RINK-RKP: A Scheme for Key Predistribution and Shared-Key Discovery in Sensor Networks

Manish Mehta Dijiang Huang Lein Harn

Computer Science and Electrical Engineering
University of Missouri - Kansas City
Kansas City, Missouri 64110, USA.
{manish.mehta, dhuang, harnl}@umkc.edu

Abstract

Efficient schemes for key predistribution and shared-key discovery play a vital role in security and efficiency of pairwise key establishment in sensor networks. In this paper, we propose a scheme for key predistribution using hash-chain and subsequent shared-key discovery. We show potential active attacks on sensor networks due to key predistribution which can have severer consequences as compared to attacks described in existing proposals. We also show that as compared to the existing schemes, our scheme is more resilient to these active attacks.

1 Introduction

Sensor networks are composed of a large number of low-power sensor devices. For secure communication among the sensors, pairwise keys are needed to be established between each pair of communicating sensors. Recent proposals [1, 2] use Random Key Predistribution (RKP) to achieve the goal. In RKP schemes, each sensor is preloaded with certain number of keys. The sensors can communicate with each other securely if they share at least one key. The probability of two sensors sharing at least one key is proportional to the number of preinstalled keys in each sensor.

The wireless nature of communication among sensors makes sensor networks vulnerable to passive and active attacks. Also, for many applications, the low-cost sensors are deployed in unattended environments which make them physically insecure. Due to the low-cost design, the sensors are not considered to be tamper-proof devices. Physical capture of sensors may lead to severe security problems. One of the goals of a secure scheme for pairwise key establishment is to minimize the effect of physical node capture in sensor networks.

In this paper, we introduce a scheme for RKP and subsequent shared-key discovery. The main idea of our scheme is to define RINK (Relationship between the node ID aNd the Keys possessed by each sensor). We use node id of each sensor to determine the keys to be preinstalled in that sensor. The RINK alleviates the security risks due to node capture by restricting the ability of an attacker to fabricate fake sensor nodes. Further, unlike the existing schemes in [1] and [2], this design obviates the need for transmission of all key identifiers during shared-key discovery phase; and unlike the scheme proposed in [3], this design does not require computationally expensive operations for shared-key discovery.

2 Background of RKP Schemes

2.1 Phases in RKP Schemes

Key predistribution phase: A centralized key server first generates a large key pool offline. Keys from this key pool are distributed as follows: 1. Assign a unique node identifier or key ring identifer to each sensor 2. Select m different keys for each sensor from the key pool to form a key ring 3. Load the node identifier and the key ring into memory of the sensor.
Sensor deployment phase: The sensors are randomly picked and uniformly distributed in a large area. Typically, the number of sensors in communication range (neighbors) of a sensor (n') is much smaller than the total number of deployed sensors (N).
Shared-key discovery phase: During the shared-key discovery (SKD) phase, each sensor attempts to find other sensors in its communication range. A set of neighbors (W) is maintained by each sensor. It then attempts to discover shared key(s) with them. Each sensor builds a *key graph* (see Definition 1) according to its view of the network. Next, each sensor shares its key graph with other sensors and updates its key

0-7803-8991-3/05/$20.00 © 2005 IEEE

graph according to the key graphs from other sensors.

Pairwise key establishment phase: If a sensor discovers shared key(s) with a given neighbor, the shared key(s) can be used as their pairwise key(s). If a sensor does not share required key(s) with a given neighbor, the sensor uses the *key graph* built during SKD phase to find a *key path* (see Definition 2) to set up the pairwise key for future communication.

Definition 1 (Key graph) A *key graph* maintained by node i is defined as $G_i = (V_i, E_i)$ where, $V_i = \{j | j \in W_i \vee j = i\}$, $E_i = \{e_{jk} | j, k \in W_i \wedge (j \; \mathcal{S} \; k)\}$, and $\mathcal{S}$ is a relation defined between two nodes if they discover shared key(s) during the SKD phase.

Definition 2 (Key path) A *key path* between node A and B is defined as a sequence of nodes A, N_1, N_2,..., N_i, B, such that, each pair of nodes (A, N_1), (N_1, N_2), ..., (N_{i-1}, N_i), (N_i, B) has discovered shared key(s) during the key discovery phase. The *length* of the *key path* is the number of pairs of consecutive nodes in the sequence.

