# Resilient Key Establishment for Mobile Sensor Networks

Kevser Karaca and Albert Levi

**Abstract— Wireless Sensor Networks (WSN) are self-organizing and resource-constrained networks composed of battery-powered small devices called sensor nodes. WSNs are typically deployed in unattended areas. In order to secure WSNs, firstly cryptographic keys must be distributed in a secure and robust way. Key distribution problem is extensively studied for static WSNs, but has not been studied widely for mobile WSNs (MWSN) in the literature. In this paper we propose a key distribution mechanism for MWSNs in which both sensor nodes and the BS are mobile. In our scheme BS acts as a mobile key distribution center which distributes pairwise keys to sensor nodes in a single hop. Our performance evaluations show that the proposed scheme outperforms mobile versions of two existing schemes in the literature. Moreover, our scheme has perfect resilience against node capture attacks.**

**Keywords– mobile wireless sensor networks; security; key distribution; resilience**

## I. INTRODUCTION

Wireless Sensor Networks (WSNs), which consist of small battery-devices called sensor nodes have gained importance in the last decade for their widespread applications [1]. The sensor nodes of the network can sense and collect data, process the data they collect or send the data to a sink node, also called *Base Station*. The nodes in the network and the *Base Station* can be either static or mobile, depending on application and environmental conditions.

It is important to provide security mechanisms for WSNs like any other kind of network. The main prerequisite of fulfilling the security requirements of WSNs is to have a robust cryptographic key distribution and management. There are various approaches to key distribution problem in WSNs, such as matrix-based approach, polynomial-based approach, probabilistic approaches and location-based approaches. Good surveys about key distribution in WSNs can be found in [2, 3].

Most of the solutions proposed for key distribution problem in WSNs are designed explicitly for static WSNs. To the best of our knowledge, there are only a few proposals in the literature [4-6], which superficially mention suitability for Mobile WSNs (MWSN), but without an analysis. Moreover, specific characteristics of mobility must be taken into account while designing a scheme, which is missing in the existing ones. Thus, there is an important gap in the literature for key distribution schemes specifically designed for MWSNs.

In this paper, we propose a key distribution scheme for MWSNs by utilizing mobility to improve the overall system performance. We use Base Station (BS) as a mobile key distribution center, which securely provides the nodes pairwise keys of their neighbors as they meet with it. We analyzed the performance of the proposed scheme using simulations. Our simulations show that, in the steady-state, our scheme achieves up to 0.6 local connectivity (probability of sharing a key for two neighboring nodes) with relatively small amount of memory usage. Moreover, our scheme does not make any node disconnected from the secure network. More importantly, our scheme has perfect resilience against node capture attacks such that an adversary cannot compromise any additional links (i.e. the links other than the captured nodes') using the nodes he/she captured previously.

The rest of the paper is organized as follows. In Section II, related studies about key distribution are summarized. We introduce our scheme and explain the protocol in Section III. Section IV gives the performance evaluation of the proposed scheme and finally Section V concludes the paper.

## II. RELATED WORK

The most widely used approach for key distribution is probabilistic key predistribution, which is first introduced by Eschenauer and Gligor [7]. In this scheme, which is referred to as the *Basic Scheme*, there is a global *key pool* composed of random keys and their identifiers. The typical size of a key pool is 10,000 – 100,000 keys. Sensor nodes are predistributed a randomly selected set of keys chosen from the global key pool before the deployment. These keys form the *key chain* of the node. After deployment, nodes find out if they have shared keys with their neighbors in order to establish pairwise keys. This approach brings a tradeoff between connectivity and resilience against node capture attacks. Basic scheme inspired many researchers; there are several studies, like [8-11], that use similar probabilistic approaches. Local connectivity of the basic scheme is not affected by node mobility since the keys put into each node are chosen at random without any regard to their location. Thus, even if the location changes the probability of sharing a key does not change. We have proved this fact via simulation (not included in the paper due to space limitations). However, although it is not sensitive to mobility, basic scheme and its successors require high amount of memory to achieve acceptable level local connectivity.

This work was supported by the Scientific and Technological Research Council of Turkey (TÜBITAK) under grant 110E180.

Kevser Karaca is with Sabanci University, Istanbul, Turkey (e-mail: kevser@sabanciuniv.edu).

Albert Levi is with Sabanci University, Istanbul, Turkey (phone: +90 216 483-9563; fax: +90 216 483-9550; e-mail: levi@sabanciuniv.edu).

