Resilient Network Coding in the Presence of Byzantine Adversaries

Sidharth Jaggi, *Member, IEEE*, Michael Langberg, Sachin Katti, Tracey Ho, *Member, IEEE*, Dina Katabi, Muriel Médard, *Fellow, IEEE*, and Michelle Effros, *Senior Member, IEEE*

***Abstract*—Network coding substantially increases network throughput. But since it involves mixing of information inside the network, a single corrupted packet generated by a malicious node can end up contaminating all the information reaching a destination, preventing decoding.**

This paper introduces distributed polynomial-time rate-optimal network codes that work in the presence of Byzantine nodes. We present algorithms that target adversaries with different attacking capabilities. When the adversary can eavesdrop on all links and jam z_O links, our first algorithm achieves a rate of $C - 2z_O$, where C is the network capacity. In contrast, when the adversary has limited eavesdropping capabilities, we provide algorithms that achieve the higher rate of $C - z_O$.

Our algorithms attain the optimal rate given the strength of the adversary. They are information-theoretically secure. They operate in a distributed manner, assume no knowledge of the topology, and can be designed and implemented in polynomial time. Furthermore, only the source and destination need to be modified; nonmalicious nodes inside the network are oblivious to the presence of adversaries and implement a classical distributed network code. Finally, our algorithms work over wired and wireless networks.

***Index Terms*—Byzantine adversaries, distributed network error-correcting codes, eavesdroppers, information-theoretically optimal, list decoding, polynomial-time algorithms.**

I. Introduction

NETWORK coding allows the routers to mix the information content in packets before forwarding them. This mixing has been theoretically proven to maximize network throughput [1], [23], [21], [15]. It can be done in a distributed manner with low complexity, and is robust to packet losses and network failures [10], [25]. Furthermore, recent implementations of network coding for wired and wireless environments demonstrate its practical benefits [18], [8].

But what if the network contains malicious nodes? A malicious node may pretend to forward packets from source to destination, while in reality it injects corrupted packets into the information flow. Since network coding makes the routers mix packets' content, a single corrupted packet can end up corrupting *all* the information reaching a destination. Unless this problem is solved, network coding may perform much worse than pure forwarding in the presence of adversaries.

The interplay of network coding and Byzantine adversaries has been examined by a few recent papers. Some detect the presence of an adversary [12], others correct the errors he injects into the codes under specific conditions [9], [14], [22], [31], and a few bound the maximum achievable rate in such adverse environments [3], [29]. But attaining optimal rates using distributed and low-complexity codes was an open problem.

This paper designs distributed polynomial-time rate-optimal network codes that combat Byzantine adversaries.[1] We present three algorithms that target adversaries with different strengths. The adversary can inject z_O packets per unit time, but his listening power varies. When the adversary is omniscient, i.e., he observes transmissions on the entire network, our codes achieve the rate of $C-2z_O$, with high probability. When the adversary's knowledge is limited, either because he eavesdrops only on a subset of the links or the source and destination have a low-rate secret channel, our algorithms deliver the higher rate of $C-z_O$.

The intuition underlying all of our algorithms is that the aggregate packets from the adversarial nodes can be thought of as a second source. The information received at the destination is a linear transform of the source's and the adversary's information. Given enough linear combinations (enough coded packets), the destination can decode both sources. The question however is how does the destination distill out the source's information from the received mixture. To do so, the source's information has to satisfy certain constraints that the attacker's data cannot satisfy. This can be done by judiciously adding redundancy at the source. For example, the source may add parity checks on the source's original data. The receiver can use the syndrome of the received packets to determine the effect of the adversary's transmissions. The challenge addressed herein is to design the parity checks for distributed network codes that achieve the optimal rates.

[1]Independently and concurrently to our work, Koetter and Kschischang [19] present results of similar nature which are discussed in detail in Section II.

Manuscript received November 23, 2006; revised February 21, 2008. This material is based upon work supported by the Air Force Office of Scientific Research under Grant FA9550-06-1-0155, the National Science Foundation under Grants CCR-0325496 and CCF-0325324, the Chinese University of Hong Kong under Direct Grant 2050394, and Caltech's Lee Center for Advanced Networking. The material in this paper was presented at the IEEE INFOCOM, Anchorage, AK, May 2007.

S. Jaggi is with the Department of Information Engineering, Chinese University of Hong Kong, Shatin, N.T., Hong Kong (e-mail: jaggi@ie.cuhk.edu.hk).

M. Langberg is with the Computer Science Division, The Open University of Israel, Raanana 43107 Israel (e-mail: mikel@openu.ac.il).

S. Katti and D. Katabi are with the Computer Science and Artificial Intelligence Laboratory, Massachusetts Institute of Technology, Cambridge, MA 02139 USA (e-mail: sachin@csail.mit.edu; dina@csail.mit.edu).

T. Ho and M. Effros are with the Department of Electrical Engineering, California Institute of Technology, Pasadena, CA 91125 USA (e-mail: tho@caltech.edu; effros@caltech.edu).

M. Médard is with the Laboratory for Information and Decision Systems, Massachusetts Institute of Technology, Cambridge, MA 02139 USA (e-mail: medard@mit.edu).

Communicated by U. Maurer, Guest Editor for Special Issue on Information Theoretic Security.

Digital Object Identifier 10.1109/TIT.2008.921711

Conceptually, our proof involves two steps. We first analyze standard network coding in the presence of Byzantine adversaries (without adding additional redundancy at the source). In this setting, as expected, destination nodes cannot uniquely decode the source's data, however, we show that they can *list decode* this data. Namely, receivers can identify a *short* list of potential messages that may have been transmitted. Once this is established, we analyze the effect of redundancy at the source in each one of our scenarios (omniscient or limited adversaries).

