Network Decoupling: A Methodology for Secure Communications in Wireless Sensor Networks

Wenjun Gu, Xiaole Bai, Sriram Chellappan and Dong Xuan

Abstract

Many wireless sensor network (WSN) applications demand secure communications. The random key pre-distribution (RKP) protocol has been well accepted in achieving secure communications in WSNs. A host of key management protocols have been proposed for WSNs based on the RKP protocol. However, due to its randomness in key distribution and strong constraint in key path construction, the RKP based protocols can only be applied in highly dense networks, which are not always feasible in practice. In this paper, we propose a methodology called *network decoupling* to address this problem. With this methodology, a wireless sensor network is decoupled into a *logical key-sharing network* and a *physical neighborhood network*, which significantly releases the constraint in key path construction of the RKP protocol. We design a new key management protocol (called RKP-DE) as well as a set of link and path dependency elimination rules in decoupled sensor networks. Our analytical and simulation data demonstrate the performance enhancement of our solutions from the perspective of connectivity, resilience and overhead, and its applicability in non-highly dense sensor networks.

Index Terms

Wireless Sensor Networks, Random Key Pre-distribution, Network Decoupling.

I. Introduction

In this paper, we address the issue of providing secure communications in Wireless Sensor Networks (WSNs). WSNs are gaining wide acceptance today with a host of new applications being realized involving many tiny wireless sensors performing sensing and communication tasks. Many of these applications are

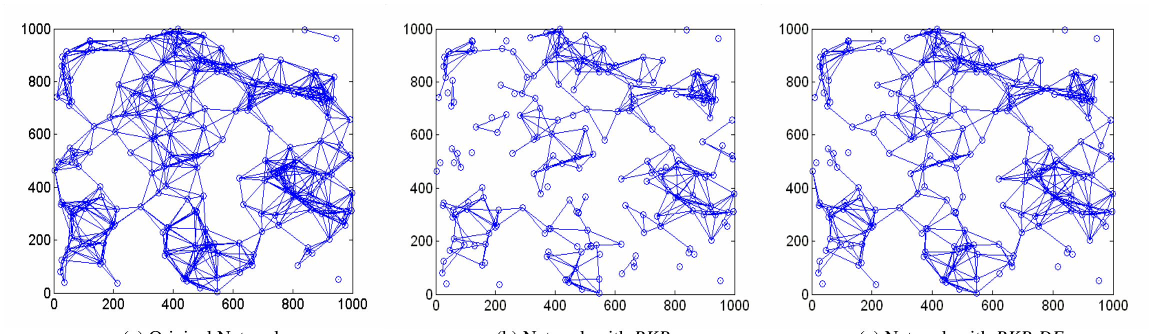
Wenjun Gu, Xiaole Bai, Sriram Chellappan and Dong Xuan are with The Department of Computer Science and Engineering, The Ohio State University, Columbus, OH 43210, U.S.A. E-mail: {gu, baixia, chellapp, xuan}@cse.ohio-state.edu. An earlier version was accepted by the fourteenth IEEE International Workshop on Quality of Service (IWQoS) 2006.

This work was partially supported by NSF under grants No. ACI-0329155 and CCF-0546668. Any opinions, findings and conclusions or recommendations expressed in this material are those of the authors and do not necessarily reflect the views of NSF.

in hostile/vulnerable environments, and their success is contingent on preventing the WSNs information from being accessible to external malicious attackers.

Motivation: In order to provide secure communications in WSNs, secret keys need to be established between communicating sensors. A host of key distribution techniques have been proposed to achieve secure communications in traditional wired networks and wireless ad hoc networks. However, they cannot be applied in WSNs due to the unique characteristics of WSNs like hostile zone deployment, ease of node capture, physical constraints in energy and memory, etc. For instance, the traditional public key cryptography [1], [2] is too energy consuming to be carried out by energy constrained sensors. The key distribution center based scheme [3] is centralized and not scalable when network size increases. Using a single master key for all communications is too vulnerable, while establishing a unique pair-wise key for each pair of nodes requires too much memory, both of which are unsuitable in WSNs.

In order to address the above concerns, the *Random Key Pre-distribution* (RKP) protocol was first proposed in [4]. There are two stages in this protocol. In the first stage, each sensor is initially pre-distributed with a small number of k distinct keys randomly chosen from a larger key pool of K keys, and then the sensors are deployed in the network. In the second stage, using the pre-distributed keys, two physically neighboring sensors (i.e, sensors within communication range of each other) will attempt to establish a pair-wise key for secure communications between themselves. If two physically neighboring sensors already share a pre-distributed key, they can *directly* establish a pair-wise key between themselves. Alternatively, two physically neighboring sensors can establish a pair-wise key *indirectly* through a key path traversing through other sensors, with the constraint that any two successive sensors on this path are physical neighbors and share at least one pre-distributed key. For the rest of the paper, physical neighbors that have established a pair-wise key are called *secure neighbors*. The established pair-wise key will be used for all further secure communications between the respective sensors. The RKP protocol has been well accepted in WSNs. It is particularly suited for large-scale WSNs, hostile zone deployment etc., where manually placing sensors is not possible, and sensors have to be sprayed from a vehicle, air-dropped in the field etc. In such scenarios, it is not possible to deterministically pre-distribute pair-wise keys to neighboring sensors, as the neighborhood of sensor nodes cannot be determined at deployment time. Other benefits of the RKP protocol are its distributed nature of execution, simplicity, energy efficiency and scalability. As such, it has served as a foundation for a host of key management protocols in WSNs



(a) Original Network (b) Network with *RKP* (c) Network with *RKP-DE*

Fig. 1. Average secure node degree comparison between RKP and $RKP\text{-}DE$. Our $RKP\text{-}DE$ achieves 40% improvement in average secure node degree. The network is of size $1000m$ * $1000m$, where 200 nodes are deployed uniformly at random. All nodes have the same communication range ($133m$) and the average physical node degree is 9.71. We set $K = 10000$ and $k = 50$.

that aim towards improving the probability of pair-wise key establishment, enhancing the resilience to node capture, or decreasing storage overhead [5], [6], [7], [8], [9], [10], etc.

However, all the RKP based protocols have an inherent limitation. The performance of RKP is satisfactory only in highly dense sensor networks, where the average number of physical neighbors per node (i.e., average physical node degree) is large ($>= 20$) [4], [5], [6]. As we know, such a high density is not always feasible in practice due to high deployment cost, increased number of collisions, low per node throughput etc. In fact, due to the randomness in key distribution and strong constraint in key path construction, it often happens that the secure node degree (i.e., number of secure neighbors for a sensor) of the RKP based protocol is very low in non-highly dense networks. Consequently, the networks will have low secure connectivity and are very likely to be partitioned as illustrated in Fig. 1. The original sensor network is shown in Fig. 1 (a). There is an edge between two nodes if they are physical neighbors. The average physical node degree is 9.71. The corresponding secure network as a result of executing the RKP protocol is shown in Fig. 1 (b) where an edge exists between two nodes if they are secure neighbors. In this example, we limit the number of intermediate nodes on a key path to be one [1]. The average secure node degree in Fig. 1 (b) is only 4.06. It is much smaller compared to the average physical node degree. As can be seen, the network in Fig. 1 (b) is partitioned into many components. Two nodes cannot communicate securely if they reside in different components.

Our Contributions: In this paper, we aim to solve the above problem. Our contributions are four-fold.

- *Network Decoupling:* We propose a methodology called *network decoupling* for secure communications in wireless sensor networks. In random key pre-distributed sensor networks, there exist two types

[1]The general case of multiple intermediate nodes on a key path will be discussed later.

of relationship between any two nodes. One is logical (sharing pre-distributed keys), and the other is physical (within communication range). In network decoupling, we decouple these two relationships in the sensor network. As such, for any two logically connected nodes, the corresponding physical path found between them need not satisfy the logical constraint. Similarly, for any two physically connected nodes, the corresponding logical path found between them need not satisfy the physical constraint. This flexibility offered by decoupling greatly enables finding more logical and physical key paths in the network, thereby enhancing the chances of pair-wise key establishment between physical neighbors in the network.

- *Protocol Design:* We design a new key management protocol for secure neighbor establishment between physical neighbors in decoupled sensor networks. We call our protocol as RKP-DE protocol. In this protocol, logical key paths are constructed based on pre-distributed key sharing information, and then corresponding physical key paths are constructed based on node neighboring information.
- *Dependency Elimination:* Our third contribution is proposing novel dependency elimination rules in our RKP-DE protocol to detect and eliminate key dependencies at link and path level without compromising existing resilience. In pair-wise key establishment, when multiple key paths are constructed, there is a possibility of some links (or paths) being dependent on other links (or paths). Such dependencies introduce unnecessary overhead in terms of communication and computation, which can be eliminated using our proposed rules. We point out that such dependencies exist in all existing RKP based protocols, where multiple key paths are used [5], [6], [9]. Our dependency elimination rules can also be applied to them to minimize their overhead.
- *Analysis:* Our final contribution is a formal analysis of our proposed RKP-DE protocol and its comparison with the traditional RKP protocol from the perspective of average secure node degree and overhead. Our analysis demonstrates that the improvement in average secure node degree in our RKP-DE protocol is up to 45% compared to the RKP protocol. Such an improvement significantly enhances the quality of secure communications in the network. In this paper, we also define and analyze a new metric called $stretch\ factor$ to quantify protocol overhead. Formally, the stretch factor is the average number of physical hops on the key path between two secure neighbors. Clearly, the stretch factor of our RKP-DE protocol is larger than that of the RKP protocol, since more key

paths can be established in our protocol. However, our analysis demonstrates that the increase in overhead is mild (around 20% only), further demonstrating the benefits of network decoupling.