Blom's Scheme [12] is a well-known matrix-based solution for key distribution problem. It is a multipurpose deterministic key pre-distribution scheme. The scheme has $\lambda$-secure property, which means an adversary cannot compromise any links if it has captured at most $\lambda$ nodes; however, it can compromise all the links once it has captured $\lambda$+1nodes. There are other works [13-15] that adopts Blom's scheme to WSNs and try to improve security and/or scalability.

There are also some schemes that try making use location information of the nodes along with the key distribution approaches explained above. One of the best known examples of this method is Du et al.'s scheme [16] which uses a probabilistic approach. The idea is to divide the global key pool and the deployment area into zones and predistribute keys to nodes such that nodes which will be neighbors will get keys from the same key pool and will have a higher chance of having common key. There are many other works, like [17-19], which make use of location information to solve key distribution problem. Generally location-based schemes are highly sensitive to node mobility. As the nodes start moving in the environment, neighboring relationships change and most of the keys which are selected from zone-based key pools become useless. Thus, the local connectivity values drop in time. We have proved this fact via simulations. Our results for the scheme proposed in [16] show that with 100 keys in the key chains, local connectivity drops to 0.2 from 0.67 when the nodes are mobile.

As mentioned before, there is limited work in the literature for key distribution problem in MWSNs. In [4], the authors propose a group-based key predistribution scheme which is suitable for static sensor nodes and for nodes that move in a swarm fashion. The scheme proposed in [5] uses a zone-based node deployment. The scheme can only tolerate mildly-mobile networks and does not work for highly mobile scenarios. The scheme proposed in [6] makes use of assistant nodes which distribute keys to nodes. The scheme works for MWSNs as well, but the number of assistant nodes needed to achieve high connectivity is very high. All of these schemes [4-6] are mainly designed for static WSNs and their applicability for MWSNs is shown in a perfunctory way without formal analyses or simulations. Other than these schemes, Zhang et al. [20] proposed a pairwise key predistribution scheme based on random perturbation of polynomials. Their scheme is shown to be perfectly resilient to node capture attacks with very high probability such that collusion of compromised nodes reveals a key between non-compromised nodes only with a negligible probability. Moreover, this scheme is topology independent, thus it inherently accommodates node mobility, as in the basic scheme [7].

## III. Our Scheme

In this paper, we propose a perfectly resilient key distribution scheme for Mobile Wireless Sensor Networks (MWSNs). In our scheme, nodes and the Base Station (BS) are mobile. The main idea of our scheme is to have the BS operate as a key distribution center. In our scheme, prior to deployment, nodes are <u>not</u> preloaded with any keys. After they are deployed to the area, BS starts to move among the nodes and distribute pairwise keys to neighboring nodes it meets along the way.

In regards to the threat model, we have two basic assumptions. We assume that the BS is tamper-proof and cannot be captured by the attacker. However, sensor nodes can be captured by the attacker. The effect of these assumptions on the resilience will be examined in Section IV.C in detail.

In our scheme, we use Random Walk Mobility Model [21] for the movement of the nodes in order to have independently moving entities rather than group movement. In our version of Random Walk Mobility Model, a node randomly selects a direction between $[0, 2\pi]$, and a speed between [*speedmin*, *speedmax*]. The node moves in that direction for one minute and chooses a new direction and speed without waiting at that point, and continues its movement. If it meets the boundaries of the area, it bounces back.

For the movement of BS, we use a deterministic approach to ensure that BS scans the whole area and meets with possibly all the nodes. In our mobility model for BS, it starts moving from southwest corner of the area, goes to the opposite edge horizontally. After it gets very close to the boundary, it starts moving vertically for a very short distance, and then starts moving horizontally again. When it scans the whole area, it diagonally goes back to the southwest corner and starts its movement from there again.

The symbols and notations we use for our scheme are listed in Table I below.

