# A KEY MANAGEMENT PROTOCOL FOR HIERACHICAL WIRELESS SENSOR NETWORKS

Pawan Kumar Goel
Mewar University Rajasthan, Department of CSE,
Rajasthan, India
Email id : pgoel0115@gmail.com

Vinit Kumar Sharma
Associate Professor Department of Mathematics & CSE,SIET,
Uttar Pradesh, India
Email id: vksharmaxyz_1@rediffmail.com

**Abstract :** Wireless sensor networks (WSNs) are deployed in hostile environments in many applications. In order to resist security threats, sensor nodes of WSNs often use pre-shared secret keys to encrypt and exchange confidential data. Accordingly, designing key management protocols that can securely distribute secret keys among sensor nodes becomes an important issue for WSNs. This paper proposes a novel key management protocol for hierarchical WSNs based on hypergraph. In the proposed protocol, a WSN is viewed as a hypergraph. That is, a sensor node is represented as a vertex and a cluster is represented as a hyperedge. Compared to previous investigations, the proposed protocol possesses the following features. (1) Scalability. No matter how many sensor nodes and how many clusters are in a WSN, the proposed protocol can use constant communication rounds to establish all cluster keys. Therefore, it is especially suitable for resource-constrained large-scale WSNs. (2) Applicability. In the proposed protocol, all sensor nodes can be deployed randomly and establish cluster keys without knowing the topology of the whole network in advance. (3) Flexibility. This paper also presents dynamic insert and remove protocols. The dynamic insert protocol allows newly deployed sensor nodes to join an existing WSN while the dynamic remove protocol can delete compromised sensor nodes from a WSN. (4) Robustness. The proposed protocol can resist node capture attacks, node cloning attacks, wormhole attacks and energy consumption attacks.

**Keyword-** Hypergraph, Wireless Sensor Network, Key Management

## I. INTRODUCTION

Due to the development of micro-electro-mechanical systems (MEMS), wireless sensor networks (WSNs) become realistic in these years. A wireless sensor network is composed of a large amount of sensor nodes which can sense variations of current environment and transmit data to the base station by using wireless communication. Sensor nodes can be randomly deployed in inaccessible place so that wireless sensor networks are useful for a broad spectrum of emerging civil and military applications (Akyildiz, Su, Sankarasubramaniam, & Cayirci, 2002), such as object tracking, environment monitoring and data gathering. However, since sensor nodes are often deployed in hostile environments, wireless sensor networks are confronted with many security attacks. For example, an adversary can intercept the communications between sensor nodes. It can also broadcast some imitation messages to interfere a wireless sensor network. Moreover, the adversary can easily capture and compromise the secret keys of sensor nodes due to the sensor nodes are not tamper-proof devices. Therefore, designing secure key management protocols for WSNs is a desirable task that attracts many cryptographers.

Traditionally, key management protocols can be classified into three categories:

1. Symmetric key based key management protocol. The communication entities use a pre-shared symmetric key to negotiate a temporary session key. Then, they use this session key to encrypt messages and authenticate one another.
2. Asymmetric key based key management protocol. Each communication entity has its own public key and private key pairs. The communication entities may apply signature schemes (Rivest, Shamir, & Adleman, 1978) to authenticate each other and utilize the Diffie-Hellman key exchange scheme (Diffie & Hellman, 1976) to produce the session key for secure communications.
3. Trusted third party based key management protocol. Each communication entity shares a symmetric key with a trusted third party. Then, the communication entities can achieve mutual authentication and secure communication through the trusted third party's assistance.

Reviewing the above categories of key management protocol, symmetric key based key management protocols are more suitable for WSNs than the other categories. Due to the resource constraints of sensor nodes, asymmetric key based key management protocols are too complex and energy consuming for WSNs. This is because they require exponential computations. On the other hand, the sensor nodes are often spread in wide area so that many sensor nodes may be deployed too far away to communicate with the trusted third party. Therefore, trusted third party based key management protocols are also hard to implement in WSNs.

