A High-Throughput and Low-Complexity Secure Linear Network Coding Protocol

Majid Adeli\(^1\) and Huaping Liu\(^1\)

\(^1\)School of Electrical Engineering and Computer Science, Oregon State University, Corvallis, Oregon, U.S.A.

Abstract—A new scheme providing security against passive attackers in linear network coding is proposed. Throughput efficiency, low algorithm complexity, and high adaptability/applicability are the major design factors that are considered in this security protocol. A bijective permutation map defined over the code field is utilized to generate the randomness that is required for masking the plain information symbols. The input arguments of the permutation map are some of the plain data symbols and one random symbol that is chosen by the source node. It is shown that as long as the attacker does not have access to all the independent channels, he cannot obtain any linear combination of the plain information symbols. Reducing data throughput by only one unit compared to the non-secure network code, and avoiding the use of complex transformations such as cryptographic algorithms or hash functions are the main advantages of the proposed security protocol.

Keywords: Permutation map, secure linear network coding, throughput efficiency.

1. Introduction

Ensuring security and achieving throughput efficiency are two important and usually conflicting needs in most data networks. A common and well-established method of providing any level of data security is to use cryptographic algorithms [1]. When cryptographic approaches are used to secure a connection, many important issues need to be closely considered. Authentication of legitimate users; generation, management and secure distribution of proper keys; and establishment of initial agreements on the encryption/decryption parameters are some of the fundamental subroutines in cryptographic algorithms which often make the communication process complex and somewhat inefficient. Hence, providing security in data networks without going through the additional burden of conventional cryptographic schemes seems to be an appealing approach, and our goal in this paper is to do so. That is, using a light-weight auxiliary function along with a random symbol, a typical linear network code is converted to a snoop-proof code in this article.

Data networks can be generally classified in two categories: a) networks that are based on the traditional store-and-forward data routing algorithms; and b) networks in which network coding is used as the data routing protocol. In network coding based networks, each intermediate node has the ability to process its input data and relay the resultant processed symbols through its outgoing channels [2]. Hence, corresponding to each network node, a function whose input arguments are the node input symbols and its output arguments are the outgoing symbols leaving that node is considered. We call this function the local encoding function. Network codes can be broadly categorized into two main classes: linear network codes and nonlinear network codes. In linear network coding, the local encoding functions are linear, and since they are multi-input multi-output functions, the application of a local encoding function on the input symbols of a given node can be denoted by matrix multiplication. The matrix representing the network coding operation at a given node is called the local encoding matrix. Therefore, each intermediate node in linear network coding simply puts a linear combination of its input symbols on each of its outgoing channels [3]. The linear combination coefficients can be either chosen randomly from the code field by the intermediate node or they may be pre-assigned to each node by a supervisory entity in a centralized manner and based on the overall network topology. The former type of linear network coding is called random linear network coding, while the latter is known as centralized linear network coding [4]. Regarding the code linearity, it is shown that nonlinear network codes generally outperform the linear ones (e.g., [5], [6]); however, linearity makes the analysis and implementation of a network code much simpler while its performance is still acceptable. Therefore, in the literatures, linear network coding is considered as the more practical way of network coding.

1.1 Related Works

Same as any other data routing protocol, linear network coding should provide various aspects of data security such as privacy, integrity, and authentication before it is considered suitably practical for real applications. There are quite a few papers focusing on the subject of network coding security. Generally, one can categorize these works into two main classes: anti-passive-attack network codes and anti-Byzantine-attack network codes. In passive attacks (i.e., eavesdropping), the attacker may only “see” and possibly take a “copy” of the content of the original data packets. However, in Byzantine (active) attacks, the attacker can impersonate himself as a legitimate network
node or he may compromise some of the genuine network nodes to execute his malicious intentions which can be injecting bogus and invalid data into the main data stream, blocking the data flow by not forwarding their received data, or manipulating the received data before forwarding it. Obviously, Byzantine attacks are more destructive than passive attacks and therefore the schemes that counteract this type of attacks in network coding are more complex than their counterparts challenging the passive attackers. Since the focus of this paper is anti-passive-attack schemes in linear network coding, in the following, we will have a brief review on some of the existing works in this particular field of secure network coding.

