POLISH: Proactive co-Operative LInk Self-Healing for Wireless Sensor Networks

Tatsuro Iida, Atsuko Miyaji and Kazumasa Omote

Japan Advanced Institute of Science and Technology (JAIST)
{s0910001,miyaji,omote}@jaist.ac.jp

Abstract. In this paper we propose the first proactive co-operative link self-healing (POLISH) scheme, in which the secure link compromised in WSNs automatically self-heals with time, without the help of a server. Our scheme updates a secure link using the random data transmitted from the neighboring sensor nodes, based on the idea of the POSH scheme. It is necessary to newly take the security of a link between sensors into consideration in our scheme since such security is not considered in the POSH scheme. We conduct analytical evaluation and a simulation experiment for our scheme, and the results indicate that our scheme is very effective in self-healing.

1 Introduction

Wireless Sensor Networks (WSNs) consist of small, battery-operated, limited memory and limited computational power sensor nodes. Hence, most of existing pairwise key establishment schemes in WSNs are not based on public key cryptography. One of the most popular schemes, referred to as *RKP* (Random Key Pre-distribution) in this paper, was first proposed by Eschenauer and Gligor [4] and has been applied to many schemes. These basic probabilistic schemes are pairwise key pre-distribution schemes based on symmetric key cryptography. However, the security of the whole network in such schemes degrades with time when there is an attacker. An attacker who corrupts several sensors can obtain a set of the pairwise symmetric keys. If the attacker is continuously corrupting sensors, they will eventually learn all the pairwise symmetric keys, and all newly deployed sensors will establish links that will immediately be compromised. This is a non-desirable property.

The WSNs are usually deployed to operate for a long period of time. Availability is very important to long-term use of WSNs under the presence of an attacker. Actually, we can find several schemes [2, 5, 6, 10], which maintain availability of the secure link. Link composed of a pairwise symmetric key in WSNs. These schemes are called *resilient multiphase WSNs*, in which a link self-heals against node-capture attacks by redeploying a sensor node when the battery of a sensor is depleted. However, as far as we know, any efficient scheme which maintains availability of the secure link between sensor nodes requires the help of a server. It is thus desirable that the link *self-heal* against node-capture attacks to

maintain availability without the help of a server. Self-healing of a secure link is the property that the compromised link recovers in a WSN.

In this paper, we propose the first proactive co-operative link self-healing (POLISH) scheme, in which the secure link compromised in a WSN automatically self-heals with time, without the help of a server. Our scheme updates a secure link using the random data transmitted from the neighboring sensor nodes, based on the idea of the POSH scheme. The POSH scheme self-heals the secret key for encrypting the sensed data on a sensor node for the purpose of data survival. In our scheme, a link self-heals in two steps: first, two neighboring sensors are self-healed, and then the link between these sensors is self-healed. It is necessary to newly take the security of a link between sensors into consideration in our scheme since such security is not considered in the POSH scheme. Furthermore, our scheme has an advantage that the probability of establishing a secure link is 100%. In addition, we conduct analytical evaluation and a simulation experiment for our scheme, and the results indicate that the proposed scheme is very effective in self-healing. Our scheme is both effective and efficient, as supported by analytical and simulation results.

The rest of this paper is organized as follows. In the next section we present related work on pairwise key distribution schemes with self-healing property for WSNs. Some preliminaries are provided in Section 3, and we review the POSH scheme in Section 4. We explain our scheme in detail in Section 5, analyze its security and efficiency in Section 6, and compare the POLISH scheme with the previous RoK scheme in Section 7. We finally conclude this paper in Section 8.

2 Related Work

One of the most popular schemes, referred to as *RKP* in this paper, was proposed by Eschenauer and Gligor [4], which has been applied to many schemes. These probabilistic pairwise key pre-distribution schemes are efficient because they are not based on public key cryptography. However, these schemes do not have self-healing feature of a link, and thus the ratio of the compromised links reaches 100% as time passes against node-capture attacks.

Castelluccia and Spognardi [2] have proposed the RKP scheme with self-healing property, named *RoK* scheme, for *multiphase WSNs*, in which a link self-heals against node-capture attacks by redeploying a sensor node (with server's help) when the battery of a sensor is depleted. The RoK scheme improves the security of the RKP scheme by limiting the lifetime of the keys, and by refreshing keys. Some recent schemes improve the resiliency of the RoK scheme. Yilmaz et al. [10] proposed a more resilient scheme than the RoK scheme to speed up the self-healing process. Kalkan et al. [6] proposed a zone-based RKP (Zo-RoK) scheme which combines the best parts of Du et al.'s scheme [3] and the RoK, and improves the resiliency of the RoK. Furthermore, Ito et al. [5] proposed a strongly-resilient polynomial-based random key pre-distribution scheme for multiphase WSNs (RPoK): a private sub-key is not directly stored in each sensor node by applying the polynomial-based scheme to the RoK scheme.