The most common network architectures of WSNs are flat WSNs and hierarchical WSNs (Shen, Guo, & Leung, 2009). All senor nodes play the same role in a flat WSN. Under this architecture, a sensor node may randomly select a neighbor node to forward the data. Thus, the data transmission path may not be fixed. Nevertheless, a hierarchical WSN requires some special sensor nodes, called Cluster Head (CH). A CH is a data collection center of a small region in a WSN. Under this architecture, each sensor node can forward data to its local CH in a short distance, and then the CH forward collected data to the BS using a fixed path via other CHs. The manuscript (Cheng & Agrawal, 2007) showed that the communication of a hierarchical architecture is better than a flat architecture in WSNs. Therefore, several recent studies focus on designing key management protocols of hierarchical WSNs.

To deal with the key management problem in hierarchical WSNs, several investigations were proposed in these years. In 2003, Jolly et al. proposed a key pre-distribution scheme LEKM (Jolly, Kuscu, Kokate, & Yuonis, 2003). Their scheme has no computational cost at both sensor node and CH by using the key pre-distribution mechanism. However, the CHs require high storage and communication overhead during the deployment phase. Later, Cheng et al. (Cheng & Agrawal, 2007) proposed IKDM, a polynomial-based protocol to overcome this problem. In the IKDM protocol, sensor nodes and CHs have fixed storage cost in pre-distribution phase. Thus, it is suitable for large-scale hierarchical WSNs. In 2009, Shen et al. (Shen, Guo, & Leung, 2009) presented a renewable key management protocol for hierarchical WSNs. Their protocol additionally provides key establishment for adding new deployed sensor nodes which is more flexible than previous schemes. However, the above protocols have a common weakness. That is, they do not provide a mechanism to discard the compromised nodes. Once an adversary captures some sensor nodes and compromises the secret keys, the adversary can intercept the communications of the WSN and even deploy some cloned nodes to interfere with the WSN.

(2) Applicability. In the proposed protocol, all sensor nodes can be deployed randomly and establish cluster keys without knowing the topology of the whole network in advance. (3) Flexibility. This paper also presents dynamic insert and remove protocols. The dynamic insert protocol allows newly deployed sensor nodes to join an existing WSN while the dynamic remove protocol can delete compromised sensor nodes from a WSN. (4) Robustness. The proposed protocol can resist node capture attacks, node cloning attacks, wormhole attacks and energy consumption attacks.

The rest of this paper is organized as follows. Section 2 summarizes the notations and underlying primitives. Section 3 demonstrates the proposed hypergraph based key management protocol. Section 4 evaluates the security of the proposed protocol. A brief conclusion is presented in Section 5.

## II. PRELIMINARIES

In this section, we summarize the notations and the underlying primitives used throughout this paper. We first describe notations and then illustrate cryptographic primitives used in the proposed protocol.

**Notations**

The notations of the proposed protocol are outlined as the following table.

Table I: Notations

| | |
|---|---|
| $BS$ | The id of the base station. |
| $CH_j$ | The id of cluster header in cluster *j*. |
| $S_i$ | The id of sensor node *i*. |
| $e_j$ | The id list of nodes in cluster *j*. |
| $G_r$ | The set of *r*-th deployed nodes, where $1 \le r \le n$. |
| $MK_r$ | The master key of the *r*-th deployed nodes, where $1 \le r \le n$. |
| $IK_i^r$ | The individual key of the sensor node $S_i$ used in the *r*-th deployment. |
| $CK_j^r$ | The cluster key of the cluster *j* used in the *r*-th deployment. |
| $GF(q)$ | A finite field of order *q*, where *q* is large prime number which can accommodate cryptographic secrets. |

**A. Symmetric encryption scheme**

The symmetric encryption scheme $\Gamma = \{E, D\}$ consists of two algorithms.

- Encryption algorithm $E_K(m)$ : This algorithm encrypts plaintext *m* using secret key *k*.
- Decryption algorithm $D_K(c)$ : This algorithm decrypts ciphertext *c* using secret key *k*.

**B. One-way hash function**

$$H(x): \{0,1\}^* \rightarrow GF(q).$$

$H(x)$ is a collision-free one-way hash function. The one-way hash function $H(x)$ can map arbitrary bit string to the finite field $GF(q)$ in this investigation.

