A Key Distribution Scheme Preventing Collusion Attacks in Ubiquitous Heterogeneous Sensor Networks

Firdous Kausar[1], Sajid Hussain[2], Jong Hyuk Park[3], and Ashraf Masood[1]

[1] College of Signals, NUST, Rawalpindi, Pakistan.
firdous.imam@gmail.com, ashrafm61@gmail.com
[2] Jodrey School of Computer Science, Acadia University, Nova Scotia, Canada.
sajid.hussain@acadiau.ca,
[3] Department of Computer Engineering, Kyungnam University, Masan, Korea.
parkjonghyuk@gmail.com

Abstract. Random key pre-distribution schemes are vulnerable to collusion attacks. In this paper, we propose a new key management scheme for ubiquitous heterogeneous sensor networks consisting of a small number of powerful high-end $\mathcal{H}$-sensors and a large number of ordinary low-end $\mathcal{L}$-sensors. The collusion attack on key pre-distribution scheme mainly takes advantage of the globally applicable keys, which are selected from the same key pool. As a result, in our scheme, after discovering the shared pairwise keys with neighbors, all $\mathcal{H}$-nodes and $\mathcal{L}$-nodes destroy their initial key rings and generate new key rings by applying one-way hash function on node's ID and initial key ring. The analysis of proposed scheme shows that even if a large number of nodes are compromised, an adversary can only exploit a small number of keys nearby the compromised nodes, while other keys in the network remain safe. It outperforms the previous random key pre-distribution schemes by considerably reducing the storage requirement, while providing more resiliency against node capture and collusion attacks.

1 Introduction

Wireless sensor networks (WSNs) have attracted wide attention due to their ubiquitous surveillant application. WSNs are formed by a large number of sensor nodes. Each sensor node contains a battery-powered embedded processor and a radio, which enables the nodes to self-organize into a network, communicate with each other and exchange data over wireless links. WSNs are commonly used in ubiquitous and pervasive applications such as military, homeland security, health-care, and industry automation.[1].

An important area of research interest is a general architecture for wide-area sensor networks that seamlessly integrates homogeneous and heterogeneous sensor networks. Heterogeneous sensor networks have different types sensors, with a large number of ordinary sensors in addition to a few powerful sensors. Further, as sensor devices are typically vulnerable to physical compromise and

they have very limited power and processing resources, it is unacceptable to completely trust the results reported from sensor networks, which are deployed outside of controlled environments without proper security.

In order to provide secret communication in a sensor network, shared secret keys are used between communicating nodes to encrypt data. Key establishment protocols are used to set up the shared secrets, but the problem is complicated by the sensor nodes' limited computational capabilities, battery energy, and available memory. Hence, asymmetric cryptography such as RSA or Elliptic Curve cryptography (ECC) is unsuitable for most sensor architectures due to high energy consumption and increased code storage requirements. Several alternative approaches have been developed to perform key management on resource-constrained sensor networks without involving the use of asymmetric cryptography such as single network-wide key, pairwise key establishment, trusted base station, and random key pre-distribution schemes [2].

In random key pre-distribution (RKP) schemes, a large key pool of random symmetric keys is generated along with the key identifiers. All nodes are given a fixed number of keys randomly selected from a key pool. In order to determine whether or not a key is shared, each node broadcasts its keys' identifiers. The, neighbors sharing a key associated with one of those identifiers, issue a challenge/response to the source. If two nodes do not share keys directly, they can establish a session key with the help of neighbors with which a key is already shared. It is highly likely that all nodes in the network will share at least one key if the following are carefully considered: a) the network density, b) the size of the key pool, and c) the number of keys pre-configured in each sensor node.

While pre-distributing pairwise keys does protect confidentiality, it still loads nodes with a large number of globally-applicable secrets. By eliminating the eavesdropping attack, the pairwise scheme makes another type of malicious behavior more attractive. As several nodes possess the same keys, any node can make use of them by simply combining the keys obtained from a few nodes, which greatly increases the attacker's chances of sharing keys with other nodes. A collusive attacker can share its pairwise keys between compromised nodes by enabling each node to present multiple 'authenticated' identities to neighboring nodes while escaping detection [3].

