A Secure Network Coding-based Data Gathering Model and Its Protocol in Wireless Sensor Networks

Qian Xiao*, Kang-feng Zheng, Shou-shan Luo
Information Security Center, Beijing University of Posts and Telecommunications
National Engineering Laboratory for Disaster Backup and Recovery, Beijing University of Posts and Telecommunications
Beijing, 100876, China

Xu Cui
Science and Technology on Communication Security Laboratory
Chengdu, 610041, China

Received 1 December 2011

Accepted 15 June 2012

Abstract

To provide security for data gathering based on network coding in wireless sensor networks (WSNs), a secure network coding-based data gathering model is proposed, and a data-privacy preserving and pollution preventing (DPP&PP) protocol using network coding is designed. DPP&PP makes use of a new proposed pollution symbol selection and pollution (PSSP) scheme based on a new obfuscation idea to pollute existing symbols. Analyses of DPP&PP show that it not only requires low overhead on computation and communication, but also provides high security on resisting brute-force attacks.

Keywords: wireless sensor networks; data gathering; network coding; data-privacy preserving; pollution preventing; data obfuscation

1. Introduction

Recently, a pioneering communication technology, i.e. network coding[1], has been widely used in the process of data gathering in wireless sensor networks (WSNs). This technology breaks the limitation of traditional routings that only allow intermediate nodes to store and forward data, it also allows them to mix the received information, which brings numerous advantages, such as improving network throughput and robustness, reducing energy consumption and time delay, etc.[2]. Moreover, to provide security for data communication, research on secure network coding (SNC) has attracted attentions of many scholars.

Some SNC protocols are designed for preserving private data from being obtained by any intermediate node that is not an intended recipient[3-5]. Some are for resisting eavesdropping attacks[6-8], and the others are for resisting pollution attacks[9-14], both of which are external attacks. Especially, Cascella et al.[5] proposed a weak data secrecy (WDS) scheme via obfuscation used in P2P content distribution systems. This scheme makes use of the vulnerability of network coding to pollution to preserve source private data, because it will be unable to reconstruct the original data when some corrupted

*Corresponding author: xiaoqianbupt@gmail.com

Published by Atlantis Press

blocks are injected[13]. Specifically, WDS fully utilizes such vulnerability to prevent intermediate nodes from acquiring source data but allows destination nodes to identify the corrupted blocks by setting and distributing secure random checksums (SRCs) to them.

Unfortunately, although research results of SNC are rich, still exists shortness: firstly, there are few SNC methods can be used in WSNs directly for providing security in the process of data gathering. That is because most of them either are concentrated or need a very high overhead on computation and communication. Secondly, almost all the existing methods about resisting eavesdropping attacks limit eavesdropping abilities of attackers. Thirdly, there is a lack of SNC methods that achieve not only avoiding dishonesty of internal nodes but also defending the communication from external attacks.

Thus, we are motivated to investigate a secure network coding-based data gathering (SNCDG) model and its protocol according to some characteristics of WSNs, e.g., distributed, self-organized and energy constrained. Our contributions are summarized as follows:

- Formalized Model SNCDG: we propose SNCDG, a formalized secure network coding model for data gathering in WSNs. It divides the threats posed by attackers into two types, one is to the source private data, which is posed by intermediate nodes and external eavesdroppers; the other is to the correctness of received data, which is posed by external pollution attackers.
- Secure Network Coding Protocol DPP&PP: we present a data-privacy preserving and pollution preventing (DPP&PP) protocol used in the model SNCDG. Our protocol learns from the method of WDS[5] that to provide security by using the vulnerability of network coding to pollution. However, unlike WDS, DPP&PP employs a new data obfuscation idea, and uses a pollution symbol selection and pollution (PSSP) scheme based on this new idea to implement pollution.
- Theoretical Analyses and Comparisons: through the analyses of DPP&PP on security, computational overhead and communicational overhead, we show that by using the new obfuscation idea, DPP&PP not only provides much higher security on resisting brute-force attacks than WDS, but also needs much lower overhead on communication.

The rest of the paper is organized as follows. In Section 2, we introduce the related work. In Section 3, we present the formalized model SNCDG. In Section 4, we give some preliminaries, mainly the details of WDS which is proposed in Ref. 5 and its shortness when used directly in our model SNCDG. In Section 5, we propose our protocol DPP&PP, analyze it and compare it with WDS in terms of security, computational overhead and communicational overhead. Section 6 concludes the paper.

2. Related Work

Generally speaking, existing SNC methods could fall into two kinds according to the type of adversaries. The first kind is mainly for resisting internal attackers, i.e., preventing intermediate nodes from acquiring source private data by using the information flowing across them[3-5]. Lima et al.[3] showed that network coding could offer information-theoretic security against internal attackers with high probability, if the source disseminates encoded blocks of data by using multiple paths and the size of the codeword field goes to infinity. The scheme WDS[5] proposed by Cascella et al. mainly uses corrupted blocks to prevent intermediate nodes from acquiring the original blocks, and uses SRCs to help destination nodes identify those corrupted blocks. The second kind is mainly for resisting the external attackers who make an eavesdropping attack or a pollution attack. In an eavesdropping attack, the attacker attempts to deduce and acquire source private data by eavesdropping on some links[6–8]. Cai and Yeung[6] presented a secure linear network coding method that achieves perfect information-theoretic security. However, their method limits the eavesdropping abilities of attackers to only getting access to a limited subset of network links. In a pollution attack, by using the vulnerability of network coding to pollution, the attacker injects bogus packets or modifies encoded ones for preventing destination nodes from decoding correctly[9-14]. In general, such attack can be resisted by designing corrupted packets detection methods[9-14] or network error correction codes[15-17].

However, these protocols are not designed for being used in WSNs, except Refs. 4, 13, 14. Besides, most of them could not be used directly in WSNs, because they all require much energy on computation and communication. Ref. 4 focuses on resisting deductive attacks by internal attackers, and Ref. 13 and Ref. 14

focus on resisting pollution attacks by external attackers. Lu et al.[4] proposed a secure multi-path network coding schemes for WSNs. This scheme prevents any intermediate node from acquiring source data by deleting some paths in order to provide any node with insufficient encoded packets. Yu et al.[13] proposed a homomorphic signature scheme so as to detect and filter pollution attacks. However, in order to be suitable for WSNs, they had to use an alternate lightweight scheme that provides much lower security. Apavatjrut et al.[14] compared several different schemes based on message authentication codes algorithms for resisting pollution attacks, but they only focused on XOR network coding[18]. In addition, among those researches, none was designed for being used to resist both internal and external attackers.

In this paper, we provide security by using data obfuscation and the vulnerability of network coding to pollution, which are also the main technologies of WDS[5]. However, on the one hand, unlike WDS, we use a new data obfuscation idea to pollute blocks, which requires much lower overhead on communication but provides much higher security. Thus, our protocol could be used in WSNs. On the other hand, our protocol is able to resist both internal and external attackers, which is unable to be achieved by WDS and most of existing protocols.

3. Formalized Model SNCDG

As mentioned above, the process of data gathering may suffer from three kinds of attacks, deductive attacks by intermediate nodes, eavesdropping attacks and pollution attacks by external attackers. The first two kinds of attacks aim at acquiring source private data; while the third one aims at preventing sink nodes from receiving source data correctly. As a result, we present the formalized model SNCDG as follows, which consists of three parts (see Fig. 1).

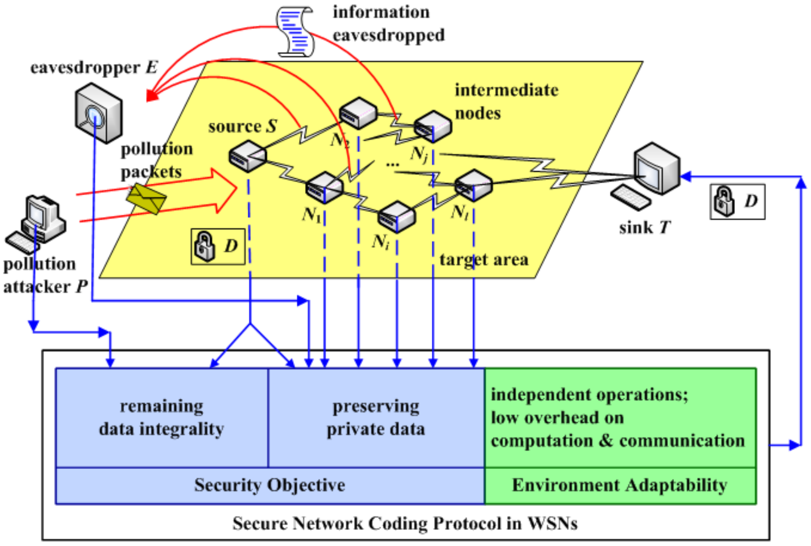

Fig. 1. Formalized model SNCDG.

(i) Network model

Consider such a kind of WSNs, and take one of them for instance. It is an event-driven WSN, which consists of a set of sensor nodes and a sink T. Sensor nodes are energy-constrained and immobile, deployed randomly in a target area. T is resource-rich and it is fixed at the edge of the area. Assume a sensor node S wants to transmit a private data D to T based on network coding with the help of intermediate nodes N_1, …, N_l along several paths previously established. There also exist an external eavesdropper E and an external pollution attacker P. E can eavesdrop on any one or several links of the WSN, and P can inject pollution packets into it.

(ii) Security objective

(a) Preserving private data: It aims at not only preventing any one or several intermediate nodes among N_1, …, N_l from deducing source data by the information forwarded by them, but also averting source data being acquired by E via eavesdropping.

(b) Remaining data integrality: It aims at resisting pollution attacks of P and making sure that T could reconstruct source data correctly and integrally.

(iii) Environment adaptability

(a) Nodes N_1, …, N_l will still re-encode independently when using a SNC protocol.

(b) Both encryption operations and communication times should be as less as possible.

As shown in Fig. 1, the source S wants to transmit the data D to the sink T by using a SNC protocol. This protocol will not only achieve the security objective of SNCDG mentioned above, but also be adapted to the environment. As a result, the aim of this paper is to design such a SNC protocol that owns the above features.

Besides, it is worth noting that, although data-privacy preserving has to be achieved in two attacks, i.e. the internal deductive attack and the external eavesdropping attack, but the requirement to the former is stricter. That is because the former has to maintain the willingness of intermediate nodes on cooperation as well. If all the source data are transmitted in ciphers, intermediate nodes may lose their willingness on cooperation, since they can't get any benefit from following the protocol. Furthermore, averting eavesdropping attacks could be finally realized by avoiding deductive attacks of intermediate nodes. The reason is that, an eavesdropping attack can be understood as the scenario that some dishonest intermediate nodes send their received data to the external eavesdropper. In summary, a data-privacy preserving protocol only needs to prevent intermediate nodes from deducing source data as well as to keep their willingness on cooperation.

4. Preliminary: Protocol WDS

In this section, we will give a detailed description of the privacy-preserving protocol WDS proposed in Ref. 5. WDS also takes data obfuscation and the vulnerability of network coding to pollution as the main technologies, and it is able to maintain willingness of intermediate nodes on cooperation. However, for some reasons we will also give in this section, WDS is unable to be used directly in our model SNCDG. The process of WDS can be divided into two parts, i.e. pollution process and detection process, details of which are as follows.

(i) Pollution process

First of all, the source devides a file into n blocks, denoted by $\mathbf{f}_1$,…, $\mathbf{f}_n$. Each block $\mathbf{f}_i$ consists of m symbols defined in the finite field $\mathbf{GF}(2^q)$, denoted by $\mathbf{f}_i=(f_{i,1}, \ldots, f_{i,m})$, i=1, …,n. In practical application, it is usual that m>>n. Secondly, the source introduces k corrupted blocks into the original n ones, where k=an and $a\in[0,1]$. These n+k processed blocks are denoted by $\mathbf{f}_1'$,…, $\mathbf{f}_{n+k}'$, and they are distributed by using a random network coding protocol[19]. Specifically, the source sends out the packet $\mathbf{f}_i'$ with the ith unit vector of length n+k packaged in the header, i=1, …, n+k. ("packet" and "block" are interchangeable in this paper). When an intermediate node wants to forward a packet, it creates an encoded one, say $\mathbf{y}$, by linear combining the blocks it saved currently with random coefficients selected from the finite field $\mathbf{GF}(2^q)$. By recursion, $\mathbf{y}$ is ultimately a linear combination of blocks $\mathbf{f}_1'$, …, $\mathbf{f}_{n+k}'$, i.e., $\mathbf{y}$= $(y_1', \ldots, y_m')$= $\sum_{i=1}^{n+k} c_i\mathbf{f}_i'$ = ($\sum_{i=1}^{n+k} c_i f_{i,1}'$, …, $\sum_{i=1}^{n+k} c_i f_{i,m}'$), where $\mathbf{c}=(c_1, \ldots, c_{n+k})$ is the coding vector of $\mathbf{y}$, which is packaged in the header of $\mathbf{y}$.

As long as the destination node receives n+k packets whose coding vectors are linear independent, it will decode and obtain blocks $\mathbf{f}_1'$, …, $\mathbf{f}_{n+k}'$ by using Gaussian elimination. However, those k corrupted blocks are included. In order to identify them, the following detection process is required.

(ii) Detection process

The source sets a secure random checksum (SRC) for each block among $\mathbf{f}_1'$, …, $\mathbf{f}_{n+k}'$. These values can not only help the destination node to identify corrupted blocks, but also induce intermediate nodes to go on cooperating. The specific way to set and transmit SRCs is as follows.

(a) The source selects m coefficients randomly from $\mathbf{GF}(2^q)$, denoted by $\mathbf{t}=(t_1,\ldots, t_m)$.

(b) It computes the SRC of $\mathbf{f}_i'$, denoted by SRC_i, where SRC_i= $\sum_{j=1}^{m} t_j f_{i,j}'$, i=1,…, n+k.

(c) The source replaces intentionally the SRCs of the k corrupted blocks with k random wrong values.

(d) It transmits all the valid SRCs to intermediate nodes in clear text, and transmits the SRCs with the k wrong ones to the destination node in ciphers.

(e) It transmits **t** to the destination node in clear text.

After the destination node decodes and obtains blocks $\mathbf{f}_1'$, …, $\mathbf{f}_{n+k}'$, it decrypts SRCs and uses them to identify all the corrupted blocks. Specifically, for each $\mathbf{f}_i'$, it checks whether $SRC(\mathbf{f}_i')= \sum_{j=1}^{m} t_j f_{i,j}'$ equals to SRC_i that it received or not. If not, $\mathbf{f}_i'$ is a corrupted block, and it drops it, i=1, …, $n+k$. At last, the destination node could drop all the corrupted blocks and reconstruct the original file.

As for intermediate nodes, they may test the validity of **y** that they receive by checking whether $SRC(\mathbf{y})= \sum_{j=1}^{m} t_j y_j'$ equals to $\sum_{i=1}^{n+k} c_i SRC_i$ or not. However, the SRCs they received are valid. Therefore, the following equation[9] will always hold.

$$\begin{aligned} SRC(\mathbf{y}) &= \sum_{j=1}^{m} t_j y_j' = \sum_{j=1}^{m} t_j (\sum_{i=1}^{n+k} c_i f_{i,j}') \\ &= \sum_{i=1}^{n+k} c_i (\sum_{j=1}^{m} t_j f_{i,j}') = \sum_{i=1}^{n+k} c_i SRC_i \end{aligned} \qquad (1)$$

As a result, they believe that **y** is valid, and their willingness on cooperation is maintained. In fact, **y** is linear combined by the blocks that include corrupted ones. That is to say, even intermediate nodes could decode, they will only obtain $\mathbf{f}_1'$, …, $\mathbf{f}_{n+k}'$ and could not identify the corrupted blocks among them. Thus, they can't acquire correct source data.

It is worth noting that, WDS is simple, but it is not suitable for being used directly in the model SNCDG. There are mainly three reasons. Firstly, this method will waste too much overhead on computation and communication when it is used in practice, as the source injects additional blocks to obfuscate the original file. Secondly, since the value of k is revealed by the length of coding vectors, the probability of performing a brute-force attack successfully under WDS is very high, which is 1/C(n+k,n)[5]. That is to say, WDS only provides very weak secrecy. Thirdly, the destination node could not tell the corrupted blocks of the source and those of external pollution attackers apart.

5. Our Protocol DPP&PP

This paper proposes a data-privacy preserving and pollution preventing (DPP&PP) network coding protocol by using a new data obfuscation idea. Comparing with the obfuscation idea used by WDS, the differences and advantages of ours are as follows. Firstly, the source only pollutes existing blocks instead of adding additional ones, which will reduce enormously the overhead on transmitting and encoding additional blocks. Secondly, the source only pollutes some symbols of a block rather than the whole block according to a pollution symbol selection and pollution (PSSP) scheme. Through this method, the security against brute-force attacks will be improved, and the computation complexity will be reduced. Thirdly, the number of polluted symbols is concealed in our method, which strengthens the ability of our protocol against brute-force attacks. Fourthly, the destination node could tell the corrupted blocks of the source S and those of the external pollution attacker P apart by our method, which ensures decoding correctly.

Firstly, we present a PSSP scheme that satisfies the above characteristics in subsection 5.1. Then, we propose our protocol DPP&PP based on this scheme in subsection 5.2. Finally, we give some analyses about the security, computational overhead and communicational overhead of DPP&PP, and compare it with WDS in subsection 5.3.

5.1. *Our PSSP scheme*

Definition 1. the source S divides the data D into n blocks, denoted by f1, …, fn. Each block consists of m symbols defined in the finite field GF(2q) (m>>n). Consider a matrix G that consists of blocks f1, …, fn, and fi is the ith row vector of it, i=1, …, n. Call the vectors g1, …, gm inclined vectors of G, where

$$\begin{aligned} \mathbf{g}_j &= \left(g_{1,j},\ldots,g_{n,j}\right) \\ &= \begin{cases} \left(f_{1,(m-n+j+1)},\ldots,f_{n-j,m},f_{n-j+1,1},\ldots,f_{n,j}\right) & j\in\{1,\ldots,n-1\} \\ \left(f_{1,j-n+1},\ldots,f_{i,j-n+i},\ldots,f_{n,j}\right) & j\in\{n,\ldots,m\} \end{cases} \end{aligned} \qquad (2)$$

Fig. 2 shows an example of inclined vectors, where n=4 and m=9. That is to say, matrix **G** could be viewed as being composed by 4×9 grids, and each grid represents a symbol. To move in parallel the n(n-1)/2 grids on the upper-right corner of **G** to the left, then the inclined vectors $\mathbf{g}_1$,…, $\mathbf{g}_m$ are shown by oblique lines in Fig. 2.

S pollutes several symbols of blocks $\mathbf{f}_1$, …, $\mathbf{f}_n$ according to the following PSSP scheme (see Fig. 3).

(i) *S* selects *h* vectors randomly from $\mathbf{g}_1,\ldots,\mathbf{g}_m$, denoted by $\mathbf{g}_{j_1},\ldots,\mathbf{g}_{j_h}$, where *h*=*bm* is an even and *b* ∈ [0,1]. Besides, *h* is concealed from any other nodes.

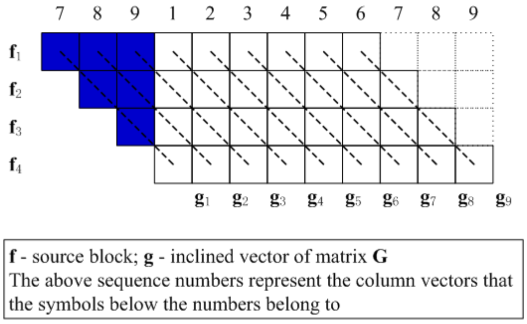
7 8 9 1 2 3 4 5 6 7 8 9
f1
f2
f3
f4
g1 g2 g3 g4 g5 g6 g7 g8 g9
f - source block; g - inclined vector of matrix G
The above sequence numbers represent the column vectors that the symbols below the numbers belong to

Fig. 2. Inclined vectors of matrix **G** (*n*=4 and *m*=9).

(ii) *S* selects randomly a pollution parameter **r**=(r_1, …, r_n) and *m* coefficients **t**=(t_1,…, t_m), where r_1,…, r_n, t_1,…, t_m are defined in **GF**(2^q).

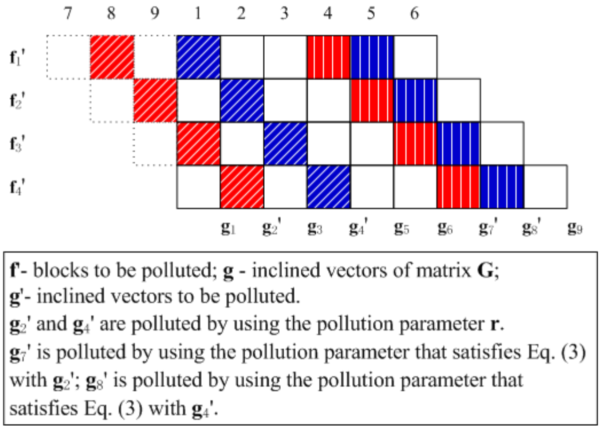
7 8 9 1 2 3 4 5 6
f1'
f2'
f3'
f4'
g1 g2' g3 g4' g5 g6 g7' g8' g9
f'- blocks to be polluted; g - inclined vectors of matrix G;
g'- inclined vectors to be polluted.
g2' and g4' are polluted by using the pollution parameter r.
g7' is polluted by using the pollution parameter that satisfies Eq. (3) with g2'; g8' is polluted by using the pollution parameter that satisfies Eq. (3) with g4'.

Fig. 3. An example about the scheme PSSP (*n*=4 and *m*=9).

(iii) *S* pollutes the first *h*/2 inclined vectors by the parameter **r**. Specifically, for $\mathbf{g}_{j_k}=\left(g_{1,j_k},\ldots,g_{n,j_k}\right)$, it replaces g_{i,j_k} with $g_{i,j_k}+r_i$, *i*=1, …, *n*, *k*=1, …, *h*/2. For the *k*th vector $\mathbf{g}_{j_k}$ of the last *h*/2 inclined vectors, *S* computes the coefficient $r_{i,k}$ that satisfies Eq. (3) below, and replaces g_{i,j_k} with $g_{i,j_k}+r_{i,k}$, *i*=1, …, *n*, *k*=*h*/2+1, …, *h*.

$$t_{j_{(k-h/2)}} r_i + t_{j_k} r_{i,k} = 0 \qquad (3)$$

Additionally, as for the reason why to select pollution symbols according to inclined vectors rather than column ones, that is because the former could ensure that the distribution of pollution symbols will be more even.

5.2. *Description of our protocol DPP&PP*

The basic idea of our protocol DPP&PP is as follows. Firstly, the source *S* pollutes some symbols according to the scheme PSSP. Besides, it computes and distributes SRCs of all the blocks, which ensures that intermediate nodes and the sink *T* could tell corrupted blocks of *S* and those of *P* apart (The detailed analyses will be given in subsection 5.3). Secondly, to recover the symbols polluted by *S* after identified and dropped all the corrupted blocks of *P*, *T* has to set SRCs of inclined vectors additionally. Finally, our protocol implements a linear all-or-nothing transformation (LAT)[20] by using an invertible matrix with non zero entries, before transmitting data based on network coding. It can increase the complication of performing a brute-force attack and provide information-theoretic security as nothing will be revealed except that intermediate nodes can guess out all the corrupted symbols. This kind of process has also been applied in WDS and some other SNC protocols[7,8].

Assume that *S* shares a symmetric key K_1 with other sensor nodes, and *S* shares a key K_2 with *T*. The protocol DPP&PP based on the above ideas is described as follows.

(i) The source pollution and transmission process

Step1: *S* divides the data *D* into *n* blocks, and performs a LAT with a matrix **M** on them, where **M** is invertible and consists of n^2 non zero entries defined in **G**(2^q). After the process, these *n* blocks are denoted by $\mathbf{f}_1$, …, $\mathbf{f}_n$.

Step2: *S* pollutes $\mathbf{f}_1$, …, $\mathbf{f}_n$ according to the scheme PSSP. After the pollution, these blocks are denoted by $\mathbf{f}_1'$, …, $\mathbf{f}_n'$. Accordingly, the inclined vectors before and after the pollution are denoted by $\mathbf{g}_1$, …, $\mathbf{g}_m$ and $\mathbf{g}_1'$, …, $\mathbf{g}_m'$, respectively; the pollution parameter is denoted by **r**=(r_1, …, r_n); the coefficients selected randomly are denoted by **t**=(t_1, …, t_m). *S* transmits blocks $\mathbf{f}_1'$, …, $\mathbf{f}_n'$ to *T* by using a random network coding protocol[19], where $\mathbf{f}_i'$ takes the *i*th unit vector of length *n* as its coding vector, packaged in its header, *i*=1, …, *n*.

Step3: *S* computes and distributes SRCs according to the following sub-steps.

(a) *S* computes SRCs of $\mathbf{f}_i$ and $\mathbf{f}_i'$, denoted by SRC_i and SRC_i' respectively, where $SRC_i=\sum_{j=1}^{m} t_j f_{i,j}$ and $SRC_i'=\sum_{j=1}^{m} t_j f_{i,j}'$, i=1, …, n.

(b) *S* selects n coefficients $\mathbf{t}'=(t_1',\ldots,t_n')$ randomly, where $t_i' \in \mathbf{GF}(2^q)$, i=1, …, n. Then, it computes SRCs of $\mathbf{g}_i$ and $\mathbf{g}_i'$, denoted by $\underline{SRC}_j$ and $\underline{SRC}_j'$ respectively, where $\underline{SRC}_j=\sum_{i=1}^{n} t_i' g_{i,j}$ and $\underline{SRC}_j'=\sum_{i=1}^{n} t_i' g_{i,j}'$, j=1, …, m.

(c) *S* encrypts SRC_1',…, SRC_n' and $\underline{SRC}_1'$,…, $\underline{SRC}_m'$ with the key K_1, and transmits them to intermediate nodes. Then, *S* encrypts $\mathbf{r}$, SRC_1,…, SRC_n and $\underline{SRC}_1$,…, $\underline{SRC}_m$ with the key K_2, and transmits them to *T*. At last, *S* transmits vectors $\mathbf{t}$, $\mathbf{t}'$ and the matrix $\mathbf{M}$ to *T* in clear text.

(ii) The intermediate nodes detection and forwarding process

Step1: When an intermediate node receives an encrypted SRC, it decrypts it and saves the result. When it receives an encoded packet, it saves it directly. Once an intermediate node receives all the SRCs, it turns to Step2.

Step2: the intermediate node checks the validity of each packet it has already saved. Specifically, for the packet $\mathbf{y}$, it checks if $SRC(\mathbf{y})=\sum_{j=1}^{m} t_j y_j'$ equals to $\sum_{i=1}^{n} c_i SRC_i'$ or not. If not, $\mathbf{y}$ is a polluted packet injected by the external pollution attacker *P*, it drops it.

Step3: According to the sequence with which it received and saved these packets, for each packet $\mathbf{y}=(y_1', \ldots, y_m')=\sum_{i=1}^{n} c_i \mathbf{f}_i'$, the intermediate node checks whether the coding vector of $\mathbf{y}$, say $\mathbf{c}=(c_1, \ldots, c_n)$, is linear independent to all those of the encoded packets which are received earlier than $\mathbf{y}$ and whose validity has already been checked. If not, it drops $\mathbf{y}$; otherwise, it saves it.

(iii) The sink decoding and identification process

Step1: When *T* receives $\mathbf{t}$, $\mathbf{t}'$, $\mathbf{M}$ or an encoded packet, it saves them directly. When it receives encrypted $\mathbf{r}$ and SRCs, it decrypts them and saves the results. Once *T* receives all the SRCs, it turns to Step2.

Step2: *T* checks the validity of each packet it has already saved. That is to say, for packet $\mathbf{y}$, it judges if $SRC(\mathbf{y})=\sum_{j=1}^{m} t_j y_j'$ equals to $\sum_{i=1}^{n} c_i SRC_i$ or not. If not, $\mathbf{y}$ is a polluted packet of *P*, it drops it.

Step3: According to the sequence with which *T* received and saved these packets, for each packet $\mathbf{y}$, it checks whether the coding vector of $\mathbf{y}$ is linear independent to all those of the encoded packets which are received earlier than $\mathbf{y}$ and whose validity has already been checked by *T*. If not, it drops $\mathbf{y}$; otherwise, it saves it. Once *T* saves n valid encoded packets, it turns to Step4.

Step4: *T* decodes packets by Gaussian elimination to obtain blocks $\mathbf{f}_1'$, …, $\mathbf{f}_n'$. Then it recovers original blocks with the decrypted $\mathbf{r}$ according to the sub-steps as follows.

(a) For inclined vector $\mathbf{g}_j'$, *T* compares $\underline{SRC}(\mathbf{g}_j')=\sum_{i=1}^{n} t_i' g_{i,j}'$ with $\underline{SRC}_j$ it received. If they are not equal, then $\mathbf{g}_j'$ is polluted by the source *S*, j=1, …, m.

(b) If $\mathbf{g}_j'$ belongs to the first h/2 polluted vectors, *T* recovers original symbols by subtracting $\mathbf{r}$ from it; otherwise, *T* computes the corresponding pollution parameter according to Eq. (3), and uses it to recover the original inclined vector $\mathbf{g}_j$. Finally, *T* recovers the original blocks $\mathbf{f}_1$, …, $\mathbf{f}_n$ according to the above steps.

Step5: *T* reconstructs *D* by performing an inverse operation of the LAT on $\mathbf{f}_1$, …, $\mathbf{f}_n$ with the matrix $\mathbf{M}^{-1}$.

5.3. *Analyses of DPP&PP*

5.3.1. Security

First of all, we will discuss the security of our protocol DPP&PP on preserving data privacy. On the one hand, all the SRCs that intermediate nodes will receive are valid values corresponding to the blocks and the inclined vectors after the pollution. Thus, based on the same reason with Ref. 5, for any encoded block $\mathbf{y}$, it will pass the validity test as long as it is not polluted by the external attacker *P*, because

$$\begin{aligned} SRC(\mathbf{y}) &= \sum_{j=1}^{m} t_j y_j' = \sum_{j=1}^{m} t_j \big(\sum_{i=1}^{n} c_i f_{i,j}'\big) \\ &= \sum_{i=1}^{n} c_i \big(\sum_{j=1}^{m} t_j f_{i,j}'\big) = \sum_{i=1}^{n} c_i SRC_i' . \end{aligned} \tag{4}$$

However, $\mathbf{y}$ is actually generated by the blocks that include source polluted blocks. As a result, even though intermediate nodes could decode, they will at most obtain blocks $\mathbf{f}_1'$, …, $\mathbf{f}_n'$ and valid SRCs corresponding to inclined vectors. Thus, they are unable to identify source polluted symbols among them, and are unable to reconstruct data D at last. On the other hand, to resist brute-force attacks, the LAT ensures that attackers can't obtain any part of original blocks $\mathbf{f}_1, \ldots, \mathbf{f}_n$ unless they can guess out all the polluted inclined vectors and guess out the value $\mathbf{r}$ to retrieve them. Since h is concealed from other nodes, the probability of performing a brute-force attack successfully under DPP&PP is only $1/2^{m/2+nq}$. It is much smaller than the probability under WDS, which is $1/C(n+k,n)$.

Secondly, we will discuss the security of DPP&PP on resisting pollution attacks by the external attacker P. On the one hand, intermediate nodes could distinguish between polluted blocks of S and those of P according to Eq. (4). That is because the former satisfies it, but the latter does not. On the other hand, S always uses the values that meet Eq. (3) to pollute symbols. As a result, when T checks the validity of $\mathbf{y}$, as long as it is not a polluted block of P, the following equation will always hold.

$$SRC(\mathbf{y}) = \sum_{j=1}^{m} t_j y_j' \overset{\text{Eq.(4)}}{=} \sum_{i=1}^{n} c_i \left(\sum_{j=1}^{m} t_j f_{i,j}'\right)$$

$$\overset{\text{Eq.(3)}}{=} \sum_{i=1}^{n} c_i \left(\sum_{j=1}^{m} t_j f_{i,j}\right) = \sum_{i=1}^{n} c_i SRC_i \qquad (5)$$

In other words, polluted blocks of S could pass the validity test of T, but those of P could not pass it. Thus, T could tell polluted blocks of S and those of P apart, and prevent external pollution attacks.

5.3.2. Computational overhead

In this paper, we only consider computational overhead on encryption, defined as the number of the blocks that the source encrypts. The length of a block is mq bits. Under DPP&PP, S needs encrypt $2(n+m)$ SRCs of length q bits and a vector $\mathbf{r}$ of length nq bits. Thus, its overhead on computation is about $O(2+3n/m)$. Similarly, the computational overhead under WDS is about $O((n+k)/m)$.

5.3.3. Communicational overhead

In this paper, we measure the overhead on communication in terms of the number of the blocks that the source transmits. Under DPP&PP, S transmits n encoded blocks of length mq bits, a vector $\mathbf{t}$ of length mq bits, a vector $\mathbf{t}'$ of length nq bits and a matrix $\mathbf{M}$ of length n^2q bits in clear text. Besides, it transmits $2(n+m)$ SRCs of length q bits and a vector $\mathbf{r}$ of length nq bits in ciphers. In summary, the overhead on communication under DPP&PP is about $O(n+3+(n^2+4n)/m)$. Similarly, the communicational overhead under WDS is about $O(n+k+1+(n^2+2(n+k))/m)$.

In summary, we draw a comparison between DPP&PP and WDS, and list the results in Table 1 as follows, where OComp is short for overhead on computation, and OComm is short for overhead on communication.

Table 1. Comparison between WDS and DPP&PP

Scheme	Performance		
	Security	OComp	OComm
WDS	$O\left(\frac{1}{C(n+k,n)}\right)$	$O\left(\frac{n+k}{m}\right)$	$O\left(n+k+1+\frac{n^2+2(n+k)}{m}\right)$
DPP&PP	$O\left(\frac{1}{2^{m/2+nq}}\right)$	$O\left(2+\frac{3n}{m}\right)$	$O\left(n+3+\frac{n^2+4n}{m}\right)$
Better	DPP&PP	WDS	DPP&PP

Firstly, as is shown in Table 1, DPP&PP provides much higher security than WDS. Secondly, when k is larger than 2, its overhead on communication is also lower than that of WDS, and k is much larger than 2 in practical. Thirdly, although DPP&PP needs higher overhead on computation than WDS, the additional amount of computational overhead that DPP&PP needs to pay is less than 3 blocks. Comparing to communicational overhead, such additional overhead could be neglected. Based on an overall consideration of these three performances, we conclude that DPP&PP is much better, since it not only needs low overhead on computation and communication, but also provides sufficient security.

6. Conclusion

In this paper, we investigated the problem of secure data gathering in WSNs based on network coding. We proposed a formalized model SNCDG, and designed a SNC protocol used in this model, called DPP&PP. This protocol achieves the security objective and the requirements of environment adaptability of our model. That is to say, on the one hand, DPP&PP can resist both internal and external attackers. In other words, it not only preserves source private data, but also prevents external pollution attacks. On the other hand, DPP&PP needs low overhead on computation and communication, but provides higher and sufficient security on resisting brute-force attacks, so it is suitable for WSNs with limited energy.

Acknowledgment

This work is supported by National Natural Science Foundation of China (61070204, 61070208, 61101108); National S&T Major Program (2011ZX03002-005-01); the Foundation of the Science and Technology on Communication Security Laboratory (9140C1105061005).

References

1. R. Ahlswede, N. Cai, S.-Y. R. Li and R. W. Yeung, Network information flow, *IEEE Trans. Inf. Theory* 46(4) (2000) 1204–1216.
2. P. A. Chou and Y. Wu, Network coding for the Internet and wireless networks, *IEEE Signal Process Mag.* 24(5) (2007) 77–85.
3. L. Lima, M. Medard and J. Barros, Random linear network coding: a free cipher? in *Proc. IEEE Int. Symp. Inf. Theor.* (IEEE, Nice, 2007), pp. 546–550.
4. F. Lu, L.J. Geng, L.-T. Chia and Y.-C. Liang, Secure multi-path in sensor networks, in *Proc. ACM Conf. Embedded Networked Sens. Syst.* (ACM, Sydney, NSW, 2007), pp. 413–414.
5. R. G. Cascella, Z. Cao, M. Gerla, B. Crispo and R. Battiti, Weak data secrecy via obfuscation in network coding based content distribution, in *IFIP Wirel. Days, WD* (Inst. of Elec. and Elec. Eng. Computer Society, Dubai, 2008), pp. 1–5.
6. N. Cai and R. W. Yeung, Secure network coding, in *Proc. IEEE Int. Symp. Inf. Theor.* (IEEE, Lausanne, 2002), p. 323.
7. M. X. Luo, Y. X. Yang, L. C. Wang and X. X. Niu, Secure network coding in the presence of eavesdroppers, *SCIENCE CHINA Information Sciences* 53(3) (2010) 648–658.
8. Q. Guo, M. X. Luo, L. X. Li and Y. X. Yang, Secure network coding against wiretapping and Byzantine attacks, *Eurasip J. Wireless Commun. Networking* 2010 (2010).
9. C. Gkantsidis and P. Rodriguez, Cooperative security for network coding file distribution, in *Proc. IEEE INFOCOM* (IEEE, Barcelona, 2006), pp. 1–13.
10. F. Zhao, T. Kalker, M. Medard and K. J. Han, Signatures for content distribution with network coding, in *Proc. IEEE Int. Symp. Inf. Theor.* (IEEE, Nice, 2007), pp. 556–560.
11. S. Jaggi, M. Langberg, S. Katti, T. Ho, D. Katabi and M. Médard, Resilient network coding in the presence of Byzantine adversaries, in *Proc. IEEE INFOCOM* (IEEE, Anchorage, AK, 2007), pp. 616–624.
12. T. Ho, B. Leong, R. Koetter, M. Medard, M. Effros and D. R. Karger, Byzantine modification detection in multicast networks with random network coding, *IEEE Trans. Inf. Theory* 54(6) (2008) 2798–2803.
13. Z. Yu, Y. W. Wei, B. Ramkumar and Y. Guan, An efficient signature-based scheme for securing network coding against pollution attacks, in *Proc. IEEE INFOCOM* (IEEE, Phoenix, AZ, 2008), pp. 1409–1417.
14. A. Apavatjrut, W. Znaidi, A. Fraboulet, C. Goursaud, C. Lauradoux and M. Minier, Energy friendly integrity for network coding in wireless sensor networks, in *Proc. Int. Conf. Netw. Syst. Secur., NSS* (IEEE Computer Society, Melbourne, VIC, 2010), pp. 223–230.
15. N. Cai and R. W. Yeung, Network coding and error correction, in *Proc. the 2002 IEEE Information Theory Workshop* (IEEE, Bangalore, 2002), pp. 119–122.
16. R. Koetter and F. R. Kschischang, Coding for errors and erasures in random network coding, *IEEE Trans. Inf. Theory* 54(8) (2008) 3579–3591.
17. Z. Zhang, Linear network error correction codes in packet networks, *IEEE Trans. Inf. Theory* 54(1) (2008) 209–218.
18. S. Katti, H. Rahul, W. Hu, D. Katabi, M. Medard and J. Crowcroft, XORs in the air: practical wireless network coding, *IEEE ACM Trans. Networking* 16(3) (2008) 497–510.
19. P. A. Chou, Y. Wu and K. Jain, Practical network coding, in *Proc. 41st Annu. Allerton Conf. Communication, Control, and Computing* (2003).
20. D. R. Stinson, Something about all or nothing (transforms), *Des. Codes Cryptography* 22(2) (2001) 133–138.