Journal of Ubiquitous Systems & Pervasive Networks
Volume 2, No. 1 (2011) pp. 39-47

Energy-Efficient and Scalable Group Key Management for Hierarchical Sensor Network

Shu Yun Lim [a], Meng-Hui Lim [b]

[a] *Faculty of Information Technology, Universiti Tun Abdul Razak, Kuala Lumpur, Malaysia, 41650*
[b] *School of Electrical and Electronic Engineering, Yonsei University, Seoul, South Korea, 120-749*

Abstract

Key management is deemed as the fundamental essential part of any secure communication. A secure sensor network communication protocol relies on the substantial secure, robust and efficient key management system. Implementing a private and public key pair for sensor network is impractical due to high computation and energy consumption. In this research, we put forward two group key management schemes for hierarchical self-organizing wireless sensor network architecture. Both of the schemes are designed in such a way that more computational and communication burden is placed on the forwarding node and the similar workload is kept as low as possible at the sensor nodes. Besides, multi-level security can be achieved to secure groups of sensors at different levels.

Keywords: *Sensor network security, Group key establishment, Group key transport*

1. Introduction

The security of sensor networks has become one of the most pressing issues in further development of these networks. Compared to the traditional wireless network, Wireless Sensor Network (WSN) provides a different computation and communication infrastructure. These differences originate not only from their physical characteristics, but also from their typical applications. For example, the physical characteristics include the large scale of deployment, limited computing capability, and constraints on power consumption. As a result, the requirements for the key management of a WSN are noticeably different from those for traditional networks.

The major requirements for the key management in a WSN are as follows. First, sensors' communication involves a key distribution procedure between the communication parties, in which the key may be transmitted through some insecure channels. Therefore, key confidentiality, integrity and ownership must be enforced in the whole procedure. Second, it must be power aware, such that the power limitations need to be taken into consideration in the design of a key management protocol for WSN. Third, the key management scheme must be scalable, whereby it must be able to support larger networks and flexible against substantial increase in the size of network even after deployment. Forth, in the event of sensor node compromisation, security credentials which are stored in a sensor node or exchanged over the radio links should not reveal any useful information about any other links in the WSN. This is essential in upholding the resiliency of the WSN. Last but not least, the key management design for WSN needs to be balanced in terms of computation and storage overhead among the participated entities. In fact, this balanced property appears to be more important than the performance of an individual entity as an unbalanced design may results in performance degradation in some entities within a sensor network.

Despite the challenges and requirements described above, we had proposed key management schemes for hierarchical self-organizing sensor networks in a formal and systematic manner. The first proposal, described in section 4, is a hybrid group key management scheme that uses high and middle powered nodes perform an asymmetric key agreement protocol to compute a group key. The group key will later be used for clustered low powered nodes' communication. During the group key transport phase, mutual authentication is performed between the low-powered sensors and the middle-powered nodes, and subsequently allow the establishment of secure group-wise local links.

The second proposal, described in section 5, is a group key establishment scheme with initial shared keys. With initial trust built from a shared key, low-cost symmetric protocols enable the low-powered sensors to authenticate and establish secure group-wise local links. Once secure group keys are established, other security services such as group key refresh can be provided. The key management scheme enables a sensor network to set up cryptographic keys in an autonomous fashion, without relying on expensive cryptography for low

* Shu Yun Lim. Tel.: +60376277368
Fax: +60376277476; E-mail: lim_sy@pintar.unirazak.edu.my

DOI: 10.5383/JUSPN.02.01.005

level nodes and requires only two shared key independent of the network size.

2. Related Work

Cryptography is an important and powerful tool for security services, namely authentication, confidentiality, integrity and non-repudiation. Cryptography has two dominant flavors, namely symmetric key and asymmetric key approach. In symmetric key approach, the same key is used to encrypt and decrypt information, while in the asymmetric cryptography, different keys are used to convert and recover information. Although the asymmetric cryptography provides authentication, integrity, confidentiality and simplicity for key distribution, symmetric keys algorithms are generally more computation-efficient than public key approach. Since security has become a hot research topic in WSN and several key management protocols have being proposed in the literature, we can evaluate current key distribution scheme and consider whether they are suitable for sensor networks.

Sensor nodes require group-wise keys to secure multicast messages. Carman et al. [5] have conducted a comprehensive analysis of various group key schemes and they have concluded that the group size is the primary factor that should be considered in choosing a scheme for generating and distributing group keys in a WSN. One of the ways to establish group-wise key is to employ asymmetric cryptography. For instance, Burmester et al. [2] have proposed the use of an asymmetric Diffie-Hellman based group key transport protocol in 2005. Another prominent asymmetric cryptography approach can be seen from ID-STAR [4], which is an identity-based public key cryptography scheme. This scheme appears to be much more efficient than any other existing public key certificate approaches as it manages to reduce energy and latency costs. Apart from asymmetric schemes, Park et al. [15] have presented a lightweight symmetric protocol that is able to achieve efficient rekeying. Their scheme basically provides key broadcast, the ability to recover lost keys and seamless key refresh. More significantly it only requires hash computations, which contributes to the efficiency of their scheme in WSN key authentication. Moreover, Zhu et al. [22] have proposed a comprehensive symmetric keying mechanism called the Localized Encryption and Authentication Protocol (LEAP). This scheme is able to establish multiple keys for neighborhood supporting as well as global information sharing by using individual, group, cluster, and pairwise shared key. Although LEAP includes several promising ideas, it does not adequately address the scalability issues concerning the distribution and the maintenance of the group key.

To address the problem of scalability, many researchers have proposed the hierarchical network architectures which are similar to the one we propose in this paper. Jolly et al. [10] have employed a hierarchical network organization to establish gateway-to-sensor keys whereas Huang et al. [9] have proposed a hybrid authenticated key establishment scheme which exploits the difference in capabilities between the security managers and the sensors, by putting cryptographic burden where the resources are less constrained. This scheme as well authenticated the two identities based on ECC certificates to avoid the typical key management problem in pure symmetric-key based protocol to maintain a good amount of scalability. However, this scheme involved high transmission communication overhead for mutual authentication, implicit certificate, link generation and explicit key confirmation.

