中央研究院
資訊科學研究所
Institute of Information Science, Academia Sinica • Taipei, Taiwan, ROC

TR-IIS-08-013

Practical Pairwise Key Establishment Schemes for Wireless Sensor Networks via Constrained Random Perturbation

Chia-Mu Yu, Chun-Shien Lu, Sy-Yen Kuo

December 17, 2008 || Technical Report No. TR-IIS-08-013

http://www.iis.sinica.edu.tw/page/library/LIB/TechReport/tr2008/tr08.html

Practical Pairwise Key Establishment Schemes for Wireless Sensor Networks via Constrained Random Perturbation

Chia-Mu Yu[1,2], Chun-Shien Lu[1], Sy-Yen Kuo[2]
[1] Institute of Information Science, Academia Sinica, Taiwan, ROC
[2] Graduate Institute of Electrical Eng., National Taiwan University, Taiwan, ROC

ABSTRACT

The resource limitation of sensor nodes poses a great challenge in designing an efficient key establishment scheme for Wireless Sensor Networks (WSNs). In spite of the fact that many elegant and clever solutions have been proposed, no practical key establishment scheme has emerged.

In this paper, a *ConstrAined Random Perturbation based pairwise keY establishment* (CARPY) scheme and its variant, a CARPY+ scheme, for WSNs, are presented. Compared to all existing schemes which satisfy only some requirements in so-called *sensor-key criteria*, including 1) resilience to various attacks, 2) directed and guaranteed key establishment, 3) resilience to network configurations, 4) efficiency, and 5) resilience to dynamic node deployment, the proposed CARPY+ scheme meets all requirements. In particular, to the best of our knowledge, CARPY+ is the first non-interactive key establishment scheme with great resilience to a large number of node compromises designed for WSNs. We examine the CARPY and CARPY+ schemes from both the theoretical and experimental aspects. They have also been practically implemented on the TelosB compatible mote to evaluate the corresponding performance and overhead.

1. INTRODUCTION AND RELATED WORK

A Wireless Sensor Network (WSN) is composed of a large number of sensor nodes with limited resources. Since WSNs could be deployed in a hostile environment, designing an efficient key establishment scheme is of great importance to the data security in WSNs. Unfortunately, when considering the extremely scarce resources available to each sensor node, the design of an efficient key establishment becomes a great challenge.

Two classical threshold-based key distribution (TKD) protocols [1, 2] are considered. As the security of both protocols is completely broken as long as the number of captured nodes is above a pre-determined threshold, which is linearly dependent on the network size and the storage overhead, they are considered not to be suitable for WSNs. To provide resilient security, a useful technique called *probabilistic key pre-distribution* (P-KPD), proposed by Eschenauer and Gligor [10], has been extensively studied. In a P-KPD scheme, a *key pool* consisting of a large number of randomly generated keys is first prepared. Then, several keys randomly selected from the key pool are stored in each sensor node to constitute a *key ring*. After sensor deployment, when required, a *shared-key discovery* procedure is performed to find a common key between two nodes, called *shared-key*, in their respective key rings. Two nodes, i and j, that fail to have a shared-key in the shared-key recovery step perform a procedure, called *path-key establishment*, in which the *path-key* generated by i is relayed along the *key path* to j and acts as the common key between i and j. Here, the key path is a path on which each pair of consecutive nodes has a shared-key. Motivated by the P-KPD, Chan *et al*. [4] proposed that, instead of relying on only one common key, q common keys between two sensor nodes are necessary to construct the shared-key used for further communications.

Due to the problem that different pairs of nodes could share the same key, when the number of compromised nodes increases, the fraction of affected keys increases quickly as well. Aiming at providing pairwise keys between each pair of nodes, Chan *et al*. [4] proposed the random-pairwise keys scheme, which stores predefined pairwise keys, instead of random keys, into certain pairs of nodes. Liu and Ning [14], and Du *et al*. [8] also proposed to treat the keys in the key pool S as bivariate polynomials and matrices, respectively, to achieve the same goal.

There are some common drawbacks in the P-KPD schemes. For example, P-KPD schemes cannot guarantee that any two sensor nodes can have common keys. Moreover, the Merkle puzzle [21] must be used to guarantee the minimal secret information leakage. In view of this, several *deterministic key pre-distribution* (D-KPD) schemes such as PIKE [3], expander graph-based scheme [6], and hybrid design scheme [5] are proposed. D-KPD schemes can guarantee that there exists at least one key path between two arbitrary nodes. Another common drawback of the P-KPD schemes is that, two nodes always rely on communications between them to find their common key. Focusing on reducing such communication overhead, a strategy, called Pseudo-Random Key pre-deployment (PRK) [19], has been proposed, in which two nodes can find their shared-key with certain probability without any communication.

A common problem existing in both P-KPD and D-KPD schemes is that not every pair of nodes can directly establish their common key. Zhang *et al*. recently proposed a Random Perturbation-Based (RPB) scheme [30] to avoid this, while maintaining resilient security.

Usually, one assumes that, prior to sensor deployment, nodes' locations are not known by the network planner. When some special deployment models are considered, prior knowledge about nodes' locations can be utilized to construct efficient location-aware key pre-distribution (L-KPD) schemes [7, 12]. In addition, based on the assumption that there is a short bootstrapping time secure after a sensor network is deployed, Localized Encryption and Authentication Protocol (LEAP) [28] is proposed to establish the pairwise keys between each pair of neighboring sensor nodes.

With the fact that the communication channels in WSNs are highly noisy [27] and that over 95% of energy consumption comes from communication [20] in mind, we can know that although numerous key establishment schemes are proposed, all of them, except the TKD schemes [1, 2] (which unfortunately cannot achieve resilient security), are inefficient and highly energy-consuming. Thus, it is desirable, but extremely challenging, to have a key establishment scheme satisfying both security and energy efficiency.

1.1 Evaluation Metrics

To evaluate the key establishment schemes, five requirements were recently presented in [30]. Nevertheless, we find that they are too weak to be utilized, as the security and performance of cer-

tain key establishment schemes have been overestimated. Hence, motivated by the five requirements in [30], a set of five new requirements is proposed as follows to thoroughly evaluate the key establishment schemes applied in the real world.

- *Resilience to Various Attacks* (RVA) – In the literature, only the node capture attack is considered as an avenue for the adversary to compromise the security of key establishment schemes. However, in the real world, a smart adversary can simultaneously mount several kinds of attacks. For example, for P-KPD and D-KPD, after several nodes are captured by the adversary, all the keys obtained from the captured nodes can be stored in multiple fabricated nodes with the IDs given from multiple captured nodes. These fabricated nodes are then placed back into the network to eavesdrop on the path-keys more efficiently than the solely node capture attack does. Such an attack can be thought of as a mixture of the node capture, Sybil [18], and sink hole [13] attacks. On the other hand, instead of compromising the security of the key establishment scheme, the adversary may only want to hinder the nodes from establishing keys by simply using, for example, selective forwarding attack [13] . As a result, it is necessary to consider the resilience and survivability under various attacks which could be mounted by the adversary, rather than only the node capture attack.

- *Directed and Guaranteed Key Establishment* (DGKE) – Each pair of sensor nodes should be able to establish a common key by their own effort wherever they reside and whenever they need, without exposing secrets to or obtaining secrets from the third parties.

- *Resilience to Network Configurations* (RNC) – Since the use of sensor networks is highly application-dependent, the heterogeneity, mobility, deployment pattern, density of sensor nodes, and the network size should not affect the effectiveness and efficiency of the key establishment schemes. In other words, key establishment schemes are necessary to be applicable whatever network configuration is applied.

- *Efficiency* (EFF) – Key establishment schemes are required to be performed efficiently in terms of storage, computation, and communication overhead. Note that the time consumed for finding the common key could be a metric for evaluating the efficiency of a key establishment scheme as well. However, in general, such latency primarily comes from the computation and communication delay; therefore, assessing the computation and communication overhead is equivalent to the assessing the latency.

- *Resilience to Dynamic Node Deployment* (RDND) – The hardware failure or energy depletion of sensor nodes could result in a WSN which cannot achieve full coverage of the sensing region, or even becomes disconnected. In light of this, new sensor nodes are necessary to be deployed in the network. A desirable key establishment scheme should be applicable under the consideration of on-the-fly addition of new sensor nodes.

For convenience, these five requirements are called *sensor-key criteria*, with which a desirable key establishment scheme for WSN should be satisfied.

1.2 Contribution

There are two major contributions of the paper:

1. Based on the proposed constrained random perturbation technique, two constrained random perturbation based pairwise key establishment schemes, CARPY and CARPY+, are introduced. While all the existing schemes only meet a part of the sensor-key criteria, CARPY+ is the only scheme satisfying all the requirements in the sensor-key criteria. In particular, CARPY+ is the first non-interactive key establishment scheme with great resilience to a large number of node compromises designed for WSNs.

2. Detailed theoretical studies with respect to the performance and security of the proposed CARPY and CARPY+ schemes are provided. In addition, the proposed CARPY and CARPY+ schemes have also been practically implemented on the TelosB compatible mote to evaluate the performance and overhead.

1.3 Organization

The proposed CARPY and CARPY+ schemes are presented in Sec. 2. Together with a comprehensive comparison with the other schemes, the theoretical and experimental results will be shown in Sec. 3. At last, the conclusion will be presented in Sec. 4.

2. THE PROPOSED METHOD

The proposed schemes, CARPY and CARPY+, are based on Blom's concept [1]. Therefore, after describing the system model in Sec. 2.1, we briefly review Blom's scheme in Sec. 2.2. Afterwards, CARPY and CARPY+ are described in detail in Sec. 2.3 and Sec. 2.4, respectively. Finally, the methods for constructing constrained random perturbation will be presented in Sec. 2.5.

2.1 System Model

Network Model. We assume that N low-cost resource-constrained sensor nodes are deployed over the sensing region and no prior deployment knowledge about the nodes' locations is known by the network planner in advance. There is a data collection unit, called *data sink*, placed in the network. We do not assume the trustworthiness and authenticity of data sink. Each sensor node is assumed to have a unique ID, which could be arbitrarily chosen in a general-purpose sensor node or fixed in a specific sensing hardware. In addition to static networks, mobile nodes are also allowed in our methods so that partial or entire nodes could have mobility. Moreover, we also do not assume the network topology. In other words, the density, deployment pattern, and other characteristics of sensor nodes could be arbitrary.

Security Model. Sensor nodes are assumed to have no tamper-resistant hardware so that once the sensor node is captured by the adversary, the secret information stored in the captured node will be exposed to the adversary. The adversary can mount attacks immediately after sensor deployment, *i.e.*, the secure bootstrapping time [28] does not exist in our model. The objective of the adversary is to either compromise the secure communications between sensor nodes which have not yet been compromised by the adversary or just to hinder the nodes from establishing keys. To achieve his/her goal, the adversary can simultaneously launch several attacks. In this paper, we assume that four categories of attacks, which are eavesdropping, node capture, routing layer, and physical layer attacks, can be mounted by the adversary. They are described in detail in Sec. 3.4.