An adversary who obtains compromised nodes' keys can inject malicious sensor nodes elsewhere in the network since the pool keys that were obtained are always valid and are used to authenticate each node. As a result, RKP is unable to protect the sensor network against collusion attack. In order to counter the collusion attacks, nodes should discard unused keys from the node's memory after the initialization phase; however, it means that new nodes can no longer join the system after the initial network deployment.

In this paper, we propose a key management scheme based on random key pre-distribution for heterogeneous sensor networks. The proposed scheme is resilient against collusion attack. The rest of the paper is organized as follows. Section 2 provides the related work and Section 3 describes the network and threat model. In Section 4, the proposed scheme is described in detail. Section 5

gives the results and performance evaluation. Finally, Section 6 concludes the paper.

2 Related Work

Key management for WSNs is a critical issue that has been addressed through many proposed schemes presented in various papers. Eschenauer and Gligor [4] propose a distributed key establishment mechanism that relies on probabilistic key sharing among the nodes of a random graph and uses a shared-key discovery protocol for key establishment. Chan et al. [5] further extended this idea and propose the q-composite key predistribution. This approach allows two sensors to setup a pairwise key only when they share at least q common keys. Chan et al. also developed a random pairwise keys scheme to defeat node capture attacks. Oliveira et al. [6] show how random key predistribution, widely studied in the context of flat networks, can be used to secure communication in hierarchical (cluster-based) protocols such as LEACH [7]. They presented SecLEACH, a protocol for securing node-to-node communication in LEACH-based networks. These and some others [8], [9], [10], [11], [12] efforts have assumed a deployment of homogeneous nodes, and have therefore suggested a balanced distribution of random keys to each of the nodes to achieve security. Most of these schemes suffer from high communication and computation overhead, and/or high storage requirement.

Perrig et al. [13] propose SPINS, a security architecture specifically designed for sensor networks. In SPINS, each sensor node shares a secret key with the base station. Two sensor nodes cannot directly establish a secret key. However, they can use the base station as a trusted third party to set up the secret key.

Blundo et al. [14] propose several schemes that allow any group of t parties to compute a common key, while being secure against collusion between some of them. These schemes focus on saving communication costs, while memory constraints are not placed on group members. When $t = 2$, one of these schemes is actually a special case of Blom's scheme [15].

Availability of some information on the sensors deployment in the field assists to improve the security of the key pre-distribution schemes. Some location-aware schemes are proposed in [16] and [17]. These techniques divide the target field into non-overlapping square areas and randomly deploy the sensors in every area. The exact location of a sensor in any area is unknown, but there is knowledge about the identity of sensors in every area. This information helps to eliminate the dependency of keys between nonadjacent cells.

Du et al.[18] propose the asymmetric pre-distribution (AP) scheme for heterogeneous sensor networks. They consider a small number of powerful high-end sensors and a large number of ordinary low-end sensors. The basic idea of the AP key management scheme is to pre-load a large number of keys in each H-sensor whereas only a small number of keys are pre-loaded in each L-sensor, in order to provide better security with low complexity and significant reduction in storage requirement. Traynor et al. [19] demonstrate that a probabilistic unbalanced dis-

tribution of keys throughout the network that leverages the existence of a small percentage of more capable sensor nodes can not only provide an equal level of security but also reduces the consequences of node compromise. Lu et al. [20] propose a framework for key management schemes in distributed wireless sensor networks with heterogeneous sensor nodes.

3 Network Model

We consider a heterogeneous sensor network (HSN) consisting of a small number of high end ($\mathcal{H}$-node) and a large number low end ($\mathcal{L}$-node) sensors. $\mathcal{L}$-nodes are ordinary sensor nodes with limited computation, communication, and storage capability. $\mathcal{H}$-nodes, however, are more powerful nodes and have higher computation, communication, energy supply and storage capability than $\mathcal{L}$-nodes. Further, the HSN includes a base station (BS) that is globally trusted and it receives data from all the nodes; the BS has unlimited resources.

We consider the hierarchical structure of the HSN in which $\mathcal{H}$-nodes act as cluster heads (CHs). Clustering of sensors enable local data processing, which reduces communication load in the network in order to provide scalable solutions.

