Fault-Tolerant and Secure Data Transmission Using Random Linear Network Coding

Pouya Ostovari and Jie Wu
Department of Computer & Information Sciences
Temple University
Philadelphia, PA 19122
Email: {ostovari, jiewu}@temple.edu

***Abstract*—Network coding is a technique used to improve both wired and wireless networks' throughput and provide reliable transmission. In network coding, original data packets can be encoded to an infinite number of coded packets. A subset of these coded packets is sufficient to decode the coded packets and retrieve the original data. In addition to providing reliable data transmission, network coding can be used as a lightweight security mechanism to protect data against eavesdroppers. An eavesdropper is not able to decode coded packets and retrieve original data unless it has access to a sufficient number of coded packets. In a data transmission application, transmitting more redundant packets increases the chance of delivering a sufficient number of coded packets to the destination. As a result, this increases the reliability of the data transmission. However, more redundant transmissions make the system more vulnerable to eavesdropper attacks because there is a higher chance that an eavesdropper will receive enough coded packets. In this work, we study network coding to provide reliable and secure data transmission schemes by performing a trade-off between the security and reliability of the data transmission. We formulate the problem as a mixed integer and linear programming problem, and we propose a linear programming approximation to solve it. We study the performance of our proposed methods through extensive simulations.**

***Index Terms*—Network coding, security, fault tolerance, unicast, eavesdropping, optimization, random linear network coding.**

I. Introduction

Providing reliable data transmission that is resilient to path failures is important. Transmitting redundant data over different paths is an effective approach to achieve a fault-tolerant data transmission. Transmitting redundant data over each of the paths provides a higher level of protection against path failures. However, more redundant transmission also increases the cost of data transmission. Finding the optimal redundancy level over each of the paths to achieve a certain level of fault-tolerance has been widely studied by the research community. Also, different techniques, such as erasure codes and fountain codes have been used to generate redundant data. Random linear network coding is one the techniques that is widely used to produce redundancy.

In random linear network coding technique, original packets are linearly coded and the coded coefficients for mixing the packets are selected randomly over a finite field. In this type of coding, each of the coded packets is in the form of $\sum_{j=1}^{m} \beta_j \times P_j$. In this equation, β_j and P_j are, respectively, the random coefficients and the packets that are coded. Assuming that m packets are coded using random linear network coding, any m linearly independent coded packets suffice for decoding the coded packets and retrieving the originals. In this method, we can produce a potentially infinite number of coded packets. Decoding of the coded packets is performed using methods that solve a system of linear equations, such as Gaussian elimination method.

Another challenge in transferring data in a network is the security of the data transmission. Assume that we want to securely transfer a file from a source node to a destination node, in an untrustworthy network; an eavesdropper may overhear some of the transmitted packets in this network. The straightforward way to prevent non-authorized users or eavesdroppers from accessing to the original data is to use cryptographic methods. The source node can encrypt the original data before transmitting it. The encrypted data can be partitioned into multiple packets and transmitted over different paths. However, the complexity of cryptographic methods is itself a challenge.

An alternative approach to achieve data security is to use network coding. In order to provide a low-complexity security mechanism, [1], [2] propose a method to ensure the security of a distributed data storage that relies on network coding technique. Assuming that m packets are coded using random line network coding, m coded packets are required to decode the coded packets and retrieve the originals. The main idea in [1], [2] is to prevent eavesdroppers or non-authorized users from accessing number of coded packets required to decode the coded packets. As a result, eavesdroppers are not able to use Gaussian elimination to decode the coded packets and construct the original data. Using random linear network coding as a security mechanism, confidentiality can be achieved without adding extra complexity and cost. In this work, the authors assume that only authorized users know the location of the data storage for a specific file. Consequently, in the proposed security schemes, location of the data storages have the same role as a secret keys in a cryptographic method. In the mentioned work, applying network coding makes distributed data storage robust against eavesdropper attacks and storage failures at the same time.

In [3], we use network coding to design a fault-tolerant and secure distributed data storage using network coding. In a similar way, network coding can be used to make data transmission robust against eavesdropper attacks and path failures. Transmitting more redundant data over different paths can enhance the fault-tolerance of the data transmission against path failures and increase the chance that the destination will receive enough coded packets to retrieve the original data. However, transmitting more redundant coded packets increases the data transmission's vulnerability to eavesdropping attacks. This is because transmitting more coded packets on each path increases the chance that an eavesdropper can access a sufficient number of coded packets. Furthermore, a secure data transmission that uses network coding is not necessarily less fault-tolerant. Coded packets are transmitted over different paths with different reliabilities and security levels. As a result, if we consider the security and reliability of the paths when distributing data, then we can use network coding to provide security and reliability concurrently.

In Figure 1, three parallel and edge disjoint paths are shown between source S and destination D. In this example, the path failure probability of each path is 0.2. The source node transmits a file consisting of 4 packets via these 3 paths. Assume that an eavesdropper accesses transmitted packets over paths 1, 2, and 3 with probabilities 0.1, 0.1, and 0.2. We consider two cases. In case one, we transmit 2 random linear network coded packets both paths 1 and 2. As a result, the only scenario that the destination node is able to decode the coded packets and retrieve the original packets is when both of paths 1 and 2 successfully deliver the transmitted packets. In this case, the probability of successful data retrieval is $0.8^2 = 0.64$. An eavesdropper cannot decode the coded packets without accessing all 4 transmitted packets. Consequently, the probability of successful eavesdropping is $0.1^2 = 0.01$.