2.2 A Review of Blom's Scheme [1]

Suppose the number of sensor nodes is N. Let $\mathbb{F}_q = \{0, \ldots, q-1\}$, $q > N$, be a finite field. For a matrix G, we denote the element in the i-th row and j-th column of G by $G_{i,j}$, i-row of G by $G_{i,-}$

and the j-th column of G by $G_{-,j}$. Assume that a symmetric matrix $D \in \mathbb{F}_q^{(\lambda+1)\times(\lambda+1)}$ and a matrix $G \in \mathbb{F}_q^{(\lambda+1)\times N}$ are randomly generated. Note that the only requirement for G is that any $\lambda+1$ columns of G should be linearly independent in order to achieve guaranteed security. Let $A = (D \cdot G)^T$ and $K = A \cdot G$. It can be easily checked that K is also a symmetric matrix as follows:

$$A \cdot G = (D \cdot G)^T \cdot G = G^T \cdot D \cdot G = (A \cdot G)^T. \quad (1)$$

Note that the above operations are all performed in the finite field $\mathbb{F}_q$. Blom's idea [1] is that for each node i, the row vector $A_{i,-}$ and the column vector $G_{-,i}$ are stored into the node i. Thus, when two nodes i and j would like to have a common key, they exchange their columns of G in plaintext and then use their private rows of A to calculate $K_{i,j}$ $(= A_{i,-} \cdot G_{-,j})$ and $K_{j,i}$ $(= A_{j,-} \cdot G_{-,i})$, respectively. Fig. 1 illustrates Blom's idea. Blom's scheme achieves so-called λ-secure [1], which ensures that as long as no more than λ nodes are compromised, the security can be perfectly preserved. Intuitively, the security of Blom's scheme comes from the privacy of the matrix D, while the matrix G acts as a public information even for the adversary. When D is totally known by the adversary, Blom's scheme becomes insecure. In spite of such guaranteed security, Blom's scheme cannot be directly applied to WSNs because the storage overhead grows rapidly when the security level must be preserved in a network of large size.

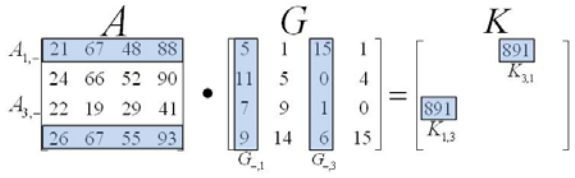

Figure 1: An illustration of the Blom's scheme.

2.3 The CARPY Scheme

We assume that the network consists of N sensor nodes with IDs, $\mathcal{I} = \{s_1, s_2, \ldots, s_N\}$ and $s_1 < s_2 < \ldots, < s_N$. We also assume that $q > N$, where q is a parameter of a finite field $\mathbb{F}_q$, and λ is an appropriate security parameter independent of N, which leverages the security level and storage overhead.

2.3.1 Basic Idea of CARPY scheme

In Blom's scheme, communications become insecure after more than λ sensor nodes are compromised. The reason for this is that the row vector $A_{i,-}$ in the sensor node i is directly related to the private matrix D. Hence, after collecting a sufficient number of row vectors of A, the adversary is able to construct the private matrix D by solving a system of linear equations since G is publicly known. An idea, similar to the one used in [30], to enhance the security is to break the direct relation between D and A by adding certain random noise* on A to distort Blom's key. However, if improper random noise is applied, either additional computation and communication are needed to extract the common bits of distorted Blom's key between two sensor nodes, or the common key cannot be found anymore. To conquer these drawbacks, we propose to apply *constrained random perturbation* (CRP). The constrained random perturbed Blom's key, when compared to the original Blom's key, will satisfy high signal-to-noise (SNR) ratio, *i.e.*, if the length of Blom's key is ℓ, then only the least r $(r < \ell)$ bits of Blom's key are perturbed after the CRP is added. Thus, the first $\ell - r$ bits of Blom's key are retained, resulting in the guaranteed establishment of the common key without the need of additional overhead. In contrast to the random perturbation [30] that incurs unnecessary computation and communication overhead, the way of constructing CRP and the corresponding efficiency gain substantially differentiate our work and [30]. The main idea of CARPY is shown in Fig. 2. Obviously, the execution of each round of the CARPY scheme can generate $\ell - r$ bits of a pairwise key. When the bit-length of desired key is $L > (\ell - r)$, the CARPY scheme should be executed $\lceil \frac{L}{\ell - r} \rceil$ rounds to generate a pairwise key with desired key length. Although $\lceil \frac{L}{\ell - r} \rceil$ rounds of CARPY are required, the overall computation overhead, which will be analyzed in Sec. 3.2, is still affordable for the current generation sensor nodes.

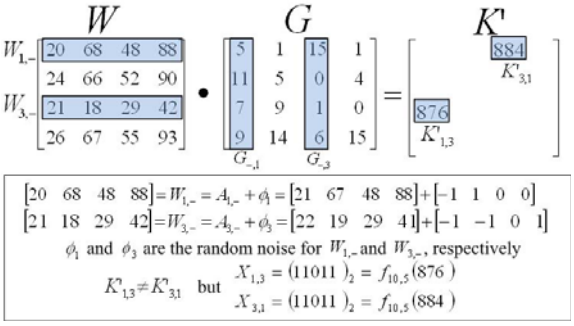

Figure 2: An illustration of the CARPY scheme

There are two steps in the CARPY scheme, the off-line step and the on-line step. In general, off-line step is performed to determine the desired key length, select appropriate parameters, and pre-install the keying materials into the sensor nodes, before deployment of sensor nodes. The on-line step is performed for each pair of sensor nodes required to find the pairwise key in common after sensor nodes are deployed.

2.3.2 Off-line Step of CARPY scheme

In addition to the parameters such as the size q of the finite field $\mathbb{F}_q$, the security parameter λ of Blom's scheme, and the set $\mathcal{I}$ of IDs of sensor nodes mentioned in the previous sections, some other parameters such as the number r of least bits perturbed by CRP for the Blom's key, and the bit-length L of desired key should be determined by the network planner before off-line step is executed. Let ℓ be the least number of bits necessary to represent the elements in $\mathbb{F}_q$. Since the execution of each round of the CARPY scheme can generate $\ell - r$ bits of a pairwise key, the CARPY scheme should be executed ξ $(= \lceil \frac{L}{\ell - r} \rceil)$ rounds to obtain a pairwise key with desired key length L.

The algorithm for the off-line step is shown in Fig. 3. Here, we explain the off-line step of the CARPY scheme from executing the t-th round of the CARPY scheme. Note that all the arithmetic operations in the subsequent descriptions are accomplished in the finite field $\mathbb{F}_q$ unless specifically noted. At first, as in Blom's scheme, the network planner randomly generates two matrices $D^{(t)} \in \mathbb{F}_q^{(\lambda+1)\times(\lambda+1)}$ and $G^{(t)} \in \mathbb{F}_q^{(\lambda+1)\times s_N}$ such that $D^{(t)}$ is symmetric and any $\lambda+1$ columns of $G^{(t)}$ are linearly independent. After that, we calculate the matrix $A^{(t)} = (D^{(t)} \cdot G^{(t)})^T$.

Let $c_{\min}(\varrho, r)$ be the value of ϱ which has least r bits of its binary representation set to 0. Similarly, let $c_{\max}(\varrho, r)$ be the value

*The terms, *random noise* and *random perturbation*, will be used interchangeably throughout this paper.

Algorithm: CARPY-Off-line-Step(q, r, $\mathcal{I}$, λ, L)
Input: q: *the elements of D and G, randomly selected from $\mathbb{F}_q$*
r: *the least r bits which will be infected by CRP*
$\mathcal{I}$: *the set of sensor nodes identities*
λ: *a security parameter*
L: *the bit-length of desired key*

Calculate ℓ
for $t = 1$ to ξ (=$\lceil \frac{L}{\ell - r} \rceil$)
  Randomly generate $D^{(t)}$ and $G^{(t)}$, and Calculate $A^{(t)}$
  for $u = 1$ to $|\mathcal{I}|$ $(= N)$
    Calculate $\Phi_{s_u}^{(t)}$
    Randomly select a row vector $\phi_{s_u}^{(t)}$ from $\Phi_{s_u}^{(t)}$
    Calculate $W_{s_u,-}^{(t)} = A_{s_u,-}^{(t)} + \phi_{s_u}^{(t)}$
    Store $W_{s_u,-}^{(t)}$ and $G_{-,s_u}^{(t)}$ into the sensor node s_u

Figure 3: Off-line step of the CARPY scheme.

of ϱ which has least r bits of its binary representation set to 1. For example, $c_{\min}(524, 5) = 512$ and $c_{\max}(524, 5) = 543$. Let $\Phi_{s_u}^{(t)}$ denote the set of CRPs applied on the row vector $A_{s_u}^{(t)}$ for $s_u \in \mathcal{I}$. When the t-th round of the CARPY scheme is performed, each CRP $\phi_{s_u}^{(t)} \in \Phi_{s_u}^{(t)}$ must satisfy the following constraints:

$$(A_{s_u,-}^{(t)} + \phi_{s_u}^{(t)}) \cdot G_{-,s_v}^{(t)} \geq c_{\min}(A_{s_u,-}^{(t)} \cdot G_{-,s_v}^{(t)}, r) \pmod q \quad (2)$$

$$(A_{s_u,-}^{(t)} + \phi_{s_u}^{(t)}) \cdot G_{-,s_v}^{(t)} \leq c_{\max}(A_{s_u,-}^{(t)} \cdot G_{-,s_v}^{(t)}, r) \pmod q \quad (3)$$

$$\phi_{s_u}^{(t)}(k) \in \mathbb{Z}, \quad (4)$$

where $u \neq v$, $1 \leq u, v \leq N$, $1 \leq k \leq (\lambda + 1)$, and $\phi_{s_u}^{(t)}(k)$ is the k-th element of $\phi_{s_u}^{(t)}$. Note that a CRP $\phi_{s_u}^{(t)}$ is a $(\lambda+1)$-dimensional row vector. Eqs. (2) and (3) mean that after CRP is added to $A_{s_u,-}^{(t)}$ of the sensor node s_u, the most significant $\ell - r$ bits of the corresponding Blom's key are retained for every other sensor node s_v. These two constraints guarantee the existence of the common part of constrained random perturbed Blom's keys without needing computation or communication overhead resulting from additional checks. The constraint indicated in Eq. (4) should be satisfied because in CARPY the elements of CRPs are constrained to be integers. As a whole, every $\phi_{s_u}^{(t)}$ that satisfies Eqs. (2)∼(4) is one of the elements in $\Phi_{s_u}^{(t)}$.

Following the construction of $\Phi_{s_u}^{(t)}$, for every $s_u \in \mathcal{I}$, a CRP $\phi_{s_u}^{(t)}$ is randomly and independently selected from $\Phi_{s_u}^{(t)}$. Then, the matrix $W^{(t)}$ is constructed by calculating $W_{s_u,-}^{(t)} = A_{s_u,-}^{(t)} + \phi_{s_u}^{(t)}$ for $1 \leq t \leq \xi$, $1 \leq u \leq N$. After the matrix $W^{(t)}$ is constructed, the row vectors $W_{s_u,-}^{(t)}$ together with the column vectors $G_{-,s_u}^{(t)}$ are stored into each node s_u.