There is another drawback in the RKP scheme; the probability of establishing a secure link is not 100%. We recall the basic pairwise key pre-distribution scheme, *polynomial-based key pre-distribution scheme* [1] which was proposed prior to the RKP scheme and which maintained the probability of establishing a secure link at 100%. To pre-distribute pairwise keys in the polynomial-based key pre-distribution scheme, a setup server randomly generates t-degree $f(x, y)$ over a finite field $\mathbb{F}_q$, where it has the symmetrical property of $f(x, y) = f(y, x)$.

As for self-healing of the secret key for the purpose of data survival, the POSH scheme [8] and the DISH scheme [7] have been proposed by Pietro et al. and Ma et al., respectively. These schemes use key evolution and sensor cooperation to self-heal the secret key which encrypts the sensed data on a sensor node, for the purpose of data survival. These schemes involve each sensor sharing an initial key with the sink (base station). At any time, sensors are either occupied (red), sick (yellow) or healthy (green). The self-healing of a sensor means that a sick sensor becomes healthy. The POSH and the DISH schemes update a secret key using the random data transmitted from other sensor nodes. That is, if at least one of the sensor nodes which send random data is not corrupted, the compromised secret key is updated and then is self-healed.

3 Preliminaries

3.1 Notation

n : Total number of sensors (i.e., Size of network)
s_i : Sensor i
ID_i : Index of s_i
m : Number of links with neighboring sensors
r : Round index (i.e., fixed-length time slot)
$K_{i,j}^r$: Pairwise symmetric key (secure link) between s_i and s_j at round r
S_i^r : Seed of s_i at round r
$c_{i_\ell}^r$: ℓ-th *contribution* received by s_i at round r
G^r : Set of green sensors at round r
Y^r : Set of yellow sensors at round r
R^r : Set of red sensors ($= k$) at round r
GL^r : Set of green links at round r
RL^r : Set of red links at round r
q : Large prime number
H : Secure hash function $H : \{0,1\}^* \rightarrow \{0,1\}^q$
$f(x, y)$: Bivariate t-degree polynomial at over a finite field $\mathbb{F}_q$

3.2 Requirements

The following requirements need to be considered when designing a link self-healing scheme in WSNs.

Highly-secure connectivity. After deployment, two sensors share a pairwise symmetric key to establish a secure link. This probability is called secure connectivity. Highly-secure connectivity is required in a pairwise symmetric key scheme in WSNs.
High resiliency. Sensor nodes may be deployed in public or hostile locations in many applications. The resiliency (self-healing) means that the ratio of compromised links is suppressed low even if the adversary regularly/continuously corrupts sensor nodes of the network. This feature is achieved by security properties, *forward and backward secrecy*[1]. Resiliency is estimated by the ratio of links that are not compromised by the capture of nodes.
Restricted resources. It is required that the WSNs consist of small, battery-operated devices with limited memory and limited computational power.

3.3 System and Network assumptions

Time is divided into equal and fixed rounds. Round synchronization can be implemented. The network is connected at all times. Any two sensors can communicate either directly or indirectly, via other sensors. Each sensor can perform cryptographic hashing and polynomial execution and has a unique ID. Also, each sensor has a Pseudo-Random Number Generator (PRNG) initialized with a unique secret seed. A sensor re-initializes secret seed values in any round.

3.4 Adversarial Model

We refer to the adversary as ADV from here on. ADV's main goal is to learn as many sensor secrets (keys or other key material) as possible, and hence ADV is only interested in learning the secrets of sensors when it compromises/captures. ADV knows the entire topology of the WSNs. Such adversary model is usually employed in the previous schemes [4, 2, 10, 6, 5]. ADV can create a table of sensor secrets and share it. This might be later used to decrypt encrypted communication. Furthermore, ADV does not stay at one local place for stealthy operation and then does not interfere with sensor's behavior, i.e., it does not delete, delay or introduce messages. ADV also eavesdrops on communication from the sensor through wireless transmission. Note that ADV leaves no trace behind (e.g., he does not establish an sniffing tool somewhere in WSNs).

Time is divided into equal and fixed rounds. At the end of each round, ADV randomly picks a subset of k sensors to be compromised in the following round (ADV preferentially aims at a green sensor). At the start of each round, the ADV releases the subset from the previous round and compromises the new subset. ADV is unable to monitor and record all communication at the same time as described in [7, 8].

[1] These security properties are defined in [8]. Forward secrecy means that ADV cannot learn any keys used to decrypt and/or authenticate before compromise, and backward secrecy means that ADV cannot learn any keys used to decrypt and/or authenticate after compromise.

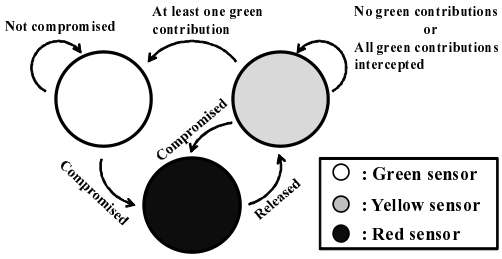

Fig. 1. Sensor state transition diagram [8].