TABLE I
List of Symbols Used in Our Scheme

| | |
|---|---|
| $n_i$ | A node with unique identification number *i,* node *i* |
| $K_{i-BS}$ | Pairwise key shared between node *i* and BS |
| $K_{ij}$ | Pairwise key shared between node *i* and node *j* |
| $E_{ij}(M)$ | Symmetric encryption of message *M* with pairwise key $K_{ij}$ |
| $D_{ij}(M)$ | Symmetric decryption of message *M* with pairwise key $K_{ij}$ |
| $L(n_j, n_k, n_l \ldots)$ | List of nodes; node *j,* node *k,* node *l,*… |
| $m$ | The maximum number of keys a node can have (key chain size) |

There are four major procedures of our scheme. These are initialization, key distribution, pairwise key establishment and update of the key chain. These procedures are explained below:

- *Initialization:* This procedure covers initial node configuration and node deployment to the area. Before deployment, each node $i$ is preloaded with a pairwise key $K_{i-BS}$ it shares with the BS. A node does not store any keys that it can use to communicate with other nodes; however it has a reserved memory space for $m$ keys to be used later. Then, the nodes are deployed to the area randomly using uniform distribution. After the nodes are deployed, they cannot communicate with each other until BS distributes pairwise keys to them. When a node meets with BS, key distribution phase begins.