2.3.3 On-line Step of CARPY scheme

Assume that two sensor nodes, $s_{\hat{u}}$ and $s_{\hat{v}} \in \mathcal{I}$, want to share a pairwise key. When the t-th round CARPY scheme is executed, they first exchange their columns of $G^{(t)}$, $G_{-,s_{\hat{u}}}^{(t)}$ and $G_{-,s_{\hat{v}}}^{(t)}$. Then, $s_{\hat{u}}$ and $s_{\hat{v}}$ calculate $\kappa_{s_{\hat{u}},s_{\hat{v}}}^{(t)} = W_{s_{\hat{u}},-}^{(t)} \cdot G_{-,s_{\hat{v}}}^{(t)}$ and $\kappa_{s_{\hat{v}},s_{\hat{u}}}^{(t)} = W_{s_{\hat{v}},-}^{(t)} \cdot G_{-,s_{\hat{u}}}^{(t)}$, respectively. We can see from Eqs. (2)∼(4) that the distortion of the constructed constrained random perturbed Blom's keys between two nodes is guaranteed to be limited within their least r bits, and, thus, the t-th part of pairwise key between $s_{\hat{u}}$ and $s_{\hat{v}}$ is $X_{s_{\hat{u}},s_{\hat{v}}}^{(t)} = f_{\ell,r}(\kappa_{s_{\hat{u}},s_{\hat{v}}}^{(t)}) = f_{\ell,r}(\kappa_{s_{\hat{v}},s_{\hat{u}}}^{(t)})$, where $f_{\ell,r}(x)$ is the most significant $\ell - r$ bits of ℓ-bit binary representation of a number x. Eventually, the pairwise key $X_{s_{\hat{u}},s_{\hat{v}}}$ between nodes $s_{\hat{u}}$ and $s_{\hat{v}}$ is $X_{s_{\hat{u}},s_{\hat{v}}}^{(1)} || X_{s_{\hat{u}},s_{\hat{v}}}^{(2)} || \cdots || X_{s_{\hat{u}},s_{\hat{v}}}^{(\xi)}$. The algorithm for the on-line step of the CARPY scheme is depicted in Fig. 4. An example illustrating the execution of CARPY is shown in Example 1.

Algorithm: CARPY-On-line-Step
Scenario: sensor nodes $s_{\hat{u}}$ and $s_{\hat{v}}$ wants to agree a pairwise key
Note: this algorithm is executed by the sensor node $s_{\hat{v}}$

for $t = 1$ to ξ $(= \lceil \frac{L}{\ell - r} \rceil)$
  Send $G_{-,s_{\hat{u}}}^{(t)}$ to the sensor node $s_{\hat{v}}$
  Receive $G_{-,s_{\hat{v}}}^{(t)}$ from the sensor node $s_{\hat{v}}$
  Calculate $\kappa_{s_{\hat{u}},s_{\hat{v}}}^{(t)} = W_{s_{\hat{u}},-}^{(t)} \cdot G_{-,s_{\hat{v}}}^{(t)}$
  Calculate $X_{s_{\hat{u}},s_{\hat{v}}}^{(t)} = f_{\ell,r}(\kappa_{s_{\hat{u}},s_{\hat{v}}}^{(t)})$
Calculate $X_{s_{\hat{u}},s_{\hat{v}}} = X_{s_{\hat{u}},s_{\hat{v}}}^{(1)} || X_{s_{\hat{u}},s_{\hat{v}}}^{(2)} || \cdots || X_{s_{\hat{u}},s_{\hat{v}}}^{(\xi)}$

Figure 4: On-line step of the CARPY scheme.

Example 1. *Given that $q = 2^{10} - 3$, $N = 4$, $\lambda = 3$, $\mathcal{I} = \{1, 2, 3, 4\}$, $r = 5$, and $L = 5$. The main idea of CARPY is shown in Fig. 2. In this example, $\ell = 10$ can be calculated. Since $\ell - r = L$, performing the CARPY scheme once is sufficient to generate a key with length L. Moreover, $W_{1,-}$ comes from $A_{1,-} + \phi_{1,-}$, where $A_{1,-}$ is shown in Fig. 1 and $\phi_{1,-}$ is randomly chosen as a row vector $\begin{bmatrix} -1 & 1 & 0 & 0 \end{bmatrix}$, satisfying Eqs. (2)∼(4). $W_{3,-}$ can be obtained similarly by having $A_{3,-} + \phi_{3,-}$, where $\phi_{3,-} = \begin{bmatrix} -1 & -1 & 0 & 1 \end{bmatrix}$, as also shown in Fig. 2. Though $K'_{1,3} = W_{1,-} \cdot G_{-,3} \neq W_{3,-} \cdot G_{-,1} = K'_{3,1}$, their most significant $\ell - r = 5$ bits are the same, i.e., $X_{1,3} = f_{10,5}(228) = (000111)_2 = f_{10,5}(218) = X_{3,1}$. Hence, $X_{1,3}$ $(= X_{3,1})$ can be used as the pairwise key between sensor nodes with IDs 1 and 3.*

2.4 Towards Communication-Free CARPY (CARPY+) scheme

In the CARPY scheme, two sensor nodes communicate with each other only for exchanging the respective column of G, which can be known by the adversary. If each column of G can be generated by each sensor node, then communications will no longer be necessary. Recall that the only requirement for G is that any $\lambda + 1$ columns of G should be linearly independent. Thus, the Vandermonde matrix is most suitable for our use because, if φ is the primitive element of $\mathbb{F}_q$, then any $\lambda+1$ columns of Vandermonde matrix, which is generated by only one element φ, are linearly independent [17]. Note that such Vandermonde matrix is of the form that the i-th column is generated by $\begin{bmatrix} 1 & \varphi^i & (\varphi^i)^2 & \cdots & (\varphi^i)^\lambda \end{bmatrix}^T$, where λ is a security parameter independent of N. Therefore, communication overhead can be eliminated if the matrix G of CARPY is selected as a Vandermonde matrix. For convenience, the CARPY scheme with G being a Vandermonde matrix is called CARPY+. The off-line and on-line steps of the CARPY+ scheme are depicted in Fig. 5 and Fig. 6, respectively.

2.5 Constructing Constrained Random Perturbation

In this section, we deal with the problem of calculating the set $\Phi_{s_u}^{(t)}$ of CRPs for $1 \leq t \leq \xi$ and $1 \leq u \leq N$. A straightforward method for obtaining $\Phi_{s_u}^{(t)}$ is to adopt an exhaustive search.

Algorithm: CARPY+-Off-line-Step(q, r, $\mathcal{I}$, λ, L)
Input: q: *the elements of D and G, randomly selected from $\mathbb{F}_q$*
r: *the least r bits which will be infected by CRP*
$\mathcal{I}$: *the set of sensor node identities*
λ: *a security parameter*
L: *the bit-length of desired key*

Calculate ℓ
for $t = 1$ to ξ (=$\lceil \frac{L}{\ell - r} \rceil$)
  Select φ and $G^{(t)}$
  Randomly generate $D^{(t)}$, and calculate $A^{(t)}$
  for $u = 1$ to $|\mathcal{I}|$ $(= N)$
    Calculate $\Phi_{s_u}^{(t)}$
    Randomly select a row vector $\phi_{s_u}^{(t)}$ from $\Phi_{s_u}^{(t)}$
    Calculate $W_{s_u,-}^{(t)} = A_{s_u,-}^{(t)} + \phi_{s_u}^{(t)}$
    Store $W_{s_u,-}^{(t)}$ and φ into the sensor node s_u

Figure 5: Off-line step of the CARPY+ scheme.

Algorithm: CARPY+-On-line-Step
Scenario: nodes $s_{\hat{u}}$ and $s_{\hat{v}}$ want to agree on a pairwise key
Note: this algorithm is executed by the sensor node $s_{\hat{u}}$

for $t = 1$ to ξ $(= \lceil \frac{L}{\ell - r} \rceil)$
  Calculate $G_{-,s_{\hat{v}}}^{(t)}$
  Calculate $\kappa_{s_{\hat{u}},s_{\hat{v}}}^{(t)} = W_{s_{\hat{u}},-}^{(t)} \cdot G_{-,s_{\hat{v}}}^{(t)}$
  Calculate $X_{s_{\hat{u}},s_{\hat{v}}}^{(t)} = f_{\ell,r}(\kappa_{s_{\hat{u}},s_{\hat{v}}}^{(t)})$
Calculate $X_{s_{\hat{u}},s_{\hat{v}}} = X_{s_{\hat{u}},s_{\hat{v}}}^{(1)} || X_{s_{\hat{u}},s_{\hat{v}}}^{(2)} || \cdots || X_{s_{\hat{u}},s_{\hat{v}}}^{(\xi)}$

Figure 6: On-line step of the CARPY+ scheme.

Specifically, given a finite field $\mathbb{F}_q$, all the $q^{\lambda+1}$ possible $(\lambda+1)$-dimensional vectors are examined in terms of Eqs. (2)∼(4). An exhaustive search can be accomplished in computational complexity $O(\xi \cdot q^{\lambda+1} \cdot N)$, is inefficient. In this section, we present two algorithms, SinLP and TwiLP, for constructing $\phi_{s_u}^{(t)}$, which are less time-consuming than the exhaustive search is in most of the cases of CARPY and CARPY+. Since our approaches take advantage of the efficiency of the linear programming, we first briefly review some terminology. After that, the algorithms, SinLP and TwiLP, will be presented.

A *linear program* (LP) of n variables is composed of a linear objective function of the form, $c_1x_1 + c_2x_2 + \cdots + c_nx_n$, where $c_1, \ldots, c_n$ are constant numbers; and a number of linear equality and inequality constraints of $x_1, \ldots, x_n$. The so-called *linear programming* is a technique for optimizing the objective function subject to the constraints. In other words, linear programming aims at finding the best assignment of $x_1, \ldots, x_n$ such that the objective function is optimized while the constraints are satisfied. For a LP, any solution $x_1, \ldots, x_n$ satisfying both the objective function and constraints is a *feasible solution*. Usually, a feasible solution can be thought of as a point in $\mathbb{R}^n$. Thus, geometrically, all the feasible solutions constitute a region, called *feasible region*, in $\mathbb{R}^n$. It can be shown that the feasible region must be a convex set if it is bounded. On the other hand, an integer linear program (ILP) is a LP with integrality constraints. As for the computational complexity, LP is shown to be solvable in polynomial time while ILP turns out to be $\mathcal{NP}$-hard [11].