We assume that sensor nodes are not mobile. Though they are deployed randomly; once placed at a particular location, they do not tend to move from that location. But it is dynamic in nature as new sensor nodes may be added after network formation or some of the nodes may die down due to energy exhaustion or malfunction. This causes change in neighbor information and overall network topology.

3.1 Threat Model

Sensor networks are often deployed in hostile environments, yet nodes cannot afford expensive tamper-resistant hardware. The threat model is assumed to be an adversary that tries to capture and compromise a number of nodes in the network. Also, there is no unconditional trust on any sensor node. If an adversary compromises a node, the memory of that node is known to the adversary; CHs can also be compromised. The goal of the adversary is to uncover the keys used in the network for secure communication. The nodes can collude with each other by sharing their keys with other attacker nodes.

3.2 Preliminaries

Definition 1 *A pseudo-random function is an efficient (deterministic) algorithm which given an h-bit seed,y, and an h-bit argument, x, returns an h-bit string, denoted $f_y(x)$, so that it is infeasible to distinguish the responses of f_y , for a uniformly chosen y, from the responses of a truly random function.*

Definition 2 *A cryptographically secure one-way hash function H has the following property: for $y = H(x, k)$, 1) given x, it is computationally infeasible to find y without knowing the value of k; 2) given y and k, it is computationally infeasible to find x.*

Notation	Definition
BS	Base Station
CH	Cluster Head
id_{L_i}	Identity of $\mathcal{L}$-node i
id_{H_i}	Identity of $\mathcal{H}$-node i
N	A random number string
R_{L_i}	Set of the keys in $\mathcal{L}$-node i initial key ring
R_{H_i}	Set of the keys in $\mathcal{H}$-node i initial key ring
R'_{L_i}	Set of the keys in $\mathcal{L}$-node i new/update key ring
R'_{H_i}	Set of the keys in $\mathcal{H}$-node i new/update key ring
$K_{X,Y}$	A shared key between X and Y
$\{m\}_K$	An encryption of message m with key K
$MAC_K(msg)$	MAC calculated using key K
$\|$	concatenation symbol

Table 1. Symbol Definition

Definition 3 *(Key Graph) Let V represent all the nodes in the sensor network. A Key-Sharing graph G(V,E) is constructed in the following manner: For any two nodes i and j in V , there exists an edge between them if and only if nodes i and j have at least one common key in their key ring. Note that $|V| = n$ for a WSN of size n, the key graph G(V;E) is connected if and only if any two nodes i and j belonging to V can reach each other via edge set E only.*

For convenience, a summary of notations and symbols used in the paper are given in Table 1.

4 Protocol

In this section we describe our key management scheme in detail.

4.1 Initial Deployment

Generate a large key pool P consisting of a S number of random symmetric keys and their ids prior to network deployment. Before deploying the nodes, each node is loaded with its assigned key ring R as follows: each $\mathcal{L}$-node is pre-loaded with γ number of keys and each $\mathcal{H}$-node is pre-loaded with ρ number of keys, randomly selected from the key pool without replacement, where $\rho >> \gamma$. As given in [21], the assigning rules are as follows:

$\mathcal{L}$-node:

for every key $k_i \in P$, where $P = (k_1, k_2, ..., k_S)$
compute $z = f_{k_i}(id_{L_x})$
if $z \equiv 0 \ mod \ (\frac{S}{\gamma})$ then
put k_i into R_{L_x}, the key ring of $\mathcal{L}$-node.

$\mathcal{H}$-node:

for every key $k_i \in P$, where $P = (k_1, k_2, ..., k_S)$
compute $z = f_{k_i}(id_{H_x})$
if $z \equiv\ 0\ mod\ (\frac{S}{\rho})$ then
put k_i into R_{H_x}, the key ring of $\mathcal{H}$-node.