This paper makes several contributions. The algorithms presented herein are distributed algorithms with polynomial-time complexity in design and implementation, yet are rate-optimal. In fact, since pure forwarding is a special case of network coding, being rate-optimal, our algorithms also achieve a higher rate than any approach that does not use network coding. They assume no knowledge of the topology and work in both wired and wireless networks. Furthermore, implementing our algorithms involves only a slight modification of the source and receiver while the internal nodes can continue to use standard network coding.

II. Related Work

Work on network coding started with a pioneering paper by Ahlswede *et al.* [1], which establishes the value of coding in the routers and provides theoretical bounds on the capacity of such networks. The combination of [23], [21], and [15] shows that, for multicast traffic, linear codes achieve the maximum capacity bounds, and both design and implementation can be done in polynomial time. Additionally, Ho *et al.* show that the above is true even when the routers perform random linear operations [10]. Researchers have extended the above results to a variety of areas including wireless networks [25], [17], [18], energy [28], secrecy [2], content distribution [8], and distributed storage [16]. For a couple of nice surveys on network coding see, e.g., [30], [7].

A Byzantine attacker is a malicious adversary hidden in a network, capable of eavesdropping and jamming communications. Prior research has examined such attacks in the presence of network coding and without it. In the *absence* of network coding, Dolev *et al.* [5] consider the problem of communicating over a known graph containing Byzantine adversaries. They show that for k adversarial nodes, reliable communication is possible only if the graph has more than $2k+1$ vertex connectivity. Subramaniam extends this result to unknown graphs [27]. Pelc *et al.* address the same problem in wireless networks by modeling malicious nodes as locally bounded Byzantine faults, i.e., nodes can overhear and jam packets only in their neighborhood [26].

The interplay of network coding and Byzantine adversaries was examined in [12], which detects the existence of an adversary but does not provide an error-correction scheme. The work of Cai and Yeung [2], [29], [3] generalizes standard bounds on error-correcting codes to networks, without providing any explicit algorithms for achieving these bounds. Our work presents a constructive design to achieve those bounds.

The problem of efficiently correcting errors in the presence of both network coding and Byzantine adversaries has been considered by a few prior proposals. Earlier work [22], [9] assumes a centralized trusted authority that provides hashes of the original packets to each node in the network. Charles *et al.* [4] obviates the need for a trusted entity under the assumption that the majority of packets received by each node is uncorrupted. Recently, Zhao *et al.* [32] have demonstrated error detection in the public key cryptographic setting. In contrast to the above schemes which are cryptographically secure, in a previous work [14], we consider an information-theoretically rate-optimal solution to Byzantine attacks for *wired* networks, which however requires a centralized design. This paper builds on the above prior schemes to combine their desirable traits; it provides a distributed solution that is information-theoretically rate optimal and can be designed and implemented in polynomial time. Furthermore, our algorithms have new features; they assume no knowledge of the topology, do not require any new functionality at internal nodes, and work for both wired and wireless networks.

The work closest in spirit to our work is that of Koetter and Kschischang [19], who also studied the presence of Byzantine adversaries in the distributed network coding setting. They concentrate on communicating against an omniscient adversary, and present a distributed scheme of optimal rate $C - 2z_O$. The proof techniques of [19] differ substantially from those presented in this work. In a nutshell, Koetter and Kschischang reduce the model of network coding to a certain point-to-point channel. They then construct generalizations of Reed–Solomon codes for this channel, which enables the authors to construct deterministic network error-correcting codes as mentioned above.

We would like to note that the abstraction used in [19] (although very elegant) comes at a price. It does not encapsulate the additional Byzantine scenarios that arise naturally in practice and are addressed in our current paper (i.e., adversaries of limited knowledge, discussed in Sections VI and VIII). More specifically, our protocol enables us to attain the higher rate of $C - z_O$, albeit only under the (weaker) requirement of list decoding. List decoding in the setting of network communication is a central ingredient in our proofs for limited adversaries. To the best of our current knowledge, the abstraction of [19] (although based on Reed–Solomon like codes) does not allow efficient list decoding.

III. Model and Definitions

We use a general model that encompasses both wired and wireless networks. To simplify notation, we consider only the problem of communicating from a single source to a single destination. But similarly to most network coding algorithms, our techniques generalize to multicast traffic.

A. Threat Model

There is a source, Alice, who communicates over a wired or wireless network to a receiver Bob. There is also an attacker Calvin, hidden somewhere in the network. Calvin aims to prevent the transfer of information from Alice to Bob, or at least to minimize it. He can observe some or all of the transmissions, and can inject his own. When he injects his own data, he pretends they are part of the information flow from Alice to Bob.

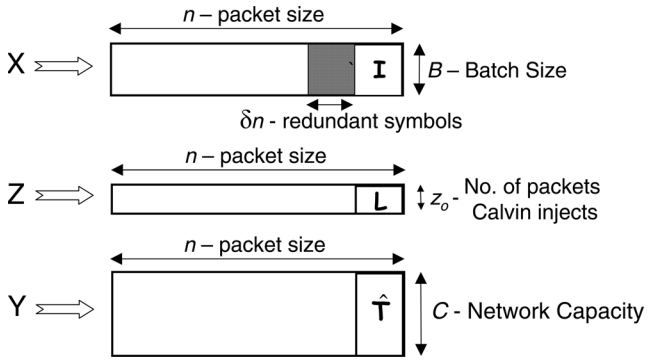

Fig. 1. Alice, Bob, and Calvin's information matrices.

Calvin is quite strong. He is computationally unbounded. He knows the encoding and decoding schemes of Alice and Bob, and the network code implemented by the interior nodes. He also knows the exact network realization.

B. Network and Code Model

Network Model: The network is modeled as a hypergraph [24]. Each transmission carries a packet of data over a hyperedge directed from the transmitting node to the set of observer nodes. The hypergraph model captures both wired and wireless networks. For wired networks, the hyperedge is a simple point-to-point link. For wireless networks, each such hyperedge is determined by instantaneous channel realizations (packets may be lost due to fading or collisions) and connects the transmitter to all nodes that hear the transmission. The hypergraph is unknown to Alice and Bob prior to transmission.