In [7], in addition to the regular global encoding coefficients, inclusion of an additional set of coding coefficients in each data packet is suggested. These “blocked coefficients” are chosen and encrypted by the source node while they are decrypted and used to decode the received data by the final destinations. This scheme uses a typical symmetric key cryptography algorithm to perform the encryption/decryption procedures while it keeps the linear network coding routines unchanged at the intermediate nodes.

Two schemes that provide weak security for linear network coding are proposed in [8]. They are designed to be bandwidth and throughput efficient as the second algorithm does not require enlarging the code field size or adding redundancy to the data packets.

In [9], a connection between strongly $k$-secure linear network coding and a secret sharing scheme called “strongly secure ramp secret sharing” is developed and based on this relationship, a generalization of strongly $k$-secure linear network coding is established. In this scheme, unlike many other algorithms with similar objectives, by increasing the number of independent wiretapped channels from $k$ to $k+j$ for $0 \leq j < n-k$, the strong security does not break, instead the rate of secure transmission decreases by $j$ symbols. An algorithm to construct such strongly $k$-secure linear network coding and an approach to convert a non-secure network code to a secure one are also proposed in this reference.

The issue of information-theoretic security against passive attackers in linear network coding and under a unicast scenario is considered in [10], where the optimal number of data symbols and the minimum number of noisy symbols in each data packet are derived. Their scheme also constructs a deterministic or random linear network code that achieves the calculated optimal transmission rate.

Use of Homomorphic Encryption Functions (HEFs) for encrypting the linear coding vectors at the source node is suggested in [11]. By utilizing HEF in this scheme, the source node is able to mix up the symbols of a coding vector with that of a plain data using a random permutation. The main idea is that based on a secret key which is only known by the source and the sink nodes, the order of the symbols in each packet (i.e., the symbols in the coding vector and the payload) is permuted. At the receiver side, after decoding the network coded data, the symbols are resorted back into the original order using the homomorphic encryption key in the decryption algorithm.

An interesting approach is outlined in [12] which utilizes MDS codes at the source node to information-theoretically secure the employed linear network code. In this scheme which stems from Ozarow-Wyner wiretap channel (see [13]), before running the linear network code at the source node, a wiretap code that is based on an MDS code is applied to the data. Assuming the eavesdropper has access to at most $k$ linearly independent channels, it is shown that this scheme guarantees information-theoretic security as long as no linear combination of any $k$ or less independent coding vectors lies in the vector space that is spanned by the rows of the MDS parity check matrix. The size of the network coding field and the procedure based on which the proper coding vectors are assigned to the network edges are also discussed in this reference.

The inner and outer capacity bounds for a multi-source multicast network using secure linear network coding are derived in [14]. The proposed quantitative definition of information leakage brings together all the security levels ranging from no security to complete security under one roof. Considering passive attacks as the threat model, the bounds derived in [15] are generalized in this work such that they fit both cases of weak and complete security.

In order to assure perfect security in multi-source multicast linear network coding, a necessary and sufficient condition on the global encoding vectors is derived in [16]. The threat model includes an eavesdropper who has full access to one predetermined subset of the network links at a time and the security goal is to prevent any information leakage. To this end, some randomness should be added to the original meaningful data, and it is shown that the sources of the randomness can be located at some nodes other than the ones that generate the meaningful data. It is essentially shown that there is no restriction on the locational distribution of the source nodes throughout the network. It is also proved that the random symbols can have any statistical distribution. In an earlier version of their work, the case of perfect security in single-source multicast linear network coding is considered in [17]. Their scheme suggests that based on a given feasible linear network code and the set containing all the possible subsets of the wiretapped links, a proper matrix is constructed at the source node. Using this matrix along with the insertion of some random symbols in the message vector, the original linear network code is transformed to a “secure code”. In [18], same authors have shown the optimality of the $r$-secure linear network coding in terms of the throughput. In other words, it is shown that in their $r$-secure linear network code, the number of the meaningful symbols in each packet is maximal. The definition of perfect (complete) security is also expanded
to the notion of imperfect security in which perfect security is a special case.

