REPORT DOCUMENTATION PAGE

Form Approved
OMB No. 0704-0188

1. REPORT DATE *(DD-MM-YYYY)*	2. REPORT TYPE	3. DATES COVERED *(From - To)*
	FINAL REPORT	15 Feb 06 – 30 Nov 08

4. TITLE AND SUBTITLE
Robust Self-Authenticating Network Coding

5a. CONTRACT NUMBER

5b. GRANT NUMBER
FA9550-06-1-0155

5c. PROGRAM ELEMENT NUMBER
61102F

6. AUTHOR(S)
Prof Muriel Medard

5d. PROJECT NUMBER
2311

5e. TASK NUMBER
FX

5f. WORK UNIT NUMBER

7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES)
UNIVERSITY OF ILLINOIS
Massachusetts Institute of Technology
Electrical Engineering and Computer Science Department
77 Massachusetts Ave
Cambridge, MA 02139-4307

8. PERFORMING ORGANIZATION REPORT NUMBER

9. SPONSORING/MONITORING AGENCY NAME(S) AND ADDRESS(ES)
AFOSR.RSL
875 NORTH RANDOLPH ST, ROOM 3112
ARLINGTON, VA 22203-1678

10. SPONSOR/MONITOR'S ACRONYM(S)

11. SPONSOR/MONITOR'S REPORT NUMBER(S)

12. DISTRIBUTION/AVAILABILITY STATEMENT
DISTRIBUTION A

AFRL-OSR-VA-TR-2012- 0035

13. SUPPLEMENTARY NOTES

14. ABSTRACT
The annual accomplishments include new algorithms that use network coding for: data hiding without the use of a key – ensuring sufficient degrees of freedom to decode over the receiver in variable settings, creating efficient coding schemes for Byzantine attacks, and providing quantification of the benefits to network coding..

15. SUBJECT TERMS

16. SECURITY CLASSIFICATION OF:			17. LIMITATION OF ABSTRACT	18. NUMBER OF PAGES	19a. NAME OF RESPONSIBLE PERSON
a. REPORT	b. ABSTRACT	c. THIS PAGE			19b. TELEPHONE NUMBER *(Include area code)*

Network Coding for Network Security

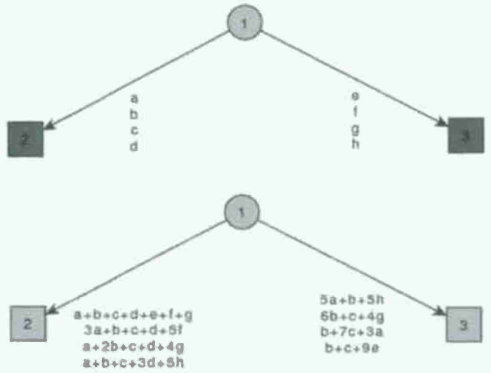
Keesook Han AFRL
Tracey Ho Caltech
Ralf Koetter Technical University Munich
Muriel Medard MIT

Network Coding for Network Security

Objective

Use network coding to enable greater robustness and security

- Reduce vulnerability eavesdroppers in networks
- Provide reliability to Byzantine nodes in changing conditions
- Provide constructive means of creating schemes that are as efficient as traditional point-to-point coding schemes



Number of symbols that an intermediate node has to guess in order to decode one of the symbols

Scientific/Technical Approach

- Use the algebraic linear mixing of data to allow intrinsic keys from other data by considering the diagonalizability of matrices
- Since robustness using network coding depends on having sufficient degrees of freedom to counteract attackers over the entire network, we develop means of tracking topology in changing P2P networks
- Use network coding for constructing codes that match Singleton bound even with unknown attackers

Accomplishments

- New algorithms that use network coding for:
  - data hiding without the use of a key – ensuring sufficient degrees of freedom to decode over at the receiver in variable settings
  - creating efficient coding schemes for Byzantine attacks
  - providing quantification of the benefits of network coding

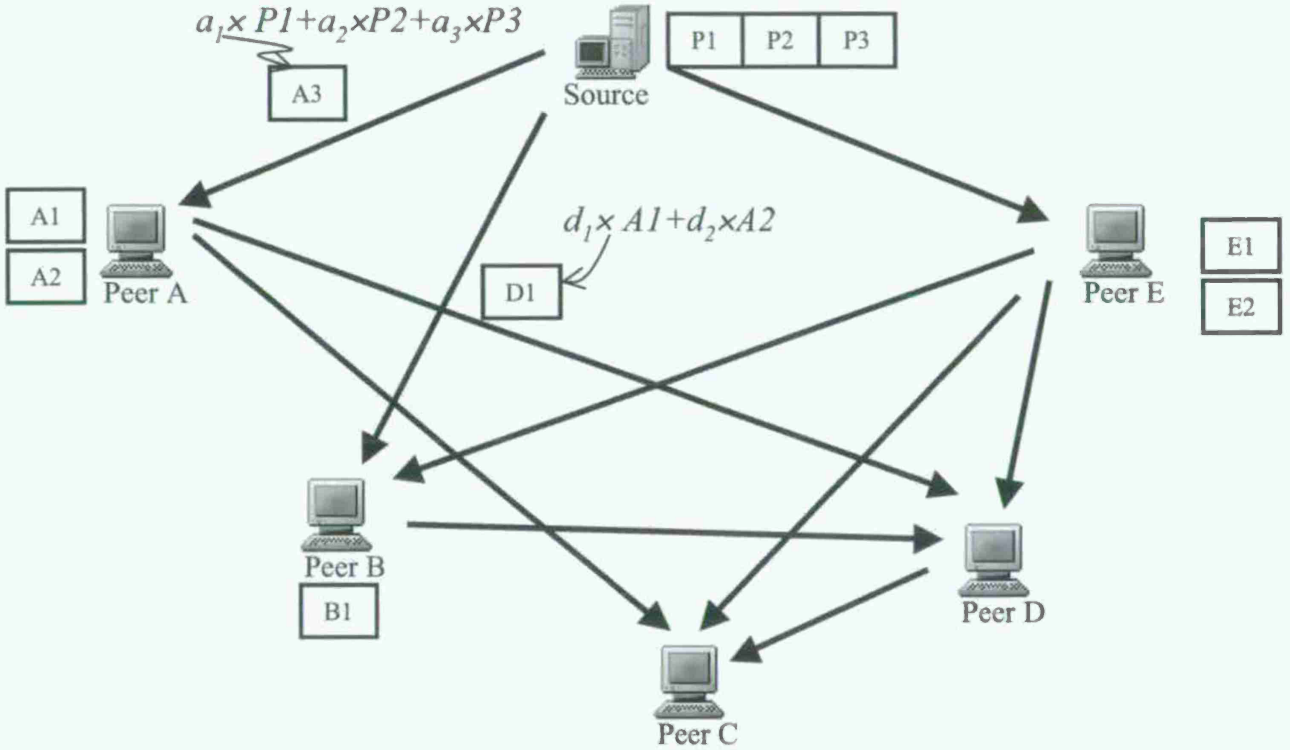
Challenges

- Integrating protection, degree of freedom design and coding .

Key Accomplishments

- Technical breakthroughs:
  - Demonstrated the use of network coding to provide intrinsic cryptographic protection for wiretapped networks
  - Provided new means of using network coding for networks under attack:
    - For distributed network coded storage networks (peer-to-peer), a method for tracking the evolving topology of a peer-to-peer network so as to ensure sufficient coded diversity against attackers
    - For general networks, a robust coding approach that matches the Singleton bound even under attack scenarios for unknown attack locations as long as a level of diversity is ensured
    - We show that random network coding provides better reliability than random dispersive routing if there is enough capacity in the network
- The support of AFOSR in this context is crucial:
  - Only program to our knowledge that considers security of network coding in wireline systems, including P2P
  - Deployment of network coded P2P systems is taking place commercially (Microsoft) and holds great promise for military applications,
  - Collaboration with industry (HP) for technology transfer and synergies with DARPA IAMANET program (which is focused entirely on MANETs but can leverage some aspects of this program), collaborations with general theoretical underpinnings for network coding through NSF program and European program

Content distribution using network coding



- Network coding operates by allowing mixing of data
- What are the security consequences of such mixtures?
- Two aspects:
  - Wiretapping aspects
  - Byzantine or pollution attacks – detection and correction A malicious user sends packets with valid linear combination in header, but garbage payload

Wiretapping aspects

- The mixture of two messages, appropriately compressed, makes one message a one-time pad to another [CY02]
- If we want such mixtures, one can derive limits on network capacity [FMSS04]
- In general
  - Difficult to know the maximum number of links that can be tapped by adversary
  - Such secure coding schemes are sometimes impossible
- Main scheme:
  - Use other messages for "encryption"
  - If no other messages are sent: identical to dispersive routing [JM04, LLF04]
  - If other messages are sent: extra security from network coding
- Security can be added via network coding with lower cost than dispersive routing [TM06]
- Define level of security provided by random linear network coding is measured by the number of symbols that an intermediate node has to guess in order to decode one of the transmitted symbols [LMB07], [LVMB08]

Random linear coding – a free cypher?

Overview:

- Random linear coding (RLC) in effect provides a one-time use pad use of data in combination
- Level of security provided by RLC:
  - Number of symbols that an intermediate node *v* has to guess in order to decode one of the transmitted symbols
  - Partial transfer matrix
- We consider these results under different topologies

x	0	1	2
0	0	0	0
1	0	1	2
2	0	2	1

Model, definitions, approach and results;

- Two cases w/ relevant information:
  1. Partial transfer matrix has full rank
  2. Partial transfer matrix has diagonizable parts

$$P(X_{lin} = 0) \leq \frac{2}{q}$$

- Linear combination of independent and uniformly distributed values in F_q

$$P(X_{lin} = 0)_{q \to \infty} = 0$$

- Product - Obtain a zero: $2q$ zeros, $(a \in F_q) \times 0$
- q^2 entries of the multiplicative table
- Probability p of having $\geq K-1$ zeros in one or more lines

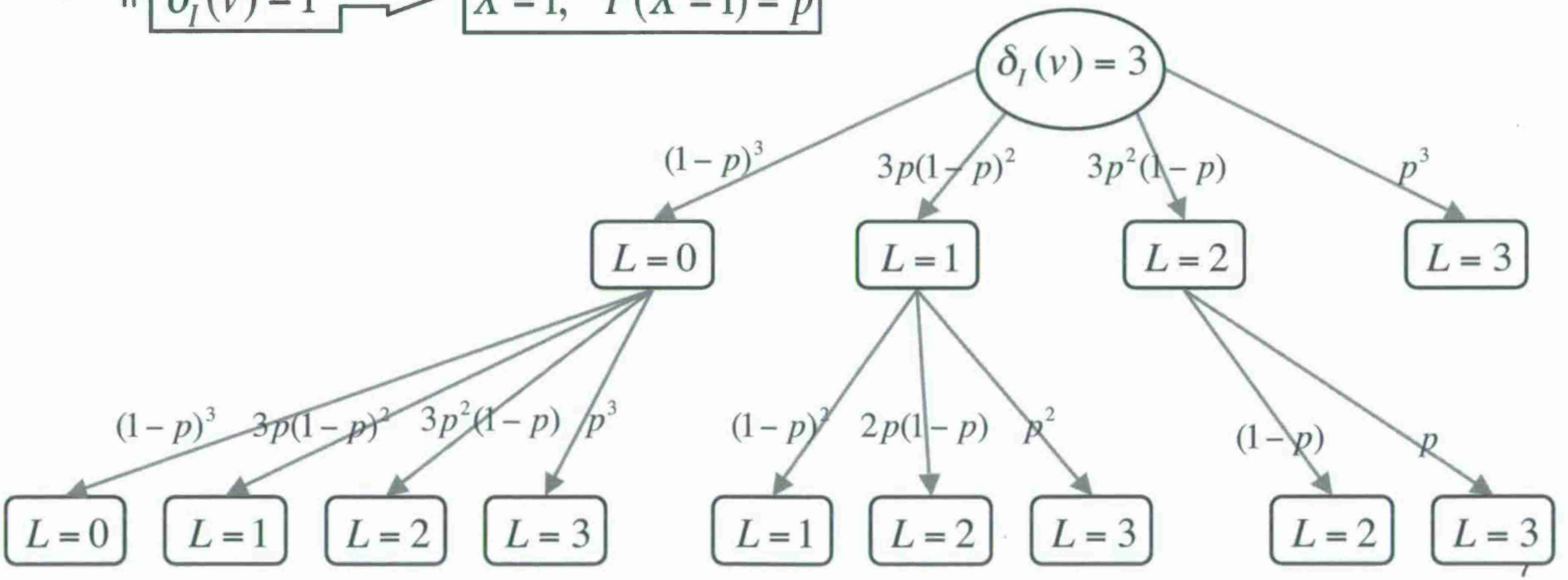
$$p = \binom{K}{K-1}\left(\frac{2}{q}\right)\left(1-\frac{2}{q}\right)^{K-1}$$

Random linear coding – a free cypher?

- Analysing the different possibilities of combinations for the lines that already have *(K-1)* zeros and the ones that can be obtained by Gaussian elimination
- recoverable number of symbols
- $\delta_l(v)$ degrees of freedom
- If $\delta_l(v)=1$ $\Rightarrow$ $X=1,\quad P(X=1)=p$

$1<\delta_l(v)<K$ $\Rightarrow$ $L=\delta_l(v)-l$ Lines to perform Gaussian elimination

$L=l$ lines with $K-1$ zeros



Byzantine and pollution attacks

- Robustness against faulty/malicious components with arbitrary behavior, e.g.
  - dropping packets
  - misdirecting packets
  - sending spurious information
- Abstraction as Byzantine generals problem [LSP82]
- Byzantine robustness in networking [P88,MR97,KMM98,CL99]

Problem setup

- Random linear network coding using coding vectors.
- A batch of r packets is multicast from a source node s to a set of sink nodes.
- A packet that is not a linear combination of its input packets is called *adversarial*.
- z_0 – the maximum number of adversarial packets
- m – the minimum source-sink cut capacity
- ρ – proportion of redundant symbols in each packet
- An *omniscient* adversary can observe transmission on the entire network
- Main results:
  - If the adversary is omniscient, the information rate of the code approaches $m\text{-}2z_0$ asymptotically as the packet size increases.
  - If the adversary is NOT omniscient, and the source and the sinks share a secret channel not observed by the adversary, a rate of $m\text{-}z_0$ is asymptotically achievable.
  - Will give details for the omniscient adversary case.

Byzantine and pollution attacks – correction at decoding time

- Distributed randomized network coding can be extended to detect Byzantine behavior
  - Small computational and communication overhead
  - small number of hash bits included with each packet, calculated as simple polynomial function of data
- Require only that a Byzantine attacker does not design and supply modified packets with complete knowledge of other nodes' packets
- Main scheme:
  - Use a polynomial hash
  - An attacker without full knowledge of the traffic will have low probability of being able to match the hash
  - The hash can be used to detect an attack [HLKMEK04]
- One can further use such a hash to decode

Omniscient adversary case

- Input matrix, $\boldsymbol{X}$, whose ith row, $\boldsymbol{x}_i$, corresponds to the ith input packet.
  - The first n-ρn-r entries of $\boldsymbol{x}_i$ are independent exogenous data symbols.
  - The next ρn are redundant symbols.
  - The last r symbols form the coding vector.
- An adversarial packet can be viewed as an additional source packet, and $\boldsymbol{Z}$ is the matrix whose ith row is the ith adversarial packet.
- The received packets at a terminal node can be represented by $\boldsymbol{Y}$, given by

$$\boldsymbol{Y} = \boldsymbol{GX} + \boldsymbol{KZ}$$

  where $\boldsymbol{G}$ and $\boldsymbol{K}$ are the linear mappings from the source and the adversarial packets respectively to the sink.
- Let $\boldsymbol{G}'$ be the last r columns of $\boldsymbol{Y}$.
- The sink knows $\boldsymbol{G}'$ but not $\boldsymbol{G}$.

Omniscient adversary case

Lemma 1:

With probability at least 1-η/q, the matrix $\boldsymbol{G}'$ has full column rank, where η is the number of links in the network, and q is the size of the finite field.

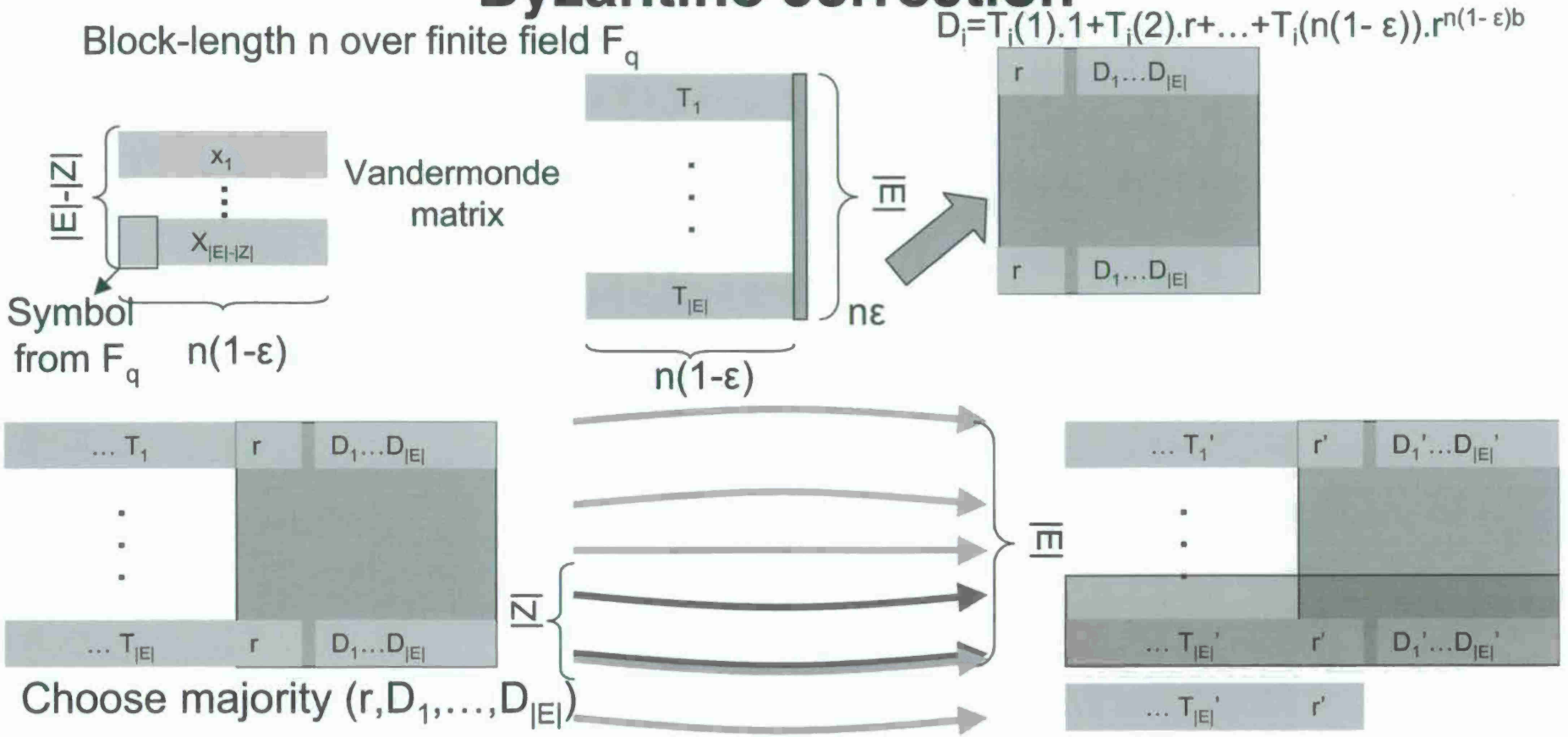
Proposition 1:

With probability greater than 1-$q^{n\varepsilon}$, the input matrix $\boldsymbol{X}$ can be recovered, and the decoding algorithm has complexity $O(n^3m^3)$.

Byzantine correction

Block-length n over finite field F_q

$D_i=T_i(1).1+T_i(2).r+\ldots+T_i(n(1-\varepsilon)).r^{n(1-\varepsilon)b}$



$D_i=T_i(1)'.1+T_i(2)'.r+\ldots+T_i(n(1-\varepsilon))'.r^{n(1-\varepsilon)}$? If so, accept T_i, else reject T_i

Use accepted T_is to decode

[Jaggi05], [JLHE05], [JKLHDKM07]

Network error correcting codes for adversarial errors in multiple source networks

Overview:

- Network error correcting codes allow reliable transmission over a network that is subject to adversarial errors [KK07, 08]
- Existing work gives bounds and explicit code constructions for single-source multicast networks
- We generalize these results to multiple source multicast networks

Model and definitions:

- We consider a directed graph G with a set U of source nodes, and an adversary that can introduce arbitrary errors on up to z links
- The region of reliable multicast transmission rates k_i from the i^{th} source to the sinks is given in terms of the minimum cut capacities m_S between sources in subsets S of U and each sink

Approach and results:

- The reliable communication rates under z adversarial errors satisfy
  - Singleton bound: $\sum_{i \in S} k_i \le m_S - 2z, \forall S \subseteq U$
  - Hamming bound: $\sum_{i \in S} k_i \le m_S - \log_q \sum_{j=0}^{z} \binom{m_S}{j} (q-1)^j, \forall S \subseteq U$

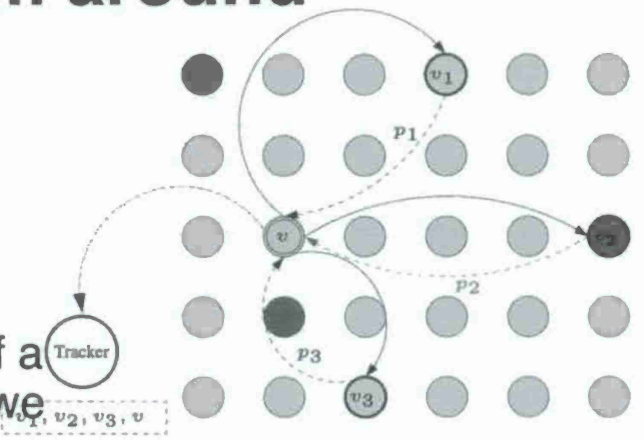
Network error correcting codes – keeping enough degrees of freedom around

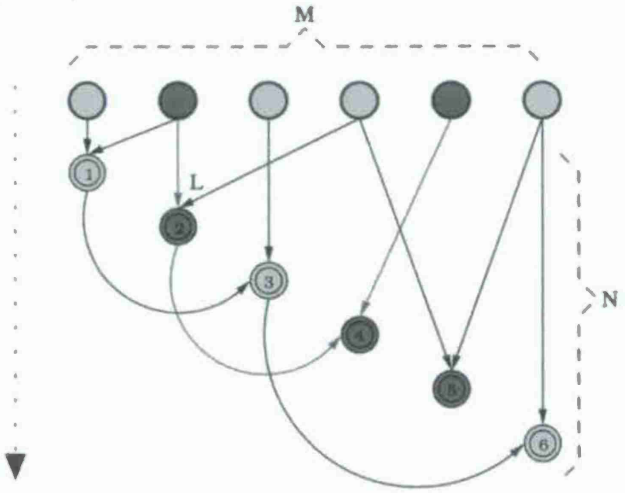
Overview:

- Determining the level of diversity against pollution is crucial in ensuring the operation of coding against attackers [LBK08]

Model, definitions and approach:

- In order to be able to model accurately the topology of a peer-to-peer distributed network with network coding we introduce the following:
  - Evolving overlay network: Scale-free random grap
  - Three types of nodes: data source, data collector, data keeper
  - A tracker keeps a record of all nodes that store packets
  - Each keeper connects to a positive number of nodes in order to create diversity in the linear independence of packets in the network





Information contact graph evolving through time.

