A Key Distribution and Management Scheme for Hierarchical Wireless Sensor Network

Md. Ibrahim Abdullah
Department of Computer Science and Engineering,
Islamic University, Kushtia, Bangladesh.
ibrahim25si@yahoo.com

Abstract

The nodes of Wireless Sensor Networks (WSNs) are deployed in hostile environments faces the security problem of eavesdropping and capturing by an adversary. To achieve the security of this resource constraint device is a challenging task. This paper proposed a secured key management scheme for hierarchical WSNs that distributed the keys within a cluster efficiently and update the pre-deployed keys to mitigate the node compromised attack. We use one way hash function and random variable that broadcasted to update the pre-deployed keys.

Key Words: *Wireless Sensor Network, Hierarchical routing, Security, Key Distribution, Rekeying, Authentication,*

1. Introduction

Wireless Sensor Networks (WSN) is simply defined as a large collection of sensor nodes, each equipped with its own sensors, processor and radio transceiver. Their low cost provides a means to deploy large sensor arrays in a variety of conditions capable of performing both military and civilian tasks. Sensor nodes are typically characterized by limited power supplies, low bandwidth, small memory sizes and limited energy [1].

When homogeneous sensor nodes are deployed in open and unattended environments without physical protection, security becomes extremely important, as they are prone to different types of malicious attacks. For example, an adversary can easily listen to the traffic, impersonate one of the network nodes, or intentionally provide misleading information to other nodes. However, due to inherent resources and computing constraints, security in sensor networks poses different challenges than conventional network security. One security aspect that receives a great deal of attention in wireless sensor networks is the area of key management.

Key management is the process in which keys are created, stored, protected, transferred, used between authorized parties and destroyed when they do not need [2]. Key management establishes the keys necessary to provide confidentiality, integrity and authentication services. However, providing key management in WSNs is difficult due to the unknown network topology prior to deployment, intermittent connectivity and resource limitations of the sensor nodes. The main goal of key management in WSNs is the establishment of secure links between neighbor nodes at network formation phase. It also ensures security requirements of WSN by encrypting messages and authenticates the communicating nodes. Therefore, it is a major concern how to secure communications between sensor nodes [3].

Currently, there are three general key agreement schemes: trusted-server or arbitrated protocol, self-enforcing, and key pre-distribution scheme. The trusted server scheme depends

on a trusted server e.g., Kerberos [4]. Since there is no trusted infrastructure in sensor networks, the trusted-server scheme is fundamentally unsuited to them. Another key agreement scheme is the self-enforcing scheme, which depends upon asymmetric protocols and algorithms. However, with the low memory and energy constraints of sensor nodes, public-key algorithm, which is commonly used in asymmetric cryptography, limits the practical use of this key distribution scheme. Presently, the only practical scheme for key distribution in large sensor networks is key pre-distribution, where key information is installed in each sensor node prior to deployment.

Typically, two distribution techniques have been used in WSNs: *i)* a single mission key where all nodes carry a master secret key or *ii)* a set of separate $n - 1$ keys, each being a pairwise set that is privately shared with another sensor node [5]. Both are inadequate for use in sensor networks since the capture of one node will compromise the whole network for single mission key and storage of $n - 1$ keys in each sensor node bounds practical adoption. Storing the master key in tamper resistant sensor nodes increases the cost and energy consumption of the sensors nodes.

Key pre-distribution only cannot achieve satisfactory security performance. Because wireless communication is broadcasted transmission in nature, an adversary can eavesdrop on all traffic. If these keys do not reconfigure for long time, an adversary can retrieve the keys by listening the wireless medium and breakdown the data packets using brute-force or dictionary-attack [6]. Moreover, as the nodes operate in an uncontrolled environment, an adversary may compromise nodes and capture the pre-deployed secret keys. To defend the sensor network from such security threats the nodes need forward security [7], i.e. if a node is captured and its secret material compromised, an adversary should not be able to decrypt messages that were intercepted by the adversary in the past. Therefore, after key pre-distribution and sensor deployment, a key updating scheme should be used to update pre-deployed keys regularly. This procedure ensures that enemies cannot acquire the keys easily.