Security against passive attacks based on the network topology for linear network coding is studied in [19]. A unicast scenario in which the information transmission rate is one symbol per time unit and every node is equipped with a perfect random number generator is considered. Knowing the entire network topology, a spanning tree connecting the receiver to every other network node is established. It is shown that after performing the preprocessing phase, one symbol can be securely sent to the receiver as long as the eavesdropper does not tap into any link on the path connecting the source node to the receiver.

The idea of using maximum-rank-distance (MRD) codes in order to make the two processes of securing and designing a linear network code independent is considered in [20]. The scheme targets passive attacks in linear network coding and provides information theoretic security against wiretappers who have access to any limited-size subset of the network channels. Inspired by the work in [12] and [13], the security is obtained by defining a coset coding scheme over an extension of the network code field. It is shown that defining the maximum-rank-distance code over an extended field relaxes a fundamental independency restriction considered in [12].

Our goal in this paper is to design a security algorithm that is easily applicable to any available linear network code. The proposed scheme has to prevent information leakage to the eavesdropper while it maximally preserves the data throughput and system simplicity. The rest of this article is organized as follows. A short review on the employed notation, definitions, and assumptions is presented in Sec. 2. The problem being addressed is stated in Sec. 3, where the explanation of what the security algorithm is expected to provide as well as the factors that affect the scheme are delivered. Sec. 4 elaborates on the proposed scheme and describes how it satisfies all the requirements specified in Sec. 3. Comparison with the existing security schemes from different viewpoints is provided in Sec. 5 which is followed by a summary of this work in Sec. 6.

2. Notation, Definitions and Assumptions

A general data network is denoted by directed acyclic graph $G(V, E)$, where $V$ is the set of network nodes and $E$ is the set containing all the network edges (also known as channels or links). The sets $S \subset V$ and $R \subset V$, where $S \cap R = \emptyset$, denote the sets of source and sink nodes, respectively. Let $m = (x_1, x_2, \ldots, x_n)^T$ be the original information vector, in which $x_i$ is a plain information symbol that belongs to the code field $GF(q)$ with $q$ being a prime power. Vector $m$ which contains $n$ i.i.d. plain symbols is generated by the source nodes and shall be entirely received by each sink node. Note that $|S| \leq n$, which implies that each source node in $S$ generates at least one information symbol because otherwise it would be redundant and will be eliminated from the network. Since the proposed scheme does not depend on the locational distribution of the nodes in set $S$, for simplicity, all the source nodes are put together and the entire set $S$ is considered as a single node called the source node. In networks that use linear network coding as the routing protocol, row vector $v^l = (v^l_1, v^l_2, \ldots, v^l_n) \in GF(q^n)$, called (global) coding vector, is assigned to network edge $l \in E$. [3]. Therefore, the symbol flowing on each edge can be stated as the inner product of the corresponding coding vector and vector $m$. Each edge can carry only one symbol per time unit. The data transmission is free of noise, fading and any type of interference. Such distortions exist in practical networks; however, they are assumed to be mitigated by other physical layer processes (e.g., channel coding) before recovering the network-coded data starts.

According to Max-Flow-Min-Cut theorem [21], in networks using network coding, each sink node should have access to $n$ independent channels (or equivalently $n$ independent coding vectors) in order to be able to decode the entire information vector $m$ [2]. This condition (i.e., feasibility of a linear network code) is assumed to be satisfied in this paper. Two channels are considered statistically independent if their corresponding global coding vectors are independent.

3. Problem Statement and Threat Model

The goal is to securely send the information vector $m$ from the source node $S$ to each element of set $R$ via network $G$ with minimum overhead and low complexity. The security requirement mandates the algorithm to prevent any information leakage to the attacker. That is, the attacker who has access to at most $n-1$ independent channels should not be able to obtain any nonzero linear combination of the plain information symbols.

The required level of security should be achieved without using any cryptographic scheme. Avoiding conventional cryptographic algorithms considerably simplifies the transmission process. Additionally, we want to minimize the security overhead, which means that during each time slot, the scheme has to deliver as many meaningful information symbols to the sink nodes as possible. Note that since there are originally $n$ plain information symbols in vector $m$, the maximum throughput determined by Max-Flow-Min-Cut theorem in the non-secure case is $n$ symbols per transmission.

