

# An Efficient Short Null Keys Based Scheme for Securing Network Coding Against Pollution Attacks

Junsheng Wang, Jin Wang<sup>(✉)</sup>, Yanqin Zhu, and Chengjin Jia

Department of Computer Science and Technology,  
Soochow University, Suzhou, China  
{20114227046,wjin1985,yqzhu,20134227049}@suda.edu.cn

**Abstract.** Network coding has gained wide attention nowadays for its significant advantages on many aspects compared with traditional routing mechanism. However, if there are malicious nodes launching pollution attacks by tampering or forging data packets in the communication network, the sink nodes will suffer from failure decoding, together with serious results such as bandwidth wasting, longer transmission delay and increasing computation overheads. The original null keys based pollution detection scheme cannot efficiently defend against pollution attacks when the system has colluding attackers because of high communication overheads. Therefore, we firstly define the concept of complete null space, with the property that no pollution packets can pass its verification. We then propose the idea of partial position detection and design an algorithm to construct short null keys. Secondly, we provide a short null keys based pollution detection scheme with network coding, which has lower overheads compared with the original null keys based pollution detection scheme in composing complete null space. Finally, rigorous theoretical proofs are given to analyze the security of the designed scheme.

**Keywords:** Network coding · Pollution detection · Null keys

## 1 Introduction

The idea of network coding has been firstly proposed by Ahlswede et al. in 2000 [1]. Compared with the traditional store-and-forward routing mechanism, network coding allows nodes to encode the received packets to generate new packets and forward the new ones. The theoretical maximum multicast rate can be achieved by using network coding [2].

Recent studies have proved the obvious advantages of network coding in improving network throughput [3], providing data confidentiality [4–8], providing

---

J. Wang—This work is supported in part by National Natural Science Foundation of China under grant No. 61202378, 61373164. It was also supported in part by China Postdoctoral Science Foundation No. 2013M531402, 2014T70544, and Application Foundation Research of Suzhou of China No. SYG201401.

data stream intraceability [9, 10], enhancing the robustness of data packets [3, 11], enhancing the reliability of the network, facilitating the recovery of redundant data storage [12] and so on. Although the introduction of network coding brings many benefits, the existing of malicious nodes, i.e., attackers, will significantly degrade the system performance in launching pollution attacks [13–17].

Pollution attacks, also known as jamming attacks [18, 19] or byzantine attacks [20], belong to active attacks. Malicious nodes tamper or forge the data packets to generate pollution packets and then spread the generated packets into the network. The participation of pollution packets during the encoding of data packets will produce new pollution packets. Consequently, it will result in the diffusion of pollution in the network and further the failure of decoding in sink nodes, without exploiting error detection or correction. Therefore it leads to the waste of bandwidth, the delay in transmission and the increasing of computational overheads.

The null keys based on-the-fly detection scheme is first proposed in [19] to defend against pollution attacks. The null keys used in [19] is referred as *original null keys (ONKs)* and the scheme is referred as *ONKs scheme* in this paper. Besides pollution detection, the scheme further has many other advantages such as good distribution characteristic, low computation overheads during verification, simple to implement, etc. However, since the null keys distributed by the source node cannot compose the complete null space, the obtained distributed null space by colluding attackers will lead to the vanished security of the system.

Based on the *ONKs scheme* proposed in [19], we aim to design an efficient pollution detection scheme in this paper. The main contributions of this paper are summarized as follows:

- We generalize the concept of null keys and propose the concept of complete null space. The existence of complete null space has the ability to put an end to the efficient generation of pollution packets in colluding attackers.
- We propose the ideas of partial position detection and null keys in short length. *Short null key (SNK)* is one kind of short length null keys. We provide the algorithm to generate SNKs, and prove that SNKs have the complete null space.
- We design SNKs based pollution detection scheme with network coding. The scheme introduces the complete null space of SNKs to improve the security level of *ONKs scheme*. *SNKs scheme* reduce the communication overheads which *ONKs scheme* needs.

The remaining parts are organized as follows. Section 2 briefly introduces the related works. Section 3 describes the notations, the network model, the adversary model used in this paper, together with the relevant theoretical descriptions of network coding and null keys for the elaborating of problem. In Sect. 4, our solutions to the problem, the designed SNKs and *SNKs scheme* are presented. Section 5 analyzes the security level of *SNKs scheme*. Finally, we conclude the paper and give prospects in Sect. 6.

## 2 Related Work

The existing pollution detection schemes can be classified into two categories: the end-to-end error detection or correction schemes [21, 22]; and the on-the-fly detection schemes [13–20].

For end-to-end schemes, errors are detected or corrected only at sink nodes. These schemes cost low overheads, but always suffer from detection hysteresis. Therefore, if there are many attackers in the network, or the network that schemes applied to are in large scale, these schemes can not effectively control the diffusion of pollution. The diffusion will lead to the considerable waste of system bandwidth and computing resources.

In order to effectively reduce the waste of network bandwidth in pollution packets transmission, control the diffusion more timely and reduce the influence of pollution attacks, researchers propose the on-the-fly detection schemes. In these schemes, nodes verify the received packets on-the-fly and discard the detected pollution packets. On-the-fly detection schemes are more practical, because it can control the pollution effectively.

Schemes in the second category include the discrete logarithm assumption based homomorphic schemes [14, 18, 20], the topology based pollution detection and attackers location schemes [16, 17], the traditional cryptography based schemes, the lattice theory based schemes and the linear network coding property based null keys schemes [13, 15, 19].