This paper presents a key management scheme for hierarchically organized wireless sensor networks [8] to mitigate the security requirements. Some nodes of this architecture are grouped to form a cluster. Leader node of a cluster called cluster head (CH) are used to process and send information to the base station while the others are used to perform sensing. In this proposed key management scheme, CH and its member nodes distributed the pre-deployed keys asymmetrically and ensures the authentication of nodes and data integrity. All nodes of the network periodically updated their pre-deployed keys to assure that only legitimate nodes send data for processing. The key updating scheme uses a random number that broadcasted by BS and one way hash function [6].

The rest of the paper organized as follows. Section 2 describes the related work of security for WSN. Terms and notations used in this paper are listed in section 3. Section 4 explains the network model used in this work. Some assumptions about security are described in section 5. Section 6 explains the proposed key management scheme in details. Section 7 presents the analysis of the proposed key management scheme. The simulation results concerning communication expenses and storage overhead are discussed in section 8. Finally in section 9, we present our concluding remarks and future work.

2. Related Works

Security problems of sensor network against different layers of network architecture are extensively discussed in [10]. Karlof [11] outline the possible attacks and threats on WSN because of their simplicity and resource constrains. Asymmetric encryption or public key cryptography has been thought to be far too heavy weight for use in wireless sensor networks.

Watro *et al.* [12] shows that portions of the RSA cryptosystem can be successfully applied to actual wireless sensors, specifically the UC Berkeley MICA2 motes [13].

Chen *et al.* [14] proposed two security protocols. First, base station to node confidentiality and authentication which states that an efficient shared-key algorithm like RC5 be used to guarantee the authenticity and privacy of information. Second, the source authentication, by implementing a hash chain functions similar to that used by TESLA (Timed Efficient Stream Loss-tolerant Authentication) to achieve node authentication. Eschenauer *et al.* [5] proposed a probabilistic key pre-distribution technique to bootstrap the initial trust between sensor nodes. First, each sensor randomly picks a set of keys from a key pool before deployment. Then, in order to establish a pairwise key, two sensor nodes only need to identify the common keys that they share.

Du *et al.*[15] present a key scheme based on deployment knowledge. This key management scheme takes advantage of the deployment knowledge where sensor position is known prior to deployment. Because of the randomness of deployment, it is not feasible to know the exact neighbor locations, but knowing the set of likely neighbors is realistic, this issue is addressed using the random key pre-distribution of Eschenauer [5]. Du also present pairwise key pre-distribution [16] is an effort to improve the resilience of the network by lowering the initial payoff of smaller scale network attacks and pushes adversary to attack at bigger scale to compromise the network. Adrian et al [17] have introduced SPINS; a collection of security protocols SNEP and µ-TESLA. SNEP (Secure Network Encryption Protocol) provides data confidentiality and two-way data authentication with minimum overhead. µ-TESLA, a micro version of TESLA provides authenticated streaming broadcast. SPINS leaves some questions like security of compromised nodes, DoS issues, network traffic analysis issues.

Zia, T.A, [18] introduce triple key scheme without third party trusted authority and consisting of three keys: two pre-deployed keys in all nodes and one in network generated cluster key for a cluster to address the hierarchical nature of sensor network. This scheme requires each node to have mutual authentication with its neighbors and cluster leaders. The encryption and authentication technique, TinySec [19], was developed as a first attempt to introduce security to the link layer of the TinyOS suite. This was done by incorporating software-based symmetric keying with low computing and storage overhead.

Jeong [20] proposed a secure communications and key management scheme to remove the compromised node from the network and employs a multi-tier network architecture in which secure sessions are established only between sensor nodes and gateways. This is based on the theory of combinatorial optimization and provides an approach to maintain security while members have changed in groups.

3. Terms and Notations Used

The following terms and notations are used in the proposed key management technique.