4 The POSH Scheme

4.1 Overview

Pietro et al. have proposed a proactive co-operative self-healing (POSH) scheme for data survival on a sensor. A sensor s_i encrypts its sensed data in round r by using secret key K_i^r. A sensor whose current key is known to ADV can regain security and compute a new key unknown to ADV, if it obtains at least one "infusion" of secure randomness from a peer sensor whose randomness is not currently compromised. Each sensor shares a secret key K_i^1 with the sink in the first round 1. ADV breaks into $k = |R^r|$ sensors and reads all keys. At any time, we identify three sets of sensors:

- *Red sensors* (R^r) are currently occupied by ADV in round r.
- *Yellow sensors* (Y^r) are those that have been compromised in some round $r' < r$, and their current keys are known to ADV in round r.
- *Green sensors* (G^r) are those that have either never been compromised, or which have regained their security by round r.

The main point of the POSH scheme is for sensors, at each round, to provide each other with contributions derived from their PRNG-s. Each sensor, having received some such contributions, uses them together with its prior keys to compute a key for the next round. Specifically, each sensor produces a certain number of contributions and recipient IDs using its PRNG, and sends each value to them. In more detail, to update its key at the end of round r, s_i computes:

$$K_i^{r+1} = H(K_i^r || c_{i_1}^r || ... || c_{i_\sigma}^r),\ r \geq 1, \tag{1}$$

where σ is the number of received contributions, and $c_{i_\ell}^r$ is the ℓ-th contribution received during current round. This key evolution holds both forward and backward security. Note that all contributions generated by red and yellow sensors are known to ADV. On the other hand, contributions by green sensors are unknown to ADV. Thus, if a yellow sensor receives a single contribution from a green sensor, ADV cannot learn the former's next key. Note that a green sensor cannot become a yellow sensor directly. The state transition diagram of a single sensor is shown in Fig. 1.

4.2 Boundary of Security Evaluation in POSH

To evaluate the healing rate of secret key for data encryption, the POSH scheme analyzes the number of green sensors in any round. The secret key K_i^r is used as a secure link between a sensor s_i and the sink at round r since the sink knows all the secret keys of sensors. However, we cannot directly achieve the secure link between sensors by the POSH scheme, since the security of a link between sensors is not considered in the POSH scheme.

5 The Proposed Scheme

In this section we propose the POLISH (Proactive co-Operative LInk Self-Healing) scheme. The primary aim of our scheme is to decrease the compromised ratio of links against node-capture attacks without help of a server, that is, links compromised in WSNs automatically self-heal with time. Our scheme updates a link using the random data transmitted from the neighboring sensors, based on the idea of the POSH scheme. Although our protocol is very simple like POSH, more importantly, our security evaluation is not achieved easily, i.e., it is necessary to newly take the security of a link between sensors into consideration in our scheme since such security is not considered in the POSH scheme.

A link self-heals in two steps: first two neighboring sensors are self-healed, and then the link between these sensors is self-healed. A major difference between POSH and POLISH is the security analysis of a link. While the POSH scheme in a sense treats the secure link between a sensor and a powerful sink, the POLISH scheme treats the secure link between sensors. In addition, our scheme uses a bivariate t-degree polynomial, and thus an attacker has to capture $(t+1)$ polynomial shares during a limited period of time (i.e., at round 1) in order to corrupt a link.

ADV breaks into $k = |R^r|$ sensors to read the pairwise symmetric keys and secret seeds of PRNG in R^r, and to monitor all the communication of R^r. At any time, we identify three sets of sensors (refer to Section 4) and two sets of links, as follows:

- *Red links* (RL^r) are those that have been compromised in some round $r' < r$ and the pairwise symmetric key of the link is known to ADV in round r.
- *Green links* (GL^r) are those that have either never been compromised or regained their security in round r.

Note that, in our scheme, a red sensor s_i at round r means that ADV knows a seed S_i^r. If s_i becomes red in round r' and is self-healed at the end of round $r > r'$, then ADV can compute the contributions of s_i from round r' to r.

5.1 The protocol

Setup. To predistribute pairwise keys, the setup server randomly generates a bivariate t-degree polynomial $f(x, y)$ over a finite field $\mathbb{F}_q$, such that it has the

property of $f(x, y) = f(y, x)$. For each sensor s_i, the setup server computes a polynomial share of $f(x, y)$, that is, $f(x, ID_i)$. Each sensor can use a secure hash function, a polynomial and a PRNG with a unique secret seed. Note that the secure degree t of polynomial is dependent on the number of adversary at each round. For instance, if we set $t \geq 10$ as the secure degree of polynomial when we assume $k = 10$, then ADV cannot recover $f(x, y)$.