2.5.1 SinLP Algorithm

Recall that our objective is to find more than two CRPs, $\phi_{s_u}^{(t)}$, satisfying Eqs. (2)∼(4), for $1 \le t \le \xi$ and $1 \le u \le N$. Unfortunately, to our knowledge, other than the exhaustive search, there is no technique useful for finding the solutions. An observation here is that, though the consideration of Eqs. (2)∼(4), which constitute the so-called *CRP criteria*, can provide all the possible CRPs, in fact, the consideration of a restricted version of Eqs. (2)∼(4), called *weak CRP criteria*, is sufficient for our use in most cases of CARPY and CARPY+. The weak CRP criteria are given as:

$$-\phi_{s_u}^{(t)} \cdot G_{-,s_v}^{(t)} \le \alpha_{u,v,r} \quad (5)$$

$$\phi_{s_u}^{(t)} \cdot G_{-,s_v}^{(t)} \le \beta_{u,v,r} \quad (6)$$

$$-A_{s_u,k}^{(t)} \le \phi_{s_u}^{(t)}(k) \le q - 1 - A_{s_u,k}^{(t)} \quad (7)$$

$$\phi_{s_u}^{(t)}(k) \in \mathbb{Z}, \quad (8)$$

where $u \ne v$, $1 \le u, v \le N$, $1 \le k \le (\lambda+1)$, $\phi_{s_u}^{(t)}(k)$ is the k-th element of $\phi_{s_u}^{(t)}$,

$$\alpha_{u,v,r} = (A_{s_u,-}^{(t)} \cdot G_{-,s_v}^{(t)}) - c_{\min}(A_{s_u,-}^{(t)} \cdot G_{-,s_v}^{(t)}, r), \quad (9)$$

and

$$\beta_{u,v,r} = c_{\max}(A_{s_u,-}^{(t)} \cdot G_{-,s_v}^{(t)}, r) - (A_{s_u,-}^{(t)} \cdot G_{-,s_v}^{(t)}). \quad (10)$$

An immediate observation is that the solutions satisfying the weak CRP criteria are a subset of the solutions satisfying the CRP criteria, because the weak CRP criteria can be regarded as the CRP criteria without considering the modular arithmetic. In particular, Eqs. (5) and (6) are, respectively, the same as Eqs. (2) and (3) except that the modular arithmetic is abandoned. Eq. (7) should be added to the weak CRP criteria. Otherwise, if an improper $\phi_{s_u}^{(t)}$ is found, the constructed pairwise keys X_{s_u,s_v} and X_{s_v,s_u} could be inconsistent between two nodes s_u and s_v. From an ILP point of view, all the CRPs can be thought of as the set of all the feasible solutions in the feasible region formed by the linear constraints of Eqs. (5)∼(8). In other words, finding a CRP amounts to finding a point in the feasible region. Thus, by introducing an arbitrary objective function, a CRP can be constructed by finding an optimum solution of the corresponding ILP. Recall that, since $\Phi_{s_u}^{(t)}$ is composed of several CRPs, the strategy we use here is to discover a CRP one by one. Once more than two CRPs are found, the construction of $\Phi_{s_u}^{(t)}$ is considered successful. In the following, we explain the construction of $\phi_{s_u}^{(t)}$, an element of $\Phi_{s_u}^{(t)}$.

Although solving an integer linear programming problem is known to be $\mathcal{NP}$-hard, a feasible solution, instead of optimizing an objective function, is sufficient for our use. In this aspect, standard techniques of handling the integer programming problem such as linear programming relaxation (LP-relaxation) [24] and randomized rounding [23] can be utilized to find the feasible solutions. The idea of SinLP is as follows. First, randomly select a $(\lambda+1)$-dimensional integer column vector c, and then construct an objective function $\phi_{s_u}^{(t)} \cdot c$. Thus, by considering two matrices $A^{(t)}$ and $G^{(t)}$, together with $\phi_{s_u}^{(t)} \cdot c$ as an objective function, we can construct an ILP, $ILP(c)$, as shown in the following.

Integer Linear Program $ILP(c)$

$$\text{minimize} \quad \phi_{s_u}^{(t)} \cdot c \tag{11}$$

subject to

$$-\phi_{s_u}^{(t)} \cdot G_{-,s_v}^{(t)} \leq \alpha_{u,v,r} \tag{12}$$

$$\phi_{s_u}^{(t)} \cdot G_{-,s_v}^{(t)} \leq \beta_{u,v,r} \tag{13}$$

$$-A_{s_u,k}^{(t)} \leq \phi_{s_u}^{(t)}(k) \leq q - 1 - A_{s_u,k}^{(t)} \tag{14}$$

$$\phi_{s_u}^{(t)}(k) \in \mathbb{Z} \tag{15}$$

Second, the LP-relaxation of $ILP(c)$ is conducted. Specifically, instead of solving $ILP(c)$, we solve the corresponding linear program, $LP(c)$, by using standard techniques such as the Simplex method [15] and Interior Point Method [15]. Here, the linear program $LP(c)$ is the same as $ILP(c)$ but without the constraint of the feasible solutions being limited to integers. Assume that a $(\lambda+1)$-dimensional rational vector π is the optimum solution, which is of course one of feasible solutions to the linear program $LP(c)$. Third, randomized rounding is conducted to transform the rational vector π into an integer vector $\tilde{\pi}$. In particular, $\tilde{\pi}_k = \lfloor \pi_k \rfloor$ with probability $\pi_k - \lfloor \pi_k \rfloor$ and $\tilde{\pi}_k = \lceil \pi_k \rceil$ with probability $1 - \pi_k + \lfloor \pi_k \rfloor$ for $1 \leq k \leq (\lambda+1)$, where π_k and $\tilde{\pi}_k$, respectively, denote the k-th element of the vectors π and $\tilde{\pi}$.

At first glance, SinLP looks efficient. However, it is, in general, inefficient because it is difficult to guarantee the number of the vectors $\tilde{\pi}$ coming from the rounding of π which are still feasible in ILP. Although iteratively applying randomized rounding on π can eventually find one of feasible solutions, there could be the case that most of the vectors $\tilde{\pi}$ are not feasible solutions anymore and, therefore, a large number of trials are required. An example of such a case is given in Example 2. For simplicity, we assume the existence of the bounded feasible region and at least one feasible solution to ILP in the subsequent discussion.

Example 2. *Consider the following integer linear program:*

Integer Linear Program (ILP-EX-1)

$$\text{minimize} \quad -3x_1 - 2x_2 \tag{16}$$

subject to

$$3x_1 + x_2 \leq 9 \tag{17}$$

$$x_1 + 3_2 \leq 7 \tag{18}$$

$$-x_1 + x_2 \leq 1 \tag{19}$$

$$x_1, x_2 \geq 0, x_1, x_2 \in \mathbb{N} \tag{20}$$

By using LP-relaxation, we can know that the optimum solution for x_1 and x_2 is $(2.5, 1.5)$. Unfortunately, in this case, only one rounding result $(2,1)$ is still a feasible solution, while the other three possible rounding results, $(2,2)$, $(3,1)$, and $(3,2)$ are not feasible solutions anymore, as depicted in Fig. 7.

2.5.2 TwiLP Algorithm

Geometrically, the rounding of an optimum solution in LP is equivalent to the shift of the corresponding point in the space. A possible explanation for the inefficiency of SinLP is that the optimum solution π is usually already in the close proximity of the boundary of the feasible region. For this reason, the point representing the optimum solution after the shift easily escapes from the feasible region. By assuming two objective functions $\mathcal{O}_1$ and $\mathcal{O}_2$ are opposite each other, we propose an algorithm, called TwiLP, in which LP-relaxation is used twice to find the two optimum solutions with respect to $\mathcal{O}_1$ and $\mathcal{O}_2$ in LPs and then performs randomized rounding on the point resulted from averaging those two

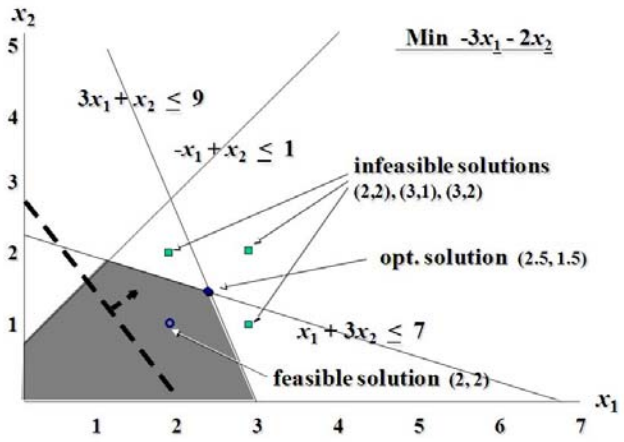

Figure 7: A bad example of directly using LP-relaxation and randomized rounding.

optimum solutions to search for a feasible solution of ILP. A more detailed description is as follows.

By a simple calculation, one can calculate the optimum solution $\pi_{LP(c)}$ for the LP-relaxation of $ILP(c)$. Let the column vector $d = -c$. Considering the integer linear program $ILP(d)$, one can also calculate the optimum solution $\pi_{LP(d)}$ for the LP-relaxation of $ILP(d)$. Then, the average $\bar{\pi} = (\pi_{LP(c)} + \pi_{LP(d)})/\omega$, where ω is an integer randomly selected, is calculated. Finally, randomized rounding is applied on the vector $\bar{\pi}$. The pseudo code of the TwiLP is shown in Fig. 8.

Algorithm: TwiLP($A^{(t)}$,$G^{(t)}$)
Scenario: The network planner wants to calculate a CRP, $\phi_{s_u}^{(t)}$
1 Randomly select an integer vector c
2 Set $d = -c$
3 Construct two ILPs, $ILP(c)$ and $ILP(d)$
4 Construct two LP-relaxations, $LP(c)$ and $LP(d)$
5 Calculate $\pi_{LP(c)}$ and $\pi_{LP(d)}$
6 Calculate $\bar{\pi} = (\pi_{LP(c)} + \pi_{LP(d)})/\omega$
7 **do**
8 Apply randomized rounding on $\bar{\pi}$ to obtain the vector $\mathfrak{V}$
9 **until** the vector $\mathfrak{V}$ is a feasible solution of $ILP(c)$

Figure 8: TwiLP Algorithm.

Since the feasible region of a LP is a convex set, all the points on the line segment connecting two arbitrary points within the feasible region are within the feasible region. Thus, it can be known that $\bar{\pi}$ is located within the feasible region. From the assumption of the existence of at least one feasible solution to ILP, it can also be known that at least one feasible solution can be found by rounding $\bar{\pi}$, resulting in the effectiveness of the TwiLP. On the other hand, since $\bar{\pi}$ comes from averaging two opposite points near the boundaries of the feasible region, geometrically, $\bar{\pi}$ is located near the center rather than the boundary of the feasible region. The number of integer feasible solutions surrounding $\bar{\pi}$ will be increased, guaranteeing the efficiency of TwiLP. Thus, we argue the TwiLP algorithm is effective and more efficient than exhaustive search and

SinLP. An example illustrating the superiority of TwiLP is shown in Example 3.

Example 3. *Consider the integer linear program $ILP(c)$. Recall that the optimum solution to its LP-relaxation is $(2.5, 1.5)$. Consider the following integer linear program whose objective function is in the opposite direction of that of ILP-EX-1 (Example 2):*

Integer Linear Program (ILP-EX-2)

$$\text{minimize} \quad 3x_1 + 2x_2 \tag{21}$$

subject to

$$3x_1 + x_2 \leq 9 \tag{22}$$

$$x_1 + 3_2 \leq 7 \tag{23}$$

$$-x_1 + x_2 \leq 1 \tag{24}$$

$$x_1, x_2 \geq 0, x_1, x_2 \in \mathbb{N} \tag{25}$$

It is obvious that the objective function considered now is opposite the objective function shown in Example 2. The optimum solution to the corresponding LP-relaxation is $(0, 0)$ by a simple calculation. The possible results by applying randomized rounding on the averaging result $(1.25, 0.75) = ((2.5, 1.5) + (0, 0))/\omega$, where $\omega = 2$, could be $(1, 0)$, $(1, 1)$, $(2, 0)$, and $(2, 1)$. This time, these four possible results, as shown in Fig. 9, are all feasible solutions.