4.2 Cluster Organization Phase

After the initial deployment, nodes enter into the cluster organization phase. Let $\mathcal{H}$-node H_a broadcasts an advertisement message adv, consisting of its id (id_{H_a}) and N as shown in message 1 of Figure 1. The nearby $\mathcal{L}$-node L_b upon receiving the adv message, L_b determines whether it shares a common key with H_a as follows: for every key $k_j \in R_{L_b}$, L_b computes $z = f_{k_j}(id_{H_a})$. If $z \equiv\ 0\ mod(\frac{S}{\rho})$, it means that H_a also has a key k_j in its key ring i.e. $R_{H_a} \cap R_{L_b} = k_j$.

As L_b could receive adv broadcast messages from several $\mathcal{H}$-nodes, it would be possible that L_b shares a common key with more than one $\mathcal{H}$-node. From these $\mathcal{H}$-nodes, it will choose the $\mathcal{H}$-node as its CH with whom it has the best received signal strength and link quality.

L_b sends the join request to the selected CH (say H_a) protected by MAC, using k_j and include the N from CH broadcast (to prevent replay attack), as well as the id of shared key (id_{k_j}) chosen to protect this link (so that the receiving CH knows which key to use to verify the MAC) as shown in message 2 of Figure 1. Both H_a and L_b will generate the shared pairwise key by applying one-way hash function on id_{L_b} and id_{H_a} by using k_j as shown in message 3 of Figure 1.

1: $H_a \rightarrow * : id_{H_a}, N$
2: $L_b \rightarrow H_a : id_{L_b}, id_{H_a}, id_{k_j}, MAC_{k_j}(id_{L_b} \| id_{H_a} \| id_{k_j} \| N)$
3: $K_{H_a,L_b} = H(k_j, id_{H_a} || id_{L_b})$.

Fig. 1. Messages Transferred between sensor nodes and CHs.

Direct key discovery phase After cluster organization phase, $\mathcal{L}$-nodes learn their neighbors through the exchange of *hello* messages, and then attempt to establish keys with their neighbors. To accomplish this, $\mathcal{L}$-nodes broadcast *hello* messages.

Consider an $\mathcal{L}$-node, L_a, it broadcast a *hello* message consisting of its id id_{L_a}. Then, it waits for hello messages from its neighboring $\mathcal{L}$-nodes. Suppose, it receive hello message from one of its neighbor L_b, it extracts the node id from message i.e. id_{L_b}. For every key $k_j \in R_{L_a}$, L_a computes $z = f_{k_j}(id_{L_b})$. If $z \equiv\ 0\ mod(\frac{S}{\gamma})$, it means that node L_b also has a key k_j in its key ring i.e. $R_{L_a} \cap R_{L_b} = k_j$. After discovering the common key in their key rings, they will generate the shared pairwise key by applying one-way hash function on id_{L_a} and id_{L_b} by using k_j.

$$K_{L_a,L_b} = H(k_j, id_{L_a}||id_{L_b})$$

.

If L_a and L_b share more than one common keys in their key rings, the key with the least id would be used to generate the shared pairwise key.

Indirect key discovery phase $\mathcal{L}$-nodes gather information about both types of neighbors: 1) nodes with which they share a key, and 2) nodes with which they do not share keys. When the direct key discovery phase ends, the nodes would have discovered the common keys, if any, with their neighbors. $\mathcal{L}$-nodes use the CH with which keys are already shared to assist it in establishing secure connections with the neighboring $\mathcal{L}$-nodes with which common keys are not found.

Let $\mathcal{L}$-nodes L_i and L_j are neighboring nodes in the same cluster; however, they do not share a common key in their key rings, $R_{L_i} \cap R_{L_j} = \phi$. The $\mathcal{L}$-node L_i, having already established a link with the its CH (H_a), transmits a message to H_a, as shown in Figure 2, requesting to transmit a key with $\mathcal{L}$-node L_j encrypted with key K_{H_a,L_i}.

The $\mathcal{H}$-node generates a key k_x and unicasts the message 2 to L_i and message 3 to L_j shown in Figure 2. When L_i (or L_j) receives its message from H_a, it decrypts the message using key K_{H_a,L_i} to get key k_x. Similarly, L_j uses key K_{H_a,L_j} for decrypting the message. Now, L_i and L_j generate the shared pairwise by applying one-way hash function on id_{L_i} and id_{L_j} by using k_x, as shown in message 4.