Many researchers are still focusing their researches on asymmetric cryptography even symmetric keys algorithms are generally more computation-efficient than public key approach. According to Du et al. [6] PKC is still feasible for WSN since Gura et. al. [7] had performed ECC signature verification on Crossbow motes taken up only 1.6s. Therefore, Du et al. have opted to PKC by implementing RSA and ECC and reduce the communication overhead by trimming down the single Merkle tree to a number of shorter trees since the communication overhead required by the authentication operation is proportional to the height of the Merkle tree. In [12], Eschenauer et. al. proposed a probabilistic pre-deployment scheme. Each node is loaded with a random subset of keys from a large pool. Two nodes agree on a pairwise key if both find a shared secret key in their subset. The disadvantage of this scheme is that it requires high memory storage requirement in a large scale wireless sensor network.

Authentication and key management require initial trust between some of the parties involved. For instance, a public key certificate is accepted as valid only if signed by a trusted authority. For the case of symmetry key cryptography, the parties must as well somehow acquire a common shared secret that will enable them to communicate securely. In the case of large networks of embedded devices, manually setting a large number of keys is not practical. In many scenarios, access to the devices for administration is impossible once the devices are deployed. Therefore, device configuration is possible only before deployment, and there are no secure offline channels. Once deployed, the network must be autonomous and self-organizing. The initial keys should then be set up securely by the devices themselves, without manual intervention. The typical scenario is for a set of wireless sensors to be deployed or dropped in the environment. At this point, the devices must discover their neighbors and self-organize in an ad hoc network. During this initial phase, the main security concerns are external attacks and possibly malicious devices already present in the environment. The sensors themselves may be assumed initially trustworthy, as it takes time for an adversary to compromise them. As the risk of device compromise increases with time, it is crucial to very quickly establish the initial secure links.

Table 1. Comparisons between key management schemes

	Cryptography	Feature	Disadvantage
Carman et. al. [4]	Asymmetric	Based on group key and ID-based cryptography	Asymmetric cryptography is not energy efficient for WSNs
Du et. al. [5]	Asymmetric	Use one-way-hash function to conduct PKI authentication and cut down communication overhead using shorter Merkle tree.	
μTesla [16]	Symmetric	Use a base station to perform key exchange for sensor nodes	Requires a KDC
Eschenauer et. al. [12]	Symmetric	Based on Random Graph theory	The memory storage requirement is high in a huge WSN

3. Preliminaries

3.1. Sensor Network model

Much of the work currently done on ad-hoc networks as well as sensor networks proposes a flat topology that may not scale well. Our scheme is proposed based on the Self-Organizing Hierarchical Sensor Network Architecture defined by Mobile Networks group (MobiNets) at WINLAB which has appeared in [7] and [21]. In the hierarchical approach, access points (APs) act as bridges between wired and wireless infrastructure while forwarding nodes (FNs) with dual radio interfaces act as radio bridges to provide access for the lowest level sensor nodes (SNs) that are most constrained in terms of battery power and processing capabilities.
FNs sit at the middle of the hierarchy and they are designated as the leader of the lower-tier SNs in the hierarchical WSN. Besides, FNs are connected among themselves to guarantee a connected backbone. In this architecture, we reasonably assume that there is no inter-cluster communication needed between SNs of different groups as for most WSN applications, sensing data collected by SNs is required solely to be transmitted back to APs for processing purpose. In other words, since SNs are required to group themselves in order to fulfill a particular task, it is necessary to derive a group key and distribute it securely to SNs for their communications. In view of the clustered behavior of the low level mobile SNs, a light weight group key management scheme is preferable for the lower-tier nodes that are power and computationally constrained.

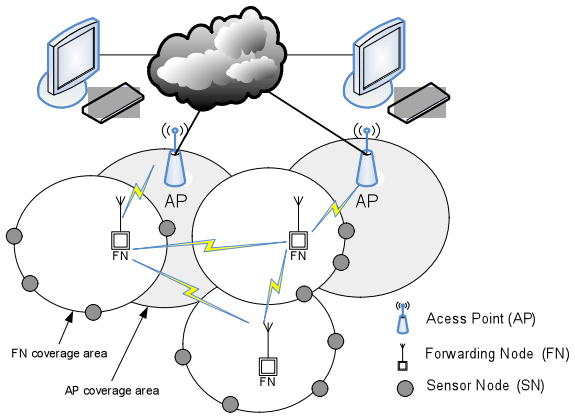

Fig. 1. Self-organizing sensor network model

3.2. Notations

We use the following notations to describe security protocols and cryptographic operations.

SN - Sensor node (low powered communicating principal)

FN - Forwarding node (medium powered communicating principal)

AP - Access point (high powered communicating principal)

N_{SN} - Nonce generated by sensor node

N_{FN} - Nonce generated by forwarding node

K_S - SN's initial secret key (known by SNs and APs but not FN)

$M_1 || M_2$ - Concatenation of messages M_1 and M_2

$MAC_K(M)$ - Message Authentication Code of M using key K

$\{M\}_K$ - Symmetric encryption of message M with key K

$E_K(M)$ - Asymmetric encryption of message M with key K

$D_K(M)$ - Decryption of message M with key K

3.3. Adversarial Model

Our powerful threat model assumes in any two-party's communication over a public channel, a malicious transceiver of the same frequency can always sniff the information carried by these signals. If an intruder gains control over a certain node, he/she can overhear messages between the communicating principals, intercept them and prevent their delivery to the intended recipient. The adversaries may insert forged information into the network through malicious nodes or compromised nodes. Also, we consider APs are always honest (a reasonable assumption) and the possibility malicious insiders to exist among FNs or SNs only in the network deployment. We do not assume that SNs to be high cost tamper-resistant. Once a SN is compromised, all the secret key data lies in that SN will then be revealed to the adversary.
To ensure the applicability and security of our hybrid scheme, we assume the presence of an intrusion detection system (IDS) [18] which is able to detect if any node has been compromised from time to time and exclude it from the network after then. After all, our assumption is that compromised sensor nodes can be detected efficiently and a group key refresh process will be triggered by IDS before the anomaly brings down the entire network.