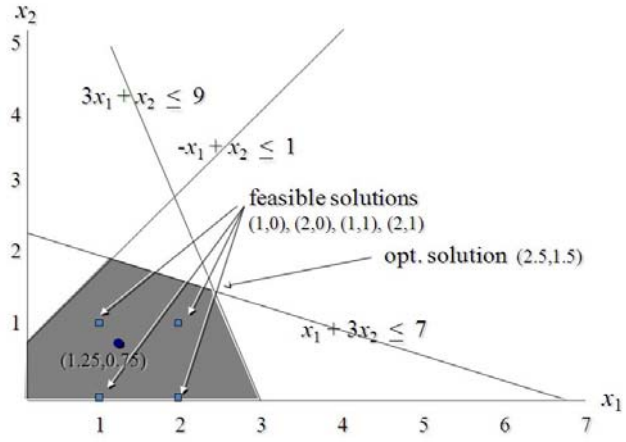

Figure 9: An example showing the effectiveness of efficiency by using TwiLP.

2.5.3 Implementation Issues

It can be observed that TwiLP could fail to find the CRPs even if several iterations are performed. From the implementation perspective, there exists another way to construct CRPs. The observation here is that if the elements of $G^{(t)}$ are relaxed to small floating point numbers instead of integers, then Eqs. (5)∼(8) can be easily satisfied. Specifically, $G^{(t)}$ can be constructed as described in Sec. 2.4, followed by a division using a large integer γ (*e.g.*, $\gamma \geq 100$). Note that after such scalar division, the property of $G^{(t)}$ that any $\lambda + 1$ columns are linearly independent can still be kept. The algorithm of constructing CRPs is described as follows. At first, an integer vector $\phi_{s_u}^{(t)}$ is randomly generated. For $\phi_{s_u}^{(t)}$, we examine if Eqs. (5)∼(8) are satisfied. The above procedure is repeated until a satisfiable vector is found. Despite its similarity to exhaustive search, in practice, after the relaxation of $G^{(t)}$, such a simple randomized algorithm is very efficient and effective for generating CRPs. For example, when $q = 2^{16} - 15, \lambda = 128$, and $r = 14$, only 10 seconds are needed to generate a CRP.

Table 1: Storage Overhead of CARPY and CARPY+ (in Byte).

Scheme	Flash Memory	RAM
CARPY	1070	2244
CARPY+	1224	2176

Table 2: Computation Overhead of CARPY and CARPY+.

Scheme	Time (in seconds)	Cycle Count
CARPY	0.2379	1903747
CARPY+	0.2620	2096170

3. PERFORMANCE EVALUATION

The prototypes of both the CARPY and CARPY+ schemes have been implemented on the TelosB compatible mote (Micro-Controller: TI MSP430F1611; Flash Memory: 48KB+256B; RAM: 10KB; Radio Chipset: ChipCon CC2420). The programming tool we used is the native C compiler on IAR Embedded Workbench† 3.40.1.9, instead of TinyOS. In our experiments, the parameters were set as follows. The number N of sensor nodes was 1024 and the desired key length L was 128. In addition, $q = 2^{16} - 15$, $\lambda = 128$, and $r = 14$. We used the diagnostic and profiling outputted from IAR Embedded Workbench to estimate storage and computation overhead. It should be noted that the elements of $G^{(t)}$ used in the experiments were selected and represented in floating points for ease of implementation.

3.1 Storage Overhead

If CARPY is used, then, for sensor node s_u, the row vectors $A_{s_u,-}^{(t)}$ and column vectors $G_{-,s_u}^{(t)}$ are needed to be stored. Since ξ rounds of CARPY need to be performed independently, the storage overhead is therefore $O(2 \cdot \xi \cdot \lambda)$. If CARPY+ is used, for sensor node s_u, only row vectors $A_{s_u,-}^{(t)}$ and an element s need to be stored. Since the CARPY+ scheme also needs to be performed ξ rounds, the storage overhead for CARPY+ is, thus, $O(\xi \cdot \lambda)$. The details of storage overhead in our experiment are shown in Table 1.

3.2 Computation Overhead

For different s_u and s_v, $\lambda + 1$ multiplications and λ additions are needed to carry out the multiplication of $A_{s_u,-}$ and G_{-,s_v} in each round of CARPY. The computation overhead of CARPY+ is larger than that of CARPY because each node calculates the needed column vectors by itself. From the s_u point of view, after the calculation of φ^{s_v}, which needs at most s_v multiplications, $\lambda + 1$ multiplications and λ additions are sufficient to simultaneously carry out the generation of G_{-,s_v}, and the multiplication of $A_{s_u,-}$ and G_{-,s_v} by using Horner's rule in each round of execution of CARPY+. The computation overhead obtained from our experiments is shown in Table 2.

3.3 Communication Overhead

In CARPY, the communications happen only when two sensor nodes exchange their respective column vectors. As the length of a column vector is $O(\lambda)$ and the expected hop distance between two

†Availiable at: www.iar.com

arbitrary nodes in a random flat network is $O(\sqrt{N})$, the communication overhead is therefore $O(\xi \cdot \lambda \cdot \sqrt{N})$. On the other hand, it can be easily observed from the scheme described in Fig. 6 that there is no communication needed in the CARPY+ scheme.

3.4 Security Analysis

In this paper, we assume that four categories of attacks could be mounted by the adversary. They are *eavesdropping attack*, *node capture attack*, *routing layer attack*, and *physical layer attack*. The resilience of CARPY and CARPY+ to these four possible attacks is described in Sec. 3.4.1∼Sec. 3.4.4, respectively. In addition to the aforementioned attacks, Denial of Service (DoS) is a common strategy mounted by the adversary to attack networks. While DoS attack has no direct impact on the information leakage of the key establishment schemes, an ill-designed key establishment scheme easily allows DoS attack. The immunity of CARPY and CARPY+ to DoS attack will be described in Sec. 3.4.5. The mixed attack is considered in Sec. 3.4.6. In the following, we denote the nodes in control of the adversary as *insider nodes*, while the *legitimate nodes* are the nodes having not been compromised by the adversary.

3.4.1 Eavesdropping Attack

In our assumption, a global eavesdropper is involved in the network so that all the traffic on the network will be immediately known by the adversary. In the CARPY scheme, the message exchanged between nodes is only the column vectors of the matrix G, which is assumed to be publicly known by everyone including the adversary. On the other hand, there is no message exchanged between nodes during the key establishment of the CARPY+ scheme. Thus, the adversary gains nothing about the pairwise key between each pair of nodes by using eavesdropping attack.

3.4.2 Node Capture Attack

The CARPY and CARPY+ schemes can be regarded as a generalization of Blom's scheme. In particular, the construction of the matrix $W^{(t)}$ in CARPY and CARPY+ comes from the elements of the matrix $A^{(t)}$ of Blom's scheme, on which the CRPs are applied. Due to this observation, directly inherited from Blom's scheme, the security of both CARPY and CARPY+ can be perfectly guaranteed before $\lambda+1$ sensor nodes are captured by an adversary. Therefore, we only consider the case where the number x of captured nodes is larger than $\lambda+1$, *i.e.*, $x \geq \lambda+1$.

After CRPs have been applied to $A^{(t)}$ to construct the matrix $W^{(t)}$, the relation between the matrices $A^{(t)}$ and $D^{(t)}$ in Blom's scheme does not exist any more. Here, a metric, called *breaking complexity* (BC), for evaluating the difficulty of recovering $D^{(t)}$ is defined. While the least number of nodes necessary to be compromised, which is easily known to be $\lambda+1$ for CARPY and CARPY+, acts as a metric for evaluating the hardness of recovering $D^{(t)}$ in terms of physical attack, breaking complexity is defined in terms of computational effort the adversary needs to pay. A lemma describing the breaking complexity of compromising the matrix $D^{(t)}$ is as follows.

Lemma 1. *Given that* $|\Phi_{s_u}^{(t)}| \geq \rho$, $1 \leq t \leq \xi$, *and* $1 \leq u \leq N$, *the breaking complexity (BC) for recovering the matrices* $D^{(t)}$ *is* $\Omega(\xi \cdot \rho^{\lambda+1})$ *for both the CARPY and CARPY+ schemes.*

The intuition behind Lemma 1 is that, after capturing a set $X \subset \mathcal{I}$ of $\lambda+1$ nodes the adversary must guess the correct CRPs applying on the row vectors of $\lambda+1$ captured nodes. Since a successful guess for one attempt is with probability $O(1/\prod_{b\in X}|\Phi_b^{(t)}|)$ and there are ξ rounds of CARPY and CARPY+ needed to be performed, if $|\Phi_{s_u}^{(t)}| \geq \rho$ for all u and t, the required computational effort is $\Omega(\sum_{t=1}^{\xi}\prod_{b\in X}|\Phi_b^{(t)}|) = \Omega(\xi \cdot \rho^{\lambda+1})$. A formal proof is as follows.

Proof: (sketch) Since the matrices $W_{s_u,-}^{(t)}$ are constructed independently for each round $t = 1 \ldots \xi$, the breaking complexity for different matrices is the same. Recall that $W_{s_u,-}^{(t)} = A_{s_u,-}^{(t)} + \phi_{s_u}^{(t)}$ holds. After x sensor nodes, $\{s_{z_1}, \ldots, s_{z_x}\} \subset \{s_1, \ldots, s_N\}$, are captured by the adversary, a system of linear equations is obtained as follows:

$$W_{s_{z_\sigma},\varsigma}^{(t)} = A_{s_{z_\sigma},\varsigma}^{(t)} + \phi_{s_{z_\sigma}}^{(t)}(\varsigma) \tag{26}$$

$$= \sum_{k=1}^{\lambda+1}(G^{(t)})_{s_{z_\sigma},k}^{T} \cdot D_{k,\varsigma}^{(t)} + \phi_{s_{z_\sigma}}^{(t)}(\varsigma), \tag{27}$$

where $1 \leq \sigma \leq x, 1 \leq \varsigma \leq \lambda+1$, and $(G^{(t)})_{s_{z_\sigma},k}^{T}$ denotes the element on the s_{z_σ}-th row and k-column of the matrix transpose of $G^{(t)}$. In the linear system shown in Eq. (27), $W_{s_{z_\sigma},\varsigma}^{(t)}$ and $(G^{(t)})_{s_{z_\sigma},k}^{T}$ are known by the adversary while $D_{k,\varsigma}^{(t)}$ and $\phi_{s_{z_\sigma},\varsigma}^{(t)}$ are unknown to the adversary. It can be seen that the total number of linear equations is $x(\lambda+1)$ and the total number of unknowns is $x(\lambda+1)+\lambda(\lambda+1)/2$. Obviously, finding a unique solution for $D^{(t)}$ is impossible.