*ONKs scheme* [19] utilizes the orthogonal property between the ONKs and the data packets to verify the received packets. ONKs are pre-distributed in the network by network coding. If not all the products of the received coded packet and the ONKs of the node equal to zero, the packet will be treated as a pollution packet. The detail analysis of *ONKs scheme* is available in Sect. 3. In literature [23], we have proposed compressed null keys and designed a compressed null keys based pollution detection scheme to reduce the communication overheads. The compressed null keys proposed are distributed in the same way with ONKs in [19].

Specifically, if the null keys distributed in the network are in network coding mode, the nodes in the scheme need to run discrete logarithm assumption based homomorphism function, proposed in [24], to protect null keys from pollution. Kehdi et al. illustrate experimentally that the *ONKs scheme* with hysteric null keys, caused by the execution of homomorphism function, can still effectively control the diffusion of pollution packets. However, the prime field, network coding performing on, must be in big size (generally the size of the field is in 256 bit length) for the consideration of guaranteeing the security of homomorphic function [24]. The big size of field will greatly affect the performance of network coding, e.g., it will greatly increase time in encoding and decoding. Moreover, these schemes are not scalable in the network whose nodes have low computational capability to operate in a large finite field, or execute the high computational complexity homomorphism function.

Therefore, compared with previous works in [19, 23], the SNKs in the *SNKs scheme* are distributed directly and secretly from the source node to every node

to avoid using the high computational complexity homomorphism encryption function. For this reason, the design of *SNKs scheme* is different and more simple to implement.

### 3 Problem Statement

In this section, we firstly describe the symbols and notations used. Secondly, we give the network model and adversary model. Finally, for the convenience of problem explanation, we provide the relevant theoretical descriptions of network coding and null keys.

#### 3.1 Symbols and Notations

The descriptions of the common symbols and notations used in this paper are listed as follows:

- (1) Symbols in bold: the lowercase ones denote vectors or one dimensional arrays, and the uppercase ones denote matrices or sets. (Vector without specification is row vector.)
- (2)  $|\cdot|$ : symbols with absolute value sign denotes the number of elements contained in a vector, one dimensional array, or set.
- (3)  $rank(\cdot)$ : indicates the rank of the matrix.
- (4)  $row(\cdot)$ : indicates the number of rows in the matrix.
- (5)  $col(\cdot)$ : indicates the number of columns in the matrix, vector or array.
- (6)  $\mathbf{x}_i$ : given a matrix  $\mathbf{X}$ ,  $\mathbf{x}_i$  denotes the  $i$ -th row vector in  $\mathbf{X}$ .
- (7)  $\mathbb{Z}_n^+$ : given a positive integer  $n$ ,  $\mathbb{Z}_n^+$  denotes the set containing all the positive integers that are no bigger than  $n$ , i.e.,  $\mathbb{Z}_n^+ = \{x | 1 \leq x \leq n\}$ .
- (8)  $\mathbb{F}_q$ : given a prime  $q$ ,  $\mathbb{F}_q$  denotes the finite field in order  $q$ . It is also known as the prime field.
- (9)  $\mathbb{F}_q^n$ : given a positive integer  $n$ ,  $\mathbb{F}_q^n = \{[\alpha_1, \alpha_2, \dots, \alpha_n] | \alpha_i \in \mathbb{F}_q, 1 \leq i \leq n\}$ . It is the set containing all the  $n$ -dimensional vectors.
- (10)  $span(\cdot)$ : provided that vectors  $\mathbf{x}_1, \mathbf{x}_2, \dots, \mathbf{x}_n \in \mathbb{F}_q^n$  compose a group of vectors,  $span(\mathbf{x}_1, \mathbf{x}_2, \dots, \mathbf{x}_n)$  will denote the vector space spanned by these vectors on  $\mathbb{F}_q$ .
- (11)  $\Pi_{\mathbf{X}}$ : for a vector set  $\mathbf{X}$ ,  $\Pi_{\mathbf{X}}$  denotes the vector space spanned by the vectors in  $\mathbf{X}$ . For a matrix  $\mathbf{X}$ ,  $\Pi_{\mathbf{X}}$  denotes the vector space spanned by the row vectors in  $\mathbf{X}$ .
- (12)  $dim(\cdot)$ : denotes the dimension of vector space.
- (13) Superscript  $T$ : the transpose of a vector or matrix.
- (14)  $\Pi_{\mathbf{X}}^\perp$ : the orthogonal vector space of  $\Pi_{\mathbf{X}}$ , i.e.,  $\Pi_{\mathbf{X}}^\perp = \{t | \mathbf{x} \cdot t^T = 0, \forall \mathbf{x} \in \Pi_{\mathbf{X}}\}$ .
- (15)  $nullity(\cdot)$ : for a vector set  $\mathbf{X}$  or a matrix  $\mathbf{X}$ ,  $nullity(\mathbf{X})$  represents the dimension of the orthogonal vector space of  $\Pi_{\mathbf{X}}$ , e.g.,  $nullity(\mathbf{X}) = dim(\Pi_{\mathbf{X}}^\perp)$ .

### 3.2 Network Model

This paper concentrates on the multicast communication. The source node is responsible for the distribution of data packets and the providing of detection information. Intermediate nodes are responsible for the verification of received packets and the transmission of data packets. Sink node belongs to a special kind of intermediate node. In addition to have the operations of intermediate nodes, sink nodes still have decoding operations to recover the original data source node transmitted.

Source node divides the original data into data blocks in equal length. All data blocks belong to  $\mathbb{F}_q^n$ . Every  $m$  data blocks compose a data block matrix, denoted by  $\mathbf{B}$ ,  $\mathbf{B} = [b_{i,j}]_{m \times n}$ ,  $\forall i \in \{1, \dots, m\}, \forall j \in \{1, \dots, n\}$ . Each row vector in  $\mathbf{B}$  denotes a data block and  $\mathbf{b}_i$  denotes the  $i$ -th row vector in it.