**C. Bivariate polynomial function**

The bivariate polynomial function was first introduced in (Blundo, Santis, Herzberg, Kutten, Vaccaro, & Yung, 1993). A *t*-degree bivariate polynomial function $f(x, y)$ is defined as

$$f(x, y) = \sum_{i,j=0}^{t} a_{ij} x^i y^j$$

where the coefficients $a_{ij} (0 \le i, j \le k)$ are randomly from *GF*(*q*). The bivariate polynomial function has the symmetric property. That is $f(x, y) = f(y, x)$. In addition, a *t*-degree bivariate polynomial function can keep secure if the number of compromised values of the polynomial $f(x, y)$ is less or equal then *t*, as shown in (Blundo, Santis, Herzberg, Kutten, Vaccaro, & Yung, 1993).

**D. Hypergraph based key management scheme**

A hypergraph is a mathematic graph structure that is defined as *G*(*V*, *HE*), where $V = \{s_1, s_2, ..., s_n\}$ is a set of vertices and $HE = \{e_1, e_2, ..., e_m\}$ is a set of hyperedge. A hyperedge $e_i = \{s_{i_1}, s_{i_2}, ..., s_{i_k}\}$ is a nonempty subset of *V*. In practical, the network topology of a WSN is very similar to a hypergraph. A sensor node can be viewed as a vertex and a cluster can be represented as a hyperedge. Therefore, A hypergraph based key management scheme can be easily applied to a WSN. In 2007, (Jeong & Lee, 2007) introduced a simple key management scheme for a hypergraph which can be complete in two rounds. Here we briefly illustrate their

concept. Given a hyperedge $e = \{s_1, s_2, ..., s_m\}$, the vertices $s_1, s_2, ..., s_m$ can establish a cluster key as follows.

R1: For each $s_i \in e$, $s_i$ first computes $k_{i-1,i}$, $k_{i,i+1}$ and $C_i = k_{i-1,i} \oplus k_{i,i+1}$, where $k_{i-1,i}$ is the pairwise key shared with $s_{i-1}$ and $k_{i,i+1}$ is the pairwise key shared with $s_{i+1}$, respectively. After that, $s_i$ broadcasts $e \| C_i$ to other vertices.

R2: After receiving all messages from other vertices, $S_i$ can use $k_{i-1,i}$ and $k_{i,i+1}$ to get all pairwise keys $k_{1,2}, k_{2,3}, ..., k_{m,1}$. Finally, $S_i$ computes $CK = H(e, k_{1,2}, k_{2,3}, ..., k_{m,1})$ as the cluster key.

## III. THE PROTOCOL

This section proposes a hypergraph based key management protocol for hierarchical WSNs. The proposed protocol consists of the setup phase, the initial deploy phase, the dynamic insert phase, and the dynamic remove phase. These four phases are demonstrated as follows.

### A. Setup phase

For the system setup, the *BS* divides the sensor nodes into *n* deployments. The *BS* first selects *n* random numbers $MK_1, MK_2, ..., MK_n$ as the master keys of the WSN. Subsequently, the *BS* also prepares *n* bivariate polynomial functions $f_1(x, y), f_2(x, y), ..., f_n(x, y)$ and randomly selects a system parameter $s_n$. Let $G_r$ be the set of *r*-th deployed nodes, for some $1 \le r \le n$. For each sensor node $S_i \in G_r$, $S_i$ computes its individual keys $IK_i^j = H(S_i, MK_j)$ where $r \le j \le n$. $S_i$ then preloads its master key $MK_r$, the system parameter $s_n$, its individual keys $IK_i^j$, and the bivariate polynomial functions $f_j(x, y)$, where $r \le j \le n$.

### B. Initial deploy phase

The initial deploy phase can be complete in four rounds.

R1: After deployment, a sensor node $S_i$ broadcasts the message $E_{MK_1}(S_i)$ and decrypts the messages broadcasted by neighbor nodes.

R2: After the first round, the cluster header $CH_j$ can decide which sensor nodes belong to the cluster *j*. Then $CH_j$ prepares the member list $e_j$ and broadcasts the message $E_{MK_1}(e_j)$ to inform its members. Subsequently, $S_i$ can decrypt the messages sent by the neighbor cluster headers and know which clusters contain it.