A strategy possibly adopted by the adversary to find a unique solution for $D^{(t)}$ in the linear system shown in Eq. (27) is to reduce the number of unknowns. Since the elements in $D^{(t)}$ are chosen from the finite field $\mathbb{F}_q$ arbitrarily and independently, the number of unknowns coming from $D^{(t)}$ cannot be reduced and is still $\lambda(\lambda+1)/2$. The remaining possibility for the adversary is to select a group $\Lambda \subset \mathcal{I}$ of $w+1$ sensor nodes $\{s_{\hat{z}_1}, \ldots, s_{\hat{z}_{w+1}}\}$ with row vectors $\{W_{s_{\hat{z}_1}}^{(t)}, \ldots, W_{s_{\hat{z}_{w+1}}}^{(t)}\}$ on which the same CRP is applied. In this case, the number of unknowns due to CRPs can be reduced by $w(\lambda+1)$. Define such a kind of group as an *ill-perturbed group*. According to this observation, the adversary may identify one or more ill-perturbed groups to reduce the number of unknowns in the linear system shown in Eq. (27) so that a unique solution for $D^{(t)}$ can be determined. The probability that an ill-perturbed group is found is $O(1/(\prod_{\hat{b}\in\Lambda}|\Phi_{\hat{b}}^{(t)}|))$, because CRPs are randomly and independently applied on the matrix $A^{(t)}$. Assume that ζ ill-perturbed groups $\{G_1, \ldots, G_\zeta\}$, each of which consists of S_η nodes, $1 \leq \eta \leq \zeta$, are identified by the adversary. To recover $D^{(t)}$, it is necessary to satisfy $\sum_{\eta=1}^{\zeta}(S_\eta - 1) \geq \lambda+1$. As a result, the probability for breaking a matrix $D^{(t)}$ is the same as the probability of correctly identifying these ζ ill-perturbed groups, which can be calculated as $1/(\prod_{\epsilon=1}^{\zeta}\prod_{\hat{b}\in G_\epsilon}|\Phi_{\hat{b}}^{(t)}|)$. Since the adversary needs to recover ξ independently constructed matrices, $D^{(t)}$, the probability of correctly identifying these ζ ill-perturbed groups for those ξ matrices, $D^{(t)}$, can be calculated as $1/(\sum_{j=1}^{\xi}\prod_{\epsilon=1}^{\zeta}\prod_{\hat{b}\in G_\epsilon}|\Phi_{\hat{b}}^{(j)}|)$. It results in the

$$\Omega(\sum_{j=1}^{\xi}\prod_{\epsilon=1}^{\zeta}\prod_{\hat{b}\in G_\epsilon}|\Phi_{\hat{b}}^{(j)}|) \tag{28}$$

computational complexity. For simplicity, under the assumption that $|\Phi_{s_u}^{(t)}| \geq \rho$, we have

$$1/\sum_{j=1}^{\xi}\prod_{\epsilon=1}^{\zeta}\prod_{\hat{b}\in G_\epsilon}|\Phi_{\hat{b}}^{(j)}| \leq 1/\rho^{\lambda+1}, \tag{29}$$

and thus the breaking complexity can be written as $\Omega(\xi \cdot \rho^{\lambda+1})$. □

Table 3: Relation Between Various Parameters

L	q	ℓ	λ	r	$\lvert\Phi_{s_u}^{(t)}\rvert$	ξ	BC
128	$2^{16}-15$	16	32	14	≥ 2	64	$\geq 2^{36}$
128	$2^{16}-15$	16	64	14	≥ 2	64	$\geq 2^{70}$
128	$2^{16}-15$	16	128	14	≥ 2	64	$\geq 2^{135}$
128	$2^{32}-5$	32	128	28	≥ 2	32	$\geq 2^{134}$

The security levels under different settings can be found in Table 3. In addition to recovering the matrices $D^{(t)}$, the adversary may also try to derive the CRP applied on each captured node by using the methods described in Sec. 2.5. Given $|\Phi_{s_u}^{(t)}| \geq \rho$, if $\lambda+1$ nodes have been captured, since $\lambda+1$ CRPs should be simultaneously and correctly guessed, exhaustive search incurs $\Omega(\xi \cdot \rho^{\lambda+1})$ computation overhead, which is infeasible for the adversary. On the other hand, TwiLP algorithm can be utilized by the adversary. However, the parameters, c, d, and ω, used in TwiLP are only known by the network planner and unknown by the adversary. The adversary is forced to examine $\Omega(\xi \cdot \rho^{\lambda+1})$ possibilities for the captured nodes. Therefore, it is also inefficient for the adversary to find CRPs by using TwiLP.

3.4.3 Routing Layer Attack

Routing layer attacks typically focus on disrupting the routing mechanisms. The adversary may not gain information about the pairwise key by directly mounting routing layer attacks. However, routing layer attacks could be used to either hinder the legitimate nodes from key establishment or even strengthen the effectiveness and efficiency of node capture attack. Accordingly, attention to the study of the resilience of key establishment schemes to routing layer attacks is of primary importance.

In general, according to different goals, routing layer attacks can be divided into three categories. 1) *Attacks on the route discovery process*: In such attacks, the adversary prevents the legitimate nodes from establishing routing paths by sending or flooding bogus routing information. In an extreme case, this type of attack can achieve DoS-like effect on the networks. 2) *Attacks on the route selection process*: Instead of preventing legitimate nodes from establishing routing paths, the adversary uses this type of attack to increase the probability that the insider nodes become a part of the routing path between legitimate nodes. After joining one routing path, the insider nodes can either eavesdrop or even manipulate the messages transmitted through the path. For example, Sybil attack and sink hole attack belong to this category, where the former takes advantage of the malicious node with multiple IDs and the latter utilizes the fake routing information to attract traffic. 3) *Attacks after establishing the routing paths*: This type of attack primarily aims at hindering the legitimate nodes from using the routing paths which have been established. For example, the black hole attack [13], in which all the packets transmitted through the malicious nodes will be dropped, belongs to this category. An overview of routing layer attacks in WSNs can be found in [13].

All the key establishment schemes for WSNs proposed in the literature suffer from routing layer attacks because communications are necessary in the key establishment procedure. For CARPY, since two nodes establish their pairwise key by exchanging their respective column vectors of G, CARPY is not resilient to routing layer attacks either. Nevertheless, when the CARPY+ scheme is exploited, since no communication is required for establishing the pairwise key, routing layer attacks cannot disrupt the key establishment procedure. Hence, the resilience of CARPY+ against routing layer attacks can be guaranteed.

3.4.4 Physical Layer Attack

Physical layer attack usually means a jamming attack [25], in which the adversary disrupts the capability of transmitting and receiving packets for some specified nodes through radio frequency interference. Solely exploiting physical layer attack cannot help the adversary gain the information about the pairwise key, but it can block the communications among a group of selected nodes so that the key establishment has the possibility of not being accomplished.

Since communications are involved in establishing keys for all the key establishment schemes for WSNs proposed in the literature, after physical attacks are applied on a group of selected nodes by the adversary, all the nodes in the group will be unable to establish the keys with the other nodes, which are probably outside of the group. The CARPY scheme encounters the same circumstance, *i.e.*, key establishment cannot be accomplished, because the column vectors of G should be exchanged between two nodes that would like to have a key sharing. Fortunately, the CARPY+ scheme is considered to be robust to the physical layer attacks since the pairwise key can be calculated without the need of communication.

3.4.5 Denial of Service (DoS) Attack

Though a DoS attack [25] cannot be directly used to help the adversary acquire the key between two legitimate nodes, the networks exploiting certain KPD schemes could attract a great threat of DoS attacks. In particular, most of P-KPD schemes suffer from DoS attacks. Note that in this paper, we only emphasize the DoS attack incurred by applying key establishment schemes. The so-called path-based DoS (PDoS) attack [9] is a strategy adopted by the adversary to exhaust nodes' limited resources by overwhelming sensor nodes with useless or spurious packets. In this paper, PDoS attack is defined such that the adversary sends a bogus messages, claiming to perform either shared-key discovery or path-key establishment with a victim node. Consider that P-KPD is used in the networks. If the bogus message sent from a PDoS attack claims to perform path-key establishment, the nodes on the path to the victim node consume their energy by not only decryption and re-encryption, but also forwarding the futile message. If the bogus message claims to perform shared-key discovery, the nodes on the path to the victim node waste their energy to forward the futile message and the victim node will be busy with the decryption and comparison of a large number of the received futile Merkle puzzles. In general, PDoS attack can affect all the key establishment schemes that need communications between sensor nodes. Note that, although the authentication scheme [29] may be exploited to mitigate the effect of PDoS attacks, it is useless under the consideration of insider nodes.

Since communications are required in CARPY, a PDoS attack can be launched so that not only the radio function of the nodes should be turned on to receive the packets but also the calculation for key establishment is performed. Although the energy waste in CARPY, compared with the existing schemes, is slightly reduced, a PDoS attack involves a moderate level of energy waste to the network. However, because no communication or interaction between nodes is required for establishing the pairwise key, in CARPY+, PDoS attacks can be resisted.

3.4.6 Mixed Attack

As stated in Sec. 1.1, the adversary could simultaneously mount several kinds of attacks to compromise the security of the key establishment schemes more efficiently than a single attack does.

From the previous discussions, it can be observed that when the pairwise key can be established without needing communications, the key establishment scheme is resilient to the mixed attack. In other words, simultaneously mounting several attacks gains nothing more than solely mounting node capture attack.

3.5 Energy Consumption

In this section, we utilize a model similar with the one considered in [22] and [26] to estimate the energy consumption of CARPY and CARPY+, and then compare it with the other schemes. In general, we consider the networks composed of N sensor nodes, in which the packet loss rate p_{loss} of each link between any two neighboring nodes is the same. In other words, delivery of single packet will fail with probability p_{loss}. Assume that the byte-length of the maximum payload in a packet is L_{packet}. The expected length of the shortest path connecting two arbitrary nodes in the network is assumed to be h. Let e_e and e_d be the energy consumption for encryption and decryption, respectively. Denote the energy consumption of transmitting and receiving one packet as e_t and e_r, respectively. In the following, we first formulate the energy consumption for key establishment between two nodes that are not neighboring in the P-KPD, D-KPD, RPB, CARPY, and CARPY+ schemes. Then, a comparison among them will be presented.

Probabilistic Key Pre-Distribution. The schemes [4, 7, 8, 10, 14] are all within the same framework of P-KPD. Without loss of generality, we consider only the scheme proposed in [10], but the evaluation results can be naturally extended to [4, 7, 8, 14]. We show how to calculate the energy consumption E_{prob} for P-KPD, which is composed of the energy consumption E_{prob}^{comm} of communications and the energy consumption E_{prob}^{comp} of computation, as follows.

