

# SECURITY IN WIRELESS SENSOR NETWORKS USING KEY MANAGEMENT SCHEMES

Soundarya.P<sup>1</sup> and Varalakshmi .L.M<sup>2</sup>

II nd year M.Tech<sup>1</sup> and Assistant proffesor<sup>2</sup>

[soundarya88@yahoo.co.in](mailto:soundarya88@yahoo.co.in)

[varalakshmi\\_1@yahoo.co.in](mailto:varalakshmi_1@yahoo.co.in)

Department of electronics and communication engineering

Sri Manakula Vinayagar Engineering College, Pondicherry.

## ***ABSTRACT***

*Wireless sensor networks pose new security and privacy challenges. One of the important challenges is how to bootstrap secure communications among nodes. Several key management schemes have been proposed. Key management plays an essential role in achieving security in wireless sensor networks (WSN). Due to resource constraints, achieving such key agreement in wireless sensor networks is nontrivial. Many key agreement schemes used in general networks, such as Diffie-Hellman and public-key based schemes, are not suitable for wireless sensor networks. Pre-distribution of secret keys for all pairs of nodes is not viable due to the large amount of memory used when the network size is large. In this paper, a new key pre-distribution scheme is proposed (DDHV SCHEME), which substantially improves the resilience of the network compared to the existing schemes (EG SCHEME). Our scheme exhibits a nice threshold property: when the number of compromised nodes is less than the threshold, the probability that any node other than these compromised nodes is affected is close to zero. This desirable property lowers the initial payoff of smaller scale network breaches to an adversary, and makes it necessary for the adversary to attack a significant proportion of the network.*

## ***KEYWORDS***

*Wireless sensor network, key predistribution, security*

## **1. INTRODUCTION**

RECENT advances in electronic and computer technologies have paved the way for the proliferation of wireless sensor networks (WSN). Sensor networks usually consist of a large number of ultra-small autonomous devices. Each device, called a sensor node, is battery powered and equipped with integrated sensors, data processing, and short-range radio communication capabilities. In typical application scenarios, sensor nodes are spread randomly over the deployment region under scrutiny and collect sensor data. Sensor networks are being deployed for a wide variety of applications, including military sensing and tracking, environment monitoring, patient monitoring and tracking, smart environments, etc. When sensor networks are deployed in a hostile environment, 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. To provide security, communication should be encrypted and authenticated. An open research problem is how to bootstrap secure communications among sensor nodes, i.e., how to set up secret keys among communicating nodes. This key agreement problem is a part of the key management problem, which has been widely studied in general network environments. There are three types of general key agreement schemes: the trusted-server scheme, the self-enforcing scheme, and the key predistribution scheme. The trusted-server scheme depends on a trusted server for key agreement between nodes eg: Kerberos [1]. This type of scheme is not suitable for sensor networks because there is usually no trusted infrastructure in sensor networks. The self-enforcing scheme depends on asymmetric cryptography, such as key agreement using public key certificates. However, limited computation and energy resources of sensor nodes often make it undesirable to use public key algorithms [2]. The third type of key agreement scheme is key predistribution, where key information is distributed among all sensor nodes prior to deployment. If we know which nodes are more likely to be in the same neighborhood before deployment, keys can be decided a priori. However, because of the randomness of deployment, it might be infeasible to learn the set of neighbors a priori. There exist a number of key predistribution schemes. A naive solution is to let all the nodes carry a master secret key. Any pair of nodes can use this global master secret key to achieve key agreement and obtain a new pairwise key. This scheme does not exhibit desirable network resilience: If one node is compromised, the security of the entire sensor network will be compromised. Some existing studies suggest storing the master key in tamper-resistant hardware to reduce the risk, but this increases the cost and energy consumption of each sensor. Furthermore, tamper resistant hardware might not always be safe [3]. Another key predistribution scheme is to let each sensor carry  $N - 1$  secret pairwise key, each of which is known only to this sensor and one of the other  $N - 1$  sensors (assuming  $N$  is the total number of sensors). The resilience of this scheme is perfect because compromising one node does not affect the security of communications among other nodes; however, this scheme is impractical for sensors with an extremely limited amount of memory because  $N$  could be large. Moreover, adding new nodes to a preexisting sensor network is difficult because the existing nodes do not have the new nodes' keys.

## 2. PROBLEM STATEMENT

In this paper, a new key pre-distribution scheme is proposed. The main contributions of this paper are as follows:

1. Substantially improved network resilience against node capture over existing schemes.
2. Pairwise keys that enable authentication.

This scheme builds on Blom's key pre-distribution scheme [4] and combines the random key pre-distribution method with it. The results show that the resilience of this scheme is substantially better than other random key pre-distribution schemes. In [4], Blom proposed a key pre-distribution scheme that allows *any* pair of nodes to find a secret pairwise key between them. Compared to the  $(N - 1)$ -pairwise-key pre-distribution scheme, Blom's scheme only uses  $\lambda + 1$  memory spaces with  $\lambda$  much smaller than  $N$ . The tradeoff is that, unlike the  $(N - 1)$ -pairwise-key scheme, Blom's scheme is not perfectly resilient against node capture. Instead it has the following  $\lambda$ -secure property: as long as an adversary compromises less than or equal to  $\lambda$  nodes, uncompromised nodes are perfectly secure; when an adversary compromises more than  $\lambda$  nodes, all pairwise keys of the entire network are compromised.

### 3. THE ESCHENAUER-GLIGOR (EG) SCHEME

The Eschenauer-Gligor scheme (referred to as the basic scheme or the EG scheme hereafter) proposed by Eschenauer and Gligor [5] consists of three phases: key predistribution, shared-key discovery, and path-key establishment. In the key predistribution phase, each sensor node randomly selects  $\tau$  distinct cryptographic keys from a key pool  $S$  and stores them in its memory. This set of  $\tau$  keys is called the node's key ring. The number of keys in the key pool,  $|S|$ , is chosen such that two random subsets of size  $\tau$  in  $S$  share at least one key with some probability  $p$ .