In the second case, the source node transmits 2 coded packets over each of the 3 paths. Because the number of original packets is 4, accessing any 4 transmitted coded packets is sufficient to decode the coded packets. In this scenario, the probability of successful decoding by the destination node is $0.8^3 + 3 \times 0.8^2 \times 0.2 = 0.896$. This is when at least two paths deliver the transmitted packets. In the same way, an eavesdropper can retrieve original data from any 4 coded packets. Thus, the eavesdropping probability in this network is $0.1^3 + 2 \times 0.1^2 \times 0.9 = 0.019$. In other words, the security of the network in this case is $1 - 0.019 = 0.981$. In the second case, we could increase the reliability of the data transmission by adding more redundancy, at the cost of increasing the system's vulnerability to eavesdropping attacks. In order to ensure that increasing the redundancy does not make the system vulnerable to eavesdroppers, we need to perform a trade-off between the reliability of the network and its security. Furthermore, the level of redundancy on each path needs to optimized depending on the reliability and the security of the path.

In this work, we use an approach similar to the secure and fault-tolerant data storage work that we present in [3] to study the problem of secure and fault-tolerant data transmission over disjoint paths. Our work seeks to use network

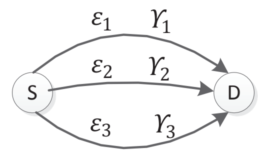


Fig. 1. Motivation example.

coding to provide reliable data transfer and as a light-weight security mechanism. We assume that there are multiple parallel disjoint paths between a source and a destination node, and that the source applies random linear network coding to the original data before the transition of its packets. We perform a trade-off between the reliability and the security of the data transmission. We formulate the problem as a mixed integer and linear programming problem for different cases, and we find solutions using approximation methods. We also analyze the performance of our proposed methods using extensive simulation results.

The remainder of the paper is as follows. We review related work and background knowledge of network coding in Section II. System model and objective of our work is discussed in Section III. We propose our reliable and secure data transmission methods in Section IV. Then, in Section V, we analyze our proposed methods though extensive simulation results. In Section VI, we conclude our work.

II. Related Work and Background

In the following subsections, we review related work and discuss the preliminaries of network coding and its applications. Our proposed secure and reliable data transmission method is based on random linear network coding. For this reason, we first provide background on network coding and its applications. Then, we review some of the applications of network coding, including providing reliable transmissions and security. We also review previous work that uses network coding as a security scheme.

A. Network Coding Preliminaries

Network coding technique [4]–[8] generalizes the traditional store-and-forward routing in networks. Network coding is proposed and used in [9] to achieve the capacity of a multicast session in wired networks. In [9], the authors prove that the maximum multicast capacity that can be achieved using network coding equals the min-cut from the source to the set of destinations of the multicast session. This is known as the min-cut max-flow theorem. It is proven in [10] that in order to achieve the capacity of a multicast session in a wired network, using linear network coding is sufficient. However, selecting the coefficients of the linear coded packets is a challenge.

The authors in [11] prove that if each intermediate node selects the coefficients of the coded packets randomly and in a distributed fashion, the generated linear coded packets will be likely linearly independent. Consequently, the proposed scheme (which is called random linear network coding) archives a rate very close to the capacity of a multicast session. The idea proposed in random linear network coding makes packet transmission very simple, by removing the complexities of coefficient selection. In [12], the authors take an algebraic look at network capacity and network coding, and they derive an interesting and useful model for linear network coding.

The coded packets that are generated in linear network coding are linear combinations of the original packets over a finite field (Galois field). In linear network coding, any linear mixture of coded packets is also a linear coded packet. In random linear network coding, the coefficients of the coded packets are selected randomly over a finite field. When the coefficients of the coded packets are selected randomly, the recoded packets are likely linearly independent. The form of the coded packets in random linear network coding is $\sum_{j=1}^{m} \beta_j \times P_j$, in which P_j is a packet. This packet can be an original packet or a coded packet. The coefficients of the linear combinations are shown as β_j.

Selecting the coefficients randomly and in a distributed manner makes random linear network coding appropriate for distributed systems and large networks. Similar to fountain codes (rateless codes) [13]–[16], the source node can potentially generate an infinite number of linearly coded packets. These coded packets can be transmitted over the network and recoded at intermediate relay nodes. Assuming that the source has m original packets to send, with a high probability the destination node is able to decode the coded packets using any m coded packets. The coding is done using Gaussian elimination. This characteristic makes random linter network coding a great tool for achieving reliable transmissions without feedback messages.

B. Applications of Network Coding

The first application of network coding was in solving the bottleneck problem and throughput maximization in wired networks. Network coding today has a wide range of applications, including providing reliable transmissions, throughput enhancement, protocol simplification, and security. Network coding is now even more attractive in wireless networks than in wired networks. This is because of both unreliability of wireless links and the broadcast nature of the medium. In the following subsections, we briefly review some of the applications of network coding and the previous work on providing security using network coding.

1) Throughput/Capacity Enhancement: COPE method [4] is one of the first practical methods proposed for data transmission in wireless networks using network coding. This method benefits from the broadcast nature of the wireless medium and overhearing among nodes, which both help to augment the throughput of the system. The main idea in COPE is that, in the cases where two crossing flows meet at a relay node and each of the destinations overhears the flow destined to the other destination, the relay node can combine these flows to reduce the number of transmissions. The authors extend this idea for greater numbers of crossing flows in their proposed method.

The work in [17]–[19] propose a one-hop reliable broadcasting using network coding. In order to ensure that all of the packets are received by the destination nodes, feedback messages are used. The proposed method has transmission and retransmission phases. In the first phase, the original packets are transmitted. Then, based on the feedback messages received from the destination nodes, the source decides how to use XOR network coding to reduce the number of required retransmissions to deliver missing packets. When network coding is used, each coded packet can deliver multiple lost packets to different destination nodes, reducing the number of retransmissions.

2) Reliable Transmission: One of the most important applications of network coding is in reliable transmission methods [20], especially in wireless networks which are more prone to packet erasures. Feedback messages are widely used in reliable transmission methods, like ARQ (automatic repeat request) method [21]. However, feedback messages have overhead, which is a major problem for multicasting. To reduce this overhead, hybrid-ARQ methods [22], [23], in which ARQ and a forward error correcting code (FEC) [24]–[26] are combined, can be used. Hybrid-ARQ methods reduce the number of feedback messages, but feedback messages are still needed. Linear network coding can be used in reliable transmission methods to provide reliability without feedback messages or with the minimum possible number of them. The important feature of linear network coding that makes it suitable for this application is that each of the coded packets contributes the same amount of information to the destination nodes. Consequently, the destination node only needs to receive a certain number of coded packets, not particular packets. Using linear network coding, the source node keeps transmitting coded packets until the destination (or destinations) receives a sufficient number of coded packets.