Suppose the P-KPD scheme [10] is applied to the network. Let the key ring size for each node be m, let the key pool size be S, and let the key ring of the sensor node s_u be a set $\{k_1^{s_u}, \ldots, k_m^{s_u}\}$. The probability p_c that two nodes share at least one common key in their respective key rings can, thus, be computed as $1 - ((\binom{S}{2m} \cdot \binom{2m}{m})/(\binom{S}{m}^2))$. For simplicity, we assume that each node has established either the shared-key or the path-key with its neighboring nodes after sensor deployment. Let L_z be the byte-length of *Merkle puzzle packet* which is of the form,

$$\langle id, \aleph_1, E_{k_1^{id}}(\aleph_1), \ldots, \aleph_m, E_{k_m^{id}}(\aleph_m)\rangle, \tag{30}$$

where $\aleph_1, \ldots, \aleph_m$ are random words. To have a common key with the node s_v, the node s_u tries to find their shared-key by sending the Merkle puzzle packet to s_v. With probability p_c, the energy consumption of communications for the shared-key discovery between s_u and s_v is $h \cdot (\frac{e_t}{1-p_{loss}} + e_r) \cdot (\lceil \frac{L_z}{L_{packet}} \rceil + 1)$. However, after the above communications, with probability $1 - p_c$, they find that they do not have the shared-key so that the path-key establishment is necessary, resulting in additional energy consumption $h \cdot (\frac{e_t}{1-p_{loss}} + e_r) \cdot (\lceil \frac{L}{8 \cdot L_{packet}} \rceil + 1)$ required for transmitting the path-key. Thus, the energy consumption E_{prob}^{comm} of communications for P-KPD can be estimated as:

$$\begin{aligned} &h \cdot (\frac{e_t}{1-p_{loss}} + e_r) \cdot (\lceil \frac{L_z}{L_{packet}} \rceil + 1) \\ &+(1-p_c) \cdot h \cdot (\frac{e_t}{1-p_{loss}} + e_r)(\lceil \frac{L}{8 \cdot L_{packet}} \rceil). \end{aligned} \tag{31}$$

The energy consumption E_{prob}^{comp} of computation for probabilistic KPD can be estimated as:

$$m \cdot e_e + m^2 \cdot e_d + (1-p_c) \cdot h \cdot (e_e + e_d), \tag{32}$$

because m encryptions in s_u and m^2 decryptions for finding the matching key in s_v are required in shared-key discovery while in path-key establishment each pair of consecutive nodes on the key path performs one decryption and re-encryption. It should be noted that for simplicity certain hidden costs are ignored in our calculation, for example, the energy consumption for establishing the shared-key or path-key between two neighboring nodes.

Deterministic Key Pre-Distribution. Here, we consider the two-dimensional PIKE scheme proposed in [3]. Note that the evaluation results can be easily extended to the other deterministic KPD schemes [5, 6]. From the key assignment of each node, the probability p_c that two nodes share a common key in the deterministic key establishment schemes can usually be directly derived. For example, $p_c = 2(\sqrt{N}-1)/(N-1)$ is derived in the two-dimensional PIKE scheme. We also assume that each node has established either the shared-key or the path-key with its neighboring nodes after sensor deployment. With probability p_c, nodes s_u and s_v have a shared-key; thus, no communication is needed. With probability $1 - p_c$, the path-key establishment is required to be performed between nodes s_u and s_v. Consequently, the energy consumption $E_{deter} = E_{deter}^{comm} + E_{deter}^{comp}$ for deterministic KPD can be derived by calculating the energy consumption E_{deter}^{comm} of communications as:

$$2 \cdot h \cdot (1-p_c)(\frac{1}{1-p_{loss}} \cdot e_t + e_r) \cdot \lceil \frac{L}{8 \cdot L_{packet}} \rceil, \tag{33}$$

and the energy consumption $E_{deter}^{comp} = 2 \cdot (e_e + e_d)$ of computation.

Random Perturbation Based Key Establishment. The Random Perturbation Based (RPB) scheme [30] is the only scheme to take advantage of random perturbation to strengthen security and reduce communication overhead. The energy consumption of the RPB scheme is $E_{RPB} = E_{RPB}^{comm} + E_{RPB}^{comp}$. Since executing RPB one time derives a part of the pairwise key, without loss of generality, we assume that ξ rounds of RPB also need to be performed. In addition, a security parameter needed in the RPB scheme is also assumed to be λ for simplicity. Since the key sharing between each pair of nodes is guaranteed and proven in [30], the energy consumption E_{RPB}^{comm} of communications for the RPB scheme can be easily estimated as:

$$h \cdot (\frac{e_t}{1-p_{loss}} + e_r) \cdot \lceil \frac{L_M}{L_{packet}} \rceil, \tag{34}$$

where L_M is the byte-length of a hash. When the node s_u wants to establish a pairwise key with s_v, the primary energy consumption E_{RPB}^{comp} of computation for RPB scheme can be estimated as $\xi \cdot \lambda \cdot (e_a + e_m) + \xi \cdot e_e + (\xi - 1) \cdot e_{XOR}$ for s_u, and $\xi \cdot \lambda \cdot (e_a + e_m) + 3 \cdot \xi \cdot e_e + 3^\xi \cdot e_{XOR}$ for s_v, where e_a and e_m, respectively, denote the energy consumption of accomplishing the addition and multiplication of two integers, and e_{XOR} means the energy consumption of accomplishing ex-clusive OR (XOR) operation between two bit-strings. Here, as in the experiment conducted in [30], the energy consumed for calculating a hash is replaced by the energy consumed by a block cipher.

CARPY and CARPY+ schemes. We calculate the energy consumptions E_{CARPY} and E_{CARPY+} for both CARPY and CARPY+, respectively. E_{CARPY} can be estimated as $E_{CARPY} = E_{CARPY}^{comm} + E_{CARPY}^{comp}$. Here, E_{CARPY}^{comm} is calculated as:

$$2 \cdot h \cdot \xi \cdot \lceil \frac{\ell \cdot (\lambda + 1)}{8 \cdot L_{packet}} \rceil \cdot (\frac{e_t}{1-p_{loss}} + e_r), \tag{35}$$

because the respective column vectors of G of two nodes s_u and s_v need to be exchanged. As for E_{CARPY}^{comp}, it can be computed as $2 \cdot \xi \cdot ((\lambda + 1) \cdot e_m + \lambda \cdot e_a)$, as the primary task needed to be

performed by two ends is to calculate an inner product.

Since, in the CARPY+ scheme, a pairwise key can be directly constructed between any pair of nodes without the need of communication, the energy consumption E^{comm}_{CARPY+} is zero. As to the computation needed for the construction of the common key between two nodes s_u and s_v, the node s_u should first generate the corresponding column vector $G^{(t)}_{-,s_v}$, requiring $\xi \cdot \log(s_v + 1)$ multiplications if the set $\{\varphi^{2^i} | i \in \mathbb{N}, 2^i \leq s_N\}$ is stored in each node. After that, the computation of the inner product, which is similar to the one used in the CARPY scheme, is carried out to construct the common key. As a result, if the maximum id of nodes is N, the energy consumption E^{comp}_{CARPY+} of the CARPY+ scheme for the computation is at most

$$\xi((2(\lambda + 1) + \log(s_N + 1))e_m + 2\lambda e_a). \quad (36)$$

Energy Calculation. We consider the energy consumption of several operations implemented on the TelosB mote. CC2420 consumes 18.8 mA current for receiving and 17.4 mA for transmission. If the battery voltage and the data rate are set to 3.6V and 250kbps, respectively, then the energy for receiving one byte needs 2.1658μJ and the energy for transmitting one byte needs 2.0045μJ. If the default setting of TinyOS packet is considered, e_r and e_t can be calculated as 77.9688μJ and 72.162μJ, respectively. Note that, although the default packet size is 36 bytes in TinyOS, only 29 bytes are used for payloads, *i.e.*, $L_{packet} = 29$. We compare these schemes in two cases[‡]: $L_{packet} = 29$ and $L_{packet} = 102$. In [16], it shows that $e_e = 9\mu$J for AES-128. When counter mode (CTR mode) is used, energy consumption for both decryption and encryption is the same. Here we assume that CTR mode is used and, therefore, $e_d = 9\mu$J. In our experiments, e_a, e_m, and e_{XOR} are about 0.2164μJ, 0.2405μJ, and 0.0132μJ, respectively. Since AES-128 is considered, we assume that the random words in the Merkle puzzle packet are selected to be $L = 128$ bits. For simplicity, we assume that L_M is 16 bytes (128 bits). In a network whose N nodes are evenly and randomly deployed, the expected hop distance is $O(\sqrt{N})$, *i.e.*, $h = O(\sqrt{N})$. For P-KPD, assume that $m = 75$ and $S = 10,000$. For the RPB, CARPY, and CARPY+ schemes, assume that $\ell = 64$, $\xi = 8$, and $\lambda = 16$. Under the setting of $p_{loss} = 30\%$, the results of the energy consumption of the P-KPD, D-KPD, RPB, CARPY, and CARPY+ schemes are depicted in Fig. 10. Note that, due to a simplified assumption used in P-KPD and D-KPD that each node has established the key with each of its neighbors, their energy consumption could be dramatically larger than those shown in Fig. 10 when, for example, the network is loosely connected or the energy consumption for the route discovery used in path-key establishment is taken into account.

Due to the fact that CARPY incurs larger packet overhead, CARPY consumes more energy than D-KPD and RPB. Note that if the parameters, such as ℓ, r, and ξ, are chosen properly, the overhead can be further reduced. Fortunately, it can be easily observed that the energy consumption in the proposed CARPY+ scheme is substantially smaller than all the known schemes chosen for comparisons. For example, in a network with 10^4 nodes, the energy consumption of P-KPD [10] is about 100 times greater than that of CARPY+. In particular, the scalability of CARPY+ is superior to the other schemes because only the energy consumption of CARPY+ is independent of the network size.

3.6 Comparisons

We present a comprehensive comparison among CARPY, CARPY+, and the other key establishment schemes, from the sensor-key criteria point of view. The results are shown in Table 4 and are described in detail in the following.

[‡]The default maximum payload size in TinyOS is 29 bytes and the maximum payload size in IEEE 802.15.4 is 102 bytes.

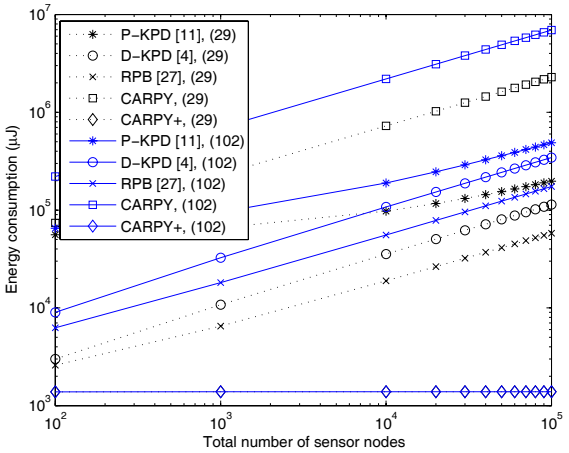

Figure 10: Energy Consumption for different key establishment schemes (The number in the parenthesis indicates the payload size).

Table 4: Comparisons Between Different Key Establishment Schemes (In Terms of The Sensor-Key Criteria)

	RVA	DGKE	RNC	EFF	RDND
TKD [1, 2]		✓	✓	✓	✓
P-KPD [4, 8, 10, 14]					✓
D-KPD [3, 5, 6]					
RPB Scheme [30]		✓			✓
L-KPD [7, 12]					✓
LEAP [28]					✓
CARPY scheme		✓	✓		✓
CARPY+ scheme	✓	✓	✓	✓	✓

Resilience to Various Attacks (RVA). Even without considering the mixed attack, since the key or message could be known by the intermediate nodes between two nodes that want to achieve key sharing if path-key establishment is applied, the P-KPDs [4, 8, 10, 14], D-KPDs [3, 5, 6], L-KPDs [7, 12], and LEAP [28] are all vulnerable to the node capture attack. For TKD schemes, which do guarantee key establishment for any pair of nodes without needing to perform path-key establishment, the guaranteed security depends on a pre-determined threshold, which is not usually large enough for adequate protection of WSNs. As the RPB scheme guarantees key sharing for any pair of nodes, it relies on the communications to establish pairwise keys. However, it is not considered to be resilient to various attacks because the key establishment always fails when attacks, such as jamming [25] and selective forwarding attack [13], are mounted by the adversary. In addition, the RPB scheme is also susceptible to DoS attacks, because the adversary can always send spurious packets and force the victim node to proceed with a large number of futile computations.

