Babel: Using a Common Bridge Node to Deliver Multiple Keys in Wireless Sensor Networks

Jing Deng
Department of Computer Science
University of New Orleans
New Orleans, LA 70148, USA
jing@cs.uno.edu

Yunghsiang S. Han
Graduate Institute of Communication Engineering
National Taipei University
Sanhsia, Taipei, 237 Taiwan
yshan@mail.ntpu.edu.tw

***Abstract*— In Wireless Sensor Networks (WSNs), symmetric key schemes may be used to provide security. Recently, a class of random key pre-distribution techniques have been proposed and investigated. Such techniques only guarantee to establish keys for some pairs of physically connected sensors. In this work, we address the issue of delivering secret link keys to each of the source's neighbors in wireless sensor networks. We propose a scheme called Babel that finds a common bridge node to deliver one key to each of the to-be-connected neighbors. The novelty of our scheme is to deliver multiple keys through a common bridge node and regular paths instead of multi-hop secure paths. Since the delivered keys are only disclosed to one node, the common bridge node, key compromise probability of the Babel scheme is significantly lower compared to other delivery techniques.**

I. Introduction

Wireless Sensor Networks (WSNs) have been the focus of active research over the past few years. Some researchers investigated efficient data delivery, energy conservation, and other related issues [1]. Network security has also been a research focus mainly due to the fragility of the sensor nodes against node capture and modification.

In order to provide security for communication and data collection in WSNs, security keys need to be delivered and used. Due to the lack of trusted servers nearby (for public/private key schemes), secret key schemes may be more viable to protect such local communications. The question is how to deliver such secret keys to the communicating nodes.

In 2002, Eschenauer and Gligor proposed a random key pre-distribution technique allowing sensors to preload a subset of keys (key ring) from a large key pool. It has been shown that, when the size of the key ring and the size of the key pool are chosen carefully, the neighboring sensors will have a relatively large probability of sharing a common key. With such a shared key, the neighboring nodes can establish secret link key securely [2]. The scheme has been extended by researchers in several research groups [3]–[5].

It has been shown that there are still some local links that are disconnected on the security plane. The lack of such secure links can be thought as intentional. This is because an extremely high local connectivity (on the security plane) would mean higher vulnerability and lower network resilience, as shown in [2], [4].

In this work, we focus on the issue of delivering secret link keys from a source to multiple neighbors. Random key pre-distribution techniques may only connect some neighbors to the source with common keys. Other neighbors are still disconnected from the source on the security plane. Eschenauer and Gligor suggested to use the secure multi-hop path scheme to deliver such secrets [2]. In their scheme, the secret is disclosed to all the nodes on the path. If any of these nodes is compromised, the secret is disclosed to the adversary. Instead, we propose a new technique called Babel that tries to find a common bridge node to deliver secret link keys to these neighbors. The novelty of our scheme is to deliver multiple keys with the use of a common bridge node and regular paths instead of multi-hop secure paths. Since the delivered keys are only disclosed to one node, the common bridge node, key compromise probability of the Babel scheme is significantly lower compared to other delivery techniques.

We expect our scheme to be implemented in the networks where the source node needs to communicate with all its physical neighbors securely. Disclosure of any information sent from the source poses a security threat. An example is the cluster-based wireless sensor networks where the cluster head needs to communicate with the sensors in its neighborhood.

Besides the related work discussed above [2], [4], [5], Chan et al. presented a technique to establish secure link keys for two neighboring nodes if they do not share enough common keys [3]. These two neighbors first identify all secure paths between themselves. Then one node generates a set of random keys (of the same size) for all the paths and send one key to the destination through each of the paths. After the destination receives all the numbers, it exclusive-ORs all of them to obtain the secret link key. Li et al. proposed to use k intermediate nodes between two sensors to establish a link key [6]. Two methods were presented to identify such k nodes, which should be securely-connected to both nodes. The maximum distance separable (MDS) codes are used to establish link keys between neighbors without common keys through multiple paths [7], [8]. Traynor et al. proposed to use a few more powerful sensors to achieve key establishment [9]. Since these sensors are expected to have tamper-resistant hardware, the keys on these sensors are considered safe. Miller and Vaidya proposed to use Bloom filter and Merkel trees to