Source: Alice generates incompressible data that she wishes to deliver to Bob over the network. To do so, Alice encodes her data as dictated by the encoding algorithm (described in subsequent sections). She divides the encoded data into batches of b packets. For clarity, we focus on the encoding and decoding of one batch.

A packet contains a sequence of n symbols from the finite field $\mathbb{F}_q$. All arithmetic operations henceforth are done over symbols from $\mathbb{F}_q$. (See the treatment in [20].) Out of the n symbols in Alice's packet, δn symbols are redundancy added by the source.

Alice organizes the data in each batch into a matrix X as shown in Fig. 1. We denote the (i,j)th element in the matrix by $x(i,j)$. The ith row in the matrix X is just the ith packet in the batch. Fig. 1 shows that similarly to standard network codes [10], some of the redundancy in the batch is devoted to sending the identity matrix I. Also, as in [10], Alice takes random linear combinations of the rows of X to generate her transmitted packets. As the packets traverse the network, the internal nodes apply a linear transform to the batch. The identity matrix receives the same linear transform. The destination discovers the linear relation, denoted by the matrix T, between the packets it receives and those transmitted. This is done by inspecting how I was transformed.

Adversary: Let the matrix Z be the information Calvin injects into each batch. The size of this matrix is $z_O \times n$, where z_O is the number of edges controlled by Calvin (alternatively, one may define z_O to be the size of the min-cut from Calvin to the destination). In some of our adversarial models we limit the eavesdropping capabilities of Calvin. Namely, we limit the number of transmitted packets Calvin can observe. In such cases, this number will be denoted by z_I.

Receiver: Analogously to how Alice generates X, the receiver Bob organizes the received packets into a matrix Y. The ith received packet corresponds to the ith row of Y. Note that the number of received packets, and therefore the number of rows of Y, is a variable dependent on the network topology. Bob attempts to reconstruct Alice's information X, using the matrix of received packets Y.

As mentioned in the Introduction, conceptually, Bob recovers the information of Alice in two steps. First, Bob identifies a set of linear constraints which must be satisfied by the transmitted information X of Alice. This set of constraints characterizes a linear subspace of *low* dimension in which X must lie. We refer to this low-dimensional subspace as a linear list decoding of X. Once list decoding is accomplished, unique decoding follows by considering additional information Bob has on the matrix X (such as its redundancy, or information transmitted by Alice over a low rate secret channel).

Network Transform: The network performs a classical distributed network code [10]. Specifically, each packet transmitted by an internal node is a random linear combination of its incoming packets. Thus, the effect of the network at the destination can be summarized as follows:

$$Y = TX + T'Z. \tag{1}$$

This can be written as

$$Y = [T|T'] \begin{bmatrix} X \\ Z \end{bmatrix} \tag{2}$$

where X is the batch of packets sent by Alice, Z refers to the packets Calvin injects into Alice's batch, and Y is the received batch. The matrix T refers to the linear transform from Alice to Bob, while T' refers to the linear transform from Calvin to Bob. Notice that neither T nor T' are known to Bob. Rather, as shown in Fig. 1, Bob receives the matrix $\hat{T}$, which cannot be directly used to recover X.

Notice that in our model the error imposed by the Byzantine adversary Calvin is assumed to be *added* to the original information transmitted on the network. One can also consider a model in which these errors *overwrite* the existing information transmitted by Alice. We stress that if Calvin is aware of transmissions on links, these two models are equivalent. Overwriting a message with Z is equivalent to adding $-X_Z + Z$ on the links controlled by Calvin, where X_Z represents the original transmissions on those links.

Definitions: Table I lists notation needed for our main results. We define the following concepts.

- The *network capacity*, denoted by C, is the time average of the maximum number of packets that can be delivered from Alice to Bob, assuming no adversarial interference, i.e., the max flow. It can be also expressed as the min-cut from source to destination. (For the corresponding multicast case, C is defined as the minimum of the min-cuts over all destinations.)

TABLE I
TERMS USED IN THE PAPER

Variable	Definition
C	Network capacity.
z_O	Number of packets Calvin can inject.
z_I	Number of packets Calvin can hear.
b	Number of packets in a batch[a].
n	Length of each packet.
δ	Alice's redundancy.

[a]Throughout this work b is defined as $C - z_O$.

- The *error probability* is the probability that Bob's reconstruction of Alice's information is inaccurate.
- The rate R is the number of *information* symbols that can be delivered on average, per time step, from Alice to Bob. Rate R is said to be achievable if for any $\epsilon_1 > 0$ and $\epsilon_2 > 0$ there exists a coding scheme of block length n with rate $\geq R - \epsilon_2$ and error probability $\leq \epsilon_1$.

IV. SUMMARY OF RESULTS

We have three main results. Each result corresponds to a distributed, rate-optimal, polynomial-time algorithm that defeats an adversary of a particular type. The optimality of these rates has been proven by prior work [2], [3], [29], [14]. Our work, however, provides a construction of distributed codes/algorithms that achieve optimal rates. To prove our results, we first study the scenario of high rate list decoding in the presence of Byzantine adversaries. In what follows, let $|\mathcal{T}|$ denote the number of receivers, and $|\mathcal{E}|$ denote the number of (hyper)-edges in the network.

A. Shared Secret Model

This model considers the transmission of information via network coding in a network where Calvin can observe all transmissions, and can inject z_O corrupt packets. However, it is assumed that Alice can transmit to Bob a message (at asymptotically negligible rate) which is unknown to Calvin over a separate secret channel. In Section VI, we prove the following.

Theorem 1: The Shared Secret algorithm achieves an optimal rate of $C - z_O$ with code-complexity $\mathcal{O}(nC^3)$.

B. Omniscient Adversary Model

This model assumes an omniscient adversary, i.e., one from whom nothing is hidden. As in the Shared Secret model, Calvin can observe all transmissions, and can inject z_O corrupt packets. However, Alice and Bob have no shared secrets hidden from Calvin. In Section VII, we prove the following.