4. The Proposed Solution

According to the discussion in Sec. 3, the three main requirements for the security protocol are i) no information leakage to the wiretapper, as long as the number of tapped
independent channels is less than \( n \); ii) minimum inflicted security overhead (maximum data throughput); and iii) low algorithm complexity. In the following, the proposed security scheme is described and it is shown that it meets all the design requirements.

### 4.1 Algorithm Description

In order to prevent the attacker from obtaining any linear combination of the components in information vector \( m = (x_1, x_2, \cdots, x_n)^T \), we substitute \( x_k \) in \( m \) with \( \tilde{x}_k \) based on the following formulation.

$$
\tilde{x}_k = x_k + \sum_{j=0}^{k-1} f^{(k-j)} (a + x_j) \quad k = 1, \cdots, n - 1, \quad (1a)
$$

$$
\tilde{x}_n = a + f \left( \sum_{i=1}^{n-1} \tilde{x}_i \right) \quad (1b)
$$

In (1), \( x_0 = 0 \) and \( f^{(m)}(\cdot) \) denotes \( m \) times composition of permutation function \( f \) with itself; for example \( f^{(3)}(\cdot) = f(f(f(\cdot))) \). Also, symbol \( a \in GF(q) \) is a uniformly distributed random symbol chosen by the source node. In Sec. 4.4, two different implementation approaches for permutation function \( f \) are explained. Hence, the source node sends out the secured information vector \( \tilde{m} = (\tilde{x}_1, \tilde{x}_2, \cdots, \tilde{x}_n) \) containing \( n - 1 \) hidden meaningful information symbols.

The concealed information is recovered at the sink node in a step-by-step fashion. That is, after a sink node decodes its received symbols and obtains vector \( \tilde{m} \) (network de-coding), it executes the following Extraction Algorithm:

1. **a)** Recover the key (the noisy symbol \( a \)) by computing
   
   \[ a = \tilde{x}_n - f \left( \sum_{i=1}^{n-1} \tilde{x}_i \right). \]

2. **b)** Set \( x_0 = 0 \) and \( i = 1 \).

3. **c)** Recover the meaningful symbol \( x_i \) by computing
   
   \[ x_i = \tilde{x}_i - \sum_{j=0}^{i-1} f^{(i-j)} (a + x_j). \]

4. **d)** While \( i < n - 1 \), increase \( i \) by 1 and repeat (c); otherwise exit the algorithm.

By following this recursive algorithm, all the meaningful information symbols are extracted from vector \( \tilde{m} \). Note that every operation such as addition, is defined over the code field \( GF(q) \).

### 4.2 Security Analysis

To verify the algorithm security, let \( r_k \triangleq \sum_{j=0}^{k-1} f^{(k-j)} (a + x_j) \) for every \( 1 \leq k \leq n - 1 \) and \( r_n \triangleq \tilde{x}_n \). Considering the facts that 1) random symbol \( a \) has uniform distribution over \( GF(q) \), 2) function \( f(\cdot) \) is bijective, and 3) the plain information symbols are i.i.d with arbitrary distribution over the code field; one can infer that every argument \( a + x_j \) \( (0 \leq j \leq k - 1) \) in \( r_k \) as well as symbol \( r_n \) is uniformly distributed over the code field. Moreover, for every two distinct plain information symbols \( x_i \) and \( x_j \), there is

\[
\forall d \geq 1 : \Pr \left( f^{(d)} (a + x) \right) = \Pr (a + x), \quad (2a)
\]

\[
\Pr(a + x_i) = \frac{1}{q}, \quad (2b)
\]

\[
\Pr(a + x_i, a + x_j) = \Pr(a + x_i) \Pr(a + x_j). \quad (2c)
\]

Hence, \( \forall 1 \leq k \leq n, r_k \) is a summation of some mutually independent and uniformly distributed random variables. That is,

\[
\Pr(r_k) = \frac{1}{q}. \quad (3)
\]

For every plain information symbol \( x_k \), the corresponding \( r_k \) (called masking symbol) depends solely on symbol \( a \) and all the plain information symbols that precede \( x_k \); hence, \( x_k \) and \( r_k \) for every \( 1 \leq k \leq n - 1 \) are independent.