4. Hybrid Group Key Establishment Scheme

Authentication and group key establishment require initial trust between some of the parties involved. For instance, a public key certificate is accepted as valid only if signed by a trusted authority. For the case of symmetry key cryptography, the parties must acquire a common shared secret that will enable them to communicate securely. Meanwhile, for large networks of embedded devices, manually setting a large number of keys is not practical. In addition, access to the devices for administration in many scenarios is impossible once the devices are deployed. Therefore, device configuration and master key assignment is possible only before deployment. Once deployed, the network must be autonomous and self-organizing. The group-wise keys should then be set up securely by the devices themselves, without manual intervention. Besides, it is crucial to establish the initial secure links as quickly as possible before any adversary compromises the network.
In the literature of key establishment, a hybrid protocol has been typically viewed as a key agreement protocol from the viewpoint of some users and a key transport protocol from the viewpoint of others. From another point of view, the hybrid protocol is the best alternative involving both symmetric and asymmetric techniques. The optimal use of these available techniques generally involves combining symmetric techniques for data encryption and data integrity with asymmetric public-key techniques for signatures and key management. Our hybrid scheme which matches the criteria mentioned above has exploited all the advantages of it where it consists of a group key agreement phase (key agreement protocol) between AP and FN, and a group key transport phase (key transport protocol) between FN and SN.

4.1. Protocol Description

We make use of the higher computational capabilities of APs and FNs to compute the group key by using our Diffie-Hellman based key agreement protocol. After that, every FN that is within a direct communication range with SNs transports the computed group key to the initially trusted SNs by using our key transport protocol. It is essential for the computation of the initial group key between APs and FNs to be done before SNs are deployed. If the initial group key is computed after the deployment of SNs, the time delay required for the initial group key to reach those SNs may pose a higher security threat for the devices to be compromised before the initial link establishment.

Now, suppose that a large group of SNs are going to be dispersed. In our hybrid scheme, each SN is initially preloaded with an initial secret K_S and it is assigned a long term public/private key pair, y_{SN}/x_{SN} by a trusted Certificate Authority (CA). In general, APs are required to agree on a number of group keys with every set of FNs in prior to SNs deployment. Here, we illustrate our hybrid protocol which involves a single AP, FN and SN for a particular group key establishment process.

4.1.1. Group Key Agreement Phase (AP ↔ FN)

Consider that AP and FN are endowed with long term public/private key pairs, y_{AP}/x_{AP} and y_{FN}/x_{FN} and their relative digital certificates $cert_{AP}$ and $cert_{FN}$ respectively, issued by a mutually trusted Certification Authority (CA) with its signature bound to the certificates. We assume g is a primitive root, where $y_{AP} = g^{x_{AP}}$ and $y_{FN} = g^{x_{FN}}$. AP then computes the initial shared secret $K_{AF} = {y_{FN}}^{x_{AP}} = g^{x_{AP} x_{FN}}$, which is used to secure the *MACs* in this protocol. In a similar manner, FN generates the shared secret $K_{AF} = {y_{AP}}^{x_{FN}} = g^{x_{AP} x_{FN}}$. Before the group key agreement begins, AP and FN each chooses an ephemeral private key r_{AP} and r_{FN}, and computes $t_{AP} = g^{r_{AP}}$ and $t_{FN} = g^{r_{FN}}$ respectively, where $r_{AP}, r_{FN} \in_R Z_q^*$. The group key agreement phase can be carried out as follows:

i. FN initiates the group key establishment by sending its ID_{FN}, t_{FN} and $MAC_{K_{AF}}(ID_{FN} \| ID_{AP} \| t_{FN})$ to AP.

$$\mathbf{FN \rightarrow AP:}\ ID_{FN}, t_{FN}, MAC_{K_{AF}}(ID_{FN} \| ID_{AP} \| t_{FN}) \quad (1)$$

ii. Upon reception of FN's message, AP verifies $MAC_{K_{AF}}(ID_{FN} \| ID_{AP} \| t_{FN})$. If it fails, AP terminates the execution. Otherwise, AP selects the initial value of incremental session counter SC from [1, p], where p is the security parameter, and computes $\{SC\}_{K_{AF}}$, the group key $K_G = {t_{FN}}^{r_{AP}}, MAC_{K_S}(ID_{AP} \| ID_{FN} \| K_G \| SC)$ and $MAC_{K_{AF}}(ID_{AP} \| ID_{FN} \| {t_{FN}}^{x_{AP}} \| t_{AP} \| \{SC\}_{K_{AF}})$ respectively. Then, AP sends its ID_{AP}, t_{AP}, $\{SC\}_{K_{AF}}$, $MAC_{K_S}(ID_{AP} \| ID_{FN} \| K_G \| SC)$, and $MAC_{K_{AF}}(ID_{AP} \| ID_{FN} \| {t_{FN}}^{x_{AP}} \| t_{AP} \| \{SC\}_{K_{AF}})$ to FN.

$$\mathbf{AP \rightarrow FN:}\ ID_{AP}, t_{AP}, \{SC\}_{K_{AF}}, MAC_{K_S}(ID_{AP} \| ID_{FN} \| K_G \| SC), MAC_{K_{AF}}(ID_{AP} \| ID_{FN} \| {t_{FN}}^{x_{AP}} \| t_{AP} \| \{SC\}_{K_{AF}}) \quad (2)$$

iii. After receiving the message from AP, FN computes and verifies the authenticity of the message by checking whether

$$MAC_{K_{AF}}(ID_{AP} \| ID_{FN} \| {y_{AP}}^{r_{FN}} \| t_{AP} \| \{SC\}_{K_{AF}}) \stackrel{?}{=} MAC_{K_{AF}}(ID_{AP} \| ID_{FN} \| {t_{FN}}^{x_{AP}} \| t_{AP} \| \{SC\}_{K_{AF}}) \quad (3)$$

If the result is negative, the protocol is terminated. Otherwise, FN performs $D_{K_{AF}}(\{SC\}_{K_{AF}})$ and computes $K_G = {t_{AP}}^{r_{FN}}$ and sends ID_{FN}, $MAC_{(K_{AF} \| K_G \| SC)}(ID_{FN} \| ID_{AP} \| {t_{AP}}^{x_{FN}} \| t_{FN} \| MAC_{K_S}(ID_{AP} \| ID_{FN} \| K_G \| SC))$ to AP.

$$\mathbf{FN \rightarrow AP:}\ ID_{FN}, MAC_{(K_{AF} \| K_G \| SC)}(ID_{FN} \| ID_{AP} \| {t_{AP}}^{x_{FN}} \| t_{FN} \| MAC_{K_S}(ID_{AP} \| ID_{FN} \| K_G \| SC)) \quad (4)$$

iv. Upon receiving FN's message, AP then computes $MAC_{(K_{AF} \| K_G \| SC)}(ID_{FN} \| ID_{AP} \| {y_{FN}}^{r_{AP}} \| t_{FN} \| MAC_{K_S}(ID_{AP} \| ID_{FN} \| K_G \| SC))$ and verifies the authenticity of the message by checking whether

$$MAC_{(K_{AF} \| K_G \| SC)}(ID_{FN} \| ID_{AP} \| {y_{FN}}^{r_{AP}} \| t_{FN} \| MAC_{K_S}(ID_{AP} \| ID_{FN} \| K_G \| SC)) \stackrel{?}{=} MAC_{(K_{AF} \| K_G \| SC)}(ID_{FN} \| ID_{AP} \| {t_{AP}}^{x_{FN}} \| t_{FN} \| MAC_{K_S}(ID_{AP} \| ID_{FN} \| K_G \| SC)) \quad (5)$$