After the nodes are deployed, a key-setup phase is performed during this phase, each pair of neighboring nodes attempt to find a common key that they share. If such a key exists, the key is used to secure the communication link between these two nodes. After key-setup is complete, a graph (called key graph) of secure links is formed. Nodes can then set up path keys with their neighbors with whom they do not share keys. If the key graph is connected, a path can always be found from a source node to any of its neighbors. The source node can then generate a path key and send it securely via the path to the target node. The size of the key pool  $S$  is critical to both the connectivity and the resilience of the scheme. Connectivity is defined as the probability that any two neighboring nodes share one key. Resilience is defined as the fraction of the secure links that are compromised after a certain number of nodes are captured by the adversaries.

At one extreme, if the size of  $S$  is one, i.e.,  $|S| = 1$ , the scheme is actually reduced to the naive master-key scheme. This scheme yields a high connectivity, but it is not resilient against node capture because the capture of one node can compromise the whole network. At the other extreme, if the key pool is very large, e.g.,  $|S| = 100,000$ , resilience becomes much better, but connectivity of the sensor network becomes low. For example, as indicated by Eschenauer and Gligor, in this case, even when each sensor selects  $\tau = 200$  keys from this large key pool  $S$ , the probability that any two neighboring nodes share at least one key is only 0.33.

### 4. THE DU-DENG-HAN-VARSHNEY (DDHV) SCHEME

Blom proposed a key predistribution method that allows any pair of nodes in a network to be able to derive a pairwise secret key [6]. It has the property that, as long as no more than  $\lambda$  nodes are compromised, all communication links of non compromised nodes remain secure.

#### 4.1 Blom's Key Predistribution Scheme

During the pre-deployment phase, the base station first constructs a  $(\lambda + 1) \times N$  matrix  $G$  over a finite field  $GF(q)$ , where  $N$  is the size of the network.  $G$  is considered as public information; any sensor can know the contents of  $G$ , and even adversaries are allowed to know  $G$ . Then the base station creates a random  $(\lambda+1) \times (\lambda+1)$  symmetric matrix  $D$  over  $GF(q)$ , and computes an  $N \times (\lambda + 1)$  matrix  $A = (D \cdot G)^T$ , where  $(D \cdot G)^T$  is the transpose of  $(D \cdot G)$ . Matrix  $D$  needs to be kept secret, and should not be disclosed to adversaries or any sensor node (although, as will be discussed later, one row of  $(D \cdot G)^T$  will be disclosed to each sensor node). Because  $D$  is symmetric, it is easy to see:

$$\begin{aligned} A \cdot G &= (D \cdot G)^T \cdot G = G^T \cdot D^T \cdot G = G^T \cdot D \cdot G \\ &= (A \cdot G)^T \end{aligned}$$

This means that  $A \cdot G$  is a symmetric matrix. If  $K = A \cdot G$ , it is known that  $K_{ij} = K_{ji}$ , where  $K_{ij}$  is the element in  $K$  located in the  $i$ th row and  $j$ th column.  $K_{ij}$  (or  $K_{ji}$ ) is considered as the pairwise key between node  $i$  and node  $j$ . Fig. 1 illustrates how the pairwise key  $K_{ij} = K_{ji}$  is generated. To

carry out the above computation, nodes  $i$  and  $j$  should be able to compute  $K_{ij}$  and  $K_{ji}$ , respectively. This can be easily achieved using the following key pre-distribution scheme, for  $k = 1 \dots N$ :

1. Store the  $k$ th row of matrix  $A$  at node  $k$ , and
2. Store the  $k$ th column of matrix  $G$  at node  $k$ .

Therefore, when nodes  $i$  and  $j$  need to find the pairwise key between them, they first exchange their columns of  $G$ , and then they can compute  $K_{ij}$  and  $K_{ji}$ , respectively, using their private rows of  $A$ . Because  $G$  is public information, its columns can be transmitted in plaintext. It has been proved in [4] that the above scheme is  $\lambda$ -secure if any  $\lambda + 1$  columns of  $G$  are linearly independent. This  $\lambda$ -secure property guarantees that no nodes other than  $i$  and  $j$  can compute  $K_{ij}$  or  $K_{ji}$  if no more than  $\lambda$  nodes are compromised.

### 4.2 Multiple-Space Key Pre-Distribution Scheme

To achieve better resilience against node capture, a new key pre-distribution scheme that uses Blom's method as a building block is proposed. The idea is based on the following observations: Blom's method guarantees that any pair of nodes can find a secret key between themselves. To represent this, concepts from graph theory is used and draw an edge between two nodes if and only if they can find a secret key between themselves. Complete graph is obtained (i.e., an edge exists between all node pairs). Although full connectivity is desirable, it is not necessary. To achieve our goal of key agreement, all we need is a connected graph, rather than a complete graph. Our hypothesis is that by requiring the graph to be only connected, each sensor node needs to carry less key information.

Before we describe our proposed scheme, we define a *key space* (or *space* in short) as a tuple  $(D, G)$ , where matrices  $D$  and  $G$  are as defined in Blom's scheme. We say a node picks a key space  $(D, G)$  if the node carries the secret information generated from  $(D, G)$  using Blom's scheme. Two nodes can calculate their pairwise key if they have picked a common key space.

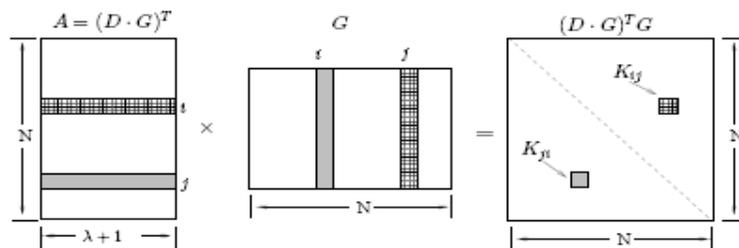


Figure 1: Generating Keys in Blom's Scheme

#### 4.2.1 Key Pre Distribution Phase

During the key pre-distribution phase, key information is assigned to each node, such that after deployment, neighboring sensor nodes can find a secret key between them. Assume that each sensor node has a unique identification, whose range is from 1 to  $N$ . We also select the security parameters  $\tau$ ,  $\omega$ , and  $\lambda$ , where  $2 \leq \tau < \omega$ . Key pre-distribution phase contains the following steps: A primitive element from a finite field  $GF(q)$  is selected, where  $q$  is the smallest prime larger