Table 1: Notation Description

Notation	Description
ID_i	Identification Number of node *i*
K_{si}	A Secret key of node *i* which is put in each sensor node before deployment
K_N	Network-Key, embedded in each sensor node before deployment and share by the entire sensor network

M	Message to transmit
$H()$	A one way hash function
$E_K(M)$	Encryption of message M with key K
$MAC(M)$	The message authentication code of message M using key K
R_t	A random number broadcast by BS at time t
$\|$	Concatenation operator
$\oplus$	Bit wise XOR operation
t_m	Minimum time after deployment a node cannot capture
T_{ch}	Duration of Cluster
T_{cap}	Time need to capture a node

4. Network Model

In this work, we considered the hierarchical structure of sensor network. We assume that WSNs are homogeneous (all nodes contain same hardware and software), symmetric (node *A* can only communicate with node *B* if and only if *B* can communicate with *A*). Node position is random in sensor field. Nodes are static *i.e.* its position do not change after deployment. There is a Base station (BS) for processing the sensed data.

After deployment, some nodes are randomly selected as CH. Other nodes choose their leader based on some parameters such as the strongest signal received from a CH [8]. The communication between member nodes and CH is single hop, but CH to BS may be multihop when BS located far way from CH. To reduce the energy consumption of a CH, after certain time interval new nodes are select as CH. Rotating CHs have the advantage of averaging energy consumption among sensor nodes [8].

5. Security Assumptions

We make the following reasonable assumptions just as in most of the current sensor network security schemes [5], [14], [15], [18], [21], [22]:

- Each sensor has a unique ID with enough length to distinguish between them.
- BS has a node member table of node ID and corresponding Session-Key. If a node adds to network, it's ID and secret keys add to node member table.
- We also consider a minimum time t_m after deployment a node cannot be compromised. (It is reasonable assumption that on presence of deployment system/agent an adversary cannot capture any node).
    - Within this time the nodes are able to construct a neighbor table and create a cluster.
- BS has authentication system [22] for any node in the network.
- We assume that an adversary need at least time T_{cap} to capture a node, connect with a computer and extract information from that node [21].
- When multihop transmission needed between CH and BS, the intermediate relay nodes do not encrypt the CH messages using their key. They simply transmit packets to other relay nodes or BS.

6. The Proposed Hierarchical Key Management Scheme

In this hierarchical key management scheme, each node consist two types of keys:

(1) A Network-Key (K_N): This is a globally shared key that is used by all nodes and BS for encrypting messages that are broadcast to the all node in the sensor network. This key is also used in cluster formation and authentication of a group of nodes.

(2) A Session-Key (K_{si}) is used by a node *i* to secure the transmission of data packets to it cluster leader.

Keys are programmed into the memory of the sensor nodes just before they are being deployed. The keys are stored in the flash RAM and hence can be deleted when required. The proposed model divided into four phases: (i) Deployment phase (ii) Secure cluster setup phase, (iii) Data transmission phase and (iv) Key updating phase.

6.1. Deployment Phase

At deployment phase nodes create neighbor database within time T_{min}. In this stage a node sends Hello message to neighbors with ID. The neighbor nodes within radio range reply with their ID. The node creates a table of neighbors. In this part any node does not share any key information with neighbor nodes.

6.2. Secure Cluster Setup Phase

In this step, a new key distribution technique is proposed for sensor network that distributed secret keys without transmitting any secret key in wireless medium [23]. This technique works similar to public key cryptography but without any costly cryptographic operation. There are two distinct steps in secure cluster setup phase: (a) Create Members Key Table of a Cluster and (b) Authentication of CH and its Member Nodes.

6.2.1. Members Key Table of a Cluster: When first time a node *j* elect or select as CH, it sends an authentication packet to BS by inserting its ID and encrypting message using its secret key K_{sj}.

ID_j	$E_{sj}(M)$	$MAC_{sj}(M)$

BS obtains the secret key of node *j* using its ID_j from its node member table and decrypts the CH message. Once BS authenticates a CH, it broadcasts encrypted CH advertisement using Network key. Nodes receive all CH advertisements within its radio range and record the CH IDs and signal strength of received advertisements. Among the receiving advertisements a node choose the strongest signal value as their CH. Node sends a message to the selected CH about its membership using own session key K_s. After receiving all membership messages a CH build a cluster-member table.

