One Hop Key Management Scheme for Wireless Sensor Network Using Deployment Knowledge

Avinash Kumar Chaurasia*†, R. K. Ghosh†
Email: {avinashc, rkg}@cse.iitk.ac.in
*Student author
†Indian Institute of Technology, Kanpur, India

***Abstract*—In this paper we propose a light-weight random key pre-distributed scheme based on deployment knowledge. It uses only one hop for key distribution. It leads to a substantial reduction in energy requirement while improving key sharing and resilience factor compared to the scheme proposed in [1]. We develop the theoretical background of the proposed scheme and explain why the proposed scheme performs better than [1]. In addition, we simulated both schemes over TOSSIM to establish that proposed scheme substantially improves key sharing over [1] in practice.**

I. Introduction

Key agreement protocols have been intensively studied for wired networks. While an extensive body of literature exists for wired and also for WiFi based wireless networks, most of the proposed scheme can not easily be extended for securing interactions involving WSNs. Tackling security of WSN poses formidable challenges due to inherent resource and computation constraints under which sensors operate. Cryptographic techniques such as RSA [5] based on asymmetric keys, or Kerberos [4] based on trusted third party are absolutely unsuitable for WSNs. However, Symmetric Key Cryptography (SKC) schemes appear to offer strong candidates for secure key management in WSNs.

A number of key pre-distribution schemes exist in literature [7]. An extremely simple minded key pre-distribution scheme could be to use a globally shared master key which all nodes in a WSN carry with them prior to deployment. Then every pair of nodes use this master key to establish a pair of shared keys which they would use for encryption and decryption of messages exchanged between them. While the above scheme provides greater connectivity and improved energy efficiency, it exposes network to vulnerability. By capturing just a single node, the global master key can be extracted, and all communication subsequently become insecure. Another simple solution could be pre-store all nodes keys in each and every node. From a security point of view such an approach is ideal as the attacker can only be privy to communication between captured node and other nodes in direct communication with the captured node. Entire network is not compromised. However, the approach relies on the fact that a sensor node has sufficient memory to store large number of keys.

An interesting random key pre-distribution scheme has been proposed in [1]. It use the knowledge of spatial closeness in deployment. The idea is that groups of sensor dropped from air together are likely to be spatially close. The scheme proposed in this paper is based on the generic idea of using spatial closeness in deployment for establishing shared keys. We show that the performance of the proposed scheme is better than that of [1] with the help of simulations and analytical explanation. We will refer to the scheme of [1] as the basic scheme for the purpose of comparison with the scheme proposed in this paper.

The main contributions of this paper can be summarized as follows:

1) Even after using deployment knowledge [1], a substantial number of nodes may still have to go through key establishment phase, because no guarantee can be given about spatially close nodes to have common keys. A path key establishment phase will be needed for this purpose. The scheme proposed in this paper requires exchange of just 3 messages to establish the key between neighbouring nodes.
2) We develop a novel key pre-distribution scheme which allows sensor nodes to share an increased percentage of keys with its neighbors compared to that of [1].
3) In the proposed scheme we are able to increase resilience against node capture because communication over a channel is secured by double encryption by using two keys.

II. Related Work

The main objective of the key management protcols for WSNs was to balance three competing requirements, namely, low memory usage, high network connectivity and the resilience factor (the nodes number to which an adversary would need to capture for extracting all keys). A comprehensive survey of research in this area is available in [7].

The key predistribution scheme proposed in this paper was developed around the one proposed by Varshney et al [1]. Varshney et al [1] improved up on the basic key predistribution steps of Eschenauer and Gligor [2]. So, the latter work is important starting point for the work reported in this paper.

The key management scheme proposed in [2] consists of three phases: key pre-distribution phase, shared key-discovery phase, and path-key establishment phase. In the key pre-distribution phase, each node in the network selects m keys from S, total number of keys $\mid S \mid$ is chosen such that any

node share at least one key with some probability p. In the shared key-discovery phase, each pair of nodes tries to find out whether they share any key or not, if they find one they used it to secure communication link between them otherwise they used next phase of the scheme to establish a key. After this phase connected graph of secure links is formed. In the path-key establishment phase, pair of nodes who don't have any key shared, tries to find out a path in a graph, if graph is connected there is always one such path exists. Using that path one of the pair of nodes send key k to other node.