3) Protocol Simplification: One of the important applications of network coding is in simplifying network protocols. For example, a challenge in peer-ro-peer (P2P) networks [27]–[29] is retrieving a file from different peers that store different parts of the file. Since the original file is stored on different peers, a tracking mechanism is needed to know the location of different parts of data on different peers. Network coding simplifies the distribution of the file and its retrieval [30]. Network coded packets are distributed over the peers, and instead of knowing the peers that store a particular part of the file, we just need to know the number of coded packets stored on each peer.

Another example of the application of network coding in protocol simplification is in content distribution. Many content distribution problems are NP-complete, so they cannot be solved in polynomial time [31], [32]. Network coding can modify these problems to new problems that can be solved in polynomial time using techniques such as linear programming optimizations [33], [34].

4) Security: In [35], the authors propose a low-complexity cryptographic mechanism using random linear network coding. The authors propose encrypting the coefficients of the network coded packets instead of encrypting the original data. In this way, the complexity and overhead of the encryption is reduced. Because the size of the coefficients is much less than the size of the original data, which consequently, reduces the amount of the data that needs to be encrypted.

In [36], the idea of coding coefficients is extend and used for broadcasting multi-resolution videos [37]–[41]. In multi-resolution (multi-layer) videos, a video is divided to multiple videos, including a base layer and several enhancement layers. The base layer is necessary to watch the video, but enhancement layers increase the quality [42]. Multi-resolution videos are useful in multicasting or broadcasting a video to a set of users with different channel conditions. Another application of multi-resolution is providing different video qualities to a set of users subscribed to different services with different qualities of services. In the mentioned work, the authors encrypt the coefficients of the network coding packets of each layer with a different key, to prevent unauthorized users from receiving the video layers.

TABLE I
THE SET OF SYMBOLS USED IN THIS PAPER.

Notation	Definition
n	Number of parallel node and edge disjoint data paths
m	Number of packets in the original file
d_i	The ith data path
ϵ_i	Failure probability of data path d_i
γ_i	Access probability of the eavesdropper to the transmitted data over data path d_i
R_j	The set of paths that did not fail
S_j	The set of paths overheard by the eavesdropper
p_j	Failure probability of the data paths not in set R_j
q_j	Access probability of the eavesdropper to the data paths in set S_j
x_i	Portion of transmitted file on the ith data path
y_j	Boolean variable which shows whether paths in set R_j transmits m coded packets
z_j	Boolean variable which shows whether paths in set S_j transmits m coded packets
U	Utility function
α_1	The assigned weights to security
α_2	The assigned weights to fault tolerance
t_1/t_2	Threshold for fault tolerance/security

III. SYSTEM MODEL

We consider a network consisting of a source, a destination and multiple relay nodes. We assume that it is possible to find n parallel paths that are node and link disjoint between the source and destination node. Each of these paths might fail with a given probability. The failure can be caused by node failure, link failure, interference, or noise. We represent the failure probability of the ith path between the source and destination as ϵ_i. Moreover, each path is subject to eavesdropping attacks. The probability that an eavesdropper has access to the packets transmitted over the ith data path is represented as γ_i.

The source node has a file to transmit to the destination node. In order to provide fault tolerance, the source node applies random linear network coding to the original file, and network coded packets are transmitted through the n disjoint data paths. In more detail, the original file is first partitioned into m packets, and then random linear network coding is performed among the m packets to generate coded packets. Using random linear network coding, the source node can transmit redundant linear coded packets through the n different paths. The destination node needs to receive at least m coded packets to be able to decode the coded packets and retrieve the original packets. Once the decoding is successful, the destination node can merge the original packets to generate the file.

More redundancy in the data transmission enhances the fault-tolerance of the system against path failures. However, it may increase the vulnerability of the system to eavesdropping attacks since more transmitted packets increase the chance that an eavesdropper will obtain a sufficient number of coded packets (m in our model). In this work, we want to perform a trade-off between the reliability of the system and its robustness against the eavesdropper. A data path might be robust against failure but lack security. In this case, we need to make a decision about the number of packets to be transmitted on the path.

IV. SECURE DATA TRANSMISSION

Our objective is to design a fault-tolerant and secure data transmission scheme. Since more redundancy can reduce security and increase fault-tolerance, we need to perform a trade-off between security and reliability. In the following subsections, we first formulate our problem as mixed integer and linear programming optimizations. We then propose our secure and fault-tolerant data transmission methods, which find the number of packets that need to be transmitted from the source to the destination on each of the parallel paths in such a way that certain levels of security and fault-tolerance are met.

A. Formulation

Similar to our previous work on fault-tolerant and secure distributed data storage [3], we can formulate the discussed fault-tolerant and secure data transmission problem in the following three cases.

- Case 1: In this case, we assume that a certain level of fault tolerance needs to be met. For this purpose, we fix the fault tolerance at a specific threshold, denoted as t_1, and set it as a constraint of the optimization. This threshold is the minimum required fault tolerance of the system. We then set the objective as minimization of the eavesdropping probability.
- Case 2: This case is the reverse of the optimization in Case 1. We assume that a certain level of security needs to be met. Therefore, we set a threshold for the security of the system, denoted as t_2, and use it as a constraint of the optimization. The objective of this optimization is to maximize the fault-tolerance of the data transmission.

- Case 3: In the third case, there is no limit on the security or fault-tolerance of the data transmission. Instead, we define the objective function as a maximizing of a function of fault tolerance and security.