Key Establishment. For any two sensors s_i and s_j, the sensor s_i can compute the key $f(ID_j, ID_i)$ by evaluating $f(x, ID_i)$, and the sensor s_j can compute the same key $f(ID_i, ID_j) = f(ID_j, ID_i)$ by evaluating $f(x, ID_j)$. As a result, sensors s_i and s_j can establish a pairwise symmetric key $K_{i,j}^1 = f(ID_i, ID_j)$ in the first round (round 1). After key establishment, s_i deletes all the coefficients of a polynomial.

Key and Seed Update. The neighboring sensors s_i and s_j have a pairwise symmetric key $K_{i,j}^1$ (secure link) when they are deployed at the beginning of the first round (round 1). At the beginning of round r, s_i produces m pseudo-random values (contributions) using its PRNG for m neighboring sensors, and sends them to the neighboring sensors using a secure link. Note that all the contributions that s_i sends are different. Then, each sensor receives contributions from the neighboring sensors during round r. The recipient uses two contributions as inputs to the secure hash function used for key update. To update the secure link at the end of round r, s_i computes:

$$K_{i,j}^{r+1} = H(K_{i,j}^r || c_{i_\eta}^r || c_{j_\lambda}^r), \tag{2}$$

where $c_{i_\eta}^r$ is the η-th contribution that s_i received at round r and $c_{j_\lambda}^r$ is the λ-th contribution that s_j received at round r. Both s_i and s_j delete $K_{i,j}^r$ after key updating.

Furthermore, each sensor updates a seed of PRNG using m contributions, which are all contributions received by the neighboring sensors. To update the seed S_i^r at the end of round r, s_i computes[2]:

$$S_i^{r+1} = H(S_i^r || c_{i_1}^r || \cdots || c_{i_m}^r) \tag{3}$$

After seed updating, s_i deletes S_i^r. A seed is updated in every round, and then m contributions are generated by PRNG with such new seed.

Remark. In the POSH scheme, each sensor receives contributions from sensors which are randomly chosen in WSNs. On the other hand, in our scheme, each sensor receives contributions from neighboring sensors. The probability that a contribution will be intercepted on the way by ADV may become high in the POSH scheme, since a contribution can be sent from a sensor which is far from the recipient.

5.2 The Link State

A link self-heals in two steps: first two neighboring sensors are self-healed, and then the link between them is self-healed. A sensor state follows the transition

[2] The update of a PRNG seed is similar to [9].

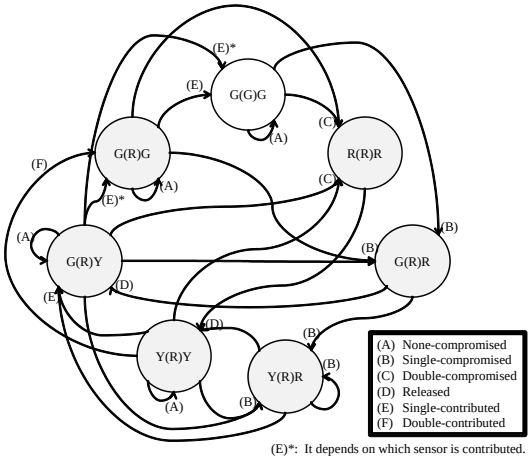

Fig. 2. Link state transition diagram.

diagram in Fig. 1. According to the state of a sensor we can generate the seven kinds of link states as described in Fig. 2 (i.e., $GL^r = \{\text{G(G)G}\}$ and $RL^r = \{\text{G(R)G, G(R)Y, Y(R)Y, Y(R)R, G(R)R, R(R)R}\}$). A link state is constituted by a pair of sensors and their common link. For example, G(R)Y means that two neighboring sensors of green and yellow are connected by the red link. The conditions of transition are as follows:

1. *Double-compromised* condition means that both of two neighboring sensors are compromised.
2. *Single-compromised* condition means that either of two neighboring sensors is compromised.
3. *None-compromised* condition means that neither of two neighboring sensors is compromised.
4. *Single-contributed* condition means that either of two neighboring sensors receives at least one "secure contribution".
5. *Double-contributed* condition means that both of two neighboring sensors receive at least one secure contribution.

Note that the secure contribution is a green contribution which is not intercepted by ADV.

A red link remains red if a red sensor is within the wireless communication range of both of two sensors which constitute the red link. On the other hand, a green link remains green as long as both of two sensors which constitute the green link are green. We notice that even if two sensors are green, the link between them can be also red (i.e., G(R)G). A green link (G(G)G) can be changed from two states G(R)G and G(R)Y when single-contributed. G(R)G becomes G(G)G when one of two neighboring sensors receives a secure contribution from the other. G(R)Y becomes G(G)G when the yellow sensor Y receives a secure

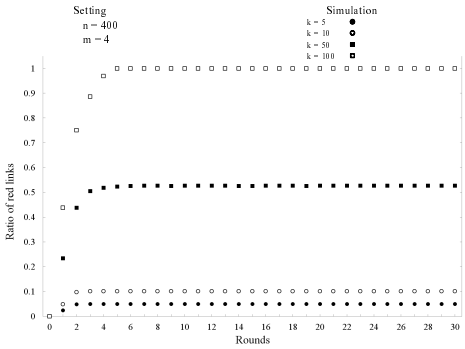

Fig. 3. Simulation results against continuous attackers.

