Noninteractive Pairwise Key Establishment for Sensor Networks

Chia-Mu Yu, *Student Member, IEEE*, Chun-Shien Lu, *Member, IEEE*, and Sy-Yen Kuo, *Fellow, IEEE*

***Abstract*—As a security primitive, key establishment plays the most crucial role in the design of the security mechanisms. Unfortunately, the resource limitation of sensor nodes poses a great challenge for designing an efficient and effective 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 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 noninteractive 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.**

***Index Terms*—Key establishment, key management, noninteractive, sensor networks.**

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

Manuscript received January 26, 2010; accepted May 04, 2010. Date of publication May 24, 2010; date of current version August 13, 2010. This work was supported by NSC 97-2221-E-001-008 and NSC 97-2221-E-002-216-MY3. This paper was published in part in Proc. IEEE Communications Society Conference on Sensor, Mesh and Ad Hoc Communications and Networks (SECON)2009, and in ACM International Symposium on Mobile Ad Hoc Networking and Computing (MobiHoc)2008. The associate editor coordinating the review of this manuscript and approving it for publication was Prof. Robert H. Deng.

C.-M. Yu is with the Department of Electrical Engineering, National Taiwan University, Taipei, 10617, Taiwan, and also with the Institute of Information Science, Academia Sinica, Taipei, 11529, Taiwan (e-mail: r91045@csie.ntu.edu.tw).

C.-S. Lu is with Institute of Information Science, Academia Sinica, Taipei, 11529, Taiwan (e-mail: lcs@iis.sinica.edu.tw).

S.-Y. Kuo is with Department of Electrical Engineering, National Taiwan University, Taipei, 10617, Taiwan (e-mail: sykuo@cc.ee.ntu.edu.tw).

Color versions of one or more of the figures in this paper are available online at http://ieeexplore.ieee.org.

Digital Object Identifier 10.1109/TIFS.2010.2050140

In the literature, there were two classical threshold-based key distribution (TKD) protocols [2], [3] proposed. As the security of both protocols is completely broken as long as the number of captured nodes is above a predetermined 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 predistribution* (P-KPD), proposed by Eschenauer and Gligor [11], 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.* [5] 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.* [5] proposed the random-pairwise keys scheme, which stores predefined pairwise keys, instead of random keys, into certain pairs of nodes. Liu and Ning [17], and Du *et al.* [9] 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 [25] must be used to guarantee the minimal secret information leakage. In view of this, several *deterministic key predistribution* (D-KPD) schemes such as PIKE [4], the expander graph-based scheme [7], and the combinatorial design-based scheme [6], [19] 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 the fact that two nodes always rely on communications between them to find their common key. Focusing on reducing such communication overhead, a strategy, called pseudorandom key predeployment (PRK) [23], 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 [40] to avoid this, while maintaining resilient security.

Usually, one assumes that, prior to sensor deployment, the nodes' locations are not known by the network planner. When some special deployment models are considered, prior knowledge about the nodes' locations can be utilized to construct efficient location-aware key predistribution (L-KPD) schemes [8], [14], [18], [33], [34]. Note that most L-KPD schemes can be thought of as a special case of P-KPD or D-KPD, where the deployment knowledge is taken into account. 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) [39] is proposed to establish the pairwise keys between each pair of neighboring sensor nodes.

In fact, many perspectives on the security in WSNs are investigated. For example, Aysal and Barner [1] consider the sensor data cryptography in WSNs from the estimation theory point of view. Exploiting location information, Ren *et al.* [26] proposed an en-route filtering scheme to achieve data security. In this paper, we focus only on the design of key establishment schemes. For the other security issues, please refer to [13], [15], and [29] for a comprehensive overview.

Problem Statement: With the fact that the communication channels in WSNs are highly noisy and prone to transmission errors [37], [38] and that over 95% of energy consumption comes from communication [24] in mind, we can know that although numerous key establishment schemes are proposed, all of them, except the TKD schemes [2], [3] (which unfortunately cannot achieve resilient security), are inefficient and highly energy-consuming due to the involved communications. 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 [40]. Nevertheless, we find that they are too weak to be utilized, 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)—The adversary may launch a wide range of attacks such as eavesdropping, node compromise, message forgery, packet dropping, and noise injection, etc., to compromise the security of a key establishment scheme. On the other hand, 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 Section IV-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)—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.
4) *Efficiency* (EFF)—Key establishment schemes are required to be performed efficiently in terms of storage, computation, and communication overhead. The time consumed for finding the common key is also a nonnegligible metric for evaluating the efficiency of a key establishment scheme. For example, when underwater sensor networks are considered, since the propagation speed of acoustic signals in water is five orders of magnitude lower than the radio propagation speed [31], the key establishment scheme should be carefully designed so that the latency for establishing keys can be minimized.
5) *Resilience to Dynamic Node Deployment* (RDND)—The hardware failure or energy depletion of sensor nodes could lead to a WSN that 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.

A. Contribution

There are two major contributions in this paper:

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). (The detailed comparison will be shown in Section IV-C.) In particular, CARPY+ is the first noninteractive key establishment scheme with great resilience to a large number of node compromises designed for WSNs, and can 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.

B. Organization

At first, the network and security models used in this paper are presented in Section II. Subsequently, the proposed CARPY and CARPY+ schemes will be presented in Section III. Together with a comprehensive comparison with some known schemes, the theoretical and experimental results will be described in Section IV. At last, the conclusion will be made in Section V.

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

	RAI	DGKE	RNC	EFF	RDND
TKD [2], [3]		✓	✓	✓	✓
P-KPD [5], [9], [11], [17]					✓
D-KPD [4], [6], [7], [19]					
RPB Scheme [40]		✓			✓
L-KPD [8], [14], [18], [33], [34]					✓
LEAP [39]					✓
CARPY scheme		✓	✓		✓
CARPY+ scheme	✓	✓	✓	✓	✓

II. 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 [39] 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 Section IV-A.

III. PROPOSED METHOD

Since the proposed schemes, CARPY and CARPY+, remarkably generalize and improve Blom's concept [2], we will briefly review Blom's scheme in Section III-A. Afterwards, CARPY and CARPY+ are described in detail in Sections III-B and C, respectively. Finally, the methods for constructing constrained random perturbation (CRP) will be presented in Section III-D.

A. Review of Blom's Scheme [2]

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 ith row and jth column of G by $G_{i,j}$, ith row of G by $G_{i,-}$ and the jth 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. The only

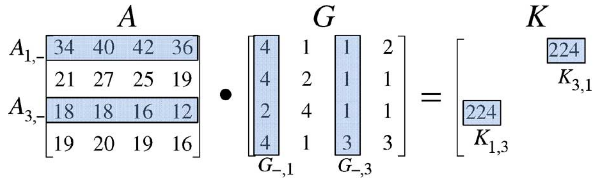

Fig. 1. Illustration of the Blom's scheme.

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 $K = A \cdot G$, where $\cdot$ denotes matrix multiplication and $(D \cdot G)^T$ is the transpose of $(D \cdot G)$. 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 [2] 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 [2], 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.

B. 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 $s_1 < s_2 < \ldots, < s_N$. Here, $s_1, s_2, \ldots, s_N$, instead of $1, 2, \ldots, N$, are used as the notations of IDs, which emphasizes that the IDs can be arbitrarily chosen. We also assume that $q > s_N$, 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. The details of parameter settings will be shown in Section IV-B.

1) Basic Idea of CARPY: 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

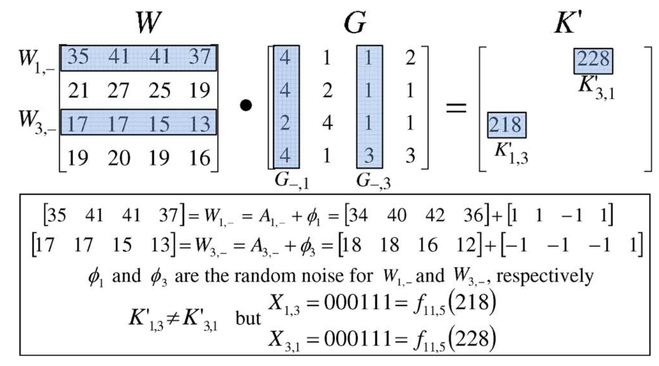
W
G
K'
W1,−
W3,−
35 41 41 37
21 27 25 19
17 17 15 13
19 20 19 16
4 1 1 2
4 2 1 1
2 4 1 1
4 1 3 3
G−,1
G−,3
228
K'3,1
218
K'1,3
[35 41 41 37] = W1,− = A1,− + φ1 = [34 40 42 36] + [1 1 −1 1]
[17 17 15 13] = W3,− = A3,− + φ3 = [18 18 16 12] + [−1 −1 −1 1]
φ1 and φ3 are the random noise for W1,− and W3,−, respectively
K'1,3 ≠ K'3,1 but X1,3 = 000111 = f11,5(218)
X3,1 = 000111 = f11,5(228)

Fig. 2. Illustration of the CARPY scheme.

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 [40], 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 ratio (SNR), 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 [40] that incurs unnecessary computation and communication overhead, the way of constructing CRP and the corresponding efficiency gain substantially differentiate our work and [40]. 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 $\lceil L/(\ell - r)\rceil$ rounds to generate a pairwise key with desired key length. Albeit $\lceil L/(\ell - r)\rceil$ rounds of CARPY are required, the overall computation overhead, which will be analyzed in Section IV-B2, is still affordable for the current generation sensor nodes.

There are two steps in the CARPY scheme, the off-line step and the on-line step. In general, the off-line step is performed, before deployment of sensor nodes, to determine the desired key length, select appropriate parameters, and preinstall 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 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 the 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 $(= \lceil L/(\ell - r)\rceil)$ rounds to obtain a pairwise key with desired key length L.

[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 $\Phi_{s_u}^{(t)}$
6         Randomly select a row vector $\phi_{s_u}^{(t)}$ from $\Phi_{s_u}^{(t)}$
7         Calculate $W_{s_u,-}^{(t)} = A_{s_u,-}^{(t)} + \phi_{s_u}^{(t)}$
8         Store $W_{s_u,-}^{(t)}$ and $G_{-,s_u}^{(t)}$ into the sensor node $s_u$
```

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

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 tth round of the CARPY scheme, where $1 \le t \le \xi$. 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 ϱ which has least r bits of its binary representation set to 0. Similarly, let $c_{\max}(\varrho, r)$ be the value 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 tth round of the CARPY scheme is performed, each CRP $\phi_{s_u}^{(t)} \in \Phi_{s_u}^{(t)}$ must satisfy the following constraints:

$$\left(A_{s_u,-}^{(t)} + \phi_{s_u}^{(t)}\right) \cdot G_{-,s_v}^{(t)} \ge c_{\min}\left(A_{s_u,-}^{(t)} \cdot G_{-,s_v}^{(t)}, r\right) \pmod q \tag{2}$$

$$\left(A_{s_u,-}^{(t)} + \phi_{s_u}^{(t)}\right) \cdot G_{-,s_v}^{(t)} \le c_{\max}\left(A_{s_u,-}^{(t)} \cdot G_{-,s_v}^{(t)}, r\right) \pmod q \tag{3}$$

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

where $u \ne v$, $1 \le u, v \le N$, $1 \le k \le (\lambda + 1)$, and $\phi_{s_u}^{(t)}(k)$ is the kth element of $\phi_{s_u}^{(t)}$. Note that a CRP $\phi_{s_u}^{(t)}$ is a $(\lambda + 1)$-dimensional row vector. Equations (2) and (3) ensure that after

```
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.

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 (4) 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 (2)–(4) 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 tth 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 (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 tth 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 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 $X^{(1)}_{s_{\hat{u}},s_{\hat{v}}} || X^{(2)}_{s_{\hat{u}},s_{\hat{v}}} || \cdots || X^{(\xi)}_{s_{\hat{u}},s_{\hat{v}}}$, where $||$ denotes the bit-string concatenation. The algorithm for the on-line step of CARPY is depicted in Fig. 4.

Example 1: Given that $q = 2^{10} - 3$, $N = 4$, $\lambda = 3$, $\mathcal{I} = \{1, 2, 3, 4\}$, $r = 5$, and $L = 5$. In this example, $\ell = 10$ can be calculated. Since $\ell - r = L$, performing the CARPY scheme (the main idea is shown in Fig. 2) 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 $[-1\ 1\ 0\ 0]$, satisfying (2)–(4). $W_{3,-}$ can be obtained similarly by having $A_{3,-} + \phi_{3,-}$, where $\phi_{3,-} = [-1\ \ -1\ 0\ 1]$, 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.

```
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 $\Phi^{(t)}_{s_u}$
7         Randomly select a row vector $\phi^{(t)}_{s_u}$ from $\Phi^{(t)}_{s_u}$
8         Calculate $W^{(t)}_{s_u,-} = A^{(t)}_{s_u,-} + \phi^{(t)}_{s_u}$
9         Store $W^{(t)}_{s_u,-}$ 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 $G^{(t)}_{-,s_{\hat{v}}}$
3     Calculate $\kappa^{(t)}_{s_{\hat{u}},s_{\hat{v}}} = W^{(t)}_{s_{\hat{u}},-} \cdot G^{(t)}_{-,s_{\hat{v}}}$
4     Calculate $X^{(t)}_{s_{\hat{u}},s_{\hat{v}}} = f_{\ell,r}(\kappa^{(t)}_{s_{\hat{u}},s_{\hat{v}}})$
5 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. 6. On-line step of the CARPY+ scheme.

C. 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 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 [21]. Note that such a Vandermonde matrix is of the form that the ith column is generated by $[1\ \varphi^i\ (\varphi^i)^2\ \cdots\ (\varphi^i)^\lambda]^T$, 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 Figs. 5 and 6, respectively.

D. Constructing CRP

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 $\Phi_{s_u}^{(t)}$ 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 (2)–(4). An exhaustive search can be accomplished in $O(\xi \cdot q^{\lambda+1} \cdot N)$ computational complexity, which is time-consuming. Thus, we present two algorithms for constructing $\phi_{s_u}^{(t)}$, which are less time-consuming than the exhaustive search in most of the cases of CARPY and CARPY+. Our approach takes advantage of the efficiency of the linear programming, and we refer readers to [20] for the knowledge of linear programming and only introduce the terminology required for our use.

A *linear program* (LP) of n variables is composed of a linear objective function of the form, $c_1x_1 + c_2x_2 +, \ldots, +c_nx_n$, 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 an 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 an 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 [12].

TwiLP Algorithm: Recall that our objective is to find more than two CRPs, $\phi_{s_u}^{(t)}$, satisfying (2)–(4), 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 (2)–(4), which constitute the so-called *CRP criteria*, can provide all the possible CRPs, in fact, the consideration of a restricted version of (2)–(4), called *weak CRP criteria*, is sufficient for our use in most cases of CARPY and CARPY+. The weak CRP criteria are derived from (2)–(4) without considering the modular arithmetic as

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

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

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

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

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

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

and

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

An immediate observation is that the solutions satisfying the weak CRP criteria are a subset of the solutions satisfying the CRP criteria. In particular, (5) and (6) are, respectively, the same as (2) and (3) except that the modular arithmetic is abandoned. Equation (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, some CRPs can be thought of as the set of all the feasible solutions in the feasible region formed by the linear constraints of (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 $\Phi_{s_u}^{(t)}$.

```
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 $\bar{\pi}$
9 until the rounding result is a feasible solution of $ILP(c)$
```

Fig. 7. TwiLP Algorithm.

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 [28] is used twice to find the two optimum solutions with respect to $\mathcal{O}_1$ and $\mathcal{O}_2$ in LPs, respectively, and then performs randomized rounding [27] on the point resulted from averaging those two optimum solutions to search for a feasible solution of ILP. TwiLP is shown in Fig. 7 and a more detailed description is as follows.

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)

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

subject to

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

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

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

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

With a simple calculation, one can obtain the optimum solution $\pi_{\mathrm{LP}(c)}$ 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 $\pi_{\mathrm{LP}(d)}$ for the LP-relaxation of ILP(d). Then, the average $\bar{\pi} = (\pi_{\mathrm{LP}(c)} + \pi_{\mathrm{LP}(d)})/\omega$, where ω is a randomly selected integer, is calculated. It should be noted that c, d, and ω should be kept as a secret only known by the network planner so that the adversary cannot follow the same rule to construct the CRPs used in CARPY and CARPY+. Finally, randomized rounding is applied on the vector $\bar{\pi}$. Since the feasible region of an 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. If ω is selected properly, $\bar{\pi}$ can be near the centroid of the feasible region. Hence, it is efficient to obtain the integer feasible solution, which is the CRP, by applying randomized rounding on $\bar{\pi}$ in most cases.

Example 2: Consider the following ILP:

Integer Linear Program (ILP − EX − 1)

$$\begin{aligned} &\text{minimize} && -3x_1 - 2x_2 && (16)\\ &\text{subject to} && && \\ & && 3x_1 + x_2 \le 9 && (17)\\ & && x_1 + 3_2 \le 7 && (18)\\ & && -x_1 + x_2 \le 1 && (19)\\ & && x_1, x_2 \ge 0, \quad x_1, x_2 \in \mathbb{N} && (20)\end{aligned}$$

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. Consider the following ILP whose objective function is in the opposite direction of that of ILP-EX-1:

Integer Linear Program (ILP − EX − 2)

$$\begin{aligned} &\text{minimize} && 3x_1 + 2x_2 && (21)\\ &\text{subject to} && && \\ & && 3x_1 + x_2 \le 9 && (22)\\ & && x_1 + 3_2 \le 7 && (23)\\ & && -x_1 + x_2 \le 1 && (24)\\ & && x_1, x_2 \ge 0, \quad x_1, x_2 \in \mathbb{N} && (25)\end{aligned}$$

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 are all feasible solutions.

Implementation Issues: When there are many constraints in LP, the solver of LP could require a long time to obtain the optimal solution. Furthermore, 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 key observation here is that if the elements of $G^{(t)}$ are relaxed to small floating point numbers instead of integers, then (5)–(8) can be easily satisfied. Specifically, $G^{(t)}$ can be constructed as described in Section III-C, followed by a division using an integer γ (e.g., $\gamma = 100$). 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 (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 less than 10 s are needed to generate a CRP. In the subsequent discussion, we assume that the above simple randomized algorithm is exploited to construct CRPs.

IV. 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 the *eavesdropping attack*, the *node capture attack*, the *routing layer attack*, and the *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 Sections IV-A1–IV-A4, respectively. In addition to the aforementioned attacks, Denial of Service (DoS) is a common strategy mounted by the adversary to attack networks. While the 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 Section IV-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. To break the security of the CARPY and CARPY+ schemes, the effort one should pay is the PBC plus CBC. A theorem describing the breaking complexity of compromising the matrix $D^{(t)}$ is as follows.

Theorem 1: Let Υ be the set of captured nodes. Let Δ_i be the ith subgroup of captured nodes, on which the same CRP is applied, correctly identified by the adversary. In the case of $|\Upsilon| < \lambda + 1$, the CARPY and CARPY+ schemes are information-theoretic perfectly secure. 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.

Proof: Blom's scheme can be regarded as a special case of CARPY and CARPY+, where the same publicly-known CRPs are applied on each row of $A^{(t)}$. Thus, in the case of $|\Upsilon| < \lambda+1$, the security of CARPY and CARPY+ directly inherits from that of Blom's scheme; it is perfectly secure.

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:

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

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

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 (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 (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 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 (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 $\{\Delta_1, \ldots, \Delta_\zeta\}$, 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 $\sum_{\eta=1}^{\zeta}(S_\eta - 1) \geq \lambda+1$ since $D^{(t)}$ is a symmetric matrix. 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 $1/(\prod_{i=1}^{\zeta} \prod_{\hat{b} \in \Delta_i} |\Phi_{\hat{b}}^{(t)}|)$. 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 $1/(\sum_{j=1}^{\xi} \prod_{i=1}^{\zeta} \prod_{\hat{b} \in \Delta_i} |\Phi_{\hat{b}}^{(j)}|)$. As a consequence, the CBC turns to be

$$\Omega\left(\sum_{j=1}^{\xi} \prod_{i=1}^{\zeta} \prod_{\hat{b} \in \Delta_i} \left|\Phi_{\hat{b}}^{(j)}\right|\right) \tag{28}$$

which completes the proof. □

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.

Proof: Simply substituting $|\Phi_{s_u}^{(t)}| \geq \rho$ into the result derived from Theorem 1, we have the probability P of correctly identifying the ζ ill-perturbed groups for ξ matrices as follows:

$$P = 1/\sum_{j=1}^{\xi} \prod_{i=1}^{\zeta} \prod_{\hat{b} \in \Delta_i} \left|\Phi_{\hat{b}}^{(j)}\right| \leq 1/(\xi \cdot \rho^{\lambda+1}). \tag{29}$$

Therefore, the CBC can be written as $1/P = \Omega(\xi \cdot \rho^{\lambda+1})$.

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 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 $\Omega(\sum_{t=1}^{\xi} \prod_{b \in \Upsilon} |\Phi_b^{(t)}|) = \Omega(\xi \cdot \rho^{\lambda+1})$. 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 input/output (I/O) operations can be ignored and each trial can be accomplished within one single cycle, the time needed for breaking $D^{(t)}$'s is over 10^{23} years on a 3-GHz 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 Section III-D. 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, the TwiLP algorithm can be utilized by the adversary. However, the parameters c, d, and ω, used in TwiLP are only known to the network planner and

TABLE II
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	8	≥ 2	16	$\geq 2^{133}$
128	$2^{16} - 15$	16	128	14	≥ 2	64	$\geq 2^{135}$

unknown to 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 [22] and sink hole [16] 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: A physical layer attack usually means a jamming attack [30], in which the adversary disrupts the capability of transmitting and receiving packets for some specified nodes through radio-frequency interference. Solely exploiting the 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) 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 [10], 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 (MicroController: TI MSP430F1611; Flash Memory: 48 KB + 25 6 B; RAM: 10 KB; 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 is 128, $q = 2^{16} - 15$, $\lambda = 128$, $\xi = 16$, 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 ξ rounds of CARPY need to be performed independently, when $G^{(t)}$, $1 \le t \le \xi$ 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 ξ rounds, the storage overhead for CARPY+ is, thus, $O(\xi \cdot \lambda)$. The storage overhead incurred in both the CARPY and CARPY+ schemes in our experiment is shown in Table III.

TABLE III
STORAGE OVERHEAD OF CARPY AND CARPY+ (IN BYTES)

Scheme	CODE Memory	DATA Memory
CARPY	288	8912
CARPY+	690	8906

TABLE IV
COMPUTATION OVERHEAD OF CARPY AND CARPY+

Scheme	Time (in seconds)	Cycle Count
CARPY	0.14	1138548
CARPY+	0.15	1209756

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 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 φ^{s_v} (see Section III-C), $\lambda+1$ 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 IV.

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 [26] and [36] 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 a 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 several known schemes and the CARPY and CARPY+ schemes. Then, a comparison among them will be presented.

Probabilistic Key Predistribution: We show how to calculate the energy consumption E_{prob} for P-KPD, which is composed of the energy consumption $E_{\text{prob}}^{\text{comm}}$ of communications and the energy consumption $E_{\text{prob}}^{\text{comp}}$ of computation, as follows. Basically, the schemes [5], [8], [9], [11], [17], [18], and [33] are all within the same framework of P-KPD. Without loss of generality, we only conduct energy consumption for [11], but the evaluation results can be naturally extended to [5], [8], [9], [17], [18], and [33].

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

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

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 $h \cdot ((e_t/(1-p_{\text{loss}})) + e_r) \cdot (\lceil L_z/L_{\text{packet}}\rceil + 1)$. 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 $h \cdot ((e_t/(1-p_{\text{loss}})) + e_r) \cdot (\lceil L/8 \cdot L_{\text{packet}}\rceil + 1)$ required for transmitting the path-key. Thus, the energy consumption $E_{\text{prob}}^{\text{comm}}$ of communications for P-KPD can be estimated as

$$h \cdot \left(\frac{e_t}{1-p_{\text{loss}}} + e_r\right) \cdot \left(\left\lceil \frac{L_z}{L_{\text{packet}}}\right\rceil + 1\right) + (1-p_c) \cdot h \cdot \left(\frac{e_t}{1-p_{\text{loss}}} + e_r\right)\left(\left\lceil \frac{L}{8 \cdot L_{\text{packet}}}\right\rceil\right). \tag{31}$$

The energy consumption $E_{\text{prob}}^{\text{comp}}$ of computation for P-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 common 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 some hidden costs, i.e., the energy consumption for establishing the shared-key or path-key between two neighboring nodes, are ignored in our calculation.

Deterministic Key Predistribution: Here, we consider the two-dimensional (2-D) PIKE scheme proposed in [4]. Note that the evaluation results can be easily extended to the other deterministic KPD schemes [6], [7], [19]. 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 2-D 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 $E_{\text{deter}} = E_{\text{deter}}^{\text{comm}} + E_{\text{deter}}^{\text{comp}}$ for deterministic KPD can be derived by calculating the energy consumption $E_{\text{deter}}^{\text{comm}}$ of communications as

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

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

RPB Key Establishment: The RPB scheme [40] 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 $E_{\text{RPB}} = E_{\text{RPB}}^{\text{comm}} + E_{\text{RPB}}^{\text{comp}}$. Since executing RPB one time derives a part of the pairwise key, without loss of generality, we assume that ξ rounds of RPB 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 [40], the energy consumption $E_{\text{RPB}}^{\text{comm}}$ of communications for the RPB scheme can be easily estimated as

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

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 $E_{\text{RPB}}^{\text{comp}}$ of computation for RPB scheme can be estimated as $\xi \cdot \lambda \cdot (e_a + e_m) + \xi \cdot e_e + (\xi - 1) \cdot e_{\text{XOR}}$ for s_u, and $\xi \cdot \lambda \cdot (e_a + e_m) + 3 \cdot \xi \cdot e_e + 3^{\xi} \cdot e_{\text{XOR}}$ 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 exclusive OR (XOR) operation between two bit-strings. Here, as in the experiment conducted in [40], 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 $E_{\text{CARPY+}}$ for both CARPY and CARPY+, respectively. E_{CARPY} can be estimated as $E_{\text{CARPY}} = E_{\text{CARPY}}^{\text{comm}} + E_{\text{CARPY}}^{\text{comp}}$. Here, $E_{\text{CARPY}}^{\text{comm}}$ is calculated as

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

because the respective column vectors of G of two nodes s_u and s_v need to be exchanged. As for $E_{\text{CARPY}}^{\text{comp}}$, 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 $E_{\text{CARPY+}}^{\text{comm}}$ 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 $G_{-,s_v}^{(t)}$, requiring $\xi \cdot \log(s_v + 1)$

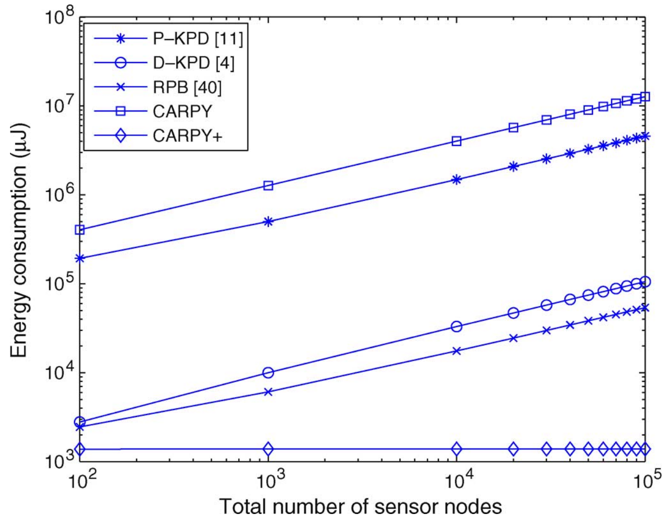

Fig. 8. Energy consumption for different key establishment schemes under the setting of $p_{\text{loss}} = 20\%$ and $L_{\text{packet}} = 29$ B.

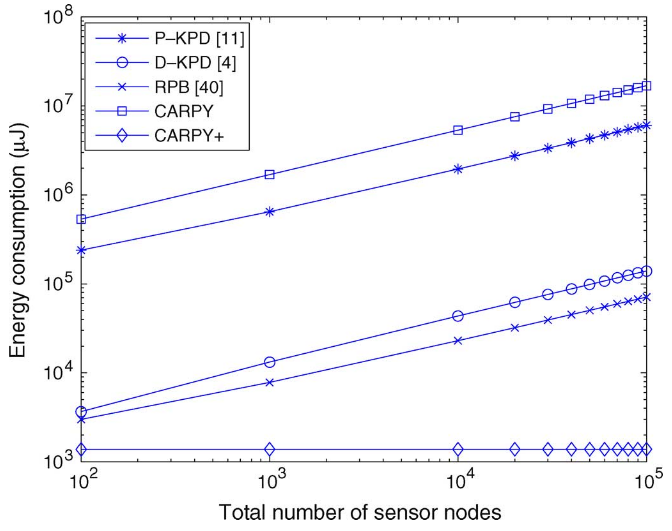

Fig. 10. Energy consumption for different key establishment schemes under the setting of $p_{\text{loss}} = 50\%$ and $L_{\text{packet}} = 29$ B.

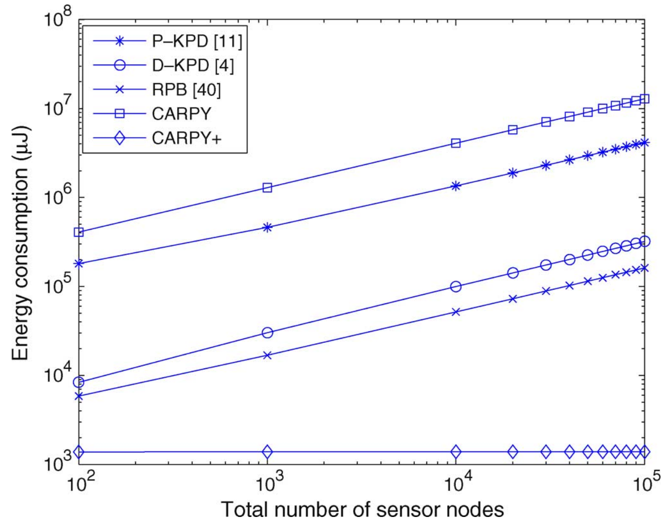

Fig. 9. Energy consumption for different key establishment schemes under the setting of $p_{\text{loss}} = 20\%$ and $L_{\text{packet}} = 102$ B.

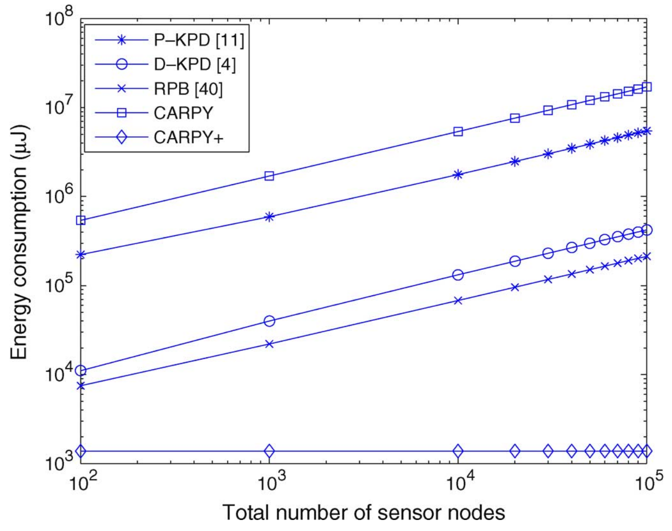

Fig. 11. Energy consumption for different key establishment schemes under the setting of $p_{\text{loss}} = 50\%$ and $L_{\text{packet}} = 102$ B.

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 $E^{\text{comp}}_{\text{CARPY+}}$ 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 3.6 V and 250 kb/s, respectively, then the energy for receiving one byte needs 2.1658 μJ and the energy for transmitting one byte needs 2.0045 μJ. In our experiments, e_a and e_m are about 0.2164 and 0.2405 μJ, 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 [11], the parameters are selected to achieve ≥ 0.999 network connectivity. For D-KPD, the 2-D PIKE [4] is adopted. The parameter setting of RPB is done according to [40]. 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] $L_{\text{packet}} = 29$ and $L_{\text{packet}} = 102$. Under the setting of different packet loss rates, the comparisons of energy consumption for establishing a key between two nodes among different schemes are shown in Figs. 8–11.

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 from Figs. 8–11 that the energy consumption in the proposed CARPY+ scheme is substantially smaller than all the known schemes chosen for comparisons and remains steady in all cases due to its communication-free property. For example,

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

we can know from Fig. 8 that the energy consumption of P-KPD [11] is about 1000 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. The effect of packet size can also be observed from Figs. 8 and 9. The energy consumption of D-KPD [4] is about 23 times greater than that of CARPY+ in a network with $L_{\text{packet}} = 29$ bytes while the energy consumption of D-KPD is about 72 times greater than that of CARPY+ in a network with $L_{\text{packet}} = 102$ bytes. Moreover, the effect of packet loss rate is clearly revealed in Figs. 9 and 11. The energy consumption of D-KPD is about 72 times greater than that of CARPY+ in a network with $p_{\text{loss}} = 20\%$ while the energy consumption of D-KPD is about 95 times greater than that of CARPY+ in a network with $p_{\text{loss}} = 50\%$. Such energy savings can also attribute to the communication-free property of CARPY+. More specifically, the higher the packet loss rate, the larger the transmission errors and the more the retransmissions. We conclude that the communication-free property of CARPY+ is extremely helpful in reducing communication cost.

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.

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. On the other hand, RAI of TKDs is also poor because TKDs involve threshold behavior in security aspect. 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.

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 Figs. 4 and 6 that key sharing can be always established between any two nodes.

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 cannot 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.

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 and can be kept as minimal.

RDND: In D-KPDs, key sharing between nodes usually relies on some fixed structures, such as the hypercube in [4], the expander graph in [7], and the combinatorial design in [6] and [19]. 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 increases 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 power saving and medium access control (MAC) 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. In addition, if the communications are required in establishing keys, MAC protocols could also be overloaded since the increased communications imply higher interference. The proposed CARPY+ scheme can be implemented without this difficulty.

V. 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 noninteractive 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

Preliminary versions of portions of this paper were presented in [32] and [35].

References

[1] T. C. Aysal and K. E. Barner, "Sensor data cryptography in wireless sensor networks," *IEEE Trans. Inf. Forensics Security*, vol. 3, no. 2, pp. 273–289, Jun. 2008.

[2] R. Blom, "An optimal class of symmetric key generation systems," in *Proc. Annu. Int. Conf. Theory and Applications of Cryptographic Techniques (EUROCRYPT)*, Paris, France, 1984, pp. 335–338.

[3] C. Blundo, A. D. Santis, A. Herzberg, S. Kutten, U. Vaccaro, and M. Yung, "Perfectly-secure key distribution for dynamic conferences," in *Proc. 29th Int. Cryptology Conf. (CRYPTO)*, Santa Barbara, CA, 1993, pp. 110–125.

[4] H. Chan and A. Perrig, "PIKE: Peer intermediaries for key establishment in sensor networks," in *Proc. 24th IEEE Conf. Computer Communications (INFOCOM)*, Miami, FL, 2005, pp. 524–535.

[5] H. Chan, A. Perrig, and D. Song, "Random key predistribution schemes for sensor networks," in *Proc. IEEE Symp. Security and Privacy (S&P)*, Oakland, CA, 2003, pp. 197–213.

[6] S. A. Çamtepe and B. Yener, "Combinatorial design of key distribution mechanisms for wireless sensor networks," *IEEE/ACM Trans. Netw.*, vol. 15, no. 2, pp. 346–358, Apr. 2007.

[7] S. A. Çamtepe, B. Yener, and M. Yung, "Expander graph based key distribution mechanisms in wireless sensor networks," in *Proc. IEEE Int. Conf. Communications (ICC)*, Istanbul, Turkey, 2006, pp. 2262–2267.

[8] 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. 24th IEEE Conf. Computer Communications (INFOCOM)*, Hong Kong, China, 2004.

[9] W. Du, J. Deng, Y. S. Han, and P. Varshney, "A pairwise key pre-distriubtion scheme for wireless sensor networks," in *Proc. Annu. ACM Computer and Communications Security (CCS)*, Washington, DC, 2003, pp. 42–51.

[10] J. Deng, R. Han, and S. Mishra, "Defending against path-based DoS attacks in wireless sensor networks," in *Proc. Fourth ACM Workshop on Security of Ad Hoc and Sensor Networks (SASN)*, Alexandria, VA, 2005, pp. 89–96.

[11] L. Eschenauer and V. Gligor, "A key-management scheme for distributed sensor networks," in *Proc. Annual ACM Computer and Communications Security (CCS)*, Washington, DC, 2002, pp. 41–47.

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

[13] V. C. Giruka, M. Singhal, J. Royalty, and S. Varanasi, "Security in wireless sensor networks," *Wireless Commun. Mobile Comput.*, vol. 8, no. 1, pp. 1–24, Jan. 2008.

[14] D. Huang, M. Mehta, D. Medhi, and L. Harn, "Location-aware key management scheme for wireless sensor networks," in *Proc. Fourth ACM Workshop on Security of Ad Hoc and Sensor Networks (SASN)*, Washington, DC, 2004, pp. 29–42.

[15] D. Kundur, W. Luh, U. N. Okorafor, and T. Zourntos, "Security and privacy for distributed multimedia sensor networks," *Proc. IEEE*, vol. 96, no. 1, pp. 112–130, Jan. 2008.

[16] C. Karlof and D. Wagner, "Secure routing in wireless sensor networks: Attacks and countermeasures," in *Proc. IEEE Int. Workshop on Sensor Network Protocols and Applications*, 2003.

[17] D. Liu and P. Ning, "Establishing pairwise keys in distributed sensor networks," in *Proc. Annu. ACM Computer and Communications Security (CCS)*, Washington, DC, 2003, pp. 52–61.

[18] D. Liu, P. Ning, and W. Du, "Group-based key predistribution for wireless sensor networks," *ACM Trans. Sensor Netw.*, vol. 4, no. 2, pp. 11:1–11:30, Mar. 2008.

[19] J. Lee and D. R. Stinson, "On the construction of practical key predistribution schemes for distributed sensor networks using combinatorial designs," *ACM Trans. Inf. Syst. Security*, vol. 11, no. 2, pp. 5:1–5:35, Mar. 2008.

[20] J. Matoušek and B. Gärtner*, Understanding and Using Linear Programming*. New York: Springer-Verlag, 2006.

[21] F. J. MacWilliams and N. J. A. Sloane*, The Theory of Error-Correcting Codes*. Amsterdam, The Netherlands: Elsevier, 1977.

[22] J. Newsome, E. Shi, D. Song, and A. Perrig, "The Sybil attack in sensor networks: Analysis and defenses," in *Proc. Third Int. Symp. Information Processing in Sensor Networks (IPSN)*, Berkeley, CA, Apr. 2004, pp. 259–268.

[23] R. D. Pietro, L. V. Mancini, and A. Mei, "Efficient and resilient key discovery based on pseudo-random key pre-deployment," in *Proc. Int. Parallel and Distributed Processing Symp. (IPDPS)—Workshop 12*, Santa Fe, NM, 2004.

[24] A. Perrig, R. Szewczyk, V. Wen, D. Culler, and D. Tyger, "SPINS: Security protocols for sensor networks," in *Proc. Seventh Annu. Int. Conf. Mobile Computing and Networks (ACM Mobicom)*, Rome, Italy, Jul. 2001.

[25] R. C. Merkle, "Secure communications over insecure channels," *Commun. ACM*, vol. 21, no. 4, pp. 294–299, 1978.

[26] K. Ren, W. Lou, and Y. Zhang, "LEDS: Providing location-aware end-to-end data security in wireless sensor networks," in *Proc. 24th IEEE Conf. Computer Communications (INFOCOM)*, Barcelona, Catalunya, Spain, 2006, pp. 1–12.

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

[28] V. V. Vazirani*, Approximation Algorithms*. New York: Springer-Verlag, 2002.

[29] Y. Wang, G. Attebury, and B. Ramamurthy, "A survey of security issues in wireless sensor networks," *IEEE Commun. Surveys Tutorials*, vol. 8, no. 2, pp. 2–23, 2006, 2nd Quarter.

[30] A. D. Wood and J. A. Stankovic, "Denial of service in sensor networks," *IEEE Computer*, vol. 35, no. 10, pp. 54–62, Oct. 2002.

[31] P. Xie and J.-H. Cui, "R-MAC: An energy-efficient MAC protocol for underwater sensor networks," in *Proc. Int. Conf. Wireless Algorithms, Systems, and Applications (WASA)*, Chicago, IL, 2007.

[32] C.-M. Yu, T.-Y. Chi, C.-S. Lu, and S.-Y. Kuo, "A constrained random perturbation vector-based pairwise key establishment scheme for wireless sensor networks," in *Proc. ACM Int. Symp. Mobile Ad Hoc Networking and Computing (MobiHoc)*, Hong Kong, China, 2008, pp. 449–450, (Poster Session).

[33] Z. Yu and Y. Guan, "A key management scheme using deployment knowledge for wireless sensor networks," *IEEE Trans. Parallel Distrib. Syst.*, vol. 19, no. 10, pp. 1411–1425, Oct. 2008.

[34] M. F. Younis, K. Ghumman, and M. Eltoweissy, "Location-aware combinatorial key management scheme for clustered sensor networks," *IEEE Trans. Parallel Distrib. Syst.*, vol. 17, no. 8, pp. 865–882, Aug. 2006.

[35] C.-M. Yu, C.-S. Lu, and S.-Y. Kuo, "A simple non-interactive pairwise key establishment scheme in sensor networks," in *Proc. IEEE Communications Society Conf. Sensor, Mesh and Ad Hoc Communications and Networks (SECON)*, Rome, Italy, 2009, pp. 1–9.

[36] F. Ye, H. Luo, S. Lu, and L. Zhang, "Statistical en-route filtering of injected false data in sensor networks," in *Proc. 24th IEEE Conf. Computer Communications (INFOCOM)*, Hong Kong, China, 2004, pp. 2446–2457.

[37] J. Zhao and R. Govindan, "Understanding packet delivery performance in dense wireless sensor networks," in *Proc. ACM Conf. Embedded Networked Sensor Systems (SenSys)*, Los Angeles, CA, 2003, pp. 1–13.

[38] J. Zhao, R. Govindan, and D. Estrin, "Computing aggregates for monitoring wireless sensor networks," in *Proc. IEEE Int. Workshop on Sensor Network Protocols and Applications*, Anchorage, AK, 2003.

[39] S. Zhu, S. Setia, and S. Jajodia, "LEAP: Efficient security mechanisms for large-scale distributed sensor networks," in *Proc. Annu. ACM Computer and Communications Security (CCS)*, Washington, DC, 2003, pp. 62–72.

[40] W. Zhang, M. Tran, S. Zhu, and G. Cao, "A random perturbation-based scheme for pairwise key establishment in sensor networks," in *Proc. ACM Int. Symp. Mobile Ad Hoc Networking and Computing (MobiHoc)*, Montreal, QC, Candada, Sep. 9–14, 2007.

Chia-Mu Yu (S'09) is currently working toward the Ph.D. degree at the Department of Electrical Engineering, National Taiwan University, Taipei, Taiwan.

He has been a research assistant with the Institute of Information Science, Academia Sinica, Taipei, Taiwan, since 2004. His research interests include experimental and theoretical aspects of sensor network security.

Chun-Shien Lu (M'99) received the Ph.D. degree in electrical engineering from National Cheng-Kung University, Tainan, Taiwan, in 1998.

From October 1998 to July 2002, he was with the Institute of Information Science, Academia Sinica, Taiwan, as a postdoctoral fellow for his military service. From August 2002 to June 2006, he was an assistant research fellow at the same institute. Since July 2006, he has been an associate research fellow. His current research interests mainly focus on various topics (including security and signal processing) of multimedia, compressive sensing, and sensor network security. He organized a special session on Multimedia Security in the 2nd and 3rd IEEE Pacific-Rim Conference on Multimedia, respectively (2001–2002). He co-organized two special sessions (in the area of media identification and DRM) in the 5th IEEE International Conference on Multimedia and Expo (ICME), 2004. He is a guest co-editor of the *Special Issue on Visual Sensor Network, EURASIP Journal on Applied Signal Processing*, in 2005. He holds two U.S. patents, three R.O.C. patents, and one Canadian patent in digital watermarking.

Dr. Lu won the Ta-You Wu Memorial Award, National Science Council in 2007 and was a co-recipient of the National Invention and Creation Award in 2004. He is a member of the ACM.

Sy-Yen Kuo received the B.S. degree in electrical engineering from National Taiwan University, in 1979, the M.S. degree in electrical and computer engineering from the University of California at Santa Barbara, in 1982, and the Ph.D. degree in computer science from the University of Illinois at Urbana-Champaign, in 1987.

(S'85–M'88–SM'98–F'01) is a Distinguished Professor at the Department of Electrical Engineering, National Taiwan University, Taipei, Taiwan, and was the Chairman at the same department from 2001 to 2004. He was a Chair Professor and Dean of the College of Electrical and Computer Engineering, National Taiwan University of Science and Technology from 2006 to 2009. He spent his sabbatical years as a Visiting Professor at the Computer Science and Engineering Department, the Chinese University of Hong Kong from 2004–2005 and as a visiting researcher at AT&T Laboratories-Research, New Jersey, from 1999 to 2000, respectively. He was the Chairman of the Department of Computer Science and Information Engineering, National Dong Hwa University, Taiwan from 1995 to 1998, a faculty member in the Department of Electrical and Computer Engineering at the University of Arizona from 1988 to 1991, and an engineer at Fairchild Semiconductor and Silvar-Lisco, both in California, from 1982 to 1984. In 1989, he also worked as a summer faculty fellow at the Jet Propulsion Laboratory of California Institute of Technology. His current research interests include dependable systems and networks, mobile computing, and quantum computing and communications. He has published more than 300 papers in journals and conferences, and also holds more than 10 U.S. and Taiwan patents.

Dr. Kuo received the distinguished research award between 1997 and 2005 consecutively from the National Science Council in Taiwan and is now a Research Fellow there. He was also a recipient of the Best Paper Award in the 1996 International Symposium on Software Reliability Engineering, the Best Paper Award in the simulation and test category at the 1986 IEEE/ACM Design Automation Conference (DAC), the National Science Foundation's Research Initiation Award in 1989, and the IEEE/ACM Design Automation Scholarship in 1990 and 1991.