than the key size, to create a generator matrix  $G$  of size  $(\lambda+1) \times N$ . Let  $G(j)$  represent the  $j$ th column of  $G$ . We provide  $G(j)$  to node  $j$ . As it is already shown in Section 4.1, although  $G(j)$  consists of  $(\lambda+1)$  elements, each sensor only needs to remember one seed (the second element of the column), which can be used to regenerate all the elements in  $G(j)$ . Therefore the memory usage for storing  $G(j)$  at a node is just a single element. Since the seed is unique for each sensor node, it can also be used for node id. Generate  $\omega$  symmetric matrices  $D_1, D_2, \dots, D_\omega$  of size  $(\lambda+1) \times (\lambda+1)$ . Each tuple  $S_i = (D_i, G)$ ,  $i = 1, \dots, \omega$  is called as key space. Then compute the matrix  $A_i = (D_i \times G)^T$ . Let  $A_i(j)$  represents the  $j$ th row of  $A_i$ . Randomly select  $\tau$  distinct key spaces from the  $\omega$  key spaces for each node. For each space  $S_i$  selected by node  $j$ , we store the  $j$ th row of  $A_i$  (i.e.  $A_i(j)$ ) at this node. This information is secret and should stay within the node; under no circumstance should a node send this secret information to any other node. According to Blom's scheme, two nodes can find a common secret key if they have both picked a common key space. Since  $A_i$  is an  $N \times (\lambda+1)$  matrix,  $A_i(j)$  consists of  $(\lambda+1)$  elements. Therefore, each node needs to store  $(\lambda+1)\tau$  elements in its memory. Because the length of each element is the same as the length of secret keys, the memory usage of each node is  $(\lambda+1)\tau$ .

#### 4.2.2 Key Agreement Phase

After deployment, each node needs to discover whether it shares any space with its neighbors. To do this, each node broadcasts a message containing the following information:

1. The node's id,
2. The indices of the spaces it carries
3. The seed of the column of  $G$  it carries.

Assume that nodes  $i$  and  $j$  are neighbors, and they have received the above broadcast messages. If they find out that they have a common space, e.g.  $S_c$ , they can compute their pairwise secret key using Blom's scheme: Initially node  $i$  has  $A_c(i)$  and seed for  $G(i)$ , and node  $j$  has  $A_c(j)$  and seed for  $G(j)$ . After exchanging the seeds, node  $i$  can regenerate  $G(j)$  and node  $j$  can regenerate  $G(i)$ ; then the pairwise secret key between nodes  $i$  and  $j$ ,  $K_{ij} = K_{ji}$ , can be computed in the following manner by these two nodes independently:

$$K_{ij} = K_{ji} = A_c(i) \bullet G(j) = A_c(j) \bullet G(i)$$

After secret keys with neighbors are set up, the entire sensor network forms the following Key-Sharing Graph: Let  $V$  represent all the nodes in the sensor network. A Key-Sharing graph  $G_{ks}(V, E)$  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 (1) nodes  $i$  and  $j$  have at least one common key space, and (2) nodes  $i$  and  $j$  can reach each other within the wireless transmission range.

If two neighboring nodes  $i$  and  $j$ , do not share a common key space, they could still come up with a pairwise secret key between them. The idea is to use the secure channels that have already been established in the key-sharing graph  $G_{ks}$ : as long as  $G_{ks}$  is connected, two neighboring nodes  $i$  and  $j$  can always find a path in  $G_{ks}$  from  $i$  to  $j$ . Assume that the path is  $i, v_1, \dots, v_t, j$ . To find a common secret key between  $i$  and  $j$ ,  $i$  first generates a random key  $K$ . Then  $i$  sends the key to  $v_1$  using the secure link between  $i$  and  $v_1$ ;  $v_1$  sends the key to  $v_2$  using the secure link between  $v_1$  and  $v_2$ , and so on until  $j$  receives the key from  $v_t$ . Nodes  $i$  and  $j$  use this secret key  $K$  as their pairwise key. Because the key is always forwarded over a secure link, no nodes beyond this path can find out the key.

## 5. CONNECTIVITY ANALYSIS

### 5.1 For EG Scheme

The probability that two key rings share at least a key is  $1 - \Pr$  [two nodes do not share any key]. To compute the probability that two key rings do not share any key, each key of a key ring should be drawn out of a pool of  $P$  keys without replacement. Thus, the number of possible key rings is:

$$\frac{P!}{k!(P-K)!}$$

Select the first key ring. The total number of possible key rings that do not share a key with this key ring is the number of key-rings that can be drawn out of the remaining  $P - k$  unused key in the pool, namely:

$$\frac{(P-K)!}{k!(P-2k)!}$$

Therefore, the probability that no key is shared between the two rings is the ratio of the number of rings without a match by the total number of rings. Thus, the probability that there is at least a shared key between two key rings is:

$$\frac{k!(P-K)!(P-K)!}{P!k!(P-2k)!}$$

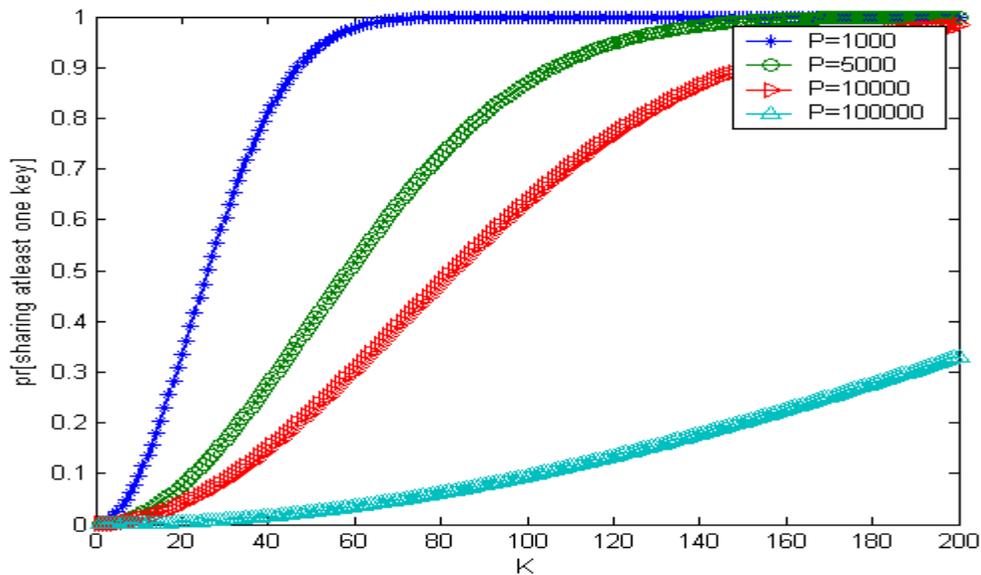


Figure 2: Probability of sharing at least one key when two nodes choose  $k$  keys from a pool of size  $P$

Figure 2 illustrates a plot of this function for various values of P. For example, one may see that for a pool size P = 10,000 keys, only 75 keys need to be distributed to any two nodes to have the probability p = 0.5 that they share a key in their key ring. If the pool is ten times larger, namely P = 100,000, the number of keys required is 250, which is only 3.3 times the number of keys distributed in the case P = 10,000. This provides intuition for the scalability of this approach. Of course, to determine the final the size of the key ring we need to provision for addition of new nodes, revocation, and re-keying. The scalability properties of the solution indicate that such provisioning will have minimal impact on the size of key rings.

### 5.1 For DDHV Scheme

To make it possible for any pair of nodes to be able to find a secret key between them, the key sharing graph G<sub>ks</sub>(V,E) needs to be connected. Given the size and the density of a network, how to select the values for ω and τ, such that the graph G<sub>ks</sub> is connected with high probability? We use the following three-step approach, which is adapted from [8].

**Computing Required Local Connectivity:** Let P<sub>c</sub> be the probability that the key-sharing graph is connected, called as global connectivity. Local connectivity is used to refer to the probability of two neighboring nodes sharing at least one space (i.e. they can find a common key between them). The global connectivity and the local connectivity are related: to achieve a desired global connectivity P<sub>c</sub>, the local connectivity must be higher than a certain value; we call this value the required local connectivity, denoted by P<sub>required</sub>. Using connectivity theory in a random-graph by Erdős and Rényi [9], the necessary expected node degree d (i.e., the average number of edges connected to each node) for a network of size N when N is large can be obtained in order to achieve a given global connectivity, P<sub>c</sub>:

$$d = \frac{(N-1)}{N} [\ln(N) - \ln(-\ln(P_c))]$$

For a given density of sensor network deployment, let n be the expected number of neighbors within wireless communication range of a node. Since the expected node degree must be at least d as calculated above, the required local connectivity p<sub>required</sub> can be estimated as:

$$p_{required} = \frac{d}{n}$$

**Computing Actual Local Connectivity:** After selecting the values for ω and τ, the actual local connectivity is determined by these values. Use p<sub>actual</sub> to represent the actual local connectivity, namely p<sub>actual</sub> is the actual probability of any two neighboring nodes sharing at least one space (i.e. they can find a common key between them). Since p<sub>actual</sub> = 1 - Pr (two nodes do not share any space).

$$p_{actual} = 1 - \frac{\binom{\omega}{\tau} \binom{\omega - \tau}{\tau}}{\binom{\omega}{\tau}^2} = 1 - \frac{((\omega - \tau)!)^2}{(\omega - 2\tau)! \omega!}$$

The values of  $p_{actual}$  have been plotted in Fig. 3 when  $\omega$  varies from  $\tau$  to 100 and  $\tau = 2, 4, 6, 8$ . For example, one can see that, when  $\tau = 4$ , the largest  $\omega$  that we can choose while achieving the local connectivity  $p_{actual} \geq 0.5$  is 25.

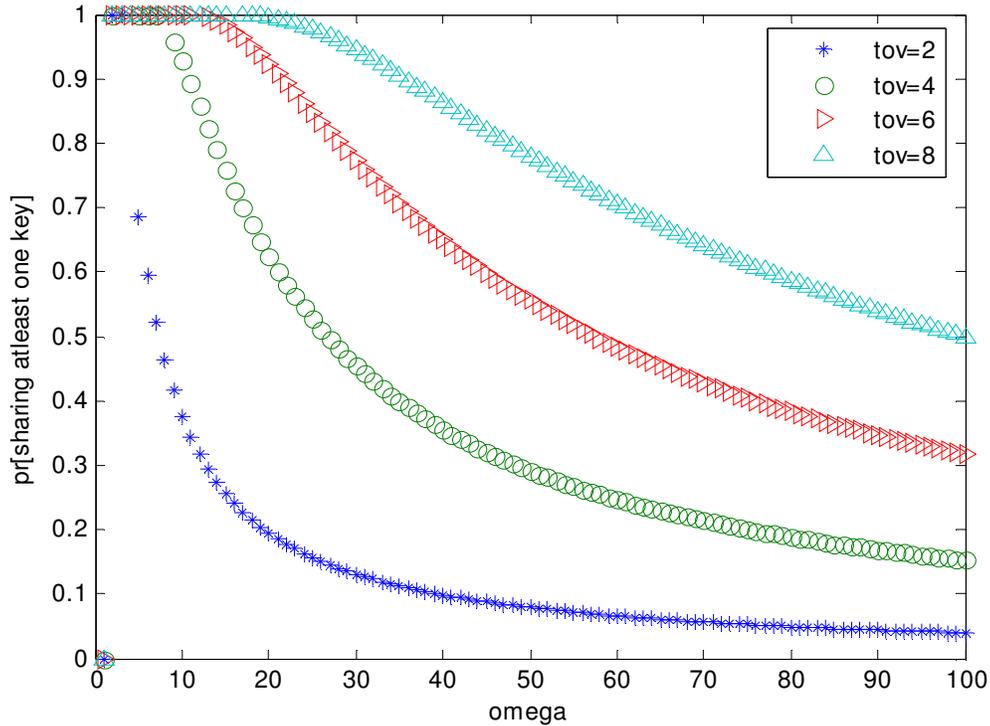


Figure 3: Probability of sharing at least one key when two nodes each randomly chooses  $\tau$  spaces from  $\omega$  spaces.