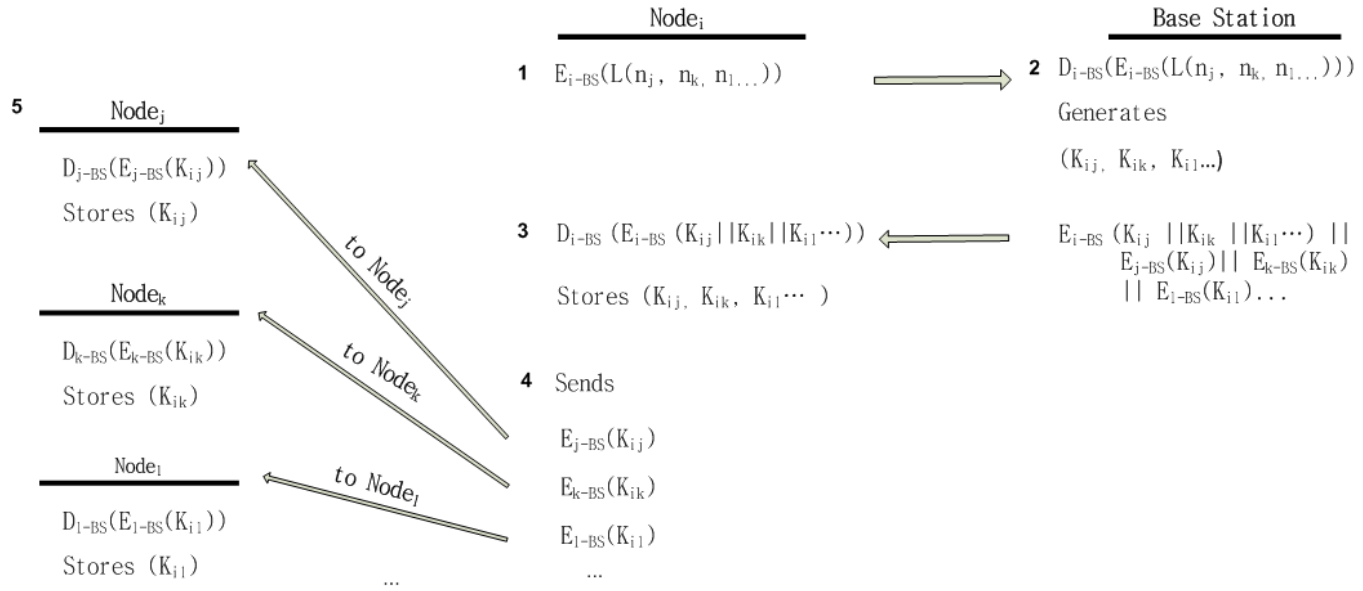


Fig. 1. Key distribution protocol between Base Station and nodes

- *Key Distribution:* When a node senses BS in its communication range, key distribution procedure starts. The flow of key distribution is depicted in Fig. 1. At Step 1 shown in Fig. 1, a node, *node$_i$* (also denoted as $n_i$), who wants to get keys from BS, prepares a list $L(n_j, n_k, n_l \dots)$ of its neighbors with whom it does not share a common key. It encrypts the list with $K_{i-BS}$, its unique pairwise key with BS, and sends the encrypted message to BS. At Step 2, BS decrypts the message it got from $n_i$. Using the pseudo random function PRF, BS generates pairwise keys $K_{ij}, K_{ik}, K_{il}, \dots$ to be used between $n_i$ and the nodes in the list. BS encrypts each created key, $K_{ix}$, with the corresponding key that it shares with $n_x$, $K_{x-BS}$. Then the BS concatenates the keys to be sent to $n_i$ and encrypts them using $K_{i-BS}$. Finally in Step 2, the BS sends all these encrypted packets to $n_i$. At Step 3, $n_i$ decrypts the received message using $K_{i-BS}$ and gets the keys it requested from BS. It adds these keys to its key chain using the update of key chain procedure explained below. Moreover, $n_i$ sends the encrypted keys $E_{j-BS}(K_{ij}), E_{k-BS}(K_{ik}), E_{l-BS}(K_{il}), \dots$ to their corresponding recipients $n_{ij}, n_k, n_l, \dots$ at Step 4. At Step 5, each node decrypts the encrypted key it received from $n_i$, obtains the pairwise key to be used with $n_i$ and adds the key to its key chain.
- *Pairwise key establishment:* When two neighboring nodes want to communicate with each other, they first exchange their node IDs. Using these IDs, they search their key chains to see if they have a pairwise key for each other. If they have a pairwise key, they can communicate with each other using that key.
- *Update of the Key Chain:* As mentioned earlier, in our scheme, each node has a fixed key chain size. Therefore, there needs to be an update mechanism to manage the use of this limited key chain. We employ a FIFO (first in first out) mechanism to update the key chain. In our scheme, when a node gets new keys from BS, it first checks whether it has enough space in its key chain or not. If it has enough space it adds the keys to its key chain. If it does not have enough space, then it selects the oldest key which is not in use at that moment and deletes it from the key chain. This way, it opens up space for the new keys and adds the new keys to the key chain. The keys are stored together with the corresponding neighboring node ID.

## IV. Performance Evaluation

The performance evaluation of the proposed scheme is done via simulations using different metrics and parameters. The detailed analyses are explained in the following subsections. The simulation code is developed using C#. The common parameters and system configuration for our scheme is as follows:

- The number of sensor nodes in the network is 10,000.
- The deployment area is 1,000m × 1,000m.
- Nodes are deployed with uniform random distribution to the simulation area.
- The wireless communication range for each node is 40m.
- The speed of the nodes is selected randomly between 5-15 meters/minute.

### *A. Local Connectivity Analysis*

Local connectivity is an important metric to show the performance of the key distribution schemes. It is defined as the probability of any two neighboring nodes sharing a common key. We simulate various cases to show how our scheme performs and how the local connectivity value changes over time. The cases and results are explained below.

#### *1) Local Connectivity for Different m Values*

In order to see how the number of keys each node has affects the local connectivity, we conduct simulations for different *m* values and calculate the local connectivity of the network versus time. The mobility models for nodes'

movement and Base Station's movement is kept as explained above. For the *m* values, we use *m*=100, *m*=150, *m*=200 and *m*=250. Speed of BS is kept constant at 400 meters/minute. The results of the simulation can be seen in Fig. 2.

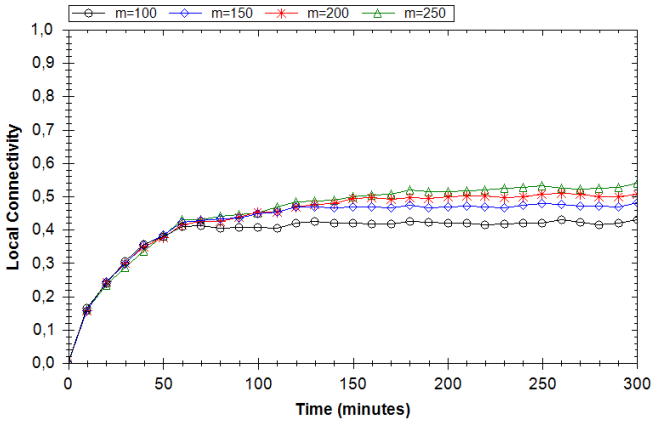


Fig.2. Local connectivity versus time for different *m* values where *BS speed*=400 meters/minute

It can be seen in Fig. 2 that, in the steady-state, local connectivity values are between 0.44 and 0.54 for different *m* values. As expected, larger *m* values cause better connectivity, but actually change in the value of *m* does not cause a big difference. The reason for that is the following: BS provides nodes with keys of their immediate neighbors at the time and nodes update their key chains with their newly acquired keys by the method described earlier. This means, they do not keep keys with their old neighbors. Also the keys, which are really useful for the nodes, are their freshest keys with their immediate neighbors, since local connectivity is about the connectivity between neighboring keys. Therefore, the old keys they keep in their memory has little help to them in terms of local connectivity since they become useless as the neighboring relationships keep changing and old neighbors move away from each other. So the decisive element in terms of local connectivity for this case is not the size of the key chain, but the number of neighbors a node can have around it. In our simulations, the maximum and average numbers of neighbors for a node is approximately 90 and 60, respectively.