Consider a node ID is ID_i and its CH ID is ID_j. For membership of this cluster, a node generate a message *M* using own secret key K_{si} and network key K_N as follows:

$$M = ((ID_i \,||\, ID_j \,||\, K_N) \oplus K_{si})$$

Now node *i* encrypt the message M using network key K_N and sends the encrypting message to CH. The packet contains following fields.

ID_i	$E_{KN}(M)$	$MAC_{KN}(M)$

where $MAC_{KN}(M)$ is message authentication code [6] using the network key K_N.

A MAC algorithm can be generated using multiple different techniques, as long as the sender and receiver have shared secret keys. If the node ID is in neighbor list that has created in deployment phase, CH decrypts the packet using K_N and recovers the secret key K_{si} of node i as follow:

$$M \oplus K_N = ((ID_i \,||\, ID_j \,||\, K_{si})$$

Now CH creates its members list table including node ID and secret key. Now all nodes discard the globally shared network key K_N. It is rebuilt in key-updating phase.

6.2.2. Authentication of Member Nodes of a Cluster: After collecting all member nodes secret key CH compute an authentication code for itself and its member nodes using one way hash function [24] for its n members as:

$$H_1 = H(K_{s1}, K_{sj})$$
$$H_2 = H(K_{s2}, H_1)$$
$$\ldots\ldots\ldots\ldots\ldots\ldots$$
$$H_n = H(K_{sn}, H_{n-1})$$

where K_{sj} is CH own secret key and $K_{s1}, K_{s2}\ldots K_{sn}$ are the secret keys of node $ID_1, ID_2 \ldots ID_n$

Now CH sends it's all member ID and hash value H_n to BS by encrypting using its secret key K_{sj}. Since BS knows all ID and their key, it can compute H_n and authenticate this group of nodes.

6.3. Data Transmission Phase

In hierarchical model of sensor network data transmission consists two distinct steps. In first step member nodes send their sense data to their CH. A member node encrypts data packets using its own secret key K_{si}. Now K_{si} is the session key to communicate with CH. As CH gets all secret keys of its members, it can decrypt message from any node i if it is member of the cluster. The data packets format is as follows:

ID_i	$E_{Ksi}(M)$	$MAC_{Ksi}(M)$

where M is the sense data, $E_{Ksi}(M)$ is the encrypted message and $MAC_{Ksi}(M)$ is the message authentication code

When CH sends data packets to BS for processing it encrypts the message using its own session key and insert its ID and encrypted message into the data packet. If the distance between CH to BS is multihop, intermediate nodes simply relay the data packets to BS.

6.4. Key Updating Phase

In an open area a node can easily be compromised without tamper proof hardware. So it is essential to update the session key K_{si}. In this stage the session key of a node is updated and new network key is created. The session key is updated periodically. To update the keys of nodes BS broadcast a random number R_t at time t before the capturing time T_{cap}. All nodes update their session keys using one way hash function and discard the old keys as (Fig. 1):

$K_{si\text{-}t1} = H(K_{si}, R_{t1})$ at time t_1
$K_{si\text{-}t2} = H(K_{si\text{-}t1}, R_{t2})$ at time t_2
……..……..

When CH receives broadcast message of random number from BS it updates its all members keys and its own keys as well. If node *i* need to sends data to its CH between time t_1 and t_2, it encrypts its message using new key $K_{si\text{-}t1}$. Since CH updates its member keys it can easily decrypt the message.

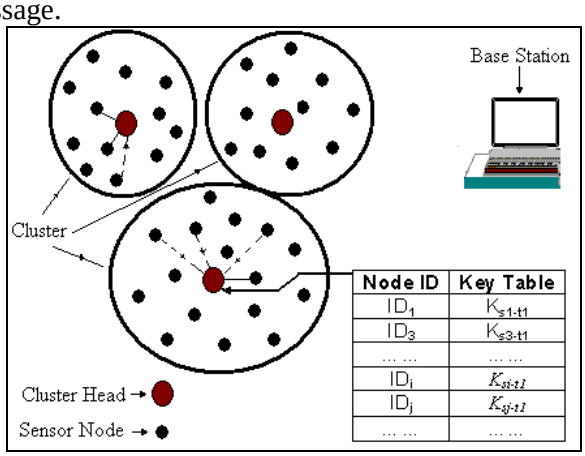

Figure 1: Updated Key Table of Cluster Head ID_j at Time t_1

Now all nodes including CH create a new network key K_N using the random number R_t and another hash function $H'( )$ as follows:

$K_{N,1} = H'(R_{t,1})$ at time t_1
$K_{N,2} = H'(K_{N,1}, R_{t,2})$ at time t_2
……
$K_{N\text{-}new} = H'(K_{N,ch\text{-}1}, R_{t,\,ch\text{-}1})$ at time T_{ch}

where T_{ch} is present cluster duration. After time T_{ch} new cluster is formed and the network key $K_{N\text{-}new}$ is used to create new cluster member table.

6.4.1. Missing Key Update Message: If a sensor node does not receive the broadcasted random number packets due unreliable wireless channels such as radio shading, fading and communication noise, it can not encrypt any message using updated keys. This node seems as a malicious node or compromised node to CH. The proposed key management technique manages this situation as follows:

Every node in the network opens a counter to count time. If a node does not receive any broadcast packets from BS within time T_{cap}, it immediately sends a message to its CH using the last update keys.

Consider that at time t_n node *i* does not receive the random number $R_{t,n}$. Now node$_i$ sends a request message to its CH using the last update key $K_{si\text{-}n\text{-}1}$ and last received broadcast number $R_{t,n\text{-}1}$ as

ID_i	$E_{Ksi\text{-}n\text{-}1}(M)$	$R_{t,n\text{-}1}$	$MAC_{Ksi}(M)$

where M is a request message to CH for new update key

CH cannot decrypt the message since it already updates the key $K_{si\text{-}n\text{-}1}$ to $K_{si\text{-}n}$. CH immediately sends this node ID to the BS for authentication [17] [21]. If BS authenticates this node it sends a message to CH to replace the old key $K_{si\text{-}n\text{-}1}$ with new update key $K_{si\text{-}n}$. CH sends the last random broadcast number $R_{t,n}$ using the old key $K_{si\text{-}n\text{-}1}$ to update the network key $K_{N\text{-}n}$.

6.5. Re-clustering of Sensor Network

To balance the energy consumption between CH and other sensor nodes, after certain time interval old cluster breaks and new cluster rebuild [8]. We consider this situation in this hierarchical model of key management technique. At the end of time of cluster duration T_{ch}, BS broadcast a special packet to all CH to erase its member table. When a new cluster formation goes on, new cluster leader make a table of its member node as in describes is section 6.2 and continue its operation.

6.6. Fixed Clustering of Sensor Network

The idea of periodic re-clustering brings extra overhead, e.g. new cluster heads selections, advertisements etc., which may diminish the gain in energy consumption. Therefore in some application of sensor network this technique avoided. We also consider this situation when a cluster head remain fixed. In this condition only the session key is updated periodically as in section 6.4. In this case BS does not send any message of re-clustering. If a CH energy level below a critical value, it sends a message to BS. BS initiates re-clustering as discussed in above section 6.5.

6.7. Sensor Death

When a node's available power drops below a certain level, node sends a Node Death message to its CH. CH removes this node from its cluster member table and broadcasts a notification to its cluster members and BS. This message instructs all nodes in the cluster to remove that node from their neighbor tables.

7. Security Analysis

Security of this scheme depends on one-way hash function, length of the key and time duration of broadcasting the random number.

In this proposed method a node do not sends any key directly to other node so it is not possible for an eavesdropper to get any information about keys. On the eve of cluster formation, the first message is encrypted using cluster key and this key is updated in regular interval using one way function, it is not possible for an eavesdropper to extract key from a transmitted message.

Another possible attack is node capturing. BS periodically broadcast random number R_t before time T_{cap}. It is assume that adversary cannot update the keys of capturing node. Captured node sends data using last updating session key. But in the mean time CH and BS already updates the keys. So CH cannot decrypt the message from a node or BS failed to decrypt the message from a compromising CH. BS come to a decision that this CH is compromised and reject all data that have received from that node.

When the clustering method rotates among the node, if a CH compromise, it cannot send false data to BS long time since after each time T_{ch} a new cluster rebuild. Adversary needs to capture new CH again. As network key K_N is rebuild periodically by hash function, if an adversary capture a dead node, it cannot participate in new cluster formation process.