2.2 Related Work

The first P-RKP scheme was proposed by Eschenauer and Gligor [1], and we refer to it as the basic scheme. The proposals that followed the basic scheme suggested improvements in terms of security. Chan et al. proposed the q-composite scheme in [2]. In this scheme, the shared-key threshold is set to a variable q. To form a secure link between two sensors, the scheme requires them to share at least q keys. The scheme proposed by Du et al. [3] and Grid-based scheme proposed by Liu and Ning [4] change the unstructured key pool in earlier schemes to a structured key pool by dividing the unstructured key pool into multiple key spaces. We refer to these schemes as Structured-Key-pool RKP (SK-RKP) schemes. Within each key space, the key structure uses the group key scheme proposed by Blom [5] and further developed by Blundo et al. [6].

The Grid-based scheme is equivalent to the scheme proposed in [3] in that it uses polynomials instead of key spaces. Recently, Peitro et al. presented a pseudo-random key predistribution in [7]. This scheme uses a pseudo-random function for predistribution of keys.

3 RINK-RKP

In this section, we introduce RINK-RKP, a new scheme for random key predistribution and subsequent shared-key discovery in sensor networks.

3.1 Key Predistribution

The key predistribution phase for RKP scheme introduced in [1] and adopted by Chan et al. [2] does not define any relationship between the node id and the keys possessed by each sensor.

The main idea behind our approach is to define a relationship between the node id and the keys possessed by each sensor while maintaining the required randomness in choice of keys. Our scheme requires the key predistribution phase to first choose a unique identifier for each sensor node. To determine the keys to be installed in the sensor, we use a secure one-way hash function as defined in [8]. It may be noted that any pseudo-random function which can produce output uniformly distributed in a given range for given input set can be used.

Before describing the use of one-way hash functions for RINK-RKP, we summarize the notations used in the rest part of the paper as follows: N is the total number of sensors to be preloaded, n is the total number of sensors to be deployed for a sensor network and $n \leq N$, n' is the number of sensors in the communication range of a sensor (i.e. neighbors), id is a unique sensor identifier, where $id = 1, 2, \ldots, N$, s_{id} is the sensor with unique identifier value id, P is size of the key pool, m is the number of keys loaded in each sensor, q is the required number of common keys between sensors in P-RKP, $||$ is concatenation operator, $\{x\}^j$ means x concatenated with itself j times, $H(M)$ is a secure one-way hash function on M producing z-bit output, k_i^{id} is the identifier of i^{th} key for s_{id}, i=1, 2, ..., m, Y_i^{id} is the actual key value for key identifier k_i^{id} (e.g. 128-bit, 256-bit key etc.), KC_{id} is the set of all key identifiers (k^{id}) for s_{id}, $\bigoplus\{A\} = \{a_1 \oplus a_2 \oplus \ldots \oplus a_{|A|} \mid a_i \in \mathrm{A}$, for $i = 1, 2, \ldots, |A|\}$, $\oplus$ is bitwise XOR, K_{ij} is the discovered key between s_i and s_j.

For key predistribution in RINK-RKP, we require key server to first generate keys (Y's) and their identifiers (k's) in a key pool of size P. The pseudo code in Procedure 1 shows our key predistribution method.

This scheme generates a chain of practically random numbers by taking the unique node id as the seed. In turn, it binds the node id with the set of keys the node possesses. This procedure is followed for each of the N sensors to be preloaded. By including the *PrevKey* and node id along with the SHA-1 [9] output as input to the next hash operation, we make the probability of merging of chains of two different sensors negligible.

3.2 Shared-Key Discovery Scheme

The shared-key discovery (SKD) phase is the next phase after deployment of the sensors. In this phase, each sensor attempts to find other sensors in its range and discovers possible shared-keys with them.

As shown in Procedure 2, in RINK-RKP, a sensor, say s_i, initiates this phase by broadcasting its identi-

Procedure 1 (Key Predistribution)

1.	Pick a unique identifier id for a sensor.
	Initialize $Output = \{1\}^z$
2.	for $i = 1$ to m
2.1.	if $(i \leq 2)$
2.1.1.	then $PrevKey = 0$
2.1.2.	else $PrevKey = k_{i-2}^{id}$
2.2.	$M_i^{id} = Output \mid\mid PrevKey \mid\mid id$
2.3.	$Output = H(M_i^{id})$
2.4.	if$(Output \geq 2^z - 1 - (2^z \bmod P))$
2.4.1.	then $i = i - 1$, goto step 2
2.5.	$Key = Output \bmod P$
2.6.	if (Key has already been generated for this sensor)
2.6.1.	then $i = i - 1$, goto step 2
2.7.	$k_i^{id} = Key$
2.8	add k_i^{id} to KC_{id}
3.	Store id, KC_{id}, and corresponding keys (Y's) in the sensor.

