the CH decodes the data, run data fusion algorithm and sends the aggregated data to the BS.

Though the scheme is said to adopt the pre-shared key pair scheme in [9] but the authors did not generously provide how SC-LEACH works in detail. So the security analysis of SC-LEACH is difficult without the clear description of it.

D. Armor LEACH

This protocol combines solutions provided by SecLEACH and TCCA (Time Controlled Clustering Algorithm) [10] into one solution to offer a high level of security on WSN with low power consumption. TCCA modified the way of CHs election by adding another condition for election. That condition is the availability of the energy comparing to the maximum energy of the sensor which is controlled by timestamp. Figure 5 below depicts the protocol.

Setup phase
1. $H \Rightarrow \mathcal{G}$: $id_H, nonce, initialTTL, remainingEnergy, timestamp,$ **adv**
   A_i : choose r such that $r \in (\mathcal{R}_H \cap \mathcal{R}_{A_i})$
2. $A_i \rightarrow H$: $id_{A_i}, id_H, r,$ **join_req**, $timestamp, remainingTTL,$ $mac_{k_r}(id_{A_i}||id_H||r||nonce)$
   $A_i \Rightarrow \mathcal{G}$: $id_H, nonce,$ $remainingTTL, remainingEnergy, timestamp,$ **adv**
3. $H \Rightarrow \mathcal{G}$: $id_H, (\dots, \langle id_{A_i}, t_{A_i} \rangle, \dots),$ **sched**

Steady-state phase
4. $A_i \rightarrow H$: $id_{A_i}, id_H, d_{A_i}, mac_{k_{|r|}}(id_{A_i}||id_H||d_{A_i}||nonce + j)$
5. $H \rightarrow BS$: $id_H, id_{BS}, \mathcal{F}(\dots, d_{A_i}, \dots), mac_{k_H}(\mathcal{F}(\dots, d_{A_i}, \dots)|c_H)$

Figure 5. Armor LEACH protocol

Additional symbols denote:
$initialTTL$: Initial Time To Live
$remainingEnergy$: the remaining energy of CH
$timestamp$: message timestamp of CH
$remainingTTL$: the remaining TTL of message

This protocol extends one-hop cluster to multi-hop cluster communication. Timestamp helps CH to approximate the relative distance of its members and to learn the best Setup phase time to be used in future rounds. TTL with time stamp helps CH to form a multi-hops view of its clusters, in order to create a collision-free transmission schedule.

Basically, the security aspect of Armor LEACH is the same with that of SecLEACH. So the security analysis of both is the same.

E. MS-LEACH

MS-LEACH was proposed to enhance the security of S-LEACH by providing data confidentiality and node to CH authentication using pairwise keys shared between CHs and their cluster members. Figure 6 shows the protocol.

Setup phase

1. 1 $H \Rightarrow \mathcal{G}$: $id_H, mac_{K_H}(id_H||c||\textbf{sec\_adv})$
   A_i : store(id_H)
   BS : if $mac_{K_H}(id_H||c||\textbf{sec\_adv})$ is valid $add(id_H, V)$
1. 2 $BS \Rightarrow \mathcal{G}$: $V, mac_{k_j}(V)$
1. 3 $BS \Rightarrow \mathcal{G}$: k_j
   A_i : if $f(k_j = k_{j+1})$ and $(id_H \in V)$, id_H is authentic
2. $A_i \rightarrow H$: id_{A_i}, id_H, **join_req**
3. H and A_i : $K_{A_i} = f_{KI}(id_{A_i})$
   $K_{HA_i} = f_{K_{A_i}}(id_H)$
4. $H \rightarrow A_i$: $E_{K_{HA_i},C}(id_H, id_{A_i}, T_{A_i}, \textbf{sched})$,
   $mac_{K_{HA_i}}\left(c||E_{K_{HA_i},C}(id_H, id_{A_i}, T_{A_i}, \textbf{sched})\right)$

Steady-state phase

5. $A_i \rightarrow H$: $E_{K_{HA_i},C}(id_{A_i}, d_{A_i}), mac_{K_{HA_i}}\left(c||E_{K_{HA_i},C}(id_{A_i}, d_{A_i})\right)$
6. $H \rightarrow BS$: $E_{K_H,C}\left(id_H, \mathcal{F}(\dots, d_{A_i}, \dots)\right)$,
   $mac_{K_H}\left(c||E_{K_H,C}\left(id_H, \mathcal{F}(\dots, d_{A_i}, \dots)\right)\right)$

Figure 6. MS-LEACH protocol

Additional symbols denote:

c : counter
KI : the symmetric last key chain held by BS preloaded in each node
f_K : a family of pseudo-random function.

Step 1 until 2 in setup phase of figure 6 is similar to SLEACH protocol. In step 3, a pairwise key is generated by both CH and its child node after sending the join request. In step 4 CH unicasts encrypted form of its secured TDMA schedule **sched** to each member using counter value and the generated pairwise key. CH also sends in this step MAC value of counter and encrypted form of TDMA schedule using the generated pairwise key before.

In steady-state phase, every member sends encrypted form its measurement data and ID to CH using generated pairwise key and counter value, and MAC value of the counter and the encrypted form of node ID and its measurement data using the pairwise key. In step 6, CH sends encrypted form of message contains its ID and aggregated data from its members to BS using the shared key between CH and BS and counter value. Also CH sends MAC value of counter and the encrypted message using the shared key between CH and BS.