Chan et al [6] proposed a random pairwise scheme for key predistribution. It combines random graph approach with pairwise key assignment. The difference between the above scheme and the scheme proposed in [2] is that $q \geq 1$ common keys instead of a single key is required to establish communication between a pair of nodes. Therefore, though Chan et al's scheme is more efficient than Eschenauer and Gligor's protocol, it requres more storage.

The scheme Proposed by Varshney et al [1] works significantly better than random key distribution based key management scheme proposed by Eschenaur-Gligor [2]. This scheme still loses lots of energy in establishing a key by broadcasting its key so that it can find a secure path to transfer one key. Most of the time when sensor nodes are deployed we already had a knowledge of their approximate location. Let us assume group of sensor nodes is deployed by airplane in patches one after another, this lead us to conclusion that sensor nodes which are in same group and are in neighbors group are near to each other with very high probability. So if there are n neighbors we just need to save n keys in a node. As deployment is random we can not guarantee that any two neighbor nodes certainly find a common key. Varshney et al [1] proposed a scheme in which any two nodes find a common key with certain probability p.

They modeled the distribution of sensor nodes as probability density function (pdf) $f_i(x,y) = \frac{1}{XY}$ where area of deployment is $X \times Y$ and $x \in [0, X]$ and $y \in [0, Y]$. When PDF is uniform, no information can be obtained on location of node. Nodes are deployed at points given by [1] is shown in Fig 1 and these points are called as deployment points.

The scheme consist of three phases to complete key establishment key pre-distribution, shared key discovery and path-key establishment. In the *key pre-distribution phase*, the given area is divided into $t \times n$ cells, each of equal size. The subset of the original key pool is assigned to each cell such that each cell shares a |Sc|, for $0 < a < 1$ keys with its vertical and horizontal neighbors, and a $b \mid Sc \mid$, for $0 < b < 1$, keys with its diagonal neighbors, where $\mid Sc \mid$ is the key pool size for each grid, a and b can vary under the constraint $4a + 4b = 1$. Within a cell each node is assigned m random keys from same cells key pool to which node belongs.

In *shared key discovery phase*, each node broadcast its set of keys to let the neighbor find the common keys it may share with the particular node. If a common key exists, the neighboring node uses this key to secure the communication with the node. Intuitively, this phase creates a graph where every node is vertex, and every the secured communication link between two nodes is an edge between the corresponding vertices.

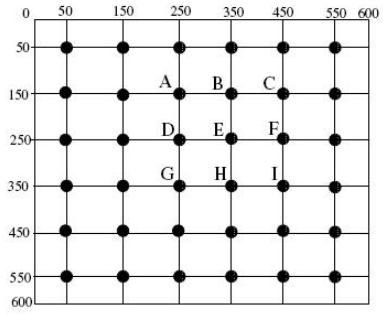

Fig. 1. Deployment point (each dot represents a deployment point).

In *path-key establishment phase*, the nodes which are neighbor to each other and do not have edge in the graph, search for the secured path so that they can transfer key using that secured path. For example, if nodes i and j do not have any key in common, then they will search for a path in graph whose two end vertices correspond to i and j. Let $i, v_1, v_2, \ldots, j$ be the path. Then to estabish a common secret key, i generates a key k and sends it to v_1 over the secure link, now v_1 send k to v_2 over secure link and this process will continue until k reaches j. Through experiment it was found that the maximum number of hops required to find such a secure path is 3.

III. Motivation for our Scheme

As mentioned earlier since the probability of finding a common key between any two nodes is < 1, there is always a chance of depletion of energy due to extra messages transferred due to the need to establish key for secure communication. If somehow we can restrict the number of messages, then node energy can be saved. Even if we restrict number of messages for key setup, there still some messages will flow for key setup between nodes having no shared key. So a simple possible approach is to increase the number of keys shared between neighboring groups. Therefore, we thought about a two pronged approach, namely, (i) improve the key sharing between neighboring groups, and at the same (ii) restrict the number of messages transfers for key establishment. Furthermore, our attempt also led to an increase in resilience against node capture as will be explained later in this paper.