Theorem 2: The Omniscient Adversary algorithm achieves an optimal rate of $C - 2z_O$ with code-complexity $\mathcal{O}((nC)^3)$.

C. Limited Adversary Model

In this model, Calvin is limited in his eavesdropping power; he can observe at most z_I transmitted packets. Exploiting this weakness of the adversary results in an algorithm that, like the Omniscient Adversary algorithm, operates without a shared secret. In Section VIII, we prove the following.

Theorem 3: If $z_I < C - 2z_O$, the Limited Adversary algorithm achieves an optimal rate of $C - z_O$ with code-complexity $\mathcal{O}(nC^3)$.

D. Linear List Decoding Model

A key building block in some of our proofs is a linear list decoding algorithm. The model assumes the Omniscient Adversary of Section IV-B. We design a code that Bob can use to output a *linear* list (of *low* dimension) that is guaranteed to contain Alice's message X. The list is then refined to obtain the results stated in Theorems 1–3. In Section V we prove the following.

Theorem 4: The Linear List Decoding algorithm achieves a rate of $C - z_O$ and outputs a list L that is guaranteed to contain X. The list L is a vector space of dimension $b(b + z_O)$. The code-complexity is $\mathcal{O}(nC^3)$.

V. LINEAR LIST DECODING IN THE OMNISCIENT ADVERSARY MODEL

Here we assume we face an omniscient adversary, i.e., Calvin can observe everything, and there are no shared secrets between Alice and Bob. We design a code that Bob can use in this scenario to output a linear list (of low dimension) that is guaranteed to contain Alice's message X. Our algorithm achieves a rate of $R = C - z_O$. The corrupted information Y Bob receives enables him to deduce a system of linear equations that X satisfies. This system of equations ensures that X lies in a low-dimensional vector space. We now present our algorithm in detail. Throughout this and upcoming sections, b is fixed as $C - z_O$.

A. Alice's Encoder

Alice's encoder is quite straightforward. She simply arranges the source symbols into the $b \times n$ matrix X, appended with a b-dimensional identity matrix. She then implements the classical random network encoder described in Section III-B to generate her transmitted packets.

B. Bob's Decoder

Bob selects $b + z_O$ linearly independent columns of Y, and denotes the corresponding matrix Y^s. Here we assume, without loss of generality (w.l.o.g.), that the column rank of Y is indeed $b + z_O$. The column rank cannot be larger than $b + z_O$ by (2). If the column rank happens to be $r < b + z_O$, Bob selects r independent rows of Y and continues in a procedure analogous to that described below. We also assume that Y^s contains the last b columns of Y (corresponding to Alice's b-dimensional identity matrix). This is justified due to (2) and the assumption (discussed below) that the intersection of the column spans of T and T' is trivial, i.e., $[T|T']$ is regular (with high probability over the random choices of internal nodes in the network). The remaining z_O columns of Y^s are chosen arbitrarily so that Y^s is invertible. The columns of X and Z corresponding to those in Y^s are denoted X^s and Z^s, respectively. By (2),

$$Y^s = [T|T'] \begin{bmatrix} X^s \\ Z^s \end{bmatrix}.$$

Also, since Y^s acts as a basis for the columns of Y, we can write $Y = Y^s F$ for some matrix F. Bob can compute F as $(Y^s)^{-1} Y$. Therefore, Y can also be written as

$$Y = [T|T'] \begin{bmatrix} X^s F \\ Z^s F \end{bmatrix}. \tag{3}$$

Comparing (2) and (3), and again using the assumption that $[T|T']$ is invertible (with high probability) gives us

$$X = X^s F \tag{4}$$

$$Z = Z^s F. \tag{5}$$

In particular, (4) gives a linear relationship on X that can be leveraged into a list-decoding scheme for Bob (the corresponding linear relationship from (5) is not very useful). The number of variables in X^s is $b(b + z_O)$. Therefore, the entries of the matrix X^s span a vector space of dimension $b(b + z_O)$ over $\mathbb{F}_q$. Bob's list is the corresponding $b(b+z_O)$-dimensional vector space $\boldsymbol{L}$ spanned by $X^s F$.

The only source of error in our argument arises if the intersection of the column-spans of T and T' is nontrivial, i.e., if $[T|T']$ is singular. But as shown in [11], as long as $b+z_O \leq C$, this is at most $|\mathcal{T}||\mathcal{E}|q^{-1}$ for any fixed network. Since Calvin can choose his locations in at most $\binom{|\mathcal{E}|}{z_O}$ ways, the total probability of error is at most $\binom{|\mathcal{E}|}{z_O}|\mathcal{T}||\mathcal{E}|q^{-1}$. The computational cost of design, encoding and decoding is dominated by the cost of computing F and thereby a representation of L. This takes $\mathcal{O}(nC^3)$ steps.

Note: In the Linear List Decoding scheme described above, Alice appends an identity matrix to her source symbols to obtain the matrix X, causing (an asymptotically negligible) loss in rate. This is also the standard protocol of [10]. We note that our scheme works just as well even if Alice does not append such an identity matrix, and X consists solely of source symbols. However, the appended identity matrix is used in the model of Section VII. We now solve (4) under different assumptions on Calvin's strength.

VI. Shared Secret Model

In the Shared Secret model Alice and Bob have use of a strong resource, namely, a secret channel over which Alice can transmit a small amount of information to Bob that is secret from Calvin. The size of this secret is asymptotically negligible in n. Note that since the internal nodes mix corrupted and uncorrupted packets, Alice cannot just sign her packets and have Bob check the signature and throw away corrupted packets—in extreme cases, Bob may not receive *any* uncorrupted packets.

Alice uses the secret channel to send a random hash of her data to Bob. Bob first uses the list-decoding scheme of Section V to obtain a low-dimensional vector space $\boldsymbol{L}$ containing X. He then uses Alice's hash to identify X from $\boldsymbol{L}$.