If it is not, the protocol is terminated. Otherwise, the key establishment process is deemed successful. Hence, both entities manage to agree with such group key:

$$\mathbf{FN:}\ K_G = {t_{AP}}^{r_{FN}} = g^{r_{FN} r_{AP}} \quad (6)$$

$$\mathbf{AP:}\ K_G = {t_{FN}}^{r_{AP}} = g^{r_{FN} r_{AP}} \quad (7)$$

4.1.2. Group Key Transport Phase (FN ↔ SN)

Once the group key has been deduced, SNs are then ready to be deployed. After the bootstrapping and the clustering process have been completed, the group key transport scheme is carried out with the initial group key request from SNs. This scheme is usually executed in a short time window after the sensors have been deployed. The scheme can be illustrated as follows:

i. SN initiates the group key transport scheme by generating $N_{SN} \in_R [1, p]$ and sending the group key request message which contains its ID_{SN}, $E_{y_{FN}}(N_{SN})$ and $MAC_{K_{FS}}(ID_{SN} \| ID_{FN} \| N_{SN})$ to FN.

$$\mathbf{SN \rightarrow FN:}\ ID_{SN}, E_{y_{FN}}(N_{SN}), MAC_{K_{FS}}(ID_{SN} \| ID_{FN} \| N_{SN}) \quad (8)$$

ii. On reception of the message, FN decrypts $E_{y_{FN}}(N_{SN})$ with x_{FN} and verifies $MAC_{K_{FS}}(ID_{SN} \| ID_{FN} \| N_{SN})$. If the verification succeeds, FN then transports its ID_{FN}, $E_{y_{SN}}(K_G \| SC)$, N_{FN}, ID_{AP}, $MAC_{K_S}(ID_{AP} \| ID_{FN} \| K_G \| SC)$ (received from AP in the previous phase), and $MAC_{K_{FS}}(ID_{FN} \| ID_{SN} \| N_{FN} \| N_{SN} \| E_{y_{SN}}(K_G \| SC))$ to SN as the response message.

$$\mathbf{FN \rightarrow SN:}\ ID_{FN}, E_{y_{SN}}(K_G \| SC), N_{FN}, ID_{AP}, MAC_{K_S}(ID_{AP} \| ID_{FN} \| K_G \| SC), MAC_{K_{FS}}(ID_{FN} \| ID_{SN} \| N_{FN} \| N_{SN} \| E_{y_{SN}}(K_G \| SC)) \quad (9)$$

iii. After receiving the response from FN, SN computes and verifies $MAC_{K_{FS}}(ID_{FN} \| ID_{SN} \| N_{SN} \| \{K_G \| SC\}_{K_{FS}})$ to ensure the message comes from the intended FN. SN then decrypts and obtains the group key, K_G and the session counter SC by using the x_{SN}. Then, SN computes $MAC_{K_S}(ID_{AP} \| ID_{FN} \| K_G \| SC)$' by using K_S and compares against the received $MAC_{K_S}(ID_{AP} \| ID_{FN} \| K_G \| SC)$ to verify the authenticity of K_G. If any failure of the verification processes is found, SN terminates the protocol immediately. On contrary, if the verification steps are successful, SN then generates $MAC_{(K_{FS} \| K_G \| SC)}(ID_{SN} \| ID_{FN} \| N_{SN} \| N_{FN})$, and sends the following confirmation message back to *FN* (for verification purpose):

SN → FN: $ID_{SN}, MAC_{(K_{FS} \| K_G \| SC)}(ID_{SN} \| ID_{FN} \| N_{SN} \| N_{FN})$ **(10)**

With this three-pass key transport phase, the received K_G can be proven to be authentic to SN. Besides, FN can also be assured that the correct group key has been transmitted. After then, the group key K_G can then be split into two subkeys K_E and K_M used for encryption and authentication of future messages respectively within the cluster.

4.2. Group Key Services

4.2.1. Addition of Sensor Nodes

The new sensors are arbitrarily deployed and they cannot be pre-assigned to a cluster. However, they are preloaded with the initial secret key K_S which is similar to the other existing sensors. Initially, each added SNs sends a request message in the similar form as shown in Eq. (8) to a nearby FN. Note that only the joining SN with the authentic certificate issued by CA can go through the key transport protocol to share the current group key and therefore, they are then able to communicate with all SNs within the cluster. The other existing SNs who have left their previous clusters can rejoin any new cluster group by going through the same procedures.

4.2.2. Group Key Refresh

In general, using the same encryption key for extended periods may incur a great cryptanalytic risk. If the adversary has compromised the group key of a cluster by some means, he would be able to break the whole cluster if the group key is fixed throughout the lifetime of the network. Hence, it is essential to have the group keys renewed occasionally. However, the power limitations of SNs often restrict the key management activities in order to preserve power. Under these circumstances, adequately tuning the key update period would attain an optimum tradeoff between the conflicting goals of keeping a high security level and minimizing the power consumption of SNs. This generally means that a more secure network may require more frequent key renewal, and hence the higher the SNs' power consumption, the shorter the SNs' lifetime would be.

In order to accomplish the renewal of a particular group key, FN and AP initially agree on a new group key and FN then transmits this key to every SN within the cluster. Before deployment, every SN should have been scheduled to expect a new group key from FN periodically depending on the key update period. Suppose that K_G and SC denote the current group key and session counter respectively. FN initially encrypts the next group key K_G' and the next session counter SC' by using ($K_{FS} \| K_G \| SC$) and subsequently transports its ID_{FN}, $\{K_G' \| SC'\}_{(K_{FS} \| K_G \| SC)}$, N_{FN}, ID_{AP}, $MAC_{K_S}(ID_{AP} \| ID_{FN} \| K_G' \| SC')$ and $MAC_{(K_{FS} \| K_G \| SC)}(ID_{FN} \| ID_{SN} \| \{K_G' \| SC'\}_{(K_{FS} \| K_G \| SC)})$ to SN.