The collection of sets of spaces assigned to each sensor form a probabilistic quorum system : the desire is that every two sensors have a space in common with high probability. Furthermore, it

can be shown that if  $\tau \geq \sqrt{\ln \frac{1}{1 - p_{actual}}} \sqrt{\omega}$ , then the probability of intersection is at least

$p_{actual}$ ; this has the similar property to the birthday paradox. For example, when  $\tau \geq \sqrt{\ln 2} \sqrt{\omega}$ , the probability of intersection is at least 1/2. This can explain the behavior of figure 3.

**Computing  $\omega$  and  $\tau$ :** Knowing the required local connectivity required and the actual local connectivity  $p_{actual}$ , in order to achieve the desired global connectivity  $P_c$ , we should have  $P_{actual} \geq P_{required}$ .

$$1 - \frac{((\omega - \tau)!)^2}{(\omega - 2\tau)! \omega!} \geq \frac{(N - 1)}{nN} [\ln(N) - \ln(-\ln(P_c))]$$

Therefore, in order to achieve a certain  $P_c$  for a network of size  $N$  and the expected number of neighbors for each node being  $n$ , to find values of  $\omega$  and  $\tau$ , such that Inequality is satisfied.

## 6. SECURITY ANALYSIS

The multiple-space key pre-distribution scheme is evaluated in terms of its resilience against node capture. The evaluation is based on two metrics: (1) When  $x$  nodes are captured, what is the probability that at least one key space is broken? Because of the  $\lambda$ -secure property of our scheme, to break a key space, an adversary needs to capture  $\lambda+1$  node that contain this key space's information; otherwise, the key space is still perfectly secure. This analysis shows when the network starts to become insecure. (2) When  $x$  nodes are captured, what fraction of the additional communication (i.e. communication among uncaptured nodes) also becomes compromised? This analysis shows how much payoff an adversary can gain after capturing a certain number of nodes.

### 6.1 Probability of Atleast One Space Being Broken

The unit of memory size is defined as the size of a secret key (e.g. 64 bits). According to Blom's scheme, if a space is  $\lambda$ -secure, each node needs to use memory of size  $\lambda + 1$  to store the space information. Therefore, if the memory usage is  $m$  and each node needs to carry  $\tau$  spaces, then the value of  $\lambda$  should be  $[m/\tau] - 1$ . In the following analysis, we choose  $\lambda = [m/\tau] - 1$ .

Let  $S_i$  be the event that space  $S_i$  is broken, where  $i = 1, \dots, \omega$ , and  $C_x$  be the event that  $x$  nodes are compromised in the network. Furthermore, let  $S_i \cup S_j$  be the joint event that either space  $S_i$  or space  $S_j$ , or both, is broken and  $\theta = \tau \omega$ . Hence,  $\Pr(\text{at least one space is broken} | C_x) = \Pr(S_1 \cup S_2 \cup \dots \cup S_\omega | C_x)$ . According to the Union Bound,

$$\Pr(S_1 \cup \dots \cup S_\omega | C_x) \leq \omega X_i \leq \sum_{i=1}^{\omega} \Pr(S_i | C_x)$$

Due to the fact that each key space is broken with equal probability,

$$\sum_{i=1}^{\omega} \Pr(S_i | C_x) = \omega \Pr(S_1 | C_x)$$

Therefore,

$$\Pr(\text{at least one space is broken} | C_x) \leq \sum_{i=1}^{\omega} \Pr(S_i | C_x) = \omega \Pr(S_1 | C_x)$$

Now need to calculate  $\Pr(S_1 | C_x)$ , the probability of space  $S_1$  being compromised when  $x$  nodes are compromised. Because each node carries information from  $\tau$  spaces, the probability that each compromised node carries information about  $S_1$  is  $\theta = \tau \omega$ . Therefore, after  $x$  nodes are compromised, the probability that exactly  $j$  of these  $x$  nodes contain information about  $S_1$

is  $\binom{x}{j} \theta^j (1-\theta)^{x-j}$  since space S1 can only be broken after at least  $\lambda+1$  node are compromised, the following result is:

$$\leq \omega \sum_{j=\lambda+1}^{\omega} \binom{x}{j} \theta^j (1-\theta)^{x-j}$$

Combining Inequality and Equation, we have the following upper bound is:

Pr (at least one space is broken | Cx)

$$\begin{aligned} &\leq \omega \sum_{j=\lambda+1}^{\omega} \binom{x}{j} \theta^j (1-\theta)^{x-j} \\ &= \leq \omega \sum_{j=\lambda+1}^{\omega} \binom{x}{j} \left(\frac{\tau}{\omega}\right)^j \left(1-\frac{\tau}{\omega}\right)^{x-j} \end{aligned}$$

Plot both simulation and analytical results in Fig. 3. From the figure, the two results match each other closely, meaning that the union bound works quite well in the scenarios as discussed. Fig. 4 shows, for example, when the memory usage is set to 200,  $\omega$  is set to 50, and  $\tau$  is set to 4, the value of  $\lambda$  for each space is  $49 = \lfloor 200/4 \rfloor - 1$ , but an adversary needs to capture about 380 nodes in order to be able to break at least one key space with non-negligible probability.