The eavesdropper needs to receive at least m linearly independent coded packets to be able to decode the coded packets and retrieve the original file. In the other words, if the eavesdropper receives less that m coded packets, it will not be able to retrieve the original file. It can be proved that if we use a sufficiently large finite field to linearly code the packets, with a high probability, any m random linear coded packets will be linearly independent and sufficient to retrieve the original file [11]. For any possible paths failure case, we represent the set of paths that do not fail as R_j, and the set of all of these sets as R. Moreover, for any eavesdropping scenario, we represent the set of paths that are overheard by the eavesdropper as S_j, and the set of all of these sets as S. Additionally, we use boolean variables y_j and z_j to show whether the destination and the eavesdropper are able to retrieve the original file from the set of packets are transmitted over paths in S_j and R_j, respectively. The set of notations used in this work are shown in Table I.

If at least m coded packets are transmitted on the set of paths in S_j, the eavesdropper can retrieve the file. In this case, z_j has a value equal to 1; otherwise, its value is 0. In the same way, if at least m packets are transmitted over the paths in R_j, the value of variable y_j is 1, which means the destination node is able to decode the coded packets and retrieve the original file. Consequently, if we represent the probability that R_j and S_j happen as q_j, the probability that the eavesdropper and the destination can retrieve the original file is $q_j z_j$ and $p_j y_j$, respectively. The probability that an eavesdropper can receive transmitted packets on path d_i equals γ_i. Thus, the probability that an eavesdropper only has access to data transmitted on the set of paths in S_j can be calculated as follows:

$$q_j = \prod_{d_i \in S_j} \gamma_i \prod_{d_i \notin S_j} (1 - \gamma_i) \tag{1}$$

Moreover, the failure probability of path d_i is denoted as ϵ_i. Thus, the probability that the paths in set R_j do not fail and the rest of the data paths fail can be calculated as:

$$p_j = \prod_{d_i \in R_j} (1 - \epsilon_i) \prod_{d_i \notin R_j} \epsilon_i \tag{2}$$

In the following subsections, we formulate our problem in the discussed three cases.

1) Case 1: In the first case, we want to achieve a minimum fault-tolerance of t_1. Our objective is to minimize the probability that an eavesdropper can receive m coded packets and retrieve the original packets. As a result, the distribution of packets over different paths needs to be done in such a way that it does not violate the minimum fault-tolerance threshold while minimizing the eavesdropper probability. This problem can be formulated as the following mixed integer and linear programming optimization:

$$\min\ U = \sum_{S_j \in S} q_j z_j \tag{3}$$

$$s.t \quad \sum_{R_j \in R} p_j y_j \geq t_1 \tag{4}$$

$$y_j \leq \sum_{d_i \in R_j} x_i \quad \forall\ R_j \in R \tag{5}$$

$$z_j \leq \sum_{d_i \in S_j} x_i \quad \forall\ S_j \in S \tag{6}$$

$$y_j, z_j \in \{0, 1\} \quad \forall R_j \in R,\ S_j \in S \tag{7}$$

The objective function of this optimization is $\min \sum_{R_j \in R} q_j z_j$, which minimizes the probability of successful eavesdropping. Constraint (4) represents the minimum reliability of the system should not be less than threshold t_1. Here, $\sum_{R_j \in R} p_j y_j$ is the probability that the original file can be retrieved by the destination node. Variables y_j and z_j are integer variables with values 0 or 1, which denote whether a given failure and eavesdropping scenario will result in a successful eavesdropping and file delivery to the destination node. We represent the fraction of the original file that is transmitted on data path d_i as x_i. Constraint (5) sets y_j to 1 in a case where the destination node can retrieve the original packets by successfully receiving packets transmitted over the paths in R_j. Constraint (6) sets z_j to 1 if an eavesdropper can retrieve the original file by overhearing the packets transmitted on the set of paths in S_j.

2) Case 2: In the second case, which is the reverse of case 1, our objective is to maximize the system fault-tolerance against path failures. Moreover, the probability of a successful eavesdropping should not be greater than threshold t_2. In this case, the problem can be formulated as the following mixed integer and linear programming optimization:

$$\max\ U = \sum_{R_j \in R} p_j y_j \tag{8}$$

$$s.t \quad \sum_{R_j \in R} q_j z_j \leq t_2 \tag{9}$$

$$y_j \leq \sum_{d_i \in R_j} x_i \quad \forall\ R_j \in R \tag{10}$$

$$z_j \leq \sum_{d_i \in S_j} x_i \quad \forall\ S_j \in S \tag{11}$$

$$y_j, z_j \in \{0, 1\} \quad \forall\ R_j \in R, S_j \in S \tag{12}$$

In this optimization, the objective function (8) is maximizing the probability of the successful delivery of the file to the destination node. Constraint (9) is the constraint on the maximum vulnerability of the system. Similar to Case 1, Constraints (11) and (10) set z_j and y_j to 1 or 0 depending on whether or not the packets transmitted over the paths in R_j result in a successful eavesdropping and the successful delivery of the packets to the destination node.

3) Case 3: In the last case, instead of setting a threshold for the fault-tolerance or the security of the system, we perform a trade-off between security and fault tolerance. Transmitting more redundant data on each of the paths enhances robustness against path failures. However, it makes the data transmission more vulnerable to eavesdropping attacks.

In order to perform a trade-off, we define a weighted sum of the security and fault-tolerance as the objective function. The objective function of the optimization is $U = \alpha_1 u_1 - \alpha_2 u_2$, in which u_1 is the probability that the eavesdropper can retrieve the original file from the overheard transmissions. Also, u_2 represents the probability that the destination node can retrieve the original file from the received coded packets. Furthermore, constants α_1 and α_2 are the assigned weights of the security and reliability of the system. We can perform the trade-off using the following mixed integer and linear programming optimization:

$$\min\ U = \sum_{S_j \in S} \alpha_1 q_j z_j - \sum_{R_j \in R} \alpha_2 p_j y_j \tag{13}$$

$$y_j \le \sum_{d_i \in R_j} x_i \quad \forall\ R_j \in R \tag{14}$$

$$z_j \le \sum_{d_i \in S_j} x_i \quad \forall\ S_j \in S \tag{15}$$

$$y_j, z_j \in \{0,1\} \quad \forall\ R_j \in R, S_j \in S \tag{16}$$

The objective function of this optimization is a weighted sum of the security and fault tolerance. Similar to Cases 1 and 2, variables y and z are integer variable. The set of Constraints (14), (15), and (16) sets the value of these two variables to 0 or 1.