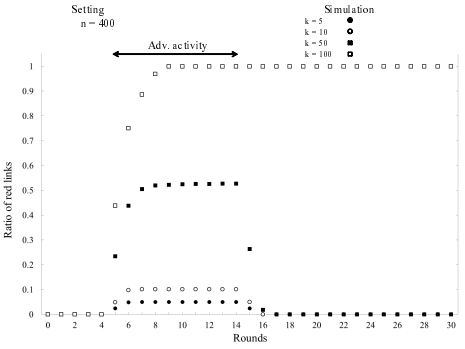

Fig. 4. Simulation results against temporary attackers.

contribution from this green sensor G. G(R)Y becomes G(G)G when Y receives at least one secure contribution from other green sensors except this G.

6 Analysis

6.1 Evaluation by Simulation

We evaluate the ratio of red links against *continuous* attackers to show the resiliency of our scheme, and we also evaluate the ratio of red links against *temporary* attackers to show the self-healing capabilities of our scheme. For ease of exposition and without loss of generality, we assume that the time slots (rounds) when sensors are compromised have the same duration and are synchronized.

Simulation Setup. The simulations are implemented in C on Windows XP SP3. All the simulations are repeated 1,000 times, and the results show the average values. To simplify the security analysis, we modeled the network as a grid of sensors of size $n = 400$ (20×20). We assume that the number of neighbors of each sensor is constant and equal to four (m=4). We can imagine a torus structure. Thus, the number of all links in WSN becomes 800. We also assume that the network topology does not change over time. The number k of ADV is 5, 10, 50 and 100 in every round.

Simulation Details. We evaluate the security of our scheme by the number of red links when ADV can compromise k sensors from the set G^r in any round. At the first round (round 1), n green sensors are deployed. We consider two different types of attackers: continuous attackers and temporary attackers. A continuous attacker keeps compromising sensors at constant rate from the deployment of the first round of sensors to the end of the network. In contrast, temporary attacker compromises sensors during a limited period of time, from round 5 to round 14 in our simulations. We then counted, in each round, the number of red links and computed the ratio. With the continuous attacker, we ran the simulation until: (1) the WSN has no more green sensors or (2) $|R^r|$ reaches a steady state.

Simulation Results. This section presents the results of our simulations for the different types of attackers. Fig. 3 displays the ratio of red links against continuous attackers. The ratio of red links reaches 100% when $k \geq 62$. For example, the ratio of red links is suppressed to 5.1% with $k = 5$, 10% with $k = 10$, 52% with $k = 50$ and 100% with $k = 100$, depicted in Fig. 3. The results for the temporary attacker are collected in Fig. 4. The action interval of the attacker (from generation 5 to generation 14) is denoted with the label "Adv. activity". We simulate a network with the same settings as the network used for the continuous attacker. Fig. 4 illustrates the self-healing property of our scheme as soon as ADV stops its activity, and the ratio of the red links starts decreasing at once. A link self-heals in only about three rounds. Note that once the ratio of red links becomes 1, the ratio remains 1 even when ADV stops its activity.

6.2 Analytical Model

Unlike the POSH scheme, a sensor in our scheme receives contributions from neighboring sensors, that is, a sensor receives m contributions. Note that the state transition of a sensor is the same as in the POSH scheme. In our scheme, it is necessary to consider the contributions from two-hop neighboring sensors. The contributions from neighboring sensors may be eavesdropped on by two-hop neighboring sensors. In this case, a green sensor is not self-healed even if it gets a contribution from a green sensor. Let $(1-(1-p_{R^r})^{m-1})$ be the probability that at least one sensor of two-hop neighboring sensors is red, that is, the probability that a green sensor's contribution is eavesdropped on by ADV (i.e., red sensor) which is within the wireless communication range of the green sensor. To become a green sensor (from yellow), the yellow sensor needs to be linked with at least one green sensor among neighboring sensors, and also a red sensor must not be within the wireless communication range of that green sensor. Thus, the probability of a yellow sensor not becoming green can be expressed as follows:

$$Pr' = \sum_{i=0}^{m} \binom{m}{i} p_{G^r}^i (1-p_{G^r})^{m-i} (1-(1-p_{R^r})^{m-1})^i, \tag{4}$$

where $p_{G^r} = \frac{|G^r|}{n-1}$, $p_{Y^r} = \frac{|Y^r|}{n-1}$ and $p_{R^r} = \frac{|R^r|}{n-1}$. The expected number of green sensors at round r is the same as in the POSH scheme, as follows[3]:

$$E[|G^{r+1}|] = |G^r| + (1-Pr')|Y^r| - |R^r| \tag{5}$$

To evaluate the link-healing rate of our scheme, we analyze the number of green links by evaluating the state of sensors in any round, i.e., the number of G(G)G in Fig. 5. The partial state transition diagram of a link is shown in Fig. 5, in which only the transition required to analyze the number of green links is depicted. That is, we consider only the input and the output of G(G)G and

[3] Since we assume that ADV corrupts only the green sensor (i.e., INF-ADV in [8]), we can use not inequality but an equation.