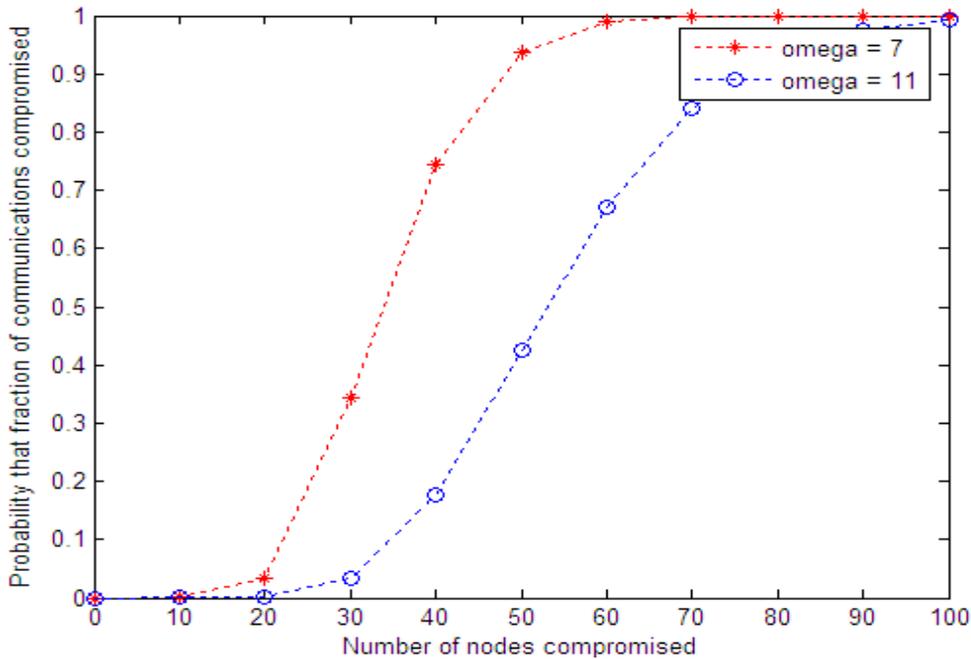


Figure 4: fraction of communication compromised

## 6.2 Authentication Property

Due to the property of Blom’s scheme, all keys generated in a space are pair wise keys. Therefore, when the space is not yet compromised, keys in this space can be used for authentication purposes. After the space is broken, adversaries can generate all the pair wise keys in that space, and keys in that space can no longer be used for authentication purposes. According to the analysis, adversaries need to compromise a significant number of nodes in order to compromise a space.

## 6.3 Fraction Of Communication Compromised

To understand the resilience of this key pre-distribution scheme, find out how the capture of  $x$  sensor nodes by an adversary affects the rest of the network. In particular, find out the fraction of additional communications (i.e., communications among uncaptured nodes) that an adversary can compromise based on the information retrieved from the  $x$  captured nodes. To compute this fraction, first compute the probability that any one of the additional communication links is compromised after  $x$  nodes are captured. Only consider the links in the key-sharing graph, and each of these links is secured using a pair wise key computed from the common key space shared by the two nodes of this link. After the key setup stage, two neighboring nodes can use the established secure links to agree upon another random key to secure their communication. Because this key is not generated from any key space, the security of this new random key does not directly depend on whether the key spaces are broken. However, if an adversary can record all the communications during the key setup stage, he/she can still compromise this new key after compromising the corresponding links in the key-sharing graph.

Let  $c$  be a link in the key-sharing graph between two nodes that are not compromised, and  $K$  be the communication key used for this link. Let  $B_i$  represent the joint event that  $K$  belongs to space  $S_i$  and space  $S_i$  is compromised.  $K \in S_i$  is used to represent that “ $K$  belongs to space  $S_i$ ”. The probability of  $c$  being broken given  $x$  nodes are compromised is:

$$\Pr(c \text{ is broken} | C_x) = \Pr(B_1 \cup B_2 \cup B_3 \dots \cup B_\omega | C_x)$$

Since  $c$  can only use one key, events  $B_1 \dots B_\omega$  are mutually exclusive.

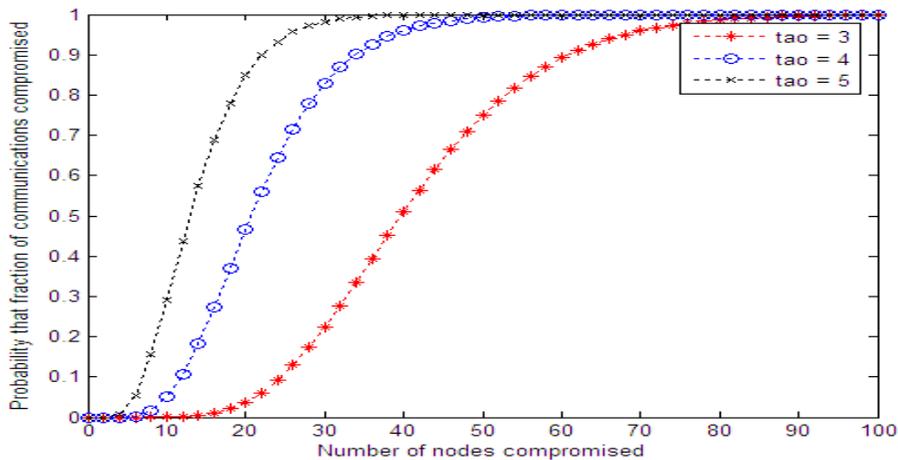


Figure 5: fraction of communication compromised

Therefore,

$$\Pr(c \text{ is broken} | C_x) = \sum_{i=1}^{\omega} \Pr(B_i | C_x) = \omega \Pr(B_1 | C_x)$$

Because all events  $B_i$  are equally likely. Note that

$$\Pr(B_1 | C_x) = \frac{\Pr((K \in S_1) \cap (S_1 \text{ is compromised}) \cap C_x)}{\Pr(C_x)}$$

Since the event  $(K \in S_1)$  is independent of the event  $C_x$  or the event  $(S_1 \text{ is compromised})$ ,

$$\begin{aligned} \Pr(B_1 | C_x) &= \frac{\Pr((K \in S_1) \bullet \Pr(S_1 \text{ is compromised} | C_x))}{\Pr(C_x)} \\ &= \Pr((K \in S_1) \bullet \Pr(S_1 \text{ is compromised} | C_x)) \end{aligned}$$

$\Pr(S_1 \text{ is compromised} | C_x)$  can be calculated. The probability that  $K$  belongs to space  $S_1$  is the probability that link  $c$  uses a key from space  $S_1$ . Since the choice of a space from  $\omega$  key spaces is equally probable,

$$\Pr(K \in S_1) = \Pr(\text{the link } c \text{ uses a key from space } S_1) = \frac{1}{\omega}$$