B. Data Transmission Scheme

All of the three optimizations discussed in these three cases are mixed integer and linear programming. In general, the solution of a mixed integer and linear programming optimization cannot be found in polynomial time. As a result, we need to modify the proposed optimization problems to optimizations that can be solved faster. One approach to find an approximation solution for a mixed integer and linear programming is to relax it to a linear programming optimization. Linear programming optimizations can be efficiently solved using different techniques, such as the Gradient method.

Using relaxation technique, the problem in Case 1 can be formulated as the following linear programming, denoted as LP1 optimization:

$$\min\ U = \sum_{S_j \in S} q_j z_j \tag{17}$$

$$s.t \quad \sum_{R_j \in R} p_j y_j \ge t_1 \tag{18}$$

$$y_j \le \sum_{d_i \in R_j} x_i \quad \forall\ R_j \in R \tag{19}$$

$$z_j \le \sum_{d_i \in S_j} x_i \quad \forall\ S_j \in S \tag{20}$$

$$y_j, z_j \in (0,1) \quad \forall\ R_j \in R, S_j \in S \tag{21}$$

In this optimization, we relaxed the integer variables z_j and y_j to variables with real values. This changes the optimization from mixed integer and linear programming to linear programming. Linear programming optimizations can be solved using the gradient method and other standard optimization techniques. However, the complexity of these methods is a polynomial function of the number of the variables and constraints in the optimization. In LP1, the number of sets in set R is exponential. As a result, in case where the number of parallel paths between the source and the destination node is high, the complexity of the solution to LP1 will be exponential. Consequently, we propose an approximation of the LP1 optimization, denoted as LP2:

$$\min\ U = \sum_{d_i \in D} \gamma_i x_i \tag{22}$$

$$s.t \quad \sum_{r_i \in S} (1 - \epsilon_i) x_i \ge t_1 \tag{23}$$

$$x_i \in (0,1) \quad \forall\ d_i \in D \tag{24}$$

Here, D represents the set of all paths between the source and the destination node. If we apply the discussed relaxations of the mixed integer and linear programming to Case 2, the optimization in Case 2 becomes as follows:

$$\max\ U = \sum_{d_i \in D} (1 - \epsilon_i) x_i \tag{25}$$

$$s.t \quad \sum_{r_i \in S} \gamma_i x_i \ge t_2 \tag{26}$$

$$x_i \in (0,1) \quad \forall\ d_i \in D \tag{27}$$

Finally, the optimization in Case 3 can be relaxed to the following optimization:

$$\max\ U = \sum_{d_i \in D} \alpha_1 \gamma_i x_i - \alpha_2 (1 - \epsilon_i) x_i \tag{28}$$

$$s.t \quad x_i \in (0,1) \quad \forall\ d_i \in D \tag{29}$$

C. Extension

In the previous sections, we considered a general model in which the eavesdropping probability of the links were random numbers. Also, we assumed that there were m parallel paths between the source and the destination node. This model can be easily extended to a more general model in Figure 2, in which the eavesdropping probability follows a special pattern without affecting our proposed method. For example, it is logical to assume that the eavesdropping probability of the links decreases based on the distance of the links from eavesdropper. The links that are closer to the eavesdropper are more likely to be overheard by the eavesdropper. The links that are not close to the eavesdropper can be still overheard, but with a lower probability. For example, we can consider the Rayleigh fading model [43] to calculate the overhearing probability:

$$P = \int_{T^*}^{\infty} \frac{2x}{\sigma^2} e^{-\frac{x^2}{\sigma^2}} dx \tag{30}$$

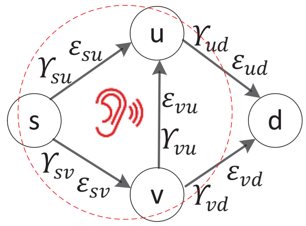


Fig. 2. Extended system model.

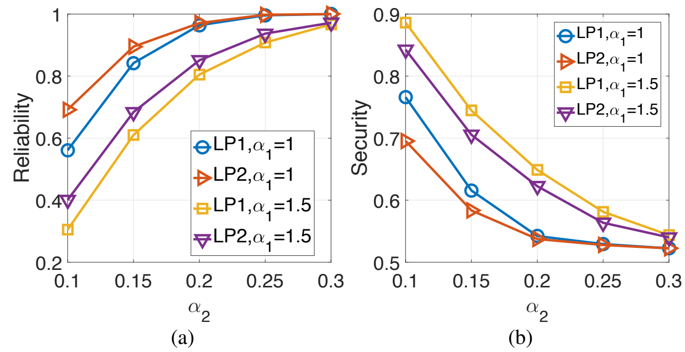


Fig. 3. The effect of α_2 on (a) the reliability and (b) the security of data transmission. $\epsilon_i \in [0, 0.1]$, $\gamma_i \in [0, 0.3]$.

where

$$\sigma^2 \triangleq \frac{1}{(4\pi)^2 L^\theta} \tag{31}$$

Here, T^* and L are the decodable SNR threshold and the distance between two nodes. Also, θ is the path loss order. This model is typically used to model the loss rate of the wireless links between two nodes. In case where the distance between two nodes is greater, the loss probability of the link between them will also be greater due to the higher noise level. Noise cannot affect the communication of two nodes that are close to each other. The same idea can be used to model the behavior of eavesdroppers. If an eavesdropper is too close to a link, there is a higher chance of overhearing transmitted packets over that link. Links that are far from the eavesdropper are less vulnerable to eavesdropping. In order to use our proposed methods in the previous section, we first need to find parallel paths between the source and the destination node. To this end, we can use methods like [44], which can find disjoint paths.