To illustrate performance improvement of our RKP-DE protocol, we show the corresponding secure network as a result of executing our RKP-DE protocol in Fig. 1 (c). As is the case in Fig. 1 (b), there is only one intermediate node on each key path in Fig. 1 (c). The average secure node degree in Fig. 1 (c) has now increased to 5.68, a 40% improvement over that in Fig. 1 (b). Extensive analysis and simulations conducted in this paper further validates this fact.

We wish to point out that the methodology of decoupling is not new in networking. The flexibilities offered by decoupling have been exploited before. A fundamental example is layering of the Internet architecture, the flexibility of which has enabled several evolvements at each layer separately enhancing the overall Internet. Across some respects, we can consider our approach as decoupling the logical layer from the physical layer in the sensor network. Apart from the Internet architecture, decoupling has been used in other areas too. The advantages of exploiting flexibilities offered by decoupling the policy from mechanisms in Internet routing are demonstrated in [11]. In [12], an approach is proposed that decouples control from data in TCP congestion control. Another work is [13], where path naming is decoupled from actual paths to enable better data delivery in dense WSNs. However, to the best of our knowledge, our work is the first one that applies this methodology for secure communications in sensor networks.

Our paper is organized as follows. We discuss random key pre-distribution protocol in Section II. The methodology of network decoupling is introduced in Section III, and our secure neighbor establishment protocol RKP-DE is detailed in Section IV. We present performance evaluations in Section V, and discuss related work in Section VI. We finally conclude our paper in Section VII.

II. The Random Key Pre-distribution Protocol in Wireless Sensor Networks

In this section, we give a brief overview of the random key pre-distribution (RKP) protocol in WSNs. As discussed before, there are two stages in this protocol [4]. In the first stage, each sensor is initially pre-distributed with a small number of k distinct keys randomly chosen from a larger key pool of K keys, followed by the deployment of the sensors. After deployment, the second stage occurs where two physically neighboring sensors will attempt to establish a pair-wise key for secure communication between themselves using the initially pre-distributed keys. We discuss both stages below.

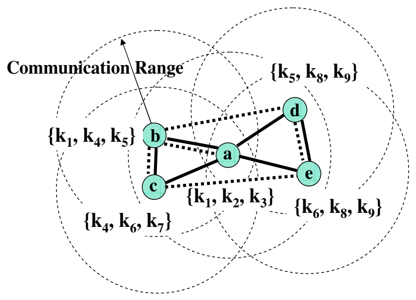


Fig. 2. Pair-wise key establishment in RKP protocol.

We denote the first stage of the RKP protocol as *key pre-distribution*, and the second stage as *secure neighbor establishment*. In the *key pre-distribution* stage, each node is pre-distributed with k distinct keys randomly chosen from a larger key pool of K keys, and then the nodes are deployed randomly in the network. The set of keys pre-distributed in node a is called the *key chain* of node a. Consider an example in Fig. 2, where five nodes are deployed as shown, and where $k = 3$ and $K = 9$. The keys pre-distributed in each node are shown beside the corresponding node in braces. A solid line exists between two nodes if they are physical neighbors (within communication range), and a dashed line exists between two nodes if they are logical neighbors (share at least one key).

Once nodes are deployed, the *secure neighbor establishment* stage follows. For ease of elucidation, the second stage is split into two sequential phases namely *neighbor discovery* and *pair-wise key establishment*. In the *neighbor discovery* phase, each node sends a message to its physical neighbors, containing its node ID and the key IDs of its pre-distributed keys. Key IDs are transmitted instead of keys themselves in order to protect against the attacker from eavesdropping and discovering the actual keys transmitted. In *pair-wise key establishment* phase, two physically neighboring nodes attempt to establish a pair-wise key via existing secure communication between them. Here the secure communication is defined as the communication between two nodes where all messages transmitted (possibly via multi-hops) are encrypted. There are two cases for pair-wise key establishment. In the first case, two physically neighboring nodes share at least one pre-distributed key. In this case, the nodes establish a pair-wise key between them directly. In Fig. 2, nodes a and b share key k_1. Hence node a can send its randomly generated pair-wise key to node b by encrypting it with the shared key k_1. In the second case, two physically neighboring nodes do not share pre-distributed key, but they may still be able to establish a pair-wise key via the help of other nodes called *proxies*. Here, a key path is attempted to be constructed comprising of one or multiple proxies, where any two successive nodes on the key path are physical neighbors and share at least one pre-distributed key, and the pair-wise key is encrypted/decrypted in *each* hop till it reaches the destination. In Fig. 2, node a does not share any key with its physical neighbor node c, however node b can be the proxy between nodes a and c. The pair-wise key between nodes a and c is first generated by node a, and sent to node b encrypted using key k_1. Node b will decrypt the pair-wise key, encrypt it using key k_4 and send it to node c. Finally node c decrypts the pair-wise key, and uses it to encrypt/decrypt future direct communication with node a. As we can see, on the established key paths, each hop needs to satisfy both the logical (sharing pre-distributed key) constraint and the physical (within communication range) constraint. In other words, both constraints are coupled together. The consequence of this constraint is that the RKP protocol is applicable only in highly dense sensor networks. As discussed in the next section, we make this protocol applicable in non-highly dense sensor networks by decoupling the logical and physical constraints in the sensor network during key path construction.

The standard attack model used in analyzing secure communications is one where the attacker does not attempt to disrupt network operation; rather it attempts to decipher as much information as possible from sensor communications [4], [5], [7]. As such, the attacker will typically launch two types of attacks: *link monitor attack* and *node capture attack*. The attacker has the ability to monitor and record all the wireless communication in the network *immediately* after node deployment (i.e., link monitor attack). Besides, the attacker is assumed to be able to physically capture a limited number of nodes in the network (i.e., node capture attack). Once a node is captured, its pre-distributed keys and pair-wise keys are all disclosed to the attacker. By combining the pre-distributed keys disclosed and the messages recorded, the attacker will be able to infer the pair-wise keys between some nodes, even if the nodes themselves are not captured. For instance, if node a sends the pair-wise key between nodes a and c to node c via node b, then this pair-wise key is inferred if either key k_1 or k_4 is disclosed (by capturing some other nodes) even though nodes a and c may not have been captured. We denote the pair-wise keys disclosed by the attacker as $compromised$, as is the corresponding secure communications between those neighboring nodes.

To evaluate the performance of RKP protocol, two types of metrics are considered. The first is *connectivity*, which includes *local connectivity* and *global connectivity*. Local connectivity is defined as the probability that two physically neighboring nodes are able to establish a pair-wise key between them. Global connectivity is defined as either the probability that the whole secure network (e.g., Fig. 1 (b) or (c)) is connected, or the percent of nodes in the largest connected component of the secure network. The other performance metric is *resilience*, which is defined as the probability that a pair-wise key between two nodes is not compromised given that those two nodes are not captured. The overall goal clearly is to make connectivity and resilience as high as possible.

III. Network Decoupling in Random Key Pre-distributed Sensor Networks

A. Network Decoupling

In random key pre-distributed sensor networks, there exist two types of relationship between any two nodes. One is logical (sharing pre-distributed keys), and the other is physical (within communication range). We can separate these two types of relationship by decoupling a random key pre-distributed sensor network into two graphs: a logical one and a physical one. Two nodes in the logical graph have an edge between them if they share at least one pre-distributed key. Similarly two nodes in the physical graph have an edge between them if they are within communication range of each other. For the example in Fig. 3 (a), its decoupled logical and physical graphs are shown in Fig. 3 (b) and (c) respectively. Detailed description on how nodes construct these graphs is presented in Section IV.B.

Recall that secure communication is defined as the communication between two nodes where all messages transmitted (possibly via multi-hops) are encrypted. Now we will show how network decoupling helps achieve secure communication. There are two cases possible, where two nodes in the network can communicate securely. The first case is where the two nodes share at least one pre-distributed key (i.e., they are *directly* connected in the logical graph) and the nodes are also connected (via one or more hops) in the physical graph. In this case, the source node can encrypt the messages using the shared key, and each intermediate node in the physical graph can simply forward the messages towards the destination, which will decrypt the messages using the shared key. Such intermediate nodes in the physical graph are called as *physical intermediate nodes*. An example of the first case is nodes b and d in Fig. 3 (a), where node a is the physical intermediate node between them. The second case is one where the two nodes do not share a key (i.e, they are *not directly* connected in the logical graph), but are connected *indirectly* in the logical graph with multiple logical hops, and the two nodes for each logical hop are connected (directly or indirectly) in the physical graph. In this case, encryption occurs at each intermediate node in the logical graph, while each intermediate node in the physical graph simply forwards the messages.

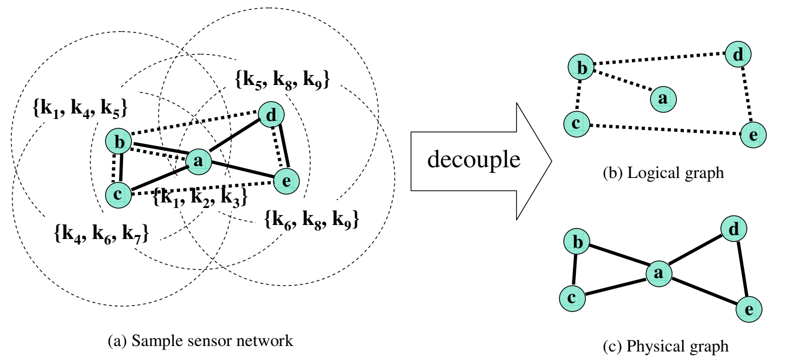


Fig. 3. Decouple a sensor network into a logical graph and a physical graph.