$\mathbf{V}$  denotes the data packet matrix generated according to  $\mathbf{B}$ ,  $\mathbf{V} = [v_{i,j}]_{m \times (m+n)} = [\mathbf{I}, \mathbf{B}]$ , where  $\mathbf{I}$  denotes the  $m \times m$  identity matrix.  $\mathbf{v}_i, \mathbf{v}_i \in$

$\mathbb{F}_q^{m+n}$ , denotes the  $i$ -th row vector in  $\mathbf{V}$ .  $\mathbf{v}_i = \underbrace{[0, 0, \dots, 0, 1, 0, 0, \dots, 0]}_{i-1}, \mathbf{b}_i]$ .

Given a vector  $\mathbf{y}$ , if  $\mathbf{y}$  meets the condition  $\mathbf{y} \in \Pi_{\mathbf{V}}$ ,  $\mathbf{y}$  is a data packet. Although the  $m+n$  length vector whose elements are all equal to zero satisfies the definition of data packets, such packet is useless in decoding and will be discarded by nodes.

### 3.3 Adversary Model

The adversary model established in this paper focuses on pollution attacks. Attackers tamper or forge packets in the network transmission. The scheme designed is above board. Source node during the communication process is believable, and any nodes excluding the source can be potential attackers. Attackers collect information from the data flowing through it, together with the sharing information gained by other attackers for maximally polluting the network.

Given a vector  $\mathbf{y}$ , if  $\mathbf{y}$  meets the conditions  $\mathbf{y} \in \mathbb{F}_q^{m+n}$  and  $\mathbf{y} \notin \Pi_{\mathbf{V}}$  simultaneously,  $\mathbf{y}$  is a pollution packet. If a node cannot detect out the pollution packet according to its existing information, we say that the node is defeated, i.e., the node is polluted.

### 3.4 Network Coding and Null Keys

Firstly, we show several relevant concepts of network coding:

**Linear Network Coding on Prime Field:** Elements in data packets are all represented by elements in  $\mathbb{F}_q$ . Further, the generated data packet during encoding is a linear combination of the node possessed data packets on  $\mathbb{F}_q$ .

**Local Coding Vector:** Also called local encoding kernel, is a vector composed by the coefficients node used to encode the possessed data packets during network coding.

**Global Coding Vector:** Also called global encoding kernel, is a vector composed by the coefficients a data packet used when it is represented by the linear combination of  $\mathbf{v}_1, \mathbf{v}_2, \dots, \mathbf{v}_m$ .

Kehdi et al. propose that nodes can detect pollution through the orthogonal property of ONKs and data packets [19]. Here are the definitions of null keys related concepts. We generalize the definition of null keys defined in [19]:

**Definition 1 (null keys):** The vectors using orthogonal property to verify data packets.

**Definition 2 (ONK):** A kind of null keys. If a vector  $\tau$  meets the condition that  $\tau \in \Pi_V^\perp$ ,  $\tau$  is the ONK of  $\Pi_V$ . If all the elements in  $\tau$  are 0,  $\tau$  is useless, and will be discarded.

**Definition 3 (null space):** The set of vector spaces spanned by a group of null keys in one kind. Each vector space is spanned by the null keys corresponding to the same verification positions in data packets.

**Definition 4 (full null space):** The largest null space spanned by all the null keys of a single kind.

**Definition 5 (complete null space):** A kind of null space which has the feature that no pollution packets can simultaneously pass the verifications of all the null keys in this space.

**Definition 6 (distributed null space):** The null space spanned by all the null keys source node distributed.

Following the Definition 2, it is known that  $\Pi_V^\perp$  is the full null space of ONKs.

Although the *ONKs scheme* functions well in defending against pollution attacks, and enjoys many advantages, the security level of the scheme is not high enough.  $\Pi_F$  denotes the distributed null space in *ONKs scheme*. Since the distributed null space in *ONKs scheme* do not cover the complete null space of ONKs, colluding attackers can easily obtain  $\Pi_F$  through the sharing of ONKs they have. (The reason of the partial cover in *ONKs scheme* is available in the analysis of Theorem 2). Consequently, attacker can easily produce pollution packet  $\mathbf{y}$ ,

$$\mathbf{y} \in \{\mathbf{y} | \mathbf{y} \in \Pi_F^\perp, \text{ and } \mathbf{y} \notin \Pi_V\}. \quad (1)$$

$\mathbf{y}$  can pass the verification of the ONKs any nodes obtained in the network. It results in the vanished security of the system.

## 4 Description of *SNKs Scheme*

In this section, we firstly illustrate our design goal and the corresponding solutions of the low security level in *ONKs scheme*. Secondly, we introduce the concept of partial verification positions set (PVPS), and give the definition and the construction algorithm of SNKs. Finally, we describe the SNKs based pollution detection scheme.

#### 4.1 Design Goals

We first give the sufficient condition that there does not exist pollution packets which can pass the verifications of all the null keys in  $\Pi_{\mathbf{r}}$ .

**Theorem 1.** *Given  $\Pi_{\mathbf{r}} = \Pi_{\mathbf{v}}^{\perp}$ , no pollution packet exists that can simultaneously pass the verifications of all the null keys in  $\Pi_{\mathbf{r}}$ .*

*Proof.* For  $\Pi_{\mathbf{r}} = \Pi_{\mathbf{v}}^{\perp}$ , we have  $\Pi_{\mathbf{v}} = \Pi_{\mathbf{r}}^{\perp}$ . Only vectors belonging to  $\Pi_{\mathbf{r}}^{\perp}$  can pass the verification of the whole null keys in  $\Pi_{\mathbf{r}}$ . It follows that only data packets belong to  $\Pi_{\mathbf{r}}^{\perp}$ . Moreover, pollution packet  $\mathbf{y}$  should meet the condition  $\mathbf{y} \notin \Pi_{\mathbf{v}}$ . Hence the proposition is proved.

