110/2022-C/GE

PEOPLE'S DEMOCRATIC REPUBLIC OF ALGERIA

UNIVERSITY OF SCIENCES AND TECHNOLOGY HOUARI BOUMEDIENE

Faculty of Electrical Engineering

A thesis submitted in fulfillment of the requirements

for the degree of Doctor

In Telecommunication

On Telecommunication and Information Processing

By : Mohamed Amine BRAHIMI

TITLE :

Investigation of Network Coding from Security and Error Control Perspectives

---

Defended publicly on 15/10/2022, in front of the jury composed of:

Ms	Lamya FERGANI	Professor at USTHB	President
Ms	Fatiha MERAZKA	Professor at USTHB	Supervisor
Ms	Gunes Karabulut KURT	Asso. Professor at Polytech Montréal Canada	Co-Supervisor
Ms	Amina SERIR	Professor at USTHB	Examinator
Mr	Abdelhakim DAHIMENE	Professor at UMBB	Examinator
Mr	Mustapha BENSSALAH	MCA at EMP Algiers	Examinator

Acknowledgements

Now that I finished typing my thesis, I feel a deep need to express my gratitude to the persons without whom, I would certainly not have reached this point with probability reaching 1.

First of all, I shall never forget the guidance, kindness as well as all kinds of help I received from my supervisor Prof. MERAZKA, the professor who taught me a great deal of things, and who believed in me far more than I believed in myself. She is a person to whom I am forever indebted. I must also thank Prof. KURT for her insightful remarks during the publication process.

Special thanks must also be given to Prof. DAHIMENE, who has accepted to be part of the jury. He is a great teacher who taught me how to think of probability as a measure by introducing me to $\sigma-$algebras.

I must extend my gratitude to Prof. FERGANI, who was very nice with all the PhD students at the department of Telecommunications as well as to both Prof. SERIR and Prof. BENSSALEH for accepting to be part of the jury and evaluate my work.

To my family, especially my life partner Zahra and my dear Djanet, I must express all love, gratitude and thanks for all their patience and support during the whole process.

Finally, I would like to pay homage to the memory of Evariste Galois, whose fields made this work possible, and whose elegant theory of the insolvability of the general quintic has always been my favorite mathematical theory.

Contents

List of Figures

List of Tables

Acronyms

AES	Advanced Encryption Standard
ARQ	Automatic Repeat Request
CDC	Constant Dimension Code
KDC	Key Distribution Center
KK	Kotter and Kschichang
LNC	Linear Network Coding
MDC	Mixed Dimension Code
MDS	Maximum Distance Separable
ML	Maximum Likelihood
MRD	Maximum Rank Metric
NC	Network Coding
PRNG	Pseudo Random Number Generator
RDIP	Relative Dimension Intersection Profile
RGRW	Relative Generalized Rank Weight
RLNC	Random Linear Network Coding
RREF	Reduced Row Echelon Form
SCS	Subspace Coding Strategy
SRLNC	Secure Random Linear Network Coding
SPOC	Secure Practical netwOrk Coding
TRNG	True Random Number Generator
USNC	Universal Secure Network Coding
XOR	Exclusive OR

List of Symbols

$\lvert \cdot \rvert$	The cardinality of a set of elements
C_e	The eavesdropping capacity of a given wiretapper
C_m	The multicast capacity of a given source
$dim(\cdot)$	The dimension of a given vector space
$\mathbb{F}_q$	The finite field with q elements.
$\mathbb{F}_q^n$	The n^{th}-dimensional vector space over $\mathbb{F}_q$
$\mathbb{F}_q^{m \times n}$	The set of all $m \times n$ matrices over $\mathbb{F}_q$
$G^{(n_i)}$	The global encoding matrix at a given intermediate node n_i
G_d	The global encoding matrix at the sink d
G_k	The gaussian coefficient correponding to $\mathcal{G}(k,n)$
$\mathcal{G}(k,n)$	The k-dimensional Grassmanian over the ambient space $\mathbb{F}_q^n$
$L^{(n_i)}$	The local encoding matrix at a given intermediate node n_i
$[\boldsymbol{M}]$	A matrix M
$\langle \boldsymbol{M} \rangle$	The row space of a matrix $\boldsymbol{M}$
$\mathcal{P}(n)$	The projective space
$P_{in}^{(n_i)}$	The input matrix of a given intermediate node n_i
$P_{out}^{(n_i)}$	The output matrix of a given intermediate node n_i
$rank(\cdot)$	The rank of a given matrix
$\begin{bmatrix} n \\ k \end{bmatrix}_q$	Another notation for the gaussian coefficient

Abstract

Network coding is a transmission paradigm that relies on packet encoding at a subset of the intermediate nodes that are usually referred to as the encoding intermediate nodes to differentiate them from their forwarding counterparts. Encoding is generally modelled as a form of packet mixing, in which the received packets at an encoding intermediate node are combined following a predefined bijective mapping to form an output packet. In the literature of NC, Random Linear Network Coding (RLNC) is the NC scheme that is considered more practical. This scheme opts for linear transformations as encoding operations along with a random selection of the encoding coefficients allowing for possible deployment in noncoherent situations. While packet mixing has allowed for many advantages such as throughput enhancement and security against wiretap attacks, it induced another set of challenges, from which error propagation is the crucial one.

Packet mixing combined with the error propagation problem as well as the nature of RLNC that assumes no prior knowledge of network topology make the use of a pre-designed classical correcting code insufficient for this system Those problems were addressed via the introduction of a new family of error correcting codes referred to as subspace codes. Those codes have been proposed to be used as outer codes in RLNC based networks to provide error correction capability in the network. In subspace codes, codewords are not vectors but rather vector spaces taken from an ambient vector space over a finite field $\mathbb{F}_q$ where q is a prime power. In this regard, information will not be treated packetwise (vectorwise) but rather as a whole subspace injected in the network.

While the main work in this thesis lies in the investigation of error control in RLNC via the use of subspace coding, our obtained results span both error control and other areas in RLNC such as its encoding complexity as well as data security. In error control, we provided a comparative analysis between subspace codes and rank metric codes, from which a set of remarks concerning their differences and uses were obtained. A scheme for secure data transmission for subspace codes has also been introduced as a solution for both error correction and data security in RLNC networks where we have opted for a randomized codeword selection to induce an ambiguity about the subspace code used for transmission.

As for security in RLNC, three schemes have been proposed to provide data transmission making use of the intrinsic security of RLNC in addition to a set of other steps aimed at

increasing the computational as well as the information-theoretic security of RLNC. A final contribution regarding the encoding complexity of RLNC has been provided, in which we have minimized the number of required multiplicative operations in RLNC.

General introduction

Since the advent of public internet, network theory has become a central part in all modern communication systems. Broadly speaking, a network is a set of communicating elements along with another set of interconnections providing paths for message transmission. Mathematically, a network may be modelled as a graph G with a set V of nodes and a set E of edges. A node in a network is essentially a communicating entity and the edges are the channels through which messages can be exchanged between those entities.

In networking, one of the main problems that we are faced with is how to efficiently convey or route traffic from a given source to some destination throughout a network. We call the rules that are made to specify how this endeavor is carried out routing protocols. The traditional way of treating routed data in communication networks is to consider it as a commodity. In other words, packets are supposed to stay intact throughout the network. The routers separating the source from destination will just receive the packets on their ingoing interfaces and forward them to the next hops in the network via their outgoing interfaces. In this system, a received packet that is not an exact copy of its original version is considered corrupt and should be retransmitted. Treating packets as a commodity has remained the de jure and de facto procedure until the very beginning of the current century where Yeung et al [1] have proposed a novel routing system called Network Coding (NC). In this new paradigm, packets can be mixed, coded or transformed into new packets and as long as the used transformation is bijective, the source packets can always be recovered. This simple yet insightful idea has resulted in remarkable outcomes in terms of throughput, security, latency ... etc.

In a secure and error free environment, NC will be the optimal solution for data transmission in terms of throughput. However, in real world scenarios, network errors are inevitable and by mixing packets, NC will provide a ground for error propagation problems and eventually loss of information. Error control solutions are therefore crucial to consider before the deployment of any NC-based transmission system.

In practice, Random Linear Network Coding (RLNC) is the NC scheme that is deemed practical given that the encoding operations are just linear transformations using randomly selected coefficients. Since packets get combined, using classical error correcting solutions is not advisable with RLNC. This problem has been solved using a new class of error correcting

codes referred to as subspace codes.

This thesis focuses on RLNC from an error control perspective as well as its applications to data security. In this regard, We provide a solution for secure transmission of subspace coded data.

Outline

The remainder of the thesis is organized as follows. Chapter 1 provides a general overview on Network Coding (NC), in which the basic theory of this paradigm is reviewed with a focus on Random Linear Network Coding (RLNC). Chapter 2 tackles the theory of network error correction codes for RLNC via the use of subspaces, where we review some mathematical preliminaries on abstract algebra and finite field theory followed by a presentation of the main results in the literature of subspace codes. Chapter 3 is an investigation of secure RLNC schemes followed by a proposition of a scheme that shows a possible scenario of using subspace codes to provide security for RLNC schemes while maintaining error correction. The last chapter will take the form of a general conclusion on the work and results obtained in this thesis with some perspectives on our future work.

Contributions

This thesis is an attempt to tackle the problem of error control in Random Linear Network coding. In the thesis, we try to investigate the main work that has been introduced in the literature as well as to add some contributions to both the area of RLNC and subspace coding. Our contributions and results span three main areas:

Random Linear Network Coding

In this area, we have introduced a novel approach to minimize the encoding complexity resulting from the multiplicative operations required by the encoding intermediate nodes. The idea was to decide on the usefulness of the multiplicative operation in terms of its effects on the decoding failure probability at the network. This approach was based on the fact that uniform distributions are preserved under auto-convolution on finite multiplicative groups.

Security in Random Linear Network Coding

In the area of secure RLNC, three schemes have been provided to preserve data confidentiality against wiretapping attacks. The three schemes rely on the idea of securing the encoding matrix in RLNC in addition to some other steps that aim to increase the computational security of the system.

Error Control in Random Linear Network Coding

A comparative analysis between rank-metric codes and lifted Rank-metric codes, which are a family of subspace codes that are based on Rank-Metric codes, has been carried out in order to get insights of the main differences between the two code families as well as the scenarios where the two codes may replace each other.

A scheme that provides both error control and security against wiretap attacks has also been introduced for subspace codes. This scheme is based on randomizing the dimension of the transmitted codeword based on a subspace code selection strategy to increase the search cost of a wiretapper while ensuring data integrity.

List of publications

Journal Papers

- 1- Mohamed Amine Brahimi, Fatiha Merazka, Data confidentiality-preserving schemes for random linear network coding-capable networks, Journal of Information Security and Applications, Volume 66:103136, 2022.

- 2- Mohamed Amine Brahimi, Fatiha Merazka, Gunes Karabulut Kurt, Secure network coding for data encoded using subspace codes, Physical Communication, Volume 48:101408, 2021.

Conference Papers

- 1-M. A. Brahimi and F. Merazka, "A secure algorithm for Random Linear Network Coding," 2020 IEEE 5th International Symposium on Smart and Wireless Systems within the Conferences on Intelligent Data Acquisition and Advanced Computing Systems (IDAACS-SWS), 2020, pp. 1-4

- 2-M. A. Brahimi and F. Merazka, "On Reducing the Encoding Complexity of Random Linear Network Coding," 2020 International Conference on Electrical Engineering (ICEE), 2020, pp. 1-5

Chapter 1

Network Coding

1.1 Introduction

Network Coding[1] is the transmission scheme upon which our work is based. Therefore, this chapter will constitute a general discussion of this paradigm with a focus on Random Linear Network Coding[2], given the fact that most of the work done on NC in the literature is about RLNC. In this regard, in this thesis, we will be using the terms "Network Coding" and "Random Linear Network Coding" interchangeably. As for our objective in this chapter, we try to provide a glimpse on the necessary background that we need, to advance into the main subject of this thesis.

1.2 Network Coding

Before we tackle the subject of Network Coding (NC), It is crucial to clarify some possible ambiguities that may arise in the treatment of the subject. One of those ambiguities engulfs the word “coding”. In communication theory, coding usually refers to either source coding or channel coding. Source coding is the suppression of unnecessary or uncontrolled redundancy from the source data before transmission. It is usually labelled as data compression in real-world applications. Channel coding, on the other side, is the process of adding controlled redundancy to the transmitted data as a way to maximize the probability of data recovery in the presence of channel errors. In the literature of Network Coding, the word “coding” is ascribed a different meaning. It is the mapping of a set of input packets to another set of output packets either deterministically using a set of rules or randomly following some probability distribution. Using this new connotation of the word “coding”, we can define Network Coding as follows.

Definition 1. *Given a network N, Network Coding (NC) is a bijective mapping of a set of input packets onto another set of output packets. It can be carried out at intermediate nodes as well as the source node.*

Before the idea of Network Coding, packets were treated as a commodity. The only difference between them and cars in vehicular networks is their ability to be replicated. Rigorously speaking, in conventional routing protocols, an intermediate node may only store, forward or replicate packets. Each transmitted packet’s payload has to stay intact for the whole transmission period from source to destination, otherwise it would be considered corrupt and re-transmission is usually required.

As the definition states, in Network Coding, intermediate nodes are equipped with the

ability to produce output packets out of the received ones following some predefined bijective mapping. Note that in some applications such as in [3], the source node may also be allowed to perform coding as well.

This paradigm has been first introduced by Yeung et al in their seminal paper [1], where they proved that this simple idea of allowing intermediate nodes to alter their received packets instead of just replicating them can offer a multitude of benefits such as those discussed below.

1.2.1 Throughput

The network of Figures 1.1 and 1.2 is called "The Butterfly Network" in the literature of Network Coding. It is commonly used to outline the throughput increase that Network Coding offers compared to conventional routing solutions. The butterfly network is modeled as a directed graph with one source s, four intermediate nodes and two sinks D_1 and D_2. The channels share the same capacity of one packet per time slot. The source tries to multicast data to both destinations using the highest rate possible. Using conventional routing solutions, the source will be able to send 3 packets to each destination in two time slots , that is a throughput of 1.5 packet/ time slot, given that the edge (I_3, I_4) can only forward one packet per time slot. By using a simple network coding solution, consisting of the XOR operation, we can transmit a packet P_1 and its XOR with a packet P_2 to D_1 and the packet P_2 and its XOR with packet P_1 to D_2. In this way, both destinations can recover the coded packet by adding the two received ones. This simple coding solution has allowed for a throughput of 2 packets per time slot, exceeding what is possible by conventional routing solutions and achieving the optimal throughput in this scenario.

Another useful scenario is the wireless network of Figure 1.3 and Figure 1.4. This network has a relay node R and two users U_1 and U_2. The links are all half-duplex. Each user has a single packet to send to the other user passing by the relay node R. Being Half-duplex, using routing alone, four transmissions are required since each packet will consume two time slots to reach the destination. Using Network coding, one may cut down on the number of required transmissions by using the broadcast nature of the wireless channel. In this scenario, each user will send its packet to R, which will then broadcast the XOR of the two packets. Each user will eventually be able to recover the intended packet by adding the received packet with its own packet. In this network, using coding has saved us one time slot compared to a routing solution.

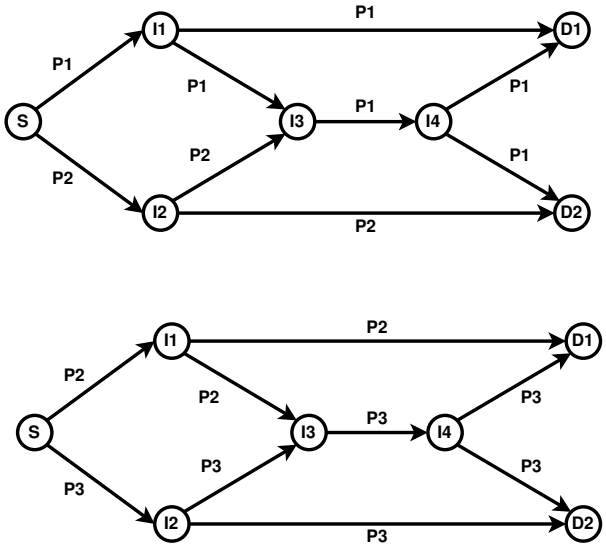

Figure 1.1: A routing scenario for the Butterfly Network.

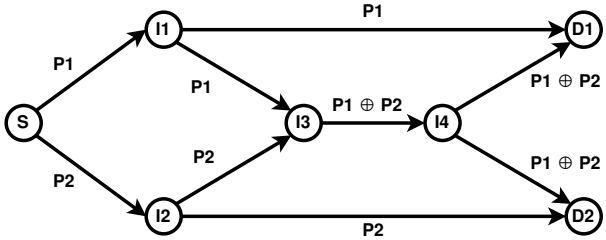

Figure 1.2: A network coding scenario for the Butterfly Network

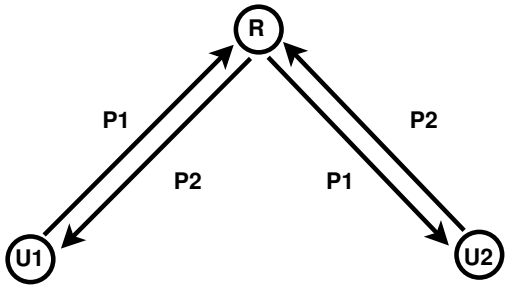

Figure 1.3: A routing scenario for a network consisting of two users, one relay node with half-duplex channels

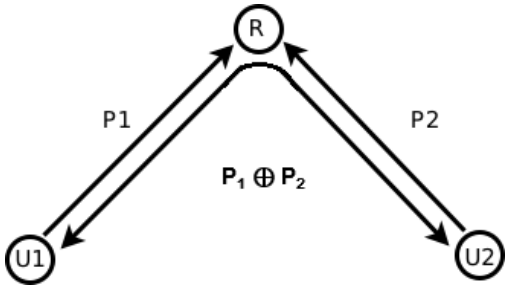

Figure 1.4: A network coding scenario for a network consisting of two users, one relay node with half-duplex channels

1.2.2 Security

Consider a wiretapper Eve that is going to wiretap the network in FIGURE 1.2 to extract some information about the source message. Suppose that Eve can only wiretap one channel of her choosing at a time. If this channel happens to be (I_3, I_4) of the network of Figure 1.2, she will not be able to get any of the packets without wiretapping another channel that is transmitting uncoded data, which is only possible if she has access to more channels. Therefore, this network has some edges that are immune against wiretappers with wiretapping capacity of one packet/time slot. This example provides an insight on the intrinsic security of Network Coding. In fact many schemes have been built around this idea where a pre-encoding step was carried out at the source and then the encoding coefficients were encrypted to hinder the decoding operation at unauthorized receivers. In SPOC [4] the source encoding matrix was encrypted and another coefficient matrix was attached to the augmented matrix for storing the encoding coefficients used across the network.[5], the source encoding matrix was permuted and transmitted as the encoding coefficient itself. P-coding[3], On the other side is based on the permutation of the augmented matrix columns to hide the coefficient matrix. In [6], Homomorphic encryption functions has been used to encrypt the coefficient matrix in order to circumvent the problem of SPOC that required the transmission of two coefficient matrices. In [7], security was based on number of wiretapped vectors as well as other steps applied at the source. These works were all based on the fact that mixing packets makes NC inherently weakly secure.

1.2.3 Data Storage

In distributed data storage systems, an intuitive solution to ensure data availability through backup is to replicate the disks on which data is stored. However, a better solution may consist of saving combinations of all the datasets in the redundant disks to increase the chances of data recovery in case of many disk failures.

In Figure 1.7, the failure of two disks may lead to permanent data loss. With coding as shown in Figure 1.8, data recovery is possible as long as 4 disks are operational

1.2.4 Packet Loss

In real world networks, especially the wireless case, there is always a non-zero probability that a packet will be lost before reaching its intended destination due to various factors such as collisions, link outage ...etc. This issue is usually addressed using ARQ protocol[8],

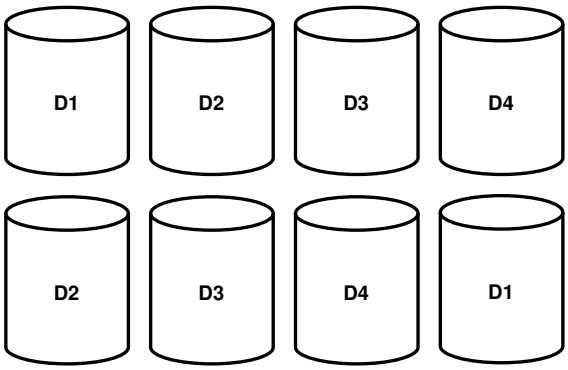

Figure 1.5: A distributed data storage system with replication based backup

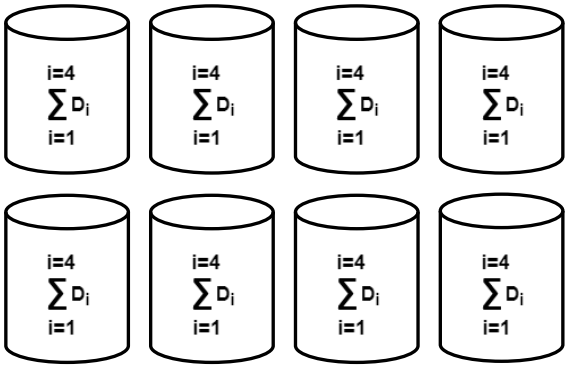

Figure 1.6: A distributed data storage system with network coding based backup

which is a system of acknowledgements and timeouts, where a packet is retransmitted if its reception is not acknowledged before a timeout event that specifies the allowed period to wait for an acknowledgement. An alternative method is to model the network channels as erasure channels and then design an erasure code that will allow for data recovery in the case of data loss.

Erasure codes are applied at the source code and decoded at destination nodes. Network Coding is coding at intermediate nodes. If the network code is also an erasure code, the maximum theoretical rate will also increase.

Consider a network N with some random acyclic topology. For the sake of simplicity, let all the links be identical erasure channels with erasure probability e. Let r and R denote the maximum rate allowed between any two adjacent nodes and the maximum rate between the source and a destination node, respectively. Since the edges are all erasure channels, we have $r = 1 - e$. A destination node separated by k edges from the source will dictate an overall rate $R = (1 - e)^K$. If the erasure code is applied at every intermediate node, R will equal to the minimum rate r on any edge in the path between the source and the given destination node. In this case, it is just $R = 1 - e$. Since network coding is applied at intermediate

nodes, a proper design of this latter would provide resilience against packet loss [9],[10].

1.3 Designing Network Coding Schemes

The aforementioned examples may have provided some insight into what can be achieved by using Network Coding. However, real world networks are huge, complex and most of them are constituted ad hoc. A network coding scheme has to be scalable and universal in the sense that it would not require a specific set of network topologies to operate. Moreover, the advantages of the designed scheme has to outweigh the incurred complexity of the encoding and decoding operations that are required by the scheme as well as the overall deployment cost resulted from the installation of coding-capable nodes.

In the literature of Network Coding, two schemes have been proven sufficient to accomplish the benefits of Network coding with minimum requirements, Linear Network Coding (LNC) [11] and Random Linear Network Coding (RLNC)[2]. In both schemes, encoding and decoding operations consist of a set of linear operations over finite fields. In this case, packets will be modelled as vectors over the underlying finite field. Encoding consists of the addition of scaled packets and decoding consists of solving a set of linear equations. The validity of both schemes lies in the proper selection of the encoding coefficients to make sure that all sinks in a network would receive a set of linearly independent vectors whose cardinality equals the number of source packets. In LNC, selecting the coefficients is carried out deterministically, taking into consideration all the encoding operations at the other intermediate nodes. In the absence of network errors, this approach will guarantee that no information is lost but it requires full awareness of the used network topology. This latter constraint is lifted in RLNC by opting for a randomized approach. The coefficients will be chosen randomly and as long as the used finite field is sufficiently large, the decoding failure probability will be significantly low. This approach has made RLNC suitable for non-coherent and ad hoc networks. In the literature, RLNC is the NC scheme that has attracted a lot of research and usually whenever NC is deployed, it is actually RLNC. In this thesis, we also focus on RLNC.

1.4 Random Linear Network Coding

1.4.1 Network Model

A network N is modelled as an acyclic multigraph $G(V, E)$, with V being the set of vertices and $E \subseteq V^2$ the set of edges. Usually, we refer to a vertex as a node and to an

edge as a channel to relate our analysis to the used terminology in communication theory. V is partitioned into three disjoint sets : S, I and D. S is the set of sources from which all information is originated. Without loss of generality, one may consider S to be a single element set, $|S| = 1$, by thinking of all the sources as being attached to one virtual super-source, in analogy to the super-node in electrical networks. Therefore s will be used to denote the source node. I is the set of intermediate nodes and D is the set of sinks or destination nodes. The elements of D are the nodes to which information is intended.

We let $e = (n_1, n_2)$ denote the edge emanating from the node n_1 and into the node n_2. Moreover, the node n_1, denoted as $n_1 = tail(e)$ is called the tail of e. Similarly, the node n_2 is the head of e and we write $n_2 = head(e)$.

$\forall n \in V$, we let $Out(n) = \{e \in E \mid tail(e) = n\}$ and $In(n) = \{e \mid head(e) = n\}$ denote the set of incoming and outgoing channels of the node n, respectively. Based on our network model, note that $head(s) = 0$ and $\forall d \in D,\ tail(d) = 0$.

A node n_1 is adjacent to a node n_2 if $(n_1, n_2) \in E$ or $(n_2, n_1) \in E$. Similarly, two edges e_1 and e_2 are adjacent if the head of one of them is the tail of the other one. A sequence of channels $e_1, e_2, \cdots, e_n$ with the property that $tail(e_i) = head(e_{i+1})\ \forall i \in \{1, 2, \cdots, n-1\}$ is referred to as a path from the node $head(e_1)$ to the node $tail(e_n)$.

The edges of E are all unit capacity edges. If two nodes are able to communicate at a higher capacity, multiple unit capacity edges will be employed to model this scenario.

A cut between a node n_1 and a node n_2 is a set of edges whose removal disconnects the two nodes. the smallest cut amongst all the cuts in terms of the number of edges is called the minimum cut, or simply the min-cut. For the edges of our network, the capacity of the cut will be equal to the number of edges in the cut.

We refer to a unicast transmission to the case when s wishes to communicate with a single element of D and to a multicast transmission when s simultaneouly transmits the same data to a multiple elements of D. We denote by C_m, the mulicast capacity of the network N, which is defined as the maximum rate at which the source can multicast data to the elements of D.

Theorem 1. *The maximum rate h at which information can be sent from the source s to any $d \in D$ equals the capacity of the min-cut between s and d. Equivalently, there exists h-edge disjoint paths from s to d.*

This theorem[1] is usually referred to us the Min-Cut Max-Flow theorem. Network

Coding is interesting in terms of improving throughput because it achieves this theorem even for the multicast case. Consider the following theorem[11].

Theorem 2. *$\forall i \in \{1, 2, \cdots, |D|\}$, let r_i denote the capacity of the min-cut between the source s and the sink $d_i \in D$. let $r = min_{<i>}(r_i)$. There exists a multicast transmission scheme over a large enough finite field $\mathbb{F}_q$, where the nodes of I linearly combine their incoming information symbols over $\mathbb{F}_q$, that simultaneously delivers information from s to each sink at a rate equal to r.*

In commodity flow networks, it is common that the maximum rate between a source and a destination node equals the capacity of the min-cut separating them as dictated by the Min-Cut Max-Flow theorem. However, generalizing this to multicast transmissions was not possible due to possible overlap between the different unicast paths. By treating Information as an abstract quantity and not as a physical commodity, Network coding has allowed for achieving the optimal multicast rate i.e. the mulicast capacity C_m. In fact, Therorm 2 shows that linearly combining the incoming packets at intermediate nodes is sufficient to attain the optimal multicast transmission rate. We refer to this scheme as Linear Network Coding (LNC).

