A Novel Key Redistribution Scheme for Wireless Sensor Networks

Chun-Fai Law, Ka-Shun Hung and Yu-Kwong Kwok
Department of Electrical and Electronic Engineering
The University of Hong Kong, Pokfulam Road, Hong Kong

***Abstract*— Key management has long been a challenging problem in wireless Distributed Sensor Networks (DSNs) due to their high security requirements and strict resource constraints. Recently, a randomized key pre-distribution scheme has been introduced to serve as a practical solution and many improvements are subsequently proposed. These schemes mainly focus on key allocations based on pre-deployment estimates of post-deployment information items, such as location data and attack probabilities. Unfortunately, such information items may be unavailable or may change over time.**

Based on adaptability to post-deployment contexts, we propose a key redistribution scheme that exploits neighboring keys from connected neighbors to reach unconnected nodes. We show that our scheme can be integrated into most existing key pre-distribution schemes to further improve their performance. We demonstrate our proposed scheme's salient features, such as high connectivity, high resilience, and efficient memory usage, by both analytical and simulation results.

Keywords—wireless sensor networks, distributed sensor networks, key management, trust bootstrapping, deployment knowledge, redistribution.

I. Introduction

Wireless Distributed Sensor Networks (DSNs) have gained remarkable attention after recent advancements of electronic and wireless technologies. A DSN usually consists of ultra small autonomous devices called sensor nodes which are battery-powered, limited in memory storage and computational power. In a typical application scenario, sensors cooperatively monitor physical and environmental conditions and then transmit collected data to a sink node or base station via wireless links for further analysis. For instance, in a military setting, DSNs are deployed in a large scale with well over 10,000 nodes for gathering a large volume of target recognition data. SmartDust [10], WINS [12], and μAmps [13] are well known examples of DSN projects.

Usually deployed in hostile areas, DSNs require a high level of data security as nodes could be captured and consequently communication may be eavesdropped or altered. Various cryptographic protection schemes are needed in a DSN to guard against various types of malicious attacks. *Key management* [7] is one of the most critical components in a DSN security system.

Currently, key pre-distribution schemes are widely considered as practicable solutions in DSNs [7]. A typical key pre-distribution scheme works by having keys distributed to all sensor nodes prior to deployment. Some methods such as using master key or pairwise private sharing of keys are also proposed [7]. However, these schemes are either insecure or impractical. Recently, Eschenauer *et al.* [6] pioneered this field of research by proposing a randomized key pre-distribution scheme, which relies on probabilistic key sharing among nodes using random graph theory [7]. In their scheme, a secure link is formed only if two neighboring nodes share the same key. The connectivity of the resultant network is then characterized by a probability. An overview of this scheme (referred to as *basic scheme* in this paper) is given in Section III.

Although most sensor nodes are static (or having limited mobility after the deployment phase), the DSN topology is dynamic over the lifetime of the network. Topology may be changed when nodes are added or removed (e.g., replacing failing nodes); or due to different sleeping schedules of nodes for energy saving. In this regard, a key management scheme should adapt to the change of topology to maintain certain level of connectivity and security. However, existing key pre-distribution schemes usually focus only on the pre-deployment phase and do not take advantage of wide variety of information collected after deployment.

In addition to adaptability, most key pre-distribution schemes exhibit the following problems:

1) *Wasted nodes*: High local connectivity does not imply high global connectivity. Isolated components may still exist. These isolated components are very useful to expand the network coverage and lifetime as shown in [1]. Key pre-distribution schemes, which are based on random graph theory, can only guarantee most nodes (i.e., about 99%) are reachable by using a considerably large key ring size.
2) *Wasted keys*: With probabilistic key sharing, large amount of keys are unused. For example, when a network has 10,000 nodes with 48 neighbors on average, a key ring size of 200 from a pool of 100,000 keys can only achieve local connectivity of 0.33. This means that the expected number of connected neighbors is about 16 and over 92% (184 out of 200) of keys are unused.
3) *Long path length*: When the network is under attack, links are broken and a fraction of nodes are sacrificed to protect the rest of the network. Thus, the base station usually relies on multiple sources of data to eliminate false alarms. As a result, abundant and reliable secure links are necessary to help reduce the impact of various attacks. However, key pre-distribution schemes usually create long paths due to low connectivity.