Let α be a parameter defined below. Let $r_1, \ldots, r_\alpha$ be α elements of $\mathbb{F}_q$ chosen at random by Alice (and unknown to Calvin). Let $D = [d_{ij}]$ be an $n \times \alpha$ matrix in which $d_{ij} = (r_j)^i$. Let $XD = H$. Alice sends to Bob a secret \$ comprising of the symbols $r_1, \ldots, r_\alpha$ and the matrix H. The size of this secret is thus $\alpha(\alpha + 1)$, which is asymptotically negligible in n.

Claim 5: For any $X' \neq X$ the probability (over $r_1, \ldots, r_\alpha$) that $X'D = H$ is at most $\left(\frac{n}{q}\right)^\alpha$.

Proof: We need to prove that $(X - X')D \neq \mathbf{0}$ with high probability, where $\mathbf{0}$ is the zero matrix. As $X \neq X'$ there is at least one row of X which differs from X'. Assume w.l.o.g. that this is the first row, denoted here as the nonzero vector $(x_1, \ldots, x_n)$. The jth entry in the first row of $(X - X')D$ is $F(r_j) = \sum_{i=1}^{n} x_i r_j^i$. As $F(r_j)$ is not the zero polynomial, the probability (over r_j) that $F(r_j) = 0$ is at most $\frac{n}{q}$. This holds for all entries of the first row of $(X - X')D$. Thus, the probability that the entire row is the zero vector is at most $\left(\frac{n}{q}\right)^\alpha$. □

Let $\alpha = b(b + z_O) + 1$. Let $\boldsymbol{L}$ be a list (containing X) of distinct matrices. Let the size of $\boldsymbol{L}$ be $q^{\alpha-1}$.

Corollary 6: The probability (over $r_1, \ldots, r_\alpha$) that there exists $X' \in \boldsymbol{L}$ such that $X' \neq X$ but $X'D = XD$ is at most n^α/q.

Proof: We use Claim 5, and the union bound on all elements of $\boldsymbol{L}$ that differ from X. □

Note: The secret channel is essential for the following reason. If the symbols $r_1, \ldots, r_\alpha$ were *not* secret from Calvin, he could carefully select his corrupted packets so that Bob's list $\boldsymbol{L}$ would indeed contain an $X' \neq X$ such that $X'D = XD$.

Bob is able to decode the original information X of Alice. Namely, Corollary 6 establishes that the system $XD = X^s FD = H$ has a single solution. This solution can be found using standard Gaussian elimination.

The above implies a scheme that achieves rate $C - z_O$. The optimality of this rate is shown in prior work [14]. The probability of error is at most $n^\alpha/q + |\mathcal{T}||\mathcal{E}|\binom{|\mathcal{E}|}{z_O}/q$. Here $\alpha = b(b+z_O)+1$. The computational cost of design, encoding, and decoding is dominated by the cost of running the Linear List Decoding algorithm, which takes time $\mathcal{O}(nC^3)$.

VII. Unique Decoding in the Omniscient Adversary Model

We now consider unique decoding. Our algorithm achieves a rate of $R = C - 2z_O$, which is lower than that possible in the list decoding scenario. Recent bounds [2], [3] on network error-correcting codes show that in fact $C-2z_O$ is the maximum achievable rate for networks with an omniscient adversary.

To move from list decoding to unique decoding in the omniscient model, we add redundancy to Alice's information as follows. Alice writes her information X in the form of a length-bn column vector $\vec{\boldsymbol{X}}$. The vector $\vec{\boldsymbol{X}}$ is chosen to satisfy $D\vec{\boldsymbol{X}} = 0$. Here, D is a $\delta n \times bn$ matrix defined as the *redundancy matrix*. The matrix D is obtained by choosing each element as an independent and uniformly random symbol from the finite field $\mathbb{F}_q$, and $\delta n > n(z_O + \varepsilon)$ for arbitrarily small ε. This choice of parameters implies that the number of *parity checks* $D\vec{\boldsymbol{X}} = 0$ is greater than the number of symbols in the z_O packets that Calvin injects into the network. We show that this allows Bob to uniquely decode, implying a rate of $C-2z_O$. The redundancy matrix D is known to all parties—Alice, Bob, and Calvin —and hence does not constitute a shared secret.

Alice encodes as in Section V. Bob's decoding is as follows.

Bob first runs the Linear List Decoding algorithm to obtain (4) and (5). We denote the matrix comprising of the first z_O rows of F by F_1, and the matrix comprising of the last b rows of F by F_2. By the constraints specified in Section V, the last b columns of X^s form an identity matrix. Thus, (4) transforms into

$$X = X_1^s F_1 + F_2 \tag{6}$$

where X_1^s comprises of the first z_O columns of X^s.

Recall that $\vec{\boldsymbol{X}}$ is a vector corresponding to the matrix X. Upon receiving Y, Bob computes F and solves the system

$$X = X_1^s F_1 + F_2 \tag{7}$$

$$D\vec{\mathbf{X}} = 0. \tag{8}$$

Here, only D and F are known to Bob. Our goal is now to show that with high probability over the entries of the matrix D, no matter which matrix F was obtained by Bob, there is a unique solution to (7) and (8). The matrix F depends on the errors Z Calvin injects. Calvin can choose these to depend on D. We take this into consideration below.

The system of linear equations (7)–(8) can be written in matrix form as

$$A\vec{\boldsymbol{X}} = \begin{bmatrix} A(F_1) \\ D \end{bmatrix} \vec{\boldsymbol{X}} = B$$

where A comprises of the submatrices $A(F_1)$ and D, $A(F_1)$ is a $bn \times bn$ matrix whose entries depend on F_1, and B is a length-$n(b+\delta)$ vector. It holds that the system (7)–(8) has a unique solution if and only if A has full column rank. However, Calvin has partial control over F, and his goal is to design his error Z so this will not be the case.