The time at which local connectivity reaches the convergence value is around 60 minutes, which is also the approximate time at which BS completes one round of its movement in the area. By this time, BS has moved in the simulation area and covered the area completely, distributing pairwise keys to the nodes it encountered along the way. If all the nodes were static, local connectivity value would have been 1.0, since all nodes would get the pairwise keys for their neighbors and neighboring relationships would never change after that. However, the nodes are mobile and neighboring relationships keep changing in our case.

The local connectivity performance of our proposed scheme is superior that of [7] and [16] in the mobile setting. As mentioned in Section II, the steady state local connectivity of [7] and [16] are 0.1 and 0.2, respectively for *m*=100. However, our scheme's steady state performance is 0.44 for the same *m* value.

*2) Local Connectivity for Different BS Speeds*

In order to observe how the speed of BS affects the local connectivity of the network, we run simulations using different BS speeds. In this scenario, the size of the key chain *m* is kept constant at *m*=100. BS speed, on the other hand, has three different values; *BS speed*=200 meters/minute, *BS speed*=400 meters/minute and *BS speed*=600 meters/minute. Fig. 3 shows local connectivity versus time for these *BS speeds.*

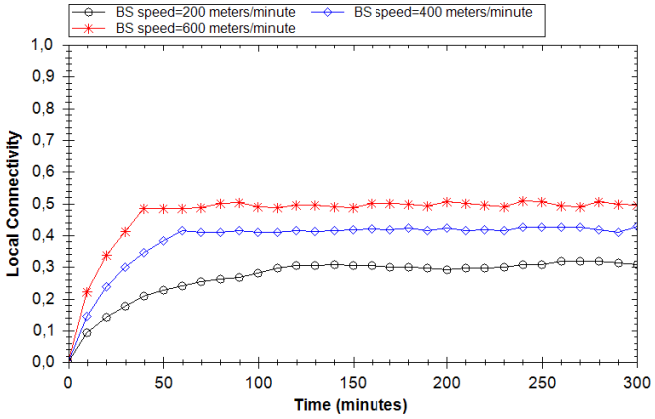


Fig..3. Local connectivity versus time for different BS speeds where m=100

Fig. 3 shows that the speed of the BS affects local connectivity in two ways. Firstly, the convergence time for local connectivity decreases as BS speed increases. The reason for this difference is that when BS moves faster, it can cover the whole area and connect the neighboring nodes with each other more quickly. As it was explained in the previous subsection, the time at which local connectivity reaches its convergence value is the time BS completes its one round of scanning in the area. With faster speeds, one round is completed faster; thus, the convergence occurs earlier as compared to the slower speeds. Secondly, local connectivity value increases when BS moves faster. When BS moves faster, the expected time it meets a node again and updates its key chain is shorter as compared to the case of slower BS. Therefore, it can update the nodes with their neighboring nodes' keys more frequently and keep the connectivity higher than the slower cases.

*3) Local Connectivity using Multiple Base Stations*

In this section we discuss how the local connectivity is affected when multiple Base Stations are employed. In this scenario we opt to use two BSs moving in the area at the same time. BSs move in the area deterministically and cover the whole area. However, one BS moves in the lower half; while the other BS moves in the upper half of the area. Both BSs distribute keys to the nodes as mentioned earlier. We conduct simulations using two BSs and compare the results to the case of single BS. The speed of BSs is kept constant at 400 meters/minute. The size of *m* is also kept constant at *m*=100.The results are shown in Fig. 4.

Fig. 4 shows that local connectivity is higher when there are two BSs in the environment. This is because sensor nodes can meet with one of the BSs more frequently and get new keys.

Here, it is worth to mention that using two BSs does not double the performance; as can be seen from Fig. 4, the performance increases approximately 50%.