1.4.2 Encoding Model

In linear network coding, the source information symbols as well as their combinations are all $n-$dimensional vectors over a finite field $\mathbb{F}_q$ where q is a prime power. In practice, we usually have $q = 2^r$ with $r \in \mathbb{N}^+$. Let $U = \{u_1, u_2, \cdots, u_{C_m}\}$ denote the source uncoded information symbols. In a network code, the source and at least some of the intermediate nodes are equipped with coding capabilities, we refer to them as NC-capable nodes. The set of all NC-capable nodes is denoted by $I_C \subseteq S \bigcup I$. While the source can choose to inject U without prior coding, some scenarios require a pre-coding step at the source node. This is usually the case of secure network coding schemes.

Linear network coding at an NC-capable node consists of summing scaled versions of its incoming packets using coefficients taken from the underlying finite field $\mathbb{F}_q$. Consider the following definitions[12].

Definition 2. *The local encoding function L_e at a node $n_i \in I_C$ is a linear mapping from $\mathbb{F}_q^{|In(n_i)|\times n}$ onto $\mathbb{F}_q^{|Out(n_i)|\times n}$.*

The local encoding function is the building block on which the network code is based.

Definition 3. *A C_m-dimensional linear network code on an acyclic network over an underlying finite field $\mathbb{F}_q$ at a node $n_i \in I_C$ consists of a scalar encoding coefficient $k_{ij} \in \mathbb{F}_q$ for every adjacent pair (i, j) of channels in the network. The $|In(n_i)| \times |Out(n_i)|$ matrix $L_{n_i} = (k_{ij})i \in In(n_i), j \in Out(n_i)$ is called the local encoding kernel at the node n_i. The local encoding kernel is the matrix form of the local encoding function L_e. All local encoding kernels constitute a local description of a linear network code.*

In order to use the encoding kernel we have to assume the existence of some order at which the incoming and the outgoing edges of a node n_i are enumerated. the set of input edges will be $In(n_i) = \{e_{in_1}, e_{in_1}, \cdots, e_{in_{|In(n_i)|}}\}$. Similarly, the set of output edges will be $Out(n_i) = \{e_{out_1}, e_{out_1}, \cdots, e_{out_{|Out(n_i)|}}\}$

This order will also help us define the $In(n_i) \times n$ input matrix $\boldsymbol{P_{in}^{(n_i)}}$ and the $Out(n_i) \times n$ output matrix $\boldsymbol{P_{out}^{(n_i)}}$. The rows of those matrices are the input and output packets of the node n_i which are $n-$dimensional vectors over $\mathbb{F}_q$. The encoding operation will then be a simple matrix multiplication as follows,

$$\boldsymbol{P_{out}^{(n_i)}} = \boldsymbol{L_n P_{in}^{(n_i)}} \tag{1.1}$$

One way to look at a linear code is to also analyse it edgewise. In this case, the j^{th} node in $Out(n_i)$ will carry the symbol $P_{out_j}^{(n_i)}$ calculated as,

$$P_{out_j}^{(n_i)} = \sum_{k=1}^{|In(n_i)|} \alpha_k^{(n_i)} P_{in_k}^{(n_i)} \tag{1.2}$$

where $\forall k \in \{1, 2, \cdots, |In(n_i)|\}$, α_k is an encoding coefficient taken from $\mathbb{F}_q$.

This edgewise description is more intuitive in relating $P_{out_j}^{(n_i)}$ to U. Each packet $P_{in_k}^{(n_i)} \in In(n_i)$ can be written as,

$$P_{in_k}^{(n_i)} = \sum_{l=1}^{|In(tail(e_{in_k})|} \beta_l^{(tail(e_{in_k}))} P_{in_l}^{(tail(e_{in_k}))} \tag{1.3}$$

Hence,

$$P_{out_j}^{(n_i)} = \sum_{k=1}^{|In(n_i)|} \alpha_k^{(n_i)} \sum_{l=1}^{|In(tail(e_{in_k})|} \beta_l^{(tail(e_{in_k}))} P_{in_l}^{(tail(e_{in_k})}$$

$$= \sum_{k=1}^{|In(n_i)|} \sum_{l=1}^{|In(tail(e_{in_k})|} \alpha_k^{(n)} \beta_l^{(tail(e_{in_k}))} P_{in_l}^{(tail(e_{in_k})} \tag{1.4}$$

$\forall k \in \{1, 2, \cdots, |In(n_i)|\}$, and $\forall l \in \{1, 2, \cdots, |In(tail(e_{in_k}))|\}$, let $\lambda_l^{tail(e_{in_k})} = \alpha_k^{(n_i)} \beta_l^{(tail(e_{in_k}))}$. Hence, $P_{out_j}^{(n_i)}$ can be expressed as,

$$P_{out_j}^{(n_i)} = \sum_{k=1}^{|In(n_i)|} \sum_{l=1}^{|In(tail(e_{in_k})|} \lambda_l^{(tail(e_{in_k}))} P_{in_l}^{(tail(e_{in_k})} \tag{1.5}$$

In equation (1.5), $P_{out_j}^{(n_i)}$ has been written as a linear combination of the input packets that precedes the node n_i in the ancestral order of the graph. Using recursion we can write $P_{out_j}^{(n_i)}$ in terms of the source packets. While $P_{out_j}^{(n_i)}$ is not necessarily a combination of all the source packets, we can just zero out the coefficients corresponding to the packets that are excluded. Let g_{out_j} be the vector formed by those coefficients,

$$P_{out_j}^{(n_i)} = g_{out_j} \cdot \boldsymbol{U} \tag{1.6}$$

We call the vector g_{out_j} the global encoding vector at n_i corresponding to the edge e_{out_j}. Rewriting this equation in terms of $\boldsymbol{P_{out}^{(n_i)}}$ results in the following equation,

$$\boldsymbol{P_{out}^{(n_i)}} = \boldsymbol{G^{(n_i)}} \cdot \boldsymbol{U} \tag{1.7}$$

$\boldsymbol{G^{(n_i)}}$ is the matrix whose row vectors are the global encoding vectors for the outgoing edges of n_i and it is referred to as the global encoding kernel of the network code at n_i. A global description of the network code consists of a local description as well as a global one. the local description is the one given in **Definition. 3** and it relates the input packets to the output packets of each node $n_i \in I_c$. The global description, on the other side, is given in **Definition. 4** and it relates the output packet of the node n_i to the source original packets.

Definition 4. *A C_m-dimensional linear network code on an acyclic network over an underlying finite field $\mathbb{F}_q$ at a node $n_i \in I_C$ consists of*

1. *a scalar encoding coefficient $k_{ij} \in \mathbb{F}_q$ for every adjacent pair (i, j) of channels in the network. The $|In(n_i)| \times |Out(n_i)|$ matrix $L_{n_i} = (k_{ij}),\ \ i \in In(n_i), j \in Out(n_i)$ is called the local encoding kernel at the node n_i. The local encoding kernel is the matrix form of the local encoding function L_e. All local encoding kernels constitute a local description*

of a linear network code.

2. *A global encoding vector g_e for each channel $e \in I_C$ relates each packet injected in an outgoing channel of the node n_i to the source original packets. The matrix whose rows are the global encoding vectors at a node n_i is called the global encoding kernel at the node n_i and denoted by $G^{(n_i)}$.*

1.4.3 Decoding model

Decoding is the inverse operation of encoding. It is the process of extracting the source original packets by the sink nodes from their received packets. Let $d \in D$ be a sink node and let C_d denote the capacity of the cut between the source and d. Clearly $C_d \geq C_m$. While some sink nodes may have cut capacities higher than C_m, the extra capacity will only carry redundant information since the source maximum multicast rate is C_m.

Definition 5. *The received packets at a sink $d \in D$ are related to $\boldsymbol{U}$ by the matrix G_d satisfying,*

$$\boldsymbol{P_{in}^{(d)}} = \boldsymbol{G_d} \cdot \boldsymbol{U} \tag{1.8}$$

provided that $Rank(\boldsymbol{G_d}) = C_m$, where $Rank(\cdot)$ denotes the rank function. $\boldsymbol{G_d}$ is called the global encoding matrix at the sink d.

Note that $\boldsymbol{G_d}$ is based only on the previous encoding operations since the sinks do not perform encoding operations. Upon receiving the encoded packets and constructing the $\boldsymbol{G_d}$ matrix whose rows are the received packets at d, the sink will try to perform the decoding operation. Decoding will be performed by applying Gauss elimination on the matrix $[\boldsymbol{G_d} \;\; \boldsymbol{G_d} \cdot \boldsymbol{U}]$. If after reduction, the rank of the resulted matrix is C_m, its row vectors will be taken as the source original information packets. Otherwise, information is lost due to some bad coefficient selection and retransmission will be required.

1.4.4 Network Code construction

In the previous sections, we have seen that NC provides many advantages over classical routing protocols, from which throughput enhancement is the main aspect due to its intrinsic relation to the Min-Cut Max-Flow theorem for multicast transmissions. However, a network based on an NC transmission scheme has to guarantee the basic requirement for any communication system: "Reliably delivering information from source to destination". In NC

schemes, reliability is monitored by a parameter called "the decoding failure probability", which is defined as the probability of unsuccessful decoding at a given sink node. In Linear Network Coding, two main factors are responsible for an eventual loss of information: the induction of external unsolicited information to that flowing across the network and bad coefficient selection. The first factor can be either caused by random network errors or a set of malicious nodes intentionally injecting corrupt packets into the network. The effect of extra unsolicited information is more serious in NC schemes compared to traditional routing protocols due to packet mixing. This latter will allow for errors to propagate and contaminate other packets, increasing the decoding failure probability in the network. When first introduced, NC was only considered for error-free environments and error-correction mechanisms were later devised for NC allowing for real-world deployment. The second problem in linear coding schemes lies in coefficient selection. To be decoded successfully, the received packets at a given sink d have to span the vector space spanned by the original source packets. In other words, if V is the vector space spanned by $P_{in}^{d} = \{P_{in_1}, P_{in_2}, \cdots, P_{in_{|P_{in}^{d}|}}\}$ for some d, then we must have $dim(V) = C_m$. Each input packet in $P_{in}^{d}$is issued by linearly combining the source packets or some encoded versions of them. For decoding to be carried out successfully at the sink nodes, the encoding coefficients have to be chosen carefully at every encoding operation to ensure that no linear dependency results between any two encoded packets across the network. Two main approaches have been proposed in the literature of NC to deal with this problem.

1.4.4.1 Deterministic Construction

The choice of the encoding coefficients is carried out deterministically based on the used network topology. This is usually done following the Jaggi-Sanders algorithm [13] that works by assigning a priori the global encoding kernels at the encoding intermediate nodes. Deterministic approach does not result in any linear dependency between the encoded packets resulting in zero decoding failure probability in the absence of errors.

1.4.4.2 Random Construction

In real world scenarios, network topologies tend to be more flexible and prone to changes. An illustration of this would be the mobile ad hoc network[14] where nodes join and leave the network constantly. These variations render a topology-based solution unattractive and unscalable. To deal with this problem, Ho et al[2],[15] proposed Random Linear Network Coding (RLNC) as a solution to the problem of coefficient selection in noncoherent networks.

In their approach, the encoding intermediate nodes will randomly choose their encoding coefficients from the underlying finite field $\mathbb{F}_q$. To minimize the probability of choosing an encoding coefficient that results in a linear dependency with another encoded packet, the size of $\mathbb{F}_q$ has to be sufficiently large[16], [17]. The decoding failure probability has been proven to be a decreasing function of q. Figure 1.9 illustrates this relation.

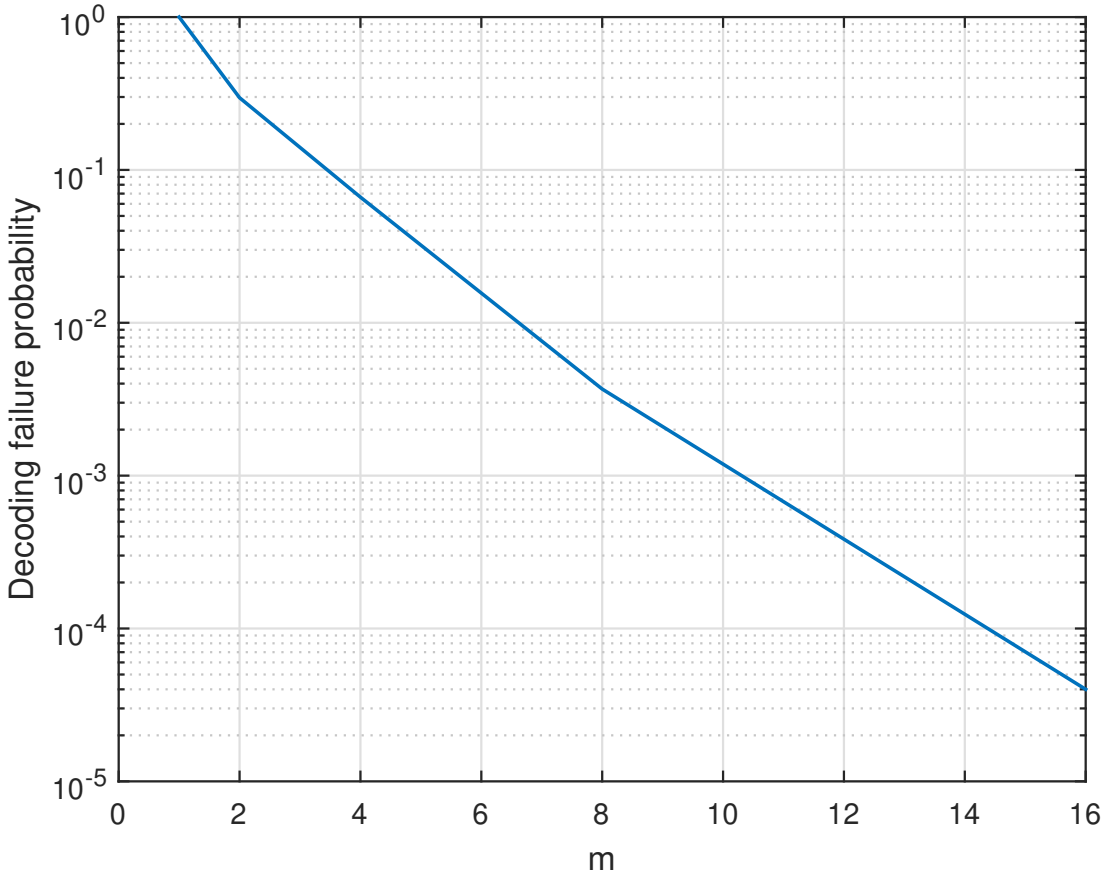

Figure 1.7: Decoding failure probability in a network of one source, 8 intermediate nodes and three sinks Vs. $\mathbb{F}_{2^m}$ for different values of m.

In the absence of network errors, the decoding failure probability is null for deterministic LNC. However, RLNC alleviates the cost of pre-designing a code for every network toplogy. Besides, It is more practical in noncoherent situations where network topology is unknown or dynamic in nature. These advantages come at the expense of a non-zero decoding failure probability that decreases as a function of q as mentioned earlier.

1.4.5 Coefficient Transmission in RLNC

Decoding at the sinks requires knowledge of the global encoding matrix $\boldsymbol{G}^{(d)}$ at every sink node $d \in D$. Since the global encoding coefficients are updated across the network, a

sink node d does not know a priori $\boldsymbol{G}^{(d)}$. Therefore, the used encoding operations have to be recorded across the network such that d would be able to construct $\boldsymbol{G}^{(d)}$ upon receiving a sufficient number of innovative packets. Those latter packets are the ones that provide information that has not yet been received.

The source will transform its data matrix $\boldsymbol{U}$ into the augmented matrix $\boldsymbol{U_{aug}}$ by attaching a distinct $1 \times C_m$ unit vector to each one of its rows such that $\boldsymbol{U_{aug}} = [\boldsymbol{I_{C_m}}\ \boldsymbol{U}]$, where $\boldsymbol{I_{C_m}}$ is the $C_m \times C_m$ identity matrix over $\mathbb{F}_q$. The transmission of $\boldsymbol{U_{aug}}$ instead of $\boldsymbol{U}$ will provide a means of recording the encoding operations across the network. In this case, a sink node $d \in D$ will receive a matrix of the form $[\boldsymbol{G}^{(d)}\ \boldsymbol{G}^{(d)} \cdot \boldsymbol{U}]$, of which reduction using Gauss Elimination will be possible. An illustration of this operation is provided in Figure 1.10.

$$\begin{bmatrix} 3d & 85 & ae & 0a & 1a & 23 \\ 76 & 3c & 65 & e2 & 43 & b8 \\ f6 & 7d & 5e & e9 & 56 & 1c \\ 8c & a0 & fc & cc & ae & a7 \end{bmatrix}$$

(a) The matrix U

$$\begin{bmatrix} 01 & 00 & 00 & 00 & 3d & 85 & ae & 0a & 1a & 23 \\ 00 & 01 & 00 & 00 & 76 & 3c & 65 & e2 & 43 & b8 \\ 00 & 00 & 01 & 00 & f6 & 7d & 5e & e9 & 56 & 1c \\ 00 & 00 & 00 & 01 & 8c & a0 & fc & cc & ae & a7 \end{bmatrix}$$

(b) The matrix U_{aug}

$$\begin{bmatrix} 9a & 4c & 16 & ed \\ b6 & 52 & 43 & bb \\ 39 & 6d & cd & 7d \\ 1e & 82 & 08 & 94 \end{bmatrix}$$

(c) The matrix $G^{(d)}$

$$\begin{bmatrix} 9a & 4c & 16 & ed & e7 & 07 & 29 & c3 & 12 & 1c \\ b6 & 52 & 43 & bb & 86 & f1 & 90 & 72 & 3b & d7 \\ 39 & 6d & cd & 7d & 3f & 23 & 71 & e2 & bb & 2c \\ 1e & 82 & 08 & 94 & 79 & b7 & 4b & db & 1a & 77 \end{bmatrix}$$

(d) The matrix $[G^{(d)}\ \ G^{(d)}U]$

Figure 1.8: An illustration of the source matrix, its augmented version as well as the global encoding matrix and the encoded version of the source message as received at a sink d over the field $\mathbb{F}_{2^8}$.

1.5 Challenges of Network Coding

1.5.1 Overhead

Applying Network Coding requires the transmission of extra information to allow for decoding at the sink nodes such as the encoding matrix in RLNC. This extra information will induce an additional amount of overhead [18],[19], [20] that will eventually affect the global throughput in the network. A proper deployment of RLNC for throughput enhancement requires justification of the possible throughput gain that the scheme offers for the application in question.

1.5.2 Complexity

Encoding and decoding operations naturally require a certain amount of time as a result of the increased computational complexity, which will eventually induce some level of latency in the network. While RLNC uses only linear operations, which are known to be fast, some authors tried to reduce this complexity to increase its efficiency. In [21], the autors attempted to solve this problem by reducing the number of required multiplications in the scheme. Their idea was based on the fact that uniform distribution is preserved under convolution on finite multiplicative groups [22]. In this scheme, the nodes will decide on the usefulness of the RLNC multiplicative operations based on the previous encoding operations performed on the received packets. Other schemes that aim to reduce the encoding complexity are also available in the literature. The authors of [23] have considered the cause of this complexity to be the number of nodes undergoing NC, which have to be equipped with NC-capabilities, making them expensive compared to their forwarding counterparts. Their solution was to find a feasible network code where a bounded number of NC-capable nodes is sufficient to solve the multicast problem. The same idea was revisited in [24], [25], and [26] for different cases and , as an evolutionary problem [27] and as an optimization problem in [28], and [29].

1.5.3 Error Propagation

When Yeung et al introduced Network Coding, they only considered error-free environments. It is evident that errors are inevitable in real-world scenarios. Therefore, one can deduce that the authors thought of Network Coding and error correction as two separate problems. Network Coding is based on packet mixing, making error propagation a real problem. In RLNC, one corrupt packet has the potential to corrupt all transmitted information. Even if some information is correctly received, decoding will still be impossible

without the whole set of valid packets. Therefore, RLNC has to be accompanied by error correction mechanisms that are suitable for the scheme. This thesis is dedicated mainly for this purpose.

1.6 Summary

Network coding is a transmission paradigm that incorporates the encoding intermediate nodes in a given network with the ability to perform encoding operations on their received packets. This feature has allowed for many advantages such as throughput enhancement, secure transmission and packet loss mitigation. While the term "Network Coding" is a global term that engulfs all transmission schemes that allow for packet alteration at the intermediate nodes. In practice, two such schemes are considered useful: Linear Network Coding (LNC) and Random Linear Network Coding (RLNC). Encoding operations for both of those schemes are based on linear transformations using coefficients taken from the underlying finite field $\mathbb{F}_q$. Those coefficients are pre-selected in LNC based on the network topology, and randomly generated in RLNC, making RLNC more practical and scalable, especially for noncoherent networks. However, this comes at the expense of a nonzero decoding failure probability that decreases as a function of q.

Aside from its benefits, Network Coding faces a set of challenges that have been heavily addressed in the literature such as the increased computational complexity, overhead and error propagation. While the first two challenges do not result in information loss, error propagation is a serious problem that may lead to a total information loss. In this thesis, we focus on this problem where we investigate the approaches that are adopted in the literature and we try to propose solutions that combine both error control and data security.

Chapter 2

Error Control in Random Linear Network Coding

2.1 Introduction

The idea of providing the set of intermediate nodes with coding capabilities has offered a set of advantages over classical routing protocols at the expense of some emerging challenges as illustrated in the previous chapter. While complexity and overhead may result in latency and decreased throughput, they do not cause any information loss unlike the error problem. Errors in network coding have the tendency to propagate [30], [31], eventually corrupting all transmitted information. Even without propagation, decoding requires the very absence of errors.

Error propagation makes errors overwhelm any traditional codes that are designed for the hamming metric [32]. Therefore, those codes are not generally adopted for network coding, which calls for novel error-correcting codes that are designed specifically for network codes. Those error codes are labelled as network error correction codes to distinguish them from their classical counterparts. Network error correction codes were first proposed in [33], followed by generalizations of bounds and constructions from the classical error coding theory [34],[35],[36],[37]. Similar to the difference between LNC and RLNC, most of those codes assumed a priori knowledge and well awareness of the network topology, making them unsuitable for noncoherent scenarios, for which RLNC is the proper choice. In RLNC, the channel transfer characteristics are assumed unknown by both the source and sink nodes, which complicates any design that is based on classical coding theory. In this chapter, we investigate network error correction codes for RLNC based networks.

2.1.1 Notations

Let $q \geq 2$ be a prime power. We denote the finite field with q elements by $\mathbb{F}_q$. The ambient space of dimenstion n over $\mathbb{F}_q$ is denoted by $\mathbb{F}_q^n$ and the set of $n \times m$ matrices over $\mathbb{F}_q$ is denoted by $\mathbb{F}_q^{n \times m}$.

The set of all subspaces of $\mathbb{F}_q^n$ or the projective space over $\mathbb{F}_q$ is denoted by $\mathcal{P}(n)$. The set of all k-dimensional subspaces of $\mathbb{F}_q^n$ with $0 \leq k \leq n$, or the Grassmannian, is denoted by $\mathcal{G}(k, n)$.

If M is a matrix, we represent the subspace spanned by the row vectors of M by $\langle M \rangle$. Similarly, we represent a subspace by choosing, as a representative, a matrix whose row vectors span that subspace. To simplify computation as well as implementation, we opt for the simplest representation of such matrices which is arguably their reduced row echelon form (RREF).

The dimension of a subspace V is denoted by $dim(V)$ and the rank of a matrix M is denoted by $Rank(M)$.

2.2 Subspace codes

Subspace codes are a family of codes whose codewords are subspaces of an ambient vector space. They have been traditionally used in authentication theory [38] as a tool to validate the source of information as well as its integrity. In this setting, they are usually referred to as linear authentication codes. They are also adopted in distributed storage systems [39],[40] to design a backup system such that when a subset of storage nodes fails, the lost data may be recovered up to a certain threshold. In [41],[42], the authors showed that those codes can be used as network error correction codes as well. In RLNC, we usually think of data as a set of vectors or packets to be transmitted across the network. Those vectors are treated as matrices for the mere purpose of encoding and decoding operations. However, what is really transmitted is the vector space spanned by those vectors and as long as this vector space is not altered across the network, information is preserved. In this system, decoding will be seen as finding the normal basis for the received vector space. Based on this observation, Kotter and Kschischang proposed the use of subspace codes as an outer code for error correction with RLNC, where RLNC is the inner code.

Definition 6. *A subspace code $\mathcal{C}$ is a non-empty subset of $\mathcal{P}(n)$. If $\mathcal{C} \subseteq \mathcal{G}(k,n)$ for some $0 \leq k \leq n$, we call $\mathcal{C}$ a Constant Dimension Code (CDC). Otherwise, $\mathcal{C}$ is a Mixed Dimension Code (MDC).*

Let N be a network following the model introduced in the previous chapter and let $\mathcal{C}$ be a subspace code used by the source node s as an outer code for error correction in RLNC. Let $V \in \mathcal{C}$ be the source codeword to be injected in its outgoing channels at a given transmission round. Note that the transmitted codeword satisfies the following inequality.

$$dim(V) \leq C_m \tag{2.1}$$

The received subspace at a given sink $d \in D$ will be denoted by R.

Errors that may occur during transmission are either erasures, which are defined as a decrease in the dimension of the transmitted codeword V, or insersions, which are defined as an increase in its dimension. This increase occurs when the codeword R contains a set of

vectors that do not belong to $\langle V \rangle$. The received codeword R can be expressed as,

$$R = V' \oplus E_s \tag{2.2}$$

Where V' is a subspace of U and $E_s \in \mathcal{P}(n)$ is the error space resulting from insertions.

Definition 7. *A random linear network channel in which insertions and erasures occur is called the operator channel whose input alphabet I and output alphabet O are subsets of P(n) satisfying,*

$$O = \mathcal{H}_k(I) \oplus E_s \tag{2.3}$$

where $\mathcal{H}_k$ is a stochastic operator that randomly returns a k-dimensional subspace of I resulting in $dim(I) - k$ erasures and E_s is the error subspace resulting in $dim(E_s)$ insertions.

Similar to classical error control theory, it is crucial to define a metric to allow for meaningful decoding at the sink nodes. In the literature of subspace codes, two main metrics are widely used, the subspace distance [41]and the injection distance [43].

The minimum subspace and injection distances of a code $\mathcal{C} \in \mathcal{P}(n)$ provides the basis to evaluate the error detection and correction capability of the code.

Definition 8. *The minimum subspace distance of a subspace code $\mathcal{C} \in \mathcal{P}(n)$ is defined as,*

$$d_{S_{min}} = \min\{d_S(U, V) | U, V \in \mathcal{C}, U \neq V\} \tag{2.4}$$

Similarly, we introduce the minimum injection distance of a code $C \in \mathcal{P}(n)$

Definition 9. *The minimum injection distance of a subspace code $\mathcal{C} \in P(n)$ is defined as,*

$$d_{I_{min}} = \min\{d_I(U, V) | U, V \in \mathcal{C}, U \neq V\} \tag{2.5}$$

Aside from the distance measure, decoding algorithms in classical coding theory have their counterparts in subspace coding. The maximum likelihood decoder and the minimum distance decoder are defined as follows.

Definition 10. *Consider a subspace code $\mathcal{C} \in \mathcal{P}(n)$. Let U be the source transmitted codeword at a given transmission round and let R be the received version of U at a given sink*

$d \in D$. A maximum likelihood decoder decodes R as $\hat{U}$, where $\hat{U} \in \mathcal{C}$ is the codeword that maximizes the probability

$$P(R|V) \tag{2.6}$$

over all $V \in \mathcal{C}$.

Definition 11. *Consider a subspace code $\mathcal{C} \in \mathcal{P}(n)$. Let U be the source transmitted codeword at a given transmission round and let R be the received version of U at a given sink $d \in D$. A minimum distance decoder chooses the closest codeword to the received word with respect to the used metric.*

$$\min d(V, R) \tag{2.7}$$

over all $V \in \mathcal{C}$, where $d(\cdot, \cdot)$ is either the subspace or injection distance.