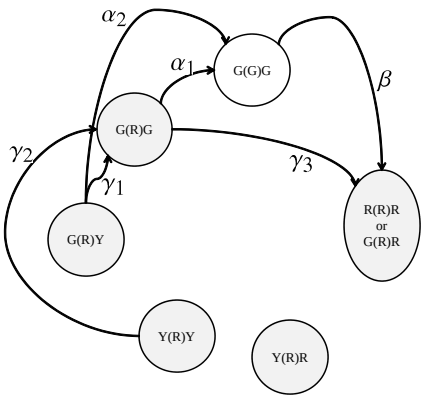

Fig. 5. Partial link state transition diagram.

G(R)G. Let α_1, α_2, β, γ_1, γ_2 and γ_3 be the number of link state transition (use not probability but a number.) and let $RL^r_{G(R)G} \subset RL^r$ be a set of the link state G(R)G. This figure shows that the expected number of green links in round r is:

$$E[|GL^{r+1}|] = |GL^r| + \alpha_1 + \alpha_2 - \beta, \tag{6}$$

where $\alpha_1 = (1 - (1 - (1 - p_{R^r})^{m-1})^2)|RL^r_{G(R)G}|$, $\alpha_2 = (1 - Pr')|Y^r|p_{\alpha_2}$ and $\beta = |R^r|p_\beta$. α_1 is the number of green links between two green sensors changed from $RL^r_{G(R)G}$. This transition occurs if neither of the green sensors is linked with a red sensor. Let p_{α_2} be the probability that a sensor needs to be linked with at least one green sensor of the neighboring sensors, and also that a red sensor must not be within the wireless communication range of that green sensor. α_2 is the number of green links between two green sensors, changed from red links between a green sensor and a yellow sensor. Let p_β be the probability that at least one green sensor in GL^r is corrupted. Hence, β is the number of red links between two red sensors, or between a yellow sensor and a red sensor changed from GL^r, since ADV corrupts only the green sensors and the number of ADV is $|R^r|$ in any round.

The number of red links between two green sensors is estimated in Fig. 5 as follows:

$$E[|RL^{r+1}_{G(R)G}|] = |RL^r_{G(R)G}| - \alpha_1 + \gamma_1 + \gamma_2 - \gamma_3, \tag{7}$$

where $\gamma_1 = (1 - Pr')|Y^r|p_{\gamma_1}$, $\gamma_2 = (1 - Pr')|Y^r|p_{\gamma_2}$ and $\gamma_3 = |R^r|p_{\gamma_3}$. Let p_{γ_1} be the probability that a sensor is linked with a green sensor, and also that a red sensor must not be within the wireless communication range of that green sensor. Let p_{γ_2} be the probability that a sensor is linked with a yellow sensor which becomes green. Moreover, let p_{γ_3} be the probability that at least one green sensor in $RL^r_{G(R)G}$ is corrupted.

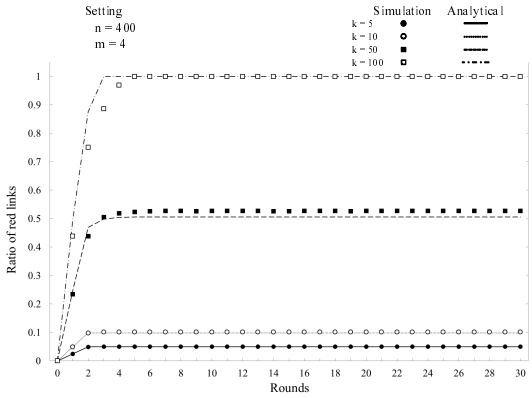

Fig. 6. Comparison of the analytical results and simulation results against continuous attackers.

Let μ be the ratio of $|GL^r|$ in a set of two neighboring green sensors which are linked each other, i.e., $\mu = \frac{|GL^r|}{|GL^r|+|RL^r_{G(R)G}|}$. We show the probability of p_{α_2}, p_β, p_{γ_1}, p_{γ_2} and p_{γ_3} as follows:

$$p_{\alpha_2} = \sum_{i=0}^{m} \binom{m}{i} (p_{G^r}(1-p_{R^r})^{m-1})^i (1-p_{G^r}(1-p_{R^r})^{m-1})^{m-i} i$$

$$p_\beta = \sum_{i=0}^{m} \binom{m}{i} (p_{G^r}\mu)^i (1-p_{G^r}\mu)^{m-i} i$$

$$p_{\gamma_1} = \sum_{i=0}^{m} \binom{m}{i} (p_{G^r}(1-(1-p_{R^r})^{m-1}))^i (1-p_{G^r}(1-(1-p_{R^r})^{m-1}))^{m-i} i$$

$$p_{\gamma_2} = \sum_{i=0}^{m} \binom{m}{i} ((1-Pr')p_{Y^r})^i (1-(1-Pr')p_{Y^r})^{m-i} i$$

$$p_{\gamma_3} = \sum_{i=0}^{m} \binom{m}{i} (p_{G^r}(1-\mu))^i (1-p_{G^r}(1-\mu))^{m-i} i$$