V. Evaluations

In this section, we report the simulation results of our proposed secure and fault-tolerant data transmission schemes. We first discuss the simulation environment and setting. Then, we present the simulation results and a summary of our findings.

A. Simulation Setting

We implemented our simulations in the Matlab environment. In order to find the solution of the proposed optimization in the previous section, we used an optimization tool built into Matlab called Linprog. This tool solves linear programming optimizations. Linprog does not accept equality constraint. Because of this, we had to convert the equality constraint to inequality equations. For this purpose, each equality constraint needs to be replaced with a greater than or equal, and a less than or equal constraint.

In order to have a reasonable confidence interval, we run our simulations on 1000 networks with randomly selected paths reliability and eavesdropping probability. In our simulations, we compare the security and reliability of the data transmissions in case 3, by measuring the effect of the following metrics:

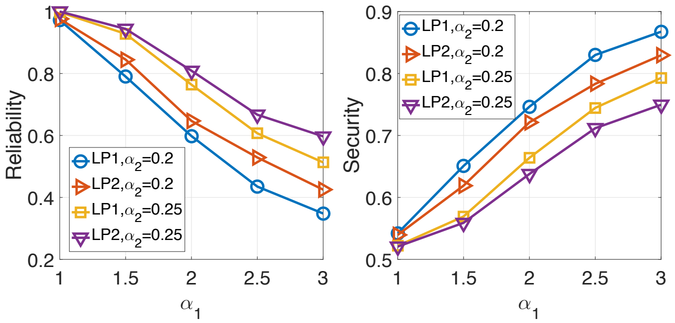
- α_1: the assigned weight to security. A greater α_1 gives more importance to security, which results in a more secure data transmission.
- α_2: the weight of the fault tolerance in the optimizations. In contrast to α_1, a greater α_2 results in a more reliable data transmission scheme.
- ϵ: the probability of data path failure. In the simulation results, we study the effect of changes in the path failure probability on the fault-tolerance and security of our proposed methods.

In the simulations, LP1 and LP2 represent the two relaxed optimizations in Case 3. In all of the simulations, there are 4 parallel (disjoint) paths between the source and the destination. The number of original packets sent is 10.

B. Simulation Result

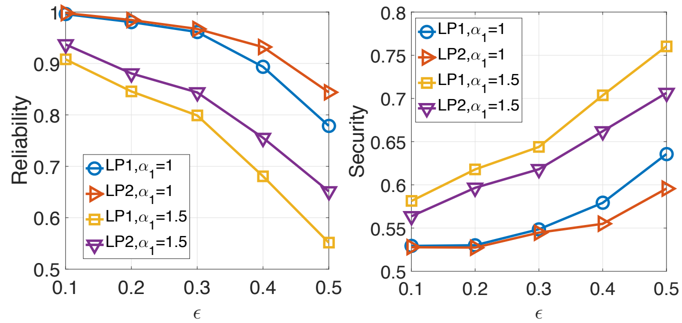
In Figure 3(a), we show the reliability of using LP1 and LP2 for case 3. The path failure probability of the links is chosen randomly in the range of $[0, 0.1]$. Moreover, the eavesdropping probability of each path (γ) is a random number in the range of $[0, 0.3]$. In Figure 3(a), the reliability of the LP1 and LP2 methods are compared when $\alpha_1 = 1$ and $\alpha_1 = 1.5$. Increasing α_2 gives more importance to the reliability of the system than to its security. For this reason, as we increase α_2, our proposed methods find more reliable transmission schemes. This is why all of the curves in Figure 3(a) have a positive slope. The figure also shows that the reliability of the LP1 and LP2 methods when $\alpha_1 = 1$ are greater than when $\alpha_1 = 1.5$. The reason is that, a greater α_1 increases the weight assigned to the security of the transmission in the optimizations. The simulation results show that the performance of the LP1 and LP2 methods are similar. As discussed in the previous section, the complexity of the LP2 method is less than that of LP1.

In our second experiment, we analyze the security of our data transmission methods. Security is defined as the probability of a successful eavesdropping. In a successful eavesdropping, the eavesdropper must receive m coded packets. In Figure 3(b), we change α_1 from 0.1 to 0.3 and measure the effect on security of the data transmission. The failure of the parallel paths and their eavesdropping probability are selected randomly in the range of $[0, 0.1]$ and $[0, 0.3]$. The security of the LP1 and LP2 methods decreases as we increase α_2, due to the increased importance of the reliability of the data



(a) (b)

Fig. 4. The effect of α_2 on (a) the reliability and (b) the security of the system. $\epsilon_i \in [0, 0.1]$, $\gamma_i \in [0, 0.5]$.



(a) (b)

Fig. 5. The effect of ϵ on the (a) reliability and (b) the security of the system. $\gamma_i \in [0, 0.5]$.

transmission. A greater α_2 results in the transmission of more redundant coded packets. Thus, there is a higher chance that an eavesdropper will receive m coded packets. The security of the LP1 and LP2 methods in Figure 3(b) are similar. From Figures 3(a) and (b), we can infer that as the reliability of our data transmission methods increases, security decreases.

In Figures 4(a) and (b), we repeat the two previous experiments, but this time, we fix α_2 and change α_1. In Figure 4(a), failure probabilities are selected in the range of $[0, 0.1]$ and the eavesdropping probability of each link is in the range of $[0, 0.3]$. As the figure shows, increasing α_1 reduces the reliability of data transmission. Also, the reliability of the data transmission methods when $\alpha_2 = 0.25$ is greater than when $\alpha_2 = 0.2$. Similar to the previous experiments, there is little difference between the LP1 and LP2 methods.