In what follows, we show that Calvin cannot succeed. Namely, we show, with high probability over the entries of D, that *no matter* what the value of F is, the system (7)–(8) has a unique solution. Our proof has the following structure. We first show that for a fixed F_1, the matrix A has full column rank with high probability over D. We then note that the number of possible different matrices F_1 is at most $q^{z_O n}$ (this follows from the size of F_1). Finally, applying the union bound we obtain our result.

We start with some notation. Assume that $\vec{\boldsymbol{X}}$ is arranged by stacking the columns of X one on top of the other, where the columns of X_2^s appear on the top of $\vec{\boldsymbol{X}}$. Also, we fix the (i,j)th entry of F_1 to be f_{ij}. Then, the matrix

$$A = \begin{bmatrix} A(F_1) \\ D \end{bmatrix}$$

has the following form:

$$\left[\begin{array}{cccc|c} (1-f_{1,1})I & -f_{2,1}I & \dots & -f_{z_O,1}I & \\ \vdots & \vdots & \vdots & \vdots & \mathbf{0} \\ -f_{1,z_O}I & -f_{2,z_O}I & \dots & (1-f_{z_O,z_O})I & \\ \hline -f_{1,z_O+1}I & -f_{2,z_O+1}I & \dots & -f_{z_O,z_O+1}I & \\ \vdots & \vdots & \vdots & \vdots & \boldsymbol{I} \\ -f_{1,n} & -f_{2,n}I & \dots & -f_{z_O,n}I & \\ \hline & & D & & \end{array}\right].$$

The matrix A is described by smaller dimensional matrices as entries. Namely, the identity matrices I appearing above have dimension b, the identity matrix $\boldsymbol{I}$ has dimension $b(n-z_O)$, and the zero matrix $\mathbf{0}$ has dimension $z_O b \times b(n-z_O)$. We now analyze the column rank of A.

Clearly, the last $b(n-z_O)$ columns of A are independent. Thus, any set of dependent columns of A must include at least one of the first bz_O columns. Let $V = \{u_1, \dots, u_{bz_O}; v_1, \dots, v_{b(n-z_O)}\}$ be the set of columns of A (here the $\{u_i\}$ vectors correspond to the leftmost bz_O columns of A). We break the $\{u_i\}$ and $\{v_j\}$ vectors into two parts. The components of the $\{u_i\}$ and $\{v_j\}$ vectors in the top bn rows of A are denoted, respectively, as $\{u_i^t\}$ and $\{v_j^t\}$. The components of the $\{u_i\}$ and $\{v_j\}$ vectors in the bottom δn rows of A are denoted, respectively, as $\{u_i^b\}$ and $\{v_j^b\}$. The matrix A is rank-deficient if and only if there exist $\{\alpha_i\}$ and $\{\beta_j\}$, not all zero, such that $\sum_i \alpha_i u_i + \sum_j \beta_j v_j = \mathbf{0}$. Note that there is a one-to-one correspondence between the values $\{\alpha_i\}$ and the values $\{\beta_j\}$ in the above equality. Namely, for each setting of $\{\alpha_i\}$, there is a unique setting of $\{\beta_j\}$ for which $\sum_i \alpha_i u_i^t + \sum_j \beta_j v_j^t = \mathbf{0}$. Further, for every setting of the values $\{\alpha_i\}$ (and a corresponding setting for $\{\beta_j\}$), the probability over D that $\sum_i \alpha_i u_i^b + \sum_j \beta_j v_j^b = \mathbf{0}$ is at most $q^{-\delta n}$. This implies that the probability $\sum_i \alpha_i u_i + \sum_j \beta_j v_j = \mathbf{0}$ is asymptotically negligible. Then, an additional use of the union bound on all q^{bz_O} possible values of $\{\alpha_i\}$ suffices to obtain our proof.

All in all, Bob fails to uniquely decode with probability $q^{z_O n} q^{bz_O} q^{-\delta n}$ (the first term corresponds to the union bound over the values of $F_1 = [f_{ij}]$, the second term corresponds to the union bound over the values of $\{\alpha_i\}$, and the third term corresponds to the failure probability). Setting $\delta = z_O + \varepsilon$ suffices for our proof. The computational cost of design, encoding, and decoding is dominated by solving the system of (7)–(8), and thus equals $\mathcal{O}((nC)^3)$.

VIII. Limited Adversary Model

In this section, we combine the strengths of the Shared Secret and the Omniscient Adversary algorithms of Sections VI and VII, respectively. We then achieve the higher rate of $C - z_O$ without the need of a secret channel. The caveat is that Calvin is more limited—he can only eavesdrop on part of the edges in the network. Specifically, the number of packets he can transmit, z_O, and the number he can eavesdrop on, z_I, satisfy the technical constraint

$$2z_O + z_I < C. \tag{9}$$

We call such an adversary a *Limited Adversary*.

The main idea underlying our Limited Adversary algorithm is simple. Alice uses the Omniscient Adversary algorithm to transmit a "short, scrambled" message to Bob at rate $C - 2z_O$. By (9), the rate z_I at which Calvin eavesdrops is strictly less than Alice's rate of transmission $C - 2z_O$. Hence, Calvin cannot decode Alice's message, but Bob can. This means Alice's scrambled message to Bob contains a secret $\mathbb{S}$ that is unknown to Calvin. Once $\mathbb{S}$ has been shared from Alice to Bob, they can

use the Shared Secret algorithm to transmit the bulk of Alice's message to Bob at the higher rate $C - z_O$.

A. Alice's Encoder

Alice's encoder follows essentially the schema described in the previous paragraph. The information $\mathbb{S}$ she transmits to Bob via the Omniscient Adversary algorithm is padded with some random symbols. This is for two reasons. First, the randomness in the padded symbols ensures strong information-theoretic secrecy of $\mathbb{S}$. That is, we show in Claim 7 that Calvin's best estimate of *any function* of $\mathbb{S}$ is no better than if he randomly guessed the value of the function. Second, since the Omniscient Adversary algorithm has a probability of error that decays exponentially with the size of the input, it is not guaranteed to perform well when only a small message is transmitted.