IV. Key pre-distribution using deployment knowledge

We assumed the same deployment model as used in [1]. A random key pre-distribution is carried out based on deployment knowledge. The sensor nodes partitioned into $t \times n$ groups $G_{i,j}$, for $i = 1, \ldots, t$, and $j = 1, \ldots, n$. There is a global key pool of size |S|, and deployment points are centers of cells as shown illustrated by figure Fig 1.

A. Key pre-distribution scheme

The stated aim of our scheme is to allow sensor node to communicate with maximum number of their neighbor nodes securely. It consist of six phases: (i) key generation, (ii) key pre-distribution, (iii) shared key discovery, (iv) key establishment, (v) alternative key establishment, and (vi) revocation of keys. The third phase of our protcol is very much similar to the shared key discovery phase of [1]. The second phase is different in the way it is used to distribute key for groups.

1) Step 1: Key Generation phase.: This is an offline phase. Before deployment |S| keys are generated over a *field* F. These keys acts as a key pool in key distribution phase. The aim of generating a keys over a field F will be clear in key establishment phase.

2) Step 2: Key pre-distribution phase.: In this phase, initially the key pool S divided into $t \times n$ key sub pools $S_{i,j}$, $i = 1, \ldots, t$, $j = 1, \ldots, n$. Each sub-pool $S_{i,j}$ corresponds to the deployment group $G_{i,j}$. If two groups are deployed in adjacent locations they are referred to as neighbors. The goal of setting sub-key pools for each group is to allow adjacent groups share as much as keys possible, while groups located apart sharing keys as little as possible. The principle behind key pool sharing is discussed subsequently. After key pool has been set up, each node randomly selects m keys from its corresponding group, and store it in its memory.

3) Step 3: Shared key discovery phase.: In this phase, each pair of nodes tries to agree upon a common key for secure communication. Each node broadcast a message which includes all key ids stored in its memory. Every other node listens to this broadcast and try to figure out if there is any common key. On finding a match, the concerned node identifies that key for a secure communication between itself and the broadcaster node. If more than one common key if found, then both nodes select two keys out of them for securing the communication with double encryption. It is also possible that some nodes may not find any common key with some of their neighbor nodes in single hop. If it is the case then such nodes use the next phase to establish a common key between them.

4) Step 4: Key establishment phase.: Consider two nodes A and B which do not have any matched key. To establish keys each of the nodes generates a random number and its inverse over a field F. Suppose A generates r_1 and r_1^{-1} over F, and B generates r_2 and r_2^{-1} over F. Now A sends one of its keys k to B using following protocol.

- First A sends $(kr_1) \bmod F$,
- B receives $p_1 = kr_1$, and modifies it as $(p_1 r_2) \bmod F$ = $(kr_1 r_2) \bmod F$ and sends it to A,
- A receives $p_2 = (kr_1 r_2) \bmod F$ and modifies it as $(p_2 r_1^{-1}) \bmod F = (kr_1 r_2 r_1^{-1}) \bmod F = (kr_2) \bmod F$ and sends it to B.
- B receives $p_3 = (kr_1 r_2 r_1^{-1}) \bmod F$ and modifies it as $(p_3 r_2^{-1}) \bmod F = (kr_2 r_2^{-1}) \bmod F = k \bmod F = k$

now B and A can use k for further communication.

Note that k can be either generated by node A over field F or it can be chosen from set of stored keys. It depends upon whether any surplus key is available for communication or not. If no surplus key is available, generating a fresh key will be required.

5) Step 5: Additional key Establishment phase.: At the end of fourth phase, every node will have at least one common key with every one of its neighbors. Therefore, a common key can be used as encryption key, and one additional key can be generated and exchanged securely. Note this this operation will be need just once. Since two keys are available, all message exchanges are done with double encryption. When one of the key gets revoked due to expiry or being otherwise detected as compromised, the additional key establishment phase can be invoked again to re-establish one additional key. This additional key can be either be generated by node over field F, or it can be chosen from set of pre-stored keys.