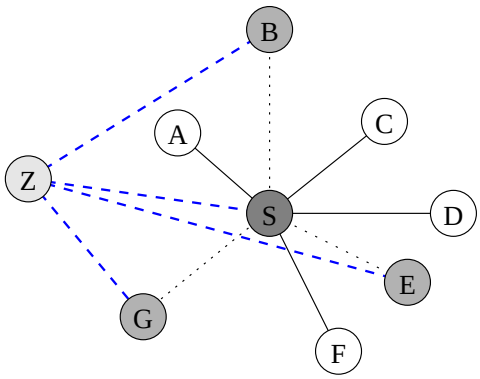

Fig. 1. Illustration of the security connectivity around node S. Solid lines represent security and physical connectivity. Dotted lines connect physical neighbors but they don't share any common key. Node Z is a node that is relatively far away from the neighborhood but it shares a key with all nodes connected with it through dashed lines.

distribute key information so that the chance of neighboring nodes establishing a key can be close to 1 [10].

II. Problem Formulation

Figure 1 illustrates the problem of delivering multiple secret link keys to neighboring nodes. Node S has several physical neighbors, which include nodes A, B, C, D, E, F, and G, clockwise. The solid lines in Fig. 1 represent security and physical connectivity. The dotted lines represent physical connectivity only. Therefore, links S-B, S-E, and S-G are disconnected on the security plane. Nodes B, E, and G are termed "to-be-connected neighbors", $\mathcal{N}_{tbc}$.

In order to secure the data transmission between node S and all its neighbors with symmetric encryption/decryption technique, we need to establish a secret link key for each of the physical links connecting to node S. In Fig. 1, the secret link key between node S and nodes A, C, D, and F can be set up easily with the shared keys. However, it is more difficult for node S to establish secret link keys with nodes B, E, and G.

Problem Statement: (Delivery of Multiple Secret Link Keys)

When random key pre-distribution techniques are employed in WSNs, it is possible that sensor nodes do not share keys with several neighboring nodes. In order to establish secure communication for these links, secret link keys need to be delivered to these neighbors. How should these secret link keys be delivered so that the chance of secret disclosure is as small as possible?

III. The Babel Scheme

In this work, we propose the Babel scheme to deliver multiple secret link keys toward the to-be-connected neighbors. Our main idea is to find one node that shares keys with the source node and each of its to-be-connected neighbors. The node is termed common bridge node. The common bridge node will be the only node other than the source and the receiving nodes knowing the secrets. All nodes in between only serve as passive routers without knowing the secrets that they help to forward. Such an approach lowers the key disclosure probability. We will discuss the chance of finding such a common bridge node and how to deliver multiple secret link keys securely when it cannot be found in Section III-B.

A. The Babel Scheme

We introduce the Babel scheme through the example shown in Fig. 1. Denote the set of keys stored on node i as K_i.

Node S collects the Message Authentication Codes (MACs) of a challenge message based on each of the keys in K_S, K_B, K_E, and K_G.[1] These information are control-flooded in the network. Each overhearing node compares the MACs of the message based on its carried keys and those on the overheard information. Only those nodes sharing a key with the source and each of the to-be-connected neighbors will respond. As an example, we assume that node Z satisfies the above condition, i.e., node Z shares at least one key with each of these nodes (nodes S, B, E, and G). The shared keys could be different or the same. It will respond and volunteer to serve as the common bridge by replying to node S. The reply message includes its response to each of the challenges with the shared keys.

Every node not satisfying the shared key criteria forwards the message by attaching its ID to the end of the message (for path record). Note that node Z may be physically multi-hop away from the source node. Controlled flooding is used, i.e., the message may only travel up to a certain number of hops, Time-To-Live (TTL). The message is discarded when it has been forwarded TTL times.

When the reply from node Z arrives at node S, node S sends them to the to-be-connected neighbors. Each of the to-be-connected neighbors validates the responses and ensures that node Z does share a key with itself. After each node verifies node Z's claim of sharing a key, node S sends the secret link keys (for nodes B, E, and G) to node Z. Upon receiving the message, Node Z encrypts the secret link keys with the shared keys with nodes B, E, and G, respectively. For example, the secret link key between nodes S and B will be encrypted at node Z with the common key between nodes Z and B. Such encrypted secret link keys are then returned to node S. Node S forwards the encrypted secret link keys to the to-be-connected neighbors, which decrypt the secret link keys.