Following are the observed drawbacks of MS-LEACH protocol:

- The counter description is not clear. In SLEACH the counter used is the counter shared between a node and the BS. Here the counter c found in the step 1.1 and 6 should be different with counter found in step 4 and 5. The proper counter management is not provided.
- MS-LEACH does not provide authentication for join request message.
- Intruders may have intention to crowd the timeslot. In step 2 the join request message is not authenticated. This situation invites anyone to freely send the join request message to CH. Though intruders do not have the symmetric last key chain KI to calculate the pairwise key which make them unable to decrypt the schedule message, CH has no way to remove the timeslot already assigned for the intruders and to rebuild the timeslot.
- Another drawback is about key chain management. Similar with that found in SLEACH, there is no key update provisioning for key k_j. If the key chain stored in the BS is all used, then BS does not have secret group key to secure the broadcast authentication mechanism.
- The schedule message is protected using the pairwise key but it requires multiple unicast communications. This way the energy of a CH can be depleted.

Regarding the intention of intruders to crowd the timeslot, an enhancement to the protocol can be made. The authentication of the request join message can be provided using the pairwise key. Because member node can calculate the pairwise key in advance prior to sending the join request message. A member then can include the MAC value of join request message using the pairwise key to the join request message. This method provides message integrity and authentication as well. While CH can calculate the pairwise key after receiving the identity information of a member node contained in the request join message. This way if CH can verify the message, CH is assured about the possession of legal key KI by the corresponding member node thus in other words the authentication of the member node. This prevents the timeslot to be occupied by intruders. Figure 7 below shows the possible enhancement for MS-LEACH.

…

2. A_i : $K_{A_i} = f_{KI}(id_{A_i})$
   $K_{HA_i} = f_{K_{A_i}}(id_H)$
   $A_i \rightarrow H$: id_{A_i}, id_H, **join_req,** $mac_{K_{HA_i}}(id_{A_i}|id_H|\textbf{join\_req})$
3. H : $K_{A_i} = f_{KI}(id_{A_i})$
   $K_{HA_i} = f_{K_{A_i}}(id_H)$
   Verify join request message

…

Figure 7. Possible enhancement for MS-LEACH protocol

IV. CONCLUSIONS

In this paper, we describe our security analysis of SLEACH, SecLEACH, SC-LEACH, Armor LEACH and MS-LEACH. We also provide some possible solution to some pointed drawbacks. Our findings about their drawbacks also directs us that it is needed to devise efficient secure protocol for WSN that combines both secure routing protocol and secure data aggregation protocols together.

ACKNOWLEDGMENT

This is the statement of acknowledgement for ICACT2014 only for the third 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. (Grant number: 2013-071188). And it also was supported by the BB21 project of Busan Metropolitan City.

REFERENCES

[1] W. R. Heinzelman, A. Chandrakasan, and H. Balakrishnan. Energy-efficient communication protocol for wireless microsensor networks. IEEE Hawaii Int. Conf. on System Sciences, pages 4-7, January 2000.

[2] A. C. Ferreira, M. A. Vilaca, L. B. Oliveira, E. Habib, H. C. Wong, and A. A. F. Loureiro. On the security of cluster-based communication protocols for wireless sensor networks. In 4th IEEE International Conference on Networking (ICN'05), volume 3420 of Lecture Notes in Computer Science, pages 449-458, Reunion Island, April 2005.

[3] Leonardo B. Oliveira, Hao C. Wong, M. Bern, Ricardo Dahab, A. A. F. Loureiro. "SecLEACH – A Random Key Distribution Solution for Securing Clustered Sensor Networks." Fifth IEEE International Symposium on Network Computing and Applications, pp. 145-154, Washington, DC, USA, July 2006.

[4] J. Wang, et al. Secure LEACH routing protocol based on low-power cluster-head selection algorithm for wireless sensor networks. Proc. IEEE ISPACS 2007.

[5] M. A. Abuhelaleh, T. M. Mismar, and A. A. Abuzneid. Armor-LEACH-energy efficient, secure wireless networks communication. Proc. IEEE ICCCN 2008.

[6] M. El-Saadawy, and E. Shaaban. Enhancing S-LEACH security for wireless sensor networks. Proc IEEE Electro/Information Technology (EIT) 2012.

[7] A. Perrig, R. Szewczyk, V. Wen, D. Culler & J. D. Tygar. "SPINS: Security Protocols for Sensor Networks". In ACM Wireless Network 8 (5), pp. 521-534, 2002.

[8] L. Eschenauer and V. D. Gligor, "A key-management scheme for distributed sensor networks", *Proceedings of the 9th ACM conference on Computer and communications security*, pp. 41-47, Washington, DC, USA, November 18-22, 2002.

[9] K. Imamoto, et. al. "A Design of Diffie-Hellman Based Key Exchange Using One-time ID in Pre-shared Key Model," The 18th International Conference on Advance Information Networking and Applications, IEEE Press, Fukuoka, Japan, 327-333, March 2004.

[10] S. Selvakennedy, and S. Sinnappan. "A Configurable Time-Controlled Clustering Algorithm for Wireless Networks". The 11th International Conference on Parallel and Distributed Systems, 2005.