6) Step 6: Key revocation and replacement phase.: Due to some security reasons such as detection of a node capture or expiry of keys, some key can be revoked. All the nodes must remove the revoked key from their memory. If any of the keys which is used in communication gets revoked it may lead follow two problems.

(i) Suppose one of the keys is revoked out of two being used in communication with particular node. In this case the additional key establishment phase (step 5) can be invoked to acquire an additional key between two nodes.

(ii) Suppose both the keys being used between a pair of nodes are revoked. In this case, first we return back to key establishment phase, and then invoke fifth phase for additinal key establishment.

B. Setting up key pool

Now we describe how keys can be assigned to each key pool $S_{i,j}$, $i = 1, \ldots, t$, $j = 1, \ldots, n$ such that they share more keys with their neighbor groups. Each $S_{i,j}$ share $a \mid Sc \mid$ keys with all its neighbors, where a is a pre-decided key overlap factor. The sharing of keys is organized in such a way that any group key pool can not receive keys from more than 3 other group key pools, at the time of key pool initialization

One initial idea was we came up with is illustrated by Fig 2. In this scheme the key pool from a cell located at the center shares the same keys with the key pools two other cells; one located diagonally below and the other located at a distance of the knight's walk from the diagonal cell. For example, as shown in Fig 2, the key pool of cell A shares same keys with the key pools belonging to cells E and F. Similarly C share same keys with E and H. The problem with this approach is that same two groups share keys more than once, if area is partitioned into grids of size more than 3×3.

We then refined the idea stated above and developed a scheme key sharing scheme as shown figure Fig 3. In this approach sharing of keys is performed in key pools in such a way that any key pool can not receive keys from more than 3 key pools, at the time of key pool initialization and same two pools never share its keys more than once. The algorithm for choosing keys for a key pool according to this new idea is as follows:

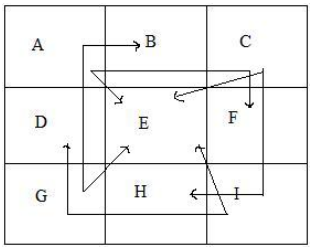

Fig. 2. Knight's tour based sharing in key pools

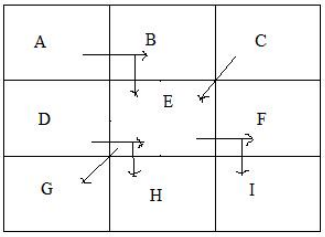

Fig. 3. Sharing of keys between neighboring key pools

1) For group $S_{1,1}$, a random set of $\mid Sc \mid$ keys chosen from the set S.
2) For the group $S_{i,1}$, where $i > 1$, $a \mid Sc \mid$ keys are taken from $S_{i-1,1}$ while another $a \mid Sc \mid$ keys are taken from the pool $S_{i-1,2}$. Finally, the rest of $w = (1 - 2a) \mid Sc \mid$ keys are chosen from S, and those chosen keys are removed from S.
3) For the remaining groups $S_{i,j}$, where $j > 1$ key pools is arranged in the following way. The group $S_{i+1,j}$, if it exists, gets $a \mid Sc \mid$ randomly selected keys are from the pool $S_{i,j-1}$, assign same chosen set to $S_{i+1,j}$ if it exists and

   if j equals 2 then another $a \mid Sc \mid$ keys from $S_{i+1,j-1}$ if it exists
   else another $a \mid Sc \mid$ keys from $S_{i-1,j+1}$ if it exists.

   The remaining $w = (1-3a)|Sc|$ keys are provided from S, and those w selected keys are removed from S.

The overlapping factor a can be variable, but in order to get good sharing we used the value $a = 0.25$.

V. Performance Evaluation

We present both analytical and simulation results in this section. In simulation results showed that only ratio of nodes able to find a common key across cells to total number of nodes tried to communicate across cells i.e. if any node belong to group $G_{i,j}$ then we find out the the probability of a node able to find a common key with nodes in its neighbors group.