Such intermediate nodes in the logical graph are called as *logical intermediate nodes*. An example of the second case is nodes a and d in Fig. 3 (a), where node b is the logical intermediate node between them. Note that if a physical (or logical) intermediate node also satisfies the logical (or physical) constraint, such node becomes a $proxy$ as defined in the RKP protocol (in the coupled network).

We point out that to apply decoupling, each sensor needs to know both the key sharing and physical neighborhood information among its physical neighbors. As an example, the constructed logical and physical graphs using one hop neighborhood information for node a are shown in Figs. 3 (b) and (c) respectively. Each node obtains information to construct these graphs in a purely localized and distributed way. For the rest of the paper, we assume each node obtains only one hop neighborhood information. Note that the chances of establishing pair-wise keys will increase with information on more than one hop. Our protocol can be easily applied in such cases, although communication overhead will be increased.

B. Analysis

In this section, we will demonstrate the benefits of network decoupling quantitatively by analysis. Specifically, we will derive the probability for the case where two physically neighboring nodes are able to communicate securely. As a matter of fact, this probability is also the probability that two physically neighboring nodes are able to establish a pair-wise key via secure communication. In this section, we first present the analysis for the simple case where one proxy or logical intermediate node is used on a key path. We subsequently present the results for the general case where multiple proxies or logical intermediate nodes are used on a key path. Detailed analysis for the general case is given in the Appendix.

1) Case1: One proxy or logical intermediate node on a key path: For two physically neighboring nodes to communicate securely using one proxy or logical intermediate node and with only one hop neighborhood information, there exist three possible situations: *(1)* The two nodes share key (directly connected in the logical graph), such as nodes a and b in Fig. 3 (a). Clearly, they can achieve secure communication directly. We denote the probability that this situation happens as P_1. *(2)* The two nodes do not share key (not directly connected in the logical graph), but they have a common physical neighbor that shares key with both of them. In Fig. 3 (a), nodes a and c do not share key, but have a common physical neighbor node b that shares key with both of them. Secure communication between nodes a and c can now be achieved via the help of node b acting as a proxy or logical intermediate node. We denote the probability that this situation happens as P_2. *(3)* The two nodes do not share key (not directly connected in the logical graph), and they cannot find a proxy or logical intermediate node satisfying the second situation above. But there exists a logical intermediate node that shares key with both of those two nodes, and is a physical neighbor of only one of them. In Fig. 3 (a), nodes a and d do not share key and there is no node satisfying the second situation. But node b shares key with both nodes a and d, and node b is a physical neighbor of only node a. Secure communication between nodes a and d can be achieved via the help of node b acting as a logical intermediate node. We denote the probability that this situation happens as P_3.

Let us define *coupled network* as the network in which the logical and physical constraints are always satisfied simultaneously for each hop on a secure communication path. Therefore two physically neighboring nodes in a coupled network can achieve secure communication with at most one proxy if and only if either of the first two situations happens. In the third situation, secure communication is not possible in a coupled network. On the other hand, in a decoupled network, secure communication with at most one logical intermediate node is possible if any of the three situations happens. We denote the probability that two physically neighboring nodes can achieve secure communication with at most one proxy in the coupled network by $P^{(1)}_{couple}$, and denote the probability that two physically neighboring nodes can achieve secure communication with at most one logical intermediate node in the decoupled network as $P^{(1)}_{decouple}$. Since the above three situations are disjoint, the expressions of $P^{(1)}_{couple}$ and $P^{(1)}_{decouple}$ are simply given by,

$$P^{(1)}_{couple} = P_1 + P_2, \tag{1}$$

$$P^{(1)}_{decouple} = P_1 + P_2 + P_3. \tag{2}$$

Clearly, $P^{(1)}_{decouple} > P^{(1)}_{couple}$. This shows that network decoupling enhances the chances for two neighboring nodes to communicate securely. In the following, we derive the expressions for P_1, P_2 and P_3.

Recall that P_1 is the probability that two nodes share at least one pre-distributed key. It is given by,

$$P_1 = 1 - \binom{K}{2k} \cdot \binom{2k}{k} / \binom{K}{k}^2. \tag{3}$$

If D_p denotes the average physical node degree, the average number of nodes in the overlapped area of the communication ranges of two physically neighboring nodes is $0.5865D_p$ [5]. The probability that among the $0.5865D_p$ nodes, n_1 nodes share key with one of those two physically neighboring nodes is $\binom{0.5865D_p}{n_1}(P_1)^{n_1}(1-P_1)^{0.5865D_p-n_1}$. The probability that at least one of the above n_1 nodes shares key with the other node is $1-(1-P_1)^{n_1}$. Therefore, the expression for P_2 is given by,

$$P_2 = (1-P_1) \cdot \sum_{n_1=1}^{0.5865D_p} \Big(\binom{0.5865D_p}{n_1} (P_1)^{n_1}(1-P_1)^{0.5865D_p-n_1} \cdot (1-(1-P_1)^{n_1}) \Big). \tag{4}$$

For two physically neighboring nodes, the average number of nodes in the communication range of one node but outside the communication range of the other node is $2(D_p - 0.5865D_p) = 0.8270D_p$. Similarly, the expression for P_3 is given by,

$$P_3 = (1-P_1) \cdot (1-P_2) \cdot \sum_{n_1=1}^{0.8270D_p} \Big(\binom{0.8270D_p}{n_1} (P_1)^{n_1}(1-P_1)^{0.8270D_p-n_1} \cdot (1-(1-P_1)^{n_1}) \Big). \tag{5}$$

2) Case2: Multiple proxies or logical intermediate nodes on a key path: In the above, we derived expressions for $P^{(1)}_{couple}$ (with one proxy on a key path) and $P^{(1)}_{decouple}$ (with one intermediate node on a key path). In general, multiple proxies or logical intermediate nodes can be used on a key path to further enable the chances of establishing pair-wise keys. In the following, we present only the final results for the general case of multiple proxies or multiple logical intermediate nodes. For detailed derivation, please refer to the Appendix. Formally, the probabilities that two physically neighboring nodes are able to establish a pair-wise key in the RKP protocol (with multiple proxies), denoted by P_{couple}, and the RKP-DE protocol (with multiple logical intermediate nodes), denoted by $P_{decouple}$, are given by,

$$P_{couple} = 1 - (1 - P_{couple}(A_a)) \cdot (1 - P_{couple}(A_b|\bar{A}_a)), \tag{6}$$

$$P_{decouple} = 1 - (1 - P_{decouple}(A_a)) \cdot (1 - P_{decouple}(A_b|\bar{A}_a)). \tag{7}$$

In the above expressions, $P_{couple}(A_a)$ and $P_{decouple}(A_a)$ denote the probability that node a is able to construct a key path to its physical neighbor node b based on its local information within its communication range area in the RKP and RKP-DE protocols respectively. The communication range area is denoted by $A_a = \pi r^2$, where r is the communication range of the sensors. In the above expressions, $P_{couple}(A_b|\bar{A}_a)$ and $P_{decouple}(A_b|\bar{A}_a)$ denote the probability that node b is able to construct a key path to node a based on its local information (within range $A_b = \pi r^2$), given node a cannot construct such key path to node b, in the RKP and RKP-DE protocols respectively. The above results will be used later in the analysis of average secure node degree in Section IV.

IV. Secure Neighbor Establishment Protocol in Decoupled Networks

A. Overview

In this section, we discuss the design of our new protocol for establishing secure neighbors (i.e., establishing pair-wise keys) in decoupled random key pre-distributed sensor networks. We call our protocol as the RKP-DE protocol. The protocol has four major components in its execution: 1) constructing local logical and physical graphs in the decoupled network for each node, 2) establishing multiple key paths between neighboring nodes, 3) eliminating dependencies among the multiple key paths, and 4) establishing pair-wise keys between neighboring nodes. The RKP-DE protocol is distributed in its execution like the traditional RKP protocol. The network model we consider is the same as that in traditional protocol, where a set of n sensors are deployed randomly. Each sensor is pre-distributed with k distinct keys randomly chosen from a key pool of size K.

The major differences between our RKP-DE protocol and the traditional RKP protocol are due to the first three components. In the traditional RKP protocol, key paths are established in a network where the logical and physical graphs are coupled. On the other hand, in our RKP-DE protocol, the logical and physical graphs are separated/decoupled. The first component of our RKP-DE protocol is each node constructing these two local graphs decoupled from each other. The logical graph is constructed based on key sharing information and the physical graph is constructed based on physical neighborhood information, following the methodology of network decoupling discussed earlier in Section III. The second component in our RKP-DE protocol is to establish logical key paths between two physically neighboring nodes based on the logical graph, and for these logical key paths, corresponding physical key paths are established based on the physical graph. The decoupling feature enables more key paths (both logical and physical) to be constructed when compared to the traditional RKP protocol. Note that when multiple key paths (each with multiple links (or hops)) are constructed, there is a possibility of some links (or paths) being dependent on other links (or paths). Such dependencies introduce unnecessary overhead in terms of communication and computation. The third component in our RKP-DE protocol proposes novel dependency elimination rules to detect and eliminate such dependencies without compromising the existing resilience. Each component in our RKP-DE protocol is described in detail below.

B. Local Graphs Construction

After node deployment, each node obtains the key sharing and physical neighborhood information in its communication range by local communication with its physical neighbors. We assume that from local communication, each node can determine whether any two of its physical neighbors are physical neighbors or not. This can be easily done by exchanging neighbor information during initial communication. With this information, each node constructs a local logical graph (G_l) and a local physical graph (G_p). In the local logical graph (e.g., Fig. 3 (b)), two nodes are connected if they share at least one key, while in the local physical graph (e.g., Fig. 3 (c)), two nodes are connected if they are within communication range of each other. Note that our protocol needs only local information exchange and is purely distributed.