Verification for content distribution without decoding

- We may want is a means of detecting that a single packet is polluted **without decoding**
- Need a homomorphic signature scheme that allows nodes to verify any linear combination of pieces without contacting the original sender or decoding packets
- Can use homomorphic hash functions [ADMK05], [GR06]
- Can use Secure Random Checksum (SRC) which requires less computation than the homomorphic hash function, but requires a secure channel to all the nodes [KFM06]
- A signature scheme without a secure channel for transmitting hash values and associated digital signatures of received and transmitted blocks
  - Weil pairing on elliptic curves provides authentication of the data in addition to pollution [CJL06]
  - Use a scheme that relies on the network coding scheme intrinsically [ZKMH07]
- We take a novel approach that uses a more **algebraic** angle [ZKMH07], [HHKMZ07]
- It can be shown that it is as hard as the $(p, m, m+n)$ Diffie-Hellman problem
- Overheads
  - Part of the public key has to be re-generated for each file
  - Signature vector
- If the file sizes are large, after the initial setup, each additional file distributed only incurs a negligible amount of overhead using our signature scheme
- Our signature scheme has to be applied on the original file, not on hashes

Conclusions

- This program has provided us the ability to develop means of:
  - Establishing new means of data protection using network coding
  - Constructing families of codes that are near-optimal theoretically to recover from Byzantine attacks without locating them
  - Creating a means for verifying validity of data without decoding or using a trusted authority
  - Creating a means of tracking reliability of network under network coding

Networks with random erasures and adversarial errors

Overview:

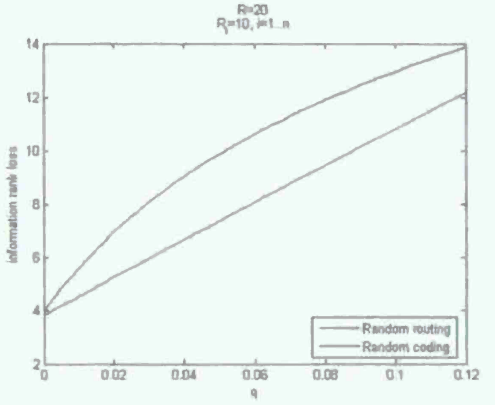
- Network codes offer useful error and erasure correction capabilities, but can also suffer from error propagation
- The extent and manner in which network coding should be applied is shown to depend on the network topology and the probability distribution of erasures and errors

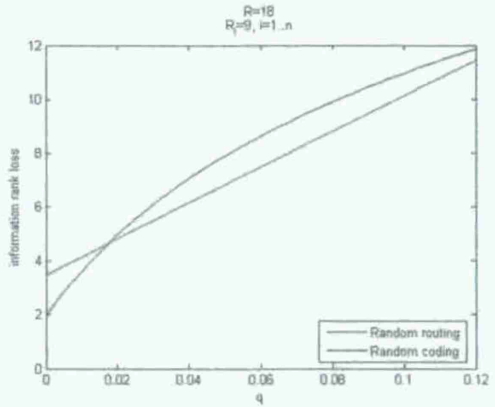
Model:

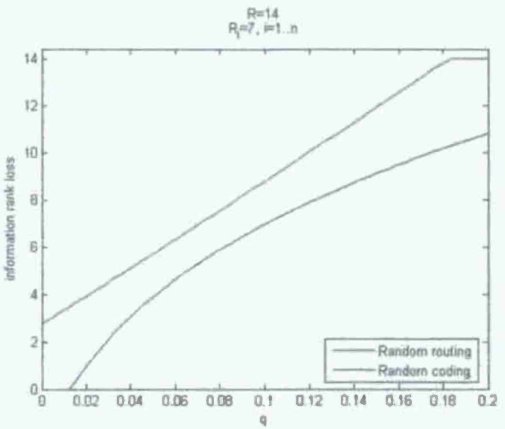
- Packet transmissions in the network are randomly subject to erasures and adversarial or arbitrary errors, with probabilities *p* and *q* respectively
- We compare different coding and routing strategies for transmitting over multiple source-sink paths

Approach and results:

- We show that when the source performs random coding ,the problem can be reduced to optimization of the strategy used on each path
- We define a quantity called information rank loss which can be used as a proxy for probability of successful decoding in the optimization problem (minimize information rank loss)
- We find that random coding becomes more beneficial relative to routing as the redundancy (minimum cut capacity *C* – source information rate *R*) increases, and as *q* decreases relative to *p* (as shown in the graphs for 2 equal paths with *p=0.1, C=20*)
- We can also optimize the trade-off between coding across paths and coding among packets transmitted on a path







Network Coding for Network Security

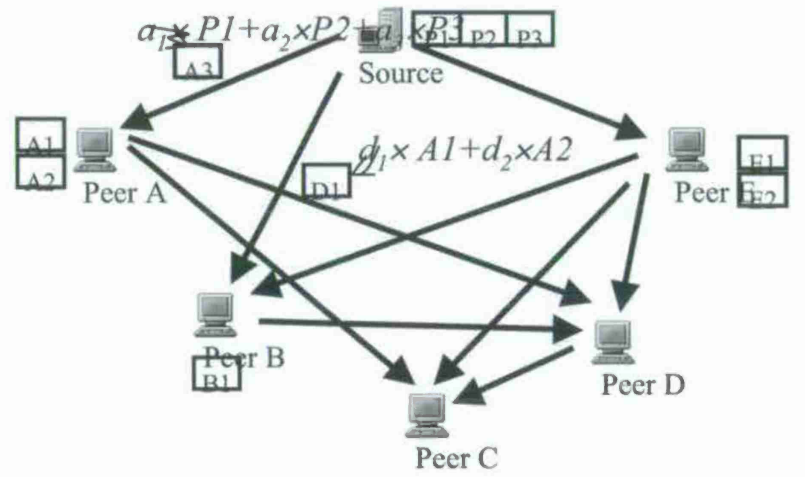
Keesook Han AFRL
Tracey Ho Caltech
Ralf Koetter Technical University Munich
Muriel Medard MIT

Network Coding for Network Security

Objective

Use network coding to enable greater Robustness and security

- Use network coding for detection and correction of Byzantine attackers
- Provide network-coding based verification systems without the need for a trusted authority
- Use network coding for providing free cyphers in the network



Verification can occur without trusted authority

Scientific/Technical Approach

- Use the algebraic linear mixing of data to detect Byzantine attackers through the use of polynomial hashes
- Extend MDS-style codes in conjunction with hash-based majority voting scheme
- Generalize coding bounds by use of q-Johnson scheme, akin to the Grassmannian manifold approach in continuous cases
- Use discrete-log approach for verification in a manner that is robust to linear operations
- Use data mixture as one-time pad in the network

Accomplishments

- New algorithms for detection and correction of attacks in the context of users with shared secrets, omniscient adversaries and limited adversaries
  - New theoretical basis for the study of errors an erasure based on q-Johnson scheme
  - A new algorithm for secure network-coding based peer-to-peer file exchanges based on new signature schemes, developed with H-P Laborat

Challenges

- Applying our verification approach on hash functions of the data.

Impact and outreach

- The support of AFOSR is important since this topic is directly rooted in adversarial settings. As such, namely as an investigation of information dissemination robustness using network coding in a hostile setting, **this topic is identified as natural research in military contexts**
- We believe that **demonstrating the robustness of network coding** when it is combined with error correction in a natural way, will be an enabling factor for the application of network coding in highly volatile, hostile scenarios. Moreover, the techniques are applicable also in non-adversary network contexts, which promises to have a high impact factor.
- Collaboration with Dina Katabi, Sachin Katti (MIT CSAIL), Sid Jaggi (Chinese University of Hong Kong), Michael Lanberg (The Open University of Israel), Michelle Effros (Caltech), Ton Kalker (HP Labs) – Commercial impact and synergistic collaboration with DARPA ITMANET and CBMANET projects for transitioning ideas to MANETs

Byzantine and pollution attacks

- Robustness against faulty/malicious components with arbitrary behavior, e.g.
  - dropping packets
  - misdirecting packets
  - sending spurious information
- Abstraction as Byzantine generals problem [LSP82]
- Byzantine robustness in networking [P88,MR97,KMM98,CL99]

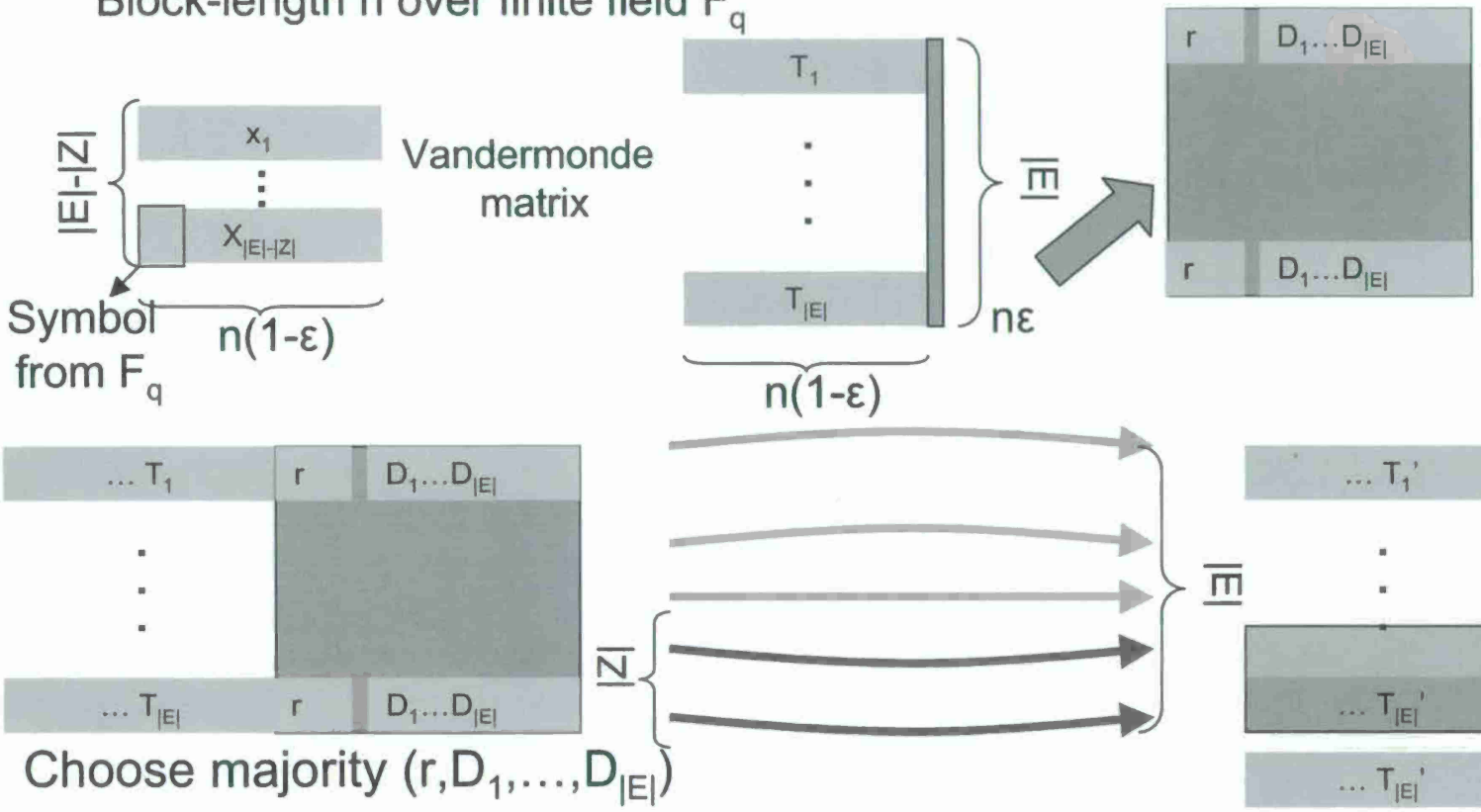
Byzantine and pollution attacks

- Distributed randomized network coding can be extended to detect Byzantine behavior
  - Small computational and communication overhead
  - Small number of hash bits included with each packet, calculated as simple polynomial function of data
- Require only that a Byzantine attacker does not design and supply modified packets with complete knowledge of other nodes' packets
- Main scheme:
  - Use a polynomial hash
  - An attacker without full knowledge of the traffic will have low probability of being able to match the hash
  - The hash can be used to detect an attack [HLKMEK04]
- One can further use such a hash to decode

Byzantine correction

$D_i=T_i(1).1+T_i(2).r+\ldots+T_i(n(1-\varepsilon)).r^{n(1-\varepsilon)}$

Block-length n over finite field F_q



Choose majority $(r,D_1,\ldots,D_{|E|})$

$D_i=T_i(1)'.1+T_i(2)'.r+\ldots+T_i(n(1-\varepsilon))'.r^{n(1-\varepsilon)}$? If so, accept T_i, else reject T_i

Use accepted T_is to decode

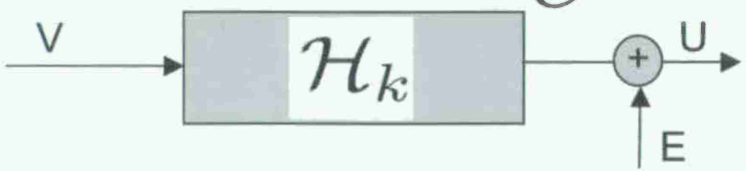
[Jaggi05], [JLHE05], [JKLHDKM07]

Network coding and network error correction

- Random network coding is susceptible to modifications of packets (adversary, jamming, non-hostile, packet erasures)
- Error correction in combination with network coding was considered by Yeung et al., and Zhang - here the network topology plays a central role
- Work in the context of Byzantine modifications in arbitrary networks: Ho et al. and Jaggi et al.
- We consider a network as a modeled by a random linear operator reflecting the
- Operation of **random network coding** on a network of **unknown** topology

Operator Channel: **Input** is a subspace V of ambient n-dimensional space W, H is a **random linear operator** mapping V to a k-dimensional subspace of V; E is an error space of dimension t(E)
Output is a subspace U of W

$$U = \mathcal{H}_k(V) \oplus E$$

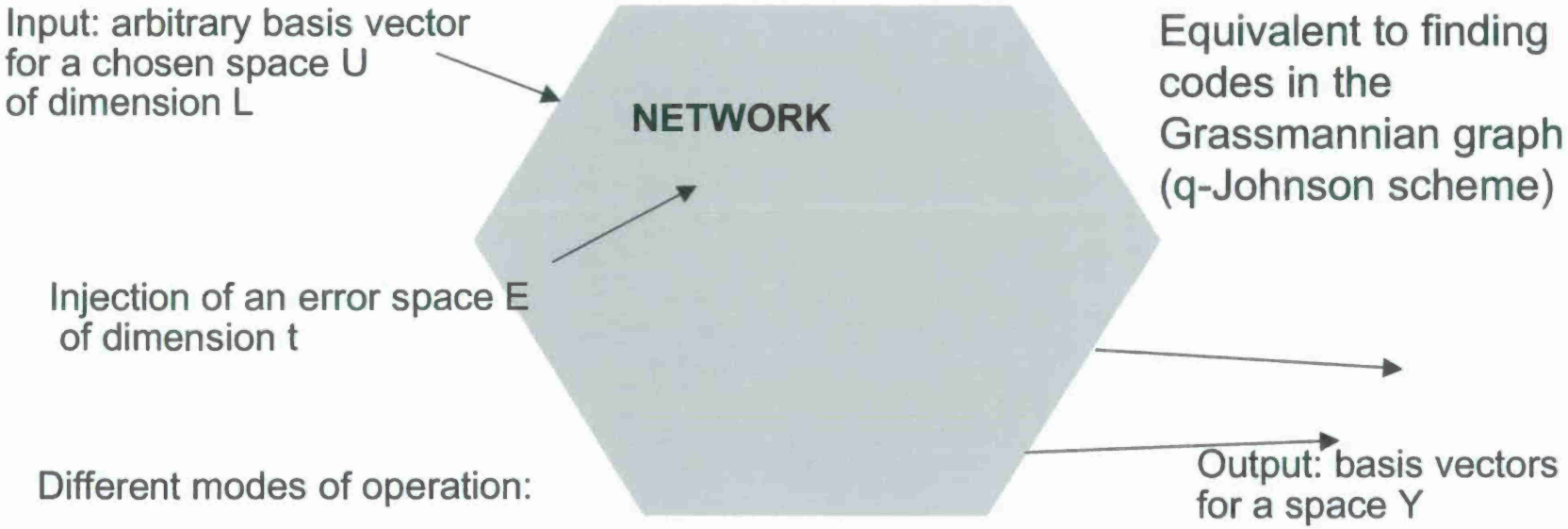


This formulation is very similar to non-coherent detection in the MIMO case: Zheng and Tse

Network coding and network error correction

Just as in the MIMO case: Constructing codes is equivalent to packing subspaces of dimension λn in ambient space of dimension n. The metric for defining distance between two spaces A,B is

$$d(A,B) := \dim(A \oplus B) - \dim(A \cap B)$$



Different modes of operation:

$\tilde{n} = dim(U) - dim(U \cap Y)$ is the number of "erasures"

$t = dim(Y) - dim(Y \cap U)$ is the number of "errors"

We can correct any number of errors and erasures as long as $2(\tilde{n} + t) < D$

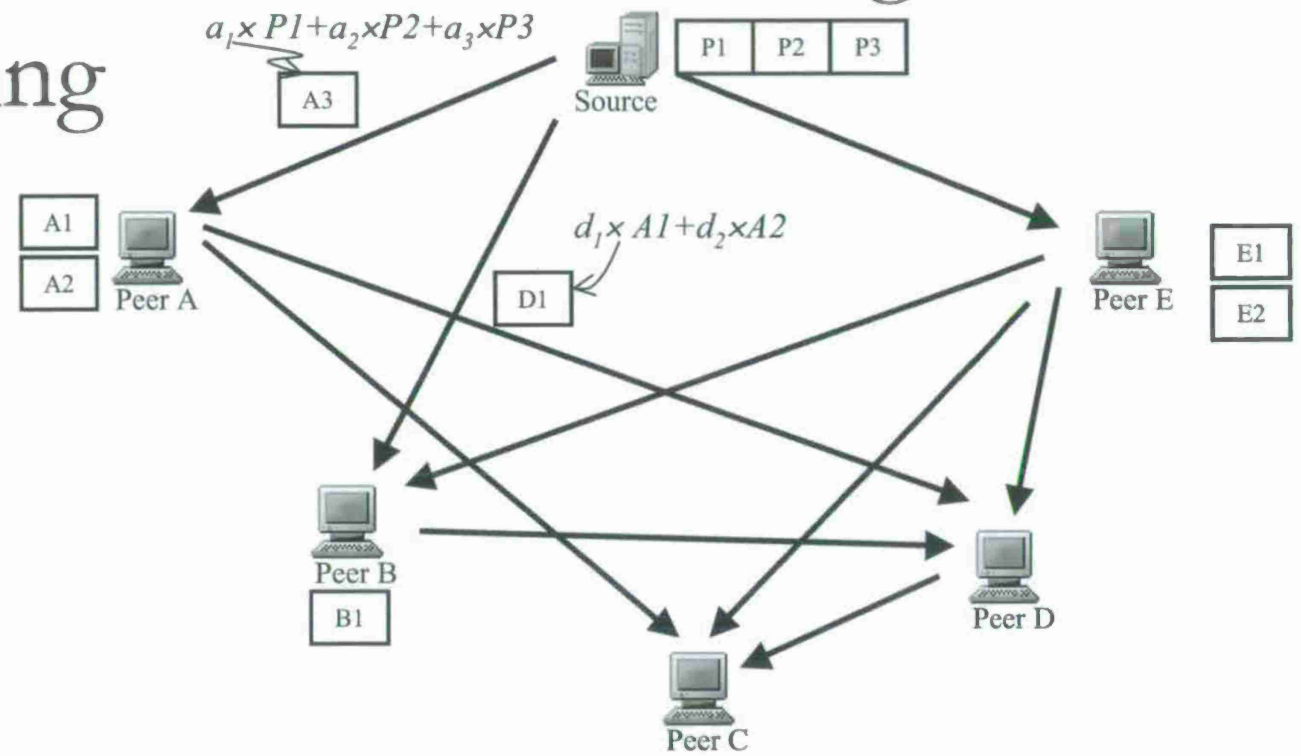
Content distribution of large files

- Distribution of large files to many users.
- Traditional solutions are based on a client-server model.
- Alternative technique - P2P swamping.
- Example - BitTorrent
  - Divide file into many pieces.
  - Client requests different pieces from server(s) or other users.
  - Client becomes server to pieces downloaded.
  - When a client has obtained all pieces, re-construct the whole file.
  - Problem: hard to do optimal scheduling of pieces to nodes.

Content distribution using network coding

- Use network coding to increase the efficiency of network coding in a P2P cooperative architecture [ADMK05], [DPR05], [GR05], [DGWR07]
- Instead of storing pieces on servers, store random linear combination of the pieces on servers
- Clients also generate random linear combination of the pieces they have received to send out
- When a client has accumulated enough degrees of freedom, decode to obtain the whole file

Content distribution using network coding



- A malicious user can send packets with valid linear combination in the header, but garbage in the payload
- The pollution of packets spreads quickly
- Need a homomorphic signature scheme that allows nodes to verify any linear combination of pieces without contacting the original sender or decoding packets

Verification for content distribution

- Can use homomorphic hash functions in content distribution systems to detect polluted packets [ADMK05], [GR06]
- Can use Secure Random Checksum (SRC) which requires less computation than the homomorphic hash function, but requires a secure channel to transmit the SRCs to all the nodes in the network [KFM06]
- A signature scheme without a secure channel for transmitting hash values and associated digital signatures of received and transmitted blocks;
  - Weil pairing on elliptic curves provides authentication of the data in addition to pollution [CJL06]
  - Use a scheme that relies on the network coding scheme intrinsically [ZKMH07]

Problem formulation

- A source s wishes to send a large file to a group of peers, T
- View the data to be transmitted as vectors $\overline{\mathbf{v}}_1, \ldots, \overline{\mathbf{v}}_m$ in n-dimensional vector space F_p^n, where p is a prime. The source node augments these vectors to $\mathbf{v}_1, \ldots, \mathbf{v}_m$ given by

$$\mathbf{v}_i = (0, \ldots, 1, \ldots, 0, \overline{v}_{i1}, \ldots, \overline{v}_{in})$$

  where the first m elements are zero except the i-th one is 1, and $\overline{v}_{ij} \in F_p$
- Each packet received by a peer is a linear combination of all the pieces.

$$\mathbf{w} = \sum_{i=1}^{m} \beta_i \mathbf{v}_i$$

Signature for network coding

- The vectors $\mathbf{v}_1, \ldots, \mathbf{v}_m$ span a subspace V of F_p^{m+n}.
- A received packet is a valid linear combination if and only if it belongs to V.
- Each node verifies the integrity of a received vector w by checking the membership of w in V.
- Our approach has the following ingredients:
  - q: a large prime such that p is a divisor of q -1.
  - g: a generator of the group G of order p in F_q.
  - Private key: $K_{pr} = \{a_i\}_{i=1,\ldots,m+n}$, a random set of elements in F_q^*.
  - Public key: $K_{pu} = \{h_i = g^{a_i}\}_{i=1,\ldots,m+n}$.

Signature for network coding

- The scheme works as follows:
  - The source finds a vector $\boldsymbol{u}$ that is orthogonal to all vectors in $\boldsymbol{V}$.
  - The source computes vector $\mathbf{x} = (u_1 / a_1, \ldots, u_{m+n} / a_{m+n})$.
  - The source signs $\mathbf{x}$ with some standard signature scheme and publishes it.
  - When a node receives a vector $\boldsymbol{w}$ and wants to verify that $\boldsymbol{w}$ is in $\boldsymbol{V}$, it computes

$$d = \prod_{i=1}^{m+n} h_i^{x_i w_i}$$

and verifies that $d = 1$.

Discussion

- It can be shown that it is as hard as the $(p, m, m+n)$ Diffie-Hellman problem
- Thus, it is as hard as the Discrete Logarithm problem to find new vectors that also satisfy the verification criterion other than those that are in V [BF99]
- Overheads
  - Part of the public key K_{pu} has to be re-generated for each file, otherwise a malicious node can use the information from the previous file
  - Signature vector, x
- If the file sizes are large, after the initial setup, each additional file distributed only incurs a negligible amount of overhead using our signature scheme
- Our signature scheme has to be applied on the original file, not on hashes.

Looking forward

- The free cypher of network coding can lead to new ways of managing security in the networks:
  - Using partial knowledge of a file as a cypher
  - Rate-based security in networks?
- Network coding for pollution detection and correction:
  - How do we locate attackers?
  - What is the effect of incorrect network management?
  - Connection with network tomography using network coding [FM05, 06]
- Verification in network coding:
  - Can we find further network coding specific schemes?
  - Can we use schemes on hashes?
  - What are the interactions with free cyphers in networks?

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.

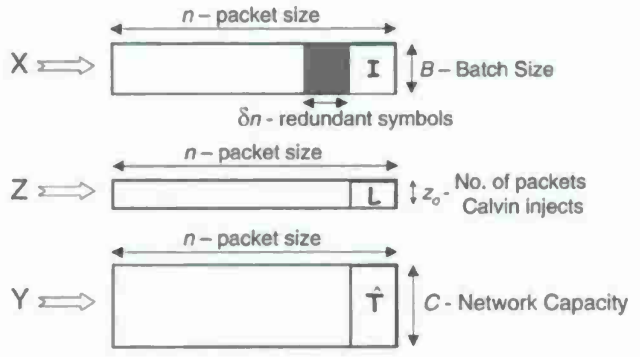


Fig. 1. Alice, Bob, and Calvin's information matrices.

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 F_q. All arithmetic operations henceforth are done over symbols from 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 S 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 $\vec{X}$ is a vector corresponding to the matrix X. Upon receiving Y, Bob computes F and solves the system

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

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

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

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

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 $\vec{X}$ 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 $\vec{X}$. 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:

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

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 $\mathbf{S}$ that is unknown to Calvin. Once $\mathbf{S}$ 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 $\vec{U}'$. Alice multiplies $\vec{U}'$ by a random square matrix to generate the input $\vec{U}$. This vector $\vec{U}$ 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 $\vec{U}$. He then obtains $\vec{U}'$ from $\vec{U}$, by inverting the mapping specified in Alice's encoder. He finally extracts from $\vec{U}'$ 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. Cai, 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.