Alice divides her information X into two parts $[X_1\, X_2]$. She uses the information she wishes to transmit to Bob (at rate $R = (C - z_O)(1 - \Delta)$) as the input to the encoder of the Shared Secret algorithm. The output of this step is the $b \times n(1 - \Delta)$ submatrix X_1. Here Δ is a parameter that enables Alice to trade between the probability of error and rate loss.

The second submatrix X_2, which we call the *secrecy matrix*, is analogous to the secret $\mathbb{S}$ used in the Secret Sharing algorithm described in Section VI. The size of X_2 is $b \times n\Delta$. In fact, X_2 is an encoding of the secret $\mathbb{S}$ Alice generates in the Shared Secret algorithm. The $\gamma = (b(b + z_O) + 1)(b + 1)$ symbols corresponding to the parity symbols $\{r_j\}$ and the hash matrix H are written in the form of a length-γ column vector. This vector is appended with symbols chosen uniformly at random from $\mathbb{F}_q$ to result in the length-$(C - z_O - \delta)n\Delta$ vector $\vec{\boldsymbol{U}}'$. Alice multiplies $\vec{\boldsymbol{U}}'$ by a random square matrix to generate the input $\vec{\boldsymbol{U}}$. This vector $\vec{\boldsymbol{U}}$ functions as the input to the Omniscient Adversary algorithm operated over a packet-size $n\Delta$ with a probability of decoding error that is exponentially small in $n\Delta$. The output of this step is X_2.

The following claim ensures that $\mathbb{S}$ is indeed secret from Calvin.

Claim 7: Let $\gamma = (b(b+z_O)+1)(b+1)$. The probability that Calvin guesses $\mathbb{S}$ correctly is at most $q^{-\gamma}$, i.e., $\mathbb{S}$ is information-theoretically secret from Calvin.

The proof of Claim 7 follows from a direct extension of the secure communication scheme of [6] to our scenario.

The two components of X, i.e., X_1 and X_2, respectively, correspond to the information Alice wishes to transmit to Bob, and an implementation of the low-rate secret channel. The fraction of the packet size corresponding to X_2 is "small," i.e., Δ. Finally, Alice implements the classical random encoder described in Section III-B.

B. Bob's Decoder

Bob arranges his received packets into the matrix $Y = [Y_1\, Y_2]$. The submatrices Y_1 and Y_2 are, respectively, the network transforms of X_1 and X_2.

Bob decodes in two steps. Bob first recovers $\mathbb{S}$ by decoding Y_2 as follows. He begins by using the Omniscient Adversary decoder to obtain the vector $\vec{\boldsymbol{U}}$. He then obtains $\vec{\boldsymbol{U}}'$ from $\vec{\boldsymbol{U}}$, by inverting the mapping specified in Alice's encoder. He finally extracts from $\vec{\boldsymbol{U}}'$ the γ symbols corresponding to $\mathbb{S}$.

Alice has now shared $\mathbb{S}$ with Bob. Bob uses $\mathbb{S}$ as the side information used by the decoder of the Shared Secret algorithm to decode Y_1. This enables him to recover X_1, which contains Alice's information at rate $R = C - z_O$. The probability of error is dominated by the sums of the probabilities of error in Theorems 1 and 2, with the parameter n replaced by $n\Delta$. The Limited Adversary algorithm is essentially a concatenation of the Shared Secret algorithm with the Omniscient Adversary algorithm, thus, the computational cost is dominated by the sum of the two (with $n\Delta$ replacing n). Choosing Δ appropriately (say $n\Delta = n^{1/3}$), one may bound the complexity by $O(nC^3)$.

TABLE II
COMPARISON OF OUR THREE ALGORITHMS

	Adversarial Strength	Rate	Complexity
Shared Secret	$z_O < C$, z_I = network	$C - z_O$	$\mathcal{O}(nC^3)$
Omniscient	$z_O < C/2$, z_I = network	$C - 2z_O$	$\mathcal{O}((nC)^3)$
Limited	$z_I + 2z_O < C$	$C - z_O$	$\mathcal{O}(nC^3)$

IX. Conclusion

Random network codes are vulnerable to Byzantine adversaries. This work makes them secure. We provide algorithms[2] which are information-theoretically secure and rate-optimal for different adversarial strengths (as shown in Table II). When the adversary is omniscient, we show how to achieve a rate of $C - 2z_O$, where z_O is the number of packets the adversary injects and C is the network capacity. If the adversary cannot observe everything, our algorithms achieve a higher rate, $C - z_O$. Both rates are optimal. Further, our algorithms are practical; they are distributed, have polynomial-time complexity, and require no changes at the internal nodes.

References

[1] R. Ahlswede, N. Cai, S. R. Li, and R. W. Yeung, "Network information flow," *IEEE Trans. Inf. Theory*, vol. 46, no. 5, pp. 1204–1216, Jul. 2000.

[2] N. Cai and R. W. Yeung, "Secure network coding," in *Proc. IEEE Int. Symp. Information Theory*, Lausanne, Switzerland, Jun./Jul. 2002, p. 323.

[3] N. Cai and R. W. Yeung, "Network error correction, Part 2: Lower bounds," *Commun. Inf. and Syst.*, vol. 6, no. 1, pp. 37–54, Jan. 2006.

[4] D. Charles, K. Jain, and K. Lauter, "Signatures for network coding," in *Proc. 40th Annu. Conf. Information Science and Systems*, Princeton, NJ, Mar. 2006.

[5] D. Dolev, C. Dwork, O. Waarts, and M. Yung, "Perfectly secure message transmission," *J. Assoc. Comput. Mach.*, vol. 40, no. 1, pp. 17–47, Jan. 1993.

[6] J. Feldman, T. Malkin, C. Stein, and R. A. Servedio, "On the capacity of secure network coding," in *Proc. 42nd Annu. Allerton Conf. Communication, Control, and Computing*, Monticello, IL, Sep. 2004.