1: $L_i \rightarrow H_a : id_{L_i}, id_{L_j}, N, MAC_{K_{H_a,L_i}}(id_{L_i} \| id_{L_j} \| N)$
2: $H_a \rightarrow L_i : id_{L_i}, id_{L_j}, N, \{k_x\}_{K_{H_a,L_i}}$
3: $H_a \rightarrow L_j : id_{L_i}, id_{L_j}, N, \{k_x\}_{K_{H_a,L_j}}$
4: $K_{L_i,L_j} = H(k_x, id_{L_i}||id_{L_j})$.

Fig. 2. Messages Transferred between sensor nodes and CHs.

4.3 Key Ring Update

After indirect key-discovery phase, all $\mathcal{L}$-nodes and $\mathcal{H}$-nodes destroy their initial key rings. Because these key rings have globally applicable secrets which can be used by adversary to launch a collusion attack, we delete these initial key rings.

First, before a node (say L_x) destroys its initial key ring, it generates a new key ring as shown in Figure 3. For every key $k_i \in R_{L_x}$, it generates a new key $\acute{k_i}$ by applying one-way hash function on its id (id_{L_x}) and k_i. In this way, it generates a set of new keys from keys in its initial key ring. Further, in order to keep record of the keys in its initial key ring, these newly generated keys are

assigned the same ids as that were of the original keys. Then, L_x deletes k_i from its key ring R_{L_x}.

procedure keyRingUpdate()
1: **for** $\forall k_i \in R_{L_x}$ **do**
2: $\quad k_i^{'} = H(k_i, id_{L_x})$
3: $\quad id_{k_i^{'}} = id_{k_i}$
4: $\quad$ delete(k_i)
5: **end for**

Fig. 3. Key Ring Update

Further, the above procedure is also applied for $\mathcal{H}$-nodes to update their key rings.

4.4 Addition of a new node

The common key pre-distribution schemes are unable to add new nodes in the network if the initial key rings are deleted from node's memory. As a result, we develop a new solution capable of handling addition of new legitimate $\mathcal{L}$-nodes beyond the initial deployment, even after the deletion of initial key rings from node's memory.

Suppose new $\mathcal{L}$-node L_x wants to join a network, it broadcasts a join request consisting of its id (id_{L_x}) and a random number N, as shown in message 1 of Figure 4. Then, it waits for reply from nearby CHs. Let L_x receives a reply message from CH (say H_a). For every key $k_j \in R_{L_x}$, L_x computes $z = f_{k_j}(id_{H_a})$. If for any k_j, $z \equiv 0\ mod(\frac{S}{\rho})$, it means that $k_j \in R_{H_a}$, but it is no longer available now because R_{H_a} has been deleted. So, L_x computes the corresponding key i.e. $k_j^{'}$ of H_a's new key ring $R_{H_a}^{'}$ by applying one-way hash function on id_{H_a} and k_j i.e. $k_j^{'} = H(k_j, id_{H_a})$. Then, L_x sends a message to H_a consisting of its id id_{L_x}, id of k_j ($id_{k_j} = id_{k_j^{'}}$), random number N and MAC is calculated on all these values using $k_j^{'}$ as shown in message 3 of Figure 4. Now, L_x and H_a generate the shared pairwise key by applying one-way hash function on id_{H_a} and id_{L_x} by using $k_j^{'}$, as shown in message 4.

1: $L_x \rightarrow * : id_{L_x}, N$
2: $H_a \rightarrow L_x : id_{H_a}, N$
3: $L_x \rightarrow H_a : id_{L_x}, MAC_{k_j^{'}}(id_{L_x} \| id_{k_j} \| N)$
4: $K_{L_x,H_a} = H(k_j^{'}, id_{L_x} || id_{H_a})$.

Fig. 4. New node addition

Then, L_x discovers the shared key with its neighboring $\mathcal{L}$-nodes by using either direct or indirect key discovery phase, as given above.

5 Analysis

This section analyzes the proposed scheme and explains its features that make this scheme feasible to implement and a better alternative option as compared to the other key pre-distribution schemes.