Our proposed CARPY scheme also suffers from the same problems the RPB scheme encounters. For CARPY+, since there is no

need of communications in key establishment, all the eavesdropping, node capture, routing layer, and physical attacks cannot degrade the security between a pair of nodes having not been compromised. Even better, for the same reason, key establishment can be guaranteed to be successfully accomplished whenever the aforementioned attacks occur. This implies the strongest survivability. In addition, it does not incur DoS attacks in that key establishment is carried out in a spontaneous way. Thus, a message claiming the request for establishing keys will simply be dropped. Hence, the proposed CARPY+ scheme is considered to be a key establishment scheme satisfying RVA.

Directed and Guaranteed Key Establishment (DGKE). Due to the storage limitations of each sensor node, pre-determined keys cannot be preloaded into each pair of nodes if P-KPDs, D-KPDs, and L-KPDs are applied, leading to the partial connectivity of key sharing. Hence, there always exists pairs of nodes that do not have shared-keys and require path-key establishment. For LEAP, a node can establish common keys with its neighbors only. However, we can know from the descriptions of the proposed CARPY and CARPY+ schemes in Fig. 4 and Fig. 6 that key sharing can be always established between any two nodes.

Resilience to Network Configurations (RNC). While L-KPDs and LEAP obviously cannot be applied to mobile networks, the efficiency for establishing key sharing will be significantly decreased if P-KPDs and D-KPDs are considered in the mobile networks. In fact, the P-KPDs, D-KPDs, and L-KPDs are also subject to the density of the network. This can be observed by simply considering an extreme case, *i.e.*, a network in which each node has only a few, say 2 or 3, neighboring nodes. The effectiveness and efficiency of key establishment in such networks will be substantially degraded. The RPB scheme, though, is guaranteed to be applicable in mobile networks and can be applied on the networks with diverse topologies. However, the advantages come from sacrificing its applicability to heterogeneous networks because the IDs of sensor nodes in the RPB scheme should be artificially assigned. In heterogeneous networks, assigning or modifying IDs to certain devices is not always feasible because IDs could have been fixed in the sensing hardware like the MAC address in current network interface cards (NICs). Moreover, the number of bits required for representing IDs of nodes could be dramatically increased in a network with a large number of nodes, implying an inefficient use of remaining packet size to carry data.

For both CARPY and CARPY+, the keying materials are always at most one row vector of length $\lambda + 1$ plus one column vector of length $\lambda+1$. It works irrespective of the network scale. In addition, key establishment is performed independent of deployment knowledge. In fact, CARPY and CARPY+ can be carried out with arbitrary network topology, because the pairwise key is calculated by the node itself without the knowledge of network topology. Finally, the CARPY and CARPY+ schemes do not assume the knowledge of hardware; thus, they can be considered to be hardware independent and are applicable in heterogeneous networks.

Efficiency (EFF). It can be easily observed that at least communications are necessary for any two nodes without having the shared-key as long as the requirement of DGKE in the sensor-key criteria cannot be satisfied. For both the RPB and the proposed CRPV schemes, although the key sharing can be achieved for any pair of nodes, the exchange of the publicly known keying materials is required. In a randomly flat network with N nodes, the average hop distance between two arbitrary nodes is $O(\sqrt{N})$. The communications for establishing the keys not only at least consumes the other $O(\sqrt{N})$ nodes' energy, but also incurs tremendous latency for key establishment because of the high packet loss rate and long path length. As for the efficiency of the CARPY+ scheme, it does not require any message exchange and only involves a constant number of additions and multiplications. Hence, the energy saving resulted from the communication-free property of CARPY+ scheme is very significant.

Resilience to Dynamic Node Deployment (RDND). In D-KPDs, key sharing between nodes usually relies on some fixed structures, such as the hypercube in [3], the expander graph in [6], and the symmetric design in [5]. If the construction of the underlying structure does not consider the nodes to be deployed in the future, on-the-fly addition of nodes is usually infeasible. A possible solution is to construct the structure with the consideration of a large number of nodes, but it also increase the storage overhead. Compared with D-KPDs, on-the-fly addition of nodes can be supported by also taking a large number of nodes to be deployed in the future into account prior to the initial node deployment. Fortunately, irrespective of the number of nodes considered, the size of keying materials necessary to be stored in each node is the same. Accordingly, by considering the nodes to be deployed in the future in the construction of $W^{(t)}$ and $G^{(t)}$, our proposed CARPY and CARPY+ schemes are resilient to dynamic node deployment

3.6.1 Other Advantages

Another unique feature possessed by CARPY+ is that it is transparent to the other network services, that is, CARPY+ can work well in cooperation with the other network services such as routing and power saving mechanisms. For example, as sensor nodes are usually powered by batteries, one method for prolonging the network lifetime is to turn off the radio function of sensor nodes. If the other existing key establishment schemes are used, communications are needed due to either shared-key discovery or path-key establishment. However, for the efficiency of key establishment, the nodes should be always in the active mode to deliver the packets with minimal latency, resulting in the faster energy depletion of sensor nodes. On the other hand, for the efficiency of power consumption, the nodes occasionally turn off the radio, leading to an unstable route between nodes and, therefore, larger latency in establishing keys. Fortunately, since the proposed CARPY+ scheme does not require communications, it does not incur such a problem.

4. CONCLUSION

Two ConstrAined Random Perturbation based pairwise keY establishment (CARPY and CARPY+) schemes are constructed via a novel constrained random perturbation technique. In terms of the so-called sensor-key criteria, while all the existing schemes only satisfy a few requirements, the proposed CARPY+ scheme meets all the requirements. In particular, CARPY+ is the first non-interactive key establishment scheme with great resilience to a large number of node compromises designed for WSNs. Together with a comprehensive comparison, theoretical and experimental results are provided to validate the performance of the CARPY and CARPY+ schemes.

5. REFERENCES

[1] R. Blom. An optimal class of symmetric key generation systems. In *EUROCRYPT*, 1984.

[2] C. Blundo, A. D. Santis, A. Herzberg, S. Kutten, U. Vaccaro, and M. Yung. Perfectly-secure key distribution for dynamic conferences. In *CRYPTO*, 1993.

[3] H. Chan and A. Perrig. PIKE: Peer Intermediaries for Key Establishment in Sensor Networks. In *IEEE INFOCOM*, 2005.

[4] H. Chan, A. Perrig, and D. Song. Random Key Predistribution Schemes for sensor networks. In *IEEE Symposium on Security and Privacy*, 2003.

[5] S. A. Çamtepe and B. Yener. Combinatorial Design of Key Distribution Mechanisms for Wireless Sensor Networks. In *IEEE/ACM Trans. on Networking*, 2007.

[6] S. A. Çamtepe, B. Yener, and M. Yung. Expander Graph based Key Distribution Mechanisms in Wireless Sensor Networks, *IEEE ICC*, 2006.

[7] 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 *IEEE INFOCOM*, 2004.

[8] W. Du, J. Deng, Y. S. Han, and P. Varshney. A Pairwise Key Pre-Distriubtion Scheme for Wireless Sensor Networks. In *ACM CCS*, 2003.

[9] J. Deng, R. Han, S. Mishra. Defending against Path-based DoS Attacks in Wireless Sensor Networks. In *ACM SASN*, 2005.

[10] L. Eschenauer and V. Gligor. A Key-management Scheme for Distributed sensor networks. In *ACM CCS*, 2002.

[11] M. R. Garey and D. S. Johnson. *Computers and Intractability, A Guide to the Theory of NP-Completeness*. W. H. Freeman, San Francisco, CA, 1979.

[12] D. Huang, M. Mehta, D. Medhi, and L. Harn. Location-aware key management scheme for wireless sensor networks. In *ACM SASN*, 2004.

[13] C. Karlof and D. Wagner. Secure routing in wireless sensor networks: attacks and countermeasures. In *IEEE International Workshop on Sensor Network Protocols and Applications*, 2003.

[14] D. Liu, P. Ning. Establishing Pairwise Keys in Distributed Sensor Networks. In *ACM CCS*, 2003.

[15] J. Matoušek and B. Gärtner, *Understanding and Using Linear Programming*. Springer Verlag, 2006.

[16] G. Meulenaer, F. Gosset, F.-X. Standaert, and L. Vandendorpe. On the Energy Cost of Communication and Cryptography in Wireless Sensor Networks. In *Twenty-ninth Symposium on Information Theory in the Benelux*, 2008.

[17] F. J. MacWilliams and N. J. A. Sloane. *The Theory of Error-Correcting Codes*. Elsevier Science Publishing Company, Inc, 1977.

[18] J. Newsome, E. Shi, D. Song, and A. Perrig. The Sybil Attack in Sensor Networks: Analysis and Defenses. *IPSN*, 2004.

[19] R. D. Pietro, L. V. Mancini, and A. Mei. Efficient and Resilient Key Discovery Based on Pseudo-Random Key Pre-Deployment. In *IEEE IPDPS Workshop*, 2004.

[20] A. Perrig, R. Szewczyk, V. Wen, D. Culler, and D. Tyger. SPINS: Security Protocols for Sensor Networks. In *ACM MobiCom*, 2001.

[21] R. C. Merkle. Secure Communications over Insecure Channels. In *Communication of ACM*, 1978.

[22] K. Ren, W. Lou, and Y. Zhang. LEDS: Providing Location-aware End-to-end Data Security in Wireless Sensor Networks. In *IEEE INFOCOM*, 2006.

[23] P. Raghavan and C. D. Tompson. Randomized rounding: A technique for provably good algorithms and algorithmic proofs. In *Combinatorica*, 1987.

[24] V. V. Vazirani. *Approximation Algorithms*. Springer Verlag, 2002.

[25] A. D. Wood and J. A. Stankovic. Denial of Service in Sensor Networks. In *IEEE Computer*, 2002.

[26] F. Ye, H. Luo, S. Lu, and L. Zhang. Statistical En-Route Filtering of Injected False Data in Sensor Networks. In *IEEE INFOCOM*, 2004.

[27] J. Zhao and R. Govindan. Understanding Packet Delivery Performance in Dense Wireless Sensor Networks. In *ACM SenSys*, 2003.

[28] S. Zhu, S.Setia, and S.Jajodia. LEAP: Efficient Security Mechanisms for Large-Scale Distributed Sensor Networks. In *ACM CCS*, 2003.

[29] W. Zhang, N. Subramanian, and G. Wang. Lightweight and Compromise-Resilient Message Authentication in Sensor Networks. In *IEEE INFOCOM*, 2008.

[30] W. Zhang, M. Tran, S. Zhu, and G. Cao. A Random Perturbation-Based Scheme for Pairwise Key Establishment in Sensor Networks. In *ACM MobiHoc*, 2007.