Algorithm 1 shows the pseudocode of key paths construction executed by each node in the network. In Algorithm 1, u denotes an arbitrary node, while $G_l(u)$ and $G_p(u)$ are its local logical and physical graphs respectively. Initially the logical key path tree (T_u) is empty. The key paths construction is executed in two steps as shown in Algorithm 1. First, T_u is constructed by node u based on its local logical graph $G_l(u)$ (lines 1 to 7). This logical key path tree T_u contains all the logical key paths between u and all its secure neighbors. Then, node u constructs corresponding physical key paths based on both T_u and its local physical graph $G_p(u)$ (lines 8 to 13). The dependency checking in lines 3 and 11 will be discussed in the next subsection.

C. Key Paths Construction

Logical key path tree construction: The protocol constructs logical key path tree (lines 1 to 7) using a variant of the standard depth-first-search algorithm, in which a node could be chosen multiple times (on

Algorithm 1 Pseudocode of Key Paths Construction

```
1:  Logical_Key_Path_Tree_Construction($u$,$G_l(u)$,$T_u$)
2:  for each $v \in N(u)$
3:    if $Link\_Dependency\_Checking(v, u, T_u) == PASS$, then
4:      $Insert(u, v, T_u)$;
5:      $Logical\_Key\_Path\_Tree\_Construction(v, G_l(u), T_u)$;
6:    end if
7:  end for

8:  Physical_Key_Paths_Construction($u$,$G_p(u)$,$T_u$)
9:  for each $v \in N(u)$
10:   $obtain\ the\ set\ of\ all\ logical\ key\ paths\ between\ u\ and\ v\ (T_{uv})\ from\ T_u$;
11:   $T'_{uv} = Path\_Dependency\_Checking(T_{uv})$;
12:   $obtain\ the\ corresponding\ set\ of\ physical\ key\ paths\ T^*_{uv}\ from\ T'_{uv}$;
13: end for

14: Insert($u$,$v$,$T_u$)
15:   $Insert\ node\ v\ into\ T_u\ as\ a\ child\ of\ node\ u.$
```

different paths). Here $N(u)$ denotes the set of physical neighbors of node u. Fig. 4 shows the resultant logical key path tree for node a in the example of Fig. 3 (b). By executing the algorithm just once on its local logical graph in Fig. 3 (b), node a is able to obtain all logical key paths to all its neighbors. Taking node e as an example, node a obtains two logical key paths between node a and node e, that are $< a, b, c, e >$ and $< a, b, d, e >$.

Physical key paths construction: After obtaining the logical key path tree (T_u), node u begins to construct physical key paths for its physical neighbors (lines 8 to 13). For each physical neighbor v, node u first obtains a set of logical key paths between u and v (T_{uv}) from T_u. Out of all such paths in T_{uv}, some of them will be eliminated based on dependency checking (as discussed in next subsection). The set of paths that pass the dependency checking is denoted as T'_{uv}. Finally, for all logical key paths in T'_{uv}, corresponding physical key paths T^*_{uv} are obtained. In Fig. 3 (b), the logical key path $< a, b, d, e >$ contains a logical hop $< b, d >$ between two non-neighboring nodes. From Fig. 3 (c), we see that a physical path $< b, a, d >$ can replace the above logical hop. Therefore, for logical key path $< a, b, d, e >$, its corresponding physical key path is $< a, b, a, d, e >$, in which each hop is between two physically neighboring nodes. Message encryption/decryption occurs for each logical hop, while message transmission occurs for each physical hop. Here, we select the physical path with fewest hops to replace logical hops between non-neighboring nodes. Other policies can be chosen if energy consumption, load balancing, etc. are to be considered.

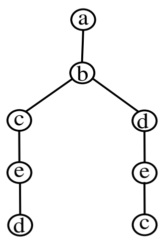


Fig. 4. Logical Key Path Tree

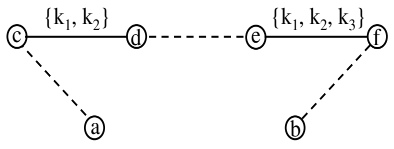


Fig. 5. Link Dependency

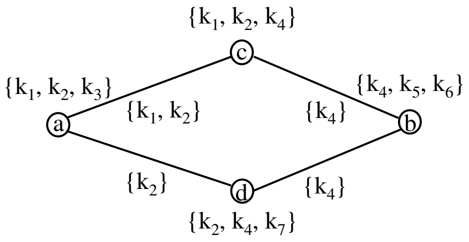


Fig. 6. Path Dependency

D. Dependency Elimination

We now discuss elimination of link and path dependencies in steps 3 and 11 of Algorithm 1. Generally, if more key paths are used, resilience is enhanced. This is because when multiple key paths exist between two nodes, the attacker needs to compromise all key paths in order to compromise the secure communication between them. However, this is not always true. Existing links (or paths) may have dependencies among them such that the compromise of some links (or paths) automatically leads to the compromise of other dependent links (or paths). Clearly, the presence of such dependency does not enhance resilience. They only increase overhead in terms of both storage and energy consumption (due to communication and computation). In this subsection, we propose two novel *dependency elimination rules* to decrease such overheads without affecting the resilience of the established pair-wise keys.

1) Link Dependency Elimination: We illustrate link dependency with an example in Fig. 5. Node a obtains a logical key path $< a, \cdots, c, d, \cdots, e, f, \cdots, b >$ to its physical neighbor node b. We denote $K(i,j)$ as the set of keys used to encrypt the messages on the logical hop $< i, j >$ in a logical key path, which actually is the set of all shared keys between nodes i and j. There exists a link dependency between the hops $< c, d >$ and $< e, f >$ in that $K(c,d) \subseteq K(e,f)$. Since both nodes c and f share keys k_1 and k_2, there must exist another shorter logical key path $< a, \cdots, c, f, \cdots, b >$, which has better resilience than the original one. This is because the compromise of any logical hop between nodes d and e will compromise the original key path, while it is possible that the shorter key path is not compromised. On the other hand, the compromise of the shorter key path will definitely compromise the original key path. Also, using a shorter key path will save overhead. We formally define link dependency below.

Link Dependency: Given two logical hops $< i_1, j_1 >$ and $< i_2, j_2 >$ in a logical key path, there exists link dependency between these two hops if either $K(i_1, j_1) \subseteq K(i_2, j_2)$ or $K(i_2, j_2) \subseteq K(i_1, j_1)$.

Our *link dependency elimination rule* is that once such a link dependency is detected on a logical

Algorithm 2 Pseudocode of Dependency Checking

```
1:  **Link_Dependency_Checking**($v$,$u$,$T$)
2:  **if** $\exists$ node $w \in$ Path($u$,$root$), s.t. $K(v, v.parent) \subseteq K(w, w.parent)$, **then**
3:      $return\ FAIL$;
4:  **else if** $\exists$ node $w \in$ Path($u$,$root$), s.t. $K(w, w.parent) \subseteq K(v, v.parent)$, **then**
5:      $return\ FAIL$;
6:  **else** $return\ PASS$;
7:  **end if**

8:  **Path_Dependency_Checking**($T_{uv}$)
9:  $T'_{uv} = T_{uv}$;
10: **while** $\exists$ paths $p$ and $q \in T'_{uv}$, s.t. $p$ is weaker than $q$ OR $q$ is weaker than $p$, **do**
11:     **if** $p$ is weaker than $q$, **then**
12:         $T'_{uv} = T'_{uv} \setminus p$;
13:     **else if** $q$ is weaker than $p$, **then**
14:         $T'_{uv} = T'_{uv} \setminus q$;
15:     **end if**
16: **end while**
17: $return\ T'_{uv}$;
```

key path, the protocol will eliminate that logical key path. In the above example, the logical key path $< a, \cdots, c, d, \cdots, e, f, \cdots, b >$ will be eliminated since a shorter key path $< a, \cdots, c, f, \cdots, b >$ with better resilience exists. The pseudocode of link dependency checking is given in Algorithm 2 (lines 1 to 7). In Algorithm 2, $root$ denotes the root node of the logical key path tree T, $Path(u, root)$ denotes the set of nodes on the logical key path from u to $root$, and $v.parent$ denotes the parent node of node v on the tree T. As we can see in Algorithm 2, link dependency will be checked to output a $PASS$ or $FAIL$, which is returned in line 3 of Algorithm 1. In the following, we formally prove that our link dependency elimination rule does not affect the resilience of the pair-wise key established.

Claim 1: The link dependency elimination rule proposed above does not affect the resilience of the pair-wise key established.

Proof: Suppose that a logical key path p $(< a, \cdots, i_1, j_1, \cdots, i_2, j_2, \cdots b >)$ has link dependency between two logical hops $< i_1, j_1 >$ and $< i_2, j_2 >$. Without loss of generality, let us assume that $K(i_1, j_1) \subseteq K(i_2, j_2)$. Therefore, there must exist a shorter logical key path q $(< a, \cdots, i_1, j_2, \cdots b >)$ since both i_1 and j_2 have the keys in $K(i_1, j_1)$. Also, the corresponding physical key path of q must exist since the underlying local physical graph is connected [2]. The compromise of path q means that at least one of the logical hops on path q is compromised, which makes path p compromised definitely. Therefore, the elimination of path p will not affect the resilience of the pair-wise key established. ■

[2]A node in the local physical graph of node a is either node a itself or a physical neighbor of node a.

2) Path Dependency Elimination: Apart from link dependency, another type called path dependency may exist. In Fig. 6, there are two logical key paths between nodes a and b. However, we can see that the compromise of the key path $< a, c, b >$ (disclosure of keys $(k_1,\ k_2)$ or (k_4)) always leads to the compromise of the other key path $< a, d, b >$, but not vice versa. Therefore, given that key path $< a, c, b >$ exists, the other key path $< a, d, b >$ becomes redundant in terms of resilience, and also incurs unnecessary overhead. Denoting the set of logical hops on a logical key path p as L_p, and denote the set of keys used on a logical hop h as $K(h)$, path dependency is formally defined as follows.