However, to satisfy condition  $\Pi_{\mathbf{r}} = \Pi_{\mathbf{v}}^{\perp}$ , the number of linearly independent ONKs distributed by the source node is large. To specify it, we introduce the rank-nullity theorem mentioned in [19]. Here is the description of the theorem:

**Theorem 2** [19]. *For any  $m \times n$  matrix  $\mathbf{A}$ ,  $m \leq n$ , we have*

$$\text{rank}(\mathbf{A}) + \text{nullity}(\mathbf{A}) = n. \quad (2)$$

That is,  $\Pi_{\mathbf{v}}^{\perp}$  is the complete null space of ONKs. For a  $m \times (m+n)$  data packet matrix  $\mathbf{V}$ , the number of linearly independent data packets the source distributing is  $m$ , while the number of linearly independent ONKs needed to construct complete null space is  $n$ . However, practically,  $n$  is much bigger than  $m$  in the consideration of the introduced overheads of global coding vectors in data packet. As the length of the ONKs is equal to that of data packets, the direct distribution of ONKs to compose complete null space will lead to high communication overhead.

In conclusion, the design goals of this paper are to: (1) improve the security level under the assumption that attackers can obtain the distributed null space; (2) reduce the overheads in introducing null keys.

If we divide each data packet into multiple parts and verify each part by using the orthogonal property of null key, the length of null keys will decrease. Moreover, if such short length null keys exist complete null space, then the communication overhead will be smaller than that of ONKs. In this paper, we design such SNKs based scheme with the mentioned two goals satisfied simultaneously.

#### 4.2 Partial Verification Positions Set (PVPS)

Since the verification positions of null keys decide the structure of null keys and the verification method, we introduce the concept of PVPS. PVPS records the positions extracted during data packets verification. Essentially, there is no order relationship among positions. However, for the convenience of easy presentation, positions are recorded in ascending order, and the set of positions is denoted by vector instead of set. The definition of PVPS is as follows:

**Definition 7 (PVPS,  $\mathbf{S}_w$ ):** A vector used to record the positions in data packets, participating in verification. The subscript  $w$  is used to distinguish each position set,  $1 \leq w \leq u$ ;  $u$  is the total number of different  $\mathbf{S}_w$  source node distributing. Given a data packet belonging to  $\mathbb{F}_q^{m+n}$ ,  $S_{w,k}$  denotes the  $k$ -th element in  $\mathbf{S}_w$ ,  $S_{w,k} \in \mathbb{Z}_{m+n}^+$ ,  $1 \leq k \leq |\mathbf{S}_w|$ .

$\bigcup_{w \in \Omega} \mathbf{S}_w$  denotes the union of PVPSs,  $\bigcup_{w \in \Omega} \mathbf{S}_w = \{S_{w,k} | w \in \Omega, 1 \leq k \leq |\mathbf{S}_w|\}$ , where  $\Omega$  denotes the set recording the subscripts of PVPSs.

### 4.3 Construction of SNKs

Before we define SNKs and introduce the construction algorithm of SNKs, we first describe several related concepts:

**The Separated Data Packet  $p_{w,i}$ :** A sub-vector composed by the selected elements in data packet  $v_i$  according to  $\mathbf{S}_w$ .  $p_{w,i,k}$ , denoting the  $k$ -th element in  $p_{w,i}$ , equals to the  $S_{w,k}$ -th element in  $v_i$ .

**Definition 8 (the separated data packet matrix,  $\mathbf{P}_w$ ):**

$\mathbf{P}_w = [p_{w,1}^T, p_{w,2}^T, \dots, p_{w,m}^T]^T = [v_{i,S_{w,k}}]_{m \times |\mathbf{S}_w|}$ , denotes the matrix composed by  $p_{w,i}$ ,  $\forall i \in \{1, \dots, m\}$ , as its row vectors.

**Definition 9 ( $\mathbf{S}_w$  of SNKs):**

- (1)  $\forall w \in \{1, \dots, u - n \bmod u\}$ ,  $\mathbf{S}_w = [1, 2, \dots, m, m + (w - 1)\lfloor n/u \rfloor + 1, m + (w - 1)\lfloor n/u \rfloor + 2, \dots, m + w\lfloor n/u \rfloor]$ ;
- (2)  $\forall w \in \{u - n \bmod u + 1, \dots, u\}$ ,  $\mathbf{S}_w = [1, 2, \dots, m, m + n - (u - w + 1)\lfloor n/u \rfloor + 1, m + n - (u - w + 1)\lfloor n/u \rfloor + 2, \dots, m + n - (u - w)\lfloor n/u \rfloor]$ .

In the formula,  $\lceil \cdot \rceil$  denotes the ceiling of a value, and  $\lfloor \cdot \rfloor$  denotes the floor of a value.

**Definition 10 (SNK):** A kind of null keys. Given a vector  $\varsigma_w$ , if it is the ONK of  $\Pi_{\mathbf{P}_w}$ , it is the SNK of  $\Pi_V$  corresponding to  $\mathbf{S}_w$ . The subscript  $w$  represents that the PVPS of the SNK is  $\mathbf{S}_w$ .

**Full null space of SNKs,  $\Pi_V^{snk}$ :**  $\Pi_V^{snk} = \{\Pi_{\mathbf{P}_w}^\perp | 1 \leq w \leq u\}$ . It is a union of all  $\Pi_{\mathbf{P}_w}^\perp$ .

$\Pi_V^{snk}$  is the complete null space of SNKs (guaranteed by Theorem 4). Algorithm 1 describes the generation of SNKs and data packets. The construction process is similar with that of the ONKs. Note that, when  $u = 1$ , the SNKs generated by Algorithm 1 will be ONKs.