Procedure 2 (Shared-Key Discovery)

1.	Broadcast node identifier, i
2.	Receive node ids transmitted by neighbors and generate set of neighbors, W. Generate *key graph* with elements of W as vertices.
3.	for $(\forall\ j \in W)$ do
3.1.	Generate KC_j using the node identifier, j, as the input to Procedure 1
3.2.	Generate Q, a set of key values (Ys) corresponding to common key identifiers in KC_i and KC_j.
3.3.	if$(\lvert Q\rvert \geq q)$ then
3.3.1.	Compute the shared-key K_{ij} as $\bigoplus\{Q\}$
3.3.2.	Add a link between node i and node j in the *key graph*
4.	Stop.

fier, i. Similarly, it receives the ids from its neighbors. At this point, the sensor builds a *key graph* with ids of all the neighbors as vertices. For each of the neighbors, say s_j, the sensor generates the key chain KC_j using the neighbor's identifier j as shown in Procedure 1. It now finds the key identifiers shared between KC_i and KC_j. If there exists at least q shared key identifiers, a key is said to be discovered between s_i and s_j. From the shared key identifiers, s_i builds a set, Q, of the corresponding key values (Ys). Now, the value of the discovered key K_{ij} is computed as $\bigoplus\{Q\}$. After this, s_i adds an edge between s_i and s_j in its *key graph*.

4 Security Analysis

RINK-RKP is closest to the schemes proposed in [1, 2, 3]. In their works, they analyze the security of sensor networks on the basis of fraction of the communication links compromised due to captured sensors. In existing schemes, the security analysis is done on the basis of random capture of sensors. However, in practice, an attacker can selectively capture sensors to learn the keys in a quicker fashion. We analyze the existing schemes under selective node capture. In the following, we introduce active attacks on sensor networks due to node capture and analyze the security of the schemes under active attacks. Specifically, we introduce and analyze node replication and node fabrication attacks.

4.1 Selective node capture attack

In all current RKP proposals, the sensors are assumed to be captured randomly. But in practice, the random capture assumption is too weak. The attacker can purposely attack certain area or a group of sensors. Thus, an attacker can purposely locate and capture the sensors which can give more information about the sensor network. For example, in P-RKP scheme, each sensor broadcasts its list of keys. An attacker can selectively attack a sensor that possesses the most number of keys that are not already compromised. In the best case for the attacker, for a key pool of size P and the m keys in each sensor, the attack can compromise all communication links by capturing $\lceil P/m \rceil$ sensors. In practice, an attacker can inspect all keys possessed by sensors and find the minimal cover set which contains the minimal number of sensors that can cover the maximum number of keys in the key pool. Alternatively, a less powerful attacker can use heuristic technique to choose the next node to capture. However, due to the purely random selection of keys in P-RKP schemes, the attacker does not gain significantly more information using selective capture attack as compared to random capture. Similarly, for RINK-RKP, the gain due to selective capture attack over random capture attack is not significant as the keys are practically randomly selected for each node.

As compared to the P-RKP schemes, the selective attack on SK-RKP scheme can cause severer problems. This is due the fact that in SK-RKP scheme, the nodes derive a shared key if they share a key space. As the number of key spaces is generally much smaller than number of individual keys used in P-RKP to derive a shared key, the attacker has better selection criterion. In SK-RKP scheme, each sensor broadcasts its node id and the key-space ids in order to discover shared-key with its neighbors. At the same time, the node id and