Note that all message transmissions between nodes S and Z, except the first broadcast message, are encrypted with the shared key between these two. This will protect the transmitted information from being disclosed to a third party. Even though node Z can send all information to nodes B, E, and G directly, transmission through node S is preferred in order to protect the scheme from nodes, especially the non-common-bridge-nodes, faking common keys. The transmission overhead is slightly higher but the scheme is more secure [7], [8].

[1]We use the challenge-response technique to verify the common keys between the common bridge node and the source and the to-be-connected neighbors. With the use of this technique, key and key index disclosure are unnecessary.

The pseudo-code of the Babel scheme is shown through Algorithms 1, 2, and 3. In particular, Algorithm 1 finds a common bridge node. Algorithm 2 verifies the responses from the common bridge node. Algorithm 3 is then used to send secret link keys through the common bridge node.

We use the following notations in our pseudo-codes:

- Time-To-Live (TTL): a predefined number of hops for the request message to travel
- λ: the number of keys carried by each node
- S: source node
- $\mathcal{N}_{tbc}$: set of to-be-connected neighbors of S
- $K_{i,t}$: keys on node i, $i \in \{S\} \cup \mathcal{N}_{tbc}$, $1 \leq t \leq \lambda$
- t_i: the index of the shared key between the common bridge node and node i, $i \in \{S\} \cup \mathcal{N}_{tbc}$
- $LK_{S,i}$: secret link key between nodes S and i, $i \in \mathcal{N}_{tbc}$

Algorithm 1 Pseudo-code to Find Common Bridge Node

S obtains $MAC(i,t)$ based on $K_{i,t}$ for each (i,t), $i \in \{S\} \cup \mathcal{N}_{tbc}$, $1 \leq t \leq \lambda$
S broadcasts a msg with MACs and TTL
for each z receiving the msg **do**
    if $\forall i \in \{S\} \cup \mathcal{N}_{tbc}, \exists t_i,$ such that $MAC(z,t_i) = MAC(i,t_i)$ **then**
                ▷ A common bridge node is found
        z prepares responses to $MAC(i,t_i)$, $i \in \{S\} \cup \mathcal{N}_{tbc}$
        z encrypts responses with K_{z,t_S}
        Msg and $MAC(S,t_S)$ are sent from z to S
    else
        TTL $\leftarrow$ TTL-1
        if $TTL > 0$ **then**
            z adds its ID to the end of msg
            z forwards the msg
        else
            z discards the msg
        end if
    end if
end for

Algorithm 2 Pseudo-code to Verify Common Bridge Node

S verifies response from z on claimed common key K_{z,t_S}
if response is valid **then**
    S decrypts msg with K_{S,t_S}
    S forwards responses from z to each $i \in \mathcal{N}_{tbc}$
    for each $i \in \mathcal{N}_{tbc}$ **do**
        i verifies response from z on claimed common key K_{z,t_i}
        if response is valid **then**
            i sends confirmation to S
        end if
    end for
    if S receives confirmation for each $i \in \mathcal{N}_{tbc}$ **then**
        S accepts z as the common bridge node
    else
        S rejects z as the common bridge node
    end if
else
    S discards msg
end if

Algorithm 3 Pseudo-code to Send Secret Link Keys

S encrypts $LK_{S,i}$, $i \in \mathcal{N}_{tbc}$, with K_{S,t_S}
S sends encrypted msg to z
z decrypts msg with K_{z,t_S}, recovering $LK_{S,i}$, $i \in \mathcal{N}_{tbc}$
$\forall i \in \mathcal{N}_{tbc}$, z encrypts $LK_{S,i}$ with K_{z,t_i}
z encrypts encrypted link keys with K_{z,t_S}
z sends msg to S
S receives msg from z, decrypts msg with K_{S,t_S}
S sends encrypted link keys to $i \in \mathcal{N}_{tbc}$
$i \in \mathcal{N}_{tbc}$ decrypts link key with K_{i,t_i}

B. Discussions

Due to the page limit, we provide the following brief discussions of the Babel scheme in this subsection.

Note that a node lying about its keys either cannot decrypt the message from node S or its responses for the challenges from the to-be-connected neighbors fail.

It is possible that such a node Z cannot be found to share a key with each of these nodes. Then multiple nodes may be used to deliver secret link keys for all these nodes. For instance, node Z_1 may be used to deliver secret link keys toward nodes B and E. Node Z_2 may be used to establish a secret link key toward node G. When each common bridge node is only required to connect the source node and one to-be-connected neighbor, the scheme is similar to the scheme in [6] with $k = 1$.