### 4.4 SNKs Scheme

Operations of nodes can be divided into five functional modules in the pollution detection schemes. These modules are **Setup**, **Encode**, **Distribute**, **Verify** and **Decode**. Source node functions the first three modules. Intermediate nodes operate the middle three ones. Sink nodes execute the last four ones. We illustrate the *SNKs scheme* through the description of these modules.



**Algorithm 1.** Generation of SNKs and data packets

---

**Input:** Input  $m \times n$  data block matrix,  $\mathbf{B}$ ;  
size of  $\mathbb{F}_q$ ,  $q$ ;  
number of parts data packets need to be separated into,  $u$ ;  
**Output:** Output  $m \times (m+n)$  data packet matrix,  $\mathbf{V}$ ;  
 $u$  matrices storing SNKs,  $\mathbf{T}_w$ ,  $1 \leq w \leq u$ . The first  $u - n \bmod u$  ones are  $\lfloor n/u \rfloor \times (m + \lfloor n/u \rfloor)$  matrices, and the remained are  $\lceil n/u \rceil \times (m + \lceil n/u \rceil)$  matrices;  
1:  $\mathbf{V} = [\mathbf{I}, \mathbf{B}]$ ;  
2: Set each  $\mathbf{S}_w$  according to Definition 9;  
3: Set each  $\mathbf{P}_w$  according to Definition 8;  
4: **for** ( $w = 1$ ;  $w \leq u$ ;  $w++$ ) **do**  
5: Find the basis of  $\prod \mathbf{P}_w^\perp$  (i.e., the basis of the solution space of homogeneous linear set  $\mathbf{P}_w \mathbf{X} = \mathbf{0}$  on  $\mathbb{F}_q$ );  
6: Set the vectors in the basis as row vectors of  $\mathbf{T}_w$ ;  
7: **end for**

---

**Setup.** Before the communication, source node needs to perform this function module to setup parameters, data packets, SNKs, and so on.

- i. Parameters preparing. Determine the value of prime number  $q$ , positive integer  $m$ ,  $n$ ,  $u$  and  $d$ .  $q$  is the size of  $\mathbb{F}_q$ .  $m = \text{row}(\mathbf{B})$ ,  $n = \text{col}(\mathbf{B})$ , and  $n \gg m$ .  $u$  is the number of parts data packets separated into, where  $1 \leq u \leq n$  and  $u|n$ .  $u|n$  means that  $n$  is divisible by  $u$ .  $d$  is the number of SNKs source node distributing to each node.
- ii. Generation of data packets and SNKs. Perform Algorithm 1 to generate data packet matrix  $\mathbf{V}$  and  $u$  matrices, denoted by  $\mathbf{T}_w$ ,  $1 \leq w \leq u$ , storing SNKs according to the parameters selected and data block matrix  $\mathbf{B}$ .

**Encode.** Encoding can be divided into the encoding of SNKs and the encoding of data packets according to the different objects. Only source node needs to execute the first kind.

- i. Encoding of SNKs. Given  $\mathbf{S}_w$ , source node randomly selects  $|\mathbf{S}_w| - m$  coefficients from  $\mathbb{F}_q$ , denoted by  $\alpha_1, \alpha_2, \dots, \alpha_{|\mathbf{S}_w| - m}$ .  $\mathbf{s}_w$  denotes the encoded SNKs corresponding to  $\mathbf{S}_w$ . We have

$$\mathbf{s}_w = \sum_{i=1}^{|\mathbf{S}_w| - m} \alpha_i \cdot \mathbf{t}_{w,i}, \quad (3)$$

where  $\mathbf{t}_{w,i}$  denotes the  $i$ -th row vector in  $\mathbf{T}_w$ .

- ii. Encoding of data packets. Assuming that a node has  $l$  linearly independent data packets, denoted by  $\mathbf{y}_1, \mathbf{y}_2, \dots, \mathbf{y}_l$ . These data packets have been verified (the verification of data packets is available in the Verify module listed below). The pollution packets which pass the verification will be seemed as data packets. The node randomly selects  $l$  coefficients from  $\mathbb{F}_q$  as local coding vector, denoted by  $\alpha_1, \alpha_2, \dots, \alpha_l$ , to encode.  $\mathbf{y}$  denotes the generated data packet. We have

$$\mathbf{y} = \sum_{i=1}^l \alpha_i \cdot \mathbf{y}_i, \quad (4)$$

**Distribute.** According to the different contents in distribution, it can be divided into the distribution of parameters, SNKs, and the distribution of data packets. Only source node needs to distribute the parameters and SNKs. All nodes in the network need to distribute data packets. The detailed process is as follows:

- i. Distribution of parameters and SNKs. Source node needs to distribute the parameters selected in Setup module, and  $d$  SNKs to every node. During the distribution process, the communication of these two information should provide integrity. The communication of SNKs should further provide confidentiality. Modern cryptography provides many ways to meet these two requirements, e.g., perform HMAC operations at first and then execute AES symmetric encryption to provide integrity and confidentiality at the same time. These ways are not the focus in this paper, so they are brief and uninformative here. The detailed descriptions of the parameters and SNKs are as follows:

---

**Algorithm 2.** The number of SNKs to be transmitted corresponding to each  $\mathbf{S}_w$

---

**Input:** Input total number of SNKs needed,  $d$ ;

number of  $\mathbf{S}_w$ ,  $u$ ;

**Output:** Output  $u$  length vector storing the number of SNKs corresponding to each  $\mathbf{S}_w$ ,  $\boldsymbol{\Upsilon}$ ;