In this paper, we exploit the idea of key redistribution to address the problem of dynamic network topology and all the issues stated above. Our proposed redistribution scheme makes effective use of keys in connected neighbors to access those unconnected ones. Analytical and simulation results show that our proposed scheme achieves much better connectivity and resilience compared to the basic scheme. Under high connectivity condition, our scheme also outperforms the *deployment knowledge scheme* [4].

The rest of this paper is organized as follows. In Section II, we discuss the related work in the literature. Section III overviews the basic scheme and introduces our proposed key redistribution mechanism which is integrated with the basic scheme. In Section IV, we evaluate the performance of the proposed scheme analytically. Afterward, simulation results and performance comparison are presented and discussed in Section VI. We conclude the paper by suggesting some future research directions in Section VII.

II. Related Work

Based on the random graph theory, Eschenauer *et al.* [6] proposed the pioneering key pre-distribution scheme in 2002. Pietro *et al.* [8] subsequently also demonstrated the connectivity properties of random sensor networks. In Section III below we give a more detailed description of this basic scheme, which is further improved by many researchers, most notably Chan *et al.* [3] and Du *et al.* [4]. We first present a brief overview of these improved schemes.

Chan *et al.* [3] proposed a q-composite key pre-distribution scheme which uses q common keys to establish a secure communication between two nodes. This scheme achieves better security under small scale attack while increasing vulnerability to large scale attack. They also proposed a random pairwise scheme to provide full resilience against node capture. However, the maximum network size supported is reduced.

Du *et al.* [4] proposed a random key pre-distribution method built on the Blom key pre-distribution scheme which also improves the resilience of the network. This scheme has a nice threshold property which guarantees that the network is perfectly secure as long as the number of compromised nodes are smaller than the threshold. Later, Du *et al.* tried to reduce the memory usage using the deployment knowledge (e.g., location) while achieving the same level of connectivity. However, such knowledge is not always available especially in hostile areas as well as dynamic network environments. Specifically, offline estimation of node distribution is considered unrealistic and inaccurate under these situations.

Eltoweissy *et al.* [5] developed a protocol for dynamic re-keying in the post-deployment phase. Under the long life cycle assumption, re-keying is necessary in the addition or revocation of nodes. Re-keying can prevent further compromising of the rest of network against node capture. When some nodes are suspected to be compromised, the base station sends re-keying instruction to the cluster controllers to trigger the corresponding re-keying operations. The drawback is that their approach requires coordination among a base station and cluster controllers.

TABLE I
Notation

d	expected degree (number of neighbors) after deployment
d_{conn}	expected number of connected neighbors
l_{pre}	links secured by a single key
l_{rev}	links secured by hashing two keys
m	key ring size
n	number of sensor nodes in the network
p_{local}	local connectivity after shared key discovery phase
p'_{local}	local connectivity after key redistribution phase
p_{global}	global connectivity
R	transmission range
$\|S\|$	size of key pool
t_{grace}	grace period for key deletion
x	number of nodes compromised

Throughout the present paper, *basic scheme, deployment knowledge scheme and key redistribution* refer to the scheme described in [6], [4], and our proposed scheme described in Section III-B, respectively. Table I contains the notation used in our work.

III. Proposed Key Redistribution Scheme

In this section, we give an overview of the *basic scheme*. Then we present our key redistribution scheme as an integration to the basic key management protocol in DSNs.

A. *Eschenauer* et al.*'s basic scheme*

Eschenauer *et al.* [6] proposed a random key pre-distribution scheme in 2002. Based on random graph theory, the *basic scheme* relies on probabilistic key sharing among nodes and uses a simple shared key discovery protocol and path key establishment for connection setup process. This scheme assumes that the sensor network forms a random graph and keys are installed in nodes prior to deployment. Each sensor node installs a random set of keys from the key pool. Any two neighbors are connected if they are able to find a common key. The basic scheme is divided into three phases:

- *Key pre-distribution phase*: Each sensor node randomly selects m distinct keys from a large pool of $|S|$ keys. The set of m keys is called key ring of the node and is loaded into the memory of each sensor before deployment.
- *Shared key discovery phase*: After nodes are deployed, each sensor node attempts to find neighbors which share at least one common key. If a common key exists, it is used to secure the link and these two nodes are then connected. After the key discovery phase, a connected graph is formed.
- *Path key establishment phase*: In the connected graph, there exists a path from a source node to any unconnected neighbors. The source node can securely deliver a path key to selected neighbors via the indirect path.

B. *Key redistribution protocol*

Our proposed scheme is illustrated in Figure 1.

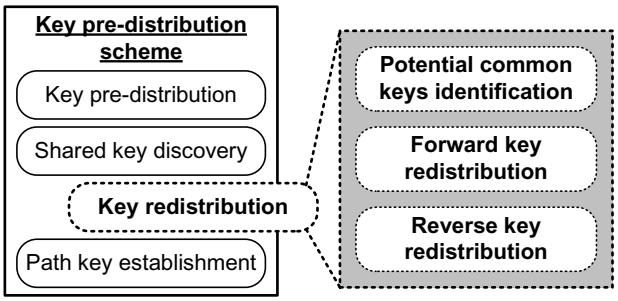

Fig. 1. The basic scheme with key redistribution phase.

1) Potential common keys identification: At the end of the shared key discovery phase, each node has received lists of key identifiers, $list_{Kid}$, broadcast from its neighbors. By intersecting $lists_{Kid}$ of the connected and that of the unconnected neighbors, a node is able to determine all potential common keys and redistribute them securely via encrypted links from connected neighbors.

2) Forward key redistribution: The *potential common keys* identified in the previous stage refer to the keys that exist in current connected neighbors and they can be used to connect to the unconnected neighbors. However, these keys may not be suitable for redistribution in all situations. Thus, a node needs to send a request to those connected neighbors which contain certain selected *potential common keys* and waits for their replies. In the case that the key cannot be used for redistribution (e.g., the key is currently in use in the connected neighbors), the node will try to request for the next selected *potential common key*. The selection preference is based on ascending order of the keys in the key occurrence pattern.

The objective of the above algorithm is to reduce the number of trials based on the observation that two connected neighbors will have certain probability to be neighborhood themselves. Specifically, the smaller the number of occurrence of the keys, the higher will be the chance of successfully getting the key for redistribution. This stage will terminate if the node successfully gets the key or all the *potential common keys* are tried. In the first case, the key redistribution phase can be regarded as completed. The ownership of the key is transferred to the requesting node and the original owner is required to delete the key for security purpose.

A simple example is shown in Figure 2 to illustrate the above processes. For the latter case, another method known as *Reverse key redistribution* is proposed to redistribute the keys and this will be discussed in Section III-B.3.

3) Reverse key redistribution.: When all the potential common key requests are failed, a node can only resort to creating a new key for connection establishment with the aid of a potential common key k_{common}. Using the concept from the q-composite key scheme [3], an unused key k_{unused} is sent to a connected neighbor, which has the required k_{common}, to generate a new communication key $k_{new} = hash(k_{common}||k_{unused})$ in which keys are hashed in some canonical order. As shown in Figure 3, the $k_{new} =$

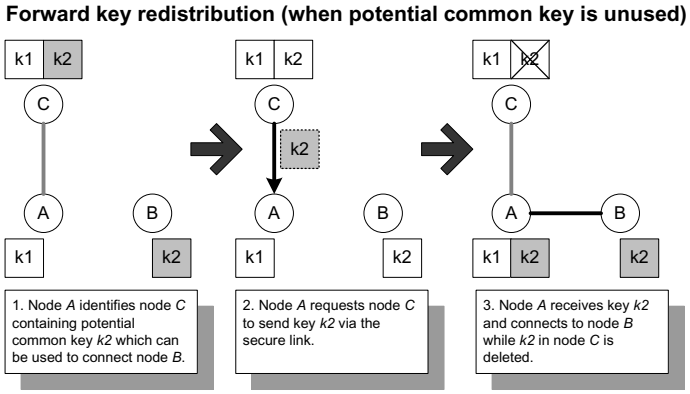

Fig. 2. Forward key redistribution.

$Hash(k2||k3)$ is encrypted using $k_{common} = k2$ in the connected neighbor and then securely delivered to the target node either directly or backward through the original node. This method does not require the original node to know k_{common} for decryption and allows redistributing new assigned key securely to the unconnected neighbor. The direction of redistribution is reversed as the key is moving towards the connected neighbor.

The memory usage of a node may increase or decrease based on its role in redistribution. To increase the chance of key matching, we define a grace period t_{grace} which allows the memory to be overloaded for a short period of time. Multiple requests on the same key are handled on a first-come-first-served basis. Unsuccessful requests will result in reverse key redistribution. At the end of the grace period, excessive unused keys are deleted randomly to restore original memory usage level as well as security requirements. The maximum memory usage is bounded by the sum of initial key ring size and number of neighbors, $O(m + d)$.

The key redistribution scheme is presented as pseudo-code in Algorithm 1. It can be executed multiple times to further increase connectivity as shown in Section IV with the tradeoff of additional computational and communication overhead.

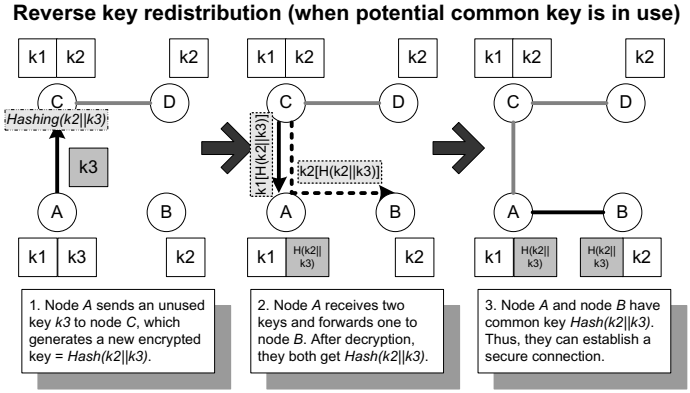

Fig. 3. Reverse key redistribution

Algorithm 1 Key redistribution mechanism

$L_c \leftarrow$ lists of connected neighbors
$L_u \leftarrow$ lists of unconnected neighbors
for $i \in L_u$ **do**
  /* potential common keys identification */
  $commonSet$ = {} /* store the potential common keys */
  for $j \in L_c$ **do**
    $sharedKeys = intersect(key(i), key(j))$
    $commonSet = union(commonSet, sharedKeys)$
  end for
  /* forward key redistribution */
  $request = rearrange(commonSet)$
  /* in ascending order of occurrence count */
  $reply = broadcast(request)$
  if $reply(key, node\_id) == successful$ **then**
    $redistribute(key, node\_id \rightarrow self)$
    /* $node\_id$ then delete that key */
  else
    /* reverse key redistribution */
    $send(unusedKey, self \rightarrow node\_id)$
    $encryptedKey = received(node\_id \rightarrow self)$
    $forward(encryptedKey, self \rightarrow i)$
  end if
end for

IV. Mathematical Analysis

A. Connectivity

Connectivity is widely considered to be an important performance metric in key management schemes in wireless sensor networks [7]. It is further divided into two categories: *local connectivity* and *global connectivity*. Local connectivity refers to the probability of establishing a secure connection between two neighbors. Thus, it is the probability of the event that two neighboring nodes share a common key. For the connectivity of the whole network, global connectivity is often used to measure the ratio of the number of nodes in the giant (largest isolated) component to the total network size. Specifically, it indicates the percentage of wasted (unreachable) nodes in the sensor network.

Since our key redistribution scheme critically depends on the number of connected neighbors after the sharing key discovery phase, we first examine at the local connectivity estimated in the basic scheme [6]. The probability that any two neighboring nodes share at least a common key is: p_{local} = 1-Pr[two nodes do not share any key], and thus:

$$p_{local} = 1 - \frac{\left(1 - \frac{m}{|S|}\right)^{2(|S|-m+\frac{1}{2})}}{\left(1 - \frac{2m}{|S|}\right)^{(|S|-2m+\frac{1}{2})}} \quad (1)$$

After the sharing key discovery phase, keys from connected neighbors are redistributed to further increase connectivity. Although those keys are distributed in different nodes, they are accessible through secure links as if from a single storage. Key ring of a particular node is combined with that of the connected neighbors virtually. Given the expected node degree d, we can compute the expected number of connected neighbors d_{conn} before redistribution:

$$d_{conn} = p_{local} \times d \quad (2)$$

When there exists at least one connected and one unconnected neighbors (i.e., $0 < d_{conn} < d$), key redistribution can take place. Let p'_{local} be the probability that a node and one of its unconnected neighbors share at least a common key after the first key redistribution, which is calculated as:

$$p'_{local} = 1 - \left[\frac{\binom{|S|-m}{m}}{\binom{|S|}{m}}\right]^{d_{conn}+1} \quad (3)$$

where $d_{conn} < d$.

The equation can be further simplified by using Stirling's approximation for $n!$ in order to obtain:

$$p'_{local} = 1 - \left(\frac{\left(1 - \frac{m}{|S|}\right)^{2(|S|-m+\frac{1}{2})}}{\left(1 - \frac{2m}{|S|}\right)^{(|S|-2m+\frac{1}{2})}}\right)^{d_{conn}+1} \quad (4)$$

Based on the configuration settings in Section VI, Figure 4 shows the analytical results of local connectivity estimated by the basic scheme and our key redistribution scheme using Equation (4). With increasing key ring size m, the initial connectivity p_{local} rises slowly, resulting in more connected neighbors. Key redistribution takes advantage of the increasing key supplies from these connected neighbors. However, the improvement of connectivity p'_{local} is at a much faster rate than p_{local} because $O(m)$ keys are accessible once an additional neighbor joins. This results in further connections in the next redistribution stage. More importantly, this plot shows that it is possible for our key redistribution scheme to boost the connectivity up to 1. This indicates that all the isolated nodes are finally rejoined and the network becomes fully connected.

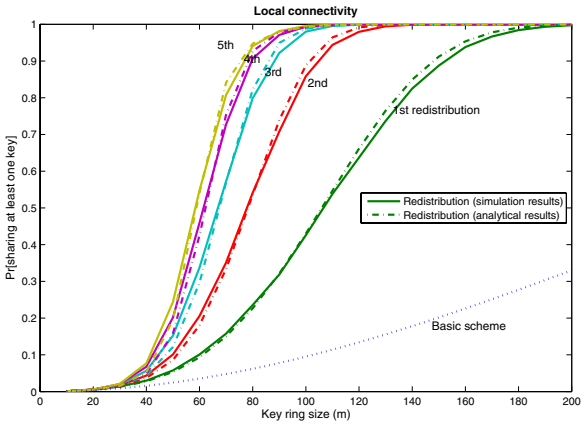

Fig. 4. Probability of sharing at least one key between two neighboring nodes.

B. Isolated nodes

A DSN with a high local connectivity may still have isolated components. Traditionally, nodes which are not connected with the giant component are considered as "unreachable"

TABLE II

COMPARISONS IN TERMS OF ISOLATED NODES AND GLOBAL CONNECTIVITY

Key ring size	Key redistribution		Basic scheme		Du's scheme	
	iso	p_{global}	iso	p_{global}	iso	p_{global}
10	9563	0.0004	9569	0.0004	3724	0.0214
50	1877	0.7694	3007	0.0484	36	0.9955
100	5	0.9995	149	0.9845	13	0.9987
150	0	1	2	0.9998	11	0.9980
200	0	1	0	1	13	1

via secure links and "wasted". Our scheme is able to change those wasted nodes from "useless" to "useable" through proper redistribution of keys. We exploit the following important observation: The common keys on the path are necessary, but not the nodes along the path. That is why the path key establishment of the basic scheme does not work.

Table II shows the simulation results of the number of isolated nodes (iso) and the global connectivity (p_{global}). Since key distribution patterns are altered by the reverse redistribution process, it is not suitable to estimate these parameters based on Erdös and Rényi [9] random graph theory which requires uniform node and key distribution. In this paper, we present the simulation results using configurations described in Section VI.

C. Resilience

Among all the potential security threats encountered by a DSN, node capture is one of the most challenging problems. Sensor resource constraints limit the usage of advanced protection technologies [11]. During node capture, the key ring may be exposed completely and those compromised keys can be used to attack other existing links.

To evaluate the resilience of our scheme against node capture, we measure the fraction of communication compromised when x nodes are captured. The expected fraction of total keys/links compromised in the basic scheme can be estimated as:

$$f_{comp}(m) = 1 - \left(1 - \frac{m}{|S|}\right)^x \tag{5}$$

where x is the number of nodes compromised. In addition, from the q-composite scheme [3] with $q = 2$, the expected fraction of total keys being compromised is calculated as $(f_{comp}(m))^2$.

In our key redistribution scheme, the total secure links of a node actually consist of l_{pre} links from pre-distribution phase and l_{rev} links from reverse redistribution phase. These l_{rev} links are special because they are made secure by hashing two keys. When a node is captured, m keys are exposed. They can be used to compromise links which are made secure by a single key in the network (*Type I*)[1]. Meanwhile, they can also be used to compromise links which are protected by hashing two keys (*Type II*).