If more than one codeword satisfies this inequality, the decoder returns "failure".

Let λ be the minimum (injection or subspace) distance of a code $\mathcal{C} \in \mathcal{P}(n)$. Let U be a transmitted codeword and R be its received version at a given sink $d \in D$. If $V \in \mathcal{C}$ is at distance that is at most $\frac{\lambda-1}{2}$, V will be always chosen by the minimum distance decoder.

2.3 Bounds on Subspace Codes

Definition 12. *The cardinality $|\mathcal{C}|$ of a subspace code $\mathcal{C} \in P(n)$ is defined as the number of codewords in the code. .*

The cardinality of a code has a direct impact on the code rate and its error correction capability. A set of bounds have been derived in the literature to provide a certain insight on its value when its hard to obtain an empirical estimation. Some of these bounds are reviewed below,

2.3.1 Sphere packing and covering bounds

Those bounds are similar to their counterparts in linear block codes. They have been derived by Koetter and Kschischang in [41].

Definition 13. *Let $\mathbb{F}_q^n$ be an n-dimensional vector space over the finite field $\mathbb{F}_q$ and let $\mathcal{G}(n, k)$ be the set of k-dimensional subspaces of $\mathbb{F}_q^n$.the sphere $\mathcal{S}(V, t)$ centred at the subspace*

V and with radius t is defined as the set of subspaces in $\mathcal{G}(n,k)$ satisfying

$$\mathcal{S}(V,t) = \{U \in \mathcal{G}(n,k) |\ \ d_S(V,U) \leq 2t\} \tag{2.8}$$

The number of subspaces in a given sphere $\mathcal{S}(V,t)$ is given by,

$$|\mathcal{S}(V,t)| = \sum_{i=0}^{t} q^{i^2} \begin{bmatrix} k \\ i \end{bmatrix}_q \begin{bmatrix} n-k \\ k \end{bmatrix}_q \tag{2.9}$$

Based on the previous definition, the sphere packing and the sphere covering bounds can be stated as:

1. **Sphere packing bound**

$$|\mathcal{C}| \leq \frac{|\mathcal{G}(k,n)|}{|\mathcal{S}(V, \frac{d-1}{2})|} \tag{2.10}$$

$$\leq \frac{\begin{bmatrix} n \\ k \end{bmatrix}_q}{\sum_{i=0}^{\frac{d-1}{2}} q^{i^2} \begin{bmatrix} k \\ i \end{bmatrix}_q \begin{bmatrix} n-k \\ i \end{bmatrix}_q} \tag{2.11}$$

2. **Sphere covering bound**

$$|\mathcal{C}| \geq \frac{|\mathcal{G}(k,n)|}{|\mathcal{S}(V, d-1)|} \tag{2.12}$$

$$\geq \frac{\begin{bmatrix} n \\ k \end{bmatrix}_q}{\sum_{i=0}^{d-1} q^{i^2} \begin{bmatrix} k \\ i \end{bmatrix}_q \begin{bmatrix} n-k \\ i \end{bmatrix}_q} \tag{2.13}$$

2.3.2 Singleton bound

In the same paper where the authors introduced the packing and covering bounds, they also derived a Singleton bound to subspace codes. As the name suggests, this bound is analogous to its counterpart in classical coding theory. In order to do so, they defined a puncturing operation where a subspace $V \in \mathcal{C}$ is replaced by one of its $(k-1)$- dimensional subspaces. This bound is stated as follows,

$$\begin{aligned} |\mathcal{C}| &\leq |\mathcal{G}(n-d+1, k-d+1)| \\ &\leq \begin{bmatrix} n-d+1 \\ n-k \end{bmatrix}_q \end{aligned} \tag{2.14}$$

2.3.3 The anticode bound

This bound has been derived in [44], in which the authors used subspace codes as linear authentication codes. The same bound has been revisited by Etzion and Vardy [45],[46], where they proved that the sphere packing bound is a special case of Delsarte's bound [47] for a Grassmannian association scheme. Note that the name "anticode" is derived from the structures that are used in obtaining this bound similar to the packing spheres. An antidote $A_n(t)$ of diameter t is defined as being any subset of $\mathcal{G}(k,n)$ satisfying,

$$\forall U, V \in A_n(t), \quad d_S(V,U) \leq 2t \tag{2.15}$$

The anticode bound is formulated as:

$$|\mathcal{C}| \leq \frac{\begin{bmatrix} n \\ k-d+1 \end{bmatrix}_q}{\begin{bmatrix} k \\ k-d+1 \end{bmatrix}_q} \tag{2.16}$$

2.4 Constructions of subspace codes

Subspace code constructions are divided into two main families: Lifted Rank-Metric codes and Orbit codes. The first family makes use of rank metric codes.

2.4.1 Rank Metric codes

Rank-metric codes are matrix codes that are designed for the Rank-metric. There are two representations of those codes : the matrix representation and the vector representation. In the first representation, those codes are taken as subsets from the vector space $\mathbb{F}_q^{n\times m}$. However, in the vector representation, the codewords are taken as m-dimensional vectors over the extension field $\mathbb{F}_{q^n}$. Clearly, the two representations are isomorphic. In our work, we are more interested in the matrix representation, given that a codeword from a subspace code is represented as the row space of given matrix in its reduced row echelon form.

Rank metric codes have been independently introduced by Gabidulin [48], Delsarte[49] and Roth[50]. They have been traditionally used to correct crisscross errors [51]–[52]. These errors occur in situations when information is stored or transmitted as an array such as in memory chip arrays and magnetic tapes. In this model, corrupted symbols are usually seen

as row errors and column errors and since those errors are generally bursty, codes designed for the hamming metric are not generally suitable in these situations. Note that those codes have also been used in the design of cryptosystems [53], [54], [55] and space-time codes [56],[57].

Since those codes are based on matrices, they provide a ground upon which subspace codes may be constructed. A formal definition of rank metric codes for matrix representation may be summarized as follows,

Definition 14. *A rank metric code is a subset of the vector space $\mathbb{F}_q^{n\times m}$ of rectangular $m \times n$ matrices.*

When the the rank metric code is a subspace of the vector space $\mathbb{F}_q^{n\times m}$, we refer to it as linear rank metric code

Definition 15. *A linear rank-metric code is a subspace of $\mathbb{F}_q^{n\times m}$.*

Rank metric codes are in fact named after their distance measure, which is referred to as the rank distance. Note that in the literature, codes are also defined based on their distance measure. In this case, the code will be seen as a subset of a normed (or metric) space endowed with a particular metric.

The rank distance is defined as follows,

Definition 16. *The rank distance d_R is a metric on $\mathbb{F}_q^{n\times m}$ defined as,*

$$\forall X, Y \in \mathbb{F}_q^{n\times m}, d_R(X,Y) = Rank(A-B) \tag{2.17}$$

Similar to all codes, a formal definition of the minimum distance measure is required to allow for meaningful decoding at destination nodes.

Definition 17. *The minimum rank distance d_{Rmin} of a code $\mathcal{C} \in \mathbb{F}_q^{n\times m}$ is defined as,*

$$d_{Rmin} = \min\{d_R(X,Y) | X, Y \in \mathcal{C}, X \neq Y\} \tag{2.18}$$

When we talk about rank metric codes, we usually refer to the maximum rank metric codes (MRD). Those codes were the first rank metric codes and they are usually referred to as Gabidulin codes to highlight their author. Those codes achieve the Singeton-like bound. In this regard, those codes are a matrix version of the maximum distance separable (MDS) codes for classical linear block codes.

Definition 18. *Let $\mathcal{C} \in \mathbb{F}_q^{n\times m}$ be a linear rank metric code with minimum rank distance λ. If the cardinality of $\mathcal{C}$ satisfies the following Singleton bound with equality,*

$$|C| \leq q^{\max\{m,n\}(min\{m,n\}-\lambda+1)} \tag{2.19}$$

we call $\mathcal{C}$ a maximum rank distance code (MRD).

We usually denote a rank metric code $\mathcal{C} \in \mathbb{F}_q^{m\times n}$ with cardinality $|\mathcal{C}|$ and a minimum rank distance $d_{R_{min}}$ by $(n, |\mathcal{C}|, d_{R_{min}}, m)$.

In the literature of MRD codes, Gabidulin MRD codes are the ones that attracted a lot of attention. Those codes may be considered as the Reed-Solomon (RS) codes for the rank metric codes. Let $F_q^{m\times n}$ be the set of $m \times n$ matrices over $\mathbb{F}_q$ with mn. Consider the m^{th} extension of F_q and let $G = \{g_1, g_2, \cdots, g_n\} \in \mathbb{F}_{q^m}$ be a linearly independent set. The generator matrix for a Gabidulin code with minimum distance $d_{R_{min}}$ for G is shown below. Note that $[i]$ denotes q^i.

$$\begin{bmatrix} \mathrm{g}_1 & \mathrm{g}_2 & \cdots & \mathrm{g}_n \\ \mathrm{g}_1^{[1]} & \mathrm{g}_2^{[1]} & \cdots & \mathrm{g}_n^{[1]} \\ \mathrm{g}_1^{[2]} & \mathrm{g}_2^{[2]} & \cdots & \mathrm{g}_n^{[2]} \\ \cdots & \cdots & \cdots & \cdots \\ \mathrm{g}_1^{[n-d_{R_{min}}]} & \mathrm{g}_2^{[n-d_{R_{min}}]} & \cdots & \mathrm{g}_n^{[n-d_{R_{min}}]} \end{bmatrix}$$

As an illustration for Gabidulin codes, let $m = n = 3$ and consider the $(3, 8, 3, 3)$ Gabidulin code. The elements of extension field $\mathbb{F}_{2^3}$ can be generated using the irreducible polynomial $P = x^3 + x + 1$. Let α be a root of this polynomial. and let $G = \{1, \alpha + 1, \alpha^2\}$. The elements of the extension fields in terms of α are $\mathbb{F}_{2^3} = \{0,\ 1,\ \alpha,\ \alpha^2,\ \alpha + 1,\ \alpha^2 + \alpha,\ \alpha^2 + \alpha + 1,\ \alpha^2 + 1\}$ Hence, the codewords will be:

(i) $[0\ \ 0\ \ 0]$.

(ii) $[1\ \ \alpha + 1\ \ \alpha^2]$.

(iii) $[\alpha\ \ \alpha^2 + \alpha\ \ \alpha + 1]$.

(iv) $[\alpha^2\ \ \alpha^2 + \alpha + 1\ \ \alpha^2 + \alpha]$.

(v) $[\alpha + 1\ \ \alpha^2 + 1\ \ \alpha^2 + \alpha + 1]$.

(vi) $[\alpha^2+\alpha \;\; 1 \;\; \alpha^2+1]$.

(vii) $[\alpha^2+\alpha+1 \;\; \alpha \;\; \alpha^2]$.

(viii) $[\alpha^2+1 \;\; \alpha^2 \;\; \alpha]$.

While the vector representation is helpful in the evaluation of the different codewords of the book, as we have already stated, we are more interested in the matrix representation of rank metric codes. Below is an illustration of the matrix representation of the code.The minimum rank distance of this code can be easily checked out by noticing that the rank of the difference of any two codewords is always 3 for this code.

$$\begin{bmatrix} 0 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \end{bmatrix}$$

(a)

$$\begin{bmatrix} 1 & 0 & 0 \\ 1 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

(b)

$$\begin{bmatrix} 0 & 1 & 0 \\ 0 & 1 & 1 \\ 1 & 1 & 0 \end{bmatrix}$$

(c)

$$\begin{bmatrix} 0 & 0 & 1 \\ 1 & 1 & 1 \\ 0 & 1 & 1 \end{bmatrix}$$

(d)

$$\begin{bmatrix} 1 & 1 & 0 \\ 1 & 0 & 1 \\ 1 & 1 & 1 \end{bmatrix}$$

(e)

$$\begin{bmatrix} 0 & 1 & 1 \\ 1 & 0 & 0 \\ 1 & 0 & 1 \end{bmatrix}$$

(f)

$$\begin{bmatrix} 1 & 1 & 1 \\ 0 & 1 & 0 \\ 1 & 0 & 0 \end{bmatrix}$$

(g)

$$\begin{bmatrix} 1 & 0 & 1 \\ 0 & 0 & 1 \\ 0 & 1 & 0 \end{bmatrix}$$

(h)

2.4.2 Lifted Rank-Metric Codes

One the most important subspace error correcting codes in the literature are the KK codes, named after their respected authors Kotter and Kschichang. Those codes are based on Gabidulin codes. In their construction, the authors have made use of the notion of lifting.

Definition 19. *Let $\mathcal{C} \in \mathbb{F}_q^{n\times m}$ be a linear rank metric code. The lifting of $\mathcal{C}$ is the set $Lft(\mathcal{C})$ defined as,*

$$Lft(\mathcal{C}) = \{\langle I_n V\rangle | V \in \mathcal{C}\} \tag{2.20}$$

The following theorem states how constant dimesion codes may be constructed from MRD codes via lifting [58].

Theorem 3. *Let $\mathcal{C} \in \mathbb{F}_q^{n\times(m-n)}$ be an MRD code. $Lft(\mathcal{C}) \subseteq \mathcal{G}_(n, m)$ is a constant dimension code.*

$$\begin{bmatrix} 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 \end{bmatrix}$$

(a)

$$\begin{bmatrix} 1 & 0 & 0 & 1 & 0 & 0 \\ 0 & 1 & 0 & 1 & 1 & 0 \\ 0 & 0 & 1 & 0 & 0 & 1 \end{bmatrix}$$

(b)

$$\begin{bmatrix} 1 & 0 & 0 & 0 & 1 & 0 \\ 0 & 1 & 0 & 0 & 1 & 1 \\ 0 & 0 & 1 & 1 & 1 & 0 \end{bmatrix}$$

(c)

$$\begin{bmatrix} 1 & 0 & 0 & 0 & 0 & 1 \\ 0 & 1 & 0 & 1 & 1 & 1 \\ 0 & 0 & 1 & 0 & 1 & 1 \end{bmatrix}$$

(d)

$$\begin{bmatrix} 1 & 0 & 0 & 1 & 1 & 0 \\ 0 & 1 & 0 & 1 & 0 & 1 \\ 0 & 0 & 1 & 1 & 1 & 1 \end{bmatrix}$$

(e)

$$\begin{bmatrix} 1 & 0 & 0 & 0 & 1 & 1 \\ 0 & 1 & 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 1 & 0 & 1 \end{bmatrix}$$

(f)

$$\begin{bmatrix} 1 & 0 & 0 & 1 & 1 & 1 \\ 0 & 1 & 0 & 0 & 1 & 0 \\ 0 & 0 & 1 & 1 & 0 & 0 \end{bmatrix}$$

(g)

$$\begin{bmatrix} 1 & 0 & 0 & 1 & 0 & 1 \\ 0 & 1 & 0 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 & 1 & 0 \end{bmatrix}$$

(h)

A legitimate quest is to investigate the main properties of Gabidulin codes and their counterparts in their lifted version i.e. KK codes.

Table 2.1 provides a comparison between the cardinality of subspace codes and Gabidulin codes for some matrix dimensions. For Gabidulin codes, the matrix dimension is the dimension of the codewords. However, for KK codes,the matrix dimension represents the dimension of the matrix representative of the code. The minimum distance of all the codes is equal to its maximum value , which is m for Gabidulin codes and $2m$ for KK codes. As an illustration, when the matrix dimension is 2×4, the minimum subspace distance of the corresponding code is 4 and that of Gabidulin code is 2.

From this table, we notice that for a given matrix dimension and highest minimum distance, Gabidulin codes enjoy higher cardinality as compared to KK codes. This table also shows that when a KK code has the same cardinality as a rank metric code, its codewords will be greater in dimension and hence, more symbols are required per transmitted codeword.

Table 2.1: Comparsion between the cardinality of a set of KK subspace codes (denoted as $\mathcal{C}$) and Gabidulin codes (denoted as $\mathcal{C}_R$) for given matrix dimensions.

Matrix dimension	**Cardinality of $\mathcal{C}$**	**Cardinality of $\mathcal{C}_R$**
2×4	4	16
3×6	8	64
4×8	16	256
4×10	64	1024
8×16	256	65536

To evaluate the correction capability of the two code families, we consider the following codes: the $(16, 256, 16, 8)_2$ KK code and the two Gabidulin codes, the $(16, 65536, 8, 8)_2$ and the $(8, 256, 8, 8)_2$. We randomly select a codeword of each code and flip a certain number of its bits at random. We proceed by decoding the erroneous codeword. This experiment is repeated 1000 times to allow for an estimation of the decoding failure probability. This latter occurs when the decoder fails to output the original codeword. FIGURE 2.4 depicts the results of this experiment. KK codes are endowed with a better performance compared to their Gabidulin counterparts for almost all the time. More specifically, it is more probable that decoding will be successful as long as error flips are less that about 40 for the KK code. As for Gabidulin codes, this is true only when error flips do not exceed 27 for the $(16, 65536, 8, 8)_2$ code and 20 for the $(8, 256, 8, 8)_2$ code. note that we can also notice that for a given minimum distance, the greater the dimension of the the rank metric code, the better its correction capability will be.

As for the effects of random error flips on the distance between a codeword and its

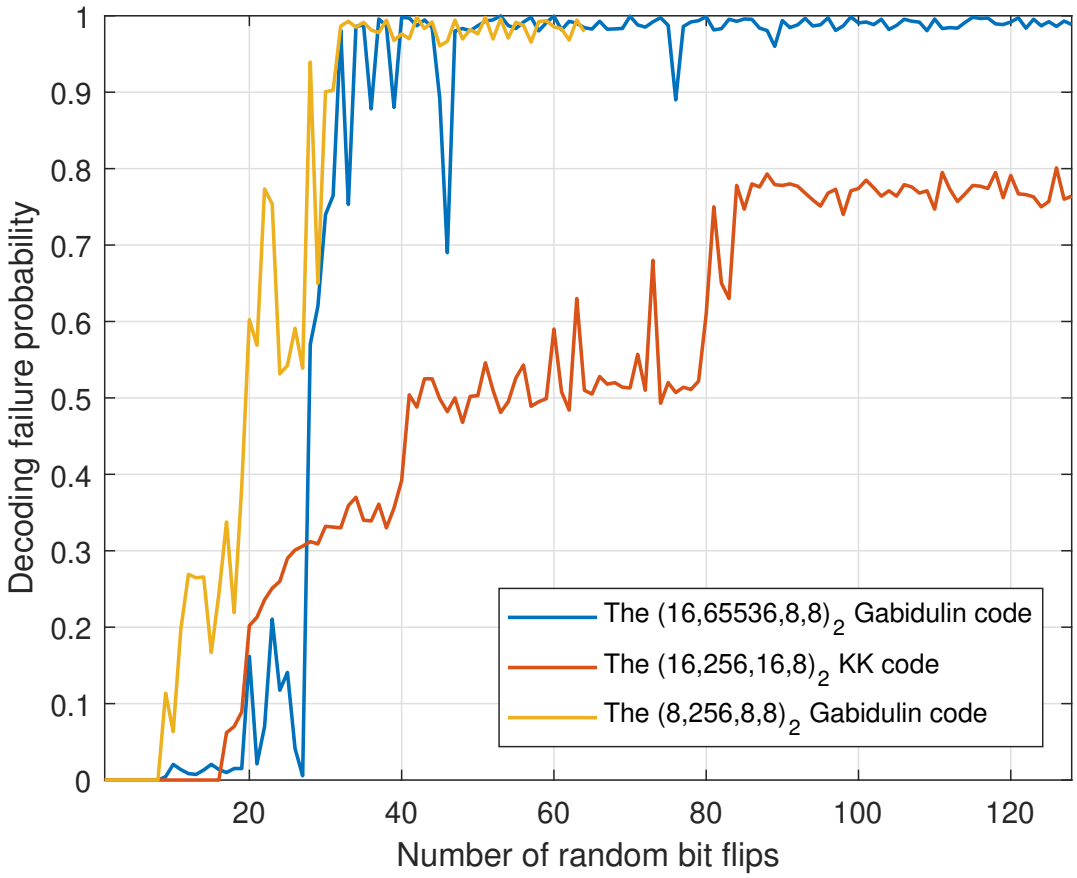

Figure 2.3: Decoding failure probability Vs. Number of random bit flips for the $(16, 256, 16, 8)_2$ KK subspace code, and the two gabidulin codes : $(16, 65536, 8, 8)_2$ and $(8, 256, 8, 8)_2$.

erroneous version, Table II shows the distance between a codeword and its erroneous version after one and two random bit flips. For subspace codes, this distance is either 0,1 or 2 for both cases. This means that random error flips on a codeword from a subspace code can result in no error , an erasure or both an erasure and an insertion. This last case is when the error transforms the affected vector into another vector that is not in the subspace spanned by the corresponding codeword. As for Gabidulin codes, error flips can never result in a 0 distance between the valid codeword and its erroneous version. However, for single bit errors, the distance is always 1 and for two random flips, it is more likely to be 2.

Based on the nature of KK codes and rank metric codes as well as the results of the previous experiments, one may draw the following remarks:

1. KK codes can replace their Gabidulin counterparts at the expense of less cardinality. In this case, however, the error correction capability of the system will increase. In case the cardinality is crucial for the system, more symbols will be required to use

Table 2.2: Occurrence probabilities for the distance between a codeword and its erroneous version after one and two bit flips for both of the $(16, 256, 16, 8)_2$ KK subspace code and the $(16, 65536, 8, 8)_2$ Gabidulin code.

Distance	**Occ. probability (KK)**		**Occ. probability (Gab.)**	
	One flip	**Two flips**	**One flip**	**Two flips**
2	0.698	0.835	0	0.921
1	0.053	0.023	1	0.079
0	0.249	0.142	0	0

subspace codes instead of rank metric codes.

2. KK codes are usually used in situations when the vectors of the transmitted codeword can be combined as in RLNC. In this case, rank metric codes cannot replace them. However, if the vectors of a transmitted codeword are not combined, rank metric codes will increase the number of possible transmitted messages given their higher cardinality at the expense of less correction capability.

3. The effects of single random errors on the distance between a codeword and its erroneous version is generally higher for subspace codes compared to rank metric codes. However, this does not affect the results on the correction capability of the two code families, which is better in the case of subspace codes. In fact, we can see that the effects of random flips are more apparent on Gabidulin codes starting from the case of two random flips.

2.4.3 Orbit codes

Orbit codes [59], [60] are subspace codes that are based on group theory. They are constructed using group actions on the subgroups of the general linear group GL_m defined below.

Definition 20. *The set of full-rank $m \times m$ square matrices on $\mathbb{F}_q$ constitutes the linear group of degree m on $\mathbb{F}_q$, denoted GL_m.*

A formal definition of orbit codes can be formulated as,

Definition 21. *Orbit codes are the orbits of subgroups of the general linear group GL_m on the Grassmanian $\mathcal{G}(n, m)$.*

An interesting quality of orbit codes is that the cardinality of the code is easily obtained in a closed form equality using group theory.

Theorem 4. *Let $V \in \mathcal{G}(n,m)$ and let G be a subgroup of the general linear group GL_m. Consider the orbit code $\mathcal{C} = VG$. The cardinality of $\mathcal{C}$ is then given by,*

$$|\mathcal{C}| = \frac{|G|}{|St_G(V)|} \tag{2.21}$$

Note that other code families can be seen as orbit codes as well. In [59],the authors have proven that KK codes are in fact orbit codes. Moreover, If the acting group is cyclic, the corresponding orbit codes are called cyclic orbit codes. Consider the following example [59].

Let $\mathcal{G}$ be a group generated by G,

$$G = \begin{bmatrix} 0 & 1 & 1 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 1 & 0 & 0 \\ 1 & 0 & 1 & 0 \end{bmatrix}$$

One can check that the order of G is 4. Consider the following matrix U whose row space span a 2-dimensional space from $\mathcal{G}(2,4)$.

$$U = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \end{bmatrix}$$

In this case the orbit code generated by $\mathcal{G}$ and U, will take the form,

$$\mathcal{C} = \{\langle UG^i \rangle | i \in \{1,2,2,4\}\} \tag{2.22}$$

2.5 Summary

Error correction in RLNC is carried out using subspace codes. In this setting, the codewords are vector spaces taken as subspaces from an ambient vector space over the base field used for representing information. The operator channel is used with those codes where it is defined as a mapping from source to destination where insertions and erasures are present. Erasures are characterized by a decrease in the dimension of a transmitted codeword, where an insertion is the opposite scenario, i.e. an increase in dimension.

Subspace codes belong to two families : Lifted Rank-Metric codes and Orbit codes. The

first family depicts constant dimension codes that are constructed by lifting Gabidulin codes. Orbit codes, on the other side, are constructed using right group actions on subgroups of the general linear group GL_m. While orbit codes are endowed with more structure compared to the lifted rank metric codes which has allowed for some advantages such as a closed form expression for the cardinality, they are still young and lack practical uses compared to the lifted rank metric codes. Those latter codes are the ones that are mainly used whenever subspace codes are considered.

In our work we are mainly interested in Lifted Rank-Metric Codes. In the next chapter, those codes will be used to ensure both error correction and data confidentiality.

Chapter 3

Investigation of Error Control in Random Linear Network Coding for Security Against Wiretap Attacks

3.1 Introduction

Error codes and encryption schemes are usually carried out separately. Information is first encrypted and the resulted bit stream will undergo coding operations to allow for data recovery in the case of erroneous data transmission. In RLNC, packets are linearly combined at every encoding intermediate node, which provide intrinsic security against wiretap attacks for eavesdroppers listening on wiretap sets of cardinality inferior to the multicast capacity of the network. The use of RLNC for security purposes is referred to as Secure RLNC (SRLNC) in the literature of Network Coding. Most of SRLNC schemes are based on encrypting or obscuring the coefficient matrix such that a wiretapper with access to C_m channels will have to guess on the global matrix before extracting the source data [4],[6],[5].

Subspace codes for both error correction and security have been used in [61], where the authors proposed a universal security scheme that can be applied to RLNC-based networks, provided that the outer code is a rank-metric code. Rank metric codes can be easily made into subspace codes using lifting as explained in the previous chapter. In [62], the authors introduced new code parameters, the relative dimension/intersection profile (RDIP), which is defined as the greatest dimension of intersections between a code and one of its subcodes for each extension of the underlying finite field up the dimension of the ambient vector space, and the relative generalized rank weight (RGRW), which stands for the dimension of the smallest subspace ensuring a dimension of intersection between a code and its subcode exceeding a predefined value that ranges from 0 to the dimension of the quotient space (code/subcode). In the same paper, they showed that those two parameters can be used to analyze the security performance of the used network code. Similar papers dealing with the problem of communication security with subspace codes for specific constructions are also available in the literature [63],[64].

In this chapter, we investigate random linear network coding security and introduce four secure schemes. The first three schemes are for error-free environments, provided to highlight the intrinsic security of RLNC. In the second and third ones, the security is based on the encryption of the encoding matrix as well as the partial permutation of the data matrix symbols after the application of field homomorphisms to control the number of symbols in the data representation . As for the third one, the idea is to obscure the global encoding matrix via the use of a predefined matrix followed by a permutation step applied on its rows. However, the fourth one is a secure scheme for data security for subspace codes based on a

codeword selection strategy that is referred to as the Subspace Coding Strategy (SCS)[7]. In this scheme, the data stream is permuted according to a permutation key P_k distributed via a Key Distribution Center (KDC) and then encoded into codewords that are chosen from a set of Grassmannian codes using the SCS. This step will increase the ambiguity on the correct codewords used for transmission, resulting in a more secure data transmission. The last step consists of inducing some correctable random errors before injecting the codeword into the network. While the first two schemes are not related to error control, they are provided to highlight the intrinsic security of RLNC upon which secure schemes can be built and which will also be useful in the last scheme.

3.2 On encrypting the coefficient and data matrices

In this section two SRLNC schemes are reviewed. The results in this section are available in [65].