[7] C. Fragouli and E. Soljanin, *Network Coding Fundamentals*. PLEASE CITE LOCATION OF PUBLISHER.: Now, 2007.

[8] C. Gkantsidis and P. Rodriguez, "Network coding for large scale content distribution," in *Proc. IEEE Conf. Computer Communications (INFOCOM)*, Miami, FL, Mar. 2005.

[2]A refinement of some of the algorithms in this work can be found in [13].

[9] C. Gkantsidis and P. Rodriguez, "Cooperative security for network coding file distribution," in *Proc. IEEE Conf. Computer Communications (INFOCOM)*, Barcelona, Spain, Apr. 2006.

[10] T. Ho, R. Koetter, M. Médard, D. Karger, and M. Effros, "The benefits of coding over routing in a randomized setting," in *Proc. IEEE Int. Symp. Information Theory*, Yokohama, Japan, Jun./Jul. 2003, p. 442.

[11] T. Ho, M. Médard, J. Shi, M. Effros, and D. Karger, "On randomized network coding," in *Proc. 41st Annu. Allerton Conf. Communication, Control, and Computing*, Monticello, IL, Oct. 2003.

[12] T. C. Ho, B. Leong, R. Koetter, M. Médard, M. Effros, and D. R. Karger, "Byzantine modification detection in multicast networks using randomized network coding," in *Proc. IEEE Int. Symp. Information Theory*, Chicago, IL, Jun. 2004, p. 144.

[13] S. Jaggi and M. Langberg, "Resilient network coding in the presence of eavesdropping byzantine adversaries," in *Proc. IEEE Int. Symp. Information Theory*, Nice, France, Jun. 2007, pp. 541–545.

[14] S. Jaggi, M. Langberg, T. Ho, and M. Effros, "Correction of adversarial errors in networks," in *Proc. IEEE Int. Symp. Information Theory*, Adelaide, Australia, Sep. 2005, pp. 1455–1459.

[15] S. Jaggi, P. Sanders, P. A. Chou, M. Effros, S. Egner, K. Jain, and L. Tolhuizen, "Polynomial time algorithms for multicast network code construction," *IEEE Trans. Inf. Theory*, vol. 51, no. 6, pp. 1973–1982, Jun. 2005.

[16] A. Jiang, "Network coding for joint storage and transmission with minimum cost," in *Proc. IEEE Int. Symp. Information Theory*, Seattle, WA, Jul. 2006, pp. 1359–1363.

[17] S. Katti, D. Katabi, W. Hu, H. S. Rahul, and M. Médard, "The importance of being opportunistic: Practical network coding for wireless environments," in *Proc. 43rd Annu. Allerton Conf. Communication, Control, and Computing*, Monticello, IL, Sep. 2005.

[18] S. Katti, H. Rahul, D. Katabi, W. H. M. Médard, and J. Crowcroft, "XORs in the air: Practical wireless network coding," in *Proc. ACM SIGCOMM*, Pisa, Italy, Sep. 2006.

[19] R. Koetter and F. Kschischang, "Coding for errors and erasures in random network coding," in *Proc. IEEE Int. Symp. Information Theory*, Nice, France, June 2007, pp. 791–795.

[20] R. Koetter and M. Médard, "Beyond routing: An algebraic approach to network coding," in *Proc. 21st Annu. Joint Conf. e IEEE Computer and Communications Societies (INFOCOM)*, 2002, vol. 1, pp. 122–130.

[21] R. Koetter and M. Médard, "An algebraic approach to network coding," *IEEE/ACM Trans. Netw.*, vol. 11, no. 5, pp. 782–795, Oct. 2003.

[22] M. N. Krohn, M. J. Freedman, and D. Mazires, "On-the-fly verification of rateless erasure codes for efficient content distribution," in *Proc. IEEE Symp. Security and Privacy*, Oakland, CA, May 2004.

[23] S.-Y. R. Li, R. W. Yeung, and N. Cai, "Linear network coding," *IEEE Trans. Inf. Theory*, vol. 49, no. 2, pp. 371–381, Feb. 2003.

[24] D. Lun, M. Médard, T. Ho, and R. Koetter, "On network coding with a cost criterion," in *Proc. IEEE Int. Symp. Information Theory and Its Applications*, Parma, Italy, Oct. 2004.

[25] D. S. Lun, M. Médard, and R. Koetter, "Efficient operation of wireless packet networks using network coding," in *Proc. Int. Workshop on Convergent Technologies (IWCT)*, Oulu, Finland, Jun. 2005.

[26] A. Pelc and D. Peleg, "Broadcasting with locally bounded byzantine faults," *Inf. Process. Lett.*, vol. 93, no. 3, pp. 109–115, Feb. 2005.

[27] L. Subramanian, "Decentralized Security Mechanisms for Routing Protocols," Ph.D. dissertation, Univ. Calif. Berkeley, Computer Science Division, Berkeley, CA, 2005.

[28] J. E. Wieselthier, G. D. Nguyen, and A. Ephremides, "On the construction of energy-efficient broadcast and multicast trees in wireless networks," in *Proc. IEEE Infocom*, 2000, vol. 2, pp. 585–594.

[29] R. W. Yeung and N. Cai, "Network error correction, part 1: Basic concepts and upper bounds," *Commun. Inf. and Syst.*, vol. 6, no. 1, pp. 19–36, Jan. 2006.

[30] R. W. Yeung, S. Li, N. Cal, and Z. Zhang, *Network Coding Theory*. Amsterdam, The Netherlands: Now, 2006.

[31] Z. Zhang, "Linear network error correction codes in packet networks," *IEEE Trans. Inf. Theory*, vol. 54, no. 1, pp. 209–218, Jan. 2008.

[32] F. Zhao, T. Kalker, M. Médard, and K. J. Han, "Signatures for content distribution with network coding," in *Proc. IEEE Int. Symp. Information Theory*, Nice, France, Jun. 2007, pp. 556–560.