[1]Forward key redistribution will not affect the overall key distribution patterns.

The probability that a link belongs to Type I or Type II are $\frac{l_{pre}}{l_{pre}+l_{rev}}$ and $\frac{l_{rev}}{l_{pre}+l_{rev}}$, respectively, and the expected fraction of total links being compromised can be estimated as:

$$\Pr[\text{TypeI}]\left(f_{comp}(m)\right) + \Pr[\text{TypeII}]\left(f_{comp}\left(m\right)\right)^2 \tag{6}$$

V. AN ILLUSTRATIVE EXAMPLE

We present a simple example here to illustrate the effect of key redistribution on the basic scheme. Consider a sensor network with $n = 10,000$ nodes and the expected number of neighbors of a node is $d = 48$. Given $|S| = 100,000$ and $m = 106$, we can use Equation (1) to calculate the connectivity after key-setup phase. The result is $p_{local} \approx 0.1$. Thus, $d_{conn} = p_{local} \times d = 0.1 \times 48 \approx 5$. After these 5 neighbors are connected, key redistribution takes place in order to connect the remaining 43 neighbors. From Equation (4), the probability of sharing at least one key with unconnected neighbors increases from 0.1 to 0.5. Consequently, 19 additional neighbors are connected and this implies at most 19 keys are redistributed. Our scheme indicates that we only need $m = 106$ keys instead of $m = 263$ keys out of a key pool size $|S| = 100,000$ to achieve a neighborhood connectivity of 0.5, i.e., on the average, 24 out of 48 neighbors are connected.

VI. SIMULATION RESULTS

A. System Configuration

Simulations are performed in MATLAB with the following parameters:

- The number of nodes, n: 10,000
- The size of key pool, $|S|$: 100,000
- The deployment region: 1000 × 1000
- The wireless transmission range for each node, R: 40
- The average number of neighbors, d: 48

B. Local connectivity

We compare our simulation results with the *basic scheme* [6] and the scheme using deployment knowledge [4]. We also integrated our key redistribution scheme into the deployment knowledge scheme to evaluate the portability and performance. The combined scheme is also investigated and plotted in the same graph. Figure 5 illustrates the local connectivity, i.e., the probability of sharing a key between two nodes against various key ring size m.

First of all, our simulation results match well with the analytical results presented in Section IV in which the redistribution scheme improves the connectivity with increasing number of connected nodes. Given $|S| = 100,000$, the connectivity rises sharply when p_{local} is around 0.1–0.3. To reach $p = 0.5$, our scheme requires $m = 106$, the basic scheme needs $m = 263$, while Du's approach only needs $m = 67$. However, at high connectivity, $p'_{local} > 0.8$, our scheme outperforms both approaches. By using the idea of key redistribution, the performance gain comes from the adequate supply of keys from connected neighbors. This is similar to obtaining sub-key pools in Du's scheme via the established secure links. When p_{local} further increases, our scheme achieves the maximum connectivity.

In the combined deployment knowledge and key redistribution scheme, the plot demonstrates the portability of our algorithm. The curve of combined scheme shows that the connectivity can be greatly increased with much fewer keys. This is highly important in the memory-constrained sensor devices.

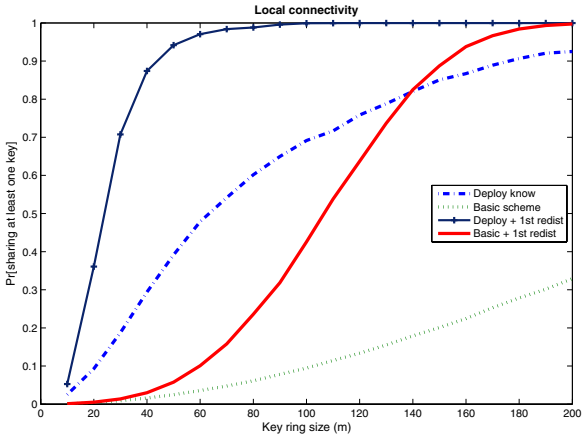

Fig. 5. Local connectivity.