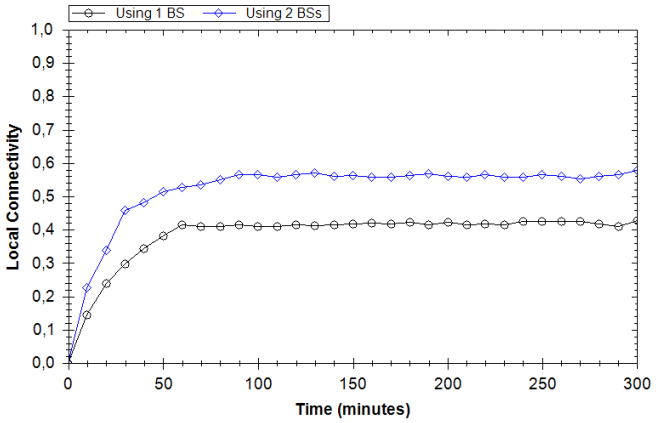


Fig.4. Local connectivity versus time using two BSs where m=100 and BS speed=400 meters/minute

*B. Global Connectivity Analysis*

Global connectivity is another important performance metric for the key distribution schemes. Let $G$ be a key sharing graph with nodes as its vertices and the secure links between nodes (i.e. the links between nodes which share a key) as its edges. Global connectivity is defined in [16] as the ratio of the size of the largest component in $G$ to the size of the whole network. It is important for a network to have a global connectivity, because higher the global connectivity means fewer nodes to be disconnected from the secure network and nodes can communicate with each other even if they have to use a few hops. We conduct simulations for various cases to see how our scheme performs in terms of global connectivity.

*1) Global Connectivity for Different m Values*

We analyze the effect of key chain size, $m$, on global connectivity. For this purpose, we use different values for $m$, namely 100, 150, 200 and 250 and calculate global connectivity versus time. BS speed is kept constant at 400 meters/minutes. The results are shown in Fig. 5.

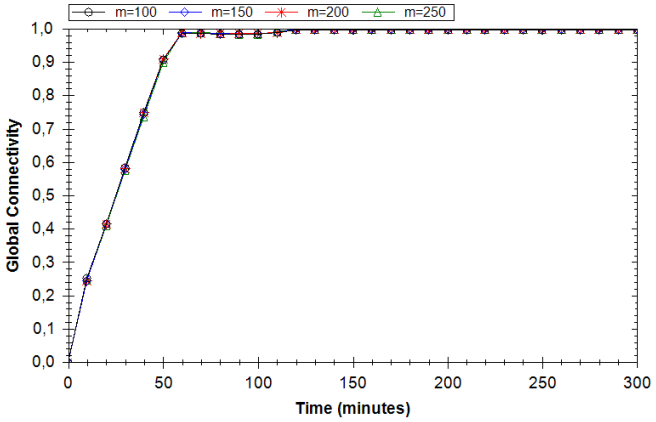


Fig.5. Global Connectivity versus time for different m values where BS speed=400 meters/minute

It can be seen in Fig. 5 that global connectivity reaches its convergence value, which is close to 1.0, at *time*=60. This is the time BS completes its one round of movement in the area. It can also be seen that there is not a significant difference between different $m$ values in terms of global connectivity. This shows that even a key chain of size 100 is enough to achieve a global connectivity which is fairly close to 1.0.

*2) Global Connectivity for Different BS Speeds*

In this scenario, the impact of BS speed on global connectivity is analyzed. The BS speeds we use are 200 meters/minute, 400 meters/minute and 600 meters/minute. The value of $m$ is kept constant at 100. Fig. 6 shows the results. As BS speed increases, the time needed for global connectivity value to reach its convergence value decreases. For *BS speed*=600, this value is reached at *time*=40, for *BS speed*=400, the convergence value is reached at *time*=60 and for *BS speed*=200 it is reached at around *time*=110. It can also be seen that for all three cases, after the area is scanned by the BS entirely, global connectivity value slowly increases and eventually becomes very close to 1.0.

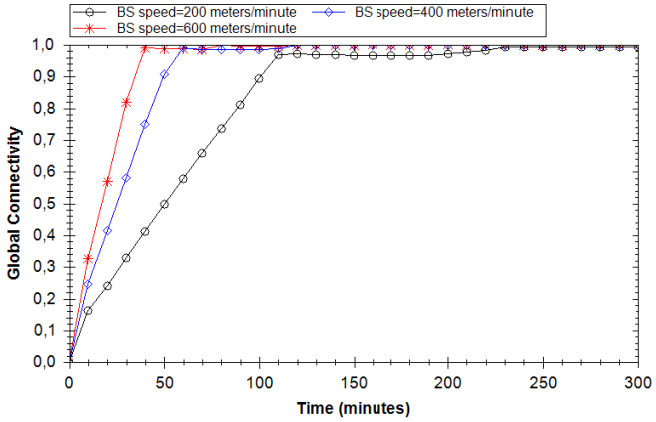


Fig. 6. Global Connectivity versus time for different BS speeds where m=100

*3) Global Connectivity using Multiple Base Stations*

In this scenario, two BSs are used in the setting explained in Section IV.A.3). In Fig. 7, the global connectivity performance in the case of using two BSs is compared to the single BS case. The speed of BS for both cases is kept constant at 400 meters/minutes and the key chain size of the nodes is set to 100 keys.