FN → SN: $ID_{FN}, \{K_G' \| SC'\}_{(K_{FS} \| K_G \| SC)}, N_{FN}, ID_{AP}, MAC_{K_S}(ID_{AP} \| ID_{FN} \| K_G' \| SC'), MAC_{(K_{FS} \| K_G \| SC)}(ID_{FN} \| ID_{SN} \| \{K_G' \| SC'\}_{(K_{FS} \| K_G \| SC)})$ **(11)**

Upon receiving the broadcast message from FN, SN performs the same verification processes illustrated in 4.1.2 (iii). Moreover, SN checks if SC' is equal to SC + 1. If the verification processes succeed, SN then overwrite the group key with K_G' and acknowledges the receipt of the broadcast message by sending back the confirmation message which comprises of its ID_{SN} and $MAC_{(K_{FS} \| K_G' \| SC')}(ID_{SN} \| ID_{FN} \| N_{FN})$.

SN → FN: $ID_{SN}, MAC_{(K_{FS} \| K_G' \| SC')}(ID_{SN} \| ID_{FN} \| N_{FN})$ **(12)**

4.3. Security Analysis

Authentication – In our hybrid protocol, data integrity and entity authentication are accomplished by sending the message together with MAC in every message transmission. Each communicating entities is required to perform MAC verification with the received message before accepting it. Besides, MAC can only be computed by using the derived shared secret, K_{AF} (in group key agreement phase), K_{FS} (in group key transport phase and group key refresh scheme) and the previous $K_G \| SC$ (in group key refresh phase). With this, the receiving entity can always be assured that the received message is originated from the intended sender through MAC verification. Hence, an adversary who has no knowledge about K_{AF}, K_{FS} or K_G, would not be able to alter the message contents without being detected in the respective phase of our scheme.

Key Confirmation – Our scheme achieve explicit key authentication, where both implicit key authentication and key confirmation hold for all phases. Note that in group key agreement phase/ group key transport phase, K_G and SC are used along with K_{AF}/ K_{FS} to compute the MAC in Eq. (4)/ Eq. (10). In this sense, AP/ FN can be assured that FN/ SN is in possession of the correct group key. Similarly in group key refresh phase, the next K_G' and SC' are used with K_{FS} in MAC computation in Eq. (12) to assure FN that the respective SN has obtained the right group key. In other words, if there is no failure message (resulted by unsuccessful verification process) in the protocol run, the responder can then be guaranteed explicitly that the group key is generated or received correctly.

Unknown Key Share Resilience – The identities of the sending and receiving entities have been included in every MAC computation. This significantly prevents the attacker from launching the unknown key-share attack in various ways on our hybrid scheme. With this, a stronger sense of authentication can be achieved significantly.

Key Control Resilience – Apparently in our hybrid protocol, neither AP nor FN could alone force the group key to a predetermined or predicted value since the group key of our protocol is governed by using the long term and ephemeral private keys of all the protocol participants. $MAC_{K_S}(ID_{AP} \| ID_{FN} \| K_G // SC)$ is computed by AP in 4.1.1 (ii) and it is then relayed from FN to SN, particularly for SN to verify that K_G and SC are agreed by both AP and FN and it is not determined by FN alone. To realize this purpose, we require K_S to be kept secret from FN and made known only to APs and SNs. For example, suppose that a malicious FN wishes to gain full control of the group key by sending its pre-computed K_G^* instead of the authentic K_G (resulted from the group key agreement phase) to SN in the group key transport phase. However, the malicious FN does not have any knowledge about K_S and therefore, it will eventually fail to trick SN since it is unable to forge $MAC_{K_S}(ID_{AP} \| ID_{FN} \| K_G^* \| SC)$. Of course, we do not refute the possibility that a malicious FN which compromise any SN or conspire with any other malicious AP (in order to acquire K_S) could violate the key control resilience property. We suggest that relaying AP's signature over K_G and SC, (denoted as $sig_A(ID_{AP} \| ID_{FN} \| K_G \| SC)$) instead of MAC_{K_S}

(ID_{AP} || ID_{FN} || K_G // SC) can completely eliminate this threat at a higher verification cost incurred in SNs.

Known Group Key Resilience – The group key of our hybrid scheme ($K_G = g^{r_{FN}r_{AP}}$) is periodically updated. Thus, it would vary with every protocol rounds since it is established according to the values of the APs and FNs' ephemeral private keys in that specific round. In other words, the group keys that are established in our protocol are independent to each other. Hence, the knowledge of previous group keys does not assist the adversary in deriving any past and future group keys.

Key Compromise Impersonation Resilience – In general, the Key Compromise Impersonation (KCI) attack involves an adversary who has obtained the long term secret key of an honest party. Instead of impersonating the corrupted party directly, an adversary may want to exploit the long term key and impersonate another party in a communication run in order to capture valuable information about the corrupted party.
Here, we analyze two practical scenarios that would possibly occur in the group key agreement phase:

- Suppose that an adversary, E_{AP} has compromised x_{FN} and computed $K_{AF} = y_{FN}{}^{x_{AP}}$. He then wishes to fool FN by masquerading as AP in a communication run. However, E_{AP} does not know how to compute the component $t_{FN}{}^{x_{AP}}$ (or $y_{AP}{}^{r_{FN}}$) in $MAC_{K_{AF}}(\ldots)$ in Eq. (2) since he has no knowledge about x_{AP} or r_{FN}. Hence, E_{AP}'s attempt will eventually be impeded when A verifies Eq. (3).
- On contrary, if an adversary, E_{FN} has compromised x_{FN}, obtained $K_{AF} = y_{AP}{}^{x_{FN}}$ and he wants to fool AP by impersonating FN in a communication run, E_{FN} would be unable to calculate $t_{AP}{}^{x_{FN}}$ (or $y_{FN}{}^{r_{AP}}$) $MAC_{K_{AF}}(\ldots)$ in Eq. (4) since he has no knowledge about x_{FN} or r_{AP}. This significantly foils E_{FN}'s attempt as AP verifies Eq. (5).

In group key transport phase, note that $E_{y_{FN}}(N_{SN})$ in Eq. (8) and $E_{y_{SN}}(K_G \| SC)$ in Eq. (9) can only be asymmetrically decrypted by using x_{FN} and x_{SN} respectively in order for the respective party to compute the next response message. Hence, compromising SN's (or FN's) secret key and impersonate FN (or SN) to communicate with the victim would not enable the protocol run to be completed successfully due to decryption failure. By considering similar scenarios in group key refresh phase, the attempt of the adversary will be thwarted since the adversary would need an extra knowledge of the current group key and session counter in order to carry out a successful KCI attack. As a result, our protocol can provide absolute immunity to the KCI attack.