1: **for** ( $w = 1$ ;  $w \leq u$ ;  $w++$ ) **do**

2:    $\Upsilon_w = \lfloor d/u \rfloor$ , where  $\Upsilon_w$  denotes the  $w$ -th element in  $\boldsymbol{\Upsilon}$ ;

3: **end for**

4: randomly select  $d \bmod u$  elements from set  $\mathbb{Z}_u^+$  to form set  $\mathbf{X}$ ;

5: **for** each element, denoted by  $x$ , in  $\mathbf{X}$  **do**

6:    $\Upsilon_x++$ ;

7: **end for**

---

- (1) The parameters are the values of  $q$ ,  $m$ ,  $n$ ,  $u$  and  $d$ .
- (2) Firstly, run Algorithm 2 to determine the number of the SNKs corresponding to each  $\mathbf{S}_w$ . These numbers will be stored in a  $u$  length vector  $\boldsymbol{\Upsilon}$ .  $\Upsilon_w$  denotes the  $w$ -th number in  $\boldsymbol{\Upsilon}$ . Secondly, generate  $\Upsilon_w$  linearly independent SNKs according to each  $\mathbf{S}_w$  via the encoding of SNKs listed in Encode module.

Algorithm 2 implements two functions. First, distributes  $\lfloor d/u \rfloor$  SNKs to each  $\mathbf{S}_w$ . Second, for the remained ones in  $d$  SNKs, randomly choose  $d \bmod u$  PVPSSs from the total  $u$   $\mathbf{S}_w$ . For the selected ones, add  $\Upsilon_w$  with one.

In order to inform the node the  $\mathbf{S}_w$  SNKs corresponding to, the subscripts of the  $\mathbf{S}_w$  to SNKs will also be distributed together.

- ii. Distribution of data packets. Node distributes encoded data packets, generated in the Encode module, to all the downstream nodes of it.

**Verify.** Every node except the source needs to detect the received parameters, SNKs and packets in communication. As source node does not have upstream nodes, it needs not to perform the verification operations.

- i. Verification of parameters and SNKs. Use corresponding methods in modern cryptography.
- ii. Verification of data packets. If and only if a packet passes the detection of all the SNKs the node gains, can the packet be considered to pass the verification. The steps in using a single SNKs, denoted by  $\mathbf{s}_w$ , to verify a packet, denoted by  $\mathbf{y}$ , are as follows:
  - (1) Firstly, select elements from  $\mathbf{y}$  to construct sub-vector  $\mathbf{y}_w$ . The elements selected are indexed by  $\mathbf{S}_w$ .
  - (2) Calculate the value of  $\mathbf{y}_w \cdot \mathbf{s}_w^T$ . If the product is not 0,  $\mathbf{y}$  is a pollution packet.

**Decode.** Sink nodes can decode and receive the original data packet after receiving  $m$  linearly independent correct packets.

Although the *SNKs scheme* described above is performed on prime field, it has no influence for the application of the scheme on Galois field, such as  $\mathbb{GF}(2^n)$ . Since the calculations on  $\mathbb{GF}(2^n)$  are XOR calculations, it accelerates the speed. Moreover, *SNKs scheme* can combine with the scheme in [15]. The split SNKs can further reduce the amount of updated null keys when changes the data block matrix (the details are not the point of this paper).

## 5 Security Analysis

In this section, we firstly analyze the attack way of malicious nodes. Then, the security level of the SNKs based scheme against random position attack are analyzed. We prove that the complete null space of the SNKs equals to its full null space. At last, we calculate the probability of nodes in composing complete null space.

### 5.1 Security Against Random Position Attack

During the communication, any transmitted data maybe tampered by attackers. Yet, as the parameters and the SNKs are distributed secretly to each node separately by the source node, their confidentiality and integrity are guaranteed by the existed cryptographic methods. Their security level is beyond the scope of this paper. In this paper, we mainly discuss the attacks on data packets.

The attacks, in which pollution packets produced according to the SNKs of colluding attackers, are not considered in this paper. Since the null keys belonging to different nodes are concealed from each other, the attackers can hardly speculate the null keys of other regular nodes to gain higher probability of successful attack. Therefore, it launches the random position attack as follows:

**Random Position Attack:** Attackers produce pollution packets by randomly selecting and modifying the elements in  $\mathbb{F}_q$ .

It can be seen that any packets belonging to  $\mathbb{F}_q^{m+n}$  can be produced by random position attack, that is, this way of attack can produce all possible pollution packets. Note that the SNKs corresponding to  $\Pi_V$  are the ONKs corresponding to  $\Pi_{P_w}$ . The following theorem analyzes the security level single node having against random position attack.

**Theorem 3 [23].** *Suppose that a node gets  $t$  SNKs  $t = \sum_{w=1}^u t_w$ ,  $t_w$  is the number of linearly independent SNKs one node got corresponding to  $\Pi_{P_w}$ . Then, the probability that the node is defeated by random position attack is*

$$\frac{q^{m+n-t} - q^m}{q^{m+n}}. \quad (5)$$

Provided that a node gets  $t$  linearly independent ONKs, the probability that the node is defeated by random position attack is also equal to  $\frac{q^{m+n-t} - q^m}{q^{m+n}}$  [19]. Combining it with the conclusion in Theorem 3, we can see that either we use SNKs to verify the data packets or we use ONKs to verify, the ability of null keys to fight against random position attack is same if given the same number of linearly independent null keys.

In addition, it can be deduced from Definition 9 and Theorem 2 that the total number of linearly independent SNKs corresponding to each  $\mathbf{s}_w$  in  $\Pi_V^{snk}$  is  $\sum_{w=1}^u (|\mathbf{s}_w| - m) = n$ . Also the number of linearly independent ONKs in  $\Pi_V^\perp$  is  $n$ . Therefore, in defending against random position attack, SNKs and ONKs have same ability level.