For any pair of nodes to find a secret key between them, the key sharing graph $G(V, E)$ needs to be connected. Given the size and the density of a network, the objective is to determine the key pool size S, the number of keys assigned to $\mathcal{L}$-nodes γ, and the number of keys assigned to $\mathcal{H}$-nodes ρ such that, the graph G is connected with high probability.

For an $\mathcal{L}$-node, the total possible number of key ring assignments are:

$$\frac{S!}{\gamma!(S-\gamma)!}$$

For an $\mathcal{H}$-node, the number of possible key ring assignments are:

$$\frac{S!}{\rho!(S-\rho)!}$$

The total number of possible key ring assignments for an $\mathcal{L}$-node and an $\mathcal{H}$-node are:

$$\frac{S!}{\gamma!(S-\gamma)!} \times \frac{S!}{\rho!(S-\rho)!}$$

The probability of an $\mathcal{L}$-node and $\mathcal{H}$-node with key rings sizes γ and ρ sharing at least one key with each other is given in Equation 1:

$$p = 1 - \frac{(S-\gamma)!(S-\rho)!}{S!(S-\gamma-\rho)!} \tag{1}$$

Figure 5 shows the probability of key sharing among $\mathcal{H}$-node and $\mathcal{L}$-node with respect to key pool size. Further, a fixed number of pre-loaded keys are used in $\mathcal{H}$-nodes, $\rho = 500$; whereas pre-loaded keys for $\mathcal{L}$-nodes vary as $\gamma = 10, 20, 30$. The graphs show that the pre-loaded keys in $\mathcal{L}$-nodes can be significantly reduced with acceptable probability of key sharing.

5.1 Security Analysis

We evaluate our key pre-distribution scheme in terms of its resilience against node capture and collusion attack. We would like to investigate when α number of captures nodes are captured, what fraction of the additional communication (i.e. communication among uncaptured nodes) would be compromised?

To compute this fraction, we first compute the probability that any one of the additional communication links is compromised after α nodes are captured.

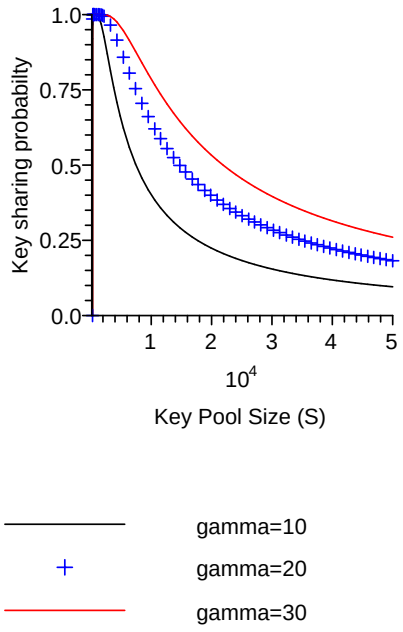

Fig. 5. The Key Sharing Probability

In our analysis, we are considering the links which are secured using a pairwise key computed from the common key shared by the two nodes of this link. We should also notice that during shared key discovery process, two neighboring nodes find the common key in their key rings and use this key to agree upon another random key to secure their communication. Because this new key is generated by applying one-way hash function on common shared key and node ids, the security of this new random key does not directly depend on whether the key rings are broken. Further, the nodes' initial key rings are also deleted from their memory, after setting up shared pairwise keys with neighbors. As a result, the fraction of communications compromised when α number of nodes being compromised can be given as

$$\frac{number\ of\ links\ in\ \alpha\ compromised\ nodes}{Total\ number\ of\ links}$$

which means that only those links will be affected which are directly connected with α compromised nodes, while the other links in the network will remain safe. Figure 6 shows the graphs of number of compromised communication links with respect to the number of compromised nodes. We compare our proposed scheme (PS) with basic scheme (EG) [4] and q-composite scheme [5]. The graphs show