In the next experiment, we fix α_2 and change α_1 from 1 to 3. Like in the previous experiments, we select the failure rates of each path randomly in the range of $[0, 0.1]$ and the eavesdropping probabilities in the range of $[0, 0.3]$. Figure 4(b) shows that increasing α_1 enhances the security of the data transmission. However, as shown in Figure 4(a), this security enhancement comes at the cost of decreasing the reliability of the data transmission.

In the next two experiments, we analyze the effect of path failure probability on the reliability and the security of our data transmission methods. For this purpose, we increase the range of failure probability from $[0, 0.1]$ to $[0.4, 0.5]$ in increments of 0.1, and we measure the performance of the LP1 and LP2 methods. The eavesdropping probability of the paths are selected in the range of $[0, 0.5]$. We set α_2 to 0.25, and we run the simulations with $\alpha_1 = 1$ and $\alpha_1 = 1.5$. In Figure 5(a), the reliability of the data transmission decreases as the path failure rates increase. Also, the reliability of the LP2 method is slightly greater than that of the $LP1$ method. Also, a greater α_1 reduces the reliability of the methods.

In our last experiment, we study the effect that the path reliability has on the security of the data transmission. In Figure 5(b), we increase the path failure probability and measure the probability of a successful eavesdropping. The figure shows that as we increase the path failure rate, the security of the data transmission enhances. This is because our objective function is a linear function of the security and reliability. In order to maximize the objective function, depending on the weights assigned to reliability and security, security or reliability needs to be increased. When the failure rate of a path increases dramatically, too many transmissions are needed to keep the system fault-tolerant. As a result, in cases where α_1 is high, transmitting fewer packets and increasing the security of the system maximizes the objective function.

C. Simulation Summary

Our findings from the simulation results can be summarized as follows:

- When random linear network coding is used to provide security and reliability, reliability and security have a negative correlation.
- Based on the simulation results, the performance of the LP1 and LP2 methods, in terms of reliability and security, are very similar to each other.
- In cases where α_1 is high, increasing the path failure rate increases the security of the system.

VI. Conclusion

Random linear network coding is a technique in which packets are mixed using an algebraic approach. This technique has many applications in both wired and wireless networks, including, but not limited to, enhancing network throughput, reliable transmissions, and fault-tolerant data storage. Using this technique, a set of packets can be encoded to a potentially infinite number of coded packets, and only a subset of these linearly coded packets are needed to recover the original packets. In addition to the mentioned applications, linear network coding can also be used to protect original packets from unauthorized access. An unauthorized user, e.g. an eavesdropper, is not able to retrieve the original packets without receiving a certain number of coded packets.

In this paper, we study the application of network coding to design a secure and fault-tolerant data transmission mechanism. In general, more redundancy increases the fault-tolerance of a system. On the other hand, more redundancy makes the system vulnerable to eavesdropping attacks by increasing the chance that an unauthorized user will receive a sufficient number of coded packets. In our work, we use random linear network coding to achieve security and fault-tolerance at the same time. We assume that packets can be transmitted from a source node to a destination by different paths. Based on this model, we propose 3 different optimization methods to find the number of packets that should be transmitted via these paths. We analyze our methods using extensive simulation results.

ACKNOWLEDGMENT

This research was supported in part by NSF grants CNS 1629746, CNS 1564128, CNS 1449860, CNS 1461932, CNS 1460971, and CNS 1439672.

REFERENCES

[1] P. F. Oliveira, L. Lima, T. T. Vinhoza, J. Barros, and M. Médard, "Trusted storage over untrusted networks," in *IEEE GLOBECOM 2010*, 2010, pp. 1–5.

[2] ——, "Coding for trusted storage in untrusted networks," *IEEE Transactions on Information Forensics and Security*, vol. 7, no. 6, pp. 1890–1899, 2012.

[3] P. Ostovari and J. Wu, "Fault-tolerant and secure distributed data storage using random linear network coding," in *IEEE WiOpt*, 2016, pp. 1–8.

[4] S. Katti, H. Rahul, W. Hu, D. Katabi, M. Médard, and J. Crowcroft, "Xors in the air: practical wireless network coding," in *ACM SIGCOMM*, 2006, pp. 243–254.

[5] S. Chachulski, M. Jennings, S. Katti, and D. Katabi, "Trading structure for randomness in wireless opportunistic routing," in *ACM SIGCOMM*, 2007.

[6] D. Koutsonikolas, C. Wang, and Y. Hu, "CCACK: Efficient network coding based opportunistic routing through cumulative coded acknowledgments," in *IEEE INFOCOM*, 2010, pp. 1–9.

[7] ——, "Efficient network coding based opportunistic routing through cumulative coded acknowledgments," *IEEE/ACM Transactions on Networking*, vol. 19, no. 5, pp. 1368–1381, Feb 2011.

[8] P. Ostovari, J. Wu, and A. Khreishah, "Network coding techniques for wireless and sensor networks," in *The Art of Wireless Sensor Networks*, H. M. Ammari, Ed. Springer, 2013.

[9] R. Ahlswede, N. Cai, S. Li, and R. Yeung, "Network information flow," *IEEE Transactions on Information Theory*, vol. 46, no. 4, pp. 1204–1216, 2000.

[10] S. Li, R. Yeung, and N. Cai, "Linear network coding," *IEEE Transactions on Information Theory*, vol. 49, no. 2, pp. 371–381, 2003.

[11] T. Ho, M. Médard, R. Koetter, D. Karger, M. Effros, J. Shi, and B. Leong, "A random linear network coding approach to multicast," *IEEE Transactions on Information Theory*, vol. 52, no. 10, pp. 4413–4430, 2006.

[12] R. Koetter and M. Medard, "An algebraic approach to network coding," *IEEE/ACM Transactions on Networking*, vol. 11, no. 5, pp. 782– 795, Oct 2003.

[13] M. Luby, "LT codes," in *The 43rd Annual IEEE Symposium on Foundations of Computer Science*, 2002, pp. 271–280.

[14] A. Shokrollahi, "Raptor codes," *IEEE Transactions on Information Theory*, vol. 52, no. 6, pp. 2551–2567, 2006.