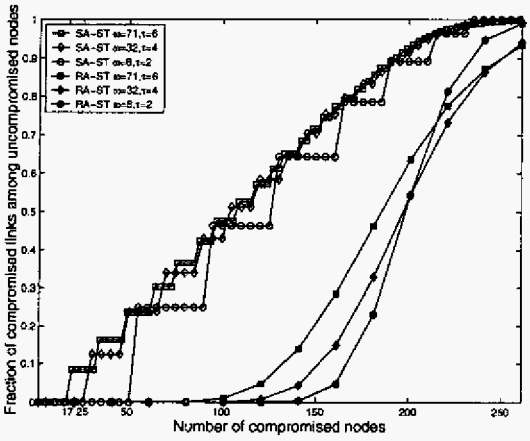

Figure 1: **Selective node capture attack on SK-RKP scheme [3] with m = 100, $p_{connect}$ = 0.432**

key-space *ids* can be extremely helpful to the attacker to launch selective attack. The attacker can selectively capture the sensors that possess keys within the same key space. Once $\lambda + 1$ sensors in a key space are compromised, all the keys in that key space are compromised. In this fashion, an attacker can incrementally capture the sensors that use same key space. Since sensors possess keys from more than one key space, the number of sensors required to be captured to compromise subsequent key spaces is less. We use $c(i)$ to represent the average number of additional sensors to be captured in order to compromise a key space when $i-1$ key spaces are already compromised. In order to compromise the first key space, the attacker needs to capture at least $\lambda + 1$ nodes (i.e. $c(1) = \lambda + 1$). Since each sensor is allocated τ key spaces ($\tau \geq 2$), a captured node also uses an uncompromised key space with probability $p' = \frac{\tau-1}{\omega-1}$. Thus, to compromise i^{th} key space, we have

$$c(i) = \lambda + 1 - \sum_{k=1}^{i-1} c(k) \cdot p', \qquad 2 \leq i \leq \omega \qquad (1)$$

Security analysis for SK-RKP scheme proposed by Du et al. in [3] assumes random node capture in sensor networks. In Fig. 1, we use (1) to show SK-RKP scheme [3] under selective node capture attack (SA) and compare it with that under random node capture attack (RA). As shown in the figure, the robustness (threshold) of SK-RKP scheme decreases dramatically under selective node capture attack.

4.2 Active attacks using captured nodes

Since sensors are low-cost devices and operate in unattended environment for many applications, they cannot be considered tamper-resistant. Under some practical assumptions about capabilities of the attacker, we now describe two related active attacks on sensor networks due to captured nodes.

Node replication attack: In this attack, the attacker captures a sensor and clones it as per requirement. Since the attacker is assumed to have the ability to listen to the traffic in the network, the attacker can deploy the clones in the other parts of the network. Due to lack of *a-priori* knowledge of post-deployment configuration, the uncompromised sensors in the other parts of the network cannot detect the cloned sensor as an anomalous sensor. This attack can have severer consequences as compared to the passive listening attacks on links between uncompromised nodes.

Node fabrication attack: In this attack, the attacker captures sensors and fabricates fake sensors using the information gathered from captured nodes. Similar to the node replication attack, the attacker can deploy the fabricated nodes in the parts of the network where the original sensor is not present. The uncompromised sensors in the network cannot detect the fabricated nodes as anomalous nodes as long as they can have expected communication with them. This attack is severer as compared to node replication attack as the attacker may have enough information to fabricate multiple sensors in order to inject, sink, modify, and reroute the sensed data.

Since node replication is a special case of node fabrication, we analyze the schemes for node fabrication attack in general. The basic aim of the attacker launching this attack is to fabricate fake nodes and deploy them in the existing system. The more the number of uncompromised nodes that can be used by the fake nodes to get connected to the network, the faster the attacker can take control over the network.

Fig. 2 shows the node fabrication attack on different schemes. In the P-RKP schemes, by capturing less than 10 nodes the attacker can gather enough information to fabricate fake nodes that can establish connection with most of the uncompromised nodes. This is possible in P-RKP because there is no defined relationship between the node *id* and the keys possessed by each sensor. By capturing only a few nodes, the attacker can fabricate fake nodes with identity of his choice with the same set of keys. For example, by capturing two nodes, the attacker can fabricate and deploy approximately $\binom{2m}{m}$ fake nodes. These nodes possess valid keys and hence cannot be detected. Unlike P-RKP schemes, the SK-RKP scheme and our scheme bind the possessed keys with the node *id* of sensors. As a result, the attacker has to capture significantly more

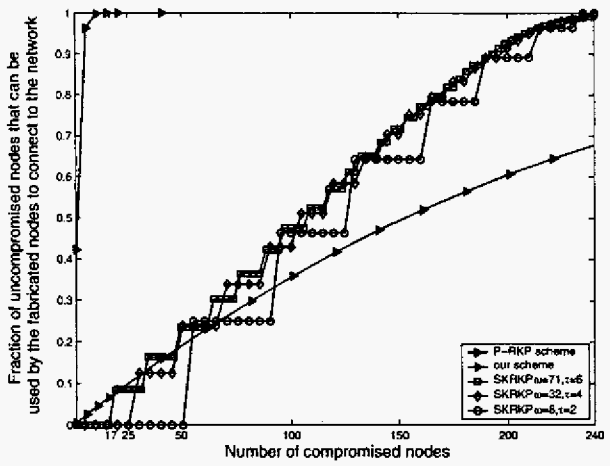

Figure 2: **Active attacks : Node fabrication attack (m = 200)**

number of sensors to achieve the same goal in SK-RKP and our scheme. In SK-RKP scheme, the attacker can incrementally compromise key spaces and fabricate fake sensors using the compromised key space. Equation (1) can be used to determine the number of additional sensors required to be captured in order to compromise each additional key space. In our scheme, in order for a fabricated node to get connect to the network via an uncompromised node, the node needs to satisfy the following conditions: 1. It should share required number of keys (q) with the uncompromised node. 2. Given that the first condition is satisfied, all the shared-keys must be already known to the attacker. The probability that a fabricated node will satisfy these conditions with x nodes already captured can be computed as:

$$p_f(x) = \frac{1}{p_{connect}} \sum_{i=q}^{m} \frac{\binom{P}{m}\binom{m}{i}\binom{P-m}{m-i}}{\binom{P}{m}^2} \left(\frac{C_x}{P}\right)^i \quad (2)$$

Where,

$$C_x = \left[1 - \left(1 - \frac{m}{P}\right)^x\right] \cdot P$$

Where x is the number of captured nodes, $[1 - (1 - \frac{m}{P})^x]$ is the fraction of keys that are compromised due to capture of x nodes, and C_x is the number of keys compromised due to capture of x nodes. As shown in Fig. 2, our scheme performs significantly better than P-RKP schemes. As compared to SK-RKP scheme, our scheme provides more robust security after a small threshold. As compared to the existing P-RKP schemes that use unstructured key pool, the ability of our scheme to resist node fabrication attack is significantly more. Unlike in P-RKP schemes, the binding between the node *id* and the possessed keys in SK-RKP scheme and our scheme largely restricts the ability of the attacker to fabricate fake nodes with identities of his choice.

5 Conclusion

In this paper, we identify a limitation in one of the existing schemes and propose a new scheme for RKP in sensor networks. In all existing proposals, the security analysis is based on random capture of sensors. By analyzing the robustness of SK-RKP scheme under selective node capture attack, we show that the assumption of random node capture is weak in practice. Also, we analyze active attacks on sensor networks due to node capture and compare our scheme with the existing schemes under these attacks.

References

[1] L. Eschenauer and V. D. Gligor, "A key-management scheme for distributed sensor networks," in *Proc. of ACM CCS*, 2002, pp. 41–47.

[2] H. Chan, A. Perrig, and D. Song, "Random key predistribution schemes for sensor networks," in *Proc. of the IEEE Symposium on Security and Privacy*, 2003, pp. 197–215.

[3] W. Du, J. Deng, Y. S. Han, and P. K. Varshney, "A pairwise key pre-distribution scheme for wireless sensor networks," in *Proc. of ACM CCS*, 2003, pp. 42–51.

[4] D. Liu and P. Ning, "Establishing pairwise keys in distributed sensor networks," in *Proc. of ACM CCS*, 2003, pp. 52–61.

[5] R. Blom, "An optimal class of symmetric key generation systems," in *Proc. of the EUROCRYPT 84*, 1985, pp. 335–338.

[6] C. Blundo, A. D. Santis, A. Herzberg, S. Kutten, U. Vaccaro, and M. Yung, "Perfectly-secure key distribution for dynamic conferences," in *Proc. of the 12th Annual International Cryptology Conference on Advances in Cryptology*, 1993, pp. 471–486.

[7] R. D. Pietro, L. V. Mancini, and A. Mei, "Efficient and resilient key discovery based on pseudo-random key pre-deployment," in *Proc. of IPDPS'04*, 2004.

[8] B. Schneier, *Applied cryptography: protocols, algorithms, and source code in C*, 2nd ed. John Wiley and Sons, Inc., 1995.

[9] "Secure hash standard," FIPS PUB 180-1, National Institute of Standards and Technology, U.S. Department Of Commerce, April 1995.