Byzantine Modification Detection in Multicast Networks With Random Network Coding

Tracey Ho, *Member, IEEE*, Ben Leong, Ralf Koetter, *Senior Member, IEEE*, Muriel Médard, *Fellow, IEEE*, Michelle Effros, *Senior Member, IEEE*, and David R. Karger, *Associate Member, IEEE*

***Abstract*—An information-theoretic approach for detecting Byzantine or adversarial modifications in networks employing random linear network coding is described. Each exogenous source packet is augmented with a flexible number of hash symbols that are obtained as a polynomial function of the data symbols. This approach depends only on the adversary not knowing the random coding coefficients of all other packets received by the sink nodes when designing its adversarial packets. We show how the detection probability varies with the overhead (ratio of hash to data symbols), coding field size, and the amount of information unknown to the adversary about the random code.**

***Index Terms*—Byzantine adversary, multicast, network coding, network error detection.**

I. Introduction

We consider the problem of information-theoretic detection of Byzantine, i.e., arbitrary, modifications of transmitted data in a network coding setting.

Interest in network coding has grown following demonstrations of its various advantages: in network capacity [1], robustness to nonergodic network failures [2] and ergodic packet erasures [3], [4], and distributed network operation [5]. Multicast in overlay and *ad hoc* networks is a promising application. Since packets are forwarded by end hosts to other end hosts, such networks are susceptible to Byzantine errors introduced by compromised end hosts.

We show that Byzantine modification detection capability can be added to a multicast scheme based on random linear block network coding [5], [6], with modest additional computational and communication overhead, by incorporating a simple polynomial hash/check value in each packet. With this approach, a sink node can detect Byzantine modifications with high probability, as long as these modifications have not been designed with knowledge of the random coding combinations present in all other packets obtained at the sink: the only essential condition is the adversary's incomplete knowledge of the random network code seen by the sink. No other assumptions are made regarding the topology of the network or the adversary's power to corrupt or inject packets. The adversary can know the entire message as well as portions of the random network code, and can have the same (or greater) transmission capacity compared to the source. This approach works even in the extreme case where every packet received by a sink has been corrupted by being coded together with an independent adversarial packet. This new adversarial model may be useful for application scenarios in which conventional assumptions of an upper bound on adversarial transmission capacity are less appropriate. For instance, in some peer-to-peer or wireless *ad hoc* settings we may not know how many adversarial nodes might join the network, while it may be more likely that there will be some transmissions that are not received by the adversarial nodes. In such cases, our approach can provide a useful alternative to existing methods.

Our approach provides much flexibility in trading off between the detection probability, the proportion of redundancy, the coding field size, and the amount of information about the random code that is not observed by the adversary. This approach can be used for low overhead monitoring during normal conditions when no adversary is known to be present, in conjunction with more complex, higher overhead techniques which are activated upon detection of a Byzantine error, such as adding more redundancy for error correction.

A preliminary version of this work with less general assumptions appeared in [7]. The security model is substantially generalized and strengthened in this work.

A. Background and Related Work

The problem of secure network communications in the presence of Byzantine adversaries has been studied extensively, e.g., [8]–[11]. A survey of information-theoretic research in this area is given in [12]. Two important issues are secrecy and authenticity;[1] this work concerns the latter. Like one-time pads [13], our approach relies on the generation of random values unknown to the adversary, though the one-time pad provides secrecy and not authenticity.

In the network coding context, the problem of ensuring secrecy in the presence of a wiretap adversary has been considered in [14]–[16]. The problem of correcting adversarial errors, which is complementary to our work, has been studied in [17]–[21].

Adversarial models in existing works on information-theoretic authenticity techniques commonly assume some upper bound on the number of adversarial transmissions, which leads to a requirement on the amount of redundant network capacity. For the problem of adversarial error correction or resilient communication, the number of links/transmissions controlled by the adversary must necessarily be limited with respect to the number of links/transmissions in a minimum source–sink cut or the amount of redundancy transmitted by the source. For instance, in the resilient communication problem of Dolev *et al.* [9], the source and sink are connected by n wires, and their model requires that no more than $(n-1)/2$ wires are disrupted by an adversary for resilient communication to be possible. In the network coding error correction problems of [17], [20], [21], the rate of redundant information that the source needs to transmit is between one and two times the maximum rate of information that can be injected by the adversary, depending on the specific adversarial model.

The above techniques can also be considered in the context of error detection. For example, in one phase of the secret sharing based algorithm in [9], the source communicates a degree τ polynomial $f(x) \in \mathbb{F}_q(x)$ by sending $f(i)$ on the ith wire. If the adversary controls at most $n-\tau$ wires, any errors it introduces can be detected. In general, for approaches based on error-correcting codes such as in [17], the number

[1]These are independent attributes of a cryptographic system [13].

Manuscript received November 24, 2006; revised February 16, 2008. This work was supported in part by AFOSR under Grant 5710001972, the Caltech Lee Center for Networking, and a gift from Microsoft Research. The material in this correspondence was presented in part at the IEEE International Symposium on Information Theory, Chicago, IL, June/July 2004.

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

B. Leong is with the National University of Singapore, Singapore, 119260 Republic of Singapore (e-mail: benleong@comp.nus.edu.sg).

R. Koetter is with the Institute for Communications Engineering, Technische Universität München, D-80290 München, Germany (e-mail: ralf.koetter@tum.de).

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

D. R. Karger is with the Computer Science and Artificial Intelligence Laboratory, Massachusetts Institute of Technology (MIT), Cambridge, MA 02139 USA (e-mail: karger@csail.mit.edu).

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

Digital Object Identifier 10.1109/TIT.2008.921894

of adversarial errors that can be detected is given by the difference between the source–sink minimum cut and the source information rate.

Such approaches have a threshold nature in that they do not offer graceful performance degradation when the number of adversarial transmissions exceeds the assumed upper bound. Their efficiency is also sensitive to overestimates of adversarial transmission capacity, which determines the amount of redundancy required.

The adversarial model considered in this work is slightly different. Instead of assuming a limit on the number of adversarial errors, our only assumption is on the incompleteness of the adversary's knowledge of the random code (the adversary can know the entire source message). In this case, the overhead (proportion of redundant information transmitted by the source) is no longer a function of the estimated upper bound on the number of adversarial errors. Instead, it is a design parameter which, as we will show, can be flexibly traded off against detection probability and coding field size. Unlike approaches based on secret sharing and its variants, where the required proportional overhead is a function of the adversarial strength, in our approach, for any nonzero proportional overhead and any adversarial strength short of full knowledge or control of network transmissions, the detection probability can be made arbitrarily high by increasing the field size. The former has the advantage of deterministic guarantees, while our approach has the advantage of greater flexibility with additional performance parameters that can be traded off against one another.

The use of our error detection technique for low-overhead monitoring under normal conditions when no adversary is known to be present, in conjunction with a more complex technique activated upon detection of an adversary, has a parallel in works such as [22] and [23]. These works optimize for normal conditions by using less complex message authentication codes and signed digests, respectively, during normal operation, resorting to more complex recovery mechanisms only upon detection of a fault.

B. Notation

In this work, we denote matrices with bold uppercase letters and vectors with bold lowercase letters. All vectors are row vectors unless indicated otherwise with a subscript T. We denote by $[\boldsymbol{x}, \boldsymbol{y}]$ the concatenation of two row vectors $\boldsymbol{x}$ and $\boldsymbol{y}$.

II. Model and Problem Formulation

Consider random linear block network coding [5], [6], [24] of a block of r exogenous packets which originate at a source node and are multicast to one or more sink nodes. We assume that the network coding subgraph is given by some separate mechanism, the details of which we are not concerned with.[2] An adversary observes some subset of packets transmitted in the network, and can corrupt, insert or delete one or more packets, or corrupt some subset of nodes. The only assumption we make is that the adversary's observations are limited such that when designing the adversarial packets, the adversary does not know the random coding combinations present in all other packets obtained at the sinks. This assumption is made precise using the notion of secret packets which we define below. The source and sinks do not share any keys or common information.

Each packet p in the network is represented by a row vector $\boldsymbol{w}_p$ of $d + c + r$ symbols from a finite field F_q, where the first d entries are data symbols, the next c are redundant hash symbols, and the last r form the packet's (global) coefficient vector $\boldsymbol{t}_p$. The field size is 2 to the power of the symbol length in bits. The hash symbols in each exogenous packet are given by a function $\psi_d : \mathsf{F}_q^d \to \mathsf{F}_q^c$ of the data symbols. The coefficient vector of the ith exogenous packet is the unit vector with a single nonzero entry in the ith position. The coefficient vectors are used for decoding at the sinks as explained below.

Each packet transmitted by the source node is an independent random linear combination of the r exogenous packets, and each packet transmitted by a nonsource node is an independent random linear combination of packets received at that node. The coefficients of these linear combinations are chosen with the uniform distribution from the finite field F_q, and the same linear operation is applied to each symbol in a packet. For instance, if packet p_3 is formed as a random linear combination of packets p_1 and p_2, then $\boldsymbol{w}_{p_3} = \gamma_{1,3}\boldsymbol{w}_{p_1} + \gamma_{2,3}\boldsymbol{w}_{p_2}$ where $\gamma_{1,3}$ and $\gamma_{2,3}$ are random scalar coding coefficients distributed uniformly over F_q.

Let row vector $\boldsymbol{m}_i \in \mathsf{F}_q^{(c+d)}$ represent the concatenation of the data and hash symbols for the ith exogenous packet, and let $\boldsymbol{M}$ be the matrix whose ith row is $\boldsymbol{m}_i$. A packet p is *genuine* if its data/hash symbols are consistent with its coefficient vector, i.e., $\boldsymbol{w}_p = [\boldsymbol{t}_p\boldsymbol{M}, \boldsymbol{t}_p]$. The exogenous packets are genuine, and any packet formed as a linear combination of genuine packets is also genuine. *Adversarial packets*, i.e., packets transmitted by the adversary, may contain arbitrary coefficient vector and data/hash values. An adversarial packet p can be represented in general by $[\boldsymbol{t}_p\boldsymbol{M} + \boldsymbol{v}_p, \boldsymbol{t}_p]$, where $\boldsymbol{v}_p$ is an arbitrary vector F_q^{c+d}. If $\boldsymbol{v}_p$ is nonzero, p (and linear combinations of p with genuine packets) are nongenuine.

A set $\mathcal{S}$ of packets can be represented as a block matrix $[\boldsymbol{T}_\mathcal{S}\boldsymbol{M} + \boldsymbol{V}_\mathcal{S}|\boldsymbol{T}_\mathcal{S}]$ whose ith row is $\boldsymbol{w}_{p_i}$, where p_i is the ith packet of the set. A sink node t attempts to decode when it has collected a *decoding set* consisting of r linearly independent packets (i.e., packets whose coefficient vectors are linearly independent). For a decoding set $\mathcal{D}$, the decoding process is equivalent to premultiplying the matrix $[\boldsymbol{T}_\mathcal{D}\boldsymbol{M} + \boldsymbol{V}_\mathcal{D}|\boldsymbol{T}_\mathcal{D}]$ with $\boldsymbol{T}_\mathcal{D}^{-1}$. This gives $\left[\boldsymbol{M} + \boldsymbol{T}_\mathcal{D}^{-1}\boldsymbol{V}_\mathcal{D}|\boldsymbol{I}\right]$, i.e., the receiver decodes to $\boldsymbol{M} + \tilde{\boldsymbol{M}}$, where

$$\tilde{\boldsymbol{M}} = \boldsymbol{T}_\mathcal{D}^{-1}\boldsymbol{V}_\mathcal{D} \tag{1}$$

gives the disparity between the decoded packets and the original packets. If at least one packet in a decoding set is nongenuine, $\boldsymbol{V}_\mathcal{D} \neq 0$, and the decoded packets will differ from the original packets. A decoded packet is *inconsistent* if its data and hash values do not match, i.e., applying the function ψ_d to its data values does not yield its hash values. If one or more decoded packets are inconsistent, the sink declares an error.

The coefficient vector of a packet transmitted by the source is uniformly distributed over F_q^r; if a packet whose coefficient vector has this uniform distribution is linearly combined with other packets, the resulting packet's coefficient vector has the same uniform distribution. We are concerned with the distribution of decoding outcomes conditioned on the adversary's information, i.e., the adversary's observed and transmitted packets, and its information on independencies/dependencies among packets. Note that in this setup, scaling a packet by some scalar element of F_q does not change the distribution of decoding outcomes.

For given $\boldsymbol{M}$, the value of a packet p is specified by the row vector $\boldsymbol{u}_p = [\boldsymbol{t}_p, \boldsymbol{v}_p]$. We call a packet p *secret* if, conditioned on the value of $\boldsymbol{v}_p$ and the adversary's information, its coefficient vector $\boldsymbol{t}_p$ is uniformly distributed over $\mathsf{F}_q^r \backslash W$ for some (possibly empty) subspace or affine space $W \subset \mathsf{F}_q^r$.[3] Intuitively, secret packets include genuine packets whose coefficient vectors are unknown (in the above sense)

[2]The network coding subgraph defines the times at which packets are or can be transmitted on each network link (see, e.g., [25]).

[3]This definition of a secret packet is conservative as it does not distinguish between packets with a nonuniform conditional distribution and packets that are fully known to the adversary. Taking this distinction into account would make the analysis more complicated but would in some cases give a better bound on detection probability.

to the adversary, as well as packets formed as linear combinations involving at least one secret packet. A set $\mathcal{S}$ of secret packets is *secrecy-independent* if each of the packets remains secret when the adversary is allowed to observe the other packets in the set; otherwise it is *secrecy-dependent*. Secrecy-dependencies arise from the network transmission topology, for instance, if a packet p is formed as a linear combination of a set $\mathcal{S}$ of secret packets (possibly with other nonsecret packets), then $\mathcal{S} \cup \{p\}$ is secrecy-dependent.

To illustrate these definitions, suppose that the adversary knows that a sink's decoding set contains an adversarial packet p_1 as well as a packet p_4 formed as some linear combination $k_2\boldsymbol{w}_{p_2} + k_3\boldsymbol{w}_{p_3}$ of an adversarial packet p_2 with a genuine packet p_3, so the adversary knows $\boldsymbol{t}_{p_1}, \boldsymbol{t}_{p_2}, \boldsymbol{v}_{p_1}, \boldsymbol{v}_{p_2}$ and $\boldsymbol{v}_{p_3} = \boldsymbol{0}$. Since a decoding set consists of packets with linearly independent coefficient vectors, the adversary knows that $\boldsymbol{t}_{p_1}$ and $\boldsymbol{t}_{p_3}$ are linearly independent. Suppose also that the adversary does not observe the contents of any packets dependent on p_3. Thus, the distribution of $\boldsymbol{t}_{p_4}$, conditioned on the adversary's information and any potential value $k_2\boldsymbol{v}_{p_2}$ for $\boldsymbol{v}_{p_4}$, is uniform over $\mathbb{F}_q^r \backslash \{k\boldsymbol{t}_{p_1} : k \in \mathbb{F}_q\}$. Also, packets p_3 and p_4 are secrecy-dependent.

Consider a decoding set $\mathcal{D}$ containing one or more secret packets. Choosing an appropriate packet ordering, we can express $[\boldsymbol{T}_\mathcal{D}|\boldsymbol{V}_\mathcal{D}]$ in the form

$$[\boldsymbol{T}_\mathcal{D}|\boldsymbol{V}_\mathcal{D}] = \left[\begin{array}{c|c} \boldsymbol{A}+\boldsymbol{B}_1 & \boldsymbol{V}_1 \\ \hline \boldsymbol{C}\boldsymbol{A}+\boldsymbol{B}_2 & \boldsymbol{V}_2 \\ \hline \boldsymbol{B}_3 & \boldsymbol{V}_3 \end{array}\right] \tag{2}$$

where for any given values of $\boldsymbol{B}_i \in \mathbb{F}_q^{s_i \times r}$, $\boldsymbol{V}_i \in \mathbb{F}_q^{s_i \times (d+c)}$, $i = 1, 2, 3$, and $\boldsymbol{C} \in \mathbb{F}_q^{s_2 \times s_1}$, the matrix $\boldsymbol{A} \in \mathbb{F}_q^{s_1 \times r}$ has a conditional distribution that is uniform over all values for which $\boldsymbol{T}_\mathcal{D}$ is nonsingular. The first $s_1 + s_2$ rows correspond to secret packets, and the first s_1 rows correspond to a set of secrecy-independent packets. $s_2 = 0$ if there are no secrecy-dependencies among the secret packets in $\mathcal{D}$.

This notion of secret packets provides the most general characterization of the conditions under which the scheme succeeds. For a given network topology, a requirement on the number of secrecy-independent secret packets received at the sink can be translated into constraints on the subsets of links/packets the adversary can observe and/or modify. For instance, if information is sent on n parallel paths from a source to a sink node, then the number of secrecy-independent secret packets is the number of linearly independent packets received on paths that are not observed or controlled by the adversary.

Note that we allow each packet of the decoding set to be corrupted with an independent adversarial packet, as long as at least one of the packets has been formed as a linear combination with some secret packet.

III. Main Results

In the following theorem, we consider decoding from a set of packets that contains some nongenuine packet, which causes the decoded packets to differ from the original exogenous packets. The first part of the theorem gives a lower bound on the number of equally likely potential values of the decoded packets—the adversary cannot narrow down the set of possible outcomes beyond this regardless of how it designs its adversarial packets. The second part provides, for a simple polynomial hash function, an upper bound on the proportion of potential decoding outcomes that can have consistent data and hash values, in terms of $k = \lceil \frac{d}{c} \rceil$, the ceiling of the ratio of the number of data symbols to hash symbols. Larger values for k correspond to lower overheads but lower probability of detecting an adversarial modification. This tradeoff is a design parameter for the network.

Theorem 1: Consider a decoding set $\mathcal{D}$ containing a secrecy-independent subset of s_1 secret (possibly nongenuine) packets, and suppose the decoding set contains at least one nongenuine packet.

a) The adversary cannot determine which of a set of at least $(q-1)^{s_1}$ equally likely values of the decoded packets will be obtained at the sink. In particular, there will be at least s_1 packets such that, for each of these, the adversary cannot determine which of a set of at least $q-1$ equally likely values will be obtained.

b) Let $\psi : \mathbb{F}_q^k \to \mathbb{F}_q$ be the function mapping $(x_1, \ldots, x_k)$, $x_i \in \mathbb{F}_q$, to

$$\psi(x_1, \ldots, x_k) = x_1^2 + \cdots + x_k^{k+1} \tag{3}$$

where $k = \lceil \frac{d}{c} \rceil$. Suppose the function ψ_d mapping the data symbols $x_1, \ldots, x_d$ to the hash symbols $y_1, \ldots, y_c$ in an exogenous packet is defined by

$$y_i = \psi(x_{(i-1)k+1}, \ldots, x_{ik}), \quad \forall\, i = 1, \ldots, c-1$$
$$y_c = \psi(x_{(c-1)k+1}, \ldots, x_d).$$

Then the probability of not detecting an error is at most $\left(\frac{k+1}{q}\right)^{s_1}$.

Corollary 1: Let the hash function ψ_d be defined as in Theorem 1b. Suppose a sink obtains more than r packets, including a secrecy-independent set of s secret packets, and at least one nongenuine packet. If the sink decodes using two or more decoding sets whose union includes all its received packets, then the probability of not detecting an error is at most $\left(\frac{k+1}{q}\right)^{s}$.

Example: With 2% overhead ($k = 50$), symbol length $= 7$ bits, $s = 5$, the detection probability is at least 98.9%; with 1% overhead ($k = 100$), symbol length $= 8$ bits, $s = 5$, the detection probability is at least 99.0%.

IV. Development, Proofs, and Ancillary Results

A. Vulnerable Scenarios

Before analyzing the scenario described in the previous sections, we first point out when this approach fails to detect adversarial modifications.

First, the sink needs some way of knowing if the source stops transmitting, otherwise, the assumption of no shared secret information results in the adversary being indistinguishable from the source. One possibility is that the source either transmits at a known rate or is inactive, and that the sink knows at what rates it should be receiving information on various subsets of incoming links when the source is active. If the adversary is unable to reproduce those information rates, e.g., because it does not control the same part of the network as the source, then the sink knows when the source is inactive.

Second, if the adversary knows that the genuine packets received at a sink have coefficient vectors that lie in some w-dimensional subspace $W \subset \mathbb{F}_q^r$, the following strategy allows it to control the decoding outcome and so ensure that the decoded packets have consistent data and hash values.

The adversary ensures that the sink receives w genuine packets with linearly independent coefficient vectors in W, by supplying additional such packets if necessary. The adversary also supplies the sink with $r - w$ nongenuine packets whose coefficient vectors $\boldsymbol{t}_1, \ldots, \boldsymbol{t}_{r-w}$ are not in W. Let $\boldsymbol{t}_{r-w+1}, \ldots, \boldsymbol{t}_r$ be a set of basis vectors for W, and let $\boldsymbol{T}$ be the matrix whose ith row is $\boldsymbol{t}_i$. Then the coefficient vectors of the r packets can be represented by the rows of the matrix

$$\left[\begin{array}{c|c} \boldsymbol{I} & \boldsymbol{0} \\ \hline \boldsymbol{0} & \boldsymbol{K} \end{array}\right] \boldsymbol{T}$$

where $\boldsymbol{K}$ is a nonsingular matrix in $\mathbb{F}_q^{w \times w}$. From (5), we have

$$\left[\begin{array}{c|c} \boldsymbol{I} & \boldsymbol{0} \\ \hline \boldsymbol{0} & \boldsymbol{K} \end{array}\right] \boldsymbol{T}\tilde{\boldsymbol{M}} = \left[\begin{array}{c} \tilde{\boldsymbol{V}} \\ \hline \boldsymbol{0} \end{array}\right]$$

$$\tilde{M} = T^{-1}\left[\begin{array}{c|c} I & 0 \\ \hline 0 & K^{-1} \end{array}\right]\left[\begin{array}{c} \tilde{V} \\ \hline 0 \end{array}\right] = T^{-1}\left[\begin{array}{c} \tilde{V} \\ \hline 0 \end{array}\right].$$

Since the adversary knows T and controls $\tilde{V}$, it can determine $\tilde{M}$.

B. Byzantine Modification Detection

We next consider the scenario described in Section II, where the adversary designs its packets without knowing the contents of one or more secret packets the receiver will use for decoding, and prove the results of Section III.

We first establish two results that are used in the proof of Theorem 1. Consider the hash function defined in (3). We call a vector $(x_1, \ldots, x_{k+1}) \in \mathbb{F}_q^{k+1}$ *consistent* if $x_{k+1} = \psi(x_1, \ldots, x_k)$.

Lemma 1: At most $k+1$ out of the q vectors in a set

$$\{\boldsymbol{u} + \gamma \boldsymbol{v} : \gamma \in \mathbb{F}_q\}$$

where $\boldsymbol{u} = (u_1, \ldots, u_{k+1})$ is a fixed vector in $\mathbb{F}_q^{k+1}$ and $\boldsymbol{v} = (v_1, \ldots, v_{k+1})$ is a fixed nonzero vector in $\mathbb{F}_q^{k+1}$, can be consistent.

Proof: Suppose some vector $\boldsymbol{u} + \gamma \boldsymbol{v}$ is consistent, i.e.,

$$u_{k+1} + \gamma v_{k+1} = (u_1 + \gamma v_1)^2 + \cdots + (u_k + \gamma v_k)^{k+1}. \quad (4)$$

Note that for any fixed value of $\boldsymbol{u}$ and any fixed nonzero value of $\boldsymbol{v}$, (4) is a polynomial equation in γ of degree equal to $1 + \tilde{k}$, where $\tilde{k} \in [1, k]$ is the highest index for which the corresponding $v_{k'}$ is nonzero, i.e., $v_{\tilde{k}} \neq 0, v_{k'} = 0\ \forall\ k' > \tilde{k}$. By the fundamental theorem of algebra, this equation can have at most $1 + \tilde{k} \leq 1 + k$ roots. Thus, the property can be satisfied for at most $1 + k$ values of γ. □

Corollary 2: Let $\boldsymbol{u}$ be a fixed row vector in $\mathbb{F}_q^n$ and $\boldsymbol{Y}$ a fixed nonzero matrix in $\mathbb{F}_q^{n \times (k+1)}$. If row vector $\boldsymbol{g}$ is distributed uniformly over $\mathbb{F}_q^n$, then the vector $\boldsymbol{u} + \boldsymbol{g}\boldsymbol{Y}$ is consistent with probability at most $\frac{k+1}{q}$.

Proof: Suppose the ith row of $\boldsymbol{Y}$, denoted $\boldsymbol{y}_i$, is nonzero. We can partition the set of possible values for $\boldsymbol{g}$ such that each partition consists of all vectors that differ only in the ith entry g_i. For each partition, the corresponding set of values of $\boldsymbol{u} + \boldsymbol{g}\boldsymbol{Y}$ is of the form $\{\boldsymbol{u}' + g_i \boldsymbol{y}_i : g_i \in \mathbb{F}_q\}$. The result follows from Lemma 1 and the fact that g_i is uniformly distributed over $\mathbb{F}_q$. □

Proof of Theorem 1: We condition on any given values of $\boldsymbol{B}_i, \boldsymbol{V}_i, i = 1, 2, 3$, and $\boldsymbol{C}$ in (2). Writing $\boldsymbol{A}' = \boldsymbol{A} + \boldsymbol{B}_1$, $\boldsymbol{T}_{\mathcal{D}}$ becomes

$$\begin{bmatrix} \boldsymbol{A}' \\ \boldsymbol{C}(\boldsymbol{A}' - \boldsymbol{B}_1) + \boldsymbol{B}_2 \\ \boldsymbol{B}_3 \end{bmatrix}.$$

From (1), we have

$$\begin{bmatrix} \boldsymbol{A}' \\ \boldsymbol{C}(\boldsymbol{A}' - \boldsymbol{B}_1) + \boldsymbol{B}_2 \\ \boldsymbol{B}_3 \end{bmatrix} \tilde{\boldsymbol{M}} = \begin{bmatrix} \boldsymbol{V}_1 \\ \boldsymbol{V}_2 \\ \boldsymbol{V}_3 \end{bmatrix}$$

$$\begin{bmatrix} \boldsymbol{A}' \\ -\boldsymbol{C}\boldsymbol{B}_1 + \boldsymbol{B}_2 \\ \boldsymbol{B}_3 \end{bmatrix} \tilde{\boldsymbol{M}} = \begin{bmatrix} \boldsymbol{V}_1 \\ \boldsymbol{V}_2 - \boldsymbol{C}\boldsymbol{V}_1 \\ \boldsymbol{V}_3 \end{bmatrix}$$

which we can simplify to

$$\begin{bmatrix} \boldsymbol{A}' \\ \boldsymbol{B}' \end{bmatrix} \tilde{\boldsymbol{M}} = \begin{bmatrix} \boldsymbol{V}_1 \\ \boldsymbol{V}_2' \end{bmatrix} \quad (5)$$

by writing

$$\boldsymbol{B}' = \begin{bmatrix} -\boldsymbol{C}\boldsymbol{B}_1 + \boldsymbol{B}_2 \\ \boldsymbol{B}_3 \end{bmatrix}, \quad \boldsymbol{V}_2' = \begin{bmatrix} \boldsymbol{V}_2 - \boldsymbol{C}\boldsymbol{V}_1 \\ \boldsymbol{V}_3 \end{bmatrix}.$$

Since the determinant of a matrix is not changed by adding a multiple of one row to another row, and $\begin{bmatrix} \boldsymbol{A}' \\ \boldsymbol{B}' \end{bmatrix}$ is obtained from $\boldsymbol{T}_{\mathcal{D}}$ by a sequence of such operations, we have

$$\begin{bmatrix} \boldsymbol{A}' \\ \boldsymbol{B}' \end{bmatrix} \text{ is nonsingular } \Leftrightarrow \boldsymbol{T}_{\mathcal{D}} \text{ is nonsingular.}$$

Thus, matrix $\boldsymbol{A}' \in \mathbb{F}_q^{s_1 \times r}$ has a conditional distribution that is uniform over the set $\mathcal{A}$ of values for which $\begin{bmatrix} \boldsymbol{A}' \\ \boldsymbol{B}' \end{bmatrix}$ is nonsingular.

The condition that the decoding set contains at least one nongenuine packet corresponds to the condition $\boldsymbol{V}_{\mathcal{D}} \neq 0$. We consider two cases. In each case, we show that we can partition the set $\mathcal{A}$ such that at most a fraction $\left(\frac{k+1}{q}\right)^{s_1}$ of values in each partition give decoding outcomes $\boldsymbol{M} + \tilde{\boldsymbol{M}}$ with consistent data and hash values. The result then follows since the conditional distribution of values within each partition is uniform.

Case 1: $\boldsymbol{V}_2' \neq 0$. Let $\boldsymbol{v}_i$ be some nonzero row of $\boldsymbol{V}_2'$, and $\boldsymbol{b}_i$ the corresponding row of $\boldsymbol{B}'$. Then $\boldsymbol{b}_i \tilde{\boldsymbol{M}} = \boldsymbol{v}_i$.

We first partition $\mathcal{A}$ into cosets

$$\mathcal{A}_n = \{\boldsymbol{A}_n + \boldsymbol{r}^T \boldsymbol{b}_i : \boldsymbol{r} \in \mathbb{F}_q^{s_1}\}, \quad n = 1, 2, \ldots, \chi$$

where

$$\chi = \frac{|\mathcal{A}|}{q^{s_1}}.$$

This can be done by the following procedure. Any element of $\mathcal{A}$ can be chosen as $\boldsymbol{A}_1$. Matrices $\boldsymbol{A}_2, \boldsymbol{A}_3, \ldots, \boldsymbol{A}_\chi$ are chosen sequentially; for each $m = 2, \ldots, \chi$, $\boldsymbol{A}_m$ is chosen to be any element of $\mathcal{A}$ not in the cosets $\mathcal{A}_n, n < m$. Note that this forms a partition of $\mathcal{A}$, since the presence of some element c in two sets $\mathcal{A}_n$ and $\mathcal{A}_m$, $n < m$, implies that $\boldsymbol{A}_m$ is also in $\mathcal{A}_n$, which is a contradiction. It is also clear that each coset has size $\left|\{\boldsymbol{r} : \boldsymbol{r} \in \mathbb{F}_q^{s_1}\}\right| = q^{s_1}$.

For each such coset $\mathcal{A}_n$, the corresponding values of $\tilde{\boldsymbol{M}}$ satisfy, from (5)

$$\begin{bmatrix} \boldsymbol{A}_n + \boldsymbol{r}^T \boldsymbol{b}_i \\ \boldsymbol{B}' \end{bmatrix} \tilde{\boldsymbol{M}} = \begin{bmatrix} \boldsymbol{V}_1 \\ \boldsymbol{V}_2' \end{bmatrix}$$

$$\begin{bmatrix} \boldsymbol{A}_n \\ \boldsymbol{B}' \end{bmatrix} \tilde{\boldsymbol{M}} = \begin{bmatrix} \boldsymbol{V}_1 - \boldsymbol{r}^T \boldsymbol{v}_i \\ \boldsymbol{V}_2' \end{bmatrix}$$

$$\tilde{\boldsymbol{M}} = \begin{bmatrix} \boldsymbol{A}_n \\ \boldsymbol{B}' \end{bmatrix}^{-1} \begin{bmatrix} \boldsymbol{V}_1 - \boldsymbol{r}^T \boldsymbol{v}_i \\ \boldsymbol{V}_2' \end{bmatrix}$$

where $\boldsymbol{r} \in \mathbb{F}_q^{s_1}$. Let $\boldsymbol{U}$ be the submatrix consisting of the first s_1 columns of $\begin{bmatrix} \boldsymbol{A}_n \\ \boldsymbol{B}' \end{bmatrix}^{-1}$. Since $\boldsymbol{U}$ is full rank, we can find a set $\mathcal{J} \subset \{1, \ldots, r\}$ of s_1 indices that correspond to linearly independent rows of $\boldsymbol{U}$. Let $[\boldsymbol{U}_1 \mid \boldsymbol{U}_2]$ be the $s_1 \times r$ submatrix of $\begin{bmatrix} \boldsymbol{A}_n \\ \boldsymbol{B}' \end{bmatrix}^{-1}$ consisting of rows with indices in $\mathcal{J}$. Consider the corresponding rows of $\boldsymbol{M} + \tilde{\boldsymbol{M}}$, which can be expressed in the form

$$\boldsymbol{M}_{\mathcal{J}} + \boldsymbol{U}_1 \boldsymbol{V}_1 - \boldsymbol{U}_1 \boldsymbol{r}^T \boldsymbol{v}_i + \boldsymbol{U}_2 \boldsymbol{V}_2' \quad (6)$$

where $\boldsymbol{M}_{\mathcal{J}}$ is the submatrix of $\boldsymbol{M}$ consisting of rows corresponding to set $\mathcal{J}$. Since $\boldsymbol{U}_1$ is nonsingular by the choice of $\mathcal{J}$, $\boldsymbol{U}_1 \boldsymbol{r}^T$ takes potentially any value in $\mathbb{F}_q^{s_1}$. Thus, the set of potential values for each row of (6), for any given value of $\boldsymbol{M}_{\mathcal{J}}, \boldsymbol{A}_n, \boldsymbol{B}', \boldsymbol{V}_1, \boldsymbol{V}_2', \boldsymbol{v}_i$, and the other rows, is of the form $\{\boldsymbol{u} + \gamma \boldsymbol{v}_i : \gamma \in \mathbb{F}_q\}$ where $\boldsymbol{u}$ is a function of $\boldsymbol{M}_{\mathcal{J}}, \boldsymbol{A}_n, \boldsymbol{B}', \boldsymbol{V}_1, \boldsymbol{V}_2'$. Applying Lemma 1 yields the result for this case.

Case 2: $\boldsymbol{V}_2' = 0$, i.e., $\boldsymbol{V}_2 - \boldsymbol{C}\boldsymbol{V}_1 = \boldsymbol{V}_3 = 0$. Then $\boldsymbol{V}_1 \neq 0$, since otherwise $\boldsymbol{V}_1 = \boldsymbol{V}_2 = 0$ and $\boldsymbol{V}_{\mathcal{D}} = 0$ which would contradict the assumption that there is at least one nongenuine packet.

We partition $\mathcal{A}$ such that each partition consists of all matrices in $\mathcal{A}$ that have the same row space

$$\mathcal{A}_n = \left\{ \boldsymbol{R}\boldsymbol{A}_n : \ \boldsymbol{R} \in \mathbb{F}_q^{s_1 \times s_1}, \ \det(\boldsymbol{R}) \neq 0 \right\}, \quad n = 1, 2, \ldots, \chi$$

where

$$|\mathcal{A}_n| = \prod_{i=0}^{s_1 - 1} \left(q^{s_1} - q^i \right), \quad \chi = \frac{|\mathcal{A}|}{|\mathcal{A}_n|}.$$

This can be done by choosing any element of $\mathcal{A}$ as $\boldsymbol{A}_1$, and choosing $\boldsymbol{A}_n, n = 2, \ldots, \chi$ sequentially such that $\boldsymbol{A}_n$ is any element of $\mathcal{A}$ not in $\mathcal{A}_m, m < n$.

For each $\mathcal{A}_n, n = 1, \ldots, \chi$, the corresponding values of $\tilde{\boldsymbol{M}}$ satisfy, from (5)

$$\begin{bmatrix} \boldsymbol{R}\boldsymbol{A}_n \\ \boldsymbol{B}' \end{bmatrix} \tilde{\boldsymbol{M}} = \begin{bmatrix} \boldsymbol{V}_1 \\ \boldsymbol{0} \end{bmatrix}$$

$$\begin{bmatrix} \boldsymbol{A}_n \\ \boldsymbol{B}' \end{bmatrix} \tilde{\boldsymbol{M}} = \begin{bmatrix} \boldsymbol{R}^{-1}\boldsymbol{V}_1 \\ \boldsymbol{0} \end{bmatrix}$$

$$\tilde{\boldsymbol{M}} = \begin{bmatrix} \boldsymbol{A}_n \\ \boldsymbol{B}' \end{bmatrix}^{-1} \begin{bmatrix} \boldsymbol{R}^{-1}\boldsymbol{V}_1 \\ \boldsymbol{0} \end{bmatrix}.$$

Let $\boldsymbol{U}$ be the submatrix consisting of the first s_1 columns of $\begin{bmatrix} \boldsymbol{A}_n \\ \boldsymbol{B}' \end{bmatrix}^{-1}$. We can find an ordered set $\mathcal{J} = \{i_1, \ldots, i_{s_1} : \ i_1 < \cdots < i_{s_1}\} \subset \{1, \ldots, r\}$ of s_1 indices that correspond to linearly independent rows of $\boldsymbol{U}$. Let $\boldsymbol{U}_{\mathcal{J}}$ and $\boldsymbol{M}_{\mathcal{J}}$ be the submatrices of $\boldsymbol{U}$ and $\tilde{\boldsymbol{M}}$, respectively, consisting of the s_1 rows corresponding to $\mathcal{J}$. Then $\boldsymbol{U}_{\mathcal{J}}$ is nonsingular, and the value of the matrix representation of the corresponding decoded packets is uniformly distributed over the set

$$\left\{ \boldsymbol{M}_{\mathcal{J}} + \boldsymbol{R}'\boldsymbol{V}_1 : \ \boldsymbol{R}' \in \mathbb{F}_q^{s_1 \times s_1}, \ \det(\boldsymbol{R}') \neq 0 \right\}. \tag{7}$$

Let ν be the rank of $\boldsymbol{V}_1$. Consider a set of ν linearly independent rows of $\boldsymbol{V}_1$. Denote by $\mathcal{I}$ the corresponding set of row indices, and denote by $\boldsymbol{V}_{\mathcal{I}}$ the submatrix of $\boldsymbol{V}_1$ consisting of those rows. We can write

$$\boldsymbol{V}_1 = \boldsymbol{L}\boldsymbol{V}_{\mathcal{I}}$$

where $\boldsymbol{L} \in \mathbb{F}_q^{s_1 \times \nu}$ has full rank ν. We define $\boldsymbol{R}_{\mathcal{I}} = \boldsymbol{R}'\boldsymbol{L}$, noting that

$$\boldsymbol{R}_{\mathcal{I}}\boldsymbol{V}_{\mathcal{I}} = \boldsymbol{R}'\boldsymbol{L}\boldsymbol{V}_{\mathcal{I}} = \boldsymbol{R}'\boldsymbol{V}_1$$

and that $\boldsymbol{R}_{\mathcal{I}}$ is uniformly distributed over all matrices in $\mathbb{F}_q^{s_1 \times \nu}$ that have full rank ν. Thus, (7) becomes

$$\left\{ \boldsymbol{M}_{\mathcal{J}} + \boldsymbol{R}_{\mathcal{I}}\boldsymbol{V}_{\mathcal{I}} : \ \boldsymbol{R}_{\mathcal{I}} \in \mathbb{F}_q^{s_1 \times \nu}, \ \operatorname{rank}(\boldsymbol{R}_{\mathcal{I}}) = \nu \right\}. \tag{8}$$

Denote by $\boldsymbol{r}_1, \ldots, \boldsymbol{r}_{s_1}$ the rows of $\boldsymbol{R}_{\mathcal{I}}$, and by $\boldsymbol{R}_n$ the submatrix of $\boldsymbol{R}_{\mathcal{I}}$ consisting of its first n rows. We consider the rows sequentially, starting with the first row $\boldsymbol{r}_1$. For $n = 1, \ldots, s_1$, we will show that conditioned on any given value of $\boldsymbol{R}_{n-1}$, the probability that the i_nth decoded packet $\boldsymbol{M}_{i_n} + \boldsymbol{r}_n\boldsymbol{V}_{\mathcal{I}}$ is consistent is at most $\frac{k+1}{q}$.

Case A: $\boldsymbol{R}_{n-1}$ has zero rank. This is the case if $n = 1$, or if $n > 1$ and $\boldsymbol{R}_{n-1} = 0$.

Suppose we remove the restriction $\operatorname{rank}(\boldsymbol{R}_{\mathcal{I}}) = \nu$, so that $\boldsymbol{r}_n$ is uniformly distributed over $\mathbb{F}_q^{\nu}$. By Corollary 2, $\boldsymbol{m}_{i_n} + \boldsymbol{r}_n\boldsymbol{V}_{\mathcal{I}}$ would have consistent data and hash values with probability at most $\frac{k+1}{q}$. With the restriction $\operatorname{rank}(\boldsymbol{R}_{\mathcal{I}}) = \nu$, the probability of $\boldsymbol{r}_n$ being equal to 0 is lowered. Since the corresponding decoded packet $\boldsymbol{m}_{i_n} + \boldsymbol{r}_n\boldsymbol{V}_{\mathcal{I}}$ is consistent for $\boldsymbol{r}_n = 0$, the probability that it is consistent is less than $\left(\frac{k+1}{q}\right)$.

Case B: $n > 1$ and $\boldsymbol{R}_{n-1}$ has nonzero rank.

Conditioned on $\boldsymbol{r}_n$ being in the row space of $\boldsymbol{R}_{n-1}$, $\boldsymbol{r}_n = \boldsymbol{g}\boldsymbol{R}_{n-1}$ where $\boldsymbol{g}$ is uniformly distributed over $\mathbb{F}_q^{n-1}$. Since $\boldsymbol{V}_{\mathcal{I}}$ has linearly independent rows, $\boldsymbol{R}_{n-1}\boldsymbol{V}_{\mathcal{I}} \neq 0$, and by Corollary 2, the corresponding decoded packet

$$\boldsymbol{m}_{i_n} + \boldsymbol{r}_n\boldsymbol{V}_{\mathcal{I}} = \boldsymbol{m}_{i_n} + \boldsymbol{g}\boldsymbol{R}_{n-1}\boldsymbol{V}_{\mathcal{I}}$$

is consistent with probability at most $\frac{k+1}{q}$.

Conditioned on $\boldsymbol{r}_n$ not being in the row space of $\boldsymbol{R}_{n-1}$, we can partition the set of possible values for $\boldsymbol{r}_n$ into cosets

$$\left\{ \boldsymbol{r} + \boldsymbol{g}\boldsymbol{R}_{n-1} : \ \boldsymbol{g} \in \mathbb{F}_q^{n-1} \right\}$$

where $\boldsymbol{r}$ is not in the row space of $\boldsymbol{R}_{n-1}$; the corresponding values of the i_nth decoded packet are given by

$$\left\{ \boldsymbol{m}_{i_n} + \boldsymbol{r}\boldsymbol{V}_{\mathcal{I}} + \boldsymbol{g}\boldsymbol{R}_{n-1}\boldsymbol{V}_{\mathcal{I}} : \ \boldsymbol{g} \in \mathbb{F}_q^{n-1} \right\}.$$

Noting as before that $\boldsymbol{R}_{n-1}\boldsymbol{V}_{\mathcal{I}} \neq 0$ and applying Corollary 2, the i_nth decoded packet is consistent with probability at most $\frac{k+1}{q}$. □

Proof of Corollary 1: Suppose two or more different sets of packets are used for decoding. If not all of them contain at least one nongenuine packet, the decoded values obtained from different decoding sets will differ: sets containing only genuine packets will be decoded to $\boldsymbol{M}$, while sets containing one or more nongenuine packets will not. This will indicate an error.

Otherwise, suppose all the decoding sets contain at least one nongenuine packet. Let $\mathcal{S}$ denote the set of s secrecy-independent packets. Consider the decoding sets in turn, denoting by s_i' the number of unmodified packets from $\mathcal{S}$ in the ith decoding set that are not in any set $j < i$. Conditioned on any fixed values of packets in sets $j < i$, there remain s_i' secrecy-independent packets in the ith decoding set, and we have from Theorem 1 that at most a fraction $\left(\frac{k+1}{q}\right)^{s_i'}$ of decoding outcomes for set i have consistent data and hash values. Thus, the overall fraction of consistent decoding outcomes is at most

$$\left(\frac{k+1}{q}\right)^{\sum_i s_i'} = \left(\frac{k+1}{q}\right)^{s}.$$ □

V. Conclusion

We have described an information-theoretic approach for detecting Byzantine modifications in networks employing random linear network coding. Byzantine modification detection capability is added by augmenting each packet with a small, flexible number of hash symbols; this overhead can be traded off against the detection probability and symbol length. The hash symbols can be obtained as a simple polynomial function of the data symbols. The only necessary condition is that the adversarial packets are not all designed with knowledge of the random coding coefficients of all other packets received by the sink nodes. This approach can be used in conjunction with higher overhead schemes that are activated only upon detection of a Byzantine node.

Acknowledgment

The authors would like to thank the Associate Editor and reviewers for their valuable comments and suggestions.

References

[1] R. Ahlswede, N. Cai, S.-Y. R. Li, and R. W. Yeung, "Network information flow," *IEEE Trans. Inf. Theory*, vol. 46, no. 4, pp. 1204–1216, Jul. 2000.

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

[3] A. F. Dana, R. Gowaikar, R. Palanki, B. Hassibi, and M. Effros, "Capacity of wireless erasure networks," *IEEE Trans. Inf. Theory*, vol. 52, no. 3, pp. 789–804, Mar. 2006.
[4] D. S. Lun, M. Médard, and M. Effros, "On coding for reliable communication over packet networks," in *Proc. 42nd Annu. Allerton Conf. Communication, Control, and Computing*, Monticello, IL, Sep./Oct. 2004.
[5] T. Ho, R. Koetter, M. Médard, D. R. 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.
[6] P. A. Chou, Y. Wu, and K. Jain, "Practical network coding," in *Proc. 41st Annu. Allerton Conf. Communication, Control, and Computing*, 2003.
[7] T. Ho, B. Leong, R. Koetter, M. Médard, M. Effros, and D. R. Karger, "Byzantine modification detection for multicast networks using randomized network coding," in *Proc. 2004 IEEE Int. Symp. Information Theory* , Chicago, IL, Jun./Jul. 2004, p. 144.
[8] R. Perlman, "Network Layer Protocols with Byzantine Robustness," Ph.D. dissertation, Massachusetts Institute of Technology, Cambridge, 1988.
[9] D. Dolev, C. Dwork, O. Waarts, and M. Yung, "Perfectly secure message transmission," *J. ACM*, vol. 40, no. 1, pp. 17–47, Jan. 1993.
[10] D. Malkhi and M. Reiter, "A high-throughput secure reliable multicast protocol," *J. Comp. Security*, vol. 5, pp. 113–127, 1997.
[11] P. Papadimitratos and Z. J. Haas, "Secure routing for mobile ad hoc networks," in *Proc. SCS Communication Netwirks and Distributed Systems Modeling and Simulation Conf. (CNDS 2002)*, San Antonio, TX, Jan. 2002.
[12] Y. Desmedt, "Unconditionally private and reliable communication in an untrusted network," in *Proc. IEEE Information Theory Workshop on Theory and Practice in Information-Theoretic Security*, Awaji Island, Japan, Oct. 2005, pp. 38–41.
[13] J. L. Massey, "Contemporary cryptography: An introduction," in *Contemporary Cryptology: The Science of Information Integrity*, G. J. Simmons, Ed. New York: Wiley/IEEE, 1999.
[14] N. Cai and R. W. Yeung, "Secure network coding," in *Proc. IEEE Int. Symp. Information Theory*, Lausanne, Switzerland, Jun./Jul. 2002, p. 323.
[15] 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./Oct. 2004.
[16] K. Bhattad and K. R. Nayayanan, "Weakly secure network coding," in *Proc. WINMEE, RAWNET, and NETCOD 2005 Workshops*, Riva del Garda, Italy, Apr. 2005.
[17] R. W. Yeung and N. Cai, "Network error correction, part I: Basic concepts and upper bounds," *Commun. Inf. Syst.*, vol. 6, no. 1, pp. 19–36, 2006.
[18] N. Cai and R. W. Yeung, "Network error correction, part II: Lower bounds," *Commun. Inf. Syst.*, vol. 6, no. 1, pp. 37–54, 2006.
[19] K. K. Chi and X. M. Wang, "Analysis of network error correction based on network coding," *IEE Proc.—Communications*, vol. 152, no. 4, 2005.
[20] S. Jaggi, M. Langberg, S. Katti, T. Ho, D. Katabi, and M. Médard, "Resilient network coding in the presence of byzantine adversaries," in *Proc. IEEE INFOCOM 2007*, Anchorage, AK, May 2007, pp. 616–624.
[21] R. Koetter and F. R. Kschischang, "Coding for errors and erasures in random network coding," *IEEE Trans. Inf. Theory*, submitted for publication.
[22] M. Castro and B. Liskov, "Practical byzantine fault tolerance," in *OSDI: Proc. Symp. Operating Systems Design and Implementation*, New Orleans, LA, 1999, pp. 173–186.
[23] K. P. Kihlstrom, L. E. Moser, and P. M. Melliar-Smith, "The SecureRing protocols for securing group communication," in *Proc. 31st Annu. Hawaii Int. Conf. System Sciences (HICSS)*, 1998, vol. 3, pp. 317–326.
[24] T. Ho, M. Médard, R. Koetter, D. R. Karger, M. Effros, J. Shi, and B. Leong, "A random linear network coding approach to multicast," *IEEE Trans. Inf. Theory*, vol. 52, no. 10, pp. 4413–4430, Oct. 2006.
[25] D. S. Lun, N. Ratnakar, M. Médard, R. Koetter, D. R. Karger, T. Ho, E. Ahmed, and F. Zhao, "Minimum-cost multicast over coded packet networks," *IEEE Transactions on Information Theory*, vol. 52, no. 6, pp. 2608–2623, Jun. 2006.

Computational Complexity of Continuous Variable Quantum Key Distribution

Yi-Bo Zhao, You-Zhen Gui, Jin-Jian Chen, Zheng-Fu Han, and Guang-Can Guo

***Abstract*—The continuous variable quantum key distribution has been considered to have the potential to provide high secret key rate. However, in present experimental demonstrations, the secret key can be distilled only under very small loss rates. Here, by calculating explicitly the computational complexity with the channel transmission, we show that under high loss rate it is hard to distill the secret key in present continuous variable scheme and one of its advantages, the potential of providing high secret key rate, may therefore be limited.**

***Index Terms*—Computational complexity, continuous variable (CV), error correction, quantum key distribution (QKD), reconciliation.**

I. Introduction

Due to its potential for achieving high modulation and detection speed, continuous variable (CV) quantum key distribution (QKD) has recently attracted more and more attention. Compared to single photon counting schemes, CVQKD does not require single photon sources and detectors which are technically challenging now. The CVQKD schemes typically use the quadrature amplitude of light beams as information carrier, and homodyne detection rather than photon counting. Some of these schemes use nonclassical states, such as squeezed states [1] or entangled states [2], while others use coherent states [3]–[6]. Because the squeezed states and entangled states are sensitive to losses in the quantum channel, coherent states are much more attractive for long distance transmission. To improve the performance of the CVQKD against the channel loss, Grosshans *et al.* proposed a reverse reconciliation (RR) protocol [11]. In the traditional direct reconciliation protocol, Alice sends Bob the quantum state and also sends the reconciliation information later.[1] Finally, Bob obtains Alice's data without any error. However, in the reverse reconciliation protocol, the quantum state is sent by Alice to Bob, but the reconciliation information is sent by Bob to Alice. Finally, Alice gets Bob's received data with no error.

Tabletop experimental setups that encode information in the phase and amplitude of coherent states have been demonstrated [7], [8], and recent experiments have shown the feasibility of CVQKD in optical fibers up to a distance of 55 km [9], [10], but without obtaining the final secret keys.

Unlike the single photon QKD schemes, many CVQKD schemes utilize the inertial quantum noise to protect information from Eve's attack [7], [12]. However, at the same time the quantum noise also causes errors between two legitimate communicators, Alice and Bob. It is widely

Manuscript received January 14, 2007; revised November 25, 2007. This work was supported in part by the National Fundamental Research Program of China under Grant 2006CB921900, National Natural Science Foundation of China under Grants 60537020 and 60621064, the Innovation Fund of the University of Science and Technology of China under Grants KD2006005, and the Knowledge Innovation Project of the Chinese Academy of Sciences (CAS).

The authors are with the Key Lab of Quantum Information, University of Science and Technology of China (CAS), Hefei, Anhui 230026, China (e-mail: zfhan@ustc.edu.cn).

Communicated by H. Imai, Guest Editor for Special Issue on Information Theoretic Security.

Digital Object Identifier 10.1109/TIT.2008.921889

[1]In the following, we use the conventional appellation. Alice is the quantum state sender and Bob is the quantum state receiver.

On network coding for security

Keesook Han	Tracey Ho	Ralf Koetter	Muriel Médard	Fang Zhao
AFRL	Computer Science	ICE	EECS	EECS
	Caltech	TUM	MIT	MIT

Abstract **The use of network coding in military networks opens many interesting issues for security. The mixing of data inherent to network coding may at first appear to pose challenges, but it also enables new security approaches. In this paper, we overview the recent current theoretical understanding and application areas for network-coding based security in the areas of robustness to Byzantine attackers and of distributed signature schemes for downloads.**

I. Introduction

The Global Information Grid (GIG) is the infrastructure used to conduct Net-Centric Operations (NCO). The GIG is intended to be a single information-sharing network with multiple levels of security and bandwidth capabilities in net-centric environment. A net-centric information environment is inclusive of Core and Communities of Interest (COI) enterprise services, a data sharing strategy, and the Task-Post-Process-Use (TPPU) paradigm. The Global Information Grid Bandwidth Expansion (GIG-BE) Program was a major DoD net-centric transformational initiative executed by Defense Information System Agency (DISA).

The ultimate purpose of the GIG-BE projects is to provide a secure and reliable platform to enable worldwide Net-Centric Operations for intelligence, surveillance and reconnaissance and command and control massive amounts of information sharing by providing "bandwidth-available" environment. Through GIG-BE, DISA leveraged DOD's existing end-to-end information transport capabilities, significantly expanding capacity and reliability to select Joint Staff-approved locations worldwide and under new hardware and software contracts to build a communications infrastructure. The GIG-BE that is intended to provide high-capacity communications linking DoD users at locations worldwide is a ground-based optical network with up to 10-Gbps connections and averaging 105 Gbps per link on the backbone networks. The GIG-BE program has greatly contributed to the development of the real-time Net-Centric Operations. However, a bottleneck link problem exists between core networks and edge networks due to the enormous difference in bandwidths.

The DoD supports NCO and GIG-BE projects to improve quality of services in net-centric environment. The current coding systems will not be appropriate in the near future. However, coding based scalable communication technology has not been applied to the Net-Centric Operations. This technology will satisfy the bandwidth requirements of tomorrow's warfighters.

Network coding is a recent development in which nodes in the network are allowed to perform algebraic operations inside the network. This scheme was first introduced in [1] and a powerful algebraic framework, which allows further developments, was provided in [2], [3]. For multicast settings, it was shown in [4], [5] that network coding performed in a distributed, random fashion is with high probability optimal. A tutorial on network coding can be found in [6], [7].

The specifics of the Scalable Information Operations (SIO) include: 1. scalable coding techniques for network coding, compression, channel coding, multimedia data transmission, encryption, data sharing, data anonymization, meta database management, caching, network security, and intrusion detection. 2. Bottleneck flow control. The purpose of this paper is to overview some of the recent developments in applying network coding to security in the areas of detection and correction of Byzantine attacks, and of cryptography for network coding based file downloads. The aim of this paper is mainly tutorial and further technical details can be found in [8], [9]. Especially, our goal is to sketch how network-coding based scalable information operations will mitigate some of the security issues in the future net-centric environment

II. Network-coding based detection and correction of Byzantine attackers

The mixture of data that occurs in network coding can lead to pollution attacks through rogue, or Byzantine, nodes in the network [10], [11]. Such nodes may be unreliable through failure or because of their being compromised. While the use of network coding would at first appear to render the problem of Byzantine attackers worse, it actually provides some strong protection for both the detection and correction of such nodes.

The results in this section have previously appeared in more detailed form in [8]. We consider network error correction in a distributed packet network setting with random linear network coding using coding vectors. A batch of r packets is multicast from a source node s to a set of sink nodes. An omniscient adversary can arbitrarily corrupt the coding vector as well as the data symbols of up to z_o packets. A packet that is not a linear combination of its input packets is called *adversarial*.

We describe below a polynomial-complexity network error-correcting code whose parameters depend on z_o, the maximum number of adversarial packets, and m, the minimum source-sink cut capacity (maximum error-free multicast rate) in units of packets over the batch. The number of packets in the batch is set as $r = m - z_o$. The proportion of redundant symbols in each packet, denoted ρ, is set as $\rho = (z_o + \epsilon)/r$ for some $\epsilon > 0$.

1-4244-1513-06/07/$25.00 ©2007 IEEE

The corresponding information rate of the code approaches $m-2z_o$ asymptotically as the packet size increases. If instead of an omniscient adversary we assume that the source and sinks share a secret channel not observed by the adversary, a higher rate of $m-z_o$ is asymptotically achievable. Below we give the details of the code for the omniscient adversary case.

For $i=1,\ldots,r$, the ith source packet is represented as a length-n row vector $\mathbf{x}_i$ with entries in a finite field $\mathbb{F}_q$. The first $n-\rho n-r$ entries of the vector are independent exogenous data symbols, the next ρn are redundant symbols, and the last r symbols form the packet's coding vector (the unit vector with a single nonzero entry in the ith position). We denote by $\mathbf{X}$ the $r\times n$ matrix whose ith row is $\mathbf{x}_i$; it can be written in the block form $\begin{bmatrix}\mathbf{U} & \mathbf{R} & \mathbf{I}\end{bmatrix}$ where $\mathbf{U}$ denotes the $r\times(n-\rho n-r)$ matrix of exogenous data symbols, $\mathbf{R}$ denotes the $r\times\rho n$ matrix of redundant symbols and $\mathbf{I}$ is the $r\times r$ identity matrix.

The $r\rho n$ redundant symbols are obtained as follows. For any matrix $\mathbf{M}$, let $\mathbf{v}_\mathbf{M}^T$ denote the column vector obtained by stacking the columns of $\mathbf{M}$ one above the other, and $\mathbf{v}_\mathbf{M}$ its transpose, a row vector. Matrix $\mathbf{X}$, represented in column vector form, is given by $\mathbf{v}_\mathbf{X}^T=[\mathbf{v}_\mathbf{U},\mathbf{v}_\mathbf{R},\mathbf{v}_\mathbf{I}]^T$. Let $\mathbf{D}$ be an $r\rho n\times rn$ matrix obtained by choosing each entry independently and uniformly at random from $\mathbb{F}_q$. The redundant symbols constituting $\mathbf{v}_\mathbf{R}$ (or $\mathbf{R}$) are obtained by solving the matrix equation

$$\mathbf{D}[\mathbf{v}_\mathbf{U},\mathbf{v}_\mathbf{R},\mathbf{v}_\mathbf{I}]^T=\mathbf{0} \tag{1}$$

for $\mathbf{v}_\mathbf{R}$. The value of $\mathbf{D}$ is known to all parties.

An adversarial packet can be viewed as an additional source packet. The vector representing the ith adversarial packet is denoted $\mathbf{z}_i$. Let $\mathbf{Z}$ denote the matrix whose ith row is $\mathbf{z}_i$.

We focus on any one of the sink nodes t. Let w be the number of linearly independent packets received by t; let $\mathbf{Y}\in\mathbb{F}_q^{w\times n}$ denote the matrix whose ith row is the vector representing the ith of these packets. Since all coding operations in the network are scalar linear operations in $\mathbb{F}_q$, $\mathbf{Y}$ can be be expressed as

$$\mathbf{Y}=\mathbf{G}\mathbf{X}+\mathbf{K}\mathbf{Z} \tag{2}$$

where matrices $\mathbf{G}\in\mathbb{F}_q^{w\times r}$ and $\mathbf{K}\in\mathbb{F}_q^{w\times z}$ represent the linear mappings from the source and adversarial packets respectively to the sink's set of linearly independent input packets.

Since the last r columns of $\mathbf{X}$ form an identity matrix, the matrix $\mathbf{G}'$ formed by the last r columns of $\mathbf{Y}$ is given by

$$\mathbf{G}'=\mathbf{G}+\mathbf{K}\mathbf{L}, \tag{3}$$

where $\mathbf{L}$ is the matrix formed by the last r columns of $\mathbf{Z}$. The sink knows $\mathbf{G}'$ but not $\mathbf{G}$. Thus, we rewrite (2) as

$$\begin{aligned}\mathbf{Y} &= \mathbf{G}'\mathbf{X}+\mathbf{K}(\mathbf{Z}-\mathbf{L}\mathbf{X})\\ &= \mathbf{G}'\mathbf{X}+\mathbf{E}\end{aligned} \tag{4}$$

Matrix $\mathbf{E}$ gives the difference between the data values in the received packets and the data values corresponding to their coding vectors; its last r columns are all zero.

Lemma 1: With probability at least $1-\eta/q$, the matrix $\mathbf{G}'$ has full column rank, where η is the number of links in the network.

The decoding process at sink t is as follows. First, the sink determines z, the minimum cut from the adversarial packets to the sink. This is with high probability equal to $w-r$. Next, it chooses z columns of $\mathbf{Y}$ that, together with the columns of $\mathbf{G}'$, form a basis for the column space of $\mathbf{Y}$. We assume without loss of generality that the first z columns are chosen, and we denote the corresponding submatrix $\mathbf{G}''$. Rewriting $\mathbf{Y}$ in the basis corresponding to the matrix $[\mathbf{G}''\ \mathbf{G}']$, we have

$$\mathbf{Y}=[\mathbf{G}''\ \mathbf{G}']\begin{bmatrix}\mathbf{I}_z & \mathbf{Y}^Z & \mathbf{0}\\ \mathbf{0} & \mathbf{Y}^X & \mathbf{I}_r\end{bmatrix} \tag{5}$$

This can be reduced by linear algebraic manipulations to

$$\mathbf{G}'\mathbf{X}_2=\mathbf{G}'(\mathbf{Y}^X+\mathbf{X}_1\mathbf{Y}^Z) \tag{6}$$

where $\mathbf{X}_1,\mathbf{X}_2$ are the matrices formed by the first z columns of $\mathbf{X}$ and the next $n-z-r$ columns of $\mathbf{X}$ respectively.

Proposition 1: With probability greater than $1-q^{n\epsilon}$, equations (1) and (6) can be solved simultaneously to recover $\mathbf{X}$. The decoding algorithm has complexity $O(n^3m^3)$.

III. Cryptography for Content Distribution with Network Coding

A. Background

Recently, several researchers explored the use of network coding in peer-to-peer (P2P) content distribution and distributed storage systems [12], [13], [14]. A P2P network has a fully distributed architecture, and the peers in the network form a cooperative network that shares the resources, such as storage, CPU, and bandwidth, of all the computers in the network. This architecture offers a cost-effective and scalable way to distribute software updates, videos, and other large files to a large number of users.

The best example of a P2P cooperative architecture is the BitTorrent system [15], which splits large files into small blocks, and after a node downloads a block from the original server or from another peer, it becomes a server for that particular block. Although BitTorrent has become extremely popular for distribution of large files over the Internet, it may suffer from a number of inefficiencies which decrease its overall performance. For example, scheduling is a key problem in BitTorrent: it is difficult to efficiently select which block(s) to download first and from where. If a rare block is only found on peers with slow connections, this would create a bottleneck for all the downloaders. Several *ad hoc* strategies are used in BitTorrent to ensure that different blocks are equally spread in the system as the system evolves. References [12], [13] propose the use of network coding to increase the efficiency of content distribution in a P2P cooperative architecture. The main idea of this approach is the following. The server breaks the file to be distributed into small blocks, and whenever a peer requests a file, the server sends a random linear combination of all the blocks. As in BitTorrent, a peer acts as a server to the blocks it has obtained. However, in a

linear coding scheme, any output from a peer node is also a random linear combination of all the blocks it has already received. A peer node can reconstruct the whole file when it has received enough degrees of freedom to decode all the blocks. This scheme is completely distributed, and eliminates the need for a scheduler, as any block transmitted contains partial information of all the blocks that the sender possesses. It has been shown both mathematically [12] and through live trials [16] that the random linear coding scheme significantly reduces the downloading time and improves the robustness of the system.

A major concern for any network coding system is the protection against malicious nodes. Take the above content distribution system for example. If a node in the P2P network behaves maliciously, it can create a polluted block with valid coding coefficients, and then sends it out. Here, coding coefficients refer to the random linear coefficients used to generate this block. If there is no mechanism for a peer to check the integrity of a received block, a receiver of this polluted block would not be able to decode anything for the file at all, even if all the other blocks it has received are valid. To make things worse, the receiver would mix this polluted block with other blocks and send them out to other peers, and the pollution can quickly propagate to the whole network. This makes coding based content distribution even more vulnerable than the traditional P2P networks, and several attempts were made to address this problem. References [12], [17] proposed to use homomorphic hash functions in content distribution systems to detect polluted packets, and [18] suggested the use of a Secure Random Checksum (SRC) which requires less computation than the homomorphic hash function. However, [18] requires a secure channel to transmit the SRCs to all the nodes in the network. Charles *et al* [19] proposed a signature scheme based on Weil pairing on elliptic curves and provides authentication of the data in addition to pollution detection, but the computation complexity of this solution is quite high. Moreover, the security offered by elliptic curves that admit Weil pairing is still a topic of debate in the scientific community.

In this section, we overview a new signature scheme, presented in greater detail in [9], that is not based on elliptic curves, and is designed specifically for random linear coded systems. We view all blocks of the file as vectors, and make use of the fact that all valid vectors transmitted in the network should belong to the subspace spanned by the original set of vectors from the file. We present a signature that can be used to easily check the membership of a received vector in the given subspace, and at the same time, it is hard for a node to generate a vector that is not in that subspace but passes the signature test. We show that this signature scheme is secure, and that the overhead for the scheme is negligible for large files.

B. Problem Setup

We model the network by a directed graph $G_d = (N, A)$, where N is the set of nodes, and A is the set of communication links. A source node $s \in N$ wishes to send a large file, of size M, to a set of client nodes, $T \subset N$, and we refer to all the clients as *peers*. The large file is divided into m blocks, and any peer receives different blocks from the source node or from other peers. In this framework, a peer is also a server to blocks it has downloaded, and always sends out random linear combinations of all the blocks it has obtained so far to other peers. When a peer has received enough degrees of freedom to decode the data, i.e., it has received m linearly independent blocks, it can re-construct the whole file.

Specifically, we view the m blocks of the file, $\bar{\mathbf{v}}_1, ..., \bar{\mathbf{v}}_m$, as elements in n-dimensional vector space $\mathbf{F}_p^n$, where p is a prime. The source node augments these vectors to create vectors $\mathbf{v}_1, ..., \mathbf{v}_m$, given by

$$\mathbf{v}_i = (0, ..., 1, ..., 0, \bar{v}_{i1}, ..., \bar{v}_{in}),$$

where the first m elements are zero except that the ith one is 1, and $\bar{v}_{ij} \in \mathbf{F}_p$ is the jth element in $\bar{\mathbf{v}}_i$. Packets received by the peers are linear combinations of the augmented vectors,

$$\mathbf{w} = \sum_{i=1}^{m} \beta_i \mathbf{v}_i,$$

where β_i is the weight of $\mathbf{v}_i$ in $\mathbf{w}$. We see that the additional m elements in the front of the augmented vector keep track of the code vector, β, of the corresponding packet.

As mentioned in the previous subsection, this kind of network coding scheme is vulnerable to pollution attacks by malicious nodes. Unlike uncoded systems where the source knows all the blocks being transmitted in the network, and therefore, can sign each one of them, in a coded system, each peer produces "new" packets, and standard digital signature schemes do not apply here. In the next subsection, we introduce a novel signature scheme for the coded system.

C. Signature scheme for network coding

We note that the vectors $\mathbf{v}_1, ..., \mathbf{v}_m$ span a subspace V of $\mathbf{F}_p^{m+n}$, and a received vector $\mathbf{w}$ is a valid linear combination of vectors $\mathbf{v}_1, ..., \mathbf{v}_m$ if and only if it belongs to the subspace V. This is the key observation for our signature scheme. In the scheme described below, we present a system that is based upon standard modulo arithmetic (in particular the hardness of the Discrete Logarithm problem) and upon an invariant signature $\sigma(V)$ for the linear span V. Each node verifies the integrity of a received vector $\mathbf{w}$ by checking the membership of $\mathbf{w}$ in V based on the signature $\sigma(V)$.

Our signature scheme is defined by the following ingredients, which are independent of the file(s) to be distributed:

- q: a large prime number such that p is a divisor of $q-1$. Note that standard techniques, such as that used in Digital Signature Algorithm (DSA), apply to find such q.
- g: a generator of the group G of order p in $\mathbf{F}_q$. Since the order of the multiplicative group $\mathbf{F}_q^*$ is $q-1$, which is a multiple of p, we can always find a subgroup, G, with order p in $\mathbf{F}_q^*$.
- Private key: $\mathbf{K}_{pr} = \{\alpha_i\}_{i=1,...,m+n}$, a random set of elements in $\mathbf{F}_p^*$. $\mathbf{K}_{pr}$ is only known to the source.

- Public key: $\mathbf{K}_{pu} = \{h_i = g^{\alpha_i}\}_{i=1,...,m+n}$. $\mathbf{K}_{pu}$ is signed by some standard signature scheme, e.g., DSA, and published by the source.

To distribute a file in a secure manner, the signature scheme works as follows.

1) Using the vectors $\mathbf{v}_1, ..., \mathbf{v}_m$ from the file, the source finds a vector $\mathbf{u} = (u_1, ..., u_{m+n}) \in \mathbb{F}_p^{m+n}$ orthogonal to all vectors in V. Specifically, the source finds a non-zero solution, $\mathbf{u}$, to the equations
$$\mathbf{v}_i \cdot \mathbf{u} = 0, \quad i = 1, ..., m.$$
2) The source computes vector $\mathbf{x} = (u_1/\alpha_1, u_2/\alpha_2, ..., u_{m+n}/\alpha_{m+n})$.
3) The source signs $\mathbf{x}$ with some standard signature scheme and publishes $\mathbf{x}$. We refer to the vector $\mathbf{x}$ as the signature, $\sigma(V)$, of the file being distributed.
4) The client node verifies that $\mathbf{x}$ is signed by the source.
5) When a node receives a vector $\mathbf{w}$ and wants to verify that $\mathbf{w}$ is in V, it computes
$$d = \prod_{i=1}^{m+n} h_i^{x_i w_i},$$
and verifies that $d = 1$.

To see that d is equal to 1 for any valid $\mathbf{w}$, we have
$$d = \prod_{i=1}^{m+n} h_i^{x_i w_i} = g^{\sum_{i=1}^{m+n}(u_i w_i)} = 1$$
where the last equality comes from the fact that $\mathbf{u}$ is orthogonal to all vectors in V.

Next, we show that the system described above is secure. In essence, the theorem below shows that given a set of vectors that satisfy the signature verification criterion, it is provably as hard as the Discrete Logarithm problem to find new vectors that also satisfy the verification criterion other than those that are in the linear span of the vectors already known.

Definition 1. Let p be a prime number and G be a multiplicative cyclic group of order p. Let k and n be two integers such that $k < n$, and $\mathbf{\Gamma} = \{h_1, ..., h_n\}$ be a set of generators of G. Given a linear subspace, V, of rank k in $\mathbb{F}_p^n$ such that for every $\mathbf{v} \in V$, the equality $\mathbf{\Gamma}^{\mathbf{v}} \triangleq \prod_{i=1}^{n} h_i^{v_i} = 1$ holds, we define the (p, k, n)-Diffie-Hellman problem as the problem of finding a vector $\mathbf{w} \in \mathbb{F}_p^n$ with $\mathbf{\Gamma}^{\mathbf{w}} = 1$ but $\mathbf{w} \notin V$.

By this definition, the problem of finding an invalid vector that satisfies our signature verification criterion is a $(p, m, m+n)$-Diffie-Hellman problem.

Theorem 1. For any $k < n - 1$, the (p, k, n)-Diffie-Hellman problem is as hard as the Discrete Logarithm problem.

Proof: Assume that we have an efficient algorithm to solve the (p, k, n)-Diffie-Hellman problem, and we wish to compute the discrete algorithm $\log_g(z)$ for some $z = g^x$, where g is a generator of a cyclic group G with order p. We can choose two random vectors $\mathbf{r} = (r_1, ..., r_n)$ and $\mathbf{s} = (s_1, ..., s_n)$ in $\mathbb{F}_p^n$, and construct $\mathbf{\Gamma} = \{h_1, ..., h_n\}$, where $h_i = z^{r_i} g^{s_i}$ for $i = 1, ..., n$. We then find k linearly independent (and otherwise random) solution vectors $\mathbf{v}_1, ..., \mathbf{v}_k$ to the equations
$$\mathbf{v} \cdot \mathbf{r} = 0 \text{ and } \mathbf{v} \cdot \mathbf{s} = 0.$$
Note that there exist $n-2$ linearly independent solutions to the above equations. Let V be the linear span of $\{\mathbf{v}_1, ..., \mathbf{v}_k\}$, it is clear that any vector $\mathbf{v} \in V$ satisfies $\mathbf{\Gamma}^{\mathbf{v}} = 1$. Now, if we have an algorithm for the (p, k, n)-Diffie-Hellman problem, we can find a vector $\mathbf{w} \notin V$ such that $\mathbf{\Gamma}^{\mathbf{w}} = 1$. This vector would satisfy $\mathbf{w} \cdot (x\mathbf{r} + \mathbf{s}) = 0$. Since $\mathbf{r}$ is statistically independent from $(x\mathbf{r} + \mathbf{s})$, with probability greater than $1 - 1/p$, we have $\mathbf{w} \cdot \mathbf{r} \neq 0$. In this case, we can compute
$$\log_g(z) = x = \frac{\mathbf{w} \cdot \mathbf{s}}{\mathbf{w} \cdot \mathbf{r}}.$$
This means the ability to solve the (p, k, n)-Diffie-Hellman problem implies the ability to solve the Discrete Logarithm problem. ■

This proof is an adaptation of a proof that appeared in an earlier publication by Boneh *et. al* [20].

D. Discussion

Our signature scheme nicely makes use of the linearity property of random linear network coding, and enables the peers to check the integrity of packets without the requirement for a secure channel. Also, the computation involved in the signature generation and verification processes is very simple.

Next, we examine the overhead incurred by this signature scheme. The size of each file block is $B = n \log(p)$ and we have $M = mn \log(p)$. The size of each augmented vector (with coding vectors in the front) is $B_a = (m + n) \log(p)$, and thus, the overhead of the coding vector is m/n times the file size. Note that this is the overhead pertaining to the linear coding scheme, not to our signature scheme, and any practical network coding system would make $m \ll n$. The initial setup of our signature scheme involves the publishing of the public key, $\mathbf{K}_{pu}$, which has size $(m + n) \log(q)$. In typical cryptographic applications, the size of p is 20 bytes (160 bits), and the size of q is 128 bytes (1024 bits), thus, the size of $\mathbf{K}_{pu}$ is approximately equal to $6(m + n)/mn$ times the file size.

For distribution of each file, the incremental overhead of our scheme consists of two parts: the public data, $\mathbf{K}_{pu}$, and the signature vector, $\mathbf{x}$.

For the public key, $\mathbf{K}_{pu}$, we note that it cannot be fully reused for multiple files, as it is possible for a malicious node to generate a invalid vector that satisfies the check $d = 1$ using information obtained from previously downloaded files. To prevent this from happening, we can publish a new public key for each file, and as mentioned above, the overhead is about $6(m + n)/mn$ times the file size, which is small as long as $6 \ll m \ll n$.

Alternatively, for every new file, we can randomly pick an integer i between 1 and $m + n$, select a new random value for α_i in the private key, and just publish the new $h_i = g^{\alpha_i}$. The overhead for this method is only $6/mn$ times the file

size. As an example, if we have a file of size 10MB, divided into $m = 100$ blocks, the value of n would be in the order of thousands, and thus, this overhead is less than 0.01% of the file size. This method should provide good security except in the case where we expect the vector $\mathbf{w}$ to have low variability, for example, has many zeros. Security can be increased by changing more elements in the private key for each new file.

In addition, for each new file distributed, we also have to publish a new signature $\mathbf{x}$, which is computed from a vector $\mathbf{u}$ that is orthogonal to the subspace V spanned by the file. Since the V has dimension m, it is sufficient to only replace m elements in $\mathbf{u}$ to generate a vector orthogonal to the new file. Since the first m elements in the vectors $\mathbf{v}_1, ..., \mathbf{v}_m$ are always linearly independent (they are the code vectors), it suffices to just modify the entries u_1 to u_m. Assume that the ith element in the private key is the only one that has been changed for the distribution of the new file, and that i is between 1 and m, then we only need to publish x_1 to x_m for the new signature vector. This part of the overhead has size $m \log(p)$, and the ratio between this overhead and the original file size N is $1/n$. Again, take a 10MB file for example, this overhead is less than 0.1% of the file size.

Therefore, after the initial setup, each additional file distributed only incurs a negligible amount of overhead using our signature scheme.

Finally, we would like to point out that, under our assumptions that there is no secure side channel from the source to all the peers and that the public key is available to all the peers, our signature scheme has to be used on the original file vectors not on hash functions. This is because to maintain the security of the system, we need to use a one-way hash function that is homomorphic, however, we are not aware of any such hash function. Although [12] and [17] suggested usage of homomorphic hash functions for network coding, [12] assumed that the intermediate nodes do not know the parameters used for generating the hash function, and [17] assumed that a secure channel is available to transmit the hash values of all the blocks from the source node to the peers. Under our more relaxed assumptions, these hash functions would not work.

IV. Conclusions

In this paper, we have overviewed some of the security capabilities of network coding, particularly in the area of robustness to Byzantine attacks and to distributed authentication in peer-to-peer downloads. The implications of network coding for security are not limited to these applications. For instance, network coding's mixture of data can be used to use data for effective countermeasures to eavesdropping. In effect, data is used, after compression, as a one-time-pad in the system, [21], [22], [23], [24]. None of these techniques or the ones summarized in this paper present in themselves a complete security solution, and we have not attempted to implement any of our security techniques. However, as network coding opens entirely new venues for the operation of networks, we expect to see security challenges inherited from traditional forms of networking, the mitigation of current problems but also the emergence of new classes of data sharing problems in Net-Centric environment. We will further develop scalable and secure network coding techniques to solve multimedia delivery and massive data sharing problems in Airborne/UAV networks.

References

[1] R. Ahlswede, N. Cai, S. Li, and R. Yeung, "Network information flow: Single source," *submitted to IEEE Transactions on Information Theory*, 1999.

[2] R. Koetter and M. Médard, "An algebraic approach to network coding," in *IEEE International Symposium on Information Theory (ISIT)*, vol. 1, p. 104, 2001.

[3] R. Koetter and M. Médard, "An algebraic approach to network coding," *IEEE/ACM Transactions on Networking (selected as one of the outstanding papers from INFOCOM for transfer to IEEE/ACM Transaction on Networking)*, vol. 11, pp. 782–796, October.

[4] T. Ho, M. Médard, R. Koetter, D. R. Karger, and M. Effros, "The benefits of coding over routing in a randomized setting," in *IEEE International Symposium on Information Theory (ISIT)*, p. 442, 2003.

[5] T. Ho, M. Mdard, R. Koetter, D. R. Karger, M. Effros, J. Shi, and B. Leong, "A random linear network coding approach to multicast," in *IEEE Transactions on Information Theory*, vol. 52, pp. 4413–4430, October 2006.

[6] M. Effros, R. Koetter, and M. Médard, "Breaking network logjams," *Scientific American*, vol. 6, pp. 78–85, June.

[7] C. Fragouli, J.-Y. L. Boudec, and J. Widmer, "Network coding: an instant primer," *SIGCOMM Comput. Commun. Rev.*, vol. 36, no. 1, pp. 63–68, 2006.

[8] S. Jaggi, M. Langberg, S. Katti, T. Ho, D. Katabi, and M. Médard, "Resilient network coding in the presence of byzantine adversaries," in *Proc. IEEE INFOCOM 2007*, (Anchorage, AK), May 2007.

[9] F. Zhao, T. Kalker, M. Médard, and K. Han, "Signatures for content distribution with network coding," in *Proc. of IEEE ISIT'07*, (Nice, France), July 2007.

[10] L. Lamport, R. Shostak, and M. Pease, "The byzantine generals problem," *ACM Trans. Program. Lang. Syst.*, vol. 4, no. 3, pp. 382–401, 1982.

[11] R. Perlman, "Network layer protocols with byzantine robustness," *MIT Ph.D. Thesis*, 1988.

[12] S. Acedański, S. Deb, M. Médard, and R. Koetter, "How good is random linear coding based distributed network storage?," in *Proc. 1st Workshop on Network Coding, Theory, and Applications (Netcod'05)*, (Riva del Garda, Italy), Apr. 2005.

[13] C. Gkantsidis and P. Rodriguez, "Network coding for large scale content distribution," in *Proc. IEEE INFOCOM'05*, (Miami, FL), Mar. 2005.

[14] A. G. Dimakis, P. B. Godfrey, M. Wainwright, and K. Ramchandran, "Network coding for distributed storage systems," in *Proc. of IEEE INFOCOM'07*, (Anchorage, Alaska), Mar. 2007.

[15] "Bittorrent file sharing protocol. http://www.bittorrent.com,"

[16] C. Gkantsidis, J. Miller, and P. Rodriguez, "Comprehensive view of a live network coding p2p system," in *Proc. ACM SIGCOMM/USENIX Internet Measurement Conference (IMC'06)*, (Rio de Janeiro, Brazil), Oct. 2006.

[17] C. Gkantsidis and P. Rodriguez, "Cooperative security for network coding file distribution," in *Proc. of IEEE INFOCOM'06*, (Barcelona, Spain), Apr. 2006.

[18] M. N. Krohn, M. J. Freedman, and D. Mazières, "On-the-fly verification of rateless erasure codes for efficient content distribution," in *Proc. of the IEEE Symposium on Security and Privacy*, (Oakland, CA), May 2004.

[19] D. Charles, K. Jain, and K. Lauter, "Signatures for network coding," in *Proc. of Conference on Information Sciences and Systems (CISS'06)*, (Princeton, NJ), Mar. 2006.

[20] D. Boneh and M. Franklin, "An efficient public key traitor tracing scheme," in *Proc. of Crypto'99, Lecture Notes in Computer Science*, 1999.

[21] N. Cai and R. W. Yeung, "Secure network coding," in *IEEE International Symposium on Information Theory*, July 2002.

[22] K. Bhattad and K. Narayanan, "Weakly secure network coding," in *Proc. of the First Workshop on Network Coding, Theory, and Applications (NetCod)*, 2005.

[23] J. Tan and M. Médard, "Secure network coding with a cost criterion," in *Proc. of the Second Workshop on Network Coding, Theory, and Applications (NetCod)*, 2006.
[24] L. Lima, M. Médard, and J. Barros, "Random network coding: A free cypher?," in *IEEE International Symposium on Information Theory (ISIT)*, 2007.

This full text paper was peer reviewed at the direction of IEEE Communications Society subject matter experts for publication in the IEEE INFOCOM 2007 proceedings.

Resilient Network Coding in the Presence of Byzantine Adversaries

S. Jaggi jaggi@mit.edu
M. Langberg mikel@caltech.edu
S. Katti skatti@mit.edu
T. Ho tho@caltech.edu
D. Katabi dk@mit.edu
M. Médard medard@mit.edu

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 *the first 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 snooping 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; non-malicious 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.

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], [19], [13]. It can be done in a distributed manner with low complexity, and is robust to packet losses and network failures [8], [23]. Furthermore, recent implementations of network coding for wired and wireless environments demonstrate its practical benefits [16], [6].

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 [10], others correct the errors he injects into the codes under specific conditions [7], [12], [20], and a few bound the maximum achievable rate in such adverse environments [3], [29]. But attaining optimal rates using distributed and low-complexity codes is still an open problem.

This paper designs distributed polynomial-time rate-optimal network codes that combat Byzantine adversaries. 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 redundancy to ensure that certain functions evaluate to zero on the original source's data, and thus can be used to distill the source's data from the adversary's. The challenge addressed herein is to design the redundancy that achieves the optimal rates.

This paper makes several contributions. The algorithms presented herein are *the first 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 destination while the internal nodes can continue to use standard network coding.

II. Illustrating Example

We illustrate the intuition underlying our approach using the toy example in Fig. 1. Calvin wants to prevent the flow of information from Alice to Bob, or at least minimize it. All links have a capacity of one packet per unit time. Further, Calvin connects to the three routers through an intermediate node. The intermediate node just relays all the packets Calvin sends him to the three routers. The network capacity, C, is by definition the min-cut from Alice to Bob. It is equal to 3 packets per unit time. The min-cut from Calvin to the destination is $z_O = 1$ packet per unit time. Hence, the maximum rate from Alice to Bob in this scenario is bounded by $C - z_O = 2$ packets per unit time as proven in [12].

0743-166X/07/$25.00 ©2007 IEEE

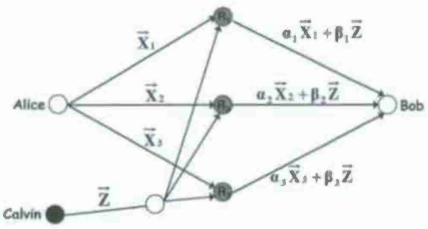


Fig. 1—A simple example. Alice transmits to Bob. Calvin injects corrupted packets into their communication. The grey nodes in the middle perform network coding.

We express each packet as a vector of n bytes, where n is a sufficiently large number. The routers create random linear combinations of the packets they receive. Hence, every unit of time Bob receives the packets:

$$\tilde{\mathbf{y}}_1 \;=\; \alpha_i\tilde{\mathbf{x}}_1 + \beta_i\tilde{\mathbf{z}},\ i \in \{1,2,3\}, \tag{1}$$

where $\tilde{\mathbf{x}}_1$'s are vectors representing the three packets Alice sent, $\tilde{\mathbf{z}}$ is the packet Calvin sent, α_i and β_i are random coefficients.

In our example, the routers operate over bytes; the i^{th} byte in an outgoing packet is a linear combination of i^{th} bytes in the incoming packets. Thus, (1) also describes the relation between the individual bytes in $\tilde{\mathbf{y}}_1$'s and the corresponding bytes in $\tilde{\mathbf{x}}_1$'s and $\tilde{\mathbf{z}}$.

Since the routers mix the content of the packets, Alice cannot just sign her packets and have Bob discard all packets with incorrect signatures. To decode, Bob has to somehow distill the $\tilde{\mathbf{x}}_1$'s from the $\tilde{\mathbf{y}}_1$'s he receives.

As a first attempt at solving the problem, let us assume that Bob knows the topology, i.e., he knows that the packets he receives are produced using (1). Further, let us assume that he knows the random coefficients used by the routers to code the packets, i.e., he knows the values of α_i's and β_i's. To decode, Bob has to solve (1). Since each packet contains n bytes, the system in (1) represents $3n$ equations, one equation per received byte. Bob has $3n$ equations and $4n$ unknowns (n unknown bytes per each packet $\tilde{\mathbf{z}}$, $\tilde{\mathbf{x}}_1$, $\tilde{\mathbf{x}}_2$ and $\tilde{\mathbf{x}}_3$). Hence, Bob cannot decode.

To address the above situation, Alice needs to add redundancy to her transmitted packets. After all, as noted above, for the particular example in Fig. 1, Alice's rate is bounded by 2 packets per unit time. Thus, Alice should send no more than 2 packets worth of information. She can use the third packet for added redundancy. Suppose Alice sets

$$\tilde{\mathbf{x}}_3 = \tilde{\mathbf{x}}_1 + \tilde{\mathbf{x}}_2. \tag{2}$$

This coding strategy is public to both Bob and Calvin. Since each packet contains n bytes, combining (2) with (1), Bob obtains a system of $4n$ equations with $4n$ unknowns, which he can solve to decode.

But in the general case, Bob knows nothing about the coefficients used by the routers, the topology, or the overall network transform. Said differently, the 6 coefficients corresponding to the α_i's and the β_i's are usually unknown to Bob. Thus, given (1) and (2), Bob is faced with $4n$ equations and $4n+6$ unknowns, and thus cannot decode. The matter is further complicated by the non-linearity of (1), which involves the product of unknown terms $\alpha_i\tilde{\mathbf{x}}_1$ and $\beta_i\tilde{\mathbf{z}}$.

The first idea we exploit in our solution is that while $\tilde{\mathbf{z}}$ is a whole unknown packet of n bytes, each of the coefficients β_i is a single byte. Thus, instead of devoting a whole vector of n bytes for added redundancy (as in (2)), Alice just needs to introduce 6 extra bytes of redundancy to compensate for the α_i's and β_i's being unknown.

Alice imposes constraints on her data to help Bob to decode. For instance, a simple constraint could be that the first byte in each packet equals zero. This constraint provides Bob with 2 additional equations (recall that the first byte in $\tilde{\mathbf{x}}_3$ is forced to 0 due to (2), and hence the new constraint produces 2 additional equations rather than 3). Rewriting (1) for the first byte of each packet, we obtain:

$$y_{i,1} = \alpha_1 x_{i,1} + \beta_i z_1 = \beta_i z_1,\ ,i \in \{1,2,3\} \tag{3}$$

where $y_{i,j}$ denotes the j^{th} byte in the i^{th} received packet. The above equations provide Bob with a scaled version of the β_i's, i.e., they are all multiplied by z_1.

Our second observation is that the scaled version of the β_i's suffices for Bob to decode $\tilde{\mathbf{x}}$. This can be seen by a simple algebraic manipulation of (1). Bob can rewrite the equations in (1) by multiplying and dividing the second term with z_1 and appending (2) to obtain

$$\tilde{\mathbf{y}}_1 \;=\; \alpha_i\tilde{\mathbf{x}}_1 + (\beta_i z_1)(\tilde{\mathbf{z}}/z_1),\ i \in \{1,2,3\}\ . \tag{4}$$

Notice that Bob already knows all three $\beta_i z_1$ terms from (3). The term $(\tilde{\mathbf{z}}/z_1)$ can be considered a single unknown because Bob does not care about estimating the exact value of $\tilde{\mathbf{z}}$.

To allow Bob to discover the α_i's, Alice similarly adds 4 more bytes of redundancy by imposing constraints on the second and third bytes in her packets. For example, she chooses $x_{1,2} = x_{2,2} = 1$ and $x_{1,3} = -x_{2,3} = 1$ (combined with (2), these constraints force $x_{3,2} = 2$ and $x_{3,3} = 0$). Substituting the values of $(\beta_1 z_1)$, $(\beta_2 z_1)$ and $(\beta_3 z_1)$ from (3) gives Bob the following equations.

$$\begin{array}{ll} y_{1,2} = \alpha_1 + y_{1,1}(z_2/z_1), & y_{1,3} = \alpha_1 + y_{1,1}(z_3/z_1) \\ y_{2,2} = \alpha_2 + y_{2,1}(z_2/z_1), & y_{2,3} = -\alpha_2 + y_{2,1}(z_3/z_1)\ . \\ y_{3,2} = 2\alpha_3 + y_{3,1}(z_2/z_1), & y_{3,3} = y_{3,1}(z_3/z_1) \end{array} \tag{5}$$

Now Bob has 6 linear equations with the 5 unknowns α_1, α_2, α_3, z_2/z_1 and z_3/z_1, and they can be solved to obtain the α_i's. Hence we are essentially back to the situation where Bob knows the α_i's and β_i's, and can solve for $\tilde{\mathbf{x}}_1$'s.

One complication still remains. If Calvin knows the constraints on Alice's data, he will try to assign values to his bytes to prevent Bob from decoding. For example, if Calvin knows that the first byte of each of Alice's packets is zero, he too would set the first byte in his packet z_1 to zero, in which case Bob does not obtain any information about the β_i's from (4).

There are two ways out of this situation. Suppose Alice could communicate to Bob a small message that is secret from Calvin. In this case, she could compute a small number of hashes of her data, and transmit them to Bob. These hashes correspond to constraints on her data, which enables Bob to decode. If Alice cannot communicate secretly with Bob, she leverages the fact that Calvin can inject only one fake packet. Since Calvin's packet is n bytes long, he can cancel out at most n hashes. If Alice injects $n+1$ hashes, there must be at least one hash Calvin cannot cancel. This hash enables Bob to find the β_i's and decode. Notice, however, that the $n+1$ additional constraints imposed on the bytes in $\vec{x}_1$ and $\vec{x}_2$ mean that Alice

can only transmit at most $n-1$ bytes of data to Bob. For a large number of bytes n in a packet, this rate is asymptotically optimal against an all-knowing adversary [3].

After giving some intuition on how our scheme works, the rest of this paper considers the general problem of network coding over completely unknown topology, in the presence of an adversary who has partial or full knowledge of the network and transmissions in it.

III. Related Work

We start with a brief summary of network coding, followed by a survey of prior work on Byzantine adversaries in networks.

A. Network Coding Background

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 [21], [19], [13] shows that, for multicast traffic, linear codes achieve the maximum capacity bounds, and coding and decoding can be done in polynomial time. Additionally, Ho et al. show that the above is true even when the routers pick random coefficients [8]. Researchers have extended the above results to a variety of areas including wireless networks [23], [15], [16], energy [28], secrecy [2], content distribution [6], and distributed storage [14].

B. Byzantine Adversaries in Networks

A Byzantine attacker is a malicious adversary hidden in a network, capable of eavesdropping and jamming communications. Prior research has examined these 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 [26]. 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 [24].

The interplay of network coding and Byzantine adversaries was first examined in [10], which detects the existence of an adversary but does not provide an error-correction scheme. This has been followed by the work of Cai and Yeung [29], [3], who generalize 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 correcting errors in the presence of both network coding and Byzantine adversaries has been considered by a few prior proposals. Earlier work [20], [7] assumes a centralized trusted authority that provides hashes of the original packets to each node in the network. More recent work by 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. In contrast to the above two schemes which are cryptographically secure, in a previous work [12], 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. Recent work [17] has considered the same problem from a different perspective, their results and bounds are similar to ours. Table I highlights similarities and differences from prior work.

Scheme	Charles et.al. [4]	Jaggi et.al. [12]	Ours
Info. Theoretic Security	No	Yes	Yes
Distributed	Yes	No	Yes
Internal Node Complexity	High	Low	Low
Decoding Complexity	High	Exponential	Low
General Graphs	No	Yes	Yes
Universal	No	No	Yes

TABLE I—Comparison between the results in this paper and some prior papers.

IV. Model & 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 similar to most network coding algorithms, our techniques generalize to multicast traffic.

A. Threat Model

There is a source, Alice, and a destination, Bob, who communicate over a wired or wireless network. 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 of the transmissions, and can inject his own. When he injects his own packets, he pretends they are part of the information flow from Alice to Bob.

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

This section describes the network model, the packet format, and how the network transforms the packets.

Network Model: The network is modeled as a hypergraph [22]. Each packet transmission corresponds to 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, 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 [18]). 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. 2. We denote the $(i,j)^{th}$ element in the matrix by $x(i,j)$. The i^{th} row in the matrix X is just the i^{th} packet in the batch. Fig. 2 shows that similarly to standard network codes [8], some of the redundancy in the batch is devoted to sending the identity matrix, I. Also, as in [8], 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 between the packets it receives and those transmitted 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 size of the min-cut from Calvin to the destination.

Destination: Analogously to how Alice generates X, the destination Bob organizes the received packets into a matrix Y. The i^{th} received packet corresponds to the i^{th} 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. The column rank of Y, however, is $b + z_O$. Bob attempts to reconstruct Alice's information, X, using the matrix of received packets Y.

C. Definitions

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.)
- The *error probability* is the probability that Bob's reconstruction of Alice's information is inaccurate.
- The rate, R, is the number of information bits in a batch amortized by the length of a packet in bits.
- The rate R is said to be *achievable* if for any $\epsilon > 0$, any $\delta > 0$, and sufficiently large n, there exists a block-length-n network code with a redundancy δ and a probability of error less than ϵ.
- A code is said to be *universal* if the code design is independent of z_O.

V. Network Transform

This section explains how Alice's packets get transformed as they travel through the network. It examines the effect the adversary has on the received packets, and Bob's decoding problem.

The network performs a classical distributed network code [8]. Specifically, each packet transmitted by an internal node is a random linear combination of its incoming packets.

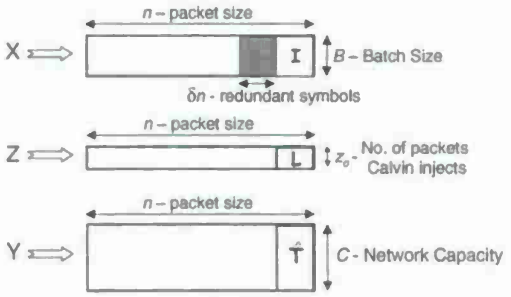


Fig. 2—Alice, Bob and Calvin's information matrices.

Variable	Definition
b	Number of packets in a batch.
z_O	Number of packets Calvin can inject.
z_I	Number of packets Calvin can hear.
n	Length of each packet.
δ	Fractional redundancy introduced by Alice.
$\hat{T}$	Proxy of the transfer matrix T representing the network transform.

TABLE II—Terms used in the paper.

Thus, the effect of the network at the destination can be summarized as follows.

$$Y = TX + T_{Z \to Y} Z, \tag{6}$$

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 variable T refers to the linear transform from Alice to Bob, while $T_{Z \to Y}$ refers to the linear transform from Calvin to Bob.

As explained in §IV, a classical random network code's X includes the identity matrix as part of each batch. The identity matrix sent by Alice incurs the same transform as the rest of the batch. Thus,

$$\hat{T} = TI + T_{Z \to Y} L, \tag{7}$$

where $\hat{T}$ and L are the columns corresponding to I's location in Y and Z respectively, as shown in Fig. 2.

In standard network coding, there is no adversary, i.e., $Z = 0$ and $L = 0$, and thus $\hat{T} = T$. The destination receives a description of the network transform in $\hat{T}$ and can decode X as $\hat{T}^{-1}Y$. In the presence of the adversary, however, the destination needs to solve (6) and (7) to extract the value of X.

By substituting T from (7), (6) can be simplified to get

$$\begin{aligned} Y &= \hat{T}X + T_{Z \to Y}(Z - LX) && (8) \\ &= \hat{T}X + E, && (9) \end{aligned}$$

where E is a $C \times n$ matrix that characterizes Calvin's interference. Note that the matrix $\hat{T}$, which Bob knows, acts as a *proxy transfer matrix* for T, which he doesn't know.

Note that in (6), all terms other than Y are unknown. Further, it is non-linear due to the cross-product terms, TX and $T_{Z \to Y} Z$. In contrast, (9) is linear in the unknowns X and E. The rest of this work focuses on solving (9) under different assumptions on Calvin's strength.

VI. 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 [3], [29], [12]. Our work, however, provides a construction of distributed codes/algorithms that achieve optimal rates. In what follows, let $|\mathcal{T}|$ denote the number of receivers, and $|\mathcal{E}|$ denote the number of transmissions in the network.

(1) Shared Secret Model: This model assumes that Alice and Bob have a very low rate secret channel, the transmissions on which are unknown to Calvin. It considers the transmission of information via network coding in a network where Calvin can observe all transmissions, and can inject some corrupt packets.

Theorem 1: The Shared Secret algorithm achieves a rate of $C - z_O$ with code-complexity $\mathcal{O}(nC^2)$. This is the maximum achievable rate.

In §VII, we prove the above theorem by constructing an algorithm that achieves the bounds. Note that [7] proves a similar result for a more constrained model where Alice shares a very low rate secret channel with all nodes in the network, and the operations performed by internal nodes are computationally expensive. Further, their result guarantees cryptographic security, while we provide information-theoretic security.

(2) Omniscient Adversary Model: This model assumes an omniscient adversary, i.e., one from whom nothing is hidden. In particular, Alice and Bob have no shared secrets hidden from Calvin. It also assumes that the min-cut from the adversary to the destination, z_O, is less than $C/2$. Prior work proves that without this condition, it is impossible for the source and the destination to reliably communicate without a secret channel [12]. In §VIII, we prove the following.

Theorem 2: The Omniscient Adversary algorithm achieves a rate of $C - 2z_O$ with code-complexity $\mathcal{O}((nC)^3)$. This is the maximum achievable rate.

(3) 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, but still achieves the higher rate possible via the Shared Secret algorithm. In particular, in §IX we prove the following.

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

VII. 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. 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 this might lead to Bob not receiving any uncorrupted packets. Alice uses the secret channel to send a hash of her information X to Bob, which Bob can use to distill the corrupted packets he receives, as explained below.

Shared Secret: Alice generates her secret message in two steps. She first chooses C *parity symbols* uniformly at random from the field $\mathbb{F}_q$. The parity symbols are labeled r_d, for $d \in \{1, \ldots, C\}$. Corresponding to the parity symbols, Alice's *parity-check matrix* P is defined as the $n \times C$ matrix whose $(i,j)^{th}$ entry equals $(r_j)^i$, i.e., r_j to the i^{th} power. The second part of Alice's secret message is the $b \times C$ *hash matrix* H, computed as the matrix product XP. We assume Alice communicates both the set of parity symbols and the hash matrix H to Bob over the secret channel. The combination of these two creates the shared secret, denoted S, between Alice and Bob. The size of S is $C(b+1)$ symbols, which is small in comparison to Alice's information X. (The size of X is $b \times n$; it can be made arbitrarily large compared to the size of S by increasing the packet size n.)

Alice's Encoder: Alice implements the classical random network encoder described in §IV-B.

Bob's Decoder: Not only is P used by Alice to generate H, but is also used by Bob in his decoding process. To be more precise, Bob computes $YP - \hat{T}H$ using the messages he gets from the network and the secret channel. We call the outcome the *syndrome matrix* S.

By substituting the value of H and using (9), we obtain

$$S = YP - \hat{T}H = (Y - \hat{T}X)P = EP. \tag{10}$$

Thus, if no adversary was present, the packets would not be corrupted (i.e., $E = 0$) and S would be an all-zero matrix. As shown in §IV, X then equals $\hat{T}^{-1}Y$. If Calvin injects corrupt packets, S will be a non-zero matrix.

Claim 1: The rank of E is at most z_O.

Claim 2: The columns of S span the same vector-space as the columns of E with probability at least $1 - Cn^C q^{-1}$.

Claim 1 follows from the definition of $E = T_{Z \to Y}(Z - LX)$. Claim 2 is proved in the Appendix. Together, they imply that Calvin's interference, E, can be written as linear combinations of the columns of a $C \times z_O$ submatrix S' of S, i.e., $E = S'A$, where A is a $z_O \times n$ matrix. This enables Bob to rewrite (9) as the matrix product

$$Y = [\hat{T}\ S'] \begin{bmatrix} X \\ A \end{bmatrix}, \tag{11}$$

Bob does not care about A, but to obtain X, he must solve (11). Let $|\mathcal{T}|$ and $|\mathcal{E}|$ be the number of terminals and links in the underlying network.

Claim 3: The matrix $[T\ T_{Z \to Y}]$, and thus the matrix $[\hat{T}\ S']$, has full column-rank with probability at least $1 - |\mathcal{T}||\mathcal{E}|q^{-1}$.

Claim 3, proved in the Appendix, means that Bob can decode by simply inverting the $C \times C$ matrix $[\hat{T}\ S']$ and multiplying the result by Y. Thus, the shared secret algorithm achieves the rate of $C - z_O - b^2/n$. Here, the asymptotically negligible term b^2/n corresponds to the overhead due to the identity matrix Alice appends to X. This rate is shown to be optimal by prior work [12]. The probability of error is at most the sums of the probabilities of error in Claims 2 and 3, i.e., $(n^C C + |\mathcal{T}||\mathcal{E}|)q^{-1}$. Of code design, encoding and decoding, both encoding and decoding require $\mathcal{O}(nC^2)$ steps. The costliest step for Alice is the computation of the hash matrix H, and for Bob is the computation of the syndrome matrix S.

The scheme presented above is universal, i.e., the parameters of the code do not depend on any knowledge about z_O, which in some sense functions as the "noise parameter" of the network. Alice therefore has flexibility in tailoring her batch size to the size of the data which she wishes to transmit and the packet size allowed by the network. □

VIII. Omniscient Adversary Model

What if we face an *omniscient adversary*, i.e., Calvin can observe everything, and there are no shared secrets between Alice and Bob? We design a network error-correcting code to defeat such a powerful adversary. Our algorithm achieves a rate of $R = C - 2z_O$, which is lower than in the Shared Secret model. This is a direct consequence of Calvin's increased strength. Recent bounds [3] on network error-correcting codes show that in fact $C - 2z_O$ is the maximum achievable rate for networks with an omniscient adversary.

Alice's Encoder: Alice encodes in two steps. To counter the adversary's interference, she first generates X by adding redundancy to her information. She then encodes X using the encoder defined in §IV-B.

Alice adds redundancy as follows. Her original information is a length-$(bn - \delta n - b^2)$ column vector $\tilde{\mathbf{U}}$. (Here the fractional redundancy δ, is dependent on z_O, the number of packets Calvin may inject into the network.) Alice converts $\tilde{\mathbf{U}}$ into $\tilde{\mathbf{X}}$, a length-bn vector $\begin{pmatrix}\tilde{\mathbf{U}} & \tilde{\mathbf{R}} & \tilde{\mathbf{I}}\end{pmatrix}^T$, where $\tilde{\mathbf{I}}$ is just the column version of the $b \times b$ identity matrix. It is generated by stacking columns of the identity matrix one after the other. The second term, $\tilde{\mathbf{R}}$ represents the redundancy Alice adds. The *redundancy vector* $\tilde{\mathbf{R}}$ is a length-δn column vector generated by solving the matrix equation for $\tilde{\mathbf{R}}$.

$$D\begin{pmatrix}\tilde{\mathbf{U}} & \tilde{\mathbf{R}} & \tilde{\mathbf{I}}\end{pmatrix}^T = 0.$$

where D is a $\delta n \times bn$ matrix defined as the *redundancy matrix*. D is obtained by choosing each element as an independent and uniformly random symbol from the finite field $\mathbb{F}_q$. Due to the dependence of D on δ and thus on z_O, the Omniscient Adversary algorithm is *not* universal. The redundancy matrix D is known to *all* parties – Alice, Bob, and Calvin – and hence does not constitute a shared secret.

Alice then proceeds to the standard network encoding. She rearranges $\tilde{\mathbf{X}}$, a length-bn vector, into the $b \times n$ matrix X. The j^{th} column of X consists of symbols from the $((j-1)b+1)^{th}$ through $(jb)^{th}$ symbols of $\tilde{\mathbf{X}}$. From this point on, Alice's encoder implements the classical random network encoder described in §IV-B, to generate her transmitted packets.

Bob's Decoder: As shown in (9), Bob's received data is related to Alice and Calvin's transmitted data as $Y = \hat{T}X + E$. Bob's objective, as in §VII, is to distill out the effect of the error matrix E and recover the vector X. He can then retrieve Alice's data by extracting the first $(bn - b^2 - \delta n)$ symbols to obtain $\tilde{\mathbf{U}}$.

To decode, Bob performs the following steps, each of which corresponds to an elementary matrix operation.

- *Determining Calvin's strength:* Bob first determines the strength of the adversary z_O, which is the column rank of $T_{Z\to Y}$. Bob does not know $T_{Z\to Y}$, but since T and $T_{Z\to Y}$ span disjoint vector spaces (Claim 3), the column rank of Y is equal to the sum of the column ranks of T and $T_{Z\to Y}$. Since the column rank of T is simply the batch size b, Bob determines z_O by subtracting b from the column rank of the matrix Y.
- *Discarding irrelevant information:* Since the classical random network code is run without any central coordinating authority, the packets of information that Bob receives may be highly redundant. Of the packets Bob receives, he selectively discards some so that the resulting matrix Y has $b + z_O$ rows, and has full row rank. For him to consider more packets is useless, since at most $b + z_O$ packets of information have been injected into the network, b from Alice and z_O from Calvin. This operation has the additional benefit of reducing the complexity of linear operations that Bob needs to perform henceforth. This reduces the dimensions of the matrix $\hat{T}$, since Bob can discard the rows corresponding to the discarded packets.
- *Estimating a "basis" for E:* If Bob could directly estimate a basis for the column space of E, then he could simply decode as in the Shared Secret algorithm. However, there is no shared secret that enables him to discover a basis for the column space of E. So, he instead chooses a *proxy error matrix* T'' whose columns (which are, in general, linear combinations of columns of both X and E) act as a *proxy error basis* for columns of E. This is analogous to step (9), where the matrix $\hat{T}$ acts as a proxy transfer matrix for the unknown matrix T.
  The matrix T'' is obtained as follows. Bob selects z_O columns from Y such that these columns, together with the b columns of $\hat{T}$, form a basis for the columns of Y. Without loss of generality, these columns correspond to the first z_O columns of Y (if not, Bob simply permutes the columns of Y to make it so). The $(b + z_O) \times z_O$ matrix corresponding to these first z_O columns is denoted T''.
- *Changing to proxy basis:* Bob rewrites Y in the basis corresponding to the columns of the $(b + z_O) \times (b + z_O)$ matrix $[T'' \ \hat{T}]$. Therefore Y can now be written as

$$Y = [T'' \ \hat{T}] \begin{bmatrix} I_{z_O} & F^Z & 0 \\ 0 & F^X & I_b \end{bmatrix}. \tag{12}$$

  Here $\begin{bmatrix} F^Z \\ F^X \end{bmatrix}$ is defined as the $(b + z_O) \times (n - (b + z_O))$ matrix representation of the columns of Y (other than those in $[T'' \ \hat{T}]$) in the new basis, with F^Z and F^X defined as the sub-matrices of appropriate dimensions.

Bob splits X as $X = [X_1 \ X_2 \ X_3]$, where X_1 corresponds to the first z_O columns of X, X_3 to the last b columns of X, and X_2 to the remaining columns of X. We perform linear algebraic manipulations on (12), to reduce it to a form in which the variables in X are related by a linear transform solely to quantities that are computable by Bob. Claim 4 summarizes the effect of these linear algebraic manipulations (proof in Appendix).

Claim 4: The matrix equation (12) is exactly equivalent to the matrix equation $\hat{T}X_2 = \hat{T}\left(F^X + X_1 F^Z\right)$.

To complete the proof of correctness of our algorithm, we need only the following claim, proved in the Appendix.

Claim 5: For $\delta n > n(z_O + \varepsilon)$, with probability greater than $1 - q^{-n\varepsilon}$, the system of linear equations

$$\hat{T}X_2 = \hat{T}\left(F^X + X_1 F^Z\right) \tag{13}$$

$$D\tilde{\mathbf{X}} = 0 \tag{14}$$

is solvable for X.

The final claim enables Bob to recover X, which contains Alice's information at asymptotic rate $R = C - 2z_O$. (There is an asymptotically negligible rate overhead equalling $b^2/n + \epsilon$.

The b^2/n term corresponds, as before, to the identity matrix appended to X. The term ϵ takes any positive value, and the probability of error also depends on it.) The probability of error equals the sums of the probabilities of error in Claims 3 and 5, i.e., $|\mathcal{T}||\mathcal{E}|q^{-1}+q^{-n\epsilon}$. Of code design, encoding and decoding, the most computationally expensive is decoding. The costliest step involves inverting the linear transform corresponding to (13)-(14), which is of dimension $\mathcal{O}(nC)$. □

IX. Limited Adversary Model

We combine the strengths of the Shared Secret algorithm and the Omniscient Adversary algorithm, to achieve the higher rate of $C = C - z_O$, without needing a secret channel. The caveat is that Calvin's strength is more limited; the number of packets he can transmit, z_O, and the number he can eavesdrop on, z_I, satisfy the technical constraint

$$2z_O + z_I < C. \tag{15}$$

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" message to Bob at rate $C - 2z_O$. By (15), $z_I < C-2z_O$, 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 message to Bob is secret from Calvin. Alice then builds upon this secret, using the Shared Secret algorithm to transmit the bulk of her message to Bob at the higher rate $C - z_O$.

Though the following algorithm requires Alice to know z_O and z_I, we describe in §IX-A how to change the algorithm to make it independent of these parameters. The price we pay is a slight decrease in rate.

Alice's Encoder: Alice's encoder follows essentially the schema described above, except for a technicality – the information she transmits to Bob via the Omniscient Adversary algorithm is padded with some random symbols. This is for two reasons. Firstly, since the Omniscient Adversary algorithm has a probability of error that decays exponentially with the size of the input, it isn't guaranteed to perform well to transmit just a small message. Secondly, the randomness in the padded symbols also ensures strong information-theoretic secrecy of the small secret message, i.e., we can then show (in Claim 6) that Calvin's best estimate of *any function* of the secret information is no better than if he made random guesses.

Alice's information X decomposes into two parts $[X_1\ X_2]$. She uses the information she wishes to transmit to Bob, at rate $R = C - z_O - \Delta$, as input to the encoder of the Shared Secret algorithm, thereby generating the $b \times n(1-\Delta)$ sub-matrix X_1. Here Δ is a parameter that enables Alice to trade off between the the probability of error and rate-loss.

The second sub-matrix, X_2, which we call the *secrecy matrix* is analogous to the secret S used in the Secret Sharing algorithm described in §VII. The size of X_2 is $b \times \Delta n$. In fact, X_2 is an encoding of the secret S Alice generates in the Shared Secret algorithm. The $b(C+1)$ symbols corresponding to the parity symbols $\{r_d\}$ and the hash matrix H are written in the form of a length-$b(C+1)$ column vector. This vector is appended with symbols chosen uniformly at random from $\mathbb{F}_q$ to result in the length-$(C - z_O - \delta)\Delta n$ vector $\tilde{\mathbf{U}}'$. This vector $\tilde{\mathbf{U}}'$ could function as the input $\tilde{\mathbf{U}}$ to the Omniscient Adversary algorithm operated over a packet-size Δn, with a probability of decoding error that is exponentially small in Δn; however, we actually use a hash of $\tilde{\mathbf{U}}'$ to generate the input $\tilde{\mathbf{U}}$ to the Omniscient Adversary algorithm. To be more precise, $\tilde{\mathbf{U}} = V\tilde{\mathbf{U}}'$, where V is any square *MDS code generator matrix* [1] of dimension $(C - z_O - \delta)\Delta n$, known to all parties Alice, Bob, and Calvin. As we see later, hashing $\tilde{\mathbf{U}}'$ with V strengthen the secrecy of S (and enables the proof of Claim 6 below). Alice then uses the encoder for the Omniscient Adversary algorithm to generate X_2 from $\tilde{\mathbf{U}}$.

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 §IV-B.

Bob's Encoder: Bob arranges his received packets into the matrix $Y = [Y_1\ Y_2]$. The sub-matrices Y_1 and Y_2 are respectively the network transforms of X_1 and X_2.

Bob decodes in two steps. Bob first decodes Y_2 to obtain S. He begins by using the Omniscient Adversary decoder to obtain the vector $\tilde{\mathbf{U}}$. He obtains $\tilde{\mathbf{U}}'$ from $\tilde{\mathbf{U}}$, by multiplying by V^{-1}. He then extracts from $\tilde{\mathbf{U}}'$ the $b(C+1)$ symbols corresponding to S. The following claim, proved in the Appendix, ensures that S is indeed secret from Calvin.

Claim 6: The probability that Calvin guesses S correctly is at most $q^{-b(C+1)}$, i.e., S is information-theoretically secret from Calvin.

Thus Alice has now shared S with Bob. Bob uses 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 $R = C - z_O$. (There is an asymptotically negligible rate overhead equalling $b^2/n + \Delta$. The b^2/n term corresponds, as before, to the identity matrix appended to X. The term Δ takes any positive value, and the probability of error also depends on it.) The probability of error equals the sums of the probabilities of error in Theorems 1 and 2. The errors in Theorem 1 are analyzed in Claims 3 and 2. Theorem 2 is used to generate codes of blocklength Δn. This probability of error is analyzed in Claim 5. Together, an upper bound on the probability of error is $(|\mathcal{T}||\mathcal{E}|+n^C C)q^{-1}+q^{-\Delta n\varepsilon}$. Since the Limited Adversary algorithm is essentially a concatenation of the Shared Secret algorithm with the Omniscient Adversary algorithm, the computational cost is the sum of the computational costs of the two (with Δn replacing n as the block-length for the Shared Secret algorithm). This quantity therefore equals $\mathcal{O}(nC^2 + (\Delta nC)^3)$. Choosing Δ appropriately (say $\Delta = (C^{-\frac{1}{3}}n^{-\frac{2}{3}})$ makes the second term vanish. □

A. Limited Adversary: Universal Codes

We now discuss how to convert the above algorithm to be independent of the network parameters z_O and z_I. Alice's challenge is to design for all possible z_O and z_I pairs that satisfy the constraint (15). For any specific z_I, Alice needs to worry only about the largest z_O that satisfies (15) because what works

[1] Secret Sharing protocols [25] demonstrate that using MDS code generator matrices guarantees that to infer even a single symbol of $\tilde{\mathbf{U}}'$ from $\tilde{\mathbf{U}}$ requires the entire vector $\tilde{\mathbf{U}}$.

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

TABLE III—Comparison of our three algorithms

against an attacker with a particular traffic injection strength works against all weaker attackers. Note that C, z_O, and z_I are all integers, and thus there are only $C - 1$ such attackers. For each of these attackers, Alice designs a different secrecy matrix X_2 as described above. She appends these $C-1$ matrices to her information X_1 and sends the result as described in the above section.

To decode Bob needs to estimate which secrecy matrix to use, i.e., which one of them is secret from the attacker. For this he needs a good upper bound on z_O. But, just as in the omniscient adversary algorithm, he can obtain this by computing the column rank of Y, and subtracting b from it. He then decodes using the secrecy matrix corresponding to $(z_O, C - 1 - 2z_O)$. This secrecy matrix suffices since z_I can at most be $C-1-2z_O$, which corresponds to Calvin's highest eavesdropping strength for this z_O. □

X. 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 I. 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.

Acknowledgments

The authors would like to thank Air Force grant FA9550-06-1-0155, Caltech's Lee Center For Advanced Networking and Microsoft Research for support.

References

[1] R. Ahlswede, N. Cai, S. R. Li, and R. W. Yeung. Network information flow. *IEEE Transactions on Information Theory*, 46(5):1204–1216, July 2000.

[2] N. Cai and R. W. Yeung. Secure network coding. In *Proceedings of International Symposium in Information Theory and Its Applications*, Lausanne, Switzerland, June 2002.

[3] N. Cai and R. W. Yeung. Network error correction, part 2: Lower bounds. submitted to Communications in Information and Systems, 2006.

[4] D. Charles, K. Jain, and K. Lauter. Signatures for network coding. In *Proceedings of the fortieth annual Conference on Information Sciences and Systems*, Princeton, NJ, USA, 2006.

[5] D. Dolev, C. Dwork, O. Waarts, and M. Yung. Perfectly secure message transmission. *Journal of the Association for Computing Machinery*, 40(1):17–47, January 1993.

[6] C. Gkantsidis and P. Rodriguez. Network coding for large scale content distribution. In *Proceedings of IEEE Conference on Computer Communications (INFOCOM)*, Miami, March 2005.

[7] C. Gkantsidis and P. Rodriguez. Cooperative security for network coding file distribution. In *Proceedings of IEEE Conference on Computer Communications (INFOCOM)*, Barcelona, April 2006.

[2] A refinement of some of the algorithms in this work can be found in [11].

[8] T. Ho., R. Koetter, M. Médard, D. Karger, and M. Effros. The benefits of coding over routing in a randomized setting. In *IEEE International Symposium on Information Theory (ISIT)*, page 442, Yokohama, July 2003.

[9] T. Ho, M. Médard, J. Shi, M. Effros, and D. Karger. On randomized network coding. In *Proceedings of 41st Annual Allerton Conference on Communication, Control, and Computing*, Monticello, IL, 2003.

[10] 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 *International Symposium on Information Theory*, Chicago, USA, June 2004.

[11] S. Jaggi and M. Langberg. Resilient network coding in the presence of eavesdropping byzantine adversaries. submitted to ISIT, 2007.

[12] S. Jaggi, M. Langberg, T. Ho, and M. Effros. Correction of adversarial errors in networks. In *Proceedings of International Symposium in Information Theory and its Applications*, Adelaide, Australia, 2005.

[13] 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 Transactions on Information Theory*, 51(6):1973–1982, June 2005.

[14] A. Jiang. Network coding for joint storage and transmission with minimum cost. In *Proceedings of International Symposium in Information Theory and its Applications*, Seattle, Washington, USA, July 2006.

[15] S. Katti, D. Katabi, W. Hu, H. S. Rahul, and M. Médard. The Importance of Being Opportunistic: Practical Network coding for Wireless Environments. In *43rd Annual Allerton Conference on Communication, Control, and Computing*, Allerton, 2005.

[16] S. Katti, H. Rahul, D. Katabi, W. H. M. Médard, and J. Crowcroft. XORs in the Air: Practical Wireless Network Coding. In *ACM SIGCOMM*, Pisa, Italy, 2006.

[17] R. Koetter and F. Kschischang. Coding for errors and erasures in random network coding. Under Submission.

[18] R. Koetter and M. Médard. Beyond routing: An algebraic approach to network coding. In *Proceedings of the 21st Annual Joint Conference of the IEEE Computer and Communications Societies (INFOCOM)*, volume 1, pages 122–130, 2002.

[19] R. Koetter and M. Médard. An algebraic approach to network coding. *IEEE/ACM Transactions on Networking*, 11(5):782–795, October 2003.

[20] M. N. Krohn, M. J. Freedman, and D. Mazires. On-the-fly verification of rateless erasure codes for efficient content distribution. In *Proceedings of the IEEE Symposium on Security and Privacy, 2004*, Oakland, California.

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

[22] D. Lun, M. Médard, T. Ho, and R. Koetter. Network coding with a cost criterion. In *Proceedings of International Symposium in Information Theory and its Applications*, October 2004.

[23] D. S. Lun, M. Médard, and R. Koetter. Efficient operation of wireless packet networks using network coding. In *International Workshop on Convergent Technologies (IWCT)*, Oulu, Finland, 2005.

[24] A. Pelc and D. Peleg. Broadcasting with locally bounded byzantine faults. *Information Processing Letters*, 93(3):109–115, February 2005.

[25] T. Rabin and M. Ben-Or. Verifiable secret sharing and multiparty protocols with honest majority. In *Proceedings of the twenty-first annual ACM symposium on Theory of computing*, pages 73–85, Seattle, Washington, United States, 1989.

[26] L. Subramanian. *Decentralized Security Mechanisms for Routing Protocols*. PhD thesis, University of California at Berkeley, Computer Science Division, Berkeley, CA 94720, 2005.

[27] H. J. Wertz. On the numerical inversion of a recurrent problem: The Vandermonde matrix. AC-10:492, Oct. 1965.

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

[29] R. W. Yeung and N. Cai. Network error correction, part 1: Basic concepts and upper bounds. submitted to Communications in Information and Systems, 2006.

Appendix

A. Proof of Claim 2

The idea behind the claim is as follows. The parity-check matrix P is, by construction, a *Vandermonde matrix* [27], and therefore has full column rank. Further, since P is hidden from Calvin, with high probability he cannot choose interference such that the matrix product EP has a lower column rank than does E.

To prove this we use a generalization of an argument used in [12]. Let $S_{i,j}$ denote the (i, j) element of $S = EP$. We note that for each (i, j), $S_{i,j}$ can be thought of as a polynomial in r_j with coefficients from the i^{th} row of E. Since $S_{i,j}$ has degree at most n in r_j, at most

n values of r_j satisfy the equation $S_{i,j} = c$, for any scalar $c \in \mathbb{F}_q$. Since Calvin does not know the values of the r_js, the probability he can choose entries in E to satisfy any such equation is at most nq^{-1}.

In particular, the probability that the first row of S consists of the length-C zero vector is at most $(nq^{-1})^C$. For a particular choice of the first row of S, the probability that the second row is linearly dependent on the first row (i.e., is any scalar multiple of the first row) is at most n^C/q^{C-1}. Similarly, the probability that the third row is any of the q^2 possible linear combinations of the first two rows is at most n^C/q^{C-2}. Continuing thus, the probability that the i^{th} row of S is linearly dependent on the previous $i-1$ is at most n^C/q^{C-i+1}. Taking the union bound over all C events, the probability that S is singular is at most $n^C \sum_{i=1}^{C} 1/q^{C-i+1}$. Since the largest summand equals n^C/q, therefore the probability of the undesirable event is at most $Cn^C q^{-1}$. Hence, with probability at least $1 - Cn^C q^{-1}$, E and S are related via an invertible transformation. Note that q is a design parameter and can be chosen to be much larger than Cn^C to make the probability of error arbitrarily small. □

B. Proof of Claim 3

The proof of Claim 3 follows directly from [9]. Essentially, it is a consequence of the following facts. First, due to [9], with probability at least $(1 - |\mathcal{T}|q^{-1})^{|\mathcal{E}|}$ over network code design, $[T\ T_{Z\to Y}]$ has full column rank. Here $|\mathcal{T}|$ is the number of terminals in the multicast connection, and $|\mathcal{E}|$ is the number of (hyper) links in the underlying network. Secondly, the matrix $[\hat{T}\ S']$ can be obtained via an invertible transformation from the matrix $[T\ T_{Z\to Y}]$. Lastly, for large enough q, the quantity $(1 - |\mathcal{T}|q^{-1})^{|\mathcal{E}|}$ is strictly greater than $1 - |\mathcal{T}||\mathcal{E}|q^{-1}$. □

C. Proof of Claim 4

Rewriting the right-hand side of (12) and substituting for Y from (8) results in

$$\hat{T}X + T_{Z\to Y}(Z - LX) = \hat{T}[0\ F^X\ I_b] + T''[I_{z_O}\ F^Z\ 0]. \quad (16)$$

Since the columns of T'' are spanned by the columns of $[\hat{T}\ T_{Z\to Y}]$, therefore we may write T'' as $\hat{T}M_1 + T_{Z\to Y}M_2$, where the matrices M_1 and M_2 represent the appropriate basis transformation. Thus (16) becomes

$$\hat{T}X + T_{Z\to Y}(Z - LX) = \hat{T}\left([0\ F^X\ I_b]\right) + \left(\hat{T}M_1 + T_{Z\to Y}M_2\right)[I_{z_O}\ F^Z\ 0]. \quad (17)$$

Since the vector spaces spanned by the columns of $\hat{T}$ and $T_{Z\to Y}$ are disjoint (except in the zero vector), therefore we may compare the term multiplying the matrix $\hat{T}$ on both sides of 17 (we may also compare the term corresponding to $T_{Z\to Y}$, but this gives us nothing useful). This comparison gives us the equation

$$\hat{T}X = \hat{T}[0\ F^X\ I_b] + \hat{T}M_1[I_{z_O}\ F^Z\ 0]. \quad (18)$$

We split the matrix equation (16) into three parts, corresponding to the sub-matrices X_1, X_2 and X_3 of X. Thus (18) now splits into the three equations

$$\hat{T}X_1 = \hat{T}M_1 I_{z_O}, \quad (19)$$

$$\hat{T}X_2 = \hat{T}F^X + \hat{T}M_1F^Z, \text{ and} \quad (20)$$

$$\hat{T}X_3 = \hat{T}. \quad (21)$$

Equation (21) is trivial, since it only reiterates that X_3 equals columns of an identity matrix. Equation (19) allows us to estimate that M_1 equals X_1. We are finally left with (20), which by substituting for M_1 from (19) reduces to

$$\hat{T}X_2 = \hat{T}\left(F^X + X_1F^Z\right). \quad (22)$$

□

D. Proof of Claim 5

For $i = 1, 2$, we denote by $\tilde{\mathbf{X}}_i$ the vector obtained by stacking the columns of X_i one after the other. Let $D = [D_1\ D_2]$, where D_2 corresponds to the last b^2 columns of D and D_1 corresponds to the remaining columns of D. Define $\alpha = n - (b + z_O)$. Denote by $\vec{F^X}$ the vector formed by stacking columns of the matrix F^X one after the other, and by $f_{i,j}$ the $(i,j)^{th}$ entry of the matrix F^Z. The system of linear equations (13)-(14) can be written in matrix form as

$$A\begin{pmatrix} \tilde{\mathbf{X}}_1 \\ \tilde{\mathbf{X}}_2 \end{pmatrix} = \begin{pmatrix} \hat{T}\vec{F^X} \\ -D_2\tilde{\mathbf{I}} \end{pmatrix}$$

where A is given by

$$\left[\begin{array}{cccc|ccccc} -f_{1,1}\hat{T} & -f_{2,1}\hat{T} & \dots & -f_{z_O,1}\hat{T} & \hat{T} & 0 & \dots & \dots & 0 \\ -f_{1,2}\hat{T} & -f_{2,2}\hat{T} & \dots & -f_{z_O,2}\hat{T} & 0 & \hat{T} & 0 & \dots & 0 \\ -f_{1,3}\hat{T} & -f_{2,3}\hat{T} & \dots & -f_{z_O,3}\hat{T} & \vdots & 0 & \hat{T} & 0 & 0 \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & 0 & \ddots & 0 \\ -f_{1,\alpha}\hat{T} & -f_{2,\alpha}\hat{T} & \dots & -f_{z_O,\alpha}\hat{T} & 0 & 0 & 0 & 0 & \hat{T} \\ \hline \multicolumn{4}{c|}{D_1} & \multicolumn{5}{c}{} \end{array}\right]$$

This matrix A is described by smaller dimensional matrices as entries. The matrix $\hat{T}$ has dimensions $(b + z_O) \times b$. The j^{th} row of matrices in the top portion of matrix A describes an equation corresponding to the j^{th} column of the matrix equation in Equation 13. The bottom portion of A corresponds to Equation 14. Bob can recover the variables $X(i,j)$ if and only if the above matrix A has full column rank. We now analyze A to show that this is indeed the case (with high probability) for sufficiently large δn. Using Claim 3, we can assume that $\hat{T}$ has full column-rank, and therefore the last ab columns of the matrix (represented by the right side of A) have full column rank.

We now address the left columns of A. Consider performing column operations from right to left, to zero out the $\hat{T}$s in the left side of the top rows of A (that is, to zero out the upper left sub-matrix of A). A has full column rank iff after this process the lower left sub-matrix of A has full column rank. We show that this is the case with high probability over the random elements of D (when δn is chosen to be sufficiently large). Let $f_{i,j}$'s be the values appearing in the upper left sub-matrix of A. We show that for any (adversarial) choice of f_{ij}'s, with high probability, the act of zeroing out the $\hat{T}$'s yields a lower left sub-matrix of A with full column rank. Then using the union bound on all possible values of f_{ij} we obtain our assertion.

For any fixed values of f_{ij}, let $C(j)$, for $j = 1$ to bz_O, denote the columns of the lower left sub-matrix of A after zeroing out the $\hat{T}$'s. For each j, the vector $C(j)$ is a linear combination of the (lower part of the) j^{th} column of A with columns from the lower right sub-matrix of A. As the entries of D_1 are independent random variables uniformly distributed in $\mathbb{F}_q$, the columns $C(j)$ for $j = 1, \dots, bz_O$ consist of independent entries that are also uniformly distributed in $\mathbb{F}_q$. Standard analysis shows that the probability that the columns $C(j)$ are not independent is $q^{bz_O - \delta n}$. For the union bound we would like this probability to be at most $q^{-\alpha z_O - n\varepsilon} = q^{-(n-(b+z_O))z_O - n\varepsilon}$. Thus, it suffices to take $\delta n = n(z_O + \varepsilon)$ for an error probability of at most $q^{-n\varepsilon}$. Recall that $b = C - z_O$.

E. Proof of Claim 6

The vector $\tilde{\mathbf{U}}$ was generated from $\tilde{\mathbf{U}}'$ via an MDS code generator matrix (see Footnote 1), and a folklore result about network codes is that with high probability over random network code design the linear transform between Alice and Calvin also has the MDS property. Thus, for Calvin to infer even a single symbol of the length-$(C - z_O - \delta)n\Delta$ vector $\tilde{\mathbf{U}}'$, he needs to have received at least $(C - z_O - \delta)n\Delta$ linear combinations of the variables in the secrecy matrix X_2. Since Calvin can overhear z_l packets, he has access to $z_l n\Delta$ equations that are linear in the unknown variables. The difference between the number of variables unknown to Calvin, and the number of equations Calvin has, is linear in $n\Delta$ – for large enough $n\Delta$, this difference is larger than $b(C+1)$, the length of the vector S. By a direct extension of [25], Calvin's probability of guessing any function of S correctly is $q^{-b(C+1)}$. □

Random Linear Network Coding: A free cipher?

Luísa Lima Muriel Médard João Barros

Abstract— **We consider the level of information security provided by random linear network coding in network scenarios in which all nodes comply with the communication protocols yet are assumed to be potential eavesdroppers (i.e. "nice but curious"). For this setup, which differs from wiretapping scenarios considered previously, we develop a natural *algebraic security* criterion, and prove several of its key properties. A preliminary analysis of the impact of network topology on the overall network coding security, in particular for complete directed acyclic graphs, is also included.**

Index Terms— **security, information theory, graph theory, network coding.**

I. Introduction

Under the classical networking paradigm, in which intermediate nodes are only allowed to store and forward packets, information security is usually viewed as an independent feature with little or no relation to other communication tasks. In fact, since intermediate nodes receive exact copies of the sent packets, data confidentiality is commonly ensured by cryptographic means at higher layers of the protocol stack. Breaking with the ruling paradigm, network coding allows intermediate nodes to mix information from different data flows [1], [2] and thus provides an intrinsic level of data security — arguably one of the least well understood benefits of network coding.

Previous work on this issue has been mostly concerned with constructing codes capable of spliting the data among different links, such that reconstruction by a wiretapper is either very difficult or impossible. In [3], the authors present a secure linear network code that achieves perfect secrecy against an attacker with access to a limited number of links. A similar problem is considered in [4], featuring a random coding approach in which only the input vector is modified. [5] introduces a different information-theoretic security model, in which a system is deemed to be secure if an eavesdropper is unable to get any decoded or decodable (also called *meaningful*) source data. Still focusing on wiretapping attacks, [6] provides a simple security protocol exploiting the network topology: an attacker is shown to be unable to get any meaningful information unless it can access those links that are necessary for the communication between the legitimate sender and the receiver, who are assumed to be using network coding. As a distributed capacity-achieving approach for the multicast case, randomized network coding [7], [8] has been shown to extend naturally to packet networks with losses [9] and Byzantine modifications (both detection and correction [10], [11], [12], [13]). [14] adds a cost criterion to the secure network coding problem, providing heuristic solutions for a coding scheme that minimizes both the network cost and the probability that the wiretapper is able to retrieve all the messages of interest.

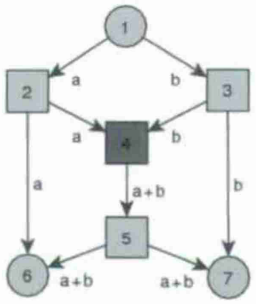


Fig. 1. Canonical Network Coding Example. In this image, intermediate nodes are represented with squares. With this code, node 4 is a vulnerability for the network since it can decode all the information sent through it. Note that the complete opposite happens for node 5, that receives no meaningful information whatsoever.

In this work, we approach network coding security from a different angle: our focus is *not* on the threat posed by external wiretappers but on the more general threat posed by intermediate nodes. We assume that the network consists entirely of "nice but curious" nodes, i.e. they comply with the communication protocols (in that sense, they are well-behaved) but may try to acquire as much information as possible from the data that passes through them (in which case, they are potentially malicious). This notion is highlighted in the following example.

Example 1: Consider the canonical network coding example with 7 nodes, shown in *Figure 1*. Node 1 sends a flow to sinks 6 and 7 through intermediate nodes 2, 3, 4 and 5. From the point of security, we can distinguish between three types of intermediate nodes in this setting: (1) those that only get a non-meaningful part of the information, such as node 5; (2) those that obtain all of the information, such as node 4; and (3) those that get partial yet meaningful information, such as nodes 2 and 3. Although this network code could be considered *secure* against single-edge external wiretapping — i.e. , the wiretapper is not able to retrieve the whole data simply

L. Lima (luisalima@ieee.org) and J. Barros (barros@dcc.fc.up.pt) are with the Instituto de Telecomunicações (IT) and the Department of Computer Science, Faculdade de Ciências da Universidade do Porto, Portugal. M. Médard (medard@mit.edu) is with the Laboratory for Information and Decision Systems at the Massachusetts Institute of Technology. This work was partly supported by the Fundação para a Ciência e Tecnologia (Portuguese Foundation for Science and Technology) under grant SFRH/BD/24718/2005 and by AFOSR under grant "Robust Self-Authenticating Network Coding" AFOSR 000106. Part of this work was done while the first author was a visiting student at the Laboratory for Information and Decision Systems at the Massachusetts Institute of Technology.

by eavesdropping on a single edge — it is clearly insecure against internal eavesdropping by an intermediate node.

Motivated by this example, we set out to investigate the security potential of network coding. Our main contributions are as follows:

- *Problem Formulation*: We formulate a secure network coding problem, in which all intermediate nodes are viewed as potential eavesdroppers and the goal is to characterize the intrinsic level of security provided by random linear network coding.
- *Algebraic Security Criterion*: Based on the notion that the number of decodable bits available to each intermediate node is limited by the degrees of freedom it receives, we are able to provide a natural secrecy constraint for network coding and to prove some of its most fundamental properties.
- *Security Analysis for Complete Directed Acyclic Graphs*: As a preliminary step towards understanding the interplay between network topology and security against eavesdropping nodes, we present a rigorous characterization of the achievable level of algebraic security for this class of complete graphs.

The remainder of this paper is organized as follows. First, a formal problem statement is in *Section II*, followed by a detailed analysis of the algebraic security of Randomized Linear Network Coding in *Section III*. In *Section IV*, this analysis is carried out specifically for complete directed acyclic graphs. The paper concludes with *Section V*.

II. Problem Setup

We adopt the network model of [2]: we represent the network as an acyclic directed graph $G = (V, E)$, where V is the set of nodes and E is the set of edges. Edges are denoted by round brackets $e = (v, v') \in E$, in which $v = \text{head}(e)$ and $v' = \text{tail}(e)$. The set of edges that end at a vertex $v \in V$ is denoted by $\Gamma_I(v) = \{e \in E : \text{head}(e) = v\}$, and the in-degree of the vertex is $\delta_I(v) = |\Gamma_I(v)|$; similarly, the set of edges originating at a vertex $v \in V$ is denoted by $\Gamma_O(v) = \{e \in E : \text{tail}(e) = v\}$, the out-degree being represented by $\delta_O(v) = |\Gamma_O(v)|$.

Discrete random processes $X_1, ...X_K$ are observable at one or more source nodes. To simplify the analysis, we shall consider that each network link is free of delays and that there are no losses. Moreover, the capacity of each link is one bit per unit time, and the random processes X_i have a constant entropy rate of one bit per unit time. Edges with larger capacities are modelled as parallel edges and sources of larger entropy rate are modelled as multiple sources at the same node. We shall consider multicast connections as it is the most general type of single connection; there are $d \geq 1$ receiver nodes. The objective is to transmit all the source processes to each of the receiver nodes.

In linear network coding, edge $e = (v, u)$ carries the process $Y(e)$, which is defined below:

$$Y(e) = \sum_{l:X_l \text{ generated at } v} \alpha_{l,e} X(v,l) + \sum_{e':head(e')=tail(e)} \beta_{e',e} Y(e')$$

The *transfer matrix* M describes the relationship between an input vector $\underline{x}$ and an output vector $\underline{z}$, $\underline{z} = \underline{x}M$; $M = A(I - F)^{-1}B^T$, where A and B represent, respectively, the linear mixings of the input vector and of the output vector, and have sizes $K \times |E|$ and $\nu \times |E|$. F is the adjacency matrix of the directed labelled line graph corresponding to the graph G. In this paper we shall not consider matrix B, which only refers to the decoding at the receivers. Thus, we shall mainly analyse parts of the matrix AG, such that $G = (I - F)^{-1}$; $\underline{a}_i$ and $\underline{c}_i$ denote column i of A and AG, respectively. We define the *partial transfer matrix* $M'_{\Gamma_I(v)}$ (also called *auxiliary encoding vector* [9]) as the observable matrix at a given node v, i.e. the observed matrix formed by the symbols received at a node v. This is equivalent to the fraction of the data that an intermediate node has access to in a multicast transmission.

Regarding the coding scheme, we consider the random linear network coding scheme introduced in [7]: and thus each coefficient of the matrices described above is chosen independently and uniformly over all elements of a finite field $\mathbb{F}_q$, $q = 2^m$.

Our goal is to evaluate the *intrinsic security* of random linear network coding, in multicast scenarios where all the intermediate nodes in the network are potentially malicious eavesdroppers. Specifically our threat model assumes that intermediate nodes perform the coding operations as outlined above, and will try to decode as much data as possible.

III. Algebraic Security of Random Linear Network Coding

A. Algebraic security

The Shannon criterion for information-theoretic security [15] corresponds in general terms to a zero mutual information between the cypher-text (C) and the original message (M), i.e. $I(M;C) = 0$. This condition implies that an attacker must guess $\leq H(M)$ symbols to be able to compromise the data. With network coding, on the other hand, if the attacker is capable of guessing M symbols, $K - M$ additional observed symbols are required for decoding — by noting that each received symbol is a linear combination of the K message symbols from the source, we can see that a receiver must receive K coded symbols in order to recover one message symbol. Thus, as will be shown later, restricted rank sets of individual symbols do not translate into immediately decodable data with high probability. This notion is illustrated in *Figure 2*. In the scheme shown on top, each intermediate node can recover half of the transmitted symbols, whereas in the bottom scheme none of the nodes can recover any portion of the sent data.

Definition 1 (Algebraic Security Criterion): The level of security provided by random linear network coding is measured by the number of symbols that an intermediate node v has to guess in order to decode *one* of the transmitted symbols. From a formal point of view,

$$\Delta_S(v) = \frac{K - (\text{rank}(M'_{\Gamma_I(v)}) + l_d}{K},$$

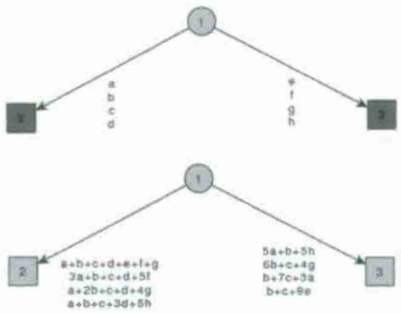


Fig. 2. Example of algebraic security. In the upper scheme data is not protected, whereas in the lower scheme nodes 2 and 3 are unable to recover any data symbols.

where l_d represents the number of partially diagonalizable lines of the matrix (i.e. the number of message symbols that can be recovered by Gaussian elimination).

Notice that the previous definition is equivalent to computing the difference between the global rank of the code and the local rank in each intermediate node v. Moreover, as more and more symbols become compromised of security criteria, the level of security tends to 0, since as we shall show in this section, with high probability the number of individually decodable symbols l_d goes to zero as the size of the field goes to infinity.

B. Security Characterization

We are now ready to solve the problem of characterizing the algebraic security of random linear network coding. The key to our proofs is to analyze the properties of the partial transfer matrix at each intermediate node. Recall that there are two cases in which the intermediate node can gain access to relevant information: (1) when the partial transfer matrix has full rank and (2) when the partial transfer matrix has diagonalizable parts. Thus, we shall carry out independent analyzes in terms of rank and in terms of partially diagonalizable matrices. The following lemmas will be useful.

Lemma 1: In the random linear network coding scheme,

$$P(\Delta_S > 0) \leq P(\exists v : \delta_I(v) > K).$$

Proof: See the *Appendix*. ∎

It follows from this lemma that it is only necessary to consider the case in which $K \leq \delta_I(v)$.

Lemma 2: The probability that a linear combination of independent and uniformly distributed values in $\mathbb{F}_q$ yields the zero result is bounded by

$$P(X_{lin} = 0) \leq \frac{2q + h(q)}{q^2},$$

where $h(q)$ is a function such that $O(h(q)) < O(q^2)$. Moreover, $P(X_{lin} = 0)$ tends to 0 when $q \to \infty$.

Proof: See the *Appendix*. ∎

Lemma 3: The probability of obtaining y zeros in one line of the $\xi \times \xi$ transfer matrix M is bounded by

$$P(Y = y) \leq \binom{\xi}{\xi - y} \left(\frac{2q + h(q)}{q^2}\right)^y \left(1 - \frac{2q + h(q)}{q^2}\right)^{\xi - y}.$$

Proof: See the *Appendix*. ∎

Theorem 1: Let $P(l_d > 0)$ be the probability of recovering a strictly positive number of symbols l_d at the intermediate nodes with $\delta_I(v) \leq K - 1$ by Gaussian elimination. Then, $P(l_d > 0) \to 0$ with $q \to \infty$ and $K \to \infty$.

Proof: Let M' be the transpose of the partial transfer matrix at some vertex v, $M' = M^T_{\Gamma_I(v)}$. We consider the process of Gaussian elimination of M'. It is unnecessary to consider rank K, since in that case the matrix, w.h.p, is invertible and hence diagonalizable [8]. Thus, M' is a $\delta_I(v) \times K$ matrix, $\delta_I(v) < K$.

We prove the theorem constructively by analysing the probability of having $K - 1$ zeros in one or more lines of M'. Let p be the probability of having $K - 1$ zeros in a line of M', and let X be a random variable representing the recoverable number of symbols when an intermediate node has $\delta_I(v)$ degrees of freedom. It follows from *Lemma 3* that

$$p = \binom{K}{K-1} \left(\frac{2q + h(q)}{q^2}\right)^1 \left(1 - \frac{2q + h(q)}{q^2}\right)^{K-1}.$$

In the base case with $\delta_I(v) = 1$, at most $X = 1$ symbols can be recovered, since there are not enough degrees of freedom to perform Gaussian elimination and the only chance for recovering a symbol is that the line of the matrix M already has $K - 1$ zeros. The probability for this is p.

In the case that $1 < \delta_I(v) < K$, we can obtain directly a number $L = l$ of lines with $K-1$ zeros, and a number $\delta_I(v)-l$ of lines in the opposite situation. Since we have $\delta_I(v)$ degrees of freedom to perform Gaussian elimination, we can obtain at most $\delta_I(v)$ symbols by successive elimination. At each step the probability of obtaining a line with $K-1$ zeros is bounded by p.

By analysing the different possibilities of combinations for the lines that already have $K - 1$ zeros and the ones that can be obtained by Gaussian elimination, we get

$$P(X = x) \leq \sum_{l=0}^{x} \binom{\delta_I(v)}{l} p^l (1-p)^{\delta_I(v)-l} P_l(X = x)$$

$$P_l(X = x) \leq \binom{\delta_I(v) - l}{x - (\delta_I(v) - l)} p^{x - \delta_I(v) + l} (1-p)^{2\delta_I(v) - 2l - x},$$

where $P_l(X = x)$ represents $P(X = x | L = l)$.

Approximating the binomial distribution by a normal distribution yields

$$P_l(X = x) \approx \frac{e'}{\sqrt{2\pi(\delta_I(v) - l)p(1-p)}},$$

where

$$e' = \exp\left(-\frac{1}{2}\frac{(x - (\delta_I(v) - l)p)^2}{(\delta_I(v) - l)p(1-p)}\right)$$

Since $p \to p* < 1$, we can state that, when $q \to \infty$ and $p \to 0$ is $\approx \exp(x^2)$. When K goes to ∞, so does x, and hence

$$\exp(x^2)_{x \to \infty} \to 0,$$

and

$$P_l(X = K - 1)_{q \to \infty, K \to \infty} \to 0.$$

Since

$$P(X = K-1) = \sum_{l=0}^{K-1} \binom{\delta_I(v)}{l} p^l (1-p)^{\delta_I(v)-l} P_l(X = K-1),$$

and $P_l(X = K-1)$ decreases exponentially, and l only increases linearly,

$$P(X = K-1)_{q\to 0, K-\infty} \to 0.$$

The probability of obtaining $X < K-1$ symbols is bounded by $P(X = K-1)$; it follows that the probability of decoding X symbols with any $\delta_I(v) < K$ goes to zero as q and K tend to infinity. ∎

IV. Algebraic Security of the Complete Graph

Notice that, in consequence of the property outlined in Lemma 1, the algebraic security of a graph is topology dependent. A node with $\delta_I(v) \geq K$ will not necessarily receive a full-rank partial transfer matrix. The rank depends on the available paths between sources and each intermediate node. More specifically, depending on the topology of the graph, some nodes may receive only combinations of symbols derived from matrices with restricted rank, i.e. less than K. This includes, for example, trees, where a node connected directly to the source by a link of capacity C can only have children that receive at most rank C.

As a first step towards general network models, we consider the case of complete acyclic directed graphs $G = (V, E)$, $n = |V|$, which can be generated as follows.

- Generate random labels for the n vertices. These have some ordering $\{e_1, e_2, ..., e_n\}$ associated to them;
- Make an outgoing (directed) edge from the vertex with the minimum label to every vