Using Combined Keying Materials for Key Distribution in Wireless Sensor Networks

Murat Ergun and Albert Levi
Sabancı University, Istanbul, Turkey
mergun@sabanciuniv.edu, levi@sabanciuniv.edu

Abstract— **In this paper, we propose a probabilistic key predistribution scheme for wireless sensor networks that increases connectivity of the basic scheme while keeping sizes of keyring and key pool fixed. We introduce the concept of XORed key, which is the bitwise XOR of two regular (a.k.a. single) keys. Sensor nodes are preloaded with a mixture of single and XORed keys. Nodes establish secure links by using shared XORed keys whenever possible. If node pairs do not have any shared XORed or single keys, they transfer keys from their secure neighbors in a couple of ways, and use them to match with their XORed keys. In this way, the probability of securing links, i.e. local connectivity, increases. The decision of which key is to be transferred from which node is given based on local information at the hand of the nodes. We aim to control the resilience of the network against node capture attacks by using XORed keys since an attacker has to know either both single key operands or the XORed key itself. Simulations show that our scheme is up to 50% more connected as compared to basic scheme. Also it has better resilience performance at the beginning of a node capture attack. When it starts to deteriorate, the difference between the resilience of our proposed scheme and basic scheme is not greater than 5%.**

Keywords— key predistribution; probabilistic key management; wireless sensor networks; security

I. Introduction

Wireless sensor networks [1] are composed of mobile, battery powered, small devices called sensors, which are capable of collecting environmental information such as temperature and humidity, making computation of easy operations, storing data in memory, and communicating with each other by transmitting data with radio add-on. Wireless sensor networks gained importance in the last decade due to widespread application areas from environmental monitoring to medical use, and from object tracking to military fields. The critical usage areas of sensor networks also provide point of attraction for researchers.

Wireless sensor networks are generally deployed in relatively wide geographic areas and require sensors nodes in large quantities to achieve high connectivity in radio communication; therefore, they need to be inexpensive. Despite the recent advances in technology, sensor nodes still have restricted computing power, limited storage capacity and short communication range because of cost and battery considerations.

There exist some potential security risks in wireless sensor networks when they carry critical data. Wireless nature of communication provides advantages to adversaries. They can surreptitiously listen to communication between sensors without being noticed. Another risk is caused of unconfined sensors in the field. When the sensor nodes are used in unattended fields, they become open to physical attacks.

In order to overcome these security threats, cryptographic security mechanisms should be employed. However integration of security systems in wireless sensor networks is not intuitive due to some limitations. Lack of network topology information prior to deployment causes an infrastructureless network. Unknown infrastructure and unreliable deployment zone make the trusted third party solutions almost inapplicable for wireless sensor networks. In recent years, important researches have been done to make public key cryptosystems feasible for resource-limited sensor nodes. However, using public key cryptosystems in large and dense sensor networks in a practical way is still a remote possibility. The remaining solution is conventional cryptosystems that do not require too much computational power, energy and running time. Here the open question is distribution of keying material within wireless sensor networks and without being compromised.

Research on key distribution in wireless sensor networks is concentrated on probabilistic key predistribution. One of the first schemes is proposed by Escheneauer and Gligor [2], which is also known as the *basic scheme*. In the basic scheme, there is a key pool that contains a large amount of keys and their unique key identifiers. The keyring of a sensor node is formed by selecting some keys from key pool uniformly random without replacement. After the deployment of the nodes, two neighboring sensor nodes try to find a common key in their keyrings to be used as their link key. Actually, they share such a common key probabilistically. This probability, which is called as *local connectivity*, depends on the size of the key pool and of the keyring. More specifically, local connectivity is directly proportional to the keyring size and inversely proportional to the key pool size.

In this paper, we propose a scheme that increases connectivity of basic scheme while keeping sizes of keyring and key pool fixed. We also propose to use XOR of two regular keys in the keyrings of the sensor nodes. There are some other papers in the literature, such as [3], that uses

This work is supported by Scientific and Technological Research Council of Turkey (TUBİTAK) under grant 104E071. Murat Ergun is supported by TÜBİTAK BİDEB scholarship.

XORed keys. However, to the best of our knowledge, our scheme is the first in the literature that uses XORed keys directly in the keyrings. XORed keys are pushed to be used as link keys in our scheme for the purpose of reinforcing resilience of the network, since an attacker has to capture either both operand regular keys used in an XORed key or the XORed key itself in order to compromise a secure link. Another contribution of the proposed scheme is that, we employ a novel key transfer phase in which missing keys to are transferred from secure neighbors to improve local connectivity. The unique characteristic of our key transfer phase is that, the transfer decisions are taken by the nodes using local information; no extra broadcast messages are flooded in the network as in path-key establishment phase of the basic scheme. With the optimal mixture of regular and XORed keys in the keyrings, local connectivity performance of our scheme is up to 50% more than the connectivity achieved in basic scheme [2]. Moreover, our scheme does not significantly deteriorate the resilience of the network against node capture attacks as compared to the basic scheme.

The rest of this paper is organized as follows. Section II gives an overview of our scheme and describes its phases in detail. Section III explains performance metrics and gives comparative simulation results. Related work in the literature is given in Section IV. Finally Section V concludes the paper.

II. Proposed Scheme

In this section a random key predistribution scheme for wireless sensor networks is proposed. In this scheme, we use a novel key type called XORed key, which is bitwise XOR of two regular keys. In order to differentiate XORed and regular keys, we rename regular keys as *single keys* throughout this paper. Our scheme uses a combination of XORed and single keys in the keyrings of sensor nodes. Since XORed keys are produced using two single keys, a secure link established using XORed keys is more resistant to attacks.

As in Chan et al.'s scheme [3], our scheme uses two single keys contributed in the establishment of link key. However, in our scheme, keyrings of nodes are composed of variety of XORed and single keys, as opposed to the homogeneous structure of keyrings in Chan et al.'s scheme [3].

Our scheme has a phase called transfer phase which increases the local connectivity of the network by transferring keys from secure neighbors whenever needed. Transferred keys are not used directly; instead their XORed forms are used. In other words, keys are transferred to complete missing operands of XORed keys in the keyring.

The symbols and notations used in this paper are given in Table 1.

A. Overview

Our scheme is based on probabilistic key predistribution like the basic scheme [2]. It includes three main phases which are (*i*) key predistribution, (*ii*) shared-key discovery, and (*iii*) key transfer phases. Although key predistribution and shared-key discovery phases in our scheme have similar characteristics with corresponding phases of the basic scheme, our scheme differs from basic scheme with one important feature, which is the keying material. There are two types of keys stored in the keyrings of the nodes. Transfer phase is a novel phase in our scheme. In this phase, some manipulations are performed to improve the local connectivity of the proposed scheme.

Table 1. List of symbols used in this paper

SP	Key pool of single keys
XP	Key pool of XORed keys
xk_i	XORed keyring of node i
sk_i	Single keyring of node i
$\lvert SP\rvert$	Global single key pool size
$\lvert XP\rvert$	Global XORed key pool size
m	Keyring size
$\lvert sk\rvert$	Single keyring size
$\lvert xk\rvert$	XORed keyring size
N	Set of all nodes
NL_i	List of neighbors of node i
$SecureNL_i$	List of neighbors of node i with at least one shared key
$A \xrightarrow{key} B$	A sends key to B in a secure way

Workflow of abovementioned phases are shown in Figure 1. Key predistribution phase and shared-key discovery phase are split into two parts as single key and XORed key subphases since the key types involved are different.

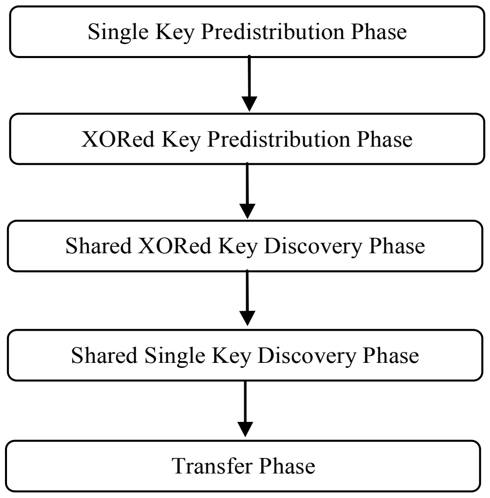
Single Key Predistribution Phase
XORed Key Predistribution Phase
Shared XORed Key Discovery Phase
Shared Single Key Discovery Phase
Transfer Phase

Figure 1. Work flow of the proposed scheme

B. Key Predistribution Phase

Key predistribution phase starts with offline generation of a large single key pool of $\lvert SP\rvert$ keys. Each single key is assigned a unique key identifier. In addition to that, we use a virtual XORed key pool, which is composed of keys obtained by XORing distinct single keys of the single key pool. The total number of XORed keys in the XORed key pool is the total number of binary combinations of all single keys in the single key pool. Thus, the size of the XORed key pool is the following:

$$|XP| = |SP| \times (|SP| - 1)/2 \qquad (1)$$

Virtuality of XORed key pool comes from the fact that it is not actually generated; whenever a key from the XORed key pool is to be drawn, two random single keys of the single key pool are XORed. In this way, such a large key pool can easily be managed. For the sake of consistency in terminology, we will not use the term "virtual" while referring XORed key pool in the rest of the paper.

Key identifier of an XORed key x is defined as $i \| j$, where i and j are key identifiers of the corresponding single keys from which x is derived.

Next step is to generate the keyrings of the sensor nodes. To do so firstly $|sk|$ distinct single keys from SP are drawn uniformly random without replacement for each node in order to form single keyrings. After this process, XORed keyrings of the sensor nodes are established by selecting $|xk|$ XORed keys from XORed key pool XP for each node uniformly random without replacement. If one of the single key operands of a selected XORed key is already in the single keyring of the node, selected XORed key is ignored and a new one is drawn. Thus, the node can derive as many new XORed keys as possible from available single keys in its keyring. We will explain the reason why the nodes need to produce XORed keys from the single keys in their keyrings in Section 2.C. Total keyring memory size of the sensor nodes is the sum of the single keyring size and the XORed keyring size, as given in Equation 2.

$$m = |sk| + |xk| \qquad (2)$$

C. Shared-Key Discovery Phase

After key predistribution phase is completed, sensor nodes with preloaded keyrings in memory are deployed over the field. After settlement, nodes scan their neighborhood independently. Each sensor node broadcasts node identity and the identifiers of single and XORed keys that they have. At the end of the transmission of own key identifiers and reception of respective keyring contents in the neighborhood, nodes try to find common XORed keys with their neighbor nodes in the first step. If no common XORed key with a neighbor is found, the nodes try to find a common single key.

The reason behind searching for XORed keys initially is that resiliency that an XORed key provides is more than a single key does. An attacker has to obtain two operands of an XORed key in order to capture a link secured by that XORed key, while it is enough to capture just one single key in single link key case. In our scheme, using XORed key as a link key is promoted; if there is a chance to use XORed key or single key to secure a link, XORed key usage is preferred. However, we should admit that it is less probable to find out a shared XORed key in keyrings of neighboring nodes, because the size of XORed key pool is much larger than the size of single key pool.

If both of these attempts fail, nodes try to derive new XORed keys using single keys pre-loaded in their memory. A node XORs two single keys in its single keyring to match an XORed key in the XORed keyring of its neighbor. For example, suppose a and b are two neighbors with no secure link yet established. If node a has single keys t and u in its keyring such that $x = t \oplus u$, and x is in b's XORed keyring, then node a XORs t and u to obtain x and the nodes a and b can use x to establish a secure link. If neighboring nodes still cannot find a common key, they start key transfer phase which is described in the next subsection.

The algorithm of shared-key discovery phase is depicted in Figure 2.

```
for all $a \in N$
  for all $b \in NL_a$
    for all $x \in xk_b$
      if $x \in xk_a$
        $SecureNL_a = SecureNL_a \cup \{b\}$
        $SecureNL_b = SecureNL_b \cup \{a\}$
      end
      if $b \notin SecureNL_a$
        for all $s \in sk_b$
          if $s \in sk_a$
            $SecureNL_a = SecureNL_a \cup \{b\}$
            $SecureNL_b = SecureNL_b \cup \{a\}$
          end
for all $a \in N$
  for all $b \in NL_a$
    if $b \notin SecureNL_a$
      for all $t \in sk_a$
        for all $u \in sk_a$
          if $t \oplus u \in xk_b$
            $SecureNL_a = SecureNL_a \cup \{b\}$
            $SecureNL_b = SecureNL_b \cup \{a\}$
          end
```

Figure 2. Pseudo-code of shared-key discovery phase.

D. Key Transfer Phase

Main contribution of our scheme is in the key transfer phase. When shared-key discovery phase ends, there would possibly be some neighboring nodes that failed on finding common key and some succeeded on setting up secure communication links. Successfully established secure links are used to help creating secure links between insecure neighbor nodes in the key transfer phase.

If two neighboring nodes a and b cannot establish a secure link in shared-key discovery phase, they apply three methods that will help them to agree on a key.

In the first method, a node searches its own single keyring and the single keyrings of its neighbors with which it has a secure communication link (from now on mentioned as *direct secure neighbor*). If there are two single keys, one in its single keyring and one in its secure neighbor's single keyring, such that XOR of these single keys matches an XORed key in its unsecure neighbor's XORed keyring, it transfers the single key from its direct secure neighbor and establishes secure communication with its unsecure neighbor using XOR of those single keys.

For example, suppose *b* has the XORed key $x = s \oplus t$ in its XORed keyring, and one of its operands *t* is found in node *a*, the other operand *s* is found in direct secure neighbor node *c* of *a*. In this method, *a* transfers the single key *s* from node *c* and XORs it with its single key *t*. In this way, it obtains a new XORed key *x* which *a* shares with *b*. For this operation, node *a* needs to know the identifiers of the single keys in direct secure neighbors' single keyrings. These key identifiers are already supplied in shared-key discovery phase, so there is no need to resend them in this phase. Since there is no extra messages sent through nodes, this method (and also the upcoming methods) has minimal effect on communication cost. Algorithm for this method is given in Figure 3.

```
Method 1 ($a$, $b$):
for all $c \in SecureNL_a$
  for all $s \in sk_c$
    for all $t \in sk_a$
      if $s \oplus t \in xk_b$
        $c \xrightarrow{s} a$
        $sk_a = sk_a \cup \{s\}$
        $SecureNL_a = SecureNL_a \cup \{b\}$
        $SecureNL_b = SecureNL_b \cup \{a\}$
      end
```

Figure 3. Method 1: nodes try to transfer single keys from their direct secure neighbors and XOR them with existing single keys in their keyring to produce an XORed key that is found in the keyrings of their neighbors.

If the first method does not work, the nodes use the second method. In method 2, a node transfers an XORed key from its direct secure neighbor, and XOR the transferred key with another XORed key in its keyring to derive a new XORed key that can be used for securing a communication link. In order to derive a new XORed key, transferred XORed key and existing XORed key in the keyring should have one common operand; when two XORed keys are XORed, this common operand cancels out and new XORed key becomes XOR of two single keys. For instance, suppose nodes *a* and *b* are two neighboring nodes with no shared key and node *c* is *a*'s direct secure neighbor. Node *c* has an XORed key *x* and *a* has an XORed key *y* in their XORed keyrings such that *x* and *y* have a common single key operand. Furthermore, XOR of *x* and *y* produces another XORed key which is found in the XORed keyring of node *b*. In this setting, *a* transfers *x* from *c* and creates XOR of *x* and *y*. Afterwards nodes *a* and *b* establish secure link using this XORed key. Algorithm for method 2 is given in Figure 4.

```
Method 2 ($a$, $b$):
for all $c \in SecureNL_a$
  for all $x \in xk_c$
    for all $y \in xk_a$
      if $x \oplus y \in xk_b$
        $c \xrightarrow{x} a$
        $xk_a = xk_a \cup \{x\}$
        $SecureNL_a = SecureNL_a \cup \{b\}$
        $SecureNL_b = SecureNL_b \cup \{a\}$
      end
```

Figure 4. Method 2: nodes try to transfer XORed keys from their direct secure neighbors and XOR them with existing XORed keys in their keyring also to produce an XORed key that is found in the keyrings of their neighbors.

If the previous methods do not help to establish a secure link, as the last chance, the nodes try method 3. In this method, a node searches for two single keys from distinct direct secure neighbors, transfers and XORs them to obtain a new XORed key. If this new XORed key is also found in the XORed keyring of the neighbor node with which the node wants to establish a secure link, then this secure link is created. Transferred XORed keys should have one common operand for the reasons mentioned above. For example, suppose nodes *a* and *b* are two neighboring nodes that do not have a shared key. Moreover, nodes *c* and *d* are direct secure neighbors of node *a*. Node *c* has a single key *s* and node *d* has a single key *t* such that their XOR yields a new XORed key which also exists in the node *b*'s XORed keyring. In this setting, node *a* requests to transfer single keys *s* and *t* from its direct secure neighbors *c* and *d*. Upon receipt, node *a* XORs these two single keys and produces a new XORed key, which also exists in the XORed keyring of node *b*. Then, they establish a secure link. Algorithm of this method is given Figure 5.

If a node runs these methods in the transfer phase and establishes new secure communication links with neighbor nodes, it adds these nodes into its direct neighbors list and re-runs these methods for the purpose of deriving new common keys with the neighbors that do not share one. This provides further optimization to the transfer phase, and incrementally improves local connectivity.

```
Method 3 ($a$, $b$):
for all $c \in SecureNL_a$
   for all $d \in SecureNL_a$
      for all $s \in sk_c$
         for all $t \in sk_d$
            if $s \oplus t \in xk_b$
               $c \xrightarrow{s} a$
               $sk_a = sk_a \cup \{s\}$
               $d \xrightarrow{t} a$
               $sk_a = sk_a \cup \{t\}$
               $SecureNL_a = SecureNL_a \cup \{b\}$
               $SecureNL_b = SecureNL_b \cup \{a\}$
         end
```

Figure 5. Method 3: nodes try to transfer two single keys from two distinct direct secure neighbors and XOR them in order to produce an XORed key that is found in the keyrings of their neighbors.

III. Performance Evaluation

For the performance evaluation of the proposed scheme, we conduct simulations. The simulations are done in MATLAB environment in a 2.4 GHz Intel Core 2 Quad desktop PC running 32-bit Windows Vista and 64-bit Linux operating systems. In simulations, 10,000 nodes are randomly placed using uniform distribution in a square field of size 1000 m × 1000 m. The radius of the communication range of a node is 40 m. Single key pool size is 10,000 keys. Thus, XORed key pool size automatically becomes 49,995,000 keys, although this key pool is a virtual one and not actually generated as discussed in Section II.D.

Local connectivity is the probability that any two sensor nodes share at least one key. In Figure 6, the straight line indicates the local connectivity of basic scheme [2] with 100 single keys. This line serves as a performance threshold to our scheme. For our scheme, single keyring sizes increase by 10 keys, while XORed keyring sizes decrease by 10 keys such that the sum of the sizes of two keyrings is 100 keys. When single keyring size is small, local connectivity of our scheme is lower than the basic scheme. The reason is that probability of sharing at least one XORed key is too small since the XORed key pool is huge. As the single keyring size increases and XORed keyring size reduces, the local connectivity of our scheme increases and eventually exceeds the basic scheme threshold. The highest connectivity of our scheme occurs when 70 single keys and 30 XORed keys are used. At this point, local connectivity value of our scheme is 0.97; whereas the local connectivity of the basic scheme with 100 keys is 0.63. This result shows that our scheme provides almost full connectivity when the keyring size ratio is selected appropriately.

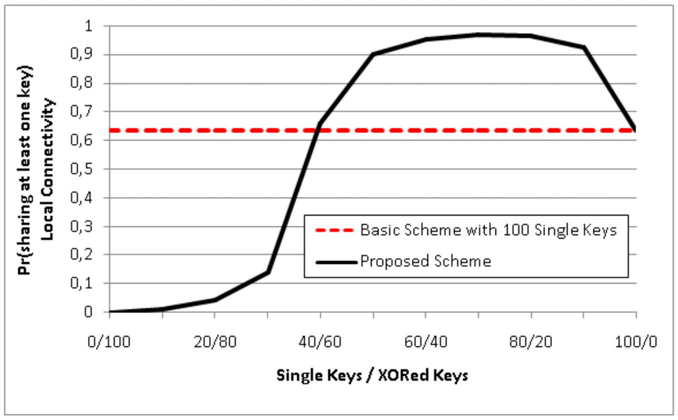

Figure 6. Local connectivity of our scheme compared to basic scheme

Figure 7 shows the resilience of our scheme compared to basic scheme. For our scheme we consider the optimal case, in which 70 single keys and 30 XORed keys are used in the key rings. Resilience is the fraction of communication links compromised over all links in case of a node capture attack. Thus, lower values indicate more resilient network. In a node capture attack, the attacker captures nodes at random locations of the network. When a node is captured, the attacker learns the keyring content of the node and these keys are used to compromise secure links of other innocent nodes in the network. We analyze the change of resiliency with respect to different amount of captured nodes. As can be seen from Figure 7, our scheme is slightly more resilient at the beginning of the attack. However, as the attack continues, basic scheme becomes slightly more resilient starting from the point where 60% of the links are compromised. Nevertheless when our scheme is compared to the basic scheme, the difference in ratio of links compromised is always lower than 5%. This means our scheme provides better connectivity with a marginal cost of resilience.

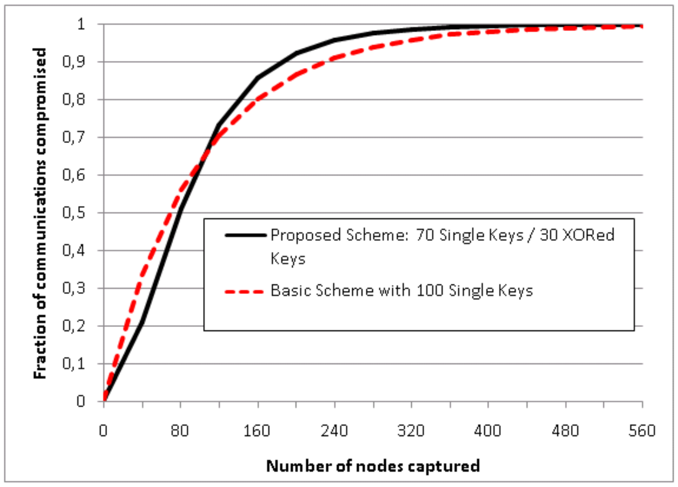

Figure 7. Resilience of our scheme compared to basic scheme.

In order to understand the extra communications that our transfer phase incurs, we analyze a metric called *transfer cost*. This *transfer cost* metric is defined as the average number of keys exchanged between sensor nodes in order to establish a link key in the transfer phase. Transferred keys are usually the missing operands of XORed link keys. In method 1 and method 2, one key (single key for method 1 and XORed key for method 2) is transferred from direct secure neighbors in order to produce a new XORed link key. Meanwhile in method 3, two single keys are transferred from two distinct direct secure neighbors. According to our simulation results, the total number of keys transferred between secure direct neighbors is 269,210 for the optimal case where 70 single keys and 30 XORed keys are used in the keyrings. In return, 140,550 link keys are established. These results imply a transfer cost of 1.92, which means on the average 1.92 keys are transferred for each new established links during transfer phase.

Here one should notice that the keys are transferred whenever needed to establish a new secure link. In other words, they are transferred deterministically, not probabilistically. One advantage of this approach is that the nodes are not alerted to help other nodes unnecessarily. The helper nodes are pinpointed by the help requesting node using the information that it already has. Only the helper node is requested to send a particular key. The request messages are not flooded in the network and extra traffic is avoided.

IV. Related Work

Key predistribution is a hot topic that constitutes the basis of security in wireless sensor networks. Many security mechanisms such as encryption and authentication can be provided by accessing to shared keys. Several techniques are previously proposed to address this issue. Extensive surveys about key distribution in sensor networks are given by Camtepe and Yener [4], Zhang and Varadharajan [5], Zhou et al. [6], and Xiao et al. [7].

Eschenauer and Gligor's basic scheme [2] is taken as a framework for many techniques using probabilistic key sharing for key management, such as [3, 8, 9, 10]. These studies compared themselves with the basic scheme as we did in this paper.

Chan et al. [3] proposed a probabilistic key sharing scheme similar to basic scheme. The main difference is that two nodes need at least q shared keys to establish a secure communication. It is known that using $q > 1$ keys instead of 1 increases resilience of the network since attacker needs to capture more than 1 key to compromise a link. q-composite scheme has also some similarities with our scheme, because in both schemes, nodes require more than 1 key to secure their communication. The difference between our scheme and [3] is that we store composite keys in the keyrings and we employ a transfer phase to improve the local connectivity of the network.

V. Conclusions

We proposed a key predistribution scheme that achieves high connectivity values as compared to basic scheme with same keyring sizes. In our scheme, a mixture of regular (a.k.a. single) and XORed keys are used in keyrings. Our scheme uses XORed keys for link keys, wherever possible. This plays an important role to keep the resilience of the network under control, since an attacker needs to compromise two single keys instead of one. Our scheme uses a novel phase called *transfer phase*, in which the nodes transfer single or XORed keys directly from their secure neighbors. Our scheme achieves high local connectivity with carefully selected values for single and XORed keyring sizes. In our proposed scheme with 70 single keyring size and 30 XORed keyring size, there is an increase in local connectivity more than 50% as compared to basic scheme [2]. More importantly, this improvement in local connectivity is not paid with decreased resiliency. The difference between resilience of our proposed scheme and basic scheme does not exceed 5%. Moreover, communication cost is minimized since keyring indices of neighbor nodes are already sent out in shared-key discovery phase. Nodes transfer only required keys from their neighbors in a deterministic way.

References

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

[2] Eschenauer, L. and Gligor, V. D., "A key-management scheme for distributed sensor networks," in proceedings of the 9th ACM Conference on Computer and Communications Security. V. Atluri, Ed. CCS '02. ACM, New York, NY, 41-47, 2002.

[3] Chan, H., Perrig, A., Song, D., "Random key predistribution schemes for sensor networks," sp, p. 197, 2003 IEEE Symposium on Security and Privacy, 2003.

[4] Çamtepe, S. A. and Yener, B., "Key Distribution Mechanisms for Wireless Sensor Networks: a Survey," Technical Report TR-05-07 Rensselaer Polytechnic Institute, Computer Science Department, March 2005.

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

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

[7] Xiao, Y., Rayi, V. K., Sun, B., Du, X., Hu, F., and Galloway, M., "A survey of key management schemes in wireless sensor networks," Comput. Commun. 30, 11-12 (Sep. 2007), 2314-2341.

[8] Du, W., Deng, J., Han, Y. S., Varshney, P. K., Katz, J., and Khalili, A., "A pairwise key pre-distribution scheme for wireless sensor networks," ACM Trans. Inf. Syst. Secur. 8, 2 (May. 2005), 228-258, 2005.

[9] Du, W., Deng, J., Han, Y. S., Varshney, P. K., "A key management scheme for sensor networks using deployment knowledge," IEEE Transactions on Dependable and Secure Computing, vol. 03, no. 1, pp. 62-77, Jan-Mar, 2006.

[10] Liu, D., Ning, P., and Li, R., "Establishing pairwise keys in distributed sensor networks," ACM Trans. Inf. Syst. Secur. 8, 1 (Feb. 2005), 41-77, 2005.