Replay Protection – Replaying message maliciously in our protocols is apparently infeasible. In our protocols, we do not employ timestamp to offer such protection since time synchronization is often not easy to attain. In group key agreement phase, the ephemeral information (t_{FN} and t_{AP}) from each communicating party is piggybacked onto the subsequent message, particularly for verification purpose. In addition, nonces are employed to ensure the freshness of the messages in group key transport phase and group key refresh phase. The knowledge of current group key and session counter are required to generate or verify the messages. More importantly, since the explicit key confirmation is provided in all phases, our protocols can guard against replay attack efficiently.

4.4. Performance Analysis

4.4.1. Communication Complexity

The communication overhead incurred in our initial group key establishment and periodical key refresh procedures are relatively low. In this model, we assume ID_{SN} and ID_{FN} as 64-bit, ID_{AP} as 32-bit, generated nonce as 32-bit, ephemeral public key and computed group key as 64-bit, session counter as 16-bit, and MAC output as 32-bit. According to Zoltak et. al [23] and Karlof et al. [11], a 4 bytes of MAC length would be sufficient to provide a well sufficient security level and enable a comfortable implementation of the system. In general, an increase in the packet length would lengthen the transmission time. Hence, our protocol is designed to be lightweight in the sense that only short data is brought into the transmission streams. The longest message in our proposed schemes would only take up 272 bits which is equivalent to 34 bytes. As this message length can be perfectly fitted into a SN packet payload per communication (TinySec-Auth / TinySec-AE packet [11]), it requires only a total of 3 messages to be exchanged in agreeing upon a group key (K_G) between AP and FN, and another 3 messages for transporting the group key from FN to SN whereas in periodic key refresh phase, a total of 2-message transmission is required. Consequently, our hybrid scheme manages to achieve a total of 116-byte data transmission for the key establishment handshake process and 46-byte data transmission for key update process.
Since SNs are much more battery and computational resources limited while FNs and APs are more powerful, we restrict our attention only to the efficiency of SN side. During the initial group key establishment phase, every SN is required to perform 1 exponentiation, 4 MAC computations, 1 asymmetric encryption and decryption, and a random number generation while in the periodic group key update session, each SN is needed to perform 1 exponentiation, 3 MAC computations and a symmetric decryption only to receive a new group key from FN.
Even though the communication overhead at SN side can be kept low, FN's communication overhead our scheme could be enormous, depending on the size of the clusters. But still, if we are able to keep the cluster size small, then the work required to be done by a specific FN can be minimized. Since FN has higher computation power and computation ability compared to SN, the effect of an increasing cluster size is deemed considerably insignificant. All in all, our protocol is able to achieve a substantial cost reduction not only in message transmission, but also in computation.

4.4.2. Storage Complexity

Given a sensor network of size n, a public key scheme would require a large storage space for keys and certificates; a Merkle Tree scheme would require each sensor to store ($1 + log\ n$) keys; a random key scheme would require to store n number of keys and a pairwise symmetric scheme would require to store $2n$ number of keys. Comparatively in the group-wise shared key approach, the key storage per SN is minimal where our hybrid scheme only requires an initial secret K_S and a public/private key pair y_{SN}/x_{SN} to be stored in each SN. Hence, the storage overhead of SN in our hybrid scheme is deemed substantially small and it is independent of the network size.

5. Group Key Establishment Scheme with Initial Shared Keys

The proposed scheme is based on the key transport protocol model of c. Boyd et al. [1]. It is a group key transport protocol whereby forwarding node is designated or elected as the leader. This leader generates and distributes a group secret to other nodes in the group. Sufficient key control can be achieved by using message encryption and authentication key from the initial secret. In this scheme, there are two types of group keys: intra-cluster, and inter-cluster. The intra-cluster group key is used for encryption and decryption of messages inside a group of sensor nodes, whereas the inter-cluster group key is used among groups of clusters. Every sensor nodes in a group establish the group key with a higher level forwarding node dynamically after deployment.

5.1. Protocol Description

Our protocol builds initial trusted links between sensors and forwarding nodes. Forwarding node that is within direct communication range of sensors transports out specific group key to the initially trusted sensors. Each sensors and forwarding nodes are loaded with two fixed keys whereby one for inter-cluster communication and one for intra-cluster communication. It is executed in a short time window after the sensors have been deployed.

5.1.1. Group Key Establishment

i. A sensor, *SN*, initiates the group key establishment by generating a random nonce N_{SN} and send the challenge message to cluster head of the following form:

$$\mathbf{SN \rightarrow FN:}\ ID_{SN}, N_{SN}, \textbf{where}\ N_{SN} \in R\ [1..t\ ] \quad (13)$$

ii. The message contains *SN*'s identity and the nonce $N_{SN.}$ On reception of such a message, the receiver *FN* is ready to transport out encrypted group key where $GK \in R\ [1..t\ ]$ as a response message.

$$\mathbf{FN \rightarrow SN:}\ \{GK\}_{FK}, ID_{FN}, N_{FN}\ MAC_{FK}\ (ID_{SN} | N_{SN} | \{GK\}_{FK}) \quad (14)$$

iii. After receiving a response from the *FN*, the initiator sensor *SN* also computes the *MAC* to ensure the message comes from legitimate *FN*. Sensor *SN* is able to decrypt and obtain the new group key, *GK*, provided that it possesses the master key *FK*. Once initiator *SN* has decrypted the key, it generates a *MAC* with decrypted *GK* and sends the following reply back to *H*:

$$\mathbf{SN \rightarrow FN:}\ MAC_{GK}\ (ID_{SN} | ID_{FN} | N_{SN} | N_{FN}) \quad (15)$$

iv. After this acknowledgement, *SN* and *FN* have proven to each other that they knew the master key, *FK*, and they are also both in possession of the new group key *GK*.

v. The group key *GK* is actually split into two subkeys, *KE* and *KM*, used for encryption and authentication of future messages, respectively.

vi. All sensor nodes with initial trust will go through this protocol to build a single group key during cluster formation phase and later on, all sensors within the cluster can communicate using established group key.