Therefore,

$$\begin{aligned} \Pr(c \text{ is broken} | C_x) &= \omega \Pr(B_1 | C_x) = \omega \bullet \frac{1}{\omega} \bullet \Pr(S_1 \text{ is compromised} | C_x) \\ &= \Pr(S_1 \text{ is compromised} | C_x) = \sum_{j=\lambda+1}^{\omega} \binom{x}{j} \left(\frac{\tau}{\omega}\right)^j \left(1 - \frac{\tau}{\omega}\right)^{x-j} \end{aligned}$$

Assume that there are  $\gamma$  secure communication links that do not involve any of the  $x$  compromised nodes. Given the probability  $\Pr(c \text{ is broken} | C_x)$ , the expected fraction of broken communication links among those  $\gamma$  links is

$$\begin{aligned} &= \gamma \bullet \Pr(c \text{ is broken} | C_x) = \Pr(c \text{ is broken} | C_x) \\ &= \Pr(S_1 \text{ is compromised} | C_x) \end{aligned}$$

The above equation indicates that, given that  $x$  nodes are compromised, the fraction of the compromised secure communication links outside of those  $x$  compromised nodes is the same as the probability of one space being compromised. This can be explained quite intuitively. Since spaces are selected in an equally likely fashion during the key pre-distribution process, after  $x$  nodes are compromised, the expected number of spaces that are compromised is about  $\omega \Pr(S_1 \text{ is compromised} | C_x)$ . Therefore, the fraction of the spaces that are compromised is  $\Pr(S_1 \text{ is compromised} | C_x)$ . Because keys from different spaces are evenly selected by the

communication links, the fraction of communication links compromised should be the same as the fraction of the spaces compromised. Therefore, the fraction of the spaces compromised is also  $\Pr(S1 \text{ is compromised} | C_x)$ .

### 6.4 Comparison:

The figure 6 clearly shows the advantage of DDHV scheme. For example, when the memory usage  $m$  is the same ( $m = 200$ ), and  $P_{actual} = 0.33$ , with Eschenauer-Gligor schemes, an adversary only needs to compromise less than 100 nodes in order to compromise 10% of the rest of the secure links, whereas in DDHV scheme, the adversary needs to compromise 500 nodes. Therefore, DDHV scheme quite substantially lowers the initial payoff to the adversary of smaller scale network breaches. The same technique can also be applied to this scheme to improve the security of our scheme as well. Regarding the original Blom's scheme, because  $m = 200$ , the network is perfectly secure if less than 200 nodes are compromised; the network is completely compromised when 200 nodes are compromised ( $P_{actual}$  is always equal to 1 in Blom's scheme).

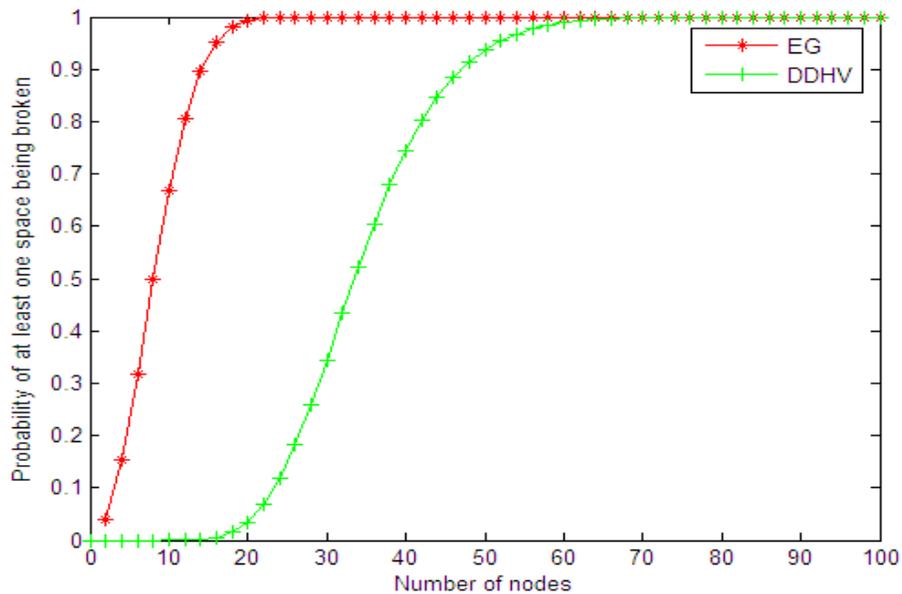


Figure 6: comparison of EG and DDHV scheme.

## 7. SECURITY IMPROVEMENT

In this section a way to further improve the security of our key pre-distribution scheme is discussed. Based on Inequality,

$$1 - \left(1 - \frac{\tau}{\omega}\right) \left(1 - \frac{\tau}{\omega-1}\right) \dots \left(1 - \frac{\tau}{\omega-\tau+1}\right) \geq \frac{(N-1)}{nN} [\ln(N) - \ln(-\ln(P_c))]$$

Notice that the left side is smaller when  $\omega$  is larger, and the right side is smaller when  $n$  is larger when other parameters are fixed. Therefore, when the network size  $N$ , the global connectivity  $P_c$ , and  $\tau$  are fixed, we can select a larger  $\omega$  if the expected number of neighbors  $n$  increases while still satisfying the above inequality. It is known from Inequality that the larger

the value of  $\omega$  is, the more resilient the network will be. Therefore, increasing  $n$  can lead to security improvement.

There are two ways to increase  $n$  for an existing sensor network: the first is to increase the communication range, but this also increases energy consumption. The second way is to use two-hop neighbors. A two-hop neighbor of node  $v$  is a node that can be reached via one of  $v$ 's one-hop (or direct) neighbors. To send a message to a two-hop neighbor,  $v$  needs to ask its direct neighbor to forward the message. Since the intermediate node only forwards the message and does not need to read the contents of the message, there is no need to establish a secure channel between the sender and the intermediate node, or between the intermediate node and the two-hop neighbor. As long as the sender and its two hop neighbor can establish a secure channel, the communication between them will be secured.