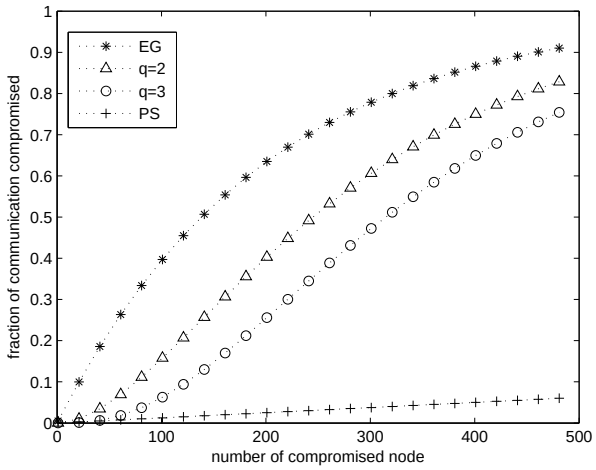

Fig. 6. The Compromising Probability

that as the number of compromised nodes increases, the traditional schemes are severely affected as compared to PS.

Further, in collusion attacks, the adversary takes advantage of the pairwise secret keys stored by each sensor node as these keys are globally applicable secrets and can be used throughout the network, yet ordinary sensors can only communicate with the small fraction of nodes within radio range. So, the adversary can launch a collusion attack by exploiting this lack of communication between nodes and can now share its pairwise keys between compromised nodes, enabling each node to present multiple 'authenticated' identities to neighboring nodes, while escaping detection. In proposed scheme, we delete the initial key rings from nodes memory after setting up shared pairwise keys with neighbors. However, nodes generate new key rings from initial key rings by applying one-way hash function on node ids and keys in their initial key rings.

Consider two arbitrary $\mathcal{L}$-nodes, L_a and L_b, where $R_{L_a} = \{k_1, k_2, \ldots, k_\gamma\}$, $R_{L_b} = \{k_1, k_2, \ldots, k_\gamma\}$, and $R_{L_a} \cap R_{L_b} = k_i$. As L_a and L_b are not within the communication range of each other, they do not use k_i. After setting up shared pairwise keys with neighbors, both L_a and L_b delete the initial key rings (R_{L_a} and R_{L_b}) and generate the new key rings (say $R^{'}_{L_a}$ and $R^{'}_{L_b}$) by applying one-way hash function on all the keys in their initial key rings and node ids. As a result, $R^{'}_{L_a} \cap R^{'}_{L_b} = \phi$. Similarly, in α number of compromised nodes, there will be no common key in their new key rings i.e $R^{'}_{L_1} \cap R^{'}_{L_2} \cap \ldots \cap R^{'}_{L_\alpha} = \phi$. As no more globally applicable secrets remain in the node's memory, it is not possible by adversary to launch a collusion attack.

6 Conclusion

As secret communication is an important requirement in many sensor network applications, shared secret keys are used between communicating nodes to encrypt data. A key pre-distribution scheme is one of the common solutions for establishing secure communication in sensor networks. Random key pre-distribution schemes are vulnerable to collusion attacks because pre-loading global secrets onto exposed devices can be used in these attacks. In this paper, we propose a key distribution scheme that is robust against the collusion attack. Our scheme provides higher resiliency against node capture and collusion attack by deleting the initial key rings from their memory, after generating the shared pairwise key with neighbors. Further, it allows new nodes to join the system after initialization, even though the initial key ring has been destroyed from the node's memory.

7 Acknowledgment

This work is in part supported by Higher Education Commission (HEC) Pakistans International Research Support Initiative Programs scholarship given to Firdous Kausar to conduct her research at Acadia University, Canada. Further, we would like to thank National Science and Engineering Research Council (NSERC) Canada for their support in providing RTI and Discovery grants to Dr. Hussain at Acadia University, Canada. This research is also supported by Kyungnam University.