R3: For each cluster *j* contains $S_i$, $S_i$ computes $k_{i-1,i}^j = f_1(S_{i-1}, S_i)$, $k_{i,i+1}^j = f_1(S_{i+1}, S_i)$ and $C_i^j = k_{i-1,i}^j \oplus k_{i,i+1}^j$. $S_i$ then concatenates all $e_j \| C_i^j$ and gets $C_i = ... \| e_j \| C_i^j \| ...$. Finally, $S_i$ broadcasts the message $C_i$ and decrypts the messages broadcasted by neighbor nodes.

R4: For each cluster *j* contains $S_i$, $S_i$ can get $k_{1,2}^j, k_{2,3}^j ... k_{m,1}^j$ from all $C_*^j$, where $m = | e_j |$. Subsequently, $S_i$ computes the cluster key $CK_j^1 = H(s_n, e_j, k_{1,2}^j, k_{2,3}^j ... k_{m,1}^j)$. Finally, $S_i$ erases the polynomial functions $f_1(x, y)$ and the master key $MK_1$.

### C. Dynamic insert phase

Suppose the sensor nodes in $G_r$ are just deployed in the *r*-th deployment. Then the dynamic insert phase works as the following five rounds.

R1: For each previously deployed cluster header $CH_j$, it broadcasts its id $CH_j$.

R2: After the first round, the new deployed sensor node $S_i \in G_r$ can discover is neighbor cluster headers. For each neighbor cluster header $CH_j$, $S_i$ can derive $CH_j$'s current individual key $IK_j^r = H(CH_j, MK_r)$. Then, $S_i$ encrypts and sends the message $E_{IK_j^r}(S_i)$ to $CH_j$.

R3: When receiving the messages sent by new deployed sensor nodes, the cluster header $CH_j$ can use $IK_j^r$ to decrypt the messages and renew the member list $e_j$. Then $CH_j$ computes and broadcasts the message $(E_{CK_j^{r-1}}(e_j) \| E_{IK_j^r}(e_j))$ to inform its members. Subsequently, the old members can get $e_j$ by decrypting $E_{CK_j^{r-1}}(e_j)$ and the new members can get $e_j$ by decrypting $E_{IK_j^r}(e_j)$.

R4: For each cluster $j$ contains $S_i$, $S_i$ computes $k_{i-1,i}^j = f_r(S_{i-1}, S_i)$, $k_{i,i+1}^j = f_r(S_{i+1}, S_i)$ and $C_i^j = k_{i-1,i}^j \oplus k_{i,i+1}^j$. $S_i$ then concatenates all $e_j \| C_i^j$ and gets $C_i = ... \| e_j \| C_i^j \| ...$. Finally, $S_i$ broadcasts the message $C_i$ and decrypts the messages broadcasted by neighbor nodes.

R5: For each cluster $j$ contains $S_i$, $S_i$ can get $k_{1,2}^j, k_{2,3}^j, ..., k_{m,1}^j$ from all $C_*^j$, where $m = | e_j |$. Subsequently, $S_i$ computes the cluster key $CK_j^r = H(s_n, e_j, k_{1,2}^j, k_{2,3}^j, ..., k_{m,1}^j)$. Finally, $S_i$ erases the polynomial functions $f_1(x, y)$ and the master key $MK_1$.

### D. Dynamic remove phase

Suppose the BS can discover some sensor nodes have been compromised in some manner. Then the BS can initiate the dynamic remove protocol to protect the good sensor nodes as follows.

R1: The *BS* prepares a black list ***B*** which includes the ids of compromised nodes and regenerate a new system parameter $s_n'$. Then, the *BS* computes and sends $E_{IK_i^r}(\boldsymbol{B}, s_n')$ to a sensor node $S_i$ for every $S_i \notin \boldsymbol{B}$.

R2: After receiving the *BS*'s message, A good sensor nodes $S_i$ can use its current individual key $IK_i^r$ to decrypt $E_{IK_i^r}(\boldsymbol{B}, s_n')$. For each cluster $j$ contains $S_i$, $S_i$ updates the member list $e_j$ by discarding the nodes in ***B***. $S_i$ also computes the new cluster key $CK_j^r{}' = H(CK_j^r, s_n')$ for each cluster $j$. Finally, $S_i$ updates the stored system parameter as $s_n'$.