This fact along with the result in (3) indicates that each original information symbol \( x_k \) is added to an independent and uniformly distributed random symbol in the proposed scheme. Adding the case of \( k = n \) in which \( \Pr(\tilde{x}_k = r_n) = \frac{1}{q} \), this fundamental noise-masking procedure is summarized below.

\[
\Pr(\tilde{x}_k) = \Pr(x_k + r_k) = \frac{1}{q} \quad 1 \leq k \leq n - 1, \quad (4a)
\]

\[
\Pr(\tilde{x}_n) = \frac{1}{q}. \quad (4b)
\]

The results in (4) indicate that obtaining any individual entry of vector \( \tilde{m} \) does not give the wiretapper anything but uniform noise.

Assume the attacker has wiretapped \( k < n \) independent channels and by taking a linear combination of the tapped symbols, he is trying to attain a linear combination of the plain information symbols. Let each row of matrix \( B_{k \times n} \) be a coding vector corresponding to one of the tapped channels. Let row vector \( c \) contain the arbitrary linear combination coefficients chosen by the attacker. The attack is modeled as

\[
c \cdot B \cdot \tilde{m} = [c_1', c_2', \cdots, c_{n-1}'] \cdot (x_1, x_2, \cdots, x_{n-1})^T + [c_1', c_2', \cdots, c_n'] \cdot (r_1, r_2, \cdots, r_n)^T
\]

\[
= L(m) + L(r). \quad (5)
\]

In (5), vector \( c' \) represents the arbitrary linear combination of the tapped independent coding vectors. As shown above, this linear combination can be decomposed into two parts: a) the arbitrary linear combination of the plain information symbols, denoted by \( L(m) \); and b) the corresponding linear combination of the noisy symbols, shown by \( L(r) \). To achieve the required level of security, it is sufficient to have \( \Pr(L(r)) = \frac{1}{q} \), which means every arbitrary linear combination of the plain symbols is still masked by a noise component (called \( L(r) \)) with uniform distribution over the
code field. To verify this, let us elaborate on vector $c'$. We know $L(r) = \sum_{i=1}^{n} c'_i r_i$. For every $c'_i \neq 0$ one has $\Pr(c'_i r_i | c'_i) = \Pr(r_i) = \frac{1}{q}$. Therefore from the wiretapper’s perspective, $L(r)$ is a summation of some mutually independent and uniformly distributed random variables. This results in uniform distribution for $L(r)$, i.e., $\Pr(L(r)) = \frac{1}{q}$. Hence, any arbitrary linear combination of the plain information symbols, $L(m)$, is masked with a uniformly distributed random symbol, $L(r)$, and therefore the security requirement is satisfied. In other words, neither any individual entry nor any linear combination of the entries of $\tilde{m}$ reveals any information to the attacker.

### 4.3 Throughput Efficiency

Substitution of only one plain information symbol in the original information vector $(m)$ with one noisy symbol $(a)$ is all the throughput reduction paid to ensure the data security in our scheme. This means instead of the maximum value of $n$ symbols per time unit, the source node delivers $n - 1$ meaningful information symbols to each sink node during each time slot. As described, conventional cryptographic security schemes do not require this substitution but they have much higher computational and hardware complexity. In our scheme, symbol $a$ can be interpreted as the key which is securely hidden in the transmitted information vector $\tilde{m}$; thus, each data packet carries its own key. This considerably simplifies the communication since the need for a trusted third party to execute the entire key management process is eliminated. It also enables the source node to change the key as frequent as it generates new packets (new information vectors) without imposing additional complexity to the system. In Sec. 5, we will discuss the throughput advantages of this scheme in greater detail.