3.2.1 System model

3.2.1.1 Network Topology

The wiretap network model as proposed by [66] will be used to model the topology of the network $\mathcal{N}$. This network may be seen as a quadruple i.e. $\mathcal{N} = (G(V, E), s, D, W)$.

In our system, $G(V, E)$ is an acyclic directed multigraph with two sets: a set V of vertices and a set E of edges. The node $s \in V$ is the source node serving as the origin of all information transmitted across the network. $D \subset V$, on the other side, is the set of legitimate destination nodes or users and $W = \{w | w \subset W\}$ is a collection of sets consisting of wiretap edges. Note that a wiretapper may wiretap the network using only one element of W at a time. In this regard, the number of wiretappers is not limited as long as they do not cooperate.

As for G, The capacity is unity for all the channels in the network. If two nodes require more capacity, parallel edges are used. We define the multicast capacity C_m as being the maximum rate at which the source can send data to the set of destination nodes in D at a given time slot. In the proposed schemes, we consider that $\forall w \in W$, the number of elements is superior or equal to C_m. Note that in this system, the network is assumed error-free and therefore error analysis is excluded for these schemes.

3.2.1.2 Security Model

The source message is divided into a set of m packets that are represented as vectors over a finite field $\mathbb{F}_q$ where $q = 2^r$ and $r \in \mathbb{Z}^+$. r is chosen to be as sufficiently large as possible to minimize the decoding failure probability [16]. Before transmission, those packets are encrypted without affecting the underlying RLNC scheme. Therefore, the intermediate nodes will just perform RLNC on the packets they receive on their incident edges. The destination nodes, on the other side, will decode the received packets using Gauss Elimination and then decrypt the resulted packets to obtain the source message.

In order to have control over the generated coefficients, the source and the destination nodes are equipped with PRNGs. Those generators are synchronized and fed with the same seed that is assumed to be generated from the same True Random Number Generator (TRNG) . Permutation keys (P_k) are also required to be used by the two schemes. Both of the permutation keys and the TRNG seed that is used for both the PRNGs are distributed using a Key Distribution Center (KDC).

3.2.1.3 Definitions

Definition 22. *Let $\mathbb{F}_{q_1}$ and $\mathbb{F}_{q_2}$ be two extensions of the binary field such that $q_1 = 2^{r_1}$ and $q_2 = 2^{r_2}$ where $r_1 = t \cdot r_2$ with $\{r_1, r_2, t\} \subset \mathbb{Z}^+$. The symbol vectorization function $T : \mathbb{F}_{q_1} \longrightarrow \mathbb{F}_{q_2}^t$ is a function transforming a symbol $\alpha \in \mathbb{F}_{q_1}$ to a vector $v \in \mathbb{F}_{q_2}^t$ such that $v = \{\alpha_1, \alpha_2, \cdots, \alpha_t\}$ and the concatenation of the elements of v yields α, i.e, $\alpha = \alpha_1\alpha_2 \cdots \alpha_t$.*

T is an isomorphism from $\mathbb{F}_{q_1}$ to $\mathbb{F}_{q_2}^t$. This function is used to increase the number of symbols in a given data stream, which will evidently increase the overall number of possible permutations of the data symbols. The inverse of T is the T^{-1} transformation.

Definition 23. *Let $\mathbb{F}_{q_1}$ and $\mathbb{F}_{q_2}$ be two extensions of the binary field such that $q_1 = 2^{r_1}$ and $q_2 = 2^{r_2}$ where $r_1 = t \cdot r_2$ with $\{r_1, r_2, t\} \subset \mathbb{Z}^+$. The function $T^{-1} : \mathbb{F}_{q_2}^t \longrightarrow \mathbb{F}_{q_1}$ denotes the inverse of the function $T : \mathbb{F}_{q_1} \longrightarrow \mathbb{F}_{q_2}^t$.*

The combination of T and T^{-1} provides us with the ability to perform data manipulations on different fields without affecting the other operations that are carried out on data in its original representation.

3.2.2 First Encryption Scheme

3.2.2.1 Encryption

The source message is transformed into the matrix $\mathcal{M}_{\mathcal{P}} = \mathcal{P}_{m\times l}$ with m and l denote the number of data packets and their length, respectively. This matrix will be RLNC-encoded using a randomly generated encoding matrix $\mathcal{L}_{m\times m}$. Note that this matrix is generated using the PRNG of the source. Unlike conventional RLNC, the $\mathcal{L}_{m\times m}$ matrix is not transmitted but rather discarded and replaced with the $\mathcal{I}_{m\times m}$ identity matrix. Let $C_{m\times l}$ denote the encoded version of $\mathcal{P}_{m\times l}$. This matrix will undergo a partial permutation after the application of the T transformation ($\mathcal{C}_{m\times l} \mapsto \mathcal{C}_{m\times(l\cdot t)}$). The new matrix will then be treated as being in its original representation by (virtually) applying the T^{-1} transformation. The transmitted matrix is then $\mathcal{M}_{\mathcal{C}} = [\mathcal{I}_{m\times m}\ \mathcal{C}_{m\times l}]$.

Algorithm 1 summarizes the aforementioned steps.

Note that we have adopted the same notation for the matrix , i.e. $\mathcal{C}_{m\times l}$ because it overrides itself at every new step.

Algorithm 1 First Encryption Algorithm

Input: $\mathcal{M}_{\mathcal{P}} = \mathcal{P}_{m\times l}$, $q_1 = 2^{t\cdot r}$, $q_2 = 2^r$, P_k
Output: $\mathcal{M}_{\mathcal{C}} = [\mathcal{I}_{m\times m}\ \mathcal{C}_{m\times l}]$
generate $\mathcal{L}_{m\times m}$;
$[\mathcal{L}_{m\times m}\ \mathcal{C}_{m\times l}] \leftarrow RLNC([\mathcal{I}_{m\times m}\ \mathcal{P}_{m\times l}],\ \mathcal{L}_{m\times m})$;
$\mathcal{C}_{m\times(l\cdot t)} \leftarrow partial\text{-}Permute(T(\mathcal{C}_{m\times l}, q_1, q_2), P_k)$;
$\mathcal{C}_{m\times l} \leftarrow T^{-1}(\mathcal{C}_{m\times(l\cdot t)}, q_1, q_2)$;
$\mathcal{M}_{\mathcal{C}} \leftarrow [\mathcal{I}_{m\times m}\ \mathcal{C}_{m\times l}]$

$T(\mathcal{C}_{m\times l}, q_1, q_2)$ and $T^{-1}(\mathcal{C}_{m\times(l\cdot t)}, q_1, q_2)$ are used to denote $T : \mathbb{F}_{q_1} \longrightarrow \mathbb{F}_{q_2}^t$ and $T^{-1} : \mathbb{F}_{q_2}^t \longrightarrow \mathbb{F}_{q_1}$, as applied to each symbol of the matrix $\mathcal{C}_{m\times l}$ and $\mathcal{C}_{m\times(l\cdot t)}$, respectively. Figure 3.1 is an illustration of the different steps of the encoding process in the first scheme.

3.2.2.2 Decryption

The received matrix at a given sink d will take the form $\mathcal{M}_{\mathcal{R}} = [\mathcal{G}_{m\times m}\ \mathcal{E}_{m\times l}]$ with $\mathcal{G}_{m\times m}$ and $\mathcal{E}_{m\times l}$ being the global encoding matrix and an RLNC–encoded version of $\mathcal{C}_{m\times l}$ respectively.

Once the RLNC decoding step is successfully done, the inverse of the encryption operations performed at the source is carried out in reverse order to extract the source plain data packets. Algorithm 2 depicts the different steps of the decryption process.

Note that $RREF$ stands for the Reduced Row Echelon Form method, $\mathcal{O}_{m\times l}$ denotes the

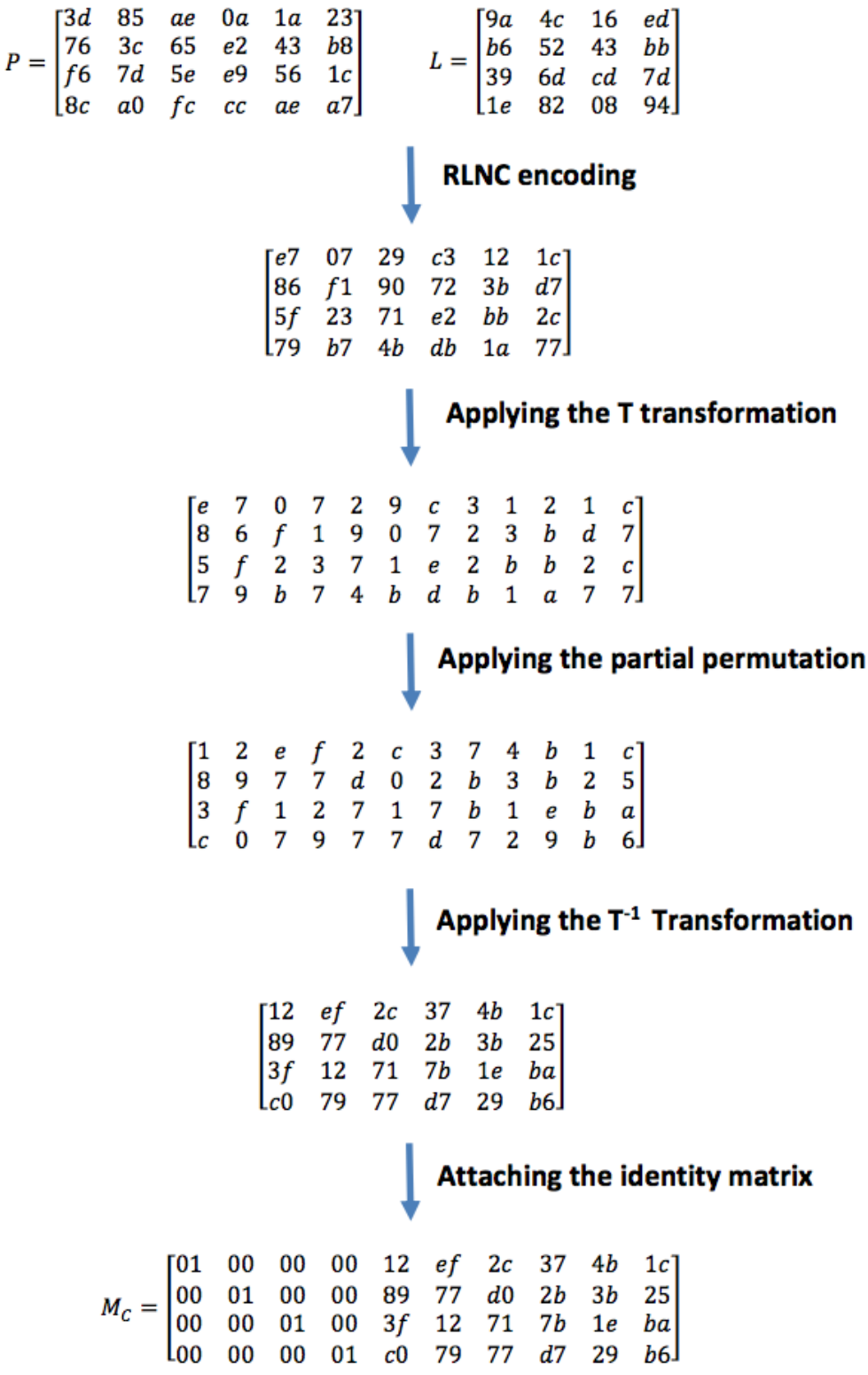

Figure 3.1: An illustration of the encryption process of the first scheme with $q_1 = 256$, $q_2 = 16$.

$m \times l$ zero matrix and $\mathcal{A}_{m\times m}$, $\mathcal{B}_{m\times m}$ are $m \times m$ matrices for storing temporary results. Figure 3.2 is an illustration of the decoding algorithm.

Algorithm 2 Decryption of the First Encryption Algorithm

Input: $\mathcal{M}_{\mathcal{R}} = [\mathcal{G}_{m\times m}\ \mathcal{E}_{m\times l}]$, $q_1 = 2^{t\cdot r}$, $q_2 = 2^r$, P_k
Output: $\mathcal{M}_{\mathcal{P}} = \mathcal{P}_{m\times l}$, $Success$
$Success \leftarrow False$;
$\mathcal{P}_{m\times l} \leftarrow \mathcal{O}_{m\times l}$
$[\mathcal{A}_{m\times m}\ \mathcal{C}_{m\times l}] \leftarrow RREF(\mathcal{M}_{\mathcal{R}})$;
if $Rank(\mathcal{A}_{m\times m}) = m$ **then**
  $\mathcal{C}_{m\times(l\cdot t)} \leftarrow partial\text{-}Permute^{-1}(T(\mathcal{C}_{m\times l}, q_1, q_2), P_k)$;
  $\mathcal{C}_{m\times l} \leftarrow T^{-1}(\mathcal{C}_{m\times(l\cdot t)}, q_1, q_2)$;
  generate $\mathcal{L}_{m\times m}$;
  $[\mathcal{B}_{m\times m}\ \mathcal{C}_{m\times l}] \leftarrow RREF([\mathcal{L}_{m\times m}\ \mathcal{C}_{m\times l}])$;

  if $Rank(\mathcal{B}_{m\times m}) = m$ **then**
    $\mathcal{P}_{m\times l} \leftarrow \mathcal{C}_{m\times l}$;
    $Success \leftarrow True$;

  end
end
$\mathcal{M}_{\mathcal{P}} \leftarrow \mathcal{P}_{m\times l}$;

3.2.3 Second Encryption Scheme

3.2.3.1 Encryption

In this scheme, the source plain data matrix $\mathcal{M}_{\mathcal{P}} = \mathcal{P}_{m\times l}$ will be transformed into an encrypted matrix $\mathcal{C}_{m\times(l+1)}$ and the transmitted matrix will be $\mathcal{M}_{\mathcal{C}} = [\mathcal{I}_{m\times m}\ \mathcal{C}_{m\times(l+1)}]$.

$\mathcal{L}_{m\times m}$ is first generated using the source PRNG and then each vector $v \in \mathcal{L}_{m\times m}$ is attached to one of the packets of $\mathcal{P}_{m\times l}$ to serve as its header as done in conventional RLNC. However, no encoding will be carried out but rather, the RREF will be applied. If after reduction, $\mathcal{L}_{m\times m}$ is transformed into the identity matrix $\mathcal{I}_{m\times m}$, we proceed to the next step. Otherwise, this step will be repeated for $N < q_1$ times at most, where N is the maximum number this step is allowed to be repeated before the encryption stops and returns failure. Note that this should not be the case under ordinary circumstances since the RLNC failure probability is very small given that q_1 is sufficiently large.

The second step, which is the partial permutation, will be identical to that of the first scheme. However, in this time, an extra symbol is concatenated to each encrypted packet from $\mathcal{C}_{m\times l}$ to get $\mathcal{C}_{m\times(l+1)}$. Given that q_1 is sufficiently large and $0 < N < q_1$, the extra

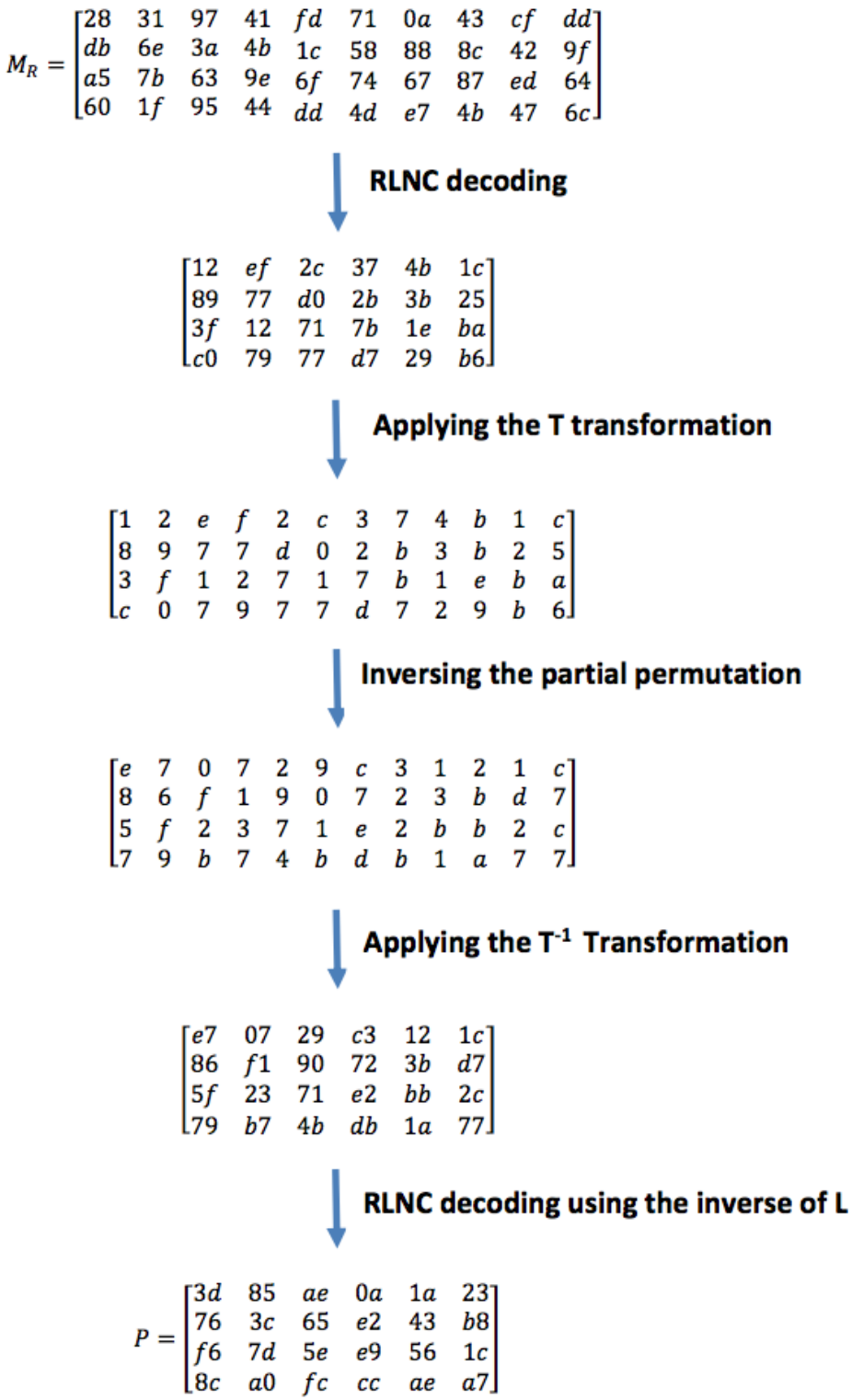

Figure 3.2: An illustration of the decryption process in the first scheme with $q_1 = 256$, $q_2 = 16$.

symbol will suffice to inform the destination nodes of the number of generations carried out at the source to get $\mathcal{L}_{m\times m}$, in order to synchronize the PRNG.

Figure 3.3 shows an example depicting the different steps of the encryption process.

Algorithm 3 Second Encryption Algorithm

Input: $\mathcal{M}_{\mathcal{P}} = \mathcal{P}_{m\times l}$, $q_1 = 2^{t\cdot r}$, $q_2 = 2^r$, P_k, N
Output: $\mathcal{M}_{\mathcal{C}} = [\mathcal{I}_{m\times m}\ \mathcal{C}_{m\times(l+1)}]$, $Success$
$n \leftarrow 0$;
$Success \leftarrow False$;
$\mathcal{M}_{\mathcal{C}} \leftarrow [\mathcal{O}_{m\times m}\ \mathcal{O}_{m\times(l+1)}]$
do
  $n \leftarrow n + 1$;
  generate $\mathcal{L}_{m\times m}$;
  $[\mathcal{A}_{m\times m}\ \mathcal{C}_{m\times l}] \leftarrow RREF([\mathcal{L}_{m\times m}\ \mathcal{P}_{m\times l}])$;
while $Rank(\mathcal{A}_{m\times m}) \neq m$ *and* $n < N$;
if $n \leq N$ **then**
  $\mathcal{C}_{m\times(l\cdot t)} \leftarrow partial\text{-}Permute(T(\mathcal{C}_{m\times l}, q_1, q_2), P_k)$;
  $\mathcal{C}_{m\times l} \leftarrow (T^{-1}(\mathcal{C}_{m\times(l\cdot t)}, q_1, q_2))$;
  $Success \leftarrow True$;
  for $c_{1\times l} \in \mathcal{C}_{m\times l}$ **do**
    $c_{1\times(l+1)} \leftarrow concatenate(n, c_{1\times l})$;
  end
  $\mathcal{M}_{\mathcal{C}} = [\mathcal{I}_{m\times m}\ \mathcal{C}_{m\times(l+1)}]$
end

3.2.3.2 Decryption

After the reception of the encoded packets, a destination node will form the matrix $\mathcal{M}_{\mathcal{R}} = [\mathcal{G}_{m\times m}\ \mathcal{E}_{m\times(l+1)}]$. This matrix will undergo a decoding operation followed by its decryption in order to extract the source message. Upon a successful decoding of $\mathcal{M}_{\mathcal{R}}$, the $\mathcal{L}_{m\times m}$ will be generated by the PRNG using the first symbol of any received packet for synchronization. Those symbols are then discarded resulting in the matrix $[\mathcal{I}_{m\times m}\ \mathcal{C}_{m\times l}]$. An inverse permutation using the key P_k is then applied on $\mathcal{C}_{m\times l}$ over a representation on $\mathbb{F}_{q_2}$ using the T transformation, followed by its inverse T^{-1}. The last step will be to RLNC–encode the resulted $\mathcal{C}_{m\times l}$ with $\mathcal{L}_{m\times m}$.

Figure 3.4 provides an illustration of the different steps of the decryption algorithm of the second scheme. Note that in the provided example, the column vector $[n_1,\ n_2,\ n_3,\ n_4]'$ is the vector resulting from the RLNC encoding of the vector $[n,\ n,\ n,\ n]'$ with $[\cdot]'$ denoting

the transpose of $[\cdot]$.

Algorithm 4 Decryption of the Second Encryption Algorithm

Input: $\mathcal{M}_{\mathcal{R}} = [\mathcal{G}_{m\times m}\ \mathcal{E}_{m\times(l+1)}]$, $q_1 = 2^{t\cdot r}$, $q_2 = 2^r$, P_k
Output: $\mathcal{M}_{\mathcal{P}} = \mathcal{P}_{m\times l}$, $Success$
$Success \leftarrow False$;
$\mathcal{P}_{m\times l} = \mathcal{O}_{m\times l}$;
$[\mathcal{A}_{m\times m}\ \mathcal{C}_{m\times(l+1)}] \leftarrow RREF(\mathcal{M}_R)$;
if $Rank(\mathcal{A}_{m\times m}) = m$ **then**
  choose a row $c_{1\times(l+1)}$ from $\mathcal{C}_{m\times(l+1)}$
  generate $\mathcal{L}_{m\times m}$ *using* $c_{1\times(l+1)}[0]$;

  for $c_{1\times(l+1)} \in \mathcal{C}_{m\times(l+1)}$ **do**
    discard $c_{1\times(l+1)}[0]$;
  end
  $\mathcal{C}_{m\times(l\cdot t)} \leftarrow partial\text{-}Permute^{-1}(T(\mathcal{C}_{m\times l}, q_1, q_2), P_k)$;
  $\mathcal{C}_{m\times l} \leftarrow (T^{-1}(\mathcal{C}_{m\times(l\cdot t)}, q_1, q_2))$;
  $[\mathcal{L}_{m\times m}\ \mathcal{P}_{m\times l}] \leftarrow RLNC([\mathcal{I}_{m\times m}\ \mathcal{C}_{m\times l}],\ \mathcal{L}_{m\times m})$;
  $Success \leftarrow True$;

end
$\mathcal{M}_{\mathcal{P}} \leftarrow \mathcal{P}_{m\times l}$;

3.2.4 Evaluation of the proposed schemes

Let Eve be a wiretapper using a set $w \in W$ with a full knowledge of the system except for the used keys: the permutation key P_k and the source local encoding matrix $\mathcal{L}_{m\times m}$.

3.2.4.1 Computational security

Eve will try to get the source data matrix by guessing on the used encryption keys. Aside from the RLNC decoding required by any receiver including the wiretapper, note that two main operations are required to extract the source plain data. The first one consists of inverting the permutation step, which is identical for both of the schemes. The second step in the first scheme is the application of the Gauss Elimination algorithm on the received matrix, after discarding the identity matrix resulted from the RLNC decoding and attaching the right encoding matrix used at the source. For the second scheme, this last step consists of matrix multiplication with the right matrix, instead of an application of the Gauss Elimination algorithm. Eventually, she will end up with a list of permutation keys P_k and another one for all probable $\mathcal{L}_{m\times m}$ matrices.

For both of the schemes, the permutation step complexity is in the order of $O((m \cdot t \cdot l)!)$ given that the permutation is performed partially on any subset of all the possible subsets

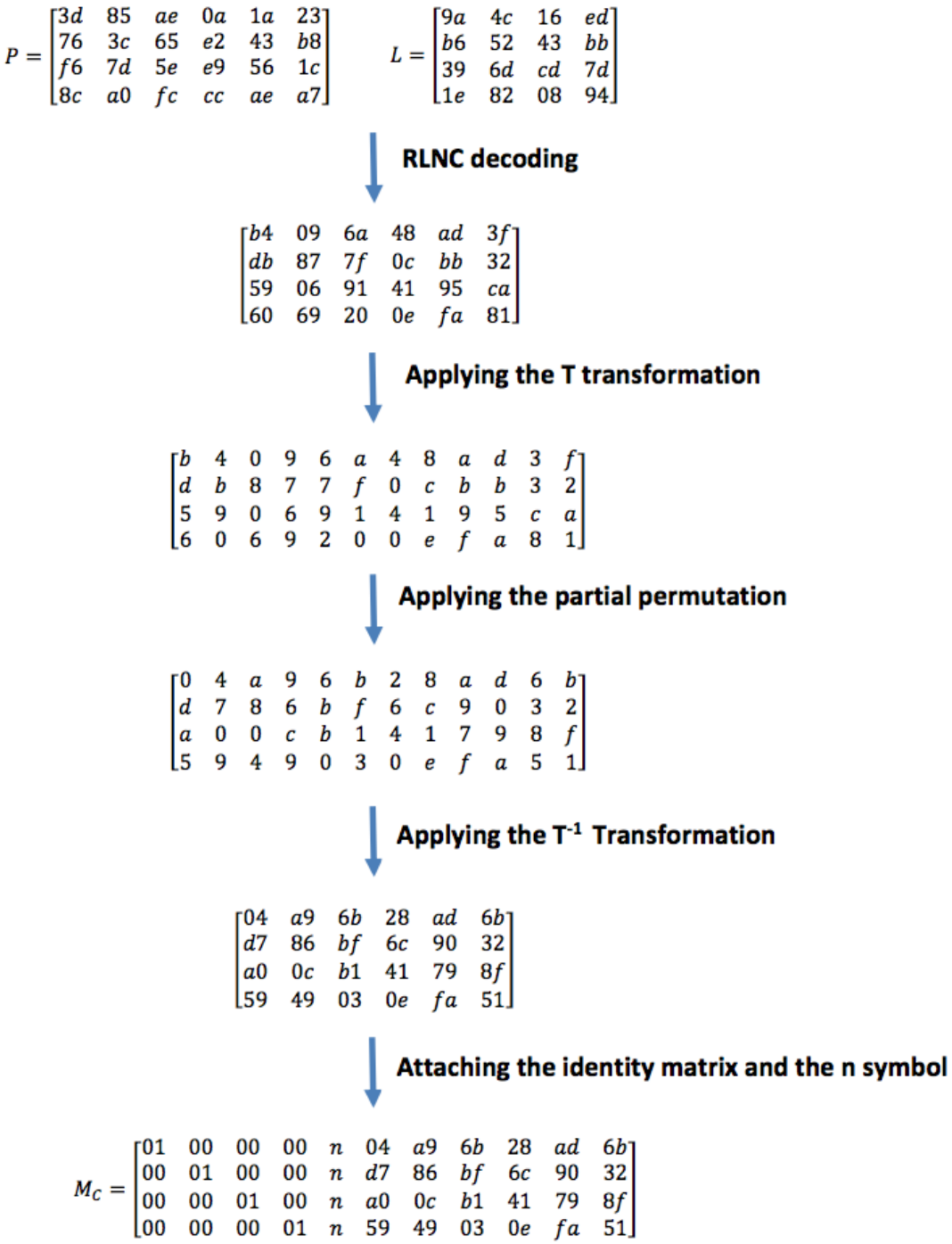

Figure 3.3: An illustration of the encryption process in the second scheme with $q_1 = 256$, $q_2 = 16$.

of the set of the plain data matrix symbols after applying T.