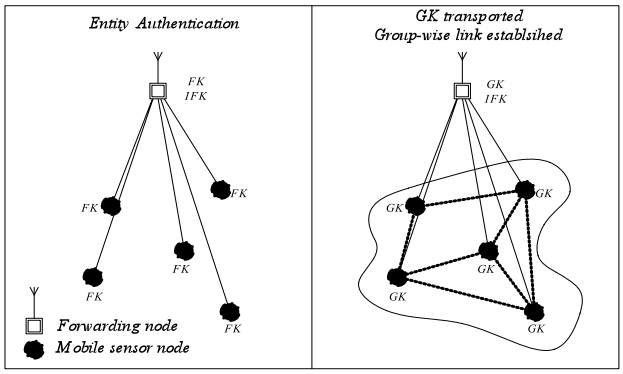

Fig. 2. Group Key Establishment Scheme

5.2. Group Key Services

5.2.1. Addition of Sensors

New sensors are arbitrarily deployed; they cannot be pre-assigned to a cluster. However, they are preloaded with two keys as other sensors. Each added sensor node sends a challenge message containing its *ID* and a generated nonce to a nearby cluster head. The *MAC* is computed with the inter-cluster fixed key *IFK*. Any joining sensor nodes with initial trust will go through the protocol stated in 5.1.1 except that master key *FK* will be replaced by *IFK*. This is to share single group key that is currently in use, for it to communicate with all sensors in the cluster with the group key.

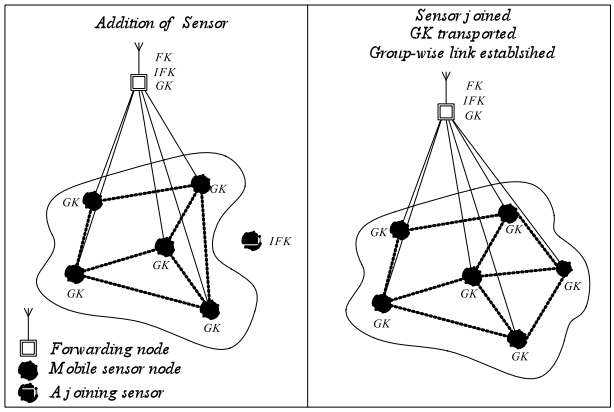

Fig. 3. Addition of a new sensor

5.2.2. Group Key Refresh

Using the same encryption key for extended periods may incur a cryptanalytic risk. It will be necessary to renew the encryption keys occasionally. In order to accomplish the renewal of the sensor keys, the command node generates the new keys, and pushes the keys to other sensors in the cluster by multicast message, as in the case of the revocation. Group key refresh can be initiated by *FN*. The initiator generates a new, random group key and optionally constructs a list of sensors to be excluded from the group.

i. A multicast key-refresh message sent by *FN* to *SNs* except those in the exclusion list. The multicast message is of the following form, *w*here *S* is the group key's sequence number and GK' as the new group key.

FN → SN: $ID_{FN}, N_{FN}, S, \{GK'\}_{KE}, MAC_{KM}\ (ID_{FN} | N_{FN} | S | \{GK\}_{KE})$ **(16)**

ii. The message is protected by using group key *KE* and *KM* that *FN* and *SN* set up during cluster formation.

iii. When a *SN* receives such a key-refresh message, it checks the message integrity using *KM*, and checks whether the message is fresh, based on the sequence number *S*. If both checks succeed, *SN* accepts the new group key carried by the message, and sends back an acknowledgement message to the *FN*.

SN → FN: $ID_{SN}, MAC_{GK'}\ (ID_{SN} | ID_{FN} | N_{FN})$ **(17)**

iv. This protocol distributes the new group key securely and robustly. As long as the good group members are connected, the flooding-like procedure distributes the new key to all good members in a robust manner.

5.3. Security Analysis

Data integrity and entity authentication are accomplished by sending the message together with MAC in every message transmission. Each communicating entities is required to perform MAC verification with the received message before accepting it. Master key *FK* is distributed to *SN*s and *FN*s based on the initial trust. After *SN* sent out challenge to establish a group key with forwarding nodes, it will receive an encrypted group key and a *MAC* with key *FK*. Based on the *MAC*, the *SN* can verify that the message came from the legitimate *FN* that poses the same *FK*. With this MAC verification, message sender can always be authenticated. Besides, MAC can only be computed by using the secret shared secret, *FK* (in group key establishment phase) and the K_M (in group key refresh phase). With this, the receiving entity can always be assured that the received message is originated from the intended sender through MAC verification. Hence, an adversary who has no knowledge about *FK* and *GK* would not be able to alter the contents of the message without being detected in all phases of our scheme.

Later on, the *SN* can decrypt the *GK* with its *FK* and send back a *MAC* with the newly decrypted *GK* as an acknowledgement to forwarding node. Therefore, with the recipient of the acknowledgement message, a group key is established between the sensor node and the forwarding node. Suppose that an adversary impersonates *FN* and send out the fake $\{GK\}_{FK}$ to *SN*, legitimate *FN* will not establish the group key with that specific node and put it in the exclusion list after the receiving the acknowledgement message since $MAC_{GK}(\ldots)$ could not be verified.

5.4. Performance Analysis

5.4.1. Communication Overhead

The communication overhead of our scheme is incurred during the initial key establishment phase and during periodic key refresh procedures. It requires a distribution process to distribute group key after authentication between *SN*s and a *FN*. A total of three transmissions are required to established the group key *GK* per sensor. On the other hand, a *FN* communication overhead relies on the size of the cluster. If we are able to keep the cluster size small thus minimizing the work need to be done on a specific *FN*. Anyhow, since *FN* has higher computation power and computation ability, the effect of increasing cluster size is considerably insignificant.

5.4.2. Storage Overhead

This scheme requires two keys set to be stored at each node. The complete key set requires 256-bit storage (here the author assume encryption scheme requires 128-bit key). IEEE 802.15.4 Low-Rate Wireless Personal Area Network (WPAN) standard defines device ID as 64-bit [14]. Consider the ID length of a specific node as 64-bit, generated nonce as 16-bit, and MAC output as 32-bit as proposed by Zoltak et. al [23] and Karlof et al. [11], the longest message would take up 240-bit which is 30 bytes. This message length can be fit into one sensor nodes packet payload per communication. As compared to pairwise symmetric scheme, public key certificate scheme and random key scheme, the storage overhead of our hybrid protocol is substantially small and independent of the network size.