If two nodes,  $i$  and  $j$ , are two-hop neighbors and both of them carry key information from a common key space, they can find a secret key between themselves using the following approach: First, they find an intermediate node  $I$  that is a neighbor to both of them. Nodes  $i$  and  $j$  then exchange their identities and public part of key space information via  $I$ . Then,  $i$  and  $j$  find a common key space, and compute their secret key in that common key space.  $i$  and  $j$  can then encrypt any future communication between themselves using this secret key. Although all future communication still needs to go through an intermediate node, e.g.,  $I$ , the intermediate node cannot decrypt the message because it does not have the key.

After all direct neighbors and two-hop neighbors have established secure channels among themselves, the entire network forms an *Extended Key-Sharing Graph*  $G_{ks}$ , in which two nodes are connected by an edge if there is a secure channel between them, i.e. these two nodes (1) have at least one common key space, and (2) are either direct neighbors or two-hop neighbors. Once  $G_{ks}$  has been formed, key agreement between any pair of two neighboring nodes  $i$  and  $j$  can be performed based on  $G_{ks}$  in the same way as it is performed based on the original Key-Sharing Graph  $G_k$ . The difference between this scheme and the  $G_{ks}$ -based key agreement scheme is that in the  $G_{ks}$ -based key agreement scheme, some edges along a secure path might be an edge between two-hop neighbors, thus forwarding is needed.

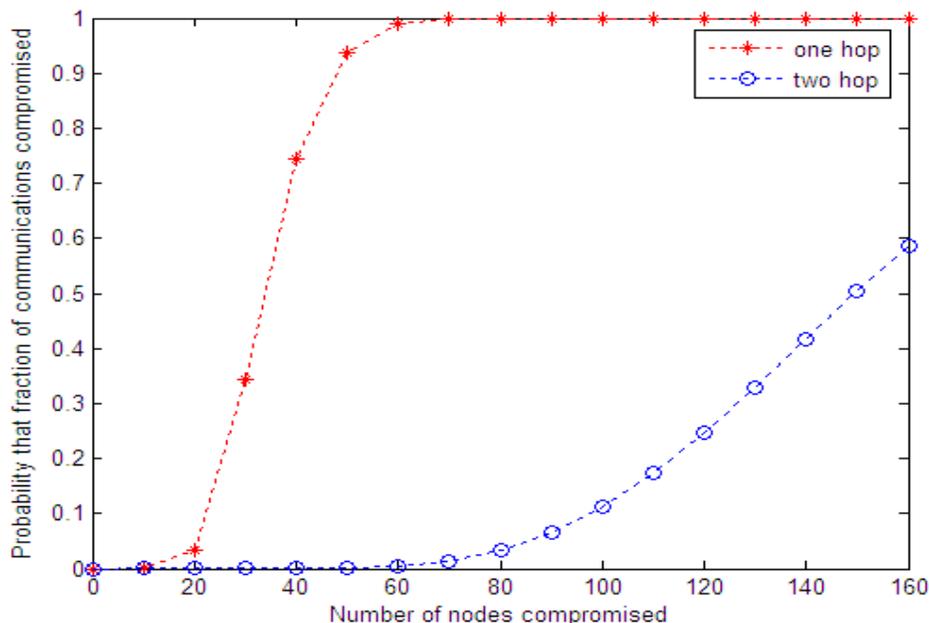


Figure 7: Security improvement using two hop method

## 8. CONCLUSION

A new pairwise key pre-distribution scheme for wireless sensor networks has been presented in this paper. This scheme has a number of appealing properties. First, this scheme is scalable and flexible. For a network that uses 64-bit secret keys, this scheme allows up to  $N = 264$  sensor nodes. These nodes do not need to be deployed at the same time; they can be added later, and still be able to establish secret keys with existing nodes. Second, compared to existing key pre-distribution schemes, this scheme is substantially more resilient against node capture. The analysis and simulation results have shown, for example, that to compromise 10% of the secure links in the network secured using DDHV scheme, an adversary has to compromise 5 times as many nodes as he/she has to compromise in a network secured by Eschenauer- Gligor scheme. Furthermore, it also shown that network resilience can be further improved if we use multi-hop neighbors.

## REFERENCES

- [1] B.C. Neuman and T. Tso, "Kerberos: An Authentication Service for Compute Networks," IEEE Comm., vol. 32, no. 9, pp. 33-38, Sept. 1994.
- [2] A. Perrig, R. Szewczyk, V. Wen, D. Cullar, and J.D. Tygar, "Spins: Security Protocols for Sensor Networks," Proc. Seventh Ann. ACM/ IEEE Int'l Conf. Mobile Computing and Networking (MobiCom), pp. 189-199, July 2001.
- [3] R. Anderson and M. Kuhn. Tamper resistance - a cautionary note. In *Proceedings of the Second Usenix Workshop on Electronic Commerce*, pages 1–11, November 1996.
- [4] R. Blom. An optimal class of symmetric key generation systems. *Advances in Cryptology: Proceedings of EUROCRYPT 84 (Thomas Beth, Norbert Cot, and Ingemar Ingemarsson, eds.), Lecture Notes in Computer Science, Springer-Verlag*, 209:335–338, 1985.
- [5] H. Chan, A. Perrig, and D. Song, "Random Key Predistribution Schemes for Sensor Networks," IEEE Symp. Security and Privacy, pp. 197-213, 2003.
- [6] F. J. MacWilliams and N. J. A. Sloane. *The Theory of Error-Correcting Codes*. New York, NY: Elsevier Science Publishing Company, Inc., 1977.
- [7] L. Eschenauer and V. D. Gligor. A key-management scheme for distributed sensor networks. In *Proceedings of the 9<sup>th</sup> ACM conference on Computer and communications security*, November 2002.
- [8] Erdős and Rényi. On random graphs I. *Publ. Math. Debrecen*, 6:290–297, 1959.
- [9] D. Malkhi, M. Reiter, A. Wool, and R. N. Wright. Probabilistic quorum systems. *Information and Computation*, (2):184–206, November 2001.

## ACKNOWLEDGEMENTS

### Authors

From Pre-KG onwards P.SOUNDARYA had her education in an independent and creative environment. Her thirst for knowledge has driven her to study up to M.Tech, and her aim is to do research in security issues in wireless sensor networks.