The limitation of this approach is that if a node failed to update its key due to communication problem it is seems as a malicious node to the network. In this key management technique we depend on other node authentication technique for sensor network. If the authentication technique is not strong enough an adversary may success to compromise a node and introduce the node as legitimate node.

Another drawback is that a node continually receives packet of random number from BS. Though the receiving energy cost of a packet much lower than transmitting a packet, frequent receiving increases the overhead of energy consumption.

8. Simulation and Results

In key management schemes, the energy required for computation is three orders of magnitude less than that required for communication [25]. Moreover, amount of energy consumed for computation varies significantly with hardware. Hence, we only consider number of packet overhead in reception for member nodes and CH, and do not consider energy costs associated with computation. We also consider that BS has powerful antenna, so any node do not relay the broadcasted random number to other node.

8.1. Communication Overhead

To compute the receiving packet overhead we simulate the situation. For simulation we considered the Mica2 [13] node parameters. Table 2 lists the important parameters used for this simulation.

Table – 2: Parameters Used in Key Management Protocols

Simulation Area	100m × 100m
No. of Node	125
Capturing Time T_{cap}	5 minutes
Simulation time	1 hour
Data sending Rate / Node	1 packet / 30 sec.

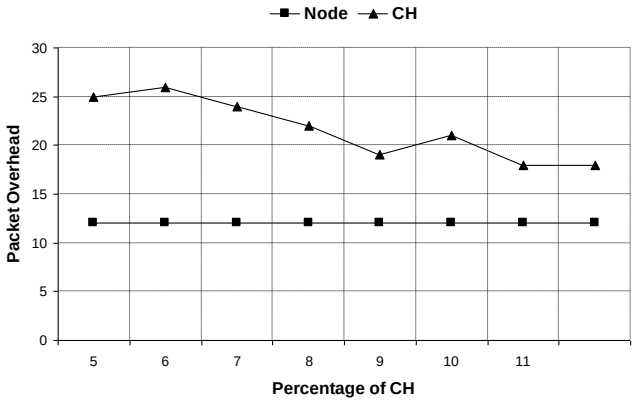

Figure 2: Packet receiving Overhead of a node when act as node and CH for an hour with 10% probability that a node do not receive random number Rt from BS

Figure 2 shows the packet over head for receiving the random number of a node when it acts as CH and as a normal node. It assumes that there is 10% probability that a node do not receive random number from BS. The figure shows that number of receiving packet overhead is constant for member nodes as they constantly receive BS broadcast. But no. of received packets of a CH increases if the no. of CH of the network decreases. Since no. of receiving packets of a CH depends upon its member nodes. If the no. members are high CH receive

more packets from its member nodes. Figure – 3 represents the receiving overhead of a CH of the proposed key management scheme. It is approximately 2% more when comparing with plain nodes.

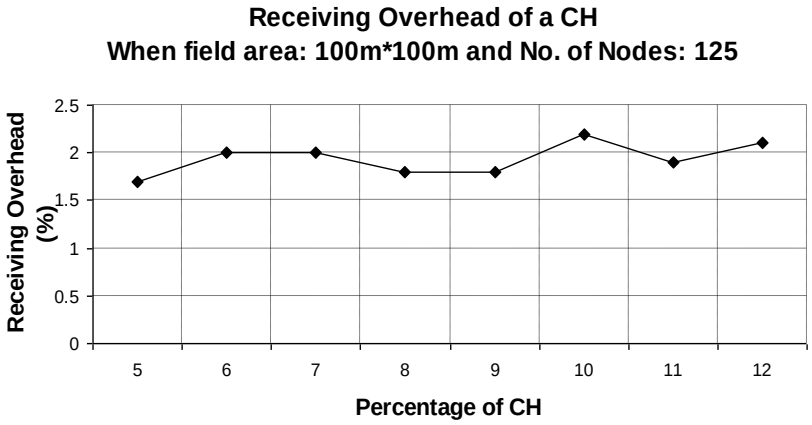

Figure 3: Receiving Overhead per CH

8.2. Storage Overhead

To measure the memory overhead we make the same assumptions for node ID that is used in TinySec [19] – 16 bits long. We also consider the symmetric keys are 128 bits long. Memory overhead for a node is 256 bits (32 Bytes). A CH need to store its member nodes ID and session keys. Figure 4 shows the average storage overhead for cluster leader. Memory overhead of a CH depends on number of member nodes. If there is few percentage of CH, memory overhead increases.