## IV. SECURITY ANALYSIS

This section evaluates the security of the proposed protocol. A secure key management protocol of WSNs should resist node capture attacks, node cloning attacks, wormhole attacks and energy consumption attacks. In order to show the proposed protocol can resist the above attacks, we need to make a critical assumption (Zhu, Setia, & Jajodia, 2006). The assumption assumes that each deployment of sensor nodes can be done within time $T_{\min}$, where $T_{\min}$ is the minimal time that an adversary can compromise a sensor node. Based on this assumption, an adversary has no chance to obtain the erased secrets. The security analysis of the proposed protocol is described as follows.

### A. Node capture attacks

Node capture attacks are the main threats of WSNs. Sensor nodes are easily to be captured and compromised because they are often deployed in hostile environments and not tamper resistant. Therefore, if an adversary captures some sensor nodes and obtains the secret data of the nodes, a secure key management protocol for WSNs should provide some mechanisms to mitigate the risk.

If an adversary captures a sensor node $S_i$ in the *r*-th deployment time interval, it can get its individual keys, cluster keys and current system parameter $s_n$ in the proposed protocol. Although the adversary can get current transmissions of clusters that contain $S_i$, the adversary has no chance to get individual keys and cluster keys of $S_i$ before the *r*-th deployment time interval. Since these previous keys are erased already. Thus, the proposed protocol provides *weak backward secrecy*. Moreover, when the BS detects that $S_i$ has been captured, the BS will launch the *dynamic remove protocol* immediately. In the dynamic protocol, the BS will send a black list ***B***

and the new system parameter $s_n'$ to all non-captured nodes. Since $S_i$ is in $\boldsymbol{B}$, the adversary cannot get $s_n'$ to compute future clusters keys. Therefore, the proposed protocol provides *weak forward secrecy*.

### B. Node cloning attacks

In node cloning attacks, an adversary prepares its own node with the keys of a compromised node, and then deploys this node in different locations in the WSN. The proposed protocol can clearly resist node cloning attacks. Suppose the adversary deploys a cloned node in the *r*-th deployment time interval. This new deployed node must use the master key $MK_r$ to compute $CH_j$'s individual key $IK_j^r = H(CH_j, MK_r)$ in R2 of the dynamic insert phase. However, since all sensor nodes erase their master keys after deployments, the adversary has no chance to get $MK_r$. Therefore, node cloning attacks cannot work in the proposed protocol.

### C. Wormhole attacks

In wormhole attacks, a malicious node tries to cheat its neighbors that it is very near the base station. Thus, the malicious node can collect and drop all data from the neighbors and let the real base station cannot get any information from that area. Typically, the wormhole attacks require two distant malicious nodes, which have an invisible link underlying sensor network. The adversary deploys one malicious node close to the base station and the other close to the interested area. The adversary could convince the nodes of the interested area that the malicious node is near the base station by using the invisible link.

The proposed protocol can easily detect wormhole attacks. After deployment, a sensor node has all member lists of its clusters. Thus, every sensor node can know the network topology. The adversary has no way to interpolate fake sensor nodes.

### D. Energy consumption attacks

Due to the limited battery volume of sensor nodes, security protocols for WSNs must be energy efficient. However, the adversary can deploy some malicious nodes which repeatly broadcast non-sense messages to consume the energy of normal nodes.

In order to join a WSN, new deployed nodes must know the secret keys to compute correct cluster keys in the proposed protocol. Thus, if the adversary does not compromise a deployed sensor node and get the secret keys, it cannot launch energy consumption attack. On the other hand, if the adversary compromise a sensor node and use it to perform energy consumption attack, the neighbors of the malicious node can detect this attack and warn the base station. After that, the base station will launch the dynamic remove protocol to omit the malicious node. Therefore, energy consumption attacks fail in the proposed protocol.

## V. CONCLUSION

This study proposes a novel key management protocol for hierarchical WSNs based on hypergraph. In the proposed protocol, the base station divides the sensor nodes into *n* groups and randomly deploys them in *n* time intervals. The base station needs not to manage the locations of the nodes before deployments. Hence, the proposed protocol has the feature of applicability. By using the key management technique of hypergraph, no matter how many sensor nodes and how many clusters are in a WSN, the sensor nodes can use constant communication rounds to establish all cluster keys. Therefore, this protocol is especially suitable for resource-constrained large-scale WSNs. To provide flexibility, this study also presents dynamic insert and remove protocols. The dynamic insert protocol allows newly deployed sensor nodes to join an existing WSN while the dynamic remove protocol can delete compromised sensor nodes from a WSN. Finally, this investigation shows that the proposed protocol can resist node capture attacks, node cloning attacks, wormhole attacks and energy consumption attacks.

