# A Simple Non-Interactive Pairwise Key Establishment Scheme in Sensor Networks

Chia-Mu Yu[†][§], Chun-Shien Lu[†][*], and Sy-Yen Kuo[§]
[†]Institute of Information Science, Academia Sinica, Taipei, Taiwan 115, ROC
[§]Department of Electrical Engineering, National Taiwan University, Taipei, Taiwan 106, ROC

*Abstract*—**In this paper, a *ConstrAined Random Perturbation based pairwise keY establishment* (CARPY) scheme and its variant, a CARPY+ scheme, for Wireless Sensor Networks (WSNs), are presented. Compared to all existing schemes which satisfy only some requirements in so-called *sensor-key criteria*, including 1) resilience to the adversary's intervention, 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. Our schemes have also been practically implemented on the TelosB compatible mote to evaluate the corresponding performance and overhead.**

## I. 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 scarce resources available to each sensor node, the efficient design of key establishment becomes a great challenge.

In the literature, there were two classical threshold-based key distribution (TKD) protocols [1], [2] proposed. 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 a trade-off to 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 [9], 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. Liu and Ning [12], and Du *et al*. [7] proposed to treat the keys in the key pool $S$ as bivariate polynomials and matrices, respectively, to provide pairwise keys between each pair of nodes. There are some other 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 [17] 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] and combinatorial design scheme [5] are proposed. D-KPD schemes can guarantee that there exists at least one key path between two arbitrary nodes.

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 [25] to conquer 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 [6], [10]. In addition, based on the assumption that there is a short bootstrapping time secure after sensor deployment, LEAP [24] is proposed to establish the pairwise keys between each pair of neighboring nodes.

With the fact that the communication channels in WSNs are highly noisy [23] and that over $95\%$ of energy consumption comes from communication [16] 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.

**Evaluation Metrics.** To evaluate the key establishment schemes, five requirements were recently presented in [25]. Nevertheless, we find that they are too weak to be utilized,

[*]Corresponding author: Dr. C. S. Lu (lcs@iis.sinica.edu.tw).

TABLE I
COMPARISONS BETWEEN DIFFERENT KEY ESTABLISHMENT SCHEMES (IN TERMS OF THE SENSOR-KEY CRITERIA)

| | RAI | DGKE | RNC | EFF | RDND |
|---|---|---|---|---|---|
| TKD [1], [2] | | ✓ | ✓ | ✓ | ✓ |
| P-KPD [4], [7], [9], [12] | | | | | ✓ |
| D-KPD [3], [5] | | | | | |
| RPB Scheme [25] | | ✓ | | | ✓ |
| L-KPD [6], [10] | | | | | ✓ |
| LEAP [24] | | | | | ✓ |
| CARPY scheme | | ✓ | ✓ | | ✓ |
| CARPY+ scheme | ✓ | ✓ | ✓ | ✓ | ✓ |

as the security and performance of certain key establishment schemes have been overestimated. Hence, a so-called *sensor-key criteria* composed of five new requirements is proposed as follows to thoroughly evaluate the key establishment schemes applied in the real world.

1) *Resilience to the Adversary's Intervention* (RAI) – Instead of compromising the security of a key establishment scheme, the adversary may only want to hinder the nodes from establishing keys. Hence, in addition to the resilience to node capture considered in the literature, it is also necessary to consider the robustness against various attacks mounted by the adversary (described in Sec. III-A).

2) *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.

3) *Resilience to Network Configurations* (RNC) – Key establishment schemes are necessary to be applicable whatever network configuration is applied.

4) *Efficiency* (EFF) – Key establishment schemes are required to be performed efficiently in terms of storage, computation, and communication overhead.

5) *Resilience to Dynamic Node Deployment* (RDND) – A desirable key establishment scheme should be applicable under the consideration of on-the-fly addition of new sensor nodes.

**Contributions.** Our major contributions are:

1) Two constrained random perturbation based pairwise key establishment schemes, CARPY and CARPY+, are presented. 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 (Table. I). In particular, CARPY+ is the first non-interactive key establishment scheme with great resilience to a large number of node compromises designed for WSNs, and thus act as the building block of other security mechanisms.

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

## II. THE PROPOSED METHOD

Since the proposed schemes, CARPY and CARPY+, remarkably generalize and improve Blom's concept [1], after describing the system model in Sec. II-A, we will briefly review Blom's scheme in Sec. II-B to facilitate the descriptions of our methods. Afterwards, CARPY and CARPY+ are described in detail in Sec. II-C and Sec. II-D, respectively. Finally, the methods for constructing constrained random perturbation will be described in Sec. II-E.

### A. 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 [24] 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 this 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. III-A.

### B. 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 $G_{i,j}$, the $i$-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. 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 $\lambda$-secure [1], which ensures that as long as no more than $\lambda$ nodes are compromised, the security can be perfectly preserved. 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. 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.

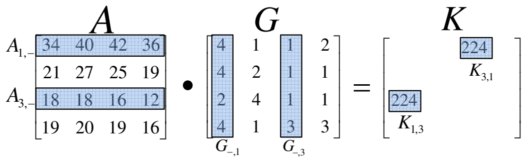

Fig. 1. An illustration of the Blom's scheme.

## C. 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 $q > s_N$, where $q$ is a parameter of a finite field $\mathbb{F}_q$, and $\lambda$ is an appropriate security parameter independent of $N$, which leverages the security level and storage overhead.

*1) Basic Idea of CARPY:* In Blom's scheme, communications become insecure after more than $\lambda$ 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 [25], to enhance the security is to break the direct relation between $D$ and $A$ by adding certain random noise[1] 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 $\ell$, 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 [25] that incurs unnecessary computation and communication overhead, the way of constructing CRP and the corresponding efficiency gain substantially differentiate our work and [25]. 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. III-B2, is still affordable for the current generation sensor nodes.

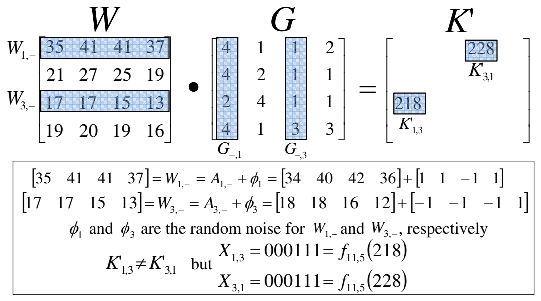
W
G
K'
$W_{1,-}$
$W_{3,-}$
35 41 41 37
21 27 25 19
17 17 15 13
19 20 19 16
4 1 1 2
4 2 1 1
2 4 1 1
4 1 3 3
$G_{-,1}$
$G_{-,3}$
228
$K'_{3,1}$
218
$K'_{1,3}$
$[35\ 41\ 41\ 37] = W_{1,-} = A_{1,-} + \phi_1 = [34\ 40\ 42\ 36] + [1\ 1\ -1\ 1]$
$[17\ 17\ 15\ 13] = W_{3,-} = A_{3,-} + \phi_3 = [18\ 18\ 16\ 12] + [-1\ -1\ -1\ 1]$
$\phi_1$ and $\phi_3$ are the random noise for $W_{1,-}$ and $W_{3,-}$, respectively
$K'_{1,3} \neq K'_{3,1}$ but $X_{1,3} = 000111 = f_{11,5}(218)$
$X_{3,1} = 000111 = f_{11,5}(228)$

Fig. 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, before deployment of sensor nodes, to determine the desired key length, select appropriate parameters, and pre-install the keying materials into the 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) Off-line Step of CARPY:* In addition to the parameters such as the size $q$ of the finite field $\mathbb{F}_q$, the security parameter $\lambda$ 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 $\ell$ 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 $\xi$ $(= \lceil \frac{L}{\ell - r} \rceil)$ 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, where $1 \leq t \leq \xi$. 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 $\varrho$ which has least $r$ bits of its binary representation set to 0. Similarly, let $c_{\max}(\varrho, r)$ be the value of $\varrho$ 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

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

```
Algorithm: CARPY-Off-line-Step($q$, $r$, $\mathcal{I}$, $\lambda$, $L$)
Input: $q$: elements of $D$ and $G$ are selected from $\mathbb{F}_q$
       $r$: the least $r$ bits which will be infected by CRP
       $\mathcal{I}$: the set of sensor nodes identities
       $\lambda$: a security parameter
       $L$: the bit-length of desired key
1  Calculate $\ell$
2  for $t = 1$ to $\xi$ (=$\lceil \frac{L}{\ell - r} \rceil$)
3    Randomly generate $D^{(t)}$ and $G^{(t)}$, and Calculate $A^{(t)}$
4    for $u = 1$ to $|\mathcal{I}|$ $(= N)$
5      Calculate $\Phi^{(t)}_{s_u}$
6      Randomly select a row vector $\phi^{(t)}_{s_u}$ from $\Phi^{(t)}_{s_u}$
7      Calculate $W^{(t)}_{s_u,-} = A^{(t)}_{s_u,-} + \phi^{(t)}_{s_u}$
8      Store $W^{(t)}_{s_u,-}$ and $G^{(t)}_{-,s_u}$ into the sensor node $s_u$
```

Fig. 3. Off-line step of the CARPY scheme.

is performed, each CRP $\phi^{(t)}_{s_u} \in \Phi^{(t)}_{s_u}$ must satisfy the following constraints:

$$(A^{(t)}_{s_u,-} + \phi^{(t)}_{s_u}) \cdot G^{(t)}_{-,s_v} \geq c_{\min}(A^{(t)}_{s_u,-} \cdot G^{(t)}_{-,s_v}, r) \pmod q \tag{1}$$

$$(A^{(t)}_{s_u,-} + \phi^{(t)}_{s_u}) \cdot G^{(t)}_{-,s_v} \leq c_{\max}(A^{(t)}_{s_u,-} \cdot G^{(t)}_{-,s_v}, r) \pmod q \tag{2}$$

$$\phi^{(t)}_{s_u}(k) \in \mathbb{Z}, \tag{3}$$

where $u \neq v$, $1 \leq u, v \leq N$, $1 \leq k \leq (\lambda + 1)$, and $\phi^{(t)}_{s_u}(k)$ is the $k$-th element of $\phi^{(t)}_{s_u}$. Note that a CRP $\phi^{(t)}_{s_u}$ is a $(\lambda + 1)$-dimensional row vector. Eqs. (1) and (2) ensure that after CRP is added to $A^{(t)}_{s_u,-}$ 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. (3) should be satisfied because in CARPY the elements of CRPs are constrained to be integers. As a whole, every $\phi^{(t)}_{s_u}$ that satisfies Eqs. (1)$\sim$(3) is one of the elements in $\Phi^{(t)}_{s_u}$.

Following the construction of $\Phi^{(t)}_{s_u}$, for every $s_u \in \mathcal{I}$, a CRP $\phi^{(t)}_{s_u}$ is randomly and independently selected from $\Phi^{(t)}_{s_u}$. Then, the matrix $W^{(t)}$ is constructed by calculating $W^{(t)}_{s_u,-} = A^{(t)}_{s_u,-} + \phi^{(t)}_{s_u}$ for $1 \leq t \leq \xi$, $1 \leq u \leq N$. After the matrix $W^{(t)}$ is constructed, the row vector $W^{(t)}_{s_u,-}$ together with the column vector $G^{(t)}_{-,s_u}$ are stored into the node $s_u$.

*3) On-line Step of CARPY:* 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 $G^{(t)}$, $G^{(t)}_{-,s_{\hat{u}}}$ and $G^{(t)}_{-,s_{\hat{v}}}$. Then, $s_{\hat{u}}$ and $s_{\hat{v}}$ calculate $\kappa^{(t)}_{s_{\hat{u}},s_{\hat{v}}} = W^{(t)}_{s_{\hat{u}},-} \cdot G^{(t)}_{-,s_{\hat{v}}}$ and $\kappa^{(t)}_{s_{\hat{v}},s_{\hat{u}}} = W^{(t)}_{s_{\hat{v}},-} \cdot G^{(t)}_{-,s_{\hat{u}}}$, respectively. We can see from Eqs. (1)$\sim$(3) 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 $X^{(t)}_{s_{\hat{u}},s_{\hat{v}}} = f_{\ell,r}(\kappa^{(t)}_{s_{\hat{u}},s_{\hat{v}}}) = f_{\ell,r}(\kappa^{(t)}_{s_{\hat{v}},s_{\hat{u}}})$, where $f_{\ell,r}(x)$ is the most significant $\ell - r$ bits of $\ell$-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 $X^{(1)}_{s_{\hat{u}},s_{\hat{v}}} || X^{(2)}_{s_{\hat{u}},s_{\hat{v}}} || \cdots || X^{(\xi)}_{s_{\hat{u}},s_{\hat{v}}}$. The algorithm for the on-line step of CARPY is depicted in Fig. 4.

```
Algorithm: CARPY-On-line-Step
Scenario: nodes $s_{\hat{u}}$ and $s_{\hat{v}}$ want to agree a pairwise key
Note: this algorithm is executed by the sensor node $s_{\hat{v}}$
1  for $t = 1$ to $\xi$ $(= \lceil \frac{L}{\ell - r} \rceil)$
2    Send $G^{(t)}_{-,s_{\hat{u}}}$ to the sensor node $s_{\hat{v}}$
3    Receive $G^{(t)}_{-,s_{\hat{v}}}$ from the sensor node $s_{\hat{v}}$
4    Calculate $\kappa^{(t)}_{s_{\hat{u}},s_{\hat{v}}} = W^{(t)}_{s_{\hat{u}},-} \cdot G^{(t)}_{-,s_{\hat{v}}}$
5    Calculate $X^{(t)}_{s_{\hat{u}},s_{\hat{v}}} = f_{\ell,r}(\kappa^{(t)}_{s_{\hat{u}},s_{\hat{v}}})$
6  Calculate $X_{s_{\hat{u}},s_{\hat{v}}} = X^{(1)}_{s_{\hat{u}},s_{\hat{v}}} || X^{(2)}_{s_{\hat{u}},s_{\hat{v}}} || \cdots || X^{(\xi)}_{s_{\hat{u}},s_{\hat{v}}}$
```

Fig. 4. On-line step of the CARPY scheme.

## D. 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 itself, 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 $\varphi$ is the primitive element of $\mathbb{F}_q$, then any $\lambda + 1$ columns of Vandermonde matrix, which is generated by only one element $\varphi$, are linearly independent [14]. 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 $\lambda$ 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.

## E. Constructing Constrained Random Perturbation

In this section, we deal with the problem of calculating the set $\Phi^{(t)}_{s_u}$ of CRPs for $1 \leq t \leq \xi$ and $1 \leq u \leq N$. Note that the calculation of CRPs is performed only by the network planner but not sensor nodes. A straightforward method for obtaining $\Phi^{(t)}_{s_u}$ is to adopt an exhaustive search. Specifically, given a finite field $\mathbb{F}_q$, all the $q^{\lambda+1}$ possible $(\lambda + 1)$-dimensional vectors are examined in terms of Eqs. (1)$\sim$(3). An exhaustive search can be accomplished in $O(\xi \cdot q^{\lambda+1} \cdot N)$ computational complexity, which is time-consuming. Thus, we present an algorithm, called TwiLP, for constructing $\phi^{(t)}_{s_u}$, which is less time-consuming than the exhaustive search in most of the cases of CARPY and CARPY+. Our approach takes advantage of

```
Algorithm: CARPY+-Off-line-Step($q$, $r$, $\mathcal{I}$, $\lambda$, $L$)
Input: $q$: elements of $D$ and $G$ are selected from $\mathbb{F}_q$
       $r$: the least $r$ bits which will be infected by CRP
       $\mathcal{I}$: the set of sensor node identities
       $\lambda$: a security parameter
       $L$: the bit-length of desired key
1  Calculate $\ell$
2  for $t = 1$ to $\xi$ (=$\lceil \frac{L}{\ell - r} \rceil$)
3     Select $\varphi$
4     Randomly generate $D^{(t)}$, and calculate $A^{(t)}$
5     for $u = 1$ to $|\mathcal{I}|$ $(= N)$
6        Calculate $\Phi_{s_u}^{(t)}$
7        Randomly select a row vector $\phi_{s_u}^{(t)}$ from $\Phi_{s_u}^{(t)}$
8        Calculate $W_{s_u,-}^{(t)} = A_{s_u,-}^{(t)} + \phi_{s_u}^{(t)}$
9        Store $W_{s_u,-}^{(t)}$ and $\varphi$ into the sensor node $s_u$
```

Fig. 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 a pairwise key
Note: this algorithm is executed by the sensor node $s_{\hat{u}}$
1  for $t = 1$ to $\xi$ $(= \lceil \frac{L}{\ell - r} \rceil)$
2     Calculate $G_{-,s_{\hat{v}}}^{(t)}$
3     Calculate $\kappa_{s_{\hat{u}},s_{\hat{v}}}^{(t)} = W_{s_{\hat{u}},-}^{(t)} \cdot G_{-,s_{\hat{v}}}^{(t)}$
4     Calculate $X_{s_{\hat{u}},s_{\hat{v}}}^{(t)} = f_{\ell,r}(\kappa_{s_{\hat{u}},s_{\hat{v}}}^{(t)})$
5  Calculate $X_{s_{\hat{u}},s_{\hat{v}}} = X_{s_{\hat{u}},s_{\hat{v}}}^{(1)} || X_{s_{\hat{u}},s_{\hat{v}}}^{(2)} || \cdots || X_{s_{\hat{u}},s_{\hat{v}}}^{(\xi)}$
```

Fig. 6. On-line step of the CARPY+ scheme.

the efficiency of the linear programming (LP), and we refer readers to [13] for the terminology of linear programming.

**TwiLP Algorithm.** Recall that our objective is to find more than two CRPs, $\phi_{s_u}^{(t)}$, satisfying Eqs. (1)∼(3), for $1 \leq t \leq \xi$ and $1 \leq u \leq N$. 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. (1)∼(3), which constitute the so-called *CRP criteria*, can provide all the possible CRPs, in fact, the consideration of a restricted version of Eqs. (1)∼(3), called *weak CRP criteria*, is sufficient for our use in most cases of CARPY and CARPY+. The weak CRP criteria are given as:

$$-\phi_{s_u}^{(t)} \cdot G_{-,s_v}^{(t)} \leq \alpha_{u,v,r} \tag{4}$$

$$\phi_{s_u}^{(t)} \cdot G_{-,s_v}^{(t)} \leq \beta_{u,v,r} \tag{5}$$

$$-A_{s_u,k}^{(t)} \leq \phi_{s_u}^{(t)}(k) \leq q - 1 - A_{s_u,k}^{(t)} \tag{6}$$

$$\phi_{s_u}^{(t)}(k) \in \mathbb{Z}, \tag{7}$$

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

$$\alpha_{u,v,r} = (A_{s_u,-}^{(t)} \cdot G_{-,s_v}^{(t)}) - c_{\min}(A_{s_u,-}^{(t)} \cdot G_{-,s_v}^{(t)}, r), \tag{8}$$

and

$$\beta_{u,v,r} = c_{\max}(A_{s_u,-}^{(t)} \cdot G_{-,s_v}^{(t)}, r) - (A_{s_u,-}^{(t)} \cdot G_{-,s_v}^{(t)}). \tag{9}$$

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. From an integer linear programming (ILP) point of view, some CRPs can be thought of as the set of all the feasible solutions in the feasible region formed by the linear constraints of Eqs. (4)∼(7). 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 $\Phi_{s_u}^{(t)}$.

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 [20] 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 [19] on the point resulted from averaging those two optimum solutions to search for a feasible solution of ILP. A more detailed description is as follows. Note that the usage of averaging is to enhance the possibility that the result is still within the feasible region.

**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. (4)∼(7) can be easily satisfied. Specifically, $G^{(t)}$ can be constructed as described in Sec. II-D, followed by a division using a large integer $\gamma$ (*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. (4)∼(7) 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 less than 10 seconds are needed to generate a CRP.

## III. Security and Performance Evaluation

### A. 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*. Note that the intention of the adversary's intervention could be the compromise of the security of the key establishment scheme, or the impediment of key establishment between two nodes. The resilience of CARPY and CARPY+ to these four possible attacks is described in Sec. III-A1$\sim$Sec. III-A4, 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 incurs DoS attack. The immunity of CARPY and CARPY+ to DoS attack will be described in Sec. III-A5.

*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.

*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 the adversary. Therefore, we only consider the case where the number of captured nodes is larger than $\lambda + 1$.

Recall that the security of both CARPY and CARPY+ is completely broken if $D^{(t)}$ is obtained by the adversary. We will study the breaking complexity of compromising the matrix $D^{(t)}$ as follows. After CRPs have been applied on $A^{(t)}$ 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 *computational breaking complexity* (CBC), for evaluating the computational difficulty of recovering $D^{(t)}$ is defined. While *physical breaking complexity* (PBC), meaning the least number of nodes necessary to be captured to compromise the security, acts as a metric for evaluating the hardness of recovering $D^{(t)}$ in terms of physical attack, the CBC is defined in terms of computational effort the adversary needs to pay. A theorem describing the breaking complexity of compromising the matrix $D^{(t)}$ is as follows.

**Theorem 1.** *Let $\Upsilon$ be the set of captured nodes. Let $\Delta_i$ be the $i$-th subgroup of captured nodes, on which the same CRP is applied, correctly identified by the adversary. In the case of $|\Upsilon| \geq \lambda+1$, the CBC for recovering the matrices $D^{(t)}$, where $1 \leq t \leq \xi$, is $\Omega(\sum_{j=1}^{\xi}\prod_{i=1}^{\zeta}\prod_{\hat{b}\in\Delta_i}|\Phi_{\hat{b}}^{(j)}|)$ if $\sum_{\eta=1}^{\zeta}(|\Delta_\eta| - 1) \geq \lambda+1$, and $\Omega(\sum_{j=1}^{\xi}\prod_{\hat{b}\in\Upsilon}|\Phi_{\hat{b}}^{(j)}|)$ otherwise, for both the CARPY and CARPY+ schemes.*

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

TABLE II
RELATION BETWEEN VARIOUS PARAMETERS

| $L$ | $q$ | $\ell$ | $\lambda$ | $r$ | $\lvert\Phi_{s_u}^{(t)}\rvert$ | $\xi$ | BC |
|---|---|---|---|---|---|---|---|
| 128 | $2^{16}-15$ | 16 | 32 | 14 | $\geq 2$ | 64 | $\geq 2^{36}$ |
| 128 | $2^{16}-15$ | 16 | 64 | 14 | $\geq 2$ | 64 | $\geq 2^{70}$ |
| 128 | $2^{16}-15$ | 16 | 128 | 8 | $\geq 2$ | 16 | $\geq 2^{133}$ |
| 128 | $2^{16}-15$ | 16 | 128 | 14 | $\geq 2$ | 64 | $\geq 2^{135}$ |

The intuition behind Theorem 1 and Corollary 1 is that, after capturing a set $\Upsilon \subset \mathcal{I}$ 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 in one attempt is with probability $O(1/\prod_{b\in\Upsilon}|\Phi_b^{(t)}|)$ and there are $\xi$ 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 $\Omega(\sum_{t=1}^{\xi}\prod_{b\in\Upsilon}|\Phi_b^{(t)}|) = \Omega(\xi \cdot \rho^{\lambda+1})$. The security levels under different settings can be found in Table II. Consider for example that the parameters, $\ell = 16$, $\lambda = 128$, $r = 14$, and $\rho = 2$ are used. Even with the unrealistic assumptions that I/O operations can be ignored and each trial can be accomplished within single one cycle, the time needed for breaking $D^{(t)}$'s is over $10^{23}$ years on a 3GHz processor.

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. II-E. 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 $\omega$, 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) 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 (such as Sybil [15] and sink hole [11] attacks) could be used to either hinder the legitimate nodes from achieving 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.

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. Nevertheless, when CARPY+ 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.

*4) Physical Layer Attack:* Physical layer attack usually means jamming attack [21], 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. However, the CARPY+ scheme is indeed robust to the physical layer attacks since the pairwise key can be calculated without the need of communication.

*5) Denial of Service (DoS) Attack:* In this paper, we only emphasize the DoS attack incurred by applying the key establishment schemes. As for the Path-based DoS (PDoS) attack [8], the bogus message, aiming at performing shared-key discovery or path-key establishment, can always be sent from the adversary to a victim node to exhaust the energies of the victim node and the nodes on the path to the victim nodes. However, because no communication or interaction between nodes is required in CARPY+ for establishing the pairwise key, DoS attacks can be resisted.

## *B. Performance Analysis*

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, instead of TinyOS. In our experiments, the parameters were set as follows. The desired key length $L$ was 128, $q = 2^{16} - 15$, $\lambda = 128$, and $r = 8$. 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.

*1) Storage Overhead:* If CARPY is used, then, for sensor node $s_u$, the row vectors $A^{(t)}_{s_u,-}$ and column vectors $G^{(t)}_{-,s_u}$ are needed to be stored. Since $\xi$ rounds of CARPY need to be performed independently, when $G^{(t)}$, $1 \leq t \leq \xi$, are selected to be the same, the storage overhead is therefore $O(\xi \cdot \lambda)$. If CARPY+ is used, for sensor node $s_u$, only row vectors $A^{(t)}_{s_u,-}$ and an element $s$ need to be stored. Since the CARPY+ scheme also needs to be performed $\xi$ rounds, the storage overhead for CARPY+ is, thus, $O(\xi \cdot \lambda)$. In our experiments, the storage overhead of CARPY is 288 bytes for CODE memory and 8912 bytes for DATA memory, while that of CARPY+ is 690 bytes for CODE memory and 8906 bytes for DATA memory.

*2) Computation Overhead:* For different $s_u$ and $s_v$, $\lambda + 1$ multiplications and $\lambda$ 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 slightly 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 $\varphi^{s_v}$ (see Sec. II-D), $\lambda + 1$ multiplications and $\lambda$ 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, *i.e.*, number of cycles required for the key generation, in CARPY is 1138548, which is approximately equivalent to 0.14 seconds. In addition, the number of cycles required for the key generation of CARPY+ is 1209756, which is about 0.15 seconds.

*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 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.

*4) Energy Consumption:* In this paper, we utilize a model similar with the one considered in [18] and [22] 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 $h$. 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 the CARPY and CARPY+ schemes. Then, a comparison among them will be presented.

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

$$2 \cdot h \cdot \xi \cdot \lceil \frac{\ell \cdot (\lambda + 1)}{8 \cdot L_{packet}} \rceil \cdot (\frac{e_t}{1 - p_{loss}} + e_r), \qquad (10)$$

because the respective column vectors of $G$ of two nodes $s_u$ and $s_v$ need to be exchanged. As for $E^{comp}_{CARPY}$, it can be computed as $2 \cdot \xi \cdot ((\lambda + 1) \cdot e_m + \lambda \cdot e_a)$, where $e_a$ and $e_m$, respectively, denote the energy consumption of accomplishing the addition and multiplication of two integers, 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, when Horner's rule is exploited, the energy consumption $E^{comp}_{CARPY+}$ of the CARPY+ scheme for the computation is at most $\xi((2(\lambda+1) + \log(s_N + 1))e_m + 2\lambda e_a)$.

**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 $2.1658\mu$J and the energy for transmitting one byte needs $2.0045\mu$J. In our experiments, $e_a$ and $e_m$ are about $0.2164\mu$J and $0.2405\mu$J, respectively. 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 [9], the parameters are selected to achieve $\geq 0.999$ network connectivity. For D-KPD, the 2-D PIKE [3] is adopted. The parameter setting of RPB is done according to [25]. For CARPY and CARPY+, the parameters, $\ell = 16$, $\xi = 20$, and $\lambda = 75$, which achieve the same level of security with RPB, are used. We compare these schemes in two cases[2]: $L_{packet} = 29$ and $L_{packet} = 102$. Under the setting of $p_{loss} = 30\%$, the comparisons of energy consumption for establishing a key between two nodes among different schemes are shown in Fig. 7.

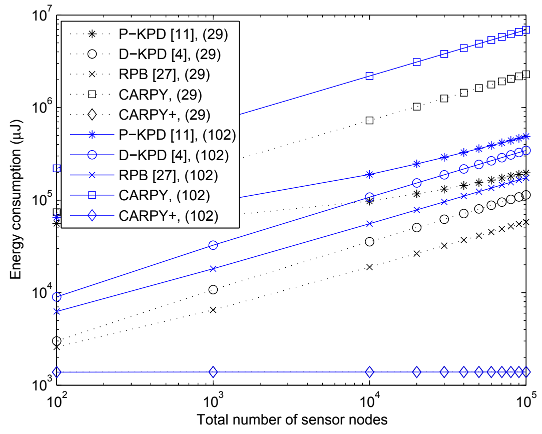

Fig. 7. Energy Consumption for different key establishment schemes (The number in the parenthesis indicates the payload size).

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 $\ell$, $r$, and $\xi$, 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 [9] 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.

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

## *C. Comparisons*

Here, we emphasize the comparisons among CARPY+ and other known key establishment schemes from the sensor-key criteria point of view. The results are shown in Table I and are described in detail in the following.

**Resilience to the Adversary's Intervention (RAI).** Due to the need of path-key establishment, the P-KPDs, D-KPDs, L-KPDs, and LEAP are all vulnerable to the node capture attack. In addition, due to the need of communications, the adversary can always impede the key establishment between two nodes in P-KPDs, D-KPDs, RPB, L-KPDs, and LEAP. 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 RAI.

**Directed and Guaranteed Key Establishment (DGKE).** Due to the storage limitations of each sensor node, predetermined 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. This is because the bootstrapping procedure involves considerable communication and computation overheads and can not be repeated quite often. The RPB scheme is applicable in a wide range of networks. 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, which could be infeasible in certain devices. For CARPY+, it works irrespective of the network scale. In addition, key establishment can be performed independent of deployment knowledge. In fact, CARPY+ can be carried out with arbitrary network topology because the pairwise key is calculated by the node itself. Finally, CARPY+ does not assume the knowledge of hardware; thus, it can be considered to be hardware independent and is applicable in heterogeneous networks.

**Efficiency (EFF).** 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, because of the communication-free property of the CARPY+ scheme, both the energy saving and reduction of latency incurred by the key establishment are 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] 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.

**Other Advantages.** A 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, radio function of sensor nodes usually should be turned off to prolong the network lifetime. However, if the other schemes requiring communications are used for establishing keys, 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 often turn off their 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 need to face such a dilemma.

## IV. 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 for WSNs. Together with a comprehensive comparison, theoretical and experimental results are provided to validate the performance of the CARPY and CARPY+ schemes.

**Acknowledgment**: Chia-Mu Yu and Chun-Shien Lu were supported by NSC 97-2221-E-001-008. Sy-Yen Kuo was supported by NSC 96-2628-E-002-138-MY3.

## References

[1] R. Blom. An optimal class of symmetric key generation systems. In *Proc. International Conference on the Theory and Applications of Cryptographic Techniques (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 *Proc. International Cryptology Conference (CRYPTO)*, 1993.

[3] H. Chan and A. Perrig. PIKE: Peer Intermediaries for Key Establishment in Sensor Networks. In *Proc. IEEE Conference on Computer Communications (INFOCOM)*, 2005.

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

[5] S. A. Çamtepe and B. Yener. Combinatorial Design of Key Distribution Mechanisms for Wireless Sensor Networks. In *IEEE/ACM Trans. on Networking*, vol. 15, no. 2, pp. 346-358, 2007.

[6] 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 *Proc. IEEE Conference on Computer Communications (INFOCOM)*, 2004.

[7] W. Du, J. Deng, Y. S. Han, and P. Varshney. A Pairwise Key Pre-Distriubtion Scheme for Wireless Sensor Networks. In *Proc. ACM Computer and Communications Security Conference (CCS)*, 2003.

[8] J. Deng, R. Han, S. Mishra. Defending against Path-based DoS Attacks in Wireless Sensor Networks. In *Proc. ACM Workshop on Security of Ad Hoc and Sensor Networks (SASN)*, 2005.

[9] L. Eschenauer and V. Gligor. A Key-management Scheme for Distributed sensor networks. In *Proc. ACM Computer and Communications Security Conference (CCS)*, 2002.

[10] D. Huang, M. Mehta, D. Medhi, and L. Harn. Location-aware key management scheme for wireless sensor networks. In *Proc. ACM Workshop on Security of Ad Hoc and Sensor Networks (SASN)*, 2004.

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

[12] D. Liu, P. Ning. Establishing Pairwise Keys in Distributed Sensor Networks. In *Proc. ACM Computer and Communications Security Conference (CCS)*, 2003.

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

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

[15] J. Newsome, E. Shi, D. Song, and A. Perrig. The Sybil Attack in Sensor Networks: Analysis and Defenses. *Proc. ACM/IEEE International Conference on Information Processing in Sensor Networks (IPSN)*, 2004.

[16] A. Perrig, R. Szewczyk, V. Wen, D. Culler, and D. Tyger. SPINS: Security Protocols for Sensor Networks. In *Proc. ACM International Conference on Mobile Computing and Networking (MobiCom)*, 2001.

[17] R. C. Merkle. Secure Communications over Insecure Channels. In *Communication of ACM*, vol. 21, no. 4, pp. 294-299, 1978.

[18] K. Ren, W. Lou, and Y. Zhang. LEDS: Providing Location-aware End-to-end Data Security in Wireless Sensor Networks. In *Proc. IEEE Conference on Computer Communications (INFOCOM)*, 2006.

[19] P. Raghavan and C. D. Tompson. Randomized rounding: A technique for provably good algorithms and algorithmic proofs. In *Combinatorica*, vol. 7, no. 4, pp. 365-374, 1987.

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

[21] A. D. Wood and J. A. Stankovic. Denial of Service in Sensor Networks. In *IEEE Computer*, vol. 35, no. 10, pp. 54-62, 2002.

[22] F. Ye, H. Luo, S. Lu, and L. Zhang. Statistical En-Route Filtering of Injected False Data in Sensor Networks. In *Proc. IEEE Conference on Computer Communications (INFOCOM)*, 2004.

[23] J. Zhao and R. Govindan. Understanding Packet Delivery Performance in Dense Wireless Sensor Networks. In *Proc. ACM Conference on Embedded Networked Sensor Systems (SenSys)*, 2003.

[24] S. Zhu, S.Setia, and S.Jajodia. LEAP: Efficient Security Mechanisms for Large-Scale Distributed Sensor Networks. In *Proc. ACM Computer and Communications Security Conference (CCS)*, 2003.

[25] W. Zhang, M. Tran, S. Zhu, and G. Cao. A Random Perturbation-Based Scheme for Pairwise Key Establishment in Sensor Networks. In *Proc. ACM International Symposium on Mobile Ad Hoc Networking and Computing (MobiHoc)*, 2007.