## 5.2 The Complete Null Space of SNKs

In this subsection, we prove that SNKs exists complete null space.

**Theorem 4.** *The complete null space of SNKs is  $\Pi_V^{snk}$ .*

*Proof.* According to Theorem 3, we know that if  $t = n$ , no pollution packets can pass the verification of all these  $n$  SNKs. On the other hand, the null space these  $n$  SNKs composing is  $\Pi_V^{snk}$ . Hence, the proposition is proved.

The proved theorem follows that the amount of SNKs in composing complete null space is smaller than that of ONKs. Since these two kinds of null keys both need  $n$  null keys to compose complete null space on one side, the length of SNKs are shorter than ONKs on the other hand.

Since each node obtains limited number of SNKs, the keys belonging to a single node can hardly composing complete null space. However, as data packets are transmitted by number of nodes, SNKs belonging to these nodes can hierarchies filter the pollution packets. It is obviously that the easier the null keys composing the complete null space for nodes, the more effectual pollution control ability the pollution detection scheme enjoying.

In this part, we mainly discuss the probability of null keys in composing complete null space among nodes. Before the discussion, we firstly introduce the concepts of matrix  $\mathbf{C}_{k \times n}$ , probability  $P_{k,j}$ , and the probability of randomly generating full rank square matrix on  $\mathbb{F}_q$ . Their notations are arranged as follows, and the details are available in [25].

**Matrix  $\mathbf{C}_{k \times n}$ :**  $\mathbf{C}_{k \times n}$  is a  $k \times n$  random matrix on  $\mathbb{F}_q$ .

**Probability  $P_{k,j}$ :**  $P_{k,j} = P\{\text{rank}(\mathbf{C}_{k \times n}) = j\}$ , i.e.,  $P_{k,j}$  is the probability that the rank of matrix  $\mathbf{C}_{k \times n}$  is  $j$ . Provided  $P_{0,0} \triangleq 1$ .

**Theorem 5** [25]. *The probability of randomly generating full rank square matrix on  $\mathbb{F}_q$  is*

$$P_{n,n} = \prod_{0 \leq i < n} (1 - q^{i-n}). \quad (6)$$

Equation (6) is a reflection to the linearly independence of randomly generated vectors to some extend.

**The Verification Positions Covering Data Packets  $n/d$  Times:** Provided  $u|d$ , each node will receive  $n/d$  linearly independent SNKs corresponding to each  $\mathbf{S}_w$ . It is called that the verification positions of these SNKs  $n/d$  times covering data packets.

**Theorem 6.** *Provided that the verification positions cover data packets once time, the probability that the SNKs, belonging to every  $n/u$  nodes, can compose the complete null space is*

$$\left( \prod_{0 \leq i < n/u} (1 - q^{i-n/u}) \right)^u. \quad (7)$$

*Proof.* Since the verification positions once time cover data packets, there are  $n/u$  SNKs corresponding to each  $\mathbf{S}_w$  for every  $n/u$  nodes.

$A_w$ ,  $1 \leq w \leq u$ , denotes the event that  $n/u$  randomly generated SNKs corresponding to  $\mathbf{S}_w$  can compose  $\Pi_{\mathbf{P}_w}^\perp$ .  $P(A_w)$  denotes the probability that event  $A_w$  will occur.  $A$  denotes the event that the SNKs, belonging to every  $n/u$  nodes, can compose the complete null space.  $P(A)$  denotes the probability that event  $A$  will occur. According to the definitions of  $A_w$  and  $A$ , we have

$$P(A) = P(A_1 A_2 \cdots A_u). \quad (8)$$

As SNKs corresponding to each  $\mathbf{S}_w$  are generated by source node via randomly network coding,  $P(A_w)$  equals to the probability of randomly generating full rank  $\mathbf{C}_{(n/u) \times (n/u)}$  on  $\mathbb{F}_q$ , i.e., equals to  $P_{n/u, n/u}$ . Following Eq. (6), we have

$$P(A_w) = P_{n/u, n/u} = \prod_{0 \leq i < n/u} (1 - q^{i-n/u}). \quad (9)$$

Since SNKs, corresponding to different  $\mathbf{S}_w$ , composing  $\Pi_{\mathbf{P}_w}^\perp$  act independently. Hence, events  $A_w$ ,  $1 \leq w \leq u$ , are independent and identically distributed. We have

$$P(A) = P(A_1 A_2 \cdots A_u) = (P_{n/u, n/u})^u = \left( \prod_{0 \leq i < n/u} (1 - q^{i-n/u}) \right)^u.$$

Provided that the verification positions cover data packets once time, every  $n/u$  nodes in the network will receive  $n$  SNKs from source node in total. Equation (7) shows the probability these SNKs composing complete null space. Given  $n = 160$ ,  $u = 10$ ,  $q = 53$  (a very small prime corresponding to the generally 256 bit  $q$  in the schemes that null keys are protected by homomorphism function), the probability, for every  $n/u$  nodes to compose the complete null space, is bigger than 80%. (The value of  $n$ , in regular value range of network coding schemes, has little impact on this probability.)

## 6 Conclusion

This paper concentrates on designing a on-the-fly SNKs based pollution detection scheme. We firstly analyze the ONKs based schemes and show the limitations of these schemes. Then, we generalize the concept of null keys and propose the concept of complete null space. We also propose the ideas of partial position detection, short null keys and provide the algorithm to generate SNKs. We design SNKs based pollution detection scheme with network coding. Through rigorous theoretical proofs, we analyze the security of the *SNKs scheme*. Analysis shows that the security level of our scheme is same with the *ONKs scheme*, and the communication overheads of ours are less than the ONKs scheme to defend against random position attacks.