Path Dependency: Given two logical key paths p and q, there exists path dependency between p and q if either of the following two conditions is satisfied. *(1)* $\forall\ logical\ hop\ h \in L_q,\ \exists\ a\ logical\ hop\ h'\ (h' \in L_p),\ s.t.\ K(h') \subseteq K(h)$; *(2)* $\forall\ logical\ hop\ h \in L_p,\ \exists\ a\ logical\ hop\ h'\ (h' \in L_q),\ s.t.\ K(h') \subseteq K(h)$.

If the first condition of path dependency is satisfied, we call path p $weaker$ than path q. Similarly, path q is $weaker$ than path p if the second condition is satisfied. Our *path dependency elimination rule* is that after detecting path dependency between two logical key paths, our protocol will eliminate the weaker one. In the above example, the logical key path $< a, d, b >$ will be eliminated. In case two paths satisfy both conditions in the path dependency, we can eliminate one of them based on certain policies (e.g., the path with more physical hops). The pseudocode of path dependency checking is given in Algorithm 2 (lines 8 to 17). In the following, we formally prove that our path dependency elimination rule does not affect the resilience of the pair-wise key established.

Claim 2: The path dependency elimination rule proposed above does not affect the resilience of the pair-wise key established.

Proof: Suppose that a logical key path p is weaker than another logical key path q. From the definition of path dependency, we can see that the compromise of path q always leads to the compromise of path p. Since a physical key path for path q exists and will be constructed by our protocol, the elimination of path p will not affect the resilience of the pair-wise key established. ∎

E. Pair-wise Key Establishment

Once key paths are constructed after dependency elimination, each sensor will generate distinct key shares at random, and send each key share on each physical key path. The messages are transmitted at each physical hop, while they are encrypted/decrypted at each logical hop. Take the logical key path

$< a, b, d >$ in Fig. 3 (b) as an example. Its corresponding physical key path is $< a, b, a, d >$. We assume a key share $k_a^{(1)}$ is transmitted on this key path. We denote $\{M\}_k$ as the message M encrypted with key k. The key share transmission of $k_a^{(1)}$ is executed as follows:

$$\begin{aligned} a \mapsto b: &\quad < 1 >, \{< a >, < a, d, 5 >, < k_a^{(1)} >\}_{k_1}, \\ b \mapsto a: &\quad < d >, < 5 >, \{< a >, < \phi >, < k_a^{(1)} >\}_{k_5}, \\ a \mapsto d: &\quad < 5 >, \{< a >, < \phi >, < k_a^{(1)} >\}_{k_5}. \end{aligned}$$

In the message that node a sends to node b, $< 1 >$ denotes the ID of the key used to encrypt the remaining message, $< a >$ denotes the source node, and $< a, d, 5 >$ denotes the remaining physical key path and the ID of the key used to encrypt the message forwarded to the next node d on the logical key path [3]. In the message that b sends to a, $< d >$ denotes the remaining physical path of the current logical hop since a cannot decrypt the message using key k_5.

Similarly, node a can transmit another random key share $k_a^{(2)}$ on another logical key path $< a, b, c, e, d >$ to node d. Node d may also construct other key paths (not shown in Fig. 3 (b)), and transmit its key shares to node a. Finally, nodes a and d can compute a common pair-wise key via some simple operation such as bit-wise XOR operation, based on all the key shares they both generated. In this way, the established pair-wise key is compromised if and only if all the key shares (key paths) are compromised.

F. Analysis

In this section, we derive expressions for the average secure node degree and overhead of our protocol.

1) Average secure node degree: In this section, we will first derive the expressions for the average secure node degree in both RKP protocol and RKP-DE protocol, denoted by D_s^{RKP} and D_s^{RKP-DE} respectively. A high average secure node degree means more secure neighbors per node, thereby indicating better performance of the protocol from the perspective of both connectivity and resilience. In Section III, we gave the expressions for P_{couple} and $P_{decouple}$ in equations (6) and (7) respectively, which denote the probability that two physically neighboring nodes are able to construct a key path in the RKP protocol and RKP-DE protocol respectively. Therefore, we can derive D_s^{RKP} and D_s^{RKP-DE} as,

$$D_s^{RKP} = D_p \cdot P_{couple}, \tag{8}$$

[3]The next node on the logical key path is the node before the first number in the list.

TABLE I

IMPROVEMENT OF $D_s^{RKP-DE(1)}$ (D_s^{RKP-DE}) OVER $D_s^{RKP(1)}$ (D_s^{RKP}) UNDER DIFFERENT D_p

D_p	5	10	15	20	25
$D_s^{RKP(1)}$	1.66	4.26	7.59	11.52	15.89
$D_s^{RKP-DE(1)}$	2.27	6.16	10.96	16.22	21.68
$IM^{(1)}$	37%	45%	44%	41%	36%
D_s^{RKP}	1.93	6.07	11.62	17.75	23.80
D_s^{RKP-DE}	2.63	8.05	14.43	19.85	24.98
IM	37%	33%	24%	12%	5%

$$D_s^{RKP-DE} = D_p \cdot P_{decouple}, \tag{9}$$

where recall that D_p denotes the average physical node degree. The improvement of D_s^{RKP-DE} over D_s^{RKP}, denoted by IM, is then given by,

$$IM = \frac{D_s^{RKP-DE} - D_s^{RKP}}{D_s^{RKP}}. \tag{10}$$

The values of D_s^{RKP} and D_s^{RKP-DE} depend on key pool size K, key chain size k and average physical node degree D_p (from equations (6), (7), (8) and (9)). Under different values of D_p, we compute the values of D_s^{RKP}, D_s^{RKP-DE} and the improvement IM in Table I ($K = 10000$, $k = 50$). In Table I, we also give the values of $D_s^{RKP(1)}$, $D_s^{RKP-DE(1)}$ and $IM^{(1)}$, which are the average secure node degree by using only key paths consisting of one proxy in the RKP protocol and one logical intermediate node in the RKP-DE protocol, and the improvement of $D_s^{RKP-DE(1)}$ over $D_s^{RKP(1)}$ respectively. The expressions for $D_s^{RKP(1)}$ and $D_s^{RKP-DE(1)}$ are the same as the equations (8) and (9) except that we replace P_{couple} and $P_{decouple}$ by $P_{couple}^{(1)}$ and $P_{decouple}^{(1)}$ (in equations (1) and (2)) respectively. The expression for $IM^{(1)}$ is the same as equation (10) except that we replace D_s^{RKP} and D_s^{RKP-DE} by $D_s^{RKP(1)}$ and $D_s^{RKP-DE(1)}$ respectively. We can see that network decoupling improves the average secure node degree under all situations. The improvement in average secure node degree helps to enhance the performance of random key pre-distribution in terms of connectivity and resilience, which will be demonstrated using simulations in the following section. We also observe that the improvement in case of multiple logical intermediate nodes (IM) diminishes for larger D_p. This is because in highly dense network, most physically neighboring nodes are able to establish pair-wise keys via the help of nearby nodes. Therefore, the value of D_s^{RKP} is close to that of D_p, and the improvement diminishes.

TABLE II
STRETCH FACTOR (SF^{RKP} AND SF^{RKP-DE}) UNDER DIFFERENT D_p

D_p	5	10	15	20	25
SF^{RKP}	1.54	1.91	2.01	2.04	2.03
SF^{RKP-DE}	2.14	2.53	2.46	2.37	2.30

2) Stretch factor: In this paper, we define a new metric called $stretch\ factor$ to study overhead of both RKP protocol and RKP-DE protocol. Formally, the stretch factor is the average number of physical hops on the key path between two secure neighbors. Physically speaking, it denotes the communication overhead of the protocol since a message needs to be transmitted/forwarded once for each physical hop on the key path for key share transmission. We denote the stretch factor for the RKP protocol and RKP-DE protocol as SF^{RKP} and SF^{RKP-DE} respectively and derive them below.

Let us denote $P_h(i)$ and $P_h(i)'$ as the probability that a node can find a logical key path to a physically neighboring node within its communication range with minimum logical hop i in RKP-DE protocol and RKP protocol respectively. We further denote α as the average number of physical hops for a logical hop on a key path (except the first logical hop). Then, SF^{RKP} and SF^{RKP-DE} are given by,

$$SF^{RKP} = \sum_{i=1}^{\infty}((1+(i-1)\alpha) \cdot P'_h(i)) / \sum_{i=1}^{\infty}(P'_h(i)), \tag{11}$$

$$SF^{RKP-DE} = \sum_{i=1}^{\infty}((1+(i-1)\alpha) \cdot P_h(i)) / \sum_{i=1}^{\infty}(P_h(i)). \tag{12}$$

In the above equations, $1+(i-1)\alpha$ denotes the average physical hops of a key path with i logical hops. This is because the first logical hop is between the source node and one of its physical neighbors (resulting in one physical hop). For each of the remaining $i-1$ logical hops, the two nodes of that logical hop are within communication range (one physical hop) with probability 0.5865 [5], and are connected by means the source node (two physical hops) for all the other situations with probability $1-0.5865=0.4135$. Therefore, each of the remaining $i-1$ logical hops has average number of physical hops $\alpha = 1 \cdot 0.5865 + 2 \cdot 0.4135 = 1.4135$. The derivation of $P_h(i)$ and $P_h(i)'$ are given in the Appendix.