A. Evaluation metrics

The proposed protocol is evaluated on the typical metrics concerning wireless sensor networks.

1) *Connectivity.* We define the same two types of connectivity as in [1]. Global connectivity defines the percentage of the nodes connected in large isolated network. The nodes out of range are not considered, as they won't be able to become part of network because of their range, not because of lack of communication key establishment. Our protocol was able to achieve 100 percent global connectivity. Local connectivity is defined as probability of any two neighbor nodes sharing at least one common key. In our scheme the number of keys that in memory of sensor nodes affects local connectivity, but not global connectivity.
2) *Communication overhead.* As probability of finding a common key with a neighbor is less than 1, there is always a possibility of sending extra messages. We define communication overhead as number of extra messages sent in order to establish key between nodes that don't have any shared key between them.
3) *Resilience against node capture.* We assume that an adversary can launch all type of attacks on nodes after they are deployed. We are interested in finding out number of communication link affected by capturing a node or x number of nodes.

B. System configuration

The simulation is carried in TOSSIM (Tiny OS simulator). The simulation parameters were as follows:

1) Size of key pool for each cell |Sc|=1800
2) The number of sensor nodes is 900
3) Deployment area is 300 $\times$ 300 m
4) Area is divided in to 3 $\times$ 3 cells each of size 100
5) Simulatd on TOSSIM with noise floor = -90db, standard deviation 6.0

C. Local connectivity

Local connectivity as defined in [1] refers to the probability, p_{local}, of two neighboring nodes having at least one common key. Let us define

(i) B is the event that two nodes share at least one key,
(ii) A is the event that nodes are neighbors.

Hence

$$p_{local} = Pr(B \mid A).$$

Let λ be ratio of key pool shared by two nodes. Then the number of keys shared is $\lambda \mid Sc \mid$, where λ varies from 0, a and 1.

Pr(two nodes have at least one key shared) = 1 - Pr(two nodes dont share a key) the probability defined in [1], and is given as

$$1 - \frac{\sum_{i=0}^{min(m,\lambda|Sc|)} \binom{\lambda \mid Sc \mid}{i} \binom{(1-\lambda) \mid Sc \mid}{m-i} \binom{\mid Sc \mid -i}{m}}{\binom{\mid Sc \mid}{m}^2}$$

Since we our overlapping factor is high compared to basic scheme proposed by Varshney et. al. [1], our protocol ensures

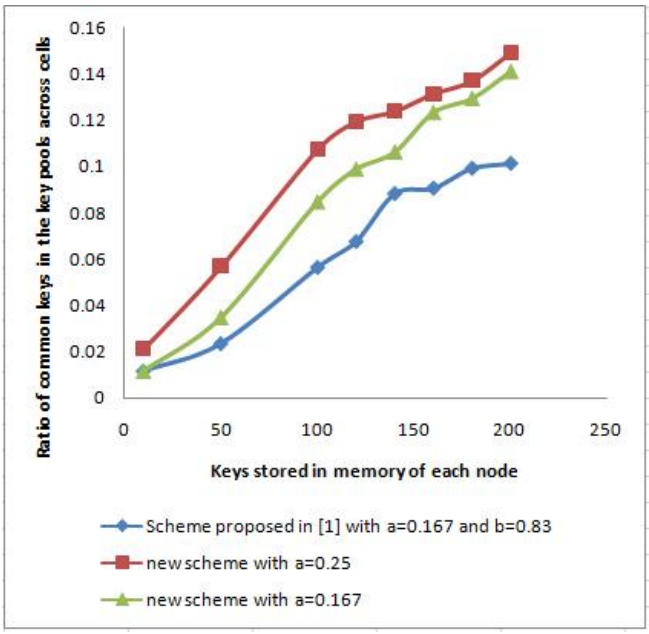

Fig. 4. Ratio of node pairs having common keys.