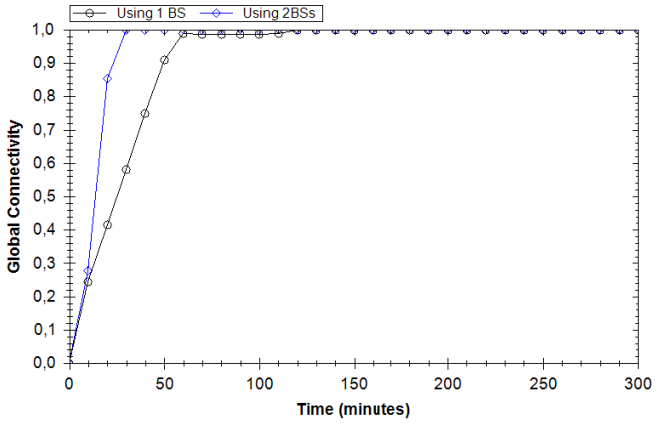


Fig. 7. Global Connectivity versus time using two BSs where m=100 and BS speed=400 meters/minute

It can be seen in Fig. 7 that using two BSs doubles the speed of convergence such that when there are two BSs, global connectivity reaches ~1.0 at *time*=30. On the other hand, when there is only one BS, global connectivity reaches ~1.0 at around *time*=60.

### C. Resilience of the Proposed Scheme

It is possible that an attacker can capture some of the nodes in the network. Since sensor nodes are generally not tamper-proof, an attacker can get access to the key chains of the captured nodes. This way, if the attacker puts the nodes to the network again, it can decrypt the messages sent to and from the captured nodes. Moreover, in most of the key distribution schemes, like [7, 8, 13, 15, 16, 17], it is possible that an attacker can even compromise the links between non-captured nodes since the keys used among these nodes may exist in the key chains of the captured nodes. Resilience is defined in [16] as a measure showing how much extra links can get compromised by the attacker other than the links of the actually captured nodes. In other words, when calculating resilience, the links of the nodes that are captured by the attacker are not counted; instead, the damage that this node capture can bring to other healthy nodes and links is calculated.

In our scheme, a node only keeps its secret key with BS and pairwise keys it has with its neighbors. These pairwise keys are generated and distributed by the BS. A node does not keep pairwise keys of any other node pairs. Thus, even if an attacker captures a number of nodes and gets access to the keys carried by those nodes, it can only compromise the links of the already captured nodes. It cannot compromise any additional links with the help of those keys. This means that our scheme has perfect resilience against node capture.

## V. Conclusions

In this paper, we propose a key distribution scheme which is designed for mobile Wireless Sensor Networks. In our scheme we have a mobile Base Station which works as a key distribution center while moving in the environment. The performance analyses show that our scheme achieves a steady-state local connectivity value higher than the mobile versions of the Basic Scheme [7] and Du et al.'s Scheme [16]. Our scheme's performance can be further improved by using multiple BSs or increasing the speed of the BS. Moreover, our scheme provides perfect resiliency such that an adversary cannot compromise any extra links using the captured nodes.

## References


[1] I. F. Akyildiz, W. Su, Y. Sankarasubramaniam, E. Cayirci, "Wireless sensor networks: a survey". *Computer Networks*, 38(4):393–422, 2002

[2] J. Zhang, V. Varadharajan, "Wireless sensor network key management survey and taxonomy". *Journal of Network and Computer Applications*, vol. 33, no.2, pp. 63-75,March 2010.

[3] Y. Zhou, Y. Fang, Y. Zhang, "Securing Wireless Sensor Networks: A Survey". *IEEE Communications Surveys & Tutorials*, vol. 10, no. 3, pp. 6-28, 2008.

[4] L. Zhou, J. Ni, C.V. Ravishankar, "Efficient key establishment for group-based wireless sensor deployments". In *Proceedings of the 4th ACM Workshop on Wireless Security* (Cologne, Germany, September 02 - 02, 2005). WiSe '05. ACM, New York, NY, 1-10

[5] A. Ünlü, A. Levi, "Two-tier, scalable and highly resilient key predistribution scheme for location-aware wireless sensor network deployments". *Mob. Netw. Appl.* 15, 4 (Aug. 2010), 517-529.