In Table II, we show the values of SF^{RKP} and SF^{RKP-DE} under various D_p for $K = 10000$ and $k = 50$. We can see that the average stretch factor in our RKP-DE protocol is only slightly larger than that of the traditional RKP protocol for the same D_p. This is because with network decoupling, many longer key paths not identified in the traditional RKP protocol can be constructed by our RKP-DE

protocol. With the increase in number of longer key paths, stretch factor increases. We wish to point out that for any key path constructed by the traditional RKP protocol, our RKP-DE protocol can construct a corresponding key path with the same or fewer logical/physical hops. Suppose there exists a key path between a source node and a destination node with multiple logical/physical hops constructed by the RKP protocol. Consider a segment (a part of the key path) such that the two end nodes of the segment (not the source/destination pair) share keys but are not physical neighbors. In our RKP-DE protocol, we can replace the above segment by a single logical hop with two physical hops [4]. This clearly decreases the number of physical hops and the stretch factor. Therefore, our RKP-DE protocol actually decreases the average stretch factor of the key paths that can be constructed by both protocols.

V. Performance Evaluations

In this section, we report experimental data to demonstrate the performance of our RKP-DE protocol compared to the traditional RKP protocol under various network and attack parameters. The metrics we study are connectivity (local connectivity and global connectivity) and resilience.

A. Simulation Environment

The sensor network is a square region of size $1000m$ * $1000m$, in which 1000 sensors are deployed uniformly at random. The communication range r is the same for all sensors and is chosen based on the desired average physical node degree D_p. Each node is aware of the key sharing and physical neighborhood information within its communication range. The following are the default values for the parameters unless otherwise specified: average physical node degree $D_p = 10$, key pool size $K = 10000$, key chain size $k = 50$. The attack model is one where the attacker can monitor all links in the network, and can capture up to x nodes. By default, $x = 50$. Each simulation is run 100 times with different random seeds, and the data presented is the average of 100 runs. In both RKP protocol and RKP-DE protocol, each node tries to establish a pair-wise key with each of its physical neighbors using multiple key paths (with arbitrary number of proxies or logical intermediate nodes) based solely on its local information.

[4]For the case with one hop information, all nodes on the key path are within communication range of the source node.

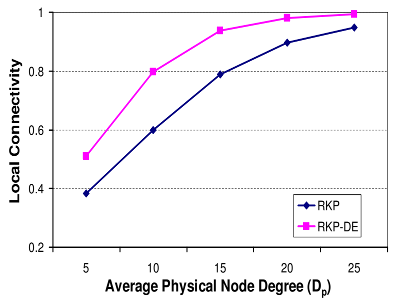


Fig. 7. Sensitivity of local connectivity to D_p

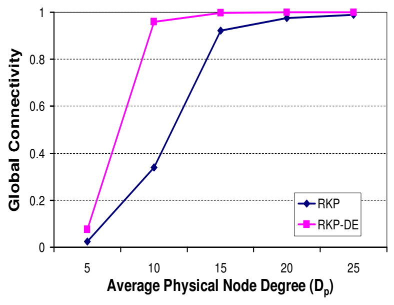


Fig. 8. Sensitivity of global connectivity to D_p

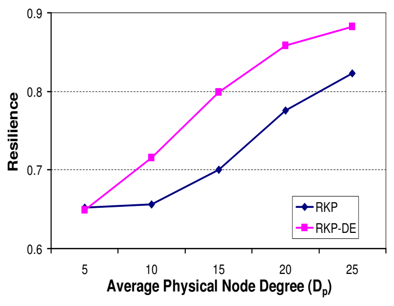


Fig. 9. Sensitivity of resilience to D_p

B. Sensitivity of Connectivity to D_p

1) Local Connectivity: In Fig. 7, we study the sensitivity of local connectivity to average physical node degree D_p. Recall that local connectivity is defined as the probability that two physically neighboring nodes are able to establish a pair-wise key in between. We observe that the local connectivity in RKP-DE protocol is consistently higher than that in the RKP protocol. The improvement is in fact more significant (about 35% improvement) for non-highly dense networks where $D_p < 20$. In RKP protocol, the consideration of both physical and logical constraints in key path construction limits the availability of key paths between physical neighbors. However, the relaxation of the constraints as a result of network decoupling enables the availability of many more key paths, which greatly enhances the local connectivity.

2) Global Connectivity: The definition of global connectivity here is the percent of nodes in the largest connected component of the secure network (e.g., Fig. 1 (b) or (c)) on average. In Fig. 8, we observe that the global connectivity of RKP-DE protocol is higher than that of the RKP protocol in all situations. The improvement is especially significant in non-highly dense networks (up to 200% improvement). This improvement is a result of the phase transition phenomenon in random graphs [14]. According to this phenomenon, the largest connected component in a random graph with n nodes jumps from $\Theta(\log n)$ to $\Theta(n)$ when the average node degree reaches beyond a certain threshold. With network decoupling in our RKP-DE protocol, such a jump in global connectivity occurs when D_p is around 10 compared to the RKP protocol when D_p is around 15. Another observation is that the global connectivity when $D_p = 10$ in our RKP-DE protocol is close to the global connectivity when $D_p = 20$ in the RKP protocol. This demonstrates that we can obtain similar levels of global connectivity with much fewer nodes compared to the number of nodes needed in the RKP protocol.

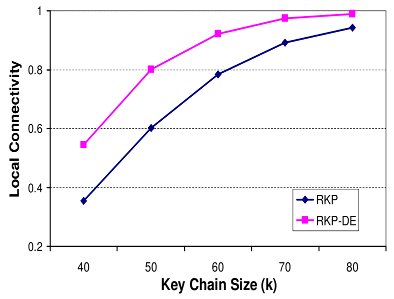


Fig. 10. Sensitivity of local connectivity to k

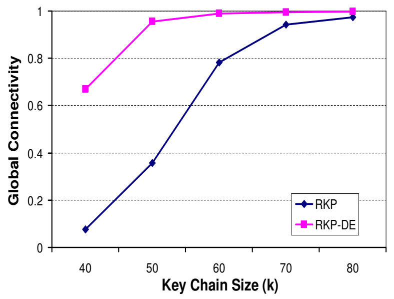


Fig. 11. Sensitivity of global connectivity to k

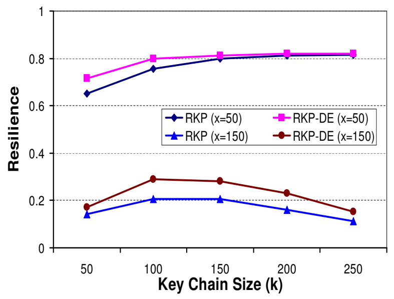


Fig. 12. Sensitivity of resilience to k and x

C. *Sensitivity of Resilience to D_p*

In Fig. 9, we study the sensitivity of resilience to D_p. Recall that resilience is defined as the probability that a pair-wise key between two nodes is not compromised given that those two nodes are not captured. We see that resilience is higher in RKP-DE compared to that of RKP in general. The improvement is consistent except when the network is very sparse ($D_p = 5$). Network decoupling not only increases the number of key paths between two physically neighboring nodes, but also decreases the logical hops of many key paths, both of which help enhance the resilience. When network becomes very sparse, only a single key path can be constructed in most situations, thus the improvement diminishes.

D. *Sensitivity of Connectivity and Resilience to k and x*

In Fig. 10, 11 and 12, we study the sensitivity of connectivity and resilience to k and x. In Fig. 10 and 11, we see similar pattern in sensitivity of connectivity to k as that to D_p. This is because the increase in k enhances the probability that two nodes share keys, which makes the local logical graphs more dense. This can also be achieved by increasing D_p as well. Overall, our RKP-DE protocol achieves better performance than that of RKP protocol, and the performance improvement is especially significant in non-highly dense network. On the other hand, given the same performance requirement, our RKP-DE protocol can save storage overhead (proportional to k) up to around 30% compared with the RKP protocol. For example, given $k = 80$ in the RKP protocol, our RKP-DE protocol can achieve similar performance with k around (or smaller than) 60.

In Fig. 12, we study the sensitivity of resilience to k under different values for number of captured nodes x. We observe that the resilience of our RKP-DE protocol is better than that of the RKP protocol for all cases. When x is relatively small ($x = 50$), the resilience increases with k (for k no more than 250) and the improvement of RKP-DE over RKP diminishes when k increases. This is because when

k increases, the local logical graphs become more dense and more key paths can be constructed. While the increased density improves resilience, it also results in diminishing the performance improvement of RKP-DE over RKP. On the other hand, when x is relatively large ($x = 150$), the resilience first increases with k, and then begins to decrease when k further increases. This is because when both k and x are relatively large, the attacker is able to disclose a significant percent of the pre-distributed keys thereby compromising many more established pair-wise keys, which degrades the resilience. The threshold value for k, beyond which resilience begins to decrease, will decrease with increase in x. Therefore, the resilience of the case when $x = 50$ will also begin to decrease when k further increases (beyond 250), which is confirmed by our simulation but not shown in Fig. 12. We also notice that the improvement of RKP-DE over RKP is more pronounced for larger x (i.e., stronger attacks) further demonstrating the effectiveness of our RKP-DE protocol. The value of x does not impact connectivity, so we do not show the sensitivity of connectivity to x.

VI. Related Work

The RKP protocol has received wide acceptance in WSNs due to its distributedness, simplicity, energy efficiency and scalability. It has served as a foundation for many other works based on random key pre-distribution, aiming to improve performance [5], [6], [7], [8], [9], [10], [15]. In [5] and [9], the performance of the basic RKP protocol is enhanced by constructing multiple key paths using proxies for pair-wise key establishment between neighbors. With multiple key paths, as long as at least one path is uncompromised, the pair-wise key is secure. Similarly, [6] uses multiple two hop key paths to enhance resilience further under a slightly weaker attack model. In [7], [8], [10] and [15], the authors extend the basic RKP protocol by pre-distributing key structures (either polynomials or vectors) instead of keys to establish pair-wise keys. When number of captured nodes is small, this protocol has much better resilience compared to the basic RKP protocol. We point out that in the above works, a very high network density (average physical node degree between 20 and 250) is assumed to achieve satisfactory performance.