## REFERENCES

[1] Akyildiz, I. F., Su, W., Sankarasubramaniam, Y., & Cayirci, E. (2002). Wireless Sensor Networks: A Survey. Computer Networks 38 , pp. 393-422.

[2] Blundo, C., Santis, A. D., Herzberg, A., Kutten, S., Vaccaro, U., & Yung, M. (1993). Perfectly-Secure Key Distribution for Dynamic Conferences. Cryptology – CRYPTO '92, LNCS 740 , pp. 471-486.

[3] Camtepe, S., & Yener, B. (2005). Key Distribution Mechanisms for Wireless Sensor Networks: A Survey. Technique Report TR-05-07, Department of Computer Science, Rensselaer Polytechnic Institute .

[4] Chan, H., Perrig, A., & Song, D. (2004). Key Distribution Techniques for Sensor Networks. Wireless Sensor Networks , pp. 277-303.

[5] Cheng, Y., & Agrawal, D. P. (2007). An improved key distribution mechanism for large-scale hierarchical wireless sensor networks. Ad Hoc Networks , 5 (1), pp. 35–48.

[6] Diffie, W., & Hellman, M. E. (1976). New directions in cryptography. IEEE Transactions on Information Theory , 22 (6), pp. 644–654.

[7] Du, W., Deng, J., Han, Y., & Varshney, P. (2006). A Key Management Scheme for Wireless Sensor Networks Using Deployment Knowledge. IEEE Transactions on Dependable and Secure Computing, 3, pp. 62-77.

[8] Du, W., Deng, J., Han, Y., Varshney, P., Katz, J., & Khalili, A. (2005, 5). A Pairwise Key Predistribution Scheme for Wireless Sensor Networks. ACM Transactions on Information and System Security (TISSEC) , 8 (2), pp. 228-258.

[9] Huang, D. M. (2004). Location-Aware Key Management Scheme for Wireless Sensor Networks. 2nd ACM workshop on Security of ad hoc and Sensor Networks, (pp. 29-42). Jeong, I., & Lee, D. (2007). Key agreement for key hypergraph. Computers & Security , 26 (7-8), pp. 452-458.
[10] Jolly, G., Kuscu, M. C., Kokate, P., & Yuonis, M. (2003). A lowenergy management protocol for wireless sensor networks. 8th IEEE International Symposium on Computers and Communication (ISCC '03), (pp. 335–340). Kemer-Antalya, Turkey.
[11] Liu, D., Ning, P., & Du, W. (2005, 9). Group-Based Key Pre-Distribution in Wireless Sensor Networks. 2005 ACM Workshop on Wireless Security (WiSe 2005) , pp. 11-20.
[12] Park, T., & Shin, K. G. (2004, 8). LiSP: A Lightweight Security Protocol for Wireless Sensor Networks. ACM Transactions on Embedded Computing Systems (TECS) , 3 (3), pp. 634-660.
[13] Rivest, R., Shamir, A., & Adleman, L. (1978). A method for obtaining digital signatures and public-key cryptosystems. Communications of the ACM , 21 (2), pp. 120–126.
[14] Shen, A.-N., Guo, S., & Leung, V. (2009). A Flexible and Efficient Key Distribution Scheme for Renewable Wireless Sensor Networks. Hindawi Publishing Corporation EURASIP Journal on Wireless Communications and Networking .
[15] Walters, J. P., Liang, Z., Shi, W., & Chaudhary, V. (2006). Wireless Sensor Network Security: A Survey. (Y. Xiao, Ed.) Security in Distributed, Grid, and Pervasive Computing .
[16] Yu, Z., & Guan, Y. (2005). A Key Pre-distribution Scheme Using Deployment Knowledge for Wireless Sensor Networks., (pp. 261-268).
[17] Zhu, S., Setia, S., & Jajodia, S. (2006). LEAP+: Efficient security mechanisms for large-scale distributed sensor networks. ACM Trans. on Sensor Networks , 2 (4), pp. 500-528.