Being aware of the multicast capacity C_m, Eve will create a dictionary of all possible full rank $m \times m$ matrices. Note that from [67], the number N_F of $m \times m'$ matrices of rank R

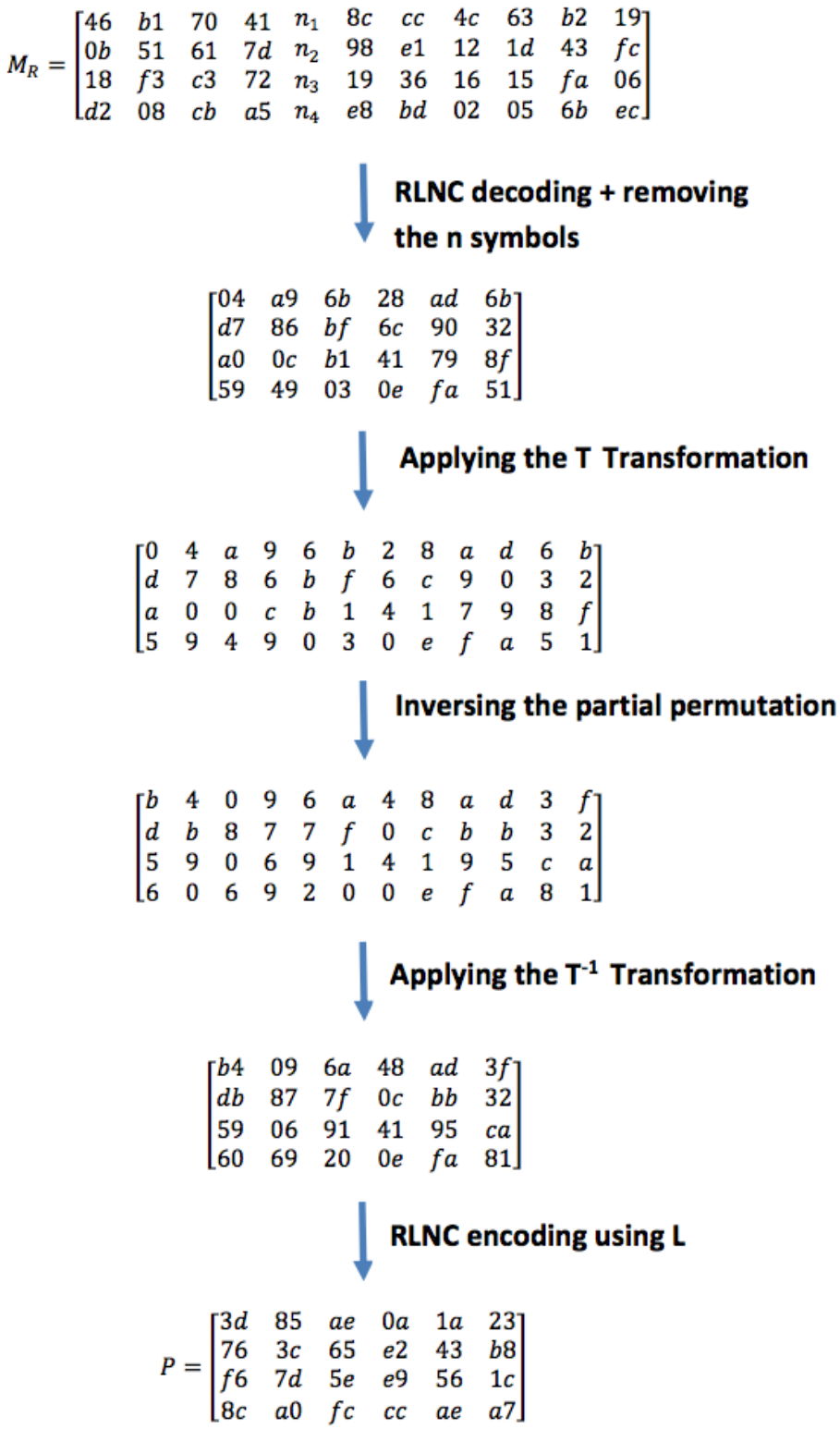

Figure 3.4: An illustration of the decryption process in the second scheme with $q_1 = 256$, $q_2 = 16$.

over $\mathbb{F}_q$ is given by the following equation,

$$N_F = \prod_{i=0}^{R-1} \frac{(q^m - q^i)(q^{m'} - q^i)}{(q^R - q^i)}, \tag{3.1}$$

In our case $m = m' = R$, and therefore (3.1) is simplified to,

$$N_F = \prod_{i=0}^{m-1} (q^m - q^i). \tag{3.2}$$

Guessing the right encoding matrix will incur a complexity in the order of $O(q^{m^2})$. Each one of those matrices will undergo a Gauss Elimination algorithm of complexity $O(m^3)$ or $O(m^2 \cdot l)$ depending on the used scheme. To get the overall complexity, one should not forget the RLNC decoding complexity which is in the order of $O(m^3)$. The overall computational complexity for the two schemes is, therefore, in the order of $O(m^3 \cdot q^{m^2} \cdot (m \cdot t \cdot l)!)$ for the first scheme and $O(m^3 + m^2 \cdot l \cdot q^{m^2} \cdot (m \cdot t \cdot l)!)$ for the second one.

Table 1 compares the two schemes to P–Coding [3] and SPOC [4] in terms of computational complexity, where we clearly notice that both of the schemes outperform P–Coding and SPOC in terms of the complexity associated with an exhaustive search attack.

Table 3.1: Computational Complexity for the proposed schemes, P–Coding and SPOC.

Scheme	**Complexity**
$1^{st} Scheme$	$O(m^3 \cdot q^{m^2} \cdot (m \cdot t \cdot l)!)$
$2^{nd} Scheme$	$O(m^3 + m^2 \cdot l \cdot q^{m^2} \cdot (m \cdot t \cdot l)!)$
P–Coding	$O(m^3 \cdot (m + l)!)$
SPOC	$O(m^3 \cdot q^{m^2})$

3.2.4.2 Guess probability

In a guessing attack, Eve will attempt to get the required security parameters for a successful decryption at random from a single guess. For more on the guess probability, see the appendix. For both of the schemes, to decrypt the encrypted matrix, the wiretapper requires knowledge of the used L matrix with a total of $\prod_{i=0}^{m-1} (q^m - q^i)$ possibilities as well as the used permutation key with a total $(m \cdot t \cdot l)!$ of possibilities , making the effective key

space K_s being,

$$K_s = ((m \cdot t \cdot l)!)(\prod_{i=0}^{m-1} (q^m - q^i) \tag{3.3}$$

Yielding a guess probability of,

$$P_g = \frac{1}{((m \cdot t \cdot l)!)(\prod_{i=0}^{m-1} (q^m - q^i)} \tag{3.4}$$

Figures 3.5, 3.6, and 3.7 provide a comparison between the two schemes reviewed in this chapter, SPOC, and P–Coding in terms of the guess probability of Eve with respect to the field size, the multicast capacity, and the value of t. . Our schemes have the same level of guess probability, which is highlighted in the aforementioned figures via their identical (overlapping) curves. As shown in the figures, they both outperform P–Coding and SPOC almost all the time. Note that P–Coding has a constant guess probability throughout the three experiments because it does not depend on the field size but rather on the number of symbols on the data packets, i.e, l.

3.2.4.3 Confusion and diffusion

Confusion and diffusion are two measures to evaluate the correlation between the key, plaintext, and ciphertext. A cipher with good confusion would require each symbol in the ciphertext to depend on many symbols from the key, and therefore one symbol change in the key should affect a large number of symbols in the ciphertext. Diffusion, on the other side, deals with how much the plaintext bits are spread on the ciphertext. In a cipher with good diffusion, one expects that each symbol in the ciphertext depends on many symbols from the plaintext.

Based on matrix algebra, one symbol change in the plain data matrix would induce at most m changed symbols in the encoded data matrix (or one entire column), while one symbol change in the encoding matrix will result in n changed symbols at most (or one entire row). This clearly holds for SPOC, given that the entire encryption process is based on RLNC. The same remark may be drawn about P–Coding, given that the permutation step does not change any symbols. Concerning our proposed schemes, an increase in the changed symbols is expected due to the T transformation and the Gauss Elimination step in the second scheme. In Table 2, we notice an improvement of the percentage of the changed symbols in the two proposed schemes as compared to SPOC and P–Coding. This is a result

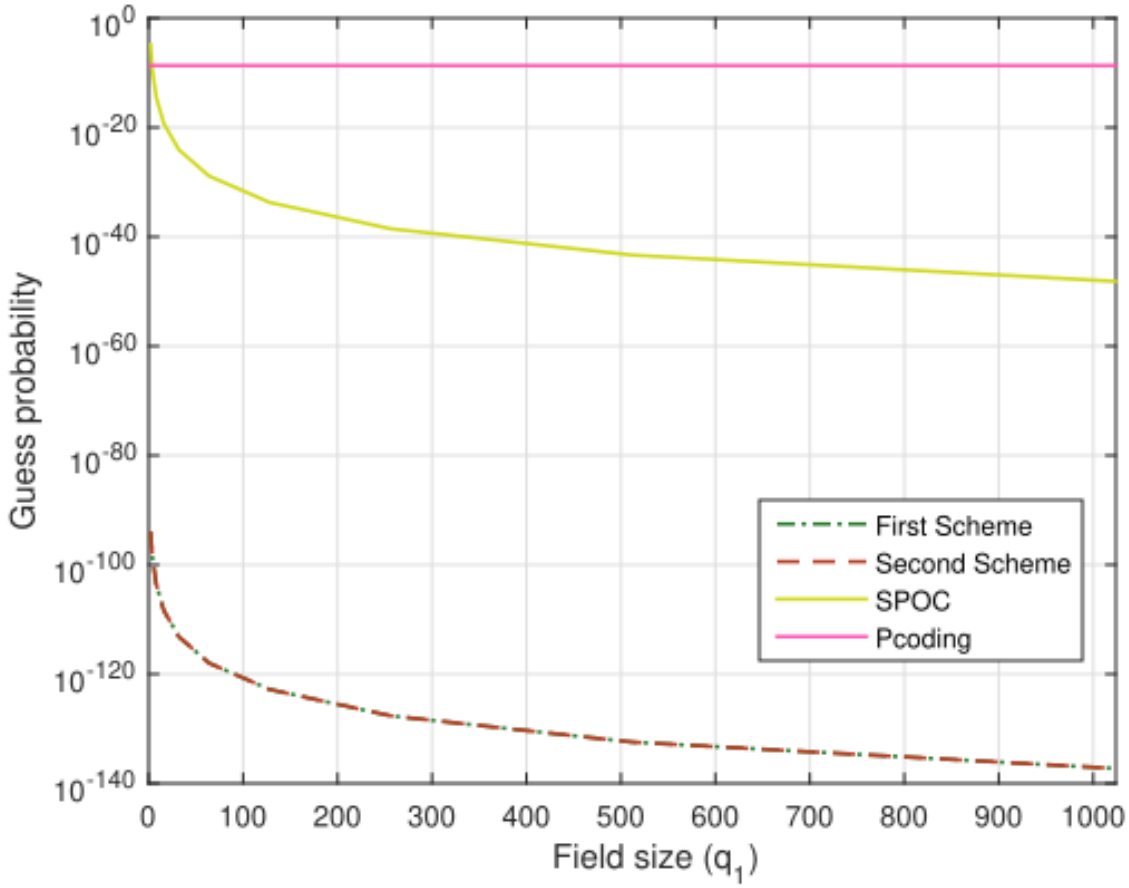

Figure 3.5: The comparison of the guess probability of a wiretapper vs the field size (q_1)as the source alters his security scheme between the first scheme, the second scheme, SPOC and P–Coding with $m = 4$, $t = 2$ and $l = 8$.

of the used T transformation. Along with this latter factor, in Table 3, the effect of the Gauss Elimination process is quite apparent, since one symbol change in the key has the potential to change the entire encrypted data matrix symbols.

Table 3.2: Average percentage of changed symbols in the encrypted data matrix when one symbol is changed in the plain data matrix for an 8×16 matrix over $\mathbb{F}_{2^8}$ with $t = 2$ for 1000 iterations.

Scheme	**Changed symbols (%)**
1^{st} *scheme*	24.56
2^{nd} *scheme*	24.56
P–Coding	6.231
SPOC	6.231

3.2.4.4 Computational Complexity and packet overhead

In this subsection, we discuss the computational complexity and the RLNC packet overhead, which is the number of symbols added to the traditional RLNC packet, as required by

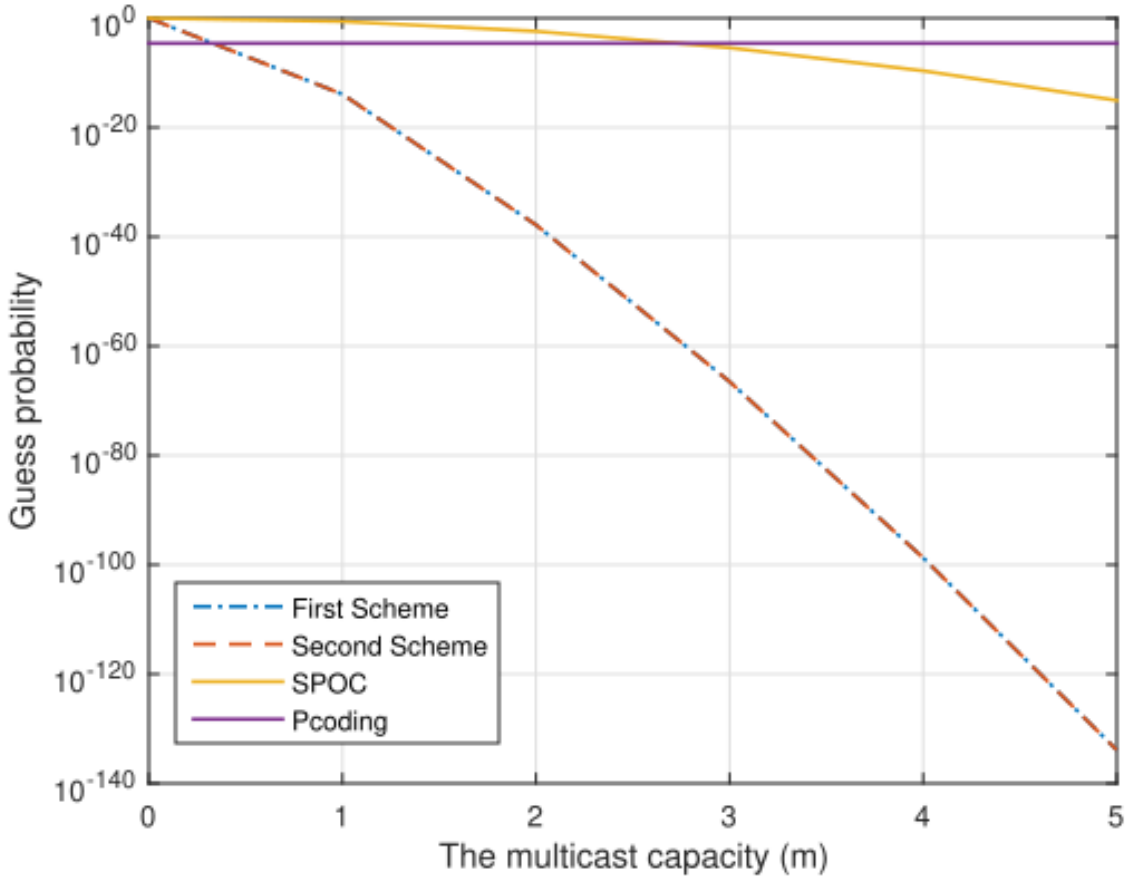

Figure 3.6: The comparison of the guess probability of a wiretapper vs the multicast capacity as the source alter his security scheme between the first scheme, the second scheme, SPOC and P–Coding with $q_1 = 4$, $t = 2$ and $l = 8$.

Table 3.3: Average percentage of changed symbols in the encrypted data matrix when one symbol is changed in the encoding matrix for an 8×16 matrix over $\mathbb{F}_{2^8}$ with $t = 2$ for 1000 iterations.

Scheme	**Changed symbols (%)**
1^{st} *scheme*	43.28
2^{nd} *scheme*	96.13
P–Coding	12.27
SPOC	12.27

the used scheme.

Let $k = m + l$. the overall computational complexity resulted from P–Coding is in the order of $O(m \cdot k + m^2 \cdot l)$ and that of SPOC is in the order of $O(m^3 + m^2 \cdot l + M \cdot N \cdot m)$ where M is the average number of re–encoding operations and N is the average number of packets re–encoded at a given intermediate node. As for the two schemes discussed in this chapter, the first scheme has a total complexity in the order of $O(m^3 + m^2 \cdot l + t \cdot m \cdot l)$ where the

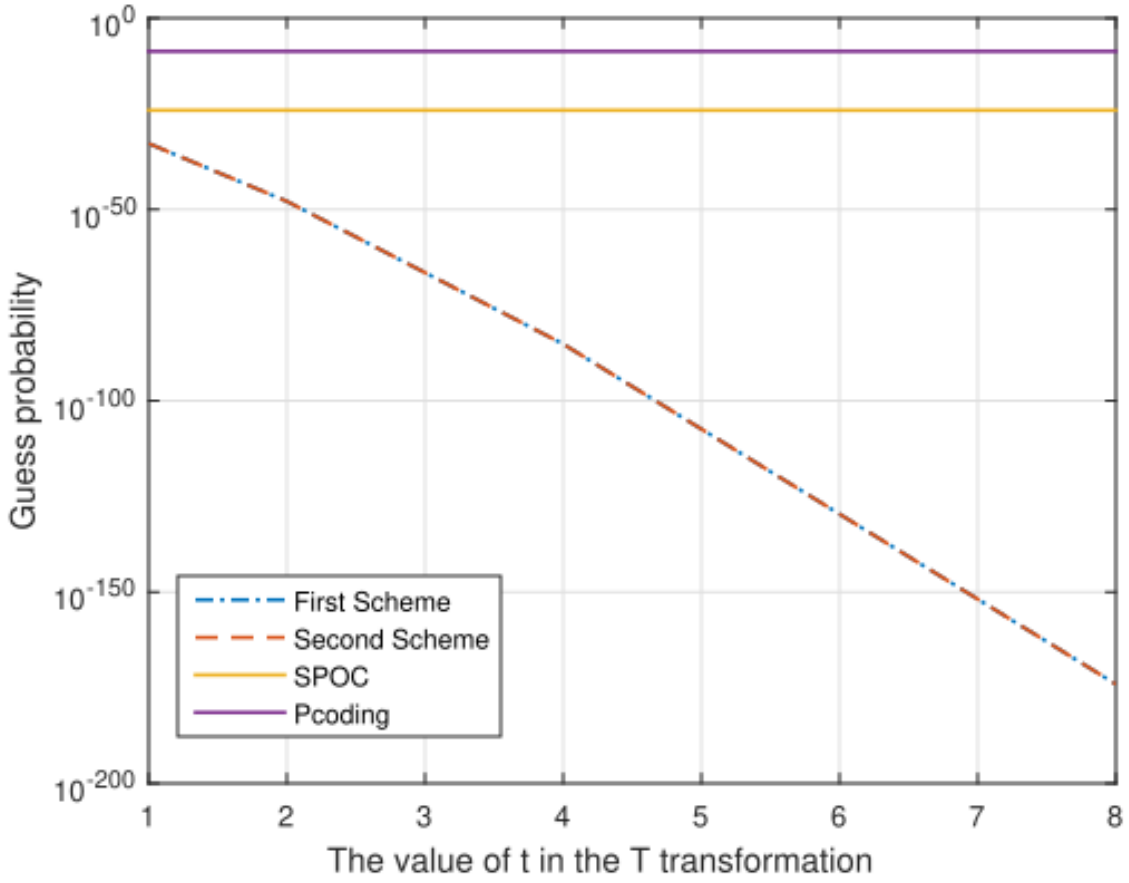

Figure 3.7: The comparison of the guess probability of a wiretapper vs the value of t as the source alter his security scheme between the first scheme, the second scheme, SPOC and P–Coding with $m = 4$, $q_1 = 32$ and $l = 8$.

first term is related to the decoding step, the second term is related to the RLNC encoding operation at the source and the last term is related to the permutation operations at the source and sink nodes. This latter term may be ignored given that t is a constant, which yields a complexity of $O(m^3 + m^2 \cdot l)$. Similarly, the second scheme has a complexity in the order of $O(m^3 + m^2 \cdot l + M \cdot N)$. The second term is due to the re-encoding operations on the extra symbol required by the second scheme, similar to the SPOC scheme.

Table 4 shows the resulted run-times for our proposed schemes, SPOC, P–Coding, and AES. Taking P-coding as a reference, one can notice that the two schemes consume 15% and 27% more time compared to that required by P–Coding. SPOC, on the other hand, requires a 57% increase and AES has an encryption time in the order of 423%. This experiment confirms the lightweight nature of the two schemes. The results of this experiment are intuitive since permutations are just memory re-indexing which makes P–Coding the fastest amongst all 5 schemes. In fact, in [3], the authors have already compared P–Coding with AES, in which

its lightweight nature has been verified. As for our proposed schemes, the second scheme is based on Gauss Elimination which consumes more time than a matrix multiplication with the same matrix dimension. In SPOC, in addition to the matrix multiplication, the encoding matrix is encrypted using AES, which is known to be time consuming [3].

Table 3.4: Average consumed time required for the encryption of 512 bytes in Windows 7 running on an Intel i5-2430M machine.

Scheme	**Time (μs)**
$1^{st} scheme$	1.15
$2^{nd} scheme$	1.27
P–Coding	1.00
SPOC	1.57
AES	4.23

Regarding packet overhead, as shown in Table 5, the first scheme does not incur any extra packet overhead whereas the second scheme incurs a small amount of overhead, which is equal to unity, as a result of the extra symbol inserted for PRNG synchronization.

Table 3.5: Packet overhead resulted from the proposed schemes, P–Coding and SPOC.

Scheme	**Packet overhead**
$1^{st} Scheme$	0
$2^{nd} Scheme$	1
P–Coding	0
SPOC	m

3.2.4.5 Decoding failure probability

In this experiment, the authors considered a network with a single source s, 3 destination nodes, 11 (encoding) intermediate nodes, and with a multicast capacity $C_m = 46$. 46 source data packets are transmitted unencrypted, encrypted using the first scheme, the second scheme, SPOC and P–Coding. Randomness has been provided via The Boost Random Library while the permutation keys have been generated using the chaotic permutation algorithm depicted in [68]. The source packets are re-encoded using RLNC as they travel across the network and the sink nodes will try to decode them upon reception. To have an estimation of the decoding failure probability, This experiment is repeated 100 times with different field sizes and with the same $q_2 = 4$. Figure 3.8 shows the results obtained from this experiment.

As clearly shown in Figure 3.8, P–Coding has the same level of decoding failure probability as conventional RLNC, which is expected given that the permutation does not affect

the RLNC encoding operations. Using SPOC or one of the two proposed schemes slightly affects the decoding failure probability at $q = 8$. This difference is almost non-existent as q gets bigger. This fact has been already explored in the literature, and it is related to the idea that for larger q, the probability of a randomly chosen coefficient producing a rank deficiency gets smaller.

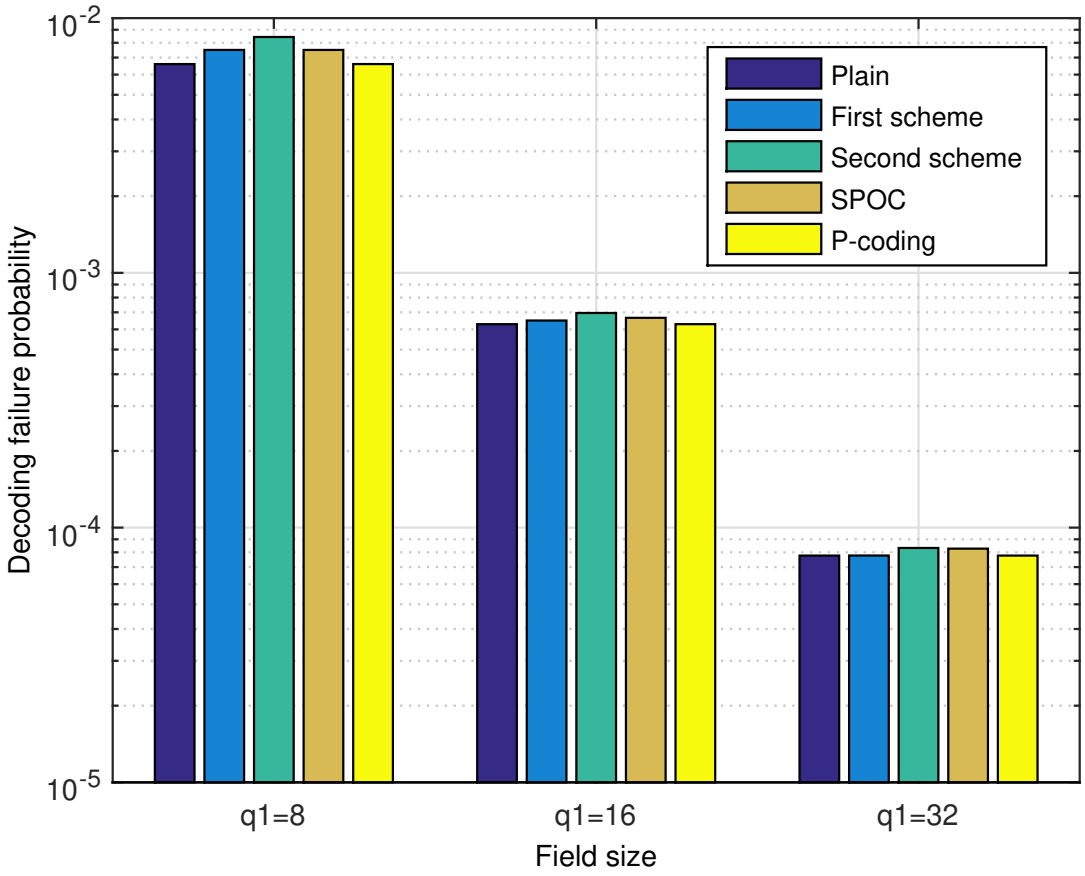

Figure 3.8: Decoding failure probability Vs. Field size (q_1) for data transmission using plain RLNC, SPOC, P–Coding and the proposed schemes.

3.3 On encrypting the coefficient matrix in RLNC

Another scheme that we proposed for providing data confidentiality against wiretap attacks was provided in [5], where a lightweight encryption algorithm for networks using Random Linear Network Coding (RLNC) has been proposed. The system model used for this scheme is the same as the one used in the previous section.

3.3.1 The Proposed Algorithm

We let $\mathcal{E} = \{e_1, e_2, \cdots, e_{|\mathcal{E}|}\}$ be a subset of $\mathbb{F}_q^{m \times n}$, that is, the set of $m \times m$ matrices over $\mathbb{F}_q$, and let $\mathcal{P} = \{p_1, p_2, \cdots, p_{|\mathcal{P}|}\}$ be a set of permutation keys. $\mathcal{E}$ and $\mathcal{P}$ are stored at the

KDC. Note that the elements of $\mathcal{E}$ are deemed full-rank matrices. The source and sink nodes will be using the same key $k_{en} \in \mathcal{E} \times \mathcal{P}$ for the encryption and decryption operations.