References

1. Akyildiz, I.F., Su, W., Sankarasubramaniam, Y., Cayirci, E.: A survey on sensor networks. IEEE Communications Magazine (August 2002)
2. Xiao, X., Rayi, V.K., Sun, B., Du, X., Hu, F., Galloway, M.: A survey of key management schemes in wireless sensor networks. Computer communications (2007)
3. Moore, T.: A collusion attack on pairwise key predistribution schemes for distributed sensor networks. In: PERCOMW '06: Proceedings of the 4th annual IEEE international conference on Pervasive Computing and Communications Workshops, Washington, DC, USA, IEEE Computer Society (2006) 251
4. Eschenauer, L., Gligor, V.D.: A key management scheme for distributed sensor networks. In: ACM CCS. (2002)
5. Chan, H., Perrig, A., Song, D.: Random key pre-distribution schemes for sensor networks. In: IEEE Symposium on Security and Privacy. (May 2003) 197–213
6. Oliveira, L.B., Wong, H.C., Bern, M., Dahab, R., Loureiro, A.A.F.: Sec leach: A random key distribution solution for securing clustered sensor networks. In: 5th IEEE international symposium on network computing and applications. (2006) 145–154
7. Heinzelman, W.R., Chandrakasan, A., Balakrishnan, H.: Energy-efficient communication protocol for wireless microsensor networks. In: IEEE Hawaii Int. Conf. on System Sciences. (2000) 4–7

8. Chan, H., Perrig, A., Song, D.: Random key pre-distribution schemes for sensor networkss. In: IEEE Symposium on Research in Security and Privacy. (2003)
9. Zhu, S., Xu, S., Setia, S., Jajodia, S.: Establishing pairwise keys for secure communication in ad hoc networks: A probabilistic approach. In: 11th IEEE International Conference on Network Protocols (ICNP'03). (2003)
10. Pietro, R.D., Mancini, L.V., Mei, A.: Random key assignment secure wireless sensor networks. In: 1st ACM workshop on Security of Ad Hoc and Sensor Networks. (2003)
11. Cheng, Y., Agrawal, D.P.: Efficient pairwise key establishment and management in static wireless sensor networks. In: Second IEEE International Conference on Mobile ad hoc and Sensor Systems. (2005)
12. Ren, K., Zeng, K., Lou, W.: A new approach for random key pre-distribution in large-scale wireless sensor networks. Wireless communication and mobile computing **6**(3) (2006) 307–318
13. Perrig, A., Szewczyk, R., Tygar, J., Victorwen, Culler, D.E.: Spins: Security protocols for sensor networks. In: Seventh Annual Int'l Conf. on Mobile Computing and Networks. (July 2001)
14. Blundo, C., Santis, A.D., Herzberg, A., Kutten, S., Vaccaro, U., Yung, M.: Perfectly-secure key distribution for dynamic conferences. In: CRYPTO '92: Proceedings of the 12th Annual International Cryptology Conference on Advances in Cryptology, London, UK, Springer-Verlag (1993) 471–486
15. Blom, R.: An optimal class of symmetric key generation systems. In: Proc. of the EUROCRYPT 84 workshop on Advances in cryptology: theory and application of cryptographic techniques, New York, NY, USA, Springer-Verlag New York, Inc. (1985) 335–338
16. Liu, D., Ning, P.: Location-based pairwise key establishments for static sensor networks. In: SASN '03: Proceedings of the 1st ACM workshop on Security of ad hoc and sensor networks, New York, NY, USA, ACM Press (2003) 72–82
17. Wadaa, A., Olariu, S., Wilson, L., Eltoweissy, M.: Scalable cryptographic key management in wireless sensor networks. In: ICDCSW '04: Proceedings of the 24th International Conference on Distributed Computing Systems Workshops - W7: EC (ICDCSW'04), Washington, DC, USA, IEEE Computer Society (2004) 796–802
18. Du, X., Xiao, Y., Guizani, M., Chen, H.H.: An effective key management scheme for heterogeneous sensor networks. Ad Hoc Networks **5**(1) (2007) 24–34
19. Traynor, P., Kumar, R., Saad, H.B., Cao, G., Porta, T.L.: Establishing pair-wise keys in heterogeneous sensor networks. In: INFOCOM 2006. 25th IEEE International Conference on Computer Communications. Proceedings. (2006) 1–12
20. Lu, K., Qian, Y., Hu, J.: A framework for distributed key management schemes in heterogeneous wireless sensor networks. In: IEEE International Performance Computing and Communications Conference. (2006) 513–519
21. Pietro, R.D., Mancini, L.V., Mei, A.: Energy efficient node-to-node authentication and communication confidentiality in wireless sensor networks. Wirel. Netw. **12**(6) (2006) 709–721