Table 2. Analysis of key management schemes

Technique	Key storage per sensor	Scalability	Affected nodes after an intrusion
PCK	2 (a key and a certificate)	Node-to-node authentication	Only sensors communication with the intruded node
Merkle Tree	$1 + \log n$	Complex tree modification	Only sensor communicating with the intruded node
Random Key	n	Simple with probability *p*	All sensors that share the same set of n keys stored at the intruded node
Symmetric Key	2n	Node-to-node authentication	Only sensor communicating with the intruded node
Our scheme	2	Node-to-node authentication	All sensors that share the same set of key within a cluster

6. Conclusions

This research proposes two group key management schemes and these schemes are designed to conserve energy by placing the cryptographic burden on sensors with higher computation power, the access points and forwarding nodes. Both of the key management schemes enable a sensor network to set up cryptographic keys in an autonomous fashion. A small amount of keys are required independent of the network size, and hence high scalability. More promisingly, sensor network is able to implement these encryption primitives in an efficient way without sacrificing its strength.

7. References

[1] Boyd, C., Mao, W., Paterson, K.G., "Deniable Authenticated Key Establishment for Internet Protocols", Proceedings of 11th International Workshop on Security Protocols, LNCS, vol. 3366, pp. 255-271 (2005)

[2] Burmerster, M., Desmedt, Y., "A Secure and Scalable Group Key Exchange System", Information Processing Letter, vol. 94, no. 3, pp. 137-143 (2005)

http.dx.doi.org/10.1016/j.ipl.2005.01.003

[3] Burmerster, M., Desmedt, Y., "A Secure and Efficient Conference Key Distribution System", Proceedings of Eurocrypt 94, LNCS, vol. 950, pp. 275-286 (1995)

[4] Carman, D., Matt, B., Cirincione, G., "Energy-efficient and Low-latency Key Management for Sensor Networks", Proceedings of 23rd Army Science Conference (2002)

[5] Carman, D., Kruus, P., Matt. B., "Constraints and Approaches for Distributed Sensor Network Security", NAI Labs: Technical Report #00-010 (2000)

[6] Du, W., Wang, R., Ning, P., "An Efficient Scheme for Authentication Public Keys in Sensor Networks", Proceeding of 6th ACM International Symposium on Mobile Ad Hoc Networking and Computing (MobiHoc), pp. 58-67 (2005)

[7] Ganu, S., Raju, L., Anepu, B., Seskar, I., Raychaudhuri, D., "Architecture and Prototyping of an 802.11-based Self-Organizing Hierarchical Ad-Hoc Wireless Network (SOHAN)", Proceedings of the 15th IEEE International Symposium on Personal, Indoor and Mobile Radio Communications, vol. 2, pp. 880-884 (2004)

[8] Gura, N., Patel, A., Wander, A., Eberle, H., Chang, S., "Comparing Elliptic Curve Cryptography and RSA on 8-bit CPUs", Proceedings of the 2004 Workshop on Cryptographic Hardware and Embedded Systems (CHES 2004), LNCS, vol. 3156, pp. 119-132 (2004)

[9] Huang, Q., Cukier, J., Kobayashi, H., Liu, B., Zhang, J., "Fast Authenticated Key Establishment Protocols for Self-organizing Sensor Networks", Proceedings of 2nd ACM International Workshop on Wireless Sensor Networks and Applications, pp. 141-150 (2003)

[10] Jolly, G., Kusu, M., Kokate, P., "A Hierarchical Key Management Method for Low-Energy Wireless MSN Networks", Proceedings of Eighth IEEE Symposium on Computers and Communication, pp. 335 (2003) http.dx.doi.org/10.1109/ISCC.2003.1214142

[11] Karlof, C., Sastry, N., Wagner, D., "TinySec: A Link Layer Security Architecture for Wireless Sensor Networks", Proceedings of the 2nd ACM Conference on Embedded Networked Sensor Systems (SenSys 2004), pp. 162-175 (2004)

[12] Laurent Eschenauer , Virgil D. Gligor, A key-management scheme for distributed sensor networks, Proceedings of the 9th ACM conference on Computer and communications security, Washington, DC, USA, 2002.

[13] Menezes, A.J., van Oorschot, P.C., Vanstone, S.A., "Handbook of Applied Cryptography", CRC Press (1996) http.dx.doi.org/10.1201/9781439821916

[14] Naveen Sastry, David Wagner, "Security Considerations for IEEE 802.15.4 Networks", ACM Workshop on Wireless Security WiSe 2004, October 2004.

[15] Park, T., Shin, K.G., "LiSP: A Lightweight Security Protocol for Wireless Sensor Networks," ACM Transactions on Embedded Computing Systems, vol. 3, no. 3 (2004)

[16] Perrig, A., Szewczyk, R., Wen, V., Culler, D., Tygar, J.D., "SPINS: Security Protocols for Sensor Networks", Proceedings of 7th Annual International Conference on Mobile Computing and Networks (MOBICOM 2001), pp. 189-199 (2001) http.dx.doi.org/10.1145/381677.381696

[17] Roman, R., J, Zhou., J, Lopez., "Applying Intrusion Detection Systems to Wireless Sensor Networks", Consumer Communications and Networking Conference 2006. 3rd IEEE, pp. 640- 644 (2006)

[18] Su, C., Chang, K., Kuo, Y., Horng, M., "The New Intrusion Prevention and Detection Approaches for Clustering-Based Sensor Networks", IEEE Wireless Communications and Networking Conference (WCNC 2005), vol. 4, pp. 1927-1932 (2005)

[19] Westhoff, D., Girao, J., Sarma, A., "Security Solutions for Wireless Sensor Networks", NEC Journal of Advanced Technology, vol. 59, no.2 (2006)

[20] Xie, L., Zhu, H., Xun, Y., Zhu, Y., "Tamper-resistance Key Pre-distribution Scheme for Wireless Sensor Networks", GCC Workshops, vol. 00, pp. 437-443 (2006)

[21] Zhao, S., Tepe, K., Seskar, I., Raychaudhuri, D., "Routing Protocols for Self-Organizing Hierarchical Ad Hoc Wireless Networks", Proceedings of the IEEE Sarnoff Symposium (2003)

[22] Zhu, S., Setia, S., Jajodia, S., "LEAP: Efficient Security Mechanisms for Large-Scale Distributed MSN Networks", Proceedings of ACM conference on Computer and Communication Security, pp. 62-72 (2003)

[23] Zoltak, B., "Tail-MAC: An Efficient Message Authentication Scheme for Stream Ciphers", Cryptology ePrint Archive, Report 048 (2004)