[6] Q. Dong, D. Liu, "Using auxiliary sensors for pair-wise key establishment in WSN". In Proceedings of the 6th international IFIP-TC6 Conference on Ad Hoc and Sensor Networks, Wireless Networks, Next Generation internet (Atlanta, GA, USA, May 14 - 18, 2007).

[7] L. Eschenauer, V. Gligor, "A key-management scheme for distributed sensor networks". In *Proceedings of the Ninth ACM Conference on Computer and Communications Security (CCS'02), ACM*, New York, NY, USA, 2002, pp. 41–47.

[8] H. Chan, A. Perrig, D. Song, "Random key pre-distribution schemes for sensor networks". In *Proceedings of the 2003 IEEE Symposium on Security and Privacy (SP'03),* IEEE Computer Society, Washington, DC, USA, 2003, pp. 197–213.

[9] S. Hussain, M. S. Rahman, L. T. Yang, "Key predistribution scheme using keyed-hash chain and multipath key reinforcement for wireless sensor networks", *2009 IEEE International Conference on Pervasive Computing and Communications (PerCom 2009)*.

[10] T. Shan, C. Liu, "Enhancing the key pre-distribution scheme on wireless sensor networks", *IEEE Asia-Pacific Conference on Services Computing*, IEEE Computer Society, Los Alamitos, CA, USA, 2008, pp. 1127–1131.

[11] C.-F. Law, K.-S. Hung, Y.-K. Kwok, "A novel key redistribution scheme for wireless sensor networks", *IEEE International Conference on Communications (ICC'07),* IEEE Computer Society, Washington, DC, USA, 2007, pp. 3437–3442.

[12] R. Blom, "An optimal class of symmetric key generation systems", in *Proceedings of the EUROCRYPT 84*, Springer, New York, NY, USA, 1985, pp. 335–338.

[13] W. Du, J. Deng, Y. Han, P. Varshney, J. Katz, A. Khalili, "A pairwise key pre-distribution scheme for wireless sensor networks", *Proceedings of the 10th ACM conference on Computer and communications security (CCS'03),* ACM, New York, NY, USA, 2003, pp. 42–51

[14] J. Lee, D. Stinson, "Deterministic key pre-distribution schemes for distributed sensor networks", in *Proceedings of SAC'2004*, LNCS 3357, Springer, Berlin/Heidelberg, 2005, pp. 294–307.

[15] H. Chien, R.-C. Chen, A. Shen, "Efficient key pre-distribution for sensor nodes with strong connectivity and low storage space", in *Proceedings of the 22nd International Conference on Advanced Information Networking and Applications (AINA'08),* IEEE Computer Society, Washington, DC, USA, 2008, pp. 327–333.

[16] W. Du, J. Deng, Y. Han, S. Chen, P. Varshney, "A key management scheme for wireless sensor networks using deployment knowledge". in *Proceedings of the 23rd Annual Joint Conference of the IEEE Computer and Communications Societies (INFOCOM'04),* IEEE Computer Society, Los Alamitos, CA, USA, 2004, pp. 586–597.

[17] D. Liu, P. Ning, "Location-based pairwise key establishments for static sensor networks", in *Proceedings of the First ACM Workshop on Security of Ad Hoc and Sensor Networks (SASN'03),* ACM, New York, NY, USA, 2003, pp. 72–82.

[18] Z. Yu, Y. Guan, "A key management scheme using deployment knowledge for wireless sensor networks", *IEEE Transactions on Parallel Distribution and Systems*, vol. 19, no. 10, pp. 1411–1425, 2008.

[19] D. Huang, M. Mehta, D. Medhi, L. Harn, "Location-aware key management scheme for wireless sensor networks". *2nd ACM workshop on Security of Ad Hoc and Sensor Networks,* 2004.

[20] W. Zhang, M. Tran, S. Zhu, G. Cao, "A random perturbation-based scheme for pairwise key establishment in sensor networks", in *Proceedings of the 8th ACM international symposium on Mobile ad hoc networking and computing (MobiHoc '07)*, ACM, New York, NY, USA, 2007, pp. 90-99.

[21] T. Camp, J., Boleng, V., Davies, "A survey of mobility models for ad hoc network research". *Wireless Communications & Mobile Computing (WCMC),* vol. 2, no. 5, pp. 483-502, 2002.