### 4.4 Algorithm Complexity

Utilizing a light-weight permutation map instead of employing computationally expensive cryptographic algorithms in order to ensure data security considerably simplifies the transmission procedure. The permutation can be implemented as a look-up table with two rows and $q$ columns, where $q$ is the code field size. The input-output assignment is chosen randomly and any assignment is acceptable except identity relationship (i.e., $f(x) \neq x$). An example of a possible permutation map for $q = 11$ is shown in the following table.

Table 1: A Permutation Map Over GF(11) Realized as A Look-Up Table

<table>
<thead>
<tr>
<th>$x$</th>
<th>0</th>
<th>1</th>
<th>...</th>
<th>10</th>
</tr>
</thead>
<tbody>
<tr>
<td>$f(x)$</td>
<td>3</td>
<td>8</td>
<td>...</td>
<td>2</td>
</tr>
</tbody>
</table>

Another simple way of implementing the permutation map is an affine function with two constant parameters $s$ and $t$ where $s, t \in GF(q)$ and $s \neq 0$. To exclude the identity permutation case, we impose that if $t = 0$, then $s \neq 0, 1$. In other words, the permutation function $f(\cdot)$ is defined as

$$f : GF(q) \rightarrow GF(q)$$

$$u \mapsto su + t. \quad (6)$$

with the above conditions on $s$ and $t$. Note that the security does not depend on the privacy of function $f(\cdot)$, and therefore all the parties including the attacker may know this function.

### 5. Comparison with The Existing Schemes

As mentioned, in networks with conventional store-and-forward routing protocols, typical cryptographic approaches such as different modes of block or stream ciphers are usually used to encrypt the plain information at the source node and decrypt it at the authorized sink nodes. This requires execution of a series of routines such as key management (including generating, securely distributing and regularly verifying/notarizing appropriate keys), user authentication, security policy management (e.g., establishing initial agreements on the security parameters between the senders and receivers) as well as running encryption/decryption procedures [1]. These inhibiting factors are time consuming and use network resources such as bandwidth, power and hardware. Considering the fact that the proposed scheme eliminates these issues along with providing more freedom and flexibility in terms of security management and key renewal, the benefits of this protocol significantly outweigh the expense of reducing the throughput from $n$ to $n - 1$ meaningful symbols per time unit.

To summarize, the main advantages of this scheme over the cryptographic based approaches are as follows. a) It eliminates the need for the entire key management routines and enables the source node to generate its keys without supervision of a trusted third party. This can even yield to a higher security level since the number of parties involved in establishing a secure connection is reduced. b) The key information used to secure the content of a data packet is covertly embedded in the corresponding data packet; therefore, each packet safely and independently carries its own key information indicating that there is no need for sending the keys to the destinations in advance through a secret channel. c) It offers the capability of dedicating different keys to different data packets and changing keys as frequent as the source node generates new data packets. This is achieved without imposing additional delay or complexity to the system, or using network resources. d) It does not need a complex encryption/decryption module; instead, only a simple non-identity permutation map is used to generate the required randomness.
There are quite a few papers discussing security algorithms tailored for linear network coding, some of which targeting the same security goals as considered in this paper (e.g., [7], [9], [11], [17], [16], [12], [22], [20], [18], [10], [19]). These works assume that the attacker has access to at most \( k < n \) independent edges. In order to prevent him from obtaining any linear combination of the plain information symbols, it is shown that \( r \geq k \) symbols in vector \( \mathbf{m} \) should be substituted with \( r \) uniformly distributed random symbols that are chosen by the source node. Therefore, in the secured information vector \( \mathbf{\tilde{m}} = (x_1, x_2, \ldots, x_{(n-r)}, z_1, z_2, \ldots, z_r) \), the last \( r \) symbols are the substituted noisy symbols. After this step, the source node runs an invertible linear transformation on \( \mathbf{\tilde{m}} \) and then sends out the resultant vector through the network. By applying the final transformation on \( \mathbf{\tilde{m}} \), the source node avoids putting plain information symbols on some of its output edges. The linear transformation can be modeled by matrix multiplication. This way on each output edge of the source node there is a linear combination of the plain information symbols and some noisy symbols, yielding a uniformly distributed random symbol.