3.3.1.1 Encryption

Let A be the plain data matrix and let $A_{aug} = [I\ A]$ be its augmented version. The effective key $k_{en} = (e, p_k)$ is to be distributed by the KDC and it is made up of the matrix e and the permutation key p_k that are required for both of the encryption and decryption operations. A_{aug} will be first multiplied by e, which yields $[e\ eA]$. The rows of e are then permuted using p_k. An RLNC encoding operation is then applied on the new version of the augmented matrix using a local $m \times m$ matrix L_m of random coefficients. The rationale behind this operation is to obscure the coefficient matrix used at the source node, resulting in a meaningless decoding operation without prior knowledge of k_{en}. The algorithm is summarized as follows

Algorithm 5 Encryption Algorithm

Input: $\mathcal{A}_{aug} = [\mathcal{I}_{m\times m}\ \mathcal{A}_{m\times n}]$, p_k, $e_{m\times m}$
Output: $\mathcal{A}_{Enc} = [\mathcal{R}_{m\times m}\ \mathcal{C}_{m\times n}]$
$\mathcal{C}_{m\times n} \leftarrow e_{m\times m}\mathcal{A}_{m\times n}$

$\mathcal{R}_{m\times m} \leftarrow permute(e_{m\times m}, p_k)$
generate $\mathcal{L}_{m\times m}$

$\mathcal{C}_{m\times n} \leftarrow \mathcal{L}_{m\times m}\mathcal{C}_{m\times n}$

$\mathcal{R}_{m\times m} \leftarrow \mathcal{L}_{m\times m}\mathcal{R}_{m\times m}$
$\mathcal{A}_{Enc} \leftarrow [\mathcal{R}_{m\times m}\ \mathcal{C}_{m\times n}]$

3.3.1.2 Decryption

Upon receiving m linearly independent packets, a sink node will create a matrix $B = [R'\ C']$ where $R' = Re_p$ and $C' = ReA$ with R being the global encoding matrix as seen by the sink node and e_p is the permuted version of e. Using k_{en}, the sink node will then try to reverse all the transformations on A_{aug} to obtain the source message. The following algorithm depicts the required steps for a successful decryption.

Algorithm 6 Decryption Algorithm

Input: $\mathcal{B} = [\mathcal{R'}_{m\times m}\, \mathcal{C'}_{m\times n}]$, p_k, $e_{m\times m}$
Output: $\mathcal{A}_{aug} = [\mathcal{I}_{m\times m}\, \mathcal{A}_{m\times n}]$
$e_{p_{m\times m}} \leftarrow permute(e_{m\times m})$
$\mathcal{R}_{m\times m} \leftarrow \mathcal{R'}_{m\times m} e^{-1}_{p_{m\times m}}$

$\mathcal{A}_{m\times n} \leftarrow e^{-1}_{m\times m} \mathcal{R}^{-1}_{m\times m} \mathcal{C'}_{m\times n}$

$\mathcal{A}_{Aug} \leftarrow [\mathcal{I}_{m\times m}\, \mathcal{A}_{m\times n}]$

3.3.2 Evaluation of the Proposed Algorithm

The proposed algorithm is evaluated in terms of its resilience against wiretap attacks as well as its performance as compared with the SPOC and P-Coding.

3.3.2.1 Computational Security

We also consider a wiretapper Eave as in the previous schemes. Eave wiretaps the network using an element $w \in W$ and tries to collect any meaningful information injected by the source into the network. The search space of Eave depends on its wiretapping capacity i.e. $|w|$, the multicast capacity C_m as well the number of linearly independent vectors amongst the wiretapped ones. The worst case scenario for the system's security is when $|w| = C_m$ and all the wiretapped vectors are linearly independent. In this case, the wiretapper will have to guess only on the used element $k_{en} = (e, p_k)$. For Eave, e can be any full-rank $m \times m$ matrix. The number of such matrices is

$$N = \prod_{i=0}^{m-1} (q^m - q^i) \tag{3.5}$$

where q is the size of the underlying field.

The computational complexity resulted from the number of all possible matrices e will be $O(q^{m2})$. The number of all possible permutation keys will be $O(m!)$. For every guess on k_{en}, the wiretapper has to perform some matrix arithmetic to find the inverse of the permuted version of e and to apply the decoding step, each with a complexity of the order $O(m^2 n)$. The overall complexity will be much reduced If Eave chose to discard the coefficient matrix and try all the possible full-rank matrices. This approach will yield a search complexity in the order of $O(q^{m2})$. Note that this computational complexity would be much worse if $|w| < C_m$ or the wiretapped vectors are not linearly independent since Eave will have to guess on the lost information as well. Table I compares the proposed algorithm to both

SPOC and P-coding in terms of computational complexity arising from the search space for the worst case scenario.

Table 3.6: Computational Complexity for the Proposed Algorithm, P–Coding and SPOC

Algorithm	**Complexity**
The proposed algorithm	$O(q^{m^2})$
P–Coding	$O((m+n)!)$
SPOC	$O(q^{m^2})$

As shown on the table, the minimum computational complexity guaranteed for a brute force attack for the proposed algorithm is the same as that provided by SPOC.

3.3.2.2 Guess probability

In order to extract the source plain matrix, the wiretapper requires knowledge of the used e matrix with a total of $\prod_{i=0}^{m-1}(q^m - q^i)$, making the effective key space K_s being,

$$K_s = (\prod_{i=0}^{m-1}(q^m - q^i) \tag{3.6}$$

Yielding a guess probability of,

$$P_g = \frac{1}{\prod_{i=0}^{m-1}(q^m - q^i)} \tag{3.7}$$

FIGURE 3.9 and FIGURE 3.10 compare the guess probability of the wiretapper Eave as a function of the multicast capacity and the field size. Both of the proposed algorithm and SPOC enjoy the same level of guess probability that exponentially decreases as a function of the multicast capacity C_m and the field size q. P–coding on the other side depends only on the number of columns of the augmented matrix A_{aug}.

3.3.2.3 Performance Analysis

In this subsection, the three algorithms are compared in terms of the amount of computational complexity required for achieving the intended security as well as the packet overhead, which is the amount of bits added to the augmented packets to ensure successful decoding at the legitimate destination nodes. The proposed algorithm requires two matrix multiplications for the encryption and encoding as well as another three ones for the decryption and decoding at the level of legitimate destination nodes, each multiplication is with a complexity in the order of $O(m^2n)$, The permutation step has a linear complexity in the

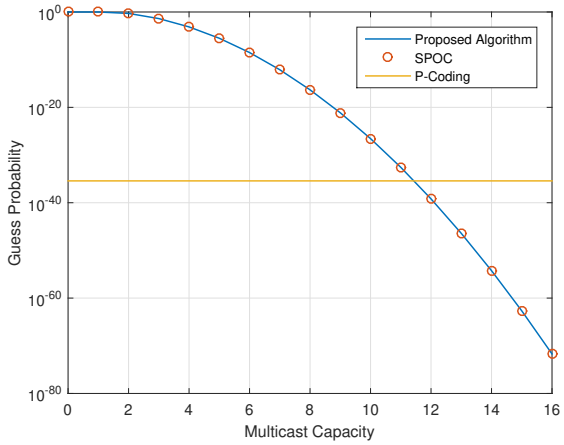

Figure 3.9: Guess probability Vs. multicast capacity for the proposed algorithm, SPOC and P–coding with $q = 2$ and $m + n = 32$.

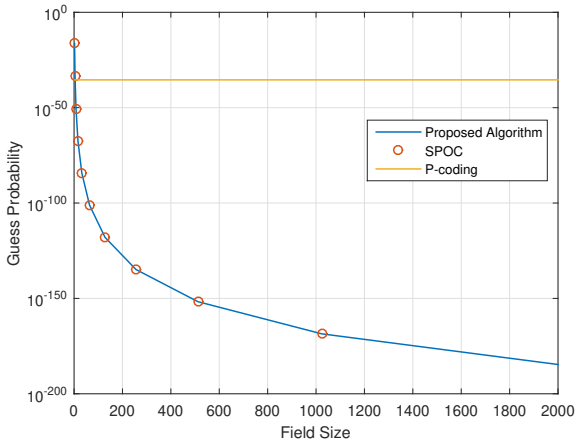

Figure 3.10: Guess probability Vs. field size for the proposed algorithm, SPOC and P–coding with m=8 and n=24.

order of $O(m)$ since it is only a memory swap. At the level of the intermediate nodes, the packets will be just RLNC encoded and therefore no computational complexity is induced by the algorithm at the level of intermediate nodes. P–coding, on the other hand, requires one matrix multiplication operation at the source node and another one at the destination nodes, each with a computational complexity in the order of $O(m^2n)$ along with a permutation operation with a complexity $O(m + n)$. The spoc algorithm is similar to the proposed algorithm in terms of the number of matrix multiplications at the source and destination nodes. However, this algorithm has an extra induced computational complexity at the level of encoding intermediate nodes due to the transmission of the encrypted coefficients.

As for the transmission overhead, the proposed algorithm does not incur any extra control information compared to conventional RLNC–encoded packets as shown in Table II.

Table 3.7: Packet Overhead Resulted From the Proposed Algorithm, P–Coding and SPOC.

Algorithm	**Packet overhead**
The proposed algorithm	0
P–Coding	0
SPOC	m

3.4 Subspace coding based secure RLNC

In the previous section, we have reviewed two SRLNC schemes that mainly make use of the security provided by the encryption matrix to ensure data confidentiality. However, both of those schemes assume error-free environment, which suggests that their deployment in real world scenarios requires the existence of another layer of error correction. In this section, we investigate another scheme that provides both error correction and secure data transmission for RLNC-based networks.

3.4.1 System Model

The same wiretap network used in the previous section is adopted here with slight notation changes concerning the network : $\mathcal{N} = (G, s, U, W_e)$. U is used for the destination nodes or valid users and W_e is the collection of sets of the wiretap edges.

For data transmission, G will maintain all of its characteristics. However, in the background an extra node that will be serving as the Key Distribution Center (KDC) is added to the network. As per wiretapping, an illustration of this network is shown on FIGURE 3.11.

a wiretapper is limited to using one single element $W \in W_e$ at a time, such that $|W| < C_m$. Since the wiretappers are working independently and without cooperation, we will focus our analysis on a single wiretapper that will be referred to as Eve.

The source information is a binary stream D of n_D bits. The source wants to multicast D to the set of sinks in U using RLNC as an inner code and a subspace code as an outer code to allow for error correction in the network. a security layer is also added to counterpart any potential wiretapping attacks.

D will be divided into a set of bitstrings, each of length m. If m does not divide n_D, padding will be used to ensure that all bitstrings are of equal length m. A permutation using a key P_k, distributd via the KDC, is then applied on those bitsrings. Finally, each permuted bitsring will be encoded using a collection $\mathcal{S}_c$ of subspace codes. Moreover, the source will adopt the notion of induced correctable errors, in which a set of valid vectors

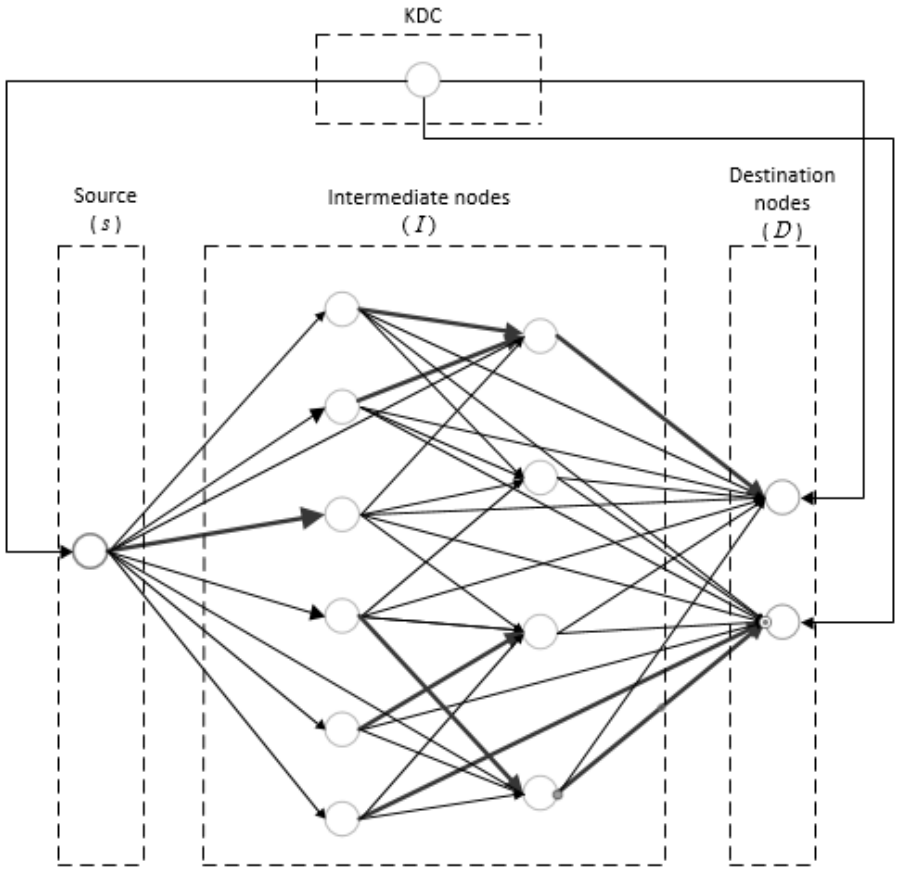

Figure 3.11: A network with one source, 10 intermediate nodes, two destination nodes and a Key Distribution Center, with a multicast capacity $C_m = 8$.

in a codeword c will be replaced by a randomly generated set of erroneous vectors up to a predefined threshold. The erroneous version of c will be RLNC-encoded and injected into the network.

A user $u \in U$ receives an RLNC-encoded version of c, with some errors, on which decoding is applied, followed by an inverse permutation in order to get the original bitstring. Once all those bitstrings are available, they are made into a stream that will be taken as the source message.

The wiretapper Eve is aware of $\mathcal{S}_c$ and C_m. Let $W \in W_e$ denote Eve's wiretap edges. The number $|W|$ of Eve's collected vectors will be referred to as the eavesdropping capacity and will be denoted by C_e with,

$$C_m = C_e + S_0. \tag{3.8}$$

We call S_0 the secure rate offset.

3.4.2 The subspace coding scheme

This scheme aims to provide security against wiretapping attacks for subspace encoded data in RLNC-based networks. It requires some operations to be carried out at the Key Distribution Center, the source as well as destination nodes.

3.4.2.1 At the Key Distribution Center

The role of the Key Distribution Center (KDC) lies in dispatching the security parameters and their eventual updates as required by the scheme. The design of the subspace codebook follows the following algorithm.

1. Choose the targeted $\widehat{C}_e$. where $\widehat{C}_e$ is a guess on the maximum wiretapping capacity C_e of Eve.

2. Choose the length m of the permutation keys and the set $\mathcal{S}_c = \{\mathcal{C}_k \subseteq \mathcal{G}(k,n) \mid C_m - \widehat{S}_0 + 1 \leqslant k \leqslant C_m\}$ of Grassmannian codes in the projective space $\mathcal{P}(n)$ with $\widehat{S}_0$ being the guess on the secure rate offset resulted from the guess on the wiretapping capacity. The selection of those parameters have to satisfy the following conditions:

   (a) The number of symbols in $A = \sum_{k=\widehat{C}_e+1}^{C_m} |C_k|$ has to satisfy: $A \geq 2^m$.

   (b) $\forall C_k \in \mathcal{S}_c$, $d_{C_k} \geq 2\widehat{C}_e + d_{min}$, where d_{C_k} is the minimum subspace distance of the code C_k and d_{min} is the minimum distance required for optimal error correction on the network.

3. Let $\mathcal{S}_{sym} = \bigcup_{k=\widehat{C}_e+1}^{C_m} \mathcal{C}_k$. Randomly create a surjective mapping $\theta : S_{sym} \longrightarrow \{0,1\}^m$.

The KDC also creates the set E_T of the encrypted tags such that there is a bijective mapping $\phi : \{\widehat{C}_e + 1, \widehat{C}_e + 2, \cdots, C_m\} \longrightarrow E_T$. Those tags carry information about the dimension of the transmitted codeword.

For security purposes, E_T, θ and ϕ are updated regularly by the KDC depending on its security policy. Note that the use of the encrypted tags requires the adoption of CDC instead of MDC in the code structure.

3.4.2.2 At the source

Three steps are required at the source:

1. D Permutation

D is divided into a set of l permutable bitstrings, each of length m. Those strings will undergo a permutation using a permutation key P_k delivered by the KDC. Note that if the

last string has a length inferior to m, padding will be used. At the end of this step, the strings are concatenated again to form D_p, the permuted version of D.

2. Data Encoding

In RLNC, we usually opt for the maximum transmission rate C_m. While this will allow for optimal throughput, it results in a vulnerability that may be exploited by potential wiretappers. If Eve succeeds in decoding one generation, she will not need to guess on the dimension again but only on the basis. To solve this problem, we will be using a varying transmission rate R with,

$$C_e + 1 \leq R \leq C_m. \tag{3.9}$$

Each permuted string out of the l substrings will be transformed into a codeword. Those codewords will also be referred to as generations since, at every transmission round, a codeword is sent. The used subspace codes, and consequently the codewords, differ in dimension as specified by equation (3.9), with the possibility that many codewords may be used to represent the same information. This variety in codeword dimension will increase the data transmission security by increasing Eve's search space.

This step is based on Eve's eavesdropping capacity C_e. However, due to the passive nature of wiretapping attacks, Eve is essentially undetectable and therefore the KDC cannot deterministically figure out C_e. Consequently, the KDC will decide a priori on the guaranteed level of transmission security by specifying a value for C_e as indicated in the previous subsection. This value is denoted by $\widehat{C}_e$. The secure rate offset in this case will be denoted by $\widehat{S}_0$. The source then alters the transmission rate R in the following interval,

$$\widehat{C}_e + 1 \leq R \leq C_m. \tag{3.10}$$

This is achieved using the Subspace Coding Strategy (SCS) defined below.

Definition 24. *Given a network with a multicast capacity C_m and a substring length of m, the subspace coding strategy (SCS) is a quintuple $(\mathcal{P}(n), \widehat{S}_0, \mathcal{S}_c, \theta, \phi)$. where,*

(i) $\mathcal{P}(n)$ *is the projective space of dimension n over $\mathbb{F}_q$.*

(ii) $\widehat{S}_0$ *is a positive integer called the secure rate offset.*

(iii) $\mathcal{S}_c = \{\mathcal{C}_k \subseteq \mathcal{G}(k,n) \mid C_m - \widehat{S}_0 + 1 \leqslant k \leqslant C_m\}$ *is a set of Grassmannian codes in $\mathcal{P}(n)$.*

(iv) $\theta : S_{sym} \longrightarrow \{0,1\}^m$ *is a randomly generated surjective mapping with* $\mathcal{S}_{sym} = \bigcup_{k=\widehat{C}_e+1}^{C_m} \mathcal{C}_k$.

(v) $\phi : \{\widehat{C}_e + 1, \widehat{C}_e + 2, \cdots, C_m\} \longrightarrow E_T$ *is a bijective mapping, assigning to each dimension* $d \in \{\widehat{C}_e + 1, \widehat{C}_e + 2, \cdots, C_m\}$ *an encrypted tag from the set* E_T.

The use of the SCS strategy dictates the codebook to be a collection of Grassmannian codes with different dimensions. The set $\mathcal{S}_c$ and the mappings θ and ϕ, as well as the set E_T, are provided by the KDC as we have previously noted.

3. Error based security

Let $\langle V \rangle \in C$ denote the chosen codeword for the current generation where $C \in \mathcal{S}_C$. Let d_C denote the minimum subspace distance of the code C. Before injecting the codeword into the network, the codeword $\langle V \rangle$ is transformed into $\langle V_e \rangle$ such that $d(\langle V \rangle, \langle V_e \rangle) = \widehat{C}_e$. Recalling that, $d_C \geq 2\widehat{C}_e + d_{min}$, the destination nodes will still be able to extract the valid codeword since network random errors have already been covered by d_{min}. On the other hand, those intended errors will be used to increase the security of the system by reducing the effectiveness of arranging the possible codewords based on their distance measure from the subspace spanned by the wiretapped vectors when performing the exhaustive search attack as explained later in this paper.

At the end of this step, the source RLNC–encodes the codeword and injects it into the network. Note that an encrypted tag is attached to each one of its vectors to specify the dimension of the used subspace for successful decoding.

3.4.2.3 At destination nodes

To reverse the operations that have been carried out at the source node, two steps are required by the destination nodes to extract the source message.

1. Data Decoding

Using one of the received packets, a destination node will extract the encrypted tag and use the lookup table to get the dimension of the transmitted codeword. Upon receiving a number of innovative packets equal to the codeword dimension, decoding is carried out to get the bitstring that corresponds to the received codeword. This step will be repeated for all the received codewords until all of them are successfully received and decoded.

2. Inverse Permutation

After a codeword is decoded, the resulted bitstring will undergo a reverse permutation to get the original bitstring. Once all the bitstrings are available, they are joined together

to form the source datastream D.

3.4.3 Security Analysis

An exhaustive search attack is considered where Eve wiretaps the network and tries to guess the source message. Depending on how Eve may attempt to solve this problem, two possible approaches may be adopted.

3.4.3.1 First Approach

In this approach, Eve assumes that at least some of the innovative vectors in the set of the wiretapped vectors are valid i.e belong to the subspace $\langle V \rangle$ where $\langle V \rangle$ is the valid codeword for the current transmission round. If this assumption turns out to be valid, Eve may help reduce its exhaustive search attack complexity at the expense of more decoding complexity. two steps may be required to carry out the exhaustive search attack, a decoding step and an exhaustive search step.

1. **The decoding step**: As seen earlier, at every transmission round, the source will inject a subspace $\langle V_e \rangle$, which is an erroneous version of $\langle V \rangle$. Eve will receive a subspace $\langle W \rangle$ where W is a set whose elements are the wiretapped vectors. In addition to the unknown $dim(\langle V \rangle) - |W|$ rank deficiency that is experienced by Eve. $\langle W \rangle$ may itself be affected by the erroneous vectors used to get $\langle V_e \rangle$ as well as possibly other network-induced errors. With all the aforementioned information taken into account, in this first approach, the wiretapper will assume that the received subspace $\langle W \rangle$ contains itself some valid subspace that will be used to reduce the number of possibilities concerning the transmitted codeword.

Regardless of the used subspace code, a decoder at a legitimate user node will always try to output the best codeword. For example, if a legitimate user receives a subspace $\langle T \rangle$ with the knowledge of the used subspace code, a distance decoder will output $\langle \widehat{V} \rangle$ as an estimation of the transmitted codeword $\langle V \rangle$ if the the following condition is satified,

$$\forall X \in C,\ d_C(\langle \widehat{V} \rangle, \langle T \rangle) \leq d_C(\langle X \rangle, \langle T \rangle) \tag{3.11}$$

where C is the subspace used in the transmission and $d_C(\cdot, \cdot)$ is the distance measure on C. Such decoders are not of much interest to Eve given the fact that they will consider the unknown $dim(\langle V \rangle) - dim(\langle W \rangle)$ as erasures and they can at best produce one codeword as a possible estimation for the valid codeword $\langle V \rangle$. Therefore, from Eve's perspective, the decoder's criteria for decoding have to change in such a way that the aforementioned

problems are addressed i.e. the unknown $dim(\langle V\rangle) - dim(\langle W\rangle)$ are not taken as erasures and all the possible codewords have to be output. Given the lack of information surrounding the subspace $\langle V\rangle$, a possible decoder for Eve may try to produce a set $\mathcal{V}$ for every wiretapped set W as follows,

$$\mathcal{V} = \{V \in C, \forall C \in \mathcal{S}_{\mathcal{C}} \mid d_C(\langle V\rangle \oplus \langle W\rangle, \langle V\rangle) \leq d_{C_{min}}/2\} \tag{3.12}$$

where $d_{C_{min}}$ is the minimum distance of the subspace code C.

The problem with this decoder is the complexity arising from having to check all the codewords in $\mathcal{S}_{sym}$. To check the distance between two subspaces $\langle V\rangle \oplus \langle W\rangle$ and $\langle V\rangle$, Eve may multiply a transposed version of the matrix representing the subspace $\langle W\rangle$ by that representing $\langle V^{\perp}\rangle$ where $\langle V^{\perp}\rangle$ is the dual subspace of $\langle V\rangle$ and output $d_C(\langle V\rangle \oplus \langle W\rangle, \langle V\rangle)$ as the number of nonzero columns in the resulted matrix, similar to syndrome decoding in classical coding theory. This latter approach will produce a decoding complexity of $O(n \cdot |W| \cdot dim(C))$ for every measure of the distance. Even for an optimized version of Eve's decoder, Eve will still have to check all the possible $|S_{sym}|$ codewords due to the dimension deficiency in the wiretapped sets. Eve may choose to arrange the possibilities according to the obtained value of d_C. Note that the use of the induced errors is meant to weaken this latter strategy by altering the value of d_C. At the end of this step, every set $\mathcal{V}$ will be transformed into a set $\mathcal{M}$ of message strings.

2. An illustration of the decoding step: As an illustration of the distance measure of the aforementioned decoder, consider a source using the SCS strategy with $C_e = 4$ and an Sc containing the code $C(12, 5, 6)$ with $q = 2$ constructed using [69], where the first two quantities are the size of the ambient finite field and the code dimension respectively, while the last number is the minimum subspace distance of the code. Suppose now that, at a given transmission round, a codeword $\langle V\rangle$ from this subspace code as represented by the matrix shown in Figure 3.9.a is transmitted (after inducing the errors). In this case, the dual space of the used codeword will be the one shown in Figure 3.9.b. Now suppose that Eve has an eavesdropping capacity $C_e = 4$ and the received matrix is that of Figure 3.9.c. After the multiplication of the matrix representing the dual space of the transmitted codeword by the received matrix, Eve will obtain a 7×4 matrix as shown in Figure 3.9.d. Given that the number of nonzero columns is inferior or equal to half of the minimum distance of the code,

Eve may choose to consider the codeword represented by the row vector space of this matrix as a possible transmitted codeword.

$$\begin{bmatrix} 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix}$$

(a) The matrix V representing the codeword $\langle V \rangle$

$$\begin{bmatrix} 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \end{bmatrix}$$

(b) The matrix $V^{\perp}$ representing $\langle V^{\perp} \rangle$, the dual subspace of $\langle V \rangle$

$$\begin{bmatrix} 0 & 1 & 1 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 1 & 0 & 1 & 0 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 1 & 1 & 0 & 0 & 0 & 0 & 1 & 1 & 0 & 0 & 0 & 0 \end{bmatrix}$$

(c) the matrix W representing the wiretapped subspace $\langle W \rangle$

$$\begin{bmatrix} 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 1 \\ 0 & 1 & 0 & 1 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \end{bmatrix}$$

(d) the resulted matrix of multiplying $V^{\perp}$ by the transpose of W

3. The exhaustive search step: Let $\mathcal{S}_W = \{W_1, W_2, \cdots, W_l\}$ with W_i being the set of the wiretapped vectors at the i^{th} transmission round. Once the decoding step is carried out for all the elements of $\mathcal{S}_W$, we will end up with a set $\mathcal{M}_W = \{\mathcal{M}_1, \mathcal{M}_2, \cdots, \mathcal{M}_l\}$ with $\mathcal{M}_i$ being the set of all possible data strings resulted from decoding $\langle W_i \rangle$. The concatenation of the elements of the tuples in $\prod_{i=1}^{i=l} \mathcal{M}_i$ will produce the different possible values of the permuted string D_p. For every such stream, Eve will keep trying all possible permutations until a meaningful data stream is found. This latter stream will be taken as D. The complexity of this step is $O(m! | \prod_{i=1}^{i=l} \mathcal{M}_i |)$ which is related to the different possible arrangements for D_p and the overall number of possible permutations.