An orthogonal extension to the basic RKP protocol is exploiting certain network properties to enhance performance or decrease overhead. Works like [16], [17], [18] and [19] use power control, node mobility, channel diversity or hierarchy to enhance performance under assumptions on sensor hardware, network topology etc. Recently, some works have used deployment knowledge to achieve comparable performance with fewer number of keys pre-distributed [20], [21], [22]. These works rely on the assumption that positions of neighboring nodes in the network are partially known a priori, helping in decreasing number of keys pre-distributed to achieve comparable performance. We point out that our methodology of network decoupling is orthogonal to all the works above, and can complement them to achieve further performance improvement and overhead reduction.

In random key pre-distribution, a clear tradeoff exists among connectivity, resilience and storage overhead under different values of k and K. Intuitively, connectivity can be improved by either increasing k or decreasing K. When the difference between k and K is smaller, the probability that two physically neighboring nodes share at least one key increases, which increases connectivity (with an increase in storage overhead due to larger k). On the other hand, it has been revealed in [5] that increasing k or decreasing K may compromise resilience as shown in Fig. 12 in Section V. This is because, when more nodes are captured, a larger percent of pre-distributed keys are disclosed which naturally decreases the resilience. In our work, the methodology of network decoupling in the $RKP\text{-}DE$ protocol achieves increased connectivity for a given k and K compared to the RKP protocol without compromising resilience or increasing storage overhead.

VII. Final Remarks

In this paper, we proposed *network decoupling* to separate logical key-sharing relationship from physical neighborhood relationship in random key pre-distributed sensor networks. We designed a new key management protocol ($RKP\text{-}DE$) in decoupled sensor networks, and also designed a set of rules for eliminating link and path level dependencies among the key paths. We conducted detailed analysis as well as extensive simulations to evaluate our proposed solutions from the perspective of connectivity, resilience and overhead. Our data showed that significant performance improvement can be achieved using our solutions in non-highly dense networks, with only a mild increase in the overhead. Our future work will consist of practically implementing our proposed solutions on the existing sensor network testbed at OSU [23], [24].

References

[1] W. Diffie and M. E. Hellman, "New directions in cryptography," *IEEE Transactions on Information Theory*, vol. IT-22, no. 6, pp. 644–654, November 1976.

[2] R. L. Rivest, A. Shamir, and L. M. Adleman, "A method for obtaining digital signatures and public-key cryptosystems," *Communications of the ACM*, vol. 21, no. 2, pp. 120–126, February 1978.

[3] B. C. Neuman and T. Tso, "Kerberos: an authentication service for computer networks," *IEEE Communications Magazine*, vol. 32, no. 9, pp. 33–38, September 1994.

[4] L. Eschenauer and V. D. Gligor, "A key-management scheme for distributed sensor networks," in *Proceedings of the 9th ACM Conference on Computer and Communications Security (CCS)*, November 2002.

[5] H. Chan, A. Perrig, and D. Song, "Random key predistribution schemes for sensor networks," in *Proceedings of IEEE Symposium on Research in Security and Privacy*, May 2003.

[6] A. Wacker, M. Knoll, T. Heiber, and K. Rothermel, "A new approach for establishing pairwise keys for securing wireless sensor networks," in *Proceedings of the 3rd ACM Conference on Embedded Networked Sensor Systems (Sensys)*, November 2005.

[7] W. Du, J. Deng, Y. S. Han, and P. K. Varshney, "A pairwise key pre-distribution scheme for wireless sensor networks," in *Proceedings of the 10th ACM Conference on Computer and Communications Security (CCS)*, October 2003.

[8] D. Liu and P. Ning, "Establishing pairwise keys in distributed sensor networks," in *Proceedings of the 10th ACM Conference on Computer and Communications Security (CCS)*, October 2003.

[9] S. Zhu, S. Xu, S. Setia, and S. Jajodia, "Establishing pairwise keys for secure communication in ad hoc networks: a probabilistic approach," in *Proceedings of the 11th IEEE International Conference on Network Protocols (ICNP)*, November 2003.

[10] F. Delgosha and F. Fekri, "Threshold key-establishment in distributed sensor networks using a multivariate scheme," in *Proceedings of the 25th IEEE Conference on Computer Communications (INFOCOM)*, April 2006.

[11] A. Snoeren and B. Raghavan, "Decoupling policy from mechanism in internet routing," in *Proceedings of the ACM SIGCOMM Workshop on Hot Topics in Networking (HotNets-II)*, November 2003.

[12] H. Kung and S. Wang, "Tcp trunking: Design, implementation and performance," in *Proceedings of the 11th IEEE International Conference on Network Protocols (ICNP)*, November, 1999.

[13] D. Niculescu and B. Nath, "Trajectory based forwarding and its applications," in *Proceedings of the 9th ACM International Conference on Mobile Computing and Networking (MOBICOM)*, September, 2003.

[14] J. Spencer, *The Strange Logic of Random Graphs, Algorithms and Combinatorics 22*, Springer-Verlag, 2000.

[15] F. Delgosha and F. Fekri, "Key predistribution in wireless sensor networks using multivariate polynomials," in *Proceedings of the 2nd IEEE Communications Society Conference on Sensor and Ad Hoc Communications and Networks (SECON)*, September 2005.

[16] J. Hwang and Y. Kim, "Revisiting random key pre-distribution schemes for wireless sensor networks," in *Proceedings of the 2nd ACM Workshop on Security of Ad Hoc and Sensor Networks (SASN)*, October 2004.

[17] Y. Mao and M. Wu, "Coordinated sensor deployment for improving secure communications and sensing coverage," in *Proceedings of the 3rd ACM Workshop on Security of Ad Hoc and Sensor Networks (SASN)*, November 2005.

[18] M. Miller and N. Vaidya, "Leveraging channel diversity for key establishment in wireless sensor networks," in *Proceedings of the 25th IEEE Conference on Computer Communications (INFOCOM)*, April 2006.

[19] P. Traynor, H. Choi, G. Cao, S. Zhu, and T. L. Porta, "Establishing pair-wise keys in heterogeneous sensor networks," in *Proceedings of the 25th IEEE Conference on Computer Communications (INFOCOM)*, April 2006.

[20] D. Huang, M. Mehta, D. Medhi, and L. Harn, "Location-aware key management scheme for wireless sensor networks," in *Proceedings of the 2nd ACM Workshop on Security of Ad Hoc and Sensor Networks (SASN)*, October 2004.

[21] W. Du, J. Deng, Y. Han, S. Chen, and P. Varshney, "A key management scheme for wireless sensor networks using deployment knowledge," in *Proceedings of the 23rd IEEE Conference on Computer Communications (INFOCOM)*, March 2004.

[22] D. Liu, P. Ning, and W. Du, "Group-based key pre-distribution in wireless sensor networks," in *Proceedings of ACM Workshop on Wireless Security (WiSe)*, September 2005.

[23] E. Ertin, A. Arora, R. Ramnath, and M. Nesterenko, "Kansei: A testbed for sensing at scale," in *Proceedings of the 4th Symposium on Information Processing in Sensor Networks (IPSN/SPOTS track)*, April 2006.

[24] A. Arora, E. Ertin, R. Ramnath, M. Nesterenko, and W. Leal, "Kansei: A high-fidelity sensing testbed," *IEEE Internet Computing, special issue on Large-Scale Sensor Networks*, vol. 10, no. 2, pp. 35–47, March/April 2006.

[25] D. Huang, M. Mehta, D. Medhi, and L. Harn, "Modeling pairwise key establishment for random key predistribution in large-scale sensor networks," Tech. Rep., Dept. of CSEE, University of Missouri-Kansas City, http://conrel.sice.umkc.edu/HRP/modelling pair-wise key establishment schemes-v2.pdf, March 2005.

Appendix

In this section, we will present the derivation of $P_{decouple}$ and P_{couple}, which are the probabilities that two physically neighboring nodes are able to establish a pair-wise key with arbitrary number of logical intermediate nodes and proxies in our $RKP\text{-}DE$ and the RKP protocols respectively. We will first derive some preliminaries, followed by the derivation of $P_{decouple}$ and P_{couple} using techniques in [25].

A. Preliminaries

Given the key pool size K and the key chain size k ($k \leq K$), the probability that two nodes share t keys is $P_{key}(t) = \binom{K}{t} \cdot \binom{K-t}{2(k-t)} \cdot \binom{2(k-t)}{k-t} / \binom{K}{k}^2 \quad (0 \leq t \leq k)$.

Each sensor is aware of the key sharing and physical neighborhood information in a local area, called *information area*, with size A. For ease of exposition, we assume each sensor is aware of its one hop information. Therefore $A = \pi r^2$, where r is the communication range. Note that our analysis can be applied directly to the situation where multi-hop information is available. Since n sensors are uniformly deployed at random in the network with area S, the average number of nodes in the information area is D_p, which is given by (ignoring boundary effect) $D_p = \frac{A}{S} \cdot n$. The average number of nodes in the overlapped information areas of two physically neighboring nodes is $D'_p = \frac{A'}{S} \cdot n$, where A' is the average size of the overlapped information areas of two physically neighboring nodes. Since the information area is a circle, we have $A' = \frac{(\pi - \frac{3\sqrt{3}}{4})}{\pi} A = 0.5865A$ (as given in [5]).