## References

1. Ahlswede, R., Cai, N., Li, S., Yeung, R.: Network information flow. *IEEE Trans. Inf. Theor. (TIT)* **46**(4), 1204–1216 (2000)
2. Li, S., Yeung, R., Cai, N.: Linear network coding. *IEEE Trans. Inf. Theor.* **49**(2), 371–381 (2003)
3. Gkantsidis, C., Rodriguez, P.: Network coding for large scale content distribution. In: *Proceedings of IEEE International Conference on Computer Communications (INFOCOM)*, pp. 2235–2245 (2005)
4. Cai, N., Yeung, R.: Secure network coding. In: *Proceeding of IEEE International Symposium on Information Theory (ISIT)*, p. 323 (2002)
5. Bhattad, K., Narayanan, K.R.: Weakly secure network coding. In: *Proceeding of the First Workshop on Network Coding, Theory, and Applications (NetCod)*, pp. 1–6 (2005)
6. Wang, J., Wang, J., Lu, K., Xiao, B., Gu, N.: Optimal linear network coding design for secure unicast with multiple streams. In: *Proceedings of IEEE International Conference on Computer Communications*, pp. 1–9 (2010)

7. Wang, J., Wang, J., Lu, K., Qian, Y., Xiao, B., Gu, N.: Optimal design of linear network coding for information theoretically secure unicast. In: Proceedings of IEEE International Conference on Computer Communications, pp. 757–765 (2011)
8. Wang, J., Wang, J., Lu, K., Xiao, B., Gu, N.: Modeling and optimal design of linear network coding for secure unicast with multiple streams. *IEEE Trans. Parallel Distrib. Syst.* **24**(10), 2025–2035 (2013)
9. Wang, J., Lu, K., Wang, J.P., Qiao, C.: Untraceability of mobile devices in wireless mesh networks using linear network coding. In: Proceedings of IEEE International Conference on Computer Communications (INFOCOM mini-conference), pp. 270–274 (2013)
10. Wang, J., Wang, J., Wu, C., Lu, K., Gu, N.: Anonymous communication with network coding against traffic analysis attack. In: Proceedings of IEEE International Conference on Computer Communications, pp. 1008–1016 (2011)
11. Koetter, R., Medard, M.: An algebraic approach to network coding. *IEEE/ACM Trans. Netw. (TON)* **11**(5), 782–795 (2003)
12. Dimakis, A.G., Godfrey, P.B., Wu, Y., Wainwright, M.J., Ramchandran, K.: Network coding for distributed storage systems. In: Proceedings of IEEE International Conference on Computer Communications, pp. 2000–2008 (2007)
13. Zhang, P., Jiang, Y., Lin, C., Yao, H., Wasef, A., Shen, X.S.: Padding for orthogonality: efficient subspace authentication for network coding. In: Proceedings of IEEE International Conference on Computer Communications (INFOCOM), pp. 1026–1034 (2011)
14. Charles, D., Jian, K., Lauter, K.: Signature for network coding. *Inf. Coding Theory* **1**(1), 3–14 (2009)
15. Newell, A., Nita-Rotaru, C.: Split null keys: a null space based defense for pollution attacks in wireless network coding. In: Proceedings of IEEE Communications Society Conference on Sensor, Mesh and Ad Hoc Communications and Networks (SECON), pp. 479–487 (2012)
16. Le, A.: Cooperative defense against pollution attacks in network coding using spacemac. *Communications* **30**(2), 442–449 (2012)
17. Wang, Q., Vu, L., Nahrstedt, K., Khurana, H.: MIS: malicious nodes identification scheme in network-coding-based peer-to-peer streaming. In: Proceedings of IEEE International Conference on Computer Communications, pp. 1–5 (2010)
18. Gkantsidis, C., Rodriguez, P.: Cooperative security for network coding file distribution. In: Proceedings of IEEE International Conference on Computer Communications (INFOCOM), pp. 1–13 (2006)
19. Kehdi, E., Li, B.: Null keys: limiting malicious attacks via null space properties of network coding. In: Proceedings of IEEE International Conference on Computer Communications (INFOCOM), pp. 1224–1232 (2009)
20. Zhao, F., Kalker, T., Medard, M., Han, K.J.: Signatures for content distribution with network coding. In: Proceedings of IEEE International Symposium on Information Theory (ISIT), pp. 24–29 (2007)
21. Yeung, R.W., Cai, N.: Network error correction. *Commun. Inf. Syst.* **6**(1), 19–35 (2006)
22. Koetter, R., Kschischang, F.: Coding for errors and erasures in random network coding. *IEEE Trans. Inf. Theory* **54**(8), 3579–3591 (2008)
23. Wang, J., Wang, J., Zhu, Y., Lu, K.: SNKC: an efficient on-the-fly pollution detection scheme for content distribution with linear network coding. In: Proceedings of International Conference on Embedded and Ubiquitous Computing, pp. 2298–2305 (2013)

24. Krohn, M., Freedman, M., Mazieres, D.: On-the-fly verification of rateless erasure codes for efficient content distribution. In: Proceedings of IEEE Symposium on Security and Privacy, pp. 226–240 (2004)
25. Zhao, Y.: The probability distribution of the matrix rank on  $\mathbb{F}_q$  and the asymptotic properties of the rank. J. Inf. Eng. Inst. **15**(4), 47–52 (1996)



Frontiers in Internet Technologies

Third CCF Internet Conference of China, ICoC 2014,  
Shanghai, China, July 10-11, 2014, Revised Selected  
Papers

Zhang, S.; Xu, K.; Xu, M.; Wu, J.; Wu, C.; Zhong, Y. (Eds.)

2015, XI, 133 p. 57 illus., Softcover

ISBN: 978-3-662-46825-8