3.4.3.2 Second Approach

In this approach, Eve will focus on the possibility that all the received vectors are erroneous and therefore she skips the decoding step and simply treats all the codewords in $|\mathcal{S}_{sym}|$ as possible guesses. This approach is less risky and eliminates the complexity that arises from the decoding process at the expense of more combinatorial complexity. The second step is the same as the first approach. The overall complexity of this approach will therefore

be $O(m! \cdot |\mathcal{S}_{sym}|^l)$.

3.4.3.3 Comparison of the two approaches

The first approach might be useful if the permutation key is too large that Eve is willing to afford more decoding complexity to reduce the cardinality of the elements of $\mathcal{M}_W$ and hence reducing the overall complexity resulting from trying the $m!$ possible permutations $|\prod_{i=1}^{i=l} \mathcal{M}_i|$ times. Note that using a decoder similar to the one discussed in the previous section will not be useful if the real wiretapping capacity satisfies $d_C \geq 2C_e$ for all the codes in $\mathcal{S}_C$. Eve may use this latter inequality to decide on the approach, which is the basis of the following corollary.

Corollary 1. *Maximum combinatorial complexity is obtained when $d_{Cmin}/2 \geq C_e$ for all codes $C \in \mathcal{S}_c$.*

Proof. The proof stems from the fact that when this inequality is satisfied. Eve will have to treat all the codewords as possible guesses. □

3.4.3.4 Comparison with other schemes

Table 3.6 provides a comparison of the exhaustive search complexity between our proposed scheme, the universal secure network coding scheme [61] (denoted as USNC in this paper), SPOC [4] and P–coding [3] for a wiretapper with $C_e < C_m$. The USNC complexity was estimated given the fact that Eve will have to guess on the missed packets. As for SPOC and P–coding, the guess on the missed packets has to be followed by a guess on the security parameters for each scheme which are the permutation key and the locked coefficients, respectively.

The USNC scheme provides both security and error correction to the transmitted data by using rank–metric codes as outer codes without altering the network code used for transmission. SPOC and P–coding, on the other side, are secure RLNC schemes that provide data confidentiality by securing the coefficient matrix. This latter is encrypted in SPOC and hidden using a permutation cipher columnwise in P–coding. However, neither of those two schemes incorporate an error–correction solution. Note that D_C and D'_C denote the overall decoding complexity for the first approach of our scheme and that of USNC, respectively. Each decoding step by Eve has a complexity in the order of $O(n \cdot |W| \cdot dim(C))$ where $C \subset \mathcal{S}$, for the first approach of our proposed scheme and $O(C_m \cdot n)$ for the universal secure network coding scheme.

The exhaustive search complexity for our scheme (both approaches) depends on the number of possible codeword combinations as well as the factorial complexity arising from the permutation of the substrings at the source. Its complexity partially depends on C_e since the search complexity depends on m as well. USNC's search complexity depends only on the number of possibilities for the missed packets, making this scheme's security highly dependent on C_e. When $C_e \geq C_m$, USNC will no longer be considered secure. In SPOC and P–coding, the search complexity depends on the number of missing packets as well as the security parameters of the two schemes.

Table 3.8: Computational complexity for the proposed algorithm, USNC, P–Coding and SPOC.

Scheme	**Search Complexity**
The proposed scheme (1st approach)	$O(m!\lvert\prod_{i=1}^{i=l}\mathcal{M}_i\rvert + D_C)$
The proposed scheme (2nd approach)	$O(m!\cdot\lvert\mathcal{S}_{sym}\rvert^l)$
USNC	$O(q^{((C_m-C_e)\cdot n)} + D'_C)$
P–Coding	$O(q^{((C_m-C_e)\cdot n)}\cdot(m+n)!)$
SPOC	$O(q^{((C_m-C_e)\cdot n)}\cdot q^{m^2})$

3.4.4 Scheme Evaluation

This scheme is an error-correcting scheme that is also intended to be used to secure data transmission via the use of two main operations:

1. A permutation step.

2. A dimension hiding step with error induction.

Both of the two operations are meant to increase the computational complexity of an exhaustive search attack of a given adversary. While the first step does not technically make any assumptions on Eve's wiretapping capacity, the second one targets a very specific set of wiretappers by guessing their wiretapping capacity C_e. This guess is used to specify the possible codeword dimensions to be used as well as their required subspace distance.

In this section, we evaluate our scheme in terms of its security as well as its performance. To achieve that, we have adopted the network shown in Figure 3.13 and a bitstream of 120 bits as the source D stream and we set our environment such that $d_{min} = 3$ i.e. single errors are more probable than multiple errors. We let $\widehat{C}_e = 3$. In this case, the codes to be used have to satisfy a minimum distance $d \geq 9$. Using these requirements we have opted for $m = 11$ with a set $\mathcal{S}_c = \{\mathcal{C}_5, \mathcal{C}_6, \mathcal{C}_7, \mathcal{C}_8\}$ where the characteristics of the used codes are stated

in Table 2. Note that we have excluded the 0 codeword from the θ mapping for each code from $\mathcal{S}_C$

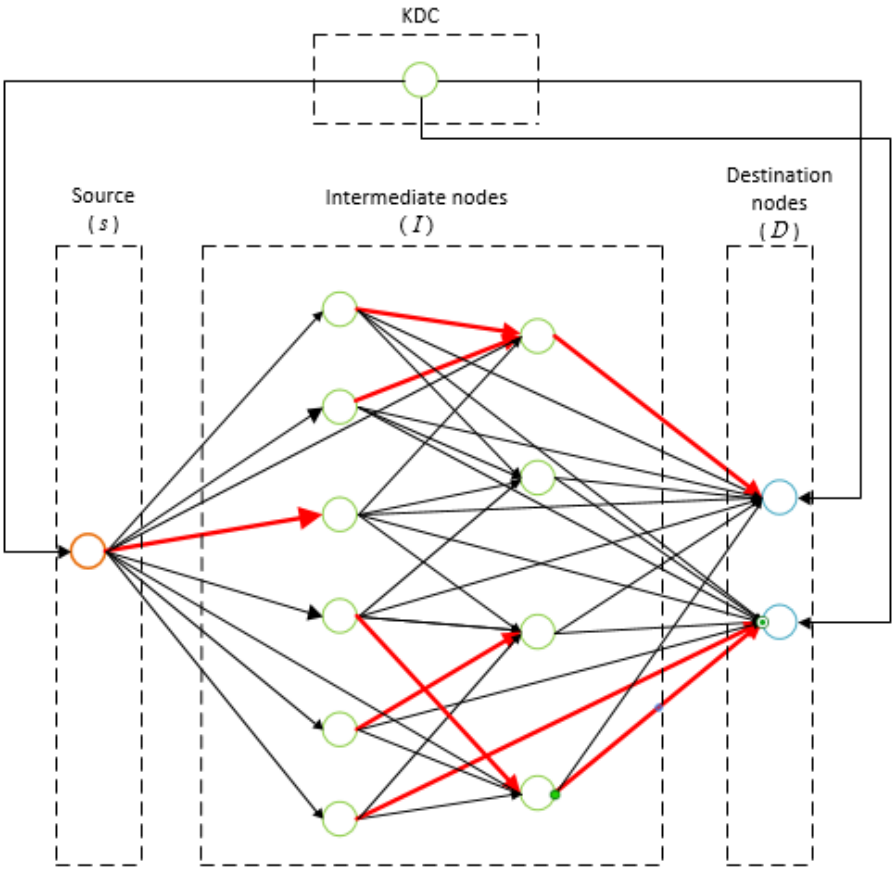

Figure 3.13: A network with one source, 10 intermediate nodes, two destination nodes and a Key Distribution Center, with a multicast capacity $C_m = 8$.

Table 3.9: The characteristics of the codes of $\mathcal{S}_c$.

Code	C_5	C_6	C_7	C_8
Ambient vector space	$\mathbb{F}_2^{16}$	$\mathbb{F}_2^{16}$	$\mathbb{F}_2^{16}$	$\mathbb{F}_2^{16}$
Dimension	5	6	7	8
Cardinality -1	2048	1024	512	256
Minimum subspace distance	10	12	14	16

The elements of the used $\mathcal{S}_C$ are subspaces from the 16-dimensional ambient vector space over $\mathbb{F}_2$. Those codes are taken from [70] where they have already been evaluated and classified. RLNC in this case will be equivalent to XOR encoding where output packets of a given node are created by XORing its incoming packets. To solve the problem of erasures that may be induced due to the underlying field size, we have adopted the network in Figure 3.13 where erasures resulting from encoding operations are excluded. The red edges on the

figure represent the edges that can be wiretapped by Eve for our experiment i.e. $0 \leq C_e \leq 8$, where 0 means that Eve has no access to network edges. We have chosen the worst case scenario where the wiretapped vectors are all linearly independent.

Figure 3.14 shows the effects of the real capacity on the search space for one transmission round with Eve using the first approach i.e. selecting only the possible codewords. As shown in Figure 3.14, the combinatorial complexity arising from the number of possible codewords depends on the validity of the guess $\widehat{C}_e$.

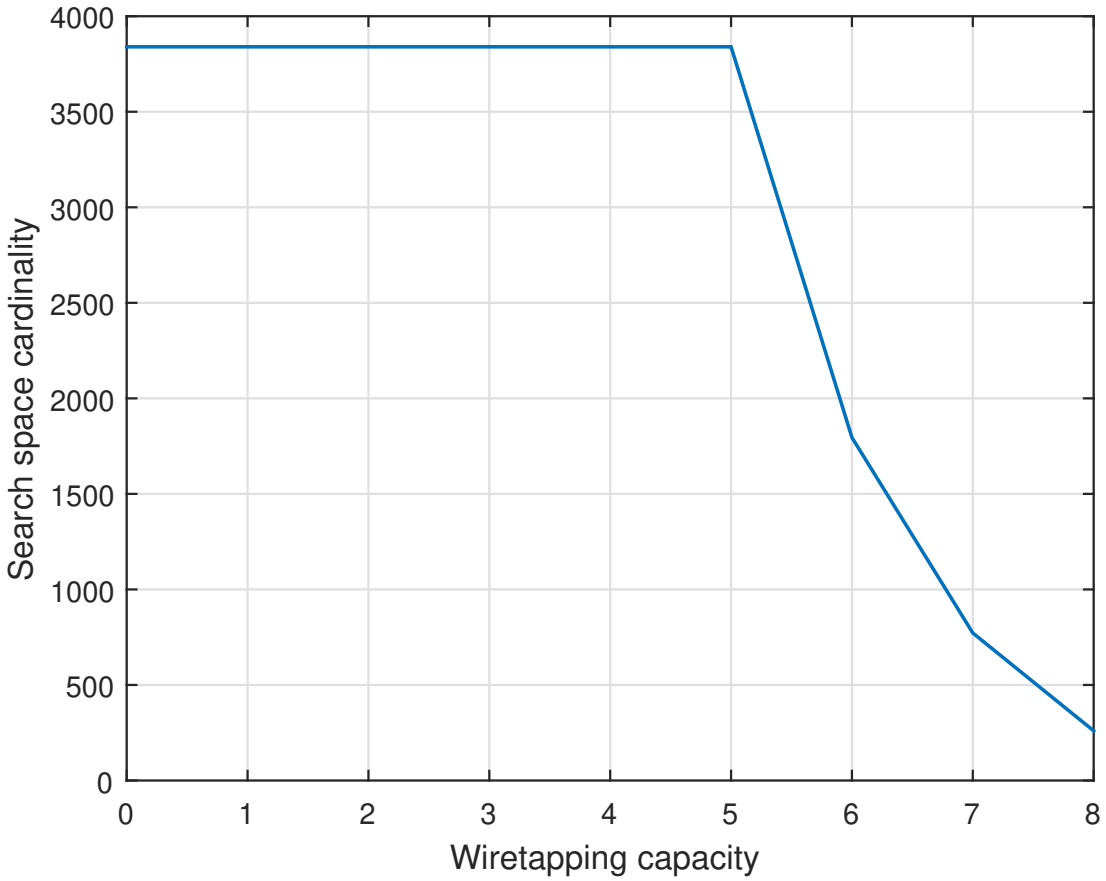

Figure 3.14: The search space cardinality for one transmission round Vs. Eve's wiretapping capacity.

To illustrate the effect of the errors induced by the source on Eve's attack, we send a codeword twice. The first time we send the codeword without any induced errors and the second time, we send it with errors as specified by the scheme. Figure 3.15 shows that a valid codeword will always maintain a zero distance with the wiretapped subspace, which is expected, compared to the one with the induced errors. This difference will be useful if Eve adopts the strategy of ranking the possible codewords according to how much the

subspace spanned by the wiretapped vectors is close to a given codeword as specified by the decoder. As we can see in Table 3.8, the greater C_e, the more the effects of the induced errors are reduced since the real codeword will tend to be closer to the subspace spanned by the wiretapped vectors compared to the other codewords. In this scenario, the effects of the dimension hiding step will practically be less effective and the system security will be guaranteed by the permutation step.

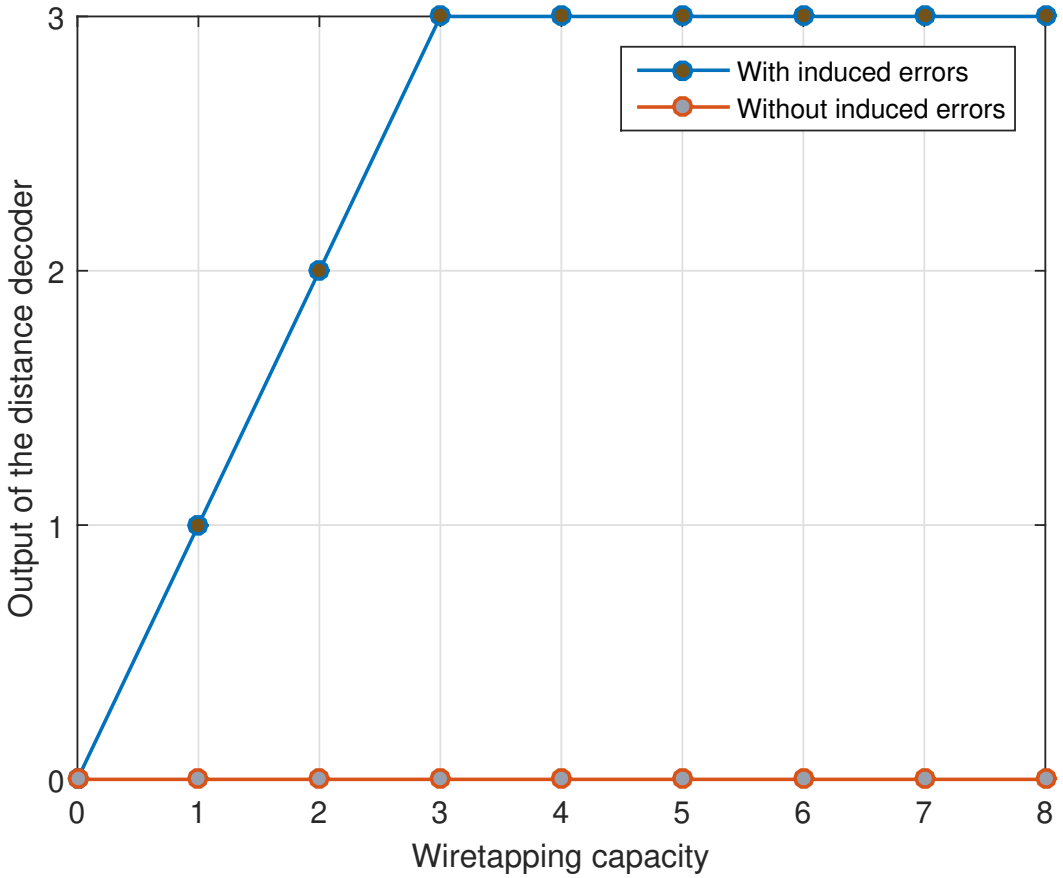

Figure 3.15: The distance between the valid codeword and the subspace $\langle W \rangle$ with and without the induced errors Vs. Eve's wiretapping capacity.

The guess probability of Eve is,

$$P_g = \frac{1}{m! |\prod_{i=1}^{i=l} \mathcal{M}_i|} \tag{3.13}$$

where $m!$ results from the number of all possible values of the m-bit string and $\prod_{i=1}^{i=l} \mathcal{M}_i$ results from the number of possible codewords that can have been transmitted by the source as indicated by Eve's distance decoder shown in Figure 3.16 for our proposed scheme, USNC, SPOC and P–coding. This quantity depicts the probability that Eve gets the source infor-

Table 3.10: The number of codewords Vs. their distance from the wiretapped subspace $\langle W \rangle$ as specified by the distance decoder as Eve's wiratapping capacity changes.

Ce \ d	0	1	2	3	4	5	6	7	8
1	4	3836	0	0	0	0	0	0	0
2	0	8	3832	0	0	0	0	0	0
3	0	0	12	3828	0	0	0	0	0
4	0	0	0	24	3816	0	0	0	0
5	0	0	0	12	4	3824	0	0	0
6	0	0	0	4	2	10	1779	0	0
7	0	0	0	2	2	2	7	759	0
8	0	0	0	1	1	1	1	5	251

mation by making a random guess based on the information obtained from the wiretapped vectors. In our scheme, this quantity is related to the number of combinations in terms of the possible codewords and the number of permutations of the m bitstring at the source. While this latter is not affected by $\widehat{C}_e$, the number of possible codewords depends on it as shown in Figure 3.14. Therefore, the search space of the wiretapper decreases as the wiretapping capacity exceeds its guessed value and as C_e reaches the multicast capacity C_m of the network, the wiretapper will find it easier to deduce all the valid codewords. The security of the system against those wiretappers will be maintained by the permutation step. The guess probability using USNC is related to the number of all possible missing packets. Along with this latter, SPOC requires also to guess the right encoding coefficients that are locked. While in P–coding, the main guess will be on the right column permutation. In this experiment, we notice that SPOC and P–coding have better guess probability compared to our scheme and USNC. Those latter two are close in terms of performance. However, USNC will provide no security as C_e reaches C_m.

Figure 3.17 focuses on the case when $C_e = C_m$ by providing a comparison between our scheme, SPOC and P–coding in terms of Eve's guess probability with $len = 24$ where len is the length of the RLNC packets for SPOC and P–coding given that in RLNC without subspace coding, the coefficient headers are sent along with the packets. As stated above, the security of the system at this stage depends mostly on the permutation step.

Table 3.9 provides a general comparison of our scheme with USNC, SPOC and P–coding in terms of confidentiality, error correction and encryption time. The four schemes provide data confidentiality for the transmitted data when $C_e < C_m$ with varying complexity as shown in Figure 3.16 and Table 3.6. Our scheme, P–coding and SPOC will maintain the

data confidentiality when $C_e = C_m$, This is not true for USNC since the security is only guaranteed as long as $C_e < C_m$. Note that the scenario when $C_e > C_m$ is excluded since the extra capacity will not provide additional information. As for error correction, only the proposed scheme and USNC provide such feature. SPOC and P–coding are only designed for security purposes and therefore additional enhancements should be made before being deployed in a lossy network. Concerning time comparison, 512 bytes of data were encrypted using the four schemes. Our scheme slightly exceeds the time required by the lightweight scheme P–coding followed by SPOC and USNC. The reason for the lightweight nature of our scheme stems from the fact that the only operations performed by the source are the permutation step and the codeword selection which are lightweight given that they are just memory swap and memory indexing operations, respectively. Note that the time experiment was repeated 100000 times and their expected value was taken. Moreover, to avoid the bias from the operating system scheduler, the ThreadMXbean interface in Java was used to measure the time individually consumed by the thread.

Table 3.11: Comparison between our scheme, USNC, SPOC and P–coding.

Scheme	**The proposed scheme**	**USNW**	**P–coding**	**SPOC**
Confidentiality for $C_e \leq C_m$	Yes	Yes	Yes	Yes
Confidentiality for $C_e = C_m$	Yes	No	Yes	Yes
Error correction	Yes	Yes	No	No
Encryption time in μs (512 bytes)	1.27	2.49	1.00	1.57

3.5 Conclusion

In this chapter, RLNC has been investigated from a security perspective. We have proposed Four schemes that aim for preserving confidentiality of RLNC-encoded data during transmission. The first three schemes were based on the encryption of the encoding matrix in addition to other steps. Those schemes were proposed for error-free environments to highlight the intrinsic security of RLNC that is mainly a byproduct of the idea of packet mixing. In the last scheme, we opted for data security in erroneous environments, where we have shown that error correction codes in RLNC can also be deployed for security against wiretap attacks. The scheme that has been proposed is a security scheme for constant dimension codes. It is based on a permutation step applied on the source data stream followed by the application of the SCS to create an ambiguity concerning the dimension of the subspace used for the encoding operation as a way to increase the search space of a wiretapping attack. Correctable errors

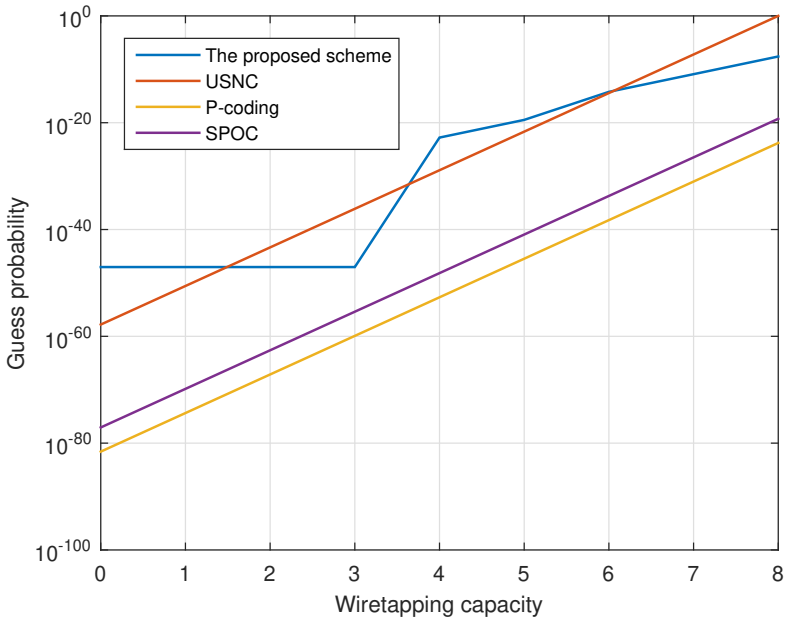

Figure 3.16: The guess probability Vs. Eve's wiretapping capacity.

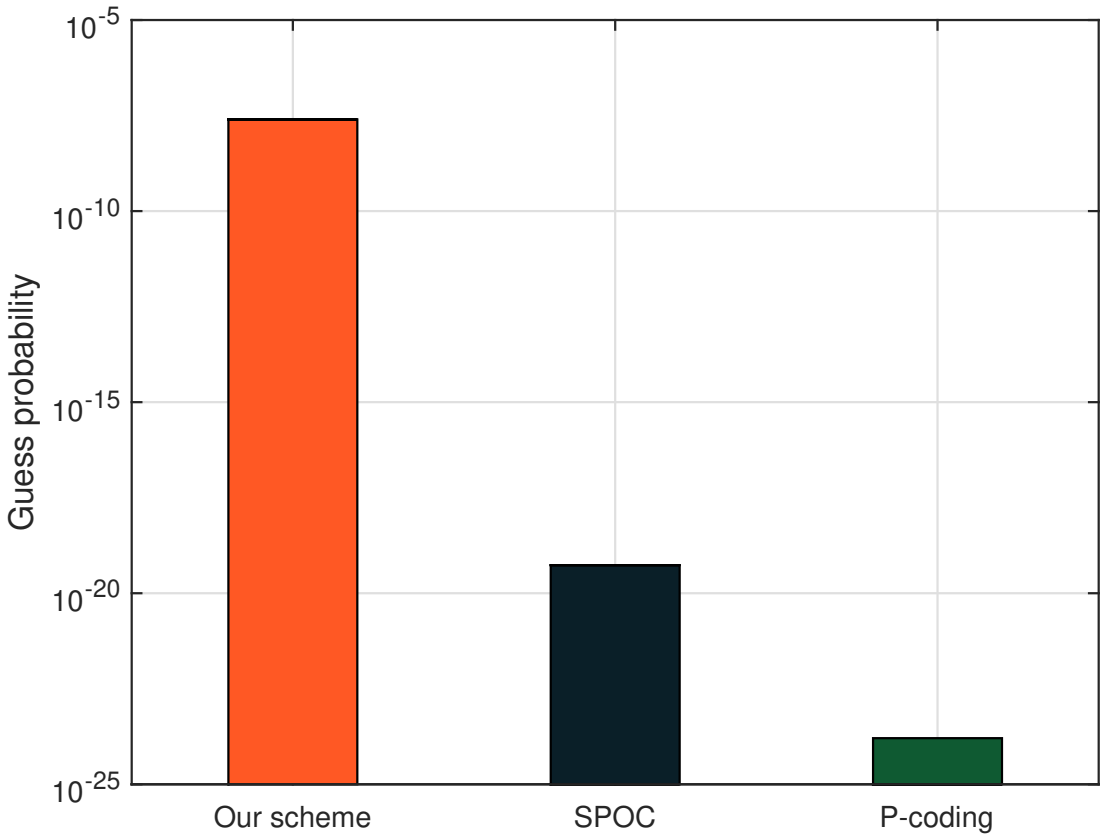

Figure 3.17: Guess probability for our scheme, SPOC and P–coding for a wiretapper with a wiretapping capacity of C_m.

are also induced into the codewords before transmission. While in general, we prefer error-free transmission, correctable errors can serve as an enhancement of data security against wiretap attacks as they help mislead the wiretapper by decreasing the efficiency of ranking the possible codewords according to their distance from a given valid codeword. Results show that our scheme performance is based on the correctness of the guess $\widehat{C}_e$ as well as the value of m. When C_e reaches C_m, the system security will be mainly maintained by the permutation step. For error free-environment SPOC and P-coding are more secure than our proposed scheme. However, With a suitable value of m, our scheme may be able to provide security for all values of C_e with a performance that can reach that provided by SPOC and P–coding. However, those latter schemes have been proposed only for lossless environments, making the proposed scheme more adequate for real-world lossy networks.

General Conclusions and Perspectives

This thesis reflects an interest in Error Control in Random Linear Network Coding. The problem of error propagation in RLNC overwhelms any end-to-end classical error correction code that is designed for the hamming metric due the packet mixing feature that characterizes NC schemes. To solve this problem, subspace codes were proposed by [41]. In those codes, codewords are vector spaces from an ambient vector space over the underlying finite field $\mathbb{F}_q$. The channel used for those codes is the operator channel that treats errors as a combination of erasures and insertions. Erasures are the result of a decrease in the dimension of a transmitted codeword, whereas an insertion is an increase in its dimension. This increase is a result of combining corrupt packets with valid ones at the encoding intermediate nodes. This channel is virtual in nature and simplifies the complexities of the different links in the network making it more appropriate for noncoherent scnenarios.

Given that data is essentially transmitted as vector spaces in RLNC, the use of subspace codes as outer codes on top of RLNC is a logical approach. In this thesis, we tried to investigate those codes from error control and security perspectives in which we have proposed a security transmission scheme to allow for secure data transmission of data encoded using constant dimension codes. The proposed scheme is a combination of a permutation step and the application of the SCS to create an ambiguity concerning the dimension of the subspace used for the encoding operation. This ambiguity is meant to increase the dimension of the search space of the wiretapper. Correctable errors are also induced to help mislead the wiretapper by decreasing the efficiency of ranking the possible codewords according to their distance from the valid codeword.

The scheme has been analysed theoretically and compared to other secure schemes for RLNC in the literature. Results have shown that with a suitable value of m, our scheme will be able to provide security for all values of C_e with a performance similar to other schemes in the literature such as SPOC and P–coding while deployed in real world situations where errors cannot be excluded.