B. Derivation of $P_{decouple}$ in our RKP-DE protocol

We denote $P_h(i)$ as the probability that a node a can find a logical key path to a physical neighbor node b in our RKP-DE protocol within the communication range of node a with minimum logical hops i. The expression for $P_h(1)$ is given by,

$$P_h(1) = 1 - P_{key}(0) = 1 - \binom{K}{2k} \cdot \binom{2k}{k} / \binom{K}{k}^2 . \tag{13}$$

In order to analyze $P_h(i)$ $(i > 1)$, we divide the nodes in the information area of node a (except nodes a and b) into one of the groups $G(a, j)$ $(j \geq 1)$. A node s is in group $G(a, j)$ if node a can find a logical key path from itself to node s within its communication range with j logical hops, where j is minimum.

We first analyze $P_h(2)$. The probability that there are n_1 $(1 \leq n_1 \leq D_p - 1)$[5] nodes in $G(a, 1)$, given node b does not share key with node a, is $\binom{D_p-1}{n_1}(P_h(1))^{n_1}(1 - P_h(1))^{D_p-1-n_1}$. The probability that at least one of these n_1 nodes shares key with node b is $1 - (1 - P_h(1))^{n_1}$. Hence $P_h(2)$ is,

$$P_h(2) = (1 - P_h(1)) \cdot \sum_{n_1=1}^{D_p-1} \Big(\binom{D_p - 1}{n_1} (P_h(1))^{n_1}(1 - P_h(1))^{D_p-1-n_1} \cdot (1 - (1 - P_h(1))^{n_1}) \Big). \tag{14}$$

We now analyze $P_h(3)$. The probability that there are n_1 $(1 \leq n_1 \leq D_p - 2)$[6] nodes in $G(a, 1)$, given there is no key path between nodes a and b within the communication range of node a with fewer than three logical hops, is $\binom{D_p-1}{n_1} v^{n_1}(1 - P_h(1))^{D_p-1-n_1}$. The expression here is different from that in deriving $P_h(2)$ because in this case, nodes in $G(a, 1)$ do not share key with node b. Otherwise, a logical key path with fewer than three logical hops exists. Denoting v as the probability that a node shares key with node a, but does not share key with node b, given nodes a and b do not share key, we have $v = (\binom{K-k}{k} - \binom{K-2k}{k})/\binom{K}{k}$. Denote w as the probability that there is at least one node in $G(a, 2)$ given there are n_1 nodes in $G(a, 1)$, and at least one of the nodes in $G(a, 2)$ shares key with node b. Thus the minimum number of logical hops of the key path between nodes a and b is three. Then w is given by,

$$w = \sum_{n_2=1}^{D_p-1-n_1} \Big(\binom{D_p - 1 - n_1}{n_2} \Big(1-(1-P_h(1))^{n_1}\Big)^{n_2} \Big((1-P_h(1))^{n_1}\Big)^{D_p-1-n_1-n_2} \cdot (1-(1-P_h(1))^{n_2}) \Big). \tag{15}$$

[5]Node b is not in $G(a, 1)$. So n_1 can be $D_p - 1$ at most.

[6]Node b is not in $G(a, 1)$, and at least one other node is in $G(a, 2)$ (thus not in $G(a, 1)$). So n_1 can be $D_p - 2$ at most.

Finally, the expression for $P_h(3)$ is given by,

$$P_h(3) = (1 - P_h(1)) \cdot \sum_{n_1=1}^{D_p-2} \Big(\binom{D_p - 1}{n_1} v^{n_1} (1 - P_h(1))^{D_p-1-n_1} \cdot w \Big). \tag{16}$$

The derivation of $P_h(l)$ $(l > 3)$ is similar to $P_h(3)$. We denote $y(i)$ $(2 \leq i \leq l-2)$ as the probability that there is at least one node in $G(a,i)$ given n_j $(1 \leq j \leq i-1)$ nodes in $G(a,j)$[7], and denote z as the probability that at least one node is in $G(a, l-1)$ and shares key with node b. We then have,

$$\begin{aligned} y(i) \ &= \sum_{n_i=1}^{D_p-1-\sum_{j=1}^{i-1} n_j-(l-1-i)} \Big(\binom{D_p - 1 - \sum_{j=1}^{i-1} n_j}{n_i} \Big((1-(1-P_h(1))^{n_{i-1}})(1-P_h(1)) \Big)^{n_i} \\ &\qquad \cdot \Big((1-P_h(1))^{n_{i-1}} \Big)^{D_p-1-\sum_{j=1}^{i} n_j} \Big), \end{aligned} \tag{17}$$

$$\begin{aligned} z \ &= \sum_{n_{l-1}=1}^{D_p-1-\sum_{j=1}^{l-2} n_j} \Big(\binom{D_p - 1 - \sum_{j=1}^{l-2} n_j}{n_{l-1}} \Big(1-(1-P_h(1))^{n_{l-2}}\Big)^{n_{l-1}} \Big((1-P_h(1))^{n_{l-2}}\Big)^{D_p-1-\sum_{j=1}^{l-1} n_j} \\ &\qquad \cdot (1-(1-P_h(1))^{n_{l-1}}) \Big). \end{aligned} \tag{18}$$

The general form of $P_h(l)$ $(l > 3)$ is given by,

$$P_h(l) = (1 - P_h(1)) \cdot \sum_{n_1=1}^{D_p-1-(l-2)} \Big(\binom{D_p - 1}{n_1} v^{n_1} (1 - P_h(1))^{D_p-1-n_1} \cdot \prod_{i=2}^{l-2} y(i) \cdot z \Big). \tag{19}$$

We have the following definitions. We define $P_{decouple}(A_a)$ as the probability that node a is able to find a key path to its physical neighbor node b within its communication range ($A_a = \pi r^2$). We define $P_{decouple}(A')$ as the probability that node a is able to find a key path to node b within the overlapped communication ranges of nodes a and b ($A' = 0.5865 \cdot \pi r^2$[5]). We define $P_{decouple}(\bar{A}')$ as the probability that node a is *not* able to find a key path to node b within the overlapped communication ranges of nodes a and b, and define $P_{decouple}(A_a|\bar{A}')$ as the probability that node a is able to find a key path to node b within its communication range given node a cannot find a key path to node b within the overlapped communication ranges of nodes a and b. By the law of total probability, we have,

$$P_{decouple}(A_a) = P_{decouple}(A') \cdot P_{decouple}(A_a|A') + P_{decouple}(\bar{A}') \cdot P_{decouple}(A_a|\bar{A}'). \tag{20}$$

In the above equation, $P_{decouple}(A_a)$ is given by $P_{decouple}(A_a) = \sum_{i=1}^{\infty} P_h(i)$. Similarly, $P_{decouple}(A')$ is given

[7]None of the nodes in $G(a,i)(1 \leq i \leq l-2)$ shares key with node b, otherwise a key path with fewer than l logical hops exists.

by $P_{decouple}(A') = \sum_{i=1}^{\infty} P_h(i|A')$, where $P_h(i|A')$ denotes the probability that node a can find a logical key path to a physical neighbor node b in our RKP-DE protocol within the overlapped communication ranges of nodes a and b with minimum logical hops i. The expression of $P_h(i|A')$ is the same as that of $P_h(i)$ except that D_p in $P_h(i)$ needs to be replaced by D'_p. The value of $P_{decouple}(A_a|A')$ is always one since a key path within the overlapped communication range (A') is certainly within the communication range of node a (A_a). Therefore, we can obtain the expression of $P_{decouple}(A_a|\bar{A}')$ by,

$$\begin{aligned} P_{decouple}(A_a|\bar{A}') &= (P_{decouple}(A_a) - P_{decouple}(A'))/P_{decouple}(\bar{A}') \\ &= (\sum_{i=1}^{\infty} P_h(i) - \sum_{i=1}^{\infty} P_h(i|A'))/(1 - \sum_{i=1}^{\infty} P_h(i|A')). \end{aligned} \tag{21}$$

Finally, we can give the expression of $P_{decouple}$ by,

$$\begin{aligned} P_{decouple} &= 1 - (1 - P_{decouple}(A_a)) \cdot (1 - P_{decouple}(A_b|\bar{A}_a)) \\ &= 1 - (1 - P_{decouple}(A_a)) \cdot (1 - P_{decouple}(A_a|\bar{A}')) \\ &= 1 - (1 - \sum_{i=1}^{\infty} P_h(i)) \cdot (1 - (\sum_{i=1}^{\infty} P_h(i) - \sum_{i=1}^{\infty} P_h(i|A'))/(1 - \sum_{i=1}^{\infty} P_h(i|A'))). \end{aligned} \tag{22}$$

C. Derivation of P_{couple} in the RKP protocol

We denote $P'_h(i)$ as the probability that a sensor a can find a logical key path to a physical neighbor node b in the RKP protocol within the communication range of node a with minimum logical hops i. The main difference in the analysis of the RKP protocol from that of the RKP-DE protocol comes from the fact that two successive nodes on the logical key path in RKP protocol have to be physical neighbors, unlike that in the RKP-DE protocol. Due to space limitation, we skip the derivation of $P'_h(i)$ $(i \geq 1)$. Interested readers are referred to [25] for details.

Similar to the derivation in Appendix.B, the expression of P_{couple} is given by,

$$\begin{aligned} P_{couple} &= 1 - (1 - P_{couple}(A_a)) \cdot (1 - P_{couple}(A_b|\bar{A}_a)) \\ &= 1 - (1 - P_{couple}(A_a)) \cdot (1 - P_{couple}(A_a|\bar{A}')) \\ &= 1 - (1 - \sum_{i=1}^{\infty} P'_h(i)) \cdot (1 - (\sum_{i=1}^{\infty} P'_h(i) - \sum_{i=1}^{\infty} P'_h(i|A'))/(1 - \sum_{i=1}^{\infty} P'_h(i|A'))). \end{aligned} \tag{23}$$