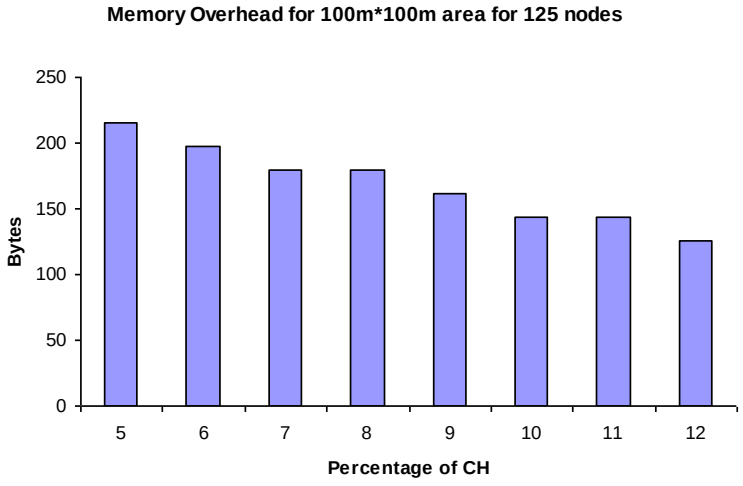

Figure 4: The Storage Overhead for CH

9. Conclusion and Future Work

In this paper, we proposed a key management technique to distribute the keys within a cluster and update the keys at regular interval to avoid node-capturing problem. The method uses pre-deployed symmetric keying. A critical observation is that sensor-to-sensor secure channel establishment is not necessary for many monitoring applications. Therefore, pre-deployed keying has become sufficient, cost-effective approach to provide a keying infrastructure for security protocols that use those keys.

The key distribution is completely local. Once keys are distributed the network key is discarded. We have proposed a key updating technique to prevent the node capturing. Network key is reconstructed when keys are updated. This scheme authenticates group of cluster nodes instead of every node in a network. Therefore it shows better scalability. Propose key management technique has little communication overhead due to receiving key update packets from base station and small memory overhead.

The proposed intrusion detection system and key management model can prevent the common network threats of hierarchical sensor network and minimize the node capturing attacks. It can ensure the secure communication between communicating nodes. This security solution is suitable when nodes are deployed by human or any instrument that act as deployment agent. If the nodes are dropped from plane or missile, it may be ineffective.

References

[1] I. F. Akyildiz, W. Su, Y. Sankarasubramaniam, and E. Cayirci, “A survey on sensor networks”, IEEE Communications Magazine, 40(8):102–114, August 2002.

[2] A. Menezes, P. van Oorschot, and S. Vanstone, “Handbook of Applied Cryptography”, CRC Press, 1996.

[3] W. Du, J. Deng, Y. S. Han, S. Chen, and P. K. Varshney, “A Key Management Scheme for Wireless Sensor Networks Using Deployment Knowledge”, Twenty-third AnnualJoint Conference of the IEEE Computer and Communications Societies, INFOCOM 2004.

[4] Neuman, B. C. and Tso, T. “Kerberos: an authentication service for computer networks”, IEEE Communications Magazine 32(9), pp. 33-38, 1994.

[5] L. Eschenauer and V. D. Gligor, “A key-management scheme for distributed sensor networks,” in Proceedings of the 9th ACM conference on Computer and communications security, pp. 41–47, Washington, DC, USA: ACM Press, 2002.

[6] Bruce Schneier, Applied Cryptography, John Wiley & Sons, Inc., Second Edition, 1996.

[7] Gunter, C.G., “An Identity-Based Key-Exchange Protocol”. In: Quisquater, J-J., Vandewalle, J. (eds.) Eurocrypt 1989. LNCS, vol. 435, pp. 29–37. Springer, Heidelberg (1990).