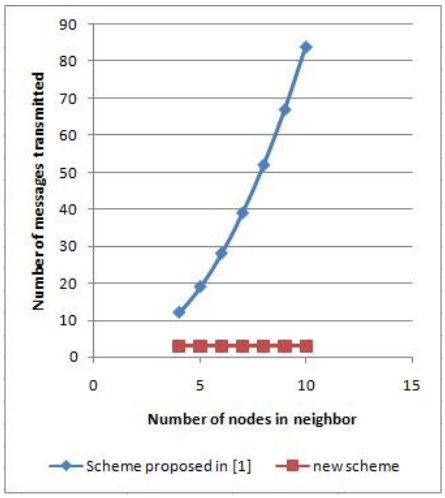

Fig. 5. Message overhead for establishing common keys.

a better local connectivity across the cells as shown in Fig. 4. In basic schemes we have taken a = 0.167 and b = 0.083. If m = 100. Our scheme shows 90% improvements over the basic scheme, similarly at m = 200, our scheme shows 47% improvement.

D. Communication overhead

The scheme has very little communication overhead, if no shared key can be found between two nodes in single hop. Our scheme uses maximum of 3 messages to set up a key between two nodes. As against this, assuming every node has 8 neighbors, the protocol proposed in [1] requires a maximum of 52 message for setting up a common key between each pair of nodes under the condition that those pair are at a distance of one hop. The comparison of message overhead in two scheme is provided in figure 5

E. Resilience against node capture

To find out the resilience against node capture, we need to find out the number of communication links that an adversary can capture through x number of compromised nodes. Since the location of the compromised nodes would affect the resilience metric. We assume that these x compromised nodes are randomly distributed over the network. Let K_1 and K_2 be a key used to secure a communication over a link between two nodes that are not compromised. When any node other than these nodes are compromised then the probability that K_1 and K_2 both will not be compromised at the same time is $1-\left(\frac{m}{|S|}\right)^2$,where m is number of keys carried by each node. Since, x nodes are compromised the probability that both K_1 and K_2 will not be compromised at same time is given by

$$\left(1-\left(\frac{m}{\mid S \mid}\right)^2\right)^x.$$

Since m is very small compared to $\mid S \mid$, it follows that the resilience factor of our protocol is very high.

VI. Conclusion And Future Work

We proposed new a scheme for secure communication based on one hop key exchange for wireless sensor networks. The sharing of the keys between groups led to improvements in energy consumption for key establishment, and resilience against node capture over the scheme presented in [1]. We have provided analytical proof of these improvements and backed up the result of analysis also through simulation results. Our future work concentrate on simulation study on the variable key sharing among the groups, and on how the local resilience of the key pool is effected by such variation in sharings.

References

[1] Pramod K. Varshney, Wenliang Du, Yungsiang S. Han and S. Chen, A Key Management Scheme for Wireless Sensor Networks Using Deployment Knowledge, IEEE infocom 2004.
[2] L. Eschenauer and V. D. Gligor, A key management scheme for distributed sensor networks, in Proceedings of 9th ACM conference of computer and communication security Washington, DC, USA, November 18-22-2002, pp 41-47.
[3] I. F. Akyildiz, W. Su, Y. Sankarasubramaniam, and E. Cayirci, A survey on sensor networks, IEEE Communication magazine, vol. 40, no. 8, pp. 102-114, August 2002.
[4] B. C. Neuman and T. Tso, Kerberos: An authentication service for computer networks. IEEE Communications, vol. 32, no. 9, pp. 33-38, September 1994.
[5] R. L. Rivest, A. Shamir and L. M. Adleman, A method for obataining digital signatures and public key cryptosystems, Communications of the ACM, vol. 21, no. 2, pp. 120-126, 1978.
[6] H. Chan, A. Perrig, and D. Song, "Random key Predistribution Schemes for Sensor Networks." IEEE Symposium on Security and Privacy, pp 197213, Berkeley, California, May 11-14 2003.
[7] Y. Xiao, V. K. Rayi, B. Sun, X. Du, F. Hu, M. Galloway, "A Survey of Key Management Schemes in Wireless Sensor Networks, Comput. Commun. 30 (11-12) (2007) 23142341.