Fig. 6 shows a comparison of the analytical results and the simulation results. Note that the simulation results are the same as in Fig. 3. We found that our analytical results well matched the simulation results of our scheme.

6.3 Secure Connectivity

Our scheme has an advantage that the probability of establishing a secure link is 100%, since a sensor s_i has a polynomial $f(x, ID_i)$ and also shares the pairwise symmetric key $K^r_{i,j} = f(ID_j, ID_i)$ with s_j in the first round (round 1). After that the pairwise symmetric key of each link is updated, and hence the secure connectivity is 100% at every round.

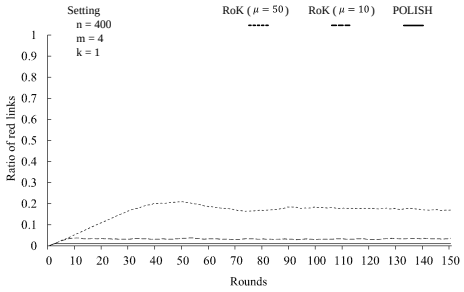

Fig. 7. Comparison of the simulation results against continuous attackers ($k = 1$).

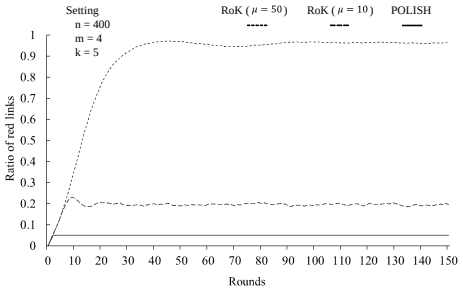

Fig. 8. Comparison of the simulation results against continuous attackers ($k = 5$).

6.4 Efficiency of POLISH

We discuss computational cost, communication cost and the size of memory in the POLISH scheme. Let R and H be the PRNG and hashing operations, respectively, and also let $|q|$ be the size of the contribution, ID, the output of hashing and the coefficient of a polynomial. The computational cost of each sensor in a round is $mR + (m + 1)H$. Note that it is required for s_i to assign the value of ID_j to a polynomial $f(x, ID_i)$ to share the pairwise symmetric key only in the first round (round 1). The communication cost of each sensor in a round is $2m|q|$ which includes the contributions of transmission and reception. Note that s_i needs to obtain IDs from m neighboring sensors in round 1. In order to setup data of a sensor at the first round, s_i requires a seed, the ID and the coefficient of a polynomial, i.e., the size of memory on a sensor requires $(t+3)|q|$ in total. After key establishment, s_i deletes all the coefficients of a polynomial, but m pairwise symmetric keys whose sizes are $m|q|$ are generated. Thus, the amount of memory on a sensor can save $(t + 1 - m)|q|$. This means that s_i can keep the contributions of transmission and reception if $(t + 1) \geq 3m$. Therefore, our scheme is efficient and is suitable for WSNs which constitute sensors with both limited memory and limited computational power.

7 Comparison

The POLISH scheme is the first proactive co-operative link self-healing scheme, without the help of a server. On the other hand, the previous link self-healing schemes need server's help, called resilient multi-phase WSNs. A multiphase WSN is a network where a sensor is redeployed with server's help when its battery is depleted. Although multiphase WSNs and our scheme are quite different in that the help of a server is necessary, we dare to compare our scheme with the previous scheme. We especially compare the RoK scheme with our scheme regarding security (the ratio of sick links) and efficiency since the RoK scheme is efficient and representative in multiphase WSNs.

7.1 Security

We compare the resiliency of our scheme with that of the RoK scheme by simulation experiments. For a fair comparison, the same size of memory is assumed between both schemes. When the length of key ring is 250 (i.e., $\omega = 250$), the total number of keys in the RoK scheme is just 500 which is the same parameters as [2]. When each size of key is 160 bits ($|q| = 160$ bits), at least 10kB memory is required in the RoK scheme. In our scheme, $(t + 1 - m)|q|$ bits memory is required as described in Section 6.4, more concretely, $20(t - 3)$ bytes memory is necessary when we follow the same simulation parameters as Section 6.1. We set $t = 497$ as the degree of polynomial from the standpoint of fairness. Actually, we can set $t = 5$ as the secure degree of polynomial when $k = 5$. In this case ADV cannot recover the polynomial.