[15] P. Cataldi, M. Shatarski, M. Grangetto, and E. Magli, "Lt codes," in *IIH-MSP'06*, 2006, pp. 263–266.

[16] D. J. MacKay, "Fountain codes," *IEE Proceedings- Communications*, vol. 152, no. 6, pp. 1062–1068, 2005.

[17] L. Lu, M. Xiao, M. Skoglund, L. Rasmussen, G. Wu, and S. Li, "Efficient network coding for wireless broadcasting," in *IEEE WCNC*, 2010, pp. 1–6.

[18] L. Lu, M. Xiao, and L. Rasmussen, "Relay-aided broadcasting with instantaneously decodable binary network codes," in *ICCCN*, 2011, pp. 1–5.

[19] D. Nguyen, T. Tran, T. Nguyen, and B. Bose, "Wireless broadcast using network coding," *IEEE Transactions on Vehicular Technology*, vol. 58, no. 2, pp. 914–925, 2009.

[20] P. Ostovari, A. Khreishah, J. Wu, and W.-S. Yang, "Trade-off between redundancy and feedback in wireless network communication," *Adhoc & Sensor Wireless Networks*, vol. 24, 2015.

[21] H. Djandji, "An efficient hybrid arq protocol for point-to-multipoint communication and its throughput performance," *IEEE Transactions on Vehicular Technology*, vol. 48, no. 5, pp. 1688–1698, 1999.

[22] B. Zhao and M. Valenti, "The throughput of hybrid-ARQ protocols for the gaussian collision channel," *IEEE Transactions on Information Theory*, vol. 47, no. 5, pp. 1971–1988, 2001.

[23] L. Rizzo and L. Vicisano, "RMDP: an FEC-based reliable multicast protocol for wireless environments," *ACM SIGMOBILE Mobile Computing and Communications Review*, vol. 2, no. 2, pp. 23–31, 1998.

[24] G. C.Clark and J. B. Cain, *Error-correction coding for digital communications*. Springer, 1981.

[25] S. Lin and D. J. Costello, *Error control coding: Fundamentals and Applications*. Prentice-hall Englewood Cliffs, NJ, 2004.

[26] W. Ryan and S. Lin, *Channel codes: classical and modern*. Cambridge University Press, 2009.

[27] G. Fox, "Peer-to-peer networks," *Computing in Science & Engineering*, vol. 3, no. 3, pp. 75–77, 2001.

[28] Y. Liu, Y. Guo, and C. Liang, "A survey on peer-to-peer video streaming systems," *Peer-to-peer Networking and Applications*, vol. 1, no. 1, pp. 18–28, 2008.

[29] J. Liu, S. G. Rao, B. Li, and H. Zhang, "Opportunities and challenges of peer-to-peer internet video broadcast," *Proceedings of the IEEE*, vol. 96, no. 1, pp. 11–24, 2008.

[30] C. Gkantsidis and P. R. Rodriguez, "Network coding for large scale content distribution," in *IEEE INFOCOM 2005*, vol. 4, 2005, pp. 2235–2245.

[31] S. Pawar, S. El Rouayheb, H. Zhang, K. Lee, and K. Ramchandran, "Codes for a distributed caching based video-on-demand system," in *IEEE ASILOMAR*, 2011, pp. 1783–1787.

[32] H. Zhang, M. Chen, A. Parekh, and K. Ramchandran, "A distributed multichannel demand-adaptive p2p vod system with optimized caching and neighbor-selection," in *SPIE*, 2011, pp. 81 350X–81 350X.

[33] P. Ostovari, J. Wu, A. Khreishah, and N. B. Shroff, "Scalable video streaming with helper nodes using random linear network coding," *IEEE/ACM Transactions on Networking*, vol. PP, no. 99, pp. 1–14, 2015.

[34] P. Ostovari and J. Wu, "Towards network coding for cyber-physical systems: Security challenges and applications," in *Security & Security and Privacy in Cyber-Physical Systems: Foundations and Applications*, H. Song, Ed. Wiley, 2017.

[35] J. P. Vilela, L. Lima, and J. Barros, "Lightweight security for network coding," in *IEEE ICC*, 2008, pp. 1750–1754.

[36] L. Lima, S. Gheorghiu, J. Barros, M. Médard, and A. L. Toledo, "Secure network coding for multi-resolution wireless video streaming," *IEEE Journal on Selected Areas in Communications*, vol. 28, no. 3, pp. 377–388, 2010.

[37] M. Shao, S. Dumitrescu, and X. Wu, "Layered multicast with interlayer network coding for multimedia streaming," *IEEE Transactions on Multimedia*, vol. 13, no. 99, pp. 353–365, 2011.

[38] S. McCanne, V. Jacobson, and M. Vetterli, "Receiver-driven layered multicast," in *ACM CCR*, 1996, pp. 117–130.

[39] M. Kim, D. Lucani, X. Shi, F. Zhao, and M. Médard, "Network coding for multi-resolution multicast," in *IEEE INFOCOM*, 2010, pp. 1–9.

[40] N. Shacham, "Multipoint communication by hierarchically encoded data," in *IEEE INFOCOM*, 1992, pp. 2107–2114.

[41] M. Effros, "Universal multiresolution source codes," *IEEE Transactions on Information Theory*, vol. 47, no. 6, pp. 2113–2129, 2001.

[42] P. Ostovari, A. Khreishah, and J. Wu, "Multi-layer video streaming with helper nodes using network coding," in *IEEE MASS*, 2013, pp. 524–532.

[43] C. Wang, A. Khreishah, and N. Shroff, "Cross-layer optimizations for intersession network coding on practical 2-hop relay networks," in *Asilomar*, 2009, pp. 771–775.

[44] D. Sidhu, R. Nair, and S. Abdallah, "Finding disjoint paths in networks," *ACM SIGCOMM Computer Communication Review*, vol. 21, no. 4, pp. 43–51, 1991.