Multiple bridge nodes may be used to deliver partial or encoded link key information from node S to nodes B, E, and G. In fact, assuming a large number of sensors in the network, we can find many nodes similar to node Z. Assuming that we find ℓ such nodes, nodes Z_1, Z_2, $\cdots$ and Z_ℓ. MDS scheme given in [8] may be used to encode the link keys. When ℓ is arbitrarily large, the compromise probability of the link keys can be arbitrarily low.

IV. Analysis

A. Probability of Finding a Common Bridge Node

We will analyze the chance of finding a common bridge node based on an independent p_{local}, sharing key probability. A more realistic analysis should consider the size of the key pool and the number of carried keys, λ, but we leave that to our future work. Note that, when the number of carried keys is much larger than the number of to-be-connected neighbors, these two models should be close to each other.

Assume there are N nodes in the sensor network. The probability of two nodes sharing at least a common key is p_{local}. Suppose there are m to-be-connected neighbors.

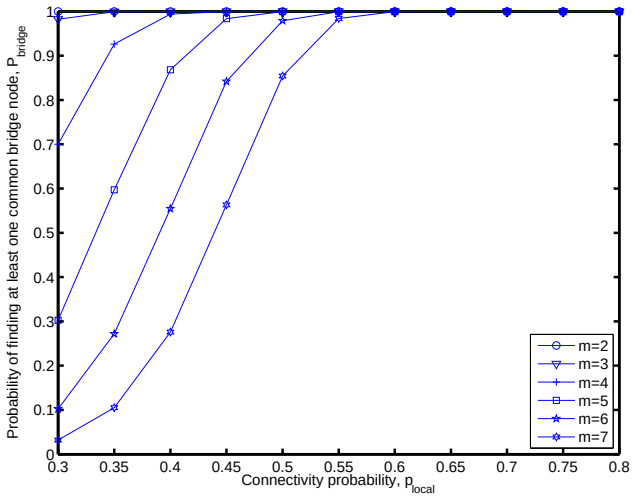

Fig. 2. The probability of finding a common bridge node. These are numerical results based on (1) when $N = 500$.

The chance of finding at least one sensor in the network that can help to deliver the secret link keys is:

$$P_{bridge} = 1 - [1 - (p_{local})^{m+1}]^{N-m-1} \ , \tag{1}$$

where $1-(p_{local})^{m+1}$ represents the probability that one node does not share a common key with the source and each of its m neighbors and we have assumed that the events of each pair of nodes sharing a common key are mutually independent.

In Fig. 2, we show the numerical results of the probability of finding at least a common bridge node in various network scenarios. The results of different number of to-be-connected neighbors, m, are presented in Fig. 2. As m increases, the probability of finding at least one common bridge node lowers. But this probability can be seen to be close to one when $p_{local} > 0.55$.

B. Security Analysis

In our security analysis, we measure the probability of any secret link key being compromised or disclosed. Since there are altogether m keys to be delivered and established, anytime one or all of these secret link keys are compromised, we treat the secret link key delivery compromised. We argue that such a security analysis provides an accurate guideline for several communication structure such as cluster-based WSNs. Since such secret link keys will be used for intra-cluster communication that should be protected from all outsiders, the disclosure of any secret link key (hence communication) will lead to security compromise in the cluster.

Our security analysis is based on the assumption that a ratio of $0 \leq x_c < 1$ sensors are compromised by the adversary. We assume that adversaries compromise sensor nodes randomly. When a sensor is compromised, all information attached to the sensor is disclosed.

Since our scheme only uses one common bridge sensor to deliver link keys to the neighbors, the chance of such keys being compromised is simply x_c plus $p_e(x_c)$, where $p_e(x_c)$ is the probability that any of the common keys shared by node Z and node S, or any of node Z and any of the to-be-connected neighbors of node S, is broken because other nodes in the network are compromised. That is,

$$P_c^{(Babel)} = x_c + p_e(x_c) \ . \tag{2}$$

Note that $p_e(x_c)$ depends on the random key pre-distribution scheme employed in the network. If we consider the multiple-space scheme proposed in [4], $p_e(x_c)$ is quite small when x_c is less than a threshold.