In this thesis, we mainly focused on subspace codes from an error control perspective, especially when combined with a security scheme to offer both data integrity and transmission security. However, we have also obtained results that belong to the area of RLNC and SRLNC. In RLNC, a novel encoding scheme that reduces the number of multiplicative

operations in RLNC has been introduced to reduce the encoding complexity in RLNC. On the other hand, three security schemes for RLNC have been also proposed. Those schemes rely on securing the coefficient matrix along with other steps to provide confidentiality to the transmitted data.

In our future work, we aim to continue our work on error correction and security in RLNC. As for error correction, we are working on the design of a list-decoder for subspace codes that is based on the hamming distance and the subspace distance at the same time. A novel construction for subspace codes that links classical error correcting codes to subspace codes is also underway. As for security, we will be working on how to provide other forms of security to RLNC networks such as authentication.

Chapter 4

Appendix

4.1 Guess Probability

Let Enc be an encryption scheme that encrypts a plain text T_p into an encrypted text T_e using an encryption key E_k from a key space K_s. Let Eve be a wiretapper that knows everything about the encryption scheme except for the used keys. Eve obtains a copy of T_e and tries to obtain the E_k in order to get the transmitted plain text T_p. We define the guess probability P_g as being the probability of successfully guessing the right encryption key E_k from a single trial. Assuming uniform distribution on the key space K_s as observed by Eve, P_g will be given as

$$P_g = \frac{1}{|K_s|}$$

In the case of uniform distribution, this probability is uniquely determined by the cardinality of the key space. This metric is used usually to attest the security performance of a given encryption scheme [71].

As an illustration of the guess probability, consider the AES–256 encryption scheme. In this case $|K_s| = 2^{256}$. Hence,

$$P_g = \frac{1}{2^{256}} \approx 8.64 \times 10^{-78}$$

For encryption schemes that are composed of many steps such as 3DES and where all keys are uniform random variables, at least as observed by the wiratapper, the cardinality of the effective key space will be the multiplication of the different cardinalities for all the key

spaces.

In 2-Key 3DES

The key space cardinality is :$2^{56} \times 2^{56} = 2^{112}$

The guess probability

$$P_g = \frac{1}{2^{112}} \approx 1.96 \times 10^{-34}$$

In 3-Key 3DES

The key space cardinality is : $2^{56} \times 2^{56} \times 2^{56} = 2^{168}$

The guess probability

$$P_g = \frac{1}{2^{168}} \approx 2.67 \times 10^{-51}$$

A more interesting illustration of the guess probability for our purposes on this thesis is that related to SPOC. In SPOC, the coefficient matrix used at the source is encrypted and another encoding matrix is attached to the data matrix for storing the encoding operations performed across the network. A simple brute force attack on SPOC will be to try all possible keys that can be used to encrypt the coefficient matrix. If the encoding matrix is an $m \times m$ matrix over some finite field $\mathbb{F}_q$, the number of all possibilities will be q^{m^2}, which is all possible $m \times m$ matrices over $\mathbb{F}_q$. However, a better strategy for a search space attack will be to search for the plain version of the encrypted coefficient matrix instead of the key that has been applied to it. The rationale for this is that, any matrix can be used as a key. However, only full rank matrices are used as coefficient matrices. In this case, the attacker will neglect all matrices that have a rank inferior to m in order to reduce its key space. It happens that the number of full rank matrices over $\mathbb{F}_q$ is $\prod_{i=0}^{m-1}(q^m - q^i)$ and therefore the guess probability will be,

$$P_g = \frac{1}{\prod_{i=0}^{m-1}(q^m - q^i)}$$

For an 8×8 coefficient matrix over $\mathbb{F}_4$

$$\begin{aligned} P_g &= \frac{1}{\prod_{i=0}^{7}(4^8 - 4^i)} \\ &= \frac{1}{65535 \times 65532 \times 65520 \times 65472 \times 65280 \times 64512 \times 61440 \times 49152} \end{aligned}$$

$$\approx 4.27 \times 10^{-39}$$

As it is shown from the number of keys or from the expression of the guess probability. The value of this latter is a decreasing function of m and q. For high values of q and m, this probability will be extremely low which shows that such systems are secure against guessing attacks. For instance, when $m = 16$ and $q = 4$,

$$P_g = \frac{1}{\prod_{i=0}^{15} (4^{16} - 4^i)} \approx 1.08 \times 10^{-154}$$

4.2 Preliminaries on abstract algebra

4.2.1 Basics

In this section, we review the algebraic structures that are used in the theory of subspace codes as well as some related definitions. The first of those definitions is that of the group, which is fundamental to all other structures.

Definition 25. *A group G is a set X with an operation $\circ : X \times X \to X$ satisfying the following axioms:*

(i) Identity: There exists an element $e \in G$ such that for any $f \in G$ we have $e \circ f = f \circ e = f$.

(ii) Inverses: For any element $f \in G$ there exists $f^{-1} \in G$ such that $f \circ f^{-1} = e$.

(iii) Associativity: For any $f, g, h \in G$, we have $(f \circ g) \circ h = f \circ (g \circ h)$.

In subspace coding, or more specifically, in orbit codes, the theory of group actions is fundamental. Group actions may be seen as homomorphisms from a group G to the symmetric group of a set X. In other words, when a group acts on a set, the elements of that group will be applied to the elements of the set to yield new elements of the set. A formal definition of group actions may be summarized as,

Definition 26. *Let G be a group and X be a set. We say that G acts on X from the right or that there is a right action of G on X if there exists a mapping,*

$$\phi : X \times G \to X$$
$$(x, g) \mapsto x \cdot g$$

satisfying the following axioms:

(i) $e \cdot x = x$ *for every* $x \in X$*, where* e *denotes the identity element of the group* G*.*

(ii) $g \cdot (h \cdot x) = (g \cdot h) \cdot x$ *for every* $x \in X$*, every* g *in* G*, and every* h *in* G*.*

Note that the previous definition is for right group actions. the same definition can be formulated for left group actions.

Group actions will intuitively induce the following equivalence relation $\mathcal{R}$.

$$\forall x, y \in X \;\; x \mathcal{R} y \Leftrightarrow \exists g \in G, x = g \cdot y \tag{4.1}$$

The equivalence classes that result from $\mathcal{R}$ are called the orbits of G. A formal definition of them is provided as follows,

Definition 27. *Let* G *be a group and let* X *be set. The orbit of an element* $x \in X$ *is defined as,*

$$xG = \{x \cdot g | g \in G\} \tag{4.2}$$

We denote by X/G *the set of all orbits of* G *on* X*.*

$$X/G = \{xG | x \in X\} \tag{4.3}$$

While orbits may be seen as the set whose elements are the objects resulted from the action of G on a single element of X, the stabilizer of an element $x \in X$ are the elements of G that keep x unchanged under the action of G.

Definition 28. *Let* G *be a group and let* X *be set. The stabilizer of an element* $x \in X$ *is a subset* $St_G(x) \subseteq G$ *satisfying,*

$$St_G(x) = \{g \in G | x \cdot g = x\} \tag{4.4}$$

The theory of orbits and stabilizers has been proven to be useful in the construction of constant dimension codes [59], [60].

The most important structures in RLNC as well as in subspace coding are undoubtedly fields and vector spaces. Both of these structures are defined below.

Definition 29. *A field F is a closed set under the two binary operations $(+)$ and $(\cdot)$, such that,*

(i) F *is an abelian group under the additive binary operation* $(+)$.

(ii) $F\backslash\{0\}$ *is an abelian group under the multiplicative binary operation* $(\cdot)$.

(iii) $\forall a, b, c \in F,\ a \cdot (b + c) = a \cdot b + a \cdot c$.

Definition 30. *Let V be a set. We say that V is a vector space if V is an additive abelian group with a field action. A vector space V may also be defined in terms of the following axioms.*

(i) V *is an abelian group under the additive binary operation.*

(ii) $\forall a, b \in F$ *and* $v \in V$, $(a + b)v = av + bv$.

(iii) $\forall a \in F$ *and* $u, v \in V$, $a(u + v) = au + av$.

(iv) $\forall a, b \in F$ *and* $v \in V$, $a(bv) = a(bv)$.

(v) $\forall v \in V$, $1_F \cdot v = v$, *where* 1_F *is the multiplicative identity of the multiplicative group* $F\backslash\{0\}$.

We call any set of linearly independent vectors of a vector space V a basis of V if its cardinality is maximum, i.e. there is no other set of linear independent vectors in V whose cardinality is greater than the cardinality of the basis. The number of vectors in a basis is referred to as the dimension of V. A subset of a vector space V that satisfies all the axioms of a vector space is called a subspace of V. In subspace coding, two important sets that are related to the set of subspaces of some vector space V are : the projective space P and the Grassmannian G, defined below,

Definition 31. *Let V be a vector space of dimension $n \geq 2$ over a field F. Let $0 \leq k < n$ be an integer. Then, the Grassmannian $G(k, n)$ over F is defined as the set of all k-dimensional subspaces of V.*

The number of distinct k-dimensional vector spaces in an n-dimensional vector space V over a finite field F with q elements is given by $\begin{bmatrix} n \\ k \end{bmatrix}_q$, where $\begin{bmatrix} n \\ k \end{bmatrix}_q$ is called the Gaussian

coefficient or the q-analog of the binomial coefficient and is defined as,

$$G_k = \begin{bmatrix} n \\ k \end{bmatrix}_q = \frac{(q^n-1)(q^{n-1}-1)\cdots(q^{n-k+1}-1)}{(q^k-1)(q^{k-1}-1)\cdots(q-1)}$$
$$= \prod_{n=0}^{k-1} \frac{(q^{n-i}-1)}{(q^{k-i}-1)}. \tag{4.5}$$

Definition 32. *The set of all subspaces of an n-dimensional vector space V over a finite field F with q elements is called the projective space $\mathcal{P}(n)$ or the projective geometry of dimension $n-1$ over F.*

The reason of the label "projective geometry" is due to the geometric properties of this structure. Note that in this thesis, we only think of the projective geometry as a set of subspaces. To relate the definition of the projective space to that of the Grassmannian, consider the following equation.

$$\mathcal{P}(n) = \cup_{0 \leq k \leq n} \ G(k,n). \tag{4.6}$$

4.3 Theorems and proofs

4.3.1 The subspace distance

Theorem 5. *The subspace distance $d_s(\cdot,\cdot)$ defined as,*

$$\forall u, v \in \mathcal{P}(n), d_s(u,v) = \dim(u \cup v) - \dim(u \cap v). \tag{4.7}$$

is a metric on P(n).

Proof. To prove that d_S is a metric on $\mathcal{P}(n)$, we need to show that,

1. $d_S(U,V) \geq 0$ with equality if and only if $U = V$, $\forall U, V \in \mathcal{P}(n)$.

2. $d_S(U,V) = d_S(V,U)$, $\forall\, U, V \in \mathcal{P}(n)$.

3. $d_S(U,V) \leq d_S(U,W) + d_S(W,V)$, $\forall\, U, V, W \in \mathcal{P}(n)$.

The two first conditions are clearly true given the properties of the $dim(\cdot)$ function. For the last one, we need to prove that $\Delta = d_S(U,V) - d_S(U,W) - d_S(W,V) \leq 0$.

Note that,

$$\begin{aligned} d(U,V) &= dim(U+V) - dim(U \cap V) && (4.8)\\ &= dim(U+V) - (dim(U) + dim(V) - dim(U+V)) \\ &= 2dim(U+V) - dim(U) - dim(V) && (4.9) \end{aligned}$$

Based on this, one can get the following,

$$\begin{aligned} \frac{1}{2}\Delta &= dim(U \cap W) + dim(V \cap W) - dim(W) - dim(U \cap V) \\ &= dim(U \cap W + V \cap W) - dim(W) + dim(U \cap V \cap W) \\ &\quad - dim(U \cap V) && (4.10) \end{aligned}$$

One can notice that $dim(U \cap W + V \cap W) - dim(W) \leq 0$ and $dim(U \cap V \cap W) - dim(U \cap V) \leq 0$. Hence, $\Delta \leq 0$ and $d_S(\cdot,\cdot)$ is indeed a metric on $\mathcal{P}(n)$.

□

4.3.2 The injection distance

Theorem 6. *The injection distance $d_I(\cdot,\cdot)$ defined as,*

$$\forall u, v \in \mathcal{P}(n), d_I(u,v) = \max\{\dim(u), \dim(v)\} - \dim(u \cap v). \quad (4.11)$$

is a metric on P(n).

Proof. Similar to what we have done in the previous proof, we only need to prove the triangle inequality. Since we have shown that the subspace distance is a metric on $\mathcal{P}(n)$, proving that the injection distance $d_I(\cdot,\cdot)$ is also a metric will be easier.

$$\begin{aligned} d_I(U,V) + d_I(V,W) = \frac{1}{2}(d_S(U,V) + d_S(V,W)) + max\{dim(U), dim(V)\} + max\{dim(V), dim(W)\} \\ - \frac{1}{2}(dim(U) + 2dim(V) + dim(W)) \end{aligned}$$

$$
\begin{aligned}
&\geq \frac{1}{2}d_S(U,W) + max\{dim(U), dim(V)\} + max\{dim(V), dim(W)\} \\
&- \frac{1}{2}(dim(U) + 2dim(V) + dim(W)) \\
&\geq \frac{1}{2}d_S(U,W) + max\{dim(U), dim(W)\} - \frac{1}{2}(dim(U) + dim(W)) \\
&= d_I(U,W) \qquad (4.12)
\end{aligned}
$$

□

Bibliography

[1] R. Ahlswede, Ning Cai, S.-Y.R. Li, and R.W. Yeung. Network information flow. *IEEE Transactions on Information Theory*, 46(4):1204–1216, July 2000.

[2] T. Ho, M. Medard, R. Koetter, D. R. Karger, M. Effros, J. Shi, and B. Leong. A random linear network coding approach to multicast. *IEEE Transactions on Information Theory*, 52(10):4413–4430, 2006.

[3] P. Zhang, C. Lin, Y. Jiang, Y. Fan, and X. Shen. A lightweight encryption scheme for network-coded mobile ad hoc networks. *IEEE Transactions on Parallel and Distributed Systems*, 25(9):2211–2221, 2014.

[4] J. P. Vilela, L. Lima, and J. Barros. Lightweight security for network coding. In *IEEE International Conference on Communications*, pages 1750–1754, 2008.

[5] Mohamed Amine Brahimi and Fatiha Merazka. A secure algorithm for random linear network coding. In *2020 IEEE 5th International Symposium on Smart and Wireless Systems within the Conferences on Intelligent Data Acquisition and Advanced Computing Systems (IDAACS-SWS)*, pages 1–4, 2020.

[6] Y. Fan, Y. Jiang, H. Zhu, and X. Shen. An efficient privacy-preserving scheme against traffic analysis attacks in network coding. In *IEEE INFOCOM*, pages 2213–2221, 2009.

[7] Mohamed Amine Brahimi, Fatiha Merazka, and Gunes Karabulut Kurt. Secure network coding for data encoded using subspace codes. *Physical Communication*, 48:101408, 2021.

[8] Hu Fei, Zhu Guangxi, and Zhu Yaoting. Enhanced arq-based packet loss recovery for real-time communication. In *2001 International Conferences on Info-Tech and Info-Net. Proceedings (Cat. No.01EX479)*, volume 2, pages 317–322 vol.2, 2001.

[9] Cuiping Jing, Xingjun Zhang, Yifei Sun, Huali Cui, and Xiaoshe Dong. A packet loss protection scheme joint deterministic network coding and random linear network coding for h.264/avc. In *2011 Fifth FTRA International Conference on Multimedia and Ubiquitous Engineering*, pages 149–154, 2011.

[10] Imen Jouili, Kawther Hassine, and Mounir Frikha. A network coding based solution to minimize packet loss during handover in lte-a systems: Highway scenario. In *2016 International Wireless Communications and Mobile Computing Conference (IWCMC)*, pages 458–462, 2016.

[11] S. . R. Li, R. W. Yeung, and Ning Cai. Linear network coding. *IEEE Transactions on Information Theory*, 49(2):371–381, 2003.

[12] Raymond W. Yeung, Shuo-Yen Robert Li, Ning Cai, and Zhen Zhang. *Network coding theory*, volume 2. Now Publ., 2005.

[13] S. Jaggi, P. Sanders, P.A. Chou, M. Effros, S. Egner, K. Jain, and L.M.G.M. Tolhuizen. Polynomial time algorithms for multicast network code construction. *IEEE Transactions on Information Theory*, 51(6):1973–1982, 2005.

[14] V. Geetha, Sridhar Aithal, and K. Chandra Sekaran. Effect of mobility over performance of the ad hoc networks. In *2006 International Symposium on Ad Hoc and Ubiquitous Computing*, pages 138–141, 2006.

[15] T. Ho, R. Koetter, M. Medard, D.R. Karger, and M. Effros. The benefits of coding over routing in a randomized setting. In *IEEE International Symposium on Information Theory, 2003. Proceedings.*, pages 442–, 2003.

[16] Carla-Fabiana Chiasserini, Emanuele Viterbo, and Claudio Casetti. Decoding probability in random linear network coding with packet losses. *IEEE Communications Letters*, 17(11):1–4, 2013.

[17] Ali Farzamnia, Ling Hui Zhen, Liau Chung Fan, and Md. Nazrul Islam. Investigation on decoding failure probability in erasure network coded channels. In *2017 IEEE 8th Control and System Graduate Research Colloquium (ICSGRC)*, pages 241–245, 2017.

[18] Chamitha de Alwis, H. Kodikara Arachchi, Anil Fernando, and Ahmet Kondoz. Towards minimising the coefficient vector overhead in random linear network coding. In *2013*

IEEE International Conference on Acoustics, Speech and Signal Processing, pages 5127–5131, 2013.

[19] Ye Li, Wai-Yip Chan, and Steven D. Blostein. Network coding with unequal size overlapping generations. In *2012 International Symposium on Network Coding (NetCod)*, pages 161–166, 2012.

[20] Danilo Gligoroski, Katina Kralevska, and Harald Øverby. Minimal header overhead for random linear network coding. In *2015 IEEE International Conference on Communication Workshop (ICCW)*, pages 680–685, 2015.

[21] Mohamed Amine Brahimi and Fatiha Merazka. On reducing the encoding complexity of random linear network coding. In *2020 International Conference on Electrical Engineering (ICEE)*, pages 1–5, 2020.

[22] Persi Diaconis. *Group representations in probability and statistics*. Institute of Mathematical Statistics, Hayward, CA, 1988.

[23] M. Langberg, A. Sprintson, and J. Bruck. The encoding complexity of network coding. *IEEE Transactions on Information Theory*, 52(6):2386–2397, 2006.

[24] C. Fragouli and E. Soljanin. Information flow decomposition for network coding. *IEEE Transactions on Information Theory*, 52(3):829–848, 2006.

[25] Daniel E. Lucani, Morten Videbæk Pedersen, Diego Ruano, Chres W. Sørensen, Frank H. P. Fitzek, Janus Heide, Olav Geil, Vu Nguyen, and Martin Reisslein. Fulcrum: Flexible network coding for heterogeneous devices. *IEEE Access*, 6:77890–77910, 2018.

[26] K. Bhattad, N. Ratnakar, R. Koetter, and K.R. Narayanan. Minimal network coding for multicast. In *Proceedings. International Symposium on Information Theory, 2005. ISIT 2005.*, pages 1730–1734, 2005.

[27] Wangshu Zhang, Jiarui Xie, and Xinjian Zhuo. An evolutionary approach to genetic algorithm on minimizing network coding resources. In *2012 3rd IEEE International Conference on Network Infrastructure and Digital Content*, pages 275–279, 2012.

[28] Huanlai Xing, Fuhong Song, Zhaoyuan Wang, Tianrui Li, and Yan Yang. On minimizing network coding resource: A modified particle swarm optimization approach. In *2016*

12th International Conference on Mobile Ad-Hoc and Sensor Networks (MSN), pages 330–334, 2016.

[29] Hanqi Tang, Qifu Tyler Sun, Zongpeng Li, Xiaolong Yang, and Keping Long. Circular-shift linear network coding. *IEEE Transactions on Information Theory*, 65(1):65–80, 2019.

[30] Zhiyuan Yan and Hongmei Xie. Enhanced algebraic error control for random linear network coding. In *MILCOM 2012 - 2012 IEEE Military Communications Conference*, pages 1–6, 2012.

[31] Ning Chen, Zhiyuan Yan, Maximilien Gadouleau, Ying Wang, and Bruce W. Suter. Rank metric decoder architectures for random linear network coding with error control. *IEEE Transactions on Very Large Scale Integration (VLSI) Systems*, 20(2):296–309, 2012.

[32] Huseyin Balli, Xijin Yan, and Zhen Zhang. On randomized linear network codes and their error correction capabilities. *IEEE Transactions on Information Theory*, 55(7):3148–3160, 2009.

[33] Ning Cai and R.W. Yeung. Network coding and error correction. In *Proceedings of the IEEE Information Theory Workshop*, pages 119–122, 2002.

[34] Zhen Zhang. Linear network error correction codes in packet networks. *IEEE Transactions on Information Theory*, 54(1):209–218, 2008.

[35] Shenghao Yang, Raymond W. Yeung, and Chi Kin Ngai. Refined coding bounds and code constructions for coherent network error correction. *IEEE Transactions on Information Theory*, 57(3):1409–1424, 2011.

[36] Xuan Guang, Fang-Wei Fu, and Zhen Zhang. Variable-rate linear network error correction mds codes. *IEEE Transactions on Information Theory*, 62(6):3147–3164, 2016.

[37] Shenghao Yang, Chi Kin Ngai, and Raymond W. Yeung. Construction of linear network codes that achieve a refined singleton bound. In *2007 IEEE International Symposium on Information Theory*, pages 1576–1580, 2007.

[38] Huaxiong Wang, Chaoping Xing, and R. Safavi-Naini. Linear authentication codes: bounds and constructions. *IEEE Transactions on Information Theory*, 49(4):866–872, 2003.

[39] Netanel Raviv and Tuvi Etzion. Distributed storage systems based on intersecting subspace codes. In *2015 IEEE International Symposium on Information Theory (ISIT)*, pages 1462–1466, 2015.

[40] Natalia Silberstein, Ankit Singh Rawat, and Sriram Vishwanath. Error resilience in distributed storage via rank-metric codes. In *2012 50th Annual Allerton Conference on Communication, Control, and Computing (Allerton)*, pages 1150–1157, 2012.

[41] Ralf Koetter and Frank R. Kschischang. Coding for errors and erasures in random network coding. *IEEE Transactions on Information Theory*, 54(8):3579–3591, 2008.

[42] Ralf Koetter and Frank R. Kschischang. Coding for errors and erasures in random network coding. In *2007 IEEE International Symposium on Information Theory*, pages 791–795, 2007.

[43] Danilo Silva and Frank R. Kschischang. On metrics for error correction in network coding. *IEEE Transactions on Information Theory*, 55(12):5479–5490, 2009.

[44] Huaxiong Wang, Chaoping Xing, and R. Safavi-Naini. Linear authentication codes: bounds and constructions. *IEEE Transactions on Information Theory*, 49(4):866–872, 2003.

[45] Tuvi Etzion and Alexander Vardy. Error-correcting codes in projective space. In *2008 IEEE International Symposium on Information Theory*, pages 871–875, 2008.

[46] Tuvi Etzion and Alexander Vardy. Error-correcting codes in projective space. *IEEE Transactions on Information Theory*, 57(2):1165–1173, 2011.

[47] Philippe Delsarte. An algebraic approach to the association schemes of coding theory. *Philips Res. Rep. Suppl.*, 10:vi+–97, 1973.

[48] Ernst Gabidulin. Theory of codes with maximum rank distance (translation). *Problems of Information Transmission*, 21:1–12, 01 1985.

[49] Ph Delsarte. Bilinear forms over a finite field, with applications to coding theory. *Journal of Combinatorial Theory, Series A*, 25(3):226–241, 1978.

[50] R.M. Roth. Maximum-rank array codes and their application to crisscross error correction. *IEEE Transactions on Information Theory*, 37(2):328–336, 1991.

[51] Antonia Wachter-Zeh. List decoding of crisscross error patterns. In *2014 IEEE International Symposium on Information Theory*, pages 1236–1240, 2014.

[52] R.M. Roth. Probabilistic crisscross error correction. *IEEE Transactions on Information Theory*, 43(5):1425–1438, 1997.

[53] Du Hoan Nguyen, Huu Loc Pham, and Linh Le Thi Trang. Security of the cryptosystem gpt based on rank codes and term-rank codes. In *2021 International Conference Engineering and Telecommunication (En T)*, pages 1–5, 2021.

[54] E.M. Gabidulin, A.V. Ourivski, B. Honary, and B. Ammar. Reducible rank codes and their applications to cryptography. *IEEE Transactions on Information Theory*, 49(12):3289–3293, 2003.

[55] Haitham Rashwan, Ernst M. Gabidulin, and Bahram Honary. A smart approach for gpt cryptosystem based on rank codes. In *2010 IEEE International Symposium on Information Theory*, pages 2463–2467, 2010.

[56] Sven Puchinger, Sebastian Stern, Martin Bossert, and Robert F.H. Fischer. Space-time codes based on rank-metric codes and their decoding. In *2016 International Symposium on Wireless Communication Systems (ISWCS)*, pages 125–130, 2016.

[57] P. Lusina, E. Gabidulin, and M. Bossert. Maximum rank distance codes as space-time codes. *IEEE Transactions on Information Theory*, 49(10):2757–2760, 2003.

[58] Danilo Silva, Frank R. Kschischang, and Ralf Koetter. A rank-metric approach to error control in random network coding. *IEEE Transactions on Information Theory*, 54(9):3951–3967, 2008.

[59] Felice Manganiello, Anna-Lena Trautmann, and Joachim Rosenthal. On conjugacy classes of subgroups of the general linear group and cyclic orbit codes. In *2011 IEEE International Symposium on Information Theory Proceedings*, pages 1916–1920, 2011.

[60] Anna-Lena Trautmann, Felice Manganiello, and Joachim Rosenthal. Orbit codes — a new concept in the area of network coding. In *2010 IEEE Information Theory Workshop*, pages 1–4, 2010.

[61] Danilo Silva and Frank R. Kschischang. Universal secure network coding via rank-metric codes. *IEEE Transactions on Information Theory*, 57(2):1124–1135, 2011.

[62] Jun Kurihara, Ryutaroh Matsumoto, and Tomohiko Uyematsu. Relative generalized rank weight of linear codes and its applications to network coding. *IEEE Transactions on Information Theory*, 61(7):3912–3936, 2015.

[63] Umberto Martínez-Peñas and Frank R. Kschischang. Reliable and secure multishot network coding using linearized reed-solomon codes. In *2018 56th Annual Allerton Conference on Communication, Control, and Computing (Allerton)*, pages 702–709, 2018.

[64] Umberto Martínez-Peñas and Ryutaroh Matsumoto. Relative generalized matrix weights of matrix codes for universal security on wire-tap networks. *IEEE Transactions on Information Theory*, 64(4):2529–2549, 2018.

[65] Mohamed Amine Brahimi and Fatiha Merazka. Data confidentiality-preserving schemes for random linear network coding-capable networks. *Journal of Information Security and Applications*, 66:103136, 2022.

[66] N. Cai and R. W. Yeung. Secure network coding on a wiretap network. *IEEE Transactions on Information Theory*, 57(1):424–435, 2011.

[67] Khaled A.S. Abdel-Ghaffar. Counting matrices over finite fields having a given number of rows of unit weight. *Linear Algebra and its Applications*, 436(7):2665 – 2669, 2012.

[68] Yohan Suryanto, Suryadi, and Kalamullah Ramli. Chaos properties of the chaotic permutation generated by multi circular shrinking and expanding movement. In *2015 International Conference on Quality in Research (QiR)*, pages 65–68, 2015.

[69] Daniel Heinlein. Generalized linkage construction for constant-dimension codes, 2019.

[70] Daniel Heinlein, Michael Kiermaier, Sascha Kurz, and Alfred Wassermann. Tables of subspace codes, 2019.

[71] Yantao Liu and Yasser Morgan. Security against passive attacks on network coding system – a survey. *Computer Networks*, 138:57–76, 2018.