Let us denote this final transformation by matrix \( \mathbf{T}_{n \times n} = [Q_{n \times (n-r)}][P_{n \times r}] \). To retain the security, it is necessary to include at least one noisy symbol in each outgoing linear combination. Therefore, the submatrix \( P_{n \times r} \) should not have any all-zero row. This condition excludes some of the possible choices for matrix \( \mathbf{T} \) and requires the source node to take into account this constraint when it is constructing this matrix. Assuming the attacker has access to exactly \( k \) independent coding vectors, then the network is secure only if the attacker is unable to obtain any linear combination of the meaningful information symbols by linearly combining his wiretapped symbols. Let \( \mathbf{C}_{k \times n} \) be the matrix whose rows are the \( k \) independent wiretapped coding vectors and let \( \mathbf{A}_{1 \times k} \) be the vector containing the linear combination coefficients chosen by the attacker. The attack may be modeled as

\[
\mathbf{A} \cdot \mathbf{C} \cdot \mathbf{\tilde{m}} = \mathbf{A} \times [\Gamma_{k \times (n-r)}] \Delta_{k \times r} \times (x_1, x_2, \ldots, x_{(n-r)}, z_1, z_2, \ldots, z_r)^T = d
\]  

For any arbitrary \( \mathbf{A} \), in order to have at least one noisy symbol (i.e., \( z \)) participating in \( d \), the rows of submatrix \( \Delta_{k \times r} \) should be linearly independent and it requires \( r \geq k \). The independency condition on the rows of \( \Delta_{k \times r} \) imposes some additional limits on the assignment of proper coding vectors to the network edges. This makes the design and management of linear network codes, especially in the case of random linear network codes, more complex. On the other hand, the aforementioned existing approaches require at least \( k \) symbols to be substituted in the original information vector \( \mathbf{m} \), which means the throughput is at least decreased by \( k \) units.

Compared to the existing security schemes, the proposed approach has the following advantages. (a) It significantly improves the data throughput. With the proposed scheme, in each information vector (data packet) only one meaningful symbol is substituted by noise, as opposed to at least \( k \) substitutions in the existing counterparts. (b) There is literally no limitation on the coding vector selection in this algorithm while in previous works some of the coding vectors are improper and should be avoided in order to retain the required level of security. (c) It relaxes a fundamental assumption on the allowed number of wiretapped independent channels. In this scheme, the security is impenetrable as long as the attacker has not acquired all the \( n \) independent coding vectors while in the other works the security is broken as soon as the attacker acquires more than \( k \) independent channels. (d) Throughput is utterly independent of the number of the attacked channels as opposed to the previous schemes in which throughput is highly dependent on the attacker’s ability such that by increasing the value of \( k \) from \( k_1 \) to \( k_2 \), in order to protect the security, the throughput at least drops from \( n - k_1 \) to \( n - k_2 \) symbols per transmission. In the presented protocol, as long as the attacker does not have access to all the independent channels, the throughput is equal to \( n - 1 \). (e) In this scheme, the need for deliberately designing, sharing, and applying an invertible linear transformation to the message vector prior sending it through the network is removed, and that is the case because every entry in vector \( \mathbf{\tilde{m}} \) has already uniform distribution and maximum entropy. (f) The security concern existing in [23] is eliminated. In order to fully protect the key information (i.e., the noisy symbol \( a \)), the scheme in [23] imposes a constraint on the admissible coding vector set. In the scheme described in this paper, by concealing the key information in the secured information vector (\( \mathbf{\tilde{m}} \)), the restriction on the coding vector selection space is removed. Additionally, the use of simple permutation function as a replacement for the complex hash functions makes this algorithm much simpler than the scheme in [23].

6. Conclusion

An algorithm for securing data transmission in linear network coding is proposed. The main idea of this scheme is to mask each plain information symbol by a noisy symbol which has uniform distribution over the code field. To generate the required noisy symbols, a recursive algorithm in which a simple permutation function is the main building block is utilized. Only one noisy symbol along with some of the original information symbols is employed to produce the masking symbols. Two possible implementation approaches for the permutation function are also discussed. The comparison of the proposed approach with the existing schemes in terms of throughput efficiency and complexity demonstrates the capabilities and benefits of the projected security protocol.
References