We shall investigate the compromise probability of the secure multi-hop path scheme [2]. In this scheme, the source finds a secure multi-hop path toward each to-be-connected neighbor. Many bridge nodes are employed in this scheme, increasing its vulnerability. Let the chance of having j, $j \geq 1$, nodes serving as bridge nodes be q_j, $\sum_{j=1}^{\infty} q_j = 1$.

When there are altogether j bridge nodes, the chance of any link key being compromised becomes

$$P_c^{(Multi)}|_j \geq 1 - (1 - x_c)^j + p_e(x_c) \ , \tag{3}$$

where inequality is because that the secret link keys are encrypted/decrypted by $j \geq 1$ sensors with different shared keys and we have used $p_e(x_c)$ as the lower bound of compromise probability. Equality appears when $j = 1$.

Therefore, the chance of having any secret link key compromised is

$$P_c^{(Multi)} \geq 1 + p_e(x_c) - \sum_{j=1}^{\infty} q_j (1 - x_c)^j \ . \tag{4}$$

V. Performance Evaluation

We used Matlab in our simulations, in which N sensors are randomly deployed to an area of 1000 by 1000 m^2. Wireless transmission range is 100 m. Every pair of nodes had a chance of p_{local} being securely connected. We chose one source randomly and tried to establish a secret link key to each of its security-disconnected neighbors. In our simulations, we used a simplified circular connectivity model and an abstract of the physical and networking models. Furthermore, we focused on key sharing among different nodes. Therefore, simulations through NS2 or OPNET are unnecessary at this stage.

A. Transmission Cost

Since the transmission overhead of our scheme depends largely on how far the common bridge node is away from the source node, we measured the average hop count as the transmission cost. In Fig. 3, we show such average hop count as a function of key sharing connectivity, p_{local}. Curves for networks with different total number of sensors are shown to represent the effect of different node density.

It is interesting to see from Fig. 3 that the number of hops toward the closest common bridge node is roughly the same for various N values when p_{local} is either large or small. This might have been caused by the opposite effects of more nodes and more to-be-connected neighbors. When p_{local} is in the middle, larger N leads to longer hops because of more to-be-connected neighbors.

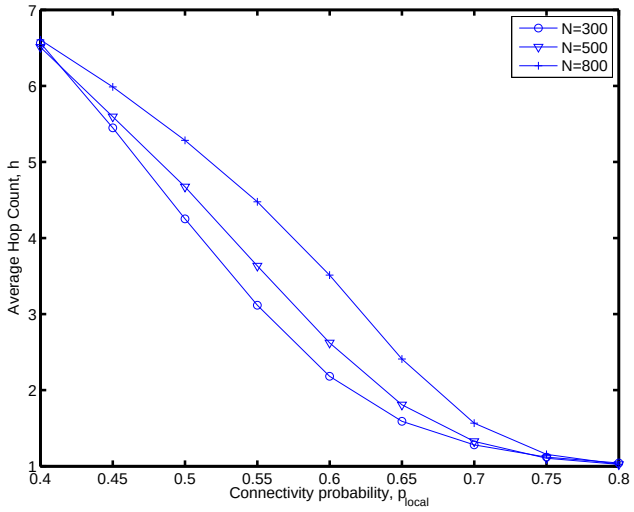

Fig. 3. The average hop count from the common bridge node toward the source.

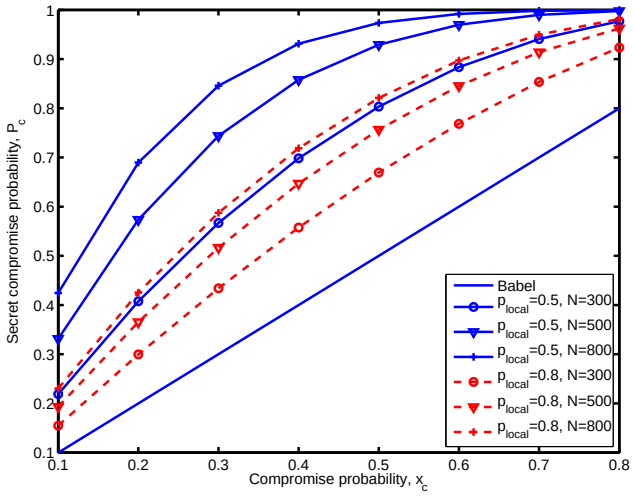

Fig. 4. The relative value of compromise probability for Babel and the secure multi-hop path scheme [2]. We neglect the term $p_e(x_c)$ in both (2) and (4). Note that the compromise probability shown for the secure multi-hop path scheme is a lower bound.