C. Network resilience against node capture attack

From Figure 6, we note that the network resilience of our key redistribution scheme is better than that of the basic scheme and approaching that of the scheme using deployment knowledge. However, our scheme does not rely on any deployment knowledge and thus, requires larger key ring size m to achieve certain level of connectivity. In case of high connectivity, our approach achieves better resilience than that of both schemes. The reason is that the larger the number of links created by the redistribution process, the higher the chance of them to be secured by hashing two keys. For example, with $p = 0.5$, only 87 out of 106 keys are used to secure 20% of the links, but the remaining 19 redistributed keys are used to secure 80% of the links of the network which are locally concentrated.

VII. Conclusions

We have presented a new key redistribution scheme which can augment Eschenauer *et al.*'s basic scheme to improve both connectivity and security of sensor networks, without additional deployment knowledge. Our scheme makes use of keys from neighbors connected after key setup phase to reach the unconnected nodes. Analytical and simulation results demonstrate significant improvements over prior schemes [4], [6] and the portability for integration with them. Currently, we are quantifying the communication and computational overheads of our scheme. Moreover, we also consider using the small world phenomenon to further enhance the resilience against node capture under various attack patterns. Finally, we are also trying to incorporate the attack probabilities [2] in our proposed key management system.

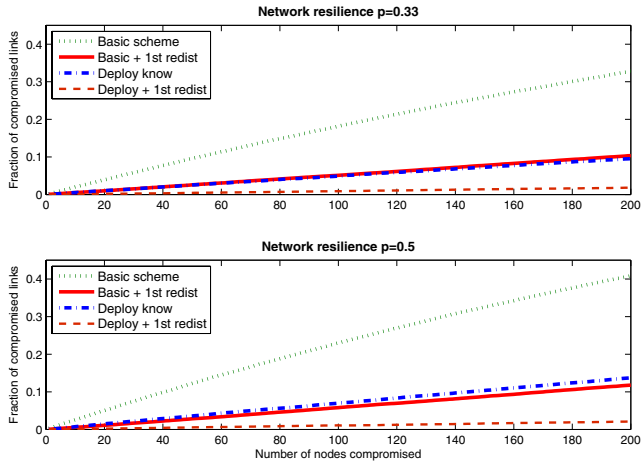

Fig. 6. Network resilience.

References

[1] J. N. Al-Karaki and A. E. Kamal, "Routing techniques in wireless sensor networks: a survey," *IEEE Wireless Communications*, vol. 11, no. 6, pp. 6–28, Dec. 2004.

[2] S.-P. Chan, R. Poovendran, and M.-T. Sun, "A key management scheme in distributed sensor networks using attack probabilities," *Proc. GLOBECOM 2005*, vol. 2, Dec. 2005.

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

[4] W. Du, J. Deng, Y. S. Han, S. Chen, and P. K. Varshney, "A key management scheme for wireless sensor networks using deployment knowledge," *Proc. INFOCOM 2004*, vol. 1, Mar. 2004.

[5] M. Eltoweissy, M. Moharrum, and R. Mukkamala, "Dynamic key management in sensor networks," *IEEE Communications Magazine*, vol. 44, no. 4, pp. 122–130, Apr. 2006.

[6] L. Eschenauer and V. D. Gligor, "A key-management scheme for distributed sensor networks," *Proc. 9th ACM Conference on Computer and Communications Security*, pp. 41–47.

[7] Y.-K. Kwok, "Key management in wireless sensor networks," in *Security in Distributed and Networking Systems*, Yang Xiao and Yi Pan (eds.), World Scientific Publishing Co., 2007.

[8] R. Di Pietro, L. V. Mancini, A. Mei, A. Panconesi, and J. Radhakrishnan, "Connectivity properties of secure wireless sensor networks," *Proc.e 2nd ACM Workshop on Security of Ad Hoc and Sensor Networks*, pp. 53–58.

[9] P. Erdös and A. Rényi, "On the evolution of random graph" *PubI. Math. Inst. Hung. Acad. Sci.*, 5 (1960), pp. 17–61.

[10] Smart Dust Project, `http://robotics.eecs.berkeley.edu/~pister/SmartDust`, 2007.

[11] F. Stajano and R. Anderson, "The resurrecting duckling: security issues for ad-hoc wireless networks," *Proc. 7th Int'l Workshop Security Protocols*, 1999.

[12] WINS (Wireless Integrated Sensor Network) Project, `http://www.janet.ucla.edu/WINS/`, 2007.

[13] μAmps Project, `http://www-mtl.mit.edu/researchgroups/icsystems/uamps/`, 2007.