[8] W.R. Heinzelman, A. Chandrakasan, H. Balakrishnan, “Energy efficient communication protocol for wireless microsensor networks”, in: Proceedings of the 33rd Annual Hawaii International Conference on System Sciences (HICSS), January 2000, pp. 3005–3014.

[9] K. Akkaya and M. Younis, "A Survey of Routing Protocols in Wireless Sensor Networks, " in the Elsevier Ad Hoc Network Journal, Vol. 3/3 pp. 325-349, 2005.

[10] C. Karlof, D. Wagner, “Secure Routing in Sensor Networks: Attacks and Countermeasures,” in Ad Hoc Networks, volume 1, issues 2--3 (Special Issue on Sensor Network Applications and Protocols), Elsevier, September 2003, pp. 293-315.

[11] A. D.Wood and J. A. Stankovic, “Denial of service in sensor networks,” Computer, vol. 35, no. 10, pp. 54–62, 2002.

[12] R. Watro, D. Kong, S. Cuti, C. Gardiner, C. Lynn, and P. Kruus. Tinypk: securing sensor networks with public key technology. In Proceedings of the 2nd ACM workshop on Security of Ad hoc and Sensor Networks (SASN ’04), pages 59–64, New York, NY, USA, 2004. ACM Press.

[13] MICA2: Crossbow Technologies Inc. 100 3317.73
http://www.xbow.com/Products/Product_pdf_files/Wireless_pdf/MICA2_Datasheet.pdf

[14] H. Chan, A. Perrig, and D. Song, "Random key predistribution schemes for sensor networks," in Security and Privacy, 2003. Proceedings. 2003 Symposium on, pp. 197–213, 2003.

[15] W. Du, J. Deng, Y. Han, S. Chen, and P. Varshney, "A key management scheme for wireless sensor networks using deployment knowledge," in INFOCOM 2004. Twenty-third Annual Joint Conference of the IEEE Computer and Communications Societies, vol. 1, p. 597, 2004.

[16] W. Du, J. Deng, Y. S. Han, and P. K. Varshney, "A Pairwise Key Pre-Distribution Scheme for Wireless Sensor Networks", ACM CCS 2003.

[17] A. Perrig, R. Szewczyk, J. D. Tygar, V. Wen, and D. E. Culler. SPINS: security protocols for sensor networks. Wireless Networking, 8(5):521–534, 2002.

[18] Zia, T.A., and Zomaya, A.Y., "Security Issues in Wireless Sensor Networks",In the proceedings of the International Conference on Systems and Networks (ICSNC'06), Nov 2-4, 2006, Tahiti, French Polynesia.

[19] Chris Karlof, Naveen Sastry, David Wagner, "TinySec: A Link Layer Security Architecture for Wireless Sensor Networks," ACM SenSys 2004, November 3-5, 2004.

[20] Yoon-Su Jeong, Bong-Keun Lee, Sang-Ho Lee, "An Efficient Key Management Scheme for Secure Sensor Networks," in Sixth IEEE International Conference on Computer and Information Technology, CIT '06, 2006.

[21] Fei Hu, Waqaas Siddiqui, Krishna Sankar, "Scalable security in Wireless Sensor and Actuator Networks (WSANs): Integration re-keying with routing", Computer Networks 51 (2007) 285–308, Science Direct, Elsevier.

[22] Donggang Liu, Peng Ning, "Efficient distribution of key chain commitments for broadcast authentication in distributed sensor networks" in: Proceedings of the 10th Annual Network and Distributed System Security Symposium, San Diego, California, February 2003, pp. 263–276.

[23] Hung-Yu Chein, Jinn-Ke Jan, "New hierarchical assignment without public key cryptography", Computer & Security, Vol 22, No 6, pp 523-526, Elsevier, 2003.

[24] Bart Preneel, "The State of Cryptographic Hash Functions", Lectures on Data Security, pp. 158-182, Springer, 1999.

[25] D. W. Carman, P. S. Krus, and B. J. Matt. "Constraints and approaches for distributed sensor network security", Technical Report 00-010, NAI Labs, Network Associates, Inc., Glenwood, MD, 2000.