B. Compromise Probability

We have also measured the chance of secret link keys being compromised and disclosed. The comparisons were made between the Babel scheme and the secure multi-hop path scheme [2]. In the secure multi-hop path scheme, the source node finds a secure multi-hop path, on which every two consecutive nodes share at least a key, to deliver a secret link key. We first counted the occurrences of having j unique intermediate bridge nodes and calculated the compromise probability based on (4). We have omitted the term of $p_e(x_c)$ in our comparisons.

The results are shown in Fig. 4, where we demonstrate the performance of the Babel scheme and the secure multi-hop path scheme. Note that the performance of the Babel scheme does not change with N or p_{local} due to its unique property of employing one common bridge node. We can see that the Babel scheme has a lower compromise probability for all x_c values. As p_{local} increases, the compromise probability of the secure multi-hop path scheme lowers. This is because of the smaller hop and smaller number of intermediate bridge nodes when p_{local} increases.

VI. Conclusions

We have proposed the Babel scheme that delivers secret link keys toward multiple to-be-connected neighbors. The Babel scheme makes use of a remote node that shares at least a key with the source and each of its to-be-connected neighbors. Our preliminary analysis and performance evaluation show that Babel outperforms existing schemes such as the secure multi-hop path scheme in security performance. In future work, we will focus on analyzing the security performance of the Babel scheme and its comparison with related techniques. This is especially important because the Babel scheme employs a single node to forward the secret keys and fault-tolerance should be investigated.

Acknowledgment

This work was supported in part by grant LBoR0078NR00C and by the National Science Council of Taiwan, R.O.C., under grants NSC 95-2221-E-305-003.

References

[1] I. F. Akyildiz, W. Su, Y. Sankarasubramaniam, and E. Cayirci, "A survey on sensor networks," *IEEE communications Magazine*, pp. 102–114, August 2002.

[2] L. Eschenauer and V. D. Gligor, "A key-management scheme for distributed sensor networks," in *Proc. of the 9th ACM conference on Computer and communications security*, Washington, DC, USA, November 18-22 2002, pp. 41–47.

[3] H. Chan, A. Perrig, and D. Song, "Random key predistribution schemes for sensor networks," in *Proc. of IEEE Symposium on Security and Privacy*, Berkeley, California, May 11-14 2003, pp. 197–213.

[4] W. Du, J. Deng, Y. S. Han, P. K. Varshney, J. Katz, and A. Khalili, "A pairwise key pre-distribution scheme for wireless sensor networks," *ACM Transactions on Information and System Security*, vol. 8, no. 2, pp. 228–258, May 2005.

[5] D. Liu and P. Ning, "Establishing pairwise keys in distributed sensor networks," in *Proc. of the 10th ACM Conference on Computer and Communications Security (CCS '03)*, Washington, DC, USA, October 27-31 2003, pp. 52–61.

[6] G. Li, H. Ling, and T. Znati, "Path key establishment using multiple secured paths in wireless sensor networks," in *Proc. of CoNEXT '05*, 2005, pp. 43–49.

[7] D. Huang and D. Medhi, "A byzantine resilient multi-path key establishment scheme and its robustness analysis for sensor networks," in *Proc. of 19th IEEE International Parallel and Distributed Processing Symposium*, Colorado, USA, April 4-8 2005, pp. 240b–240b.

[8] J. Deng and Y. S. Han, "Using MDS codes for the key establishment of wireless sensor networks," in *Proc. of the International Conference on Mobile Ad-hoc and Sensor Networks (MSN '05)*, X. Jia, J. Wu, and Y. He, Eds., Wuhan, P. R. China, December 13-15 2005, vol. 3784 of *Lecture Notes in Computer Science (LNCS)*, pp. 732–744, Springer-Verlag.

[9] P. Traynor, H. Choi, G. Cao, S. Zhu, and T. LaPorta, "Establishing pair-wise keys in heterogeneous sensor networks," in *Proc. of the 25th Conference of the IEEE Communications Society (Infocom '06)*, Barcelona, Spain, April 23-29 2006.

[10] M. J. Miller and N. H. Vaidya, "Leveraging channel diversity for key establishment in wireless sensor networks," in *Proc. of the 25th Conference of the IEEE Communications Society (Infocom '06)*, Barcelona, Spain, April 23-29 2006.