We evaluate the ratio of red links against continuous attackers. A red link implies the compromised link in the RoK scheme. The network topology of both simulation is the same, in which a grid of sensors of size $n = 400$ and a sensor does not move over time. Fig. 7 and Fig. 8 are the comparisons of the simulation results between the RoK scheme and our scheme when the number of ADV is 1 and 5, respectively. In the RoK scheme, we simulate sensors expiration by assigning to each sensor a random expiration date, chosen according to a Gaussian distribution with mean $\mu = 50.0$ rounds and with standard deviation $\sigma = 16.7$ whose parameters are the same as [2], and also with $\mu = 10.0$ and with $\sigma = 3.33$. When the sensor expiration becomes short in the RoK scheme, the ratio of red links also decreases as described in Fig. 7 and Fig. 8. However, it is difficult to lower the ratio of red links substantially since this is contrary to the battery extension of sensor life which the WSNs aim at.

Remark. In [2] the compromised ratio (i.e., the ratio of red links) is evaluated as a probability that a link is "indirectly" compromised by ADV. On the other hand, we evaluate all the number of red links in both schemes in this simulation. Since we re-evaluate the RoK scheme by the total red links, the ratio of red links of the RoK scheme in Fig. 7 and Fig. 8 becomes a little higher than the original results.

7.2 Efficiency of Key Update

Both the RoK scheme and our scheme update keys and key materials at every round. We think that it is important to especially reduce the size of memory since the amount of computations and communications is not so frequent (i.e., they are executed at the beginning of each round). While the key materials are updated only by each sensor in the RoK scheme, keys and key materials are updated by cooperation of the neighboring sensors in our scheme. Hence more communications are required in our scheme but more memory is required in the RoK scheme.

Since most links are compromised in the RoK scheme when $k = 5$ as described in Fig. 8, we consider $k \leq 5$ as the number of ADVs. We thus set $t = 5$ in order that at most five ADVs cannot recover the polynomial of our scheme.

Furthermore, we set $m = 4$, $|q| = 160$ bits and $\omega = 250$. The computational cost, the communication cost and the size of memory in our scheme are $mR+(m+1)H$, $(t + 1 - m)|q|$ and $2m|q|$, respectively, by Section 6.4. On the other hand, the computational cost and the size of memory in the RoK scheme are $2H$ and $2w|q|$, respectively, and then the communication cost is not necessary for the update of key materials. As a result, although a little computation and a little communication are required in our scheme, the size of memory is much lower than the RoK scheme.

8 Conclusion

We have proposed the POLISH scheme, in which together with key evolution, our scheme provides both forward and backward security of a link in the presence of an adversary. Both analytical and simulation results show that our scheme is very effective. Our simulation shows that POLISH based network that is continuously attacked is resilient when $k < 62$ ($n = 400$ and $m = 4$). Our simulation also shows that a network that is temporarily attacked automatically self-heals in only about three rounds. Furthermore, we found that our analytical results well matched the simulation results of our scheme.

References

1. C. Blundo, A.D. Santis, A. Herzberg, S. Kutten, U. Vaccaro, and M. Yung. Perfectly-secure key distribution for dynamic conferences. In *CRYPTO'92*, LNCS, 740, pages 471–486, 1993.
2. C. Castelluccia and A. Spognardi. Rok: A robust key pre-distribution protocol for multi-phase wireless sensor networks. In *SecureComm2007*, pages 351–360, September 2007.
3. W. Du, J. Deng, Y.S. Han, S. Chen, and P.K. Varshney. A key management scheme for wireless sensor networks using deployment knowledge. In *INFOCOM'04*, pages 586–597, March 2004.
4. L. Eschenauer and V.D. Gligor. A key-management scheme for distributed sensor networks. In *CCS'02*, pages 41–47, November 2002.
5. H. Ito, A. Miyaji, and K. Omote. RPoK: A strongly resilient polynomial-based random key pre-distribution scheme for multiphase wireless sensor networks. In *Globecom*, pages 1–5, December 2010.
6. K. Kalkan, S. Yilmaz, O.Z. Yilmaz, and A. Levi. A highly resilient and zone-based key predistribution protocol for multiphase wireless sensor networks. In *Q2SWinet'09*, pages 29–36, October 2009.
7. D. Ma, and G. Tsudik. DISH: Distributed self-healing. In *SSS'08*, pages 47–62, 2008.
8. R.D. Pietro, D. Ma, C. Soriente, and G. Tsudik. POSH: Proactive co-operative self-healing in unattended wireless sensor networks. In *SRDS'08*, pages 185–194, 2008.
9. R.D. Pietro, G. Oligeri, C. Soriente, and G. Tsudik. Intrusion-Resilience in Mobile Unattended WSNs. In *INFOCOM*, pages 2303–2311, 2010.
10. O.Z. Yilmaz, A. Levi, and E. Savas. Multiphase deployment models for fast self healing in wireless sensor networks. In *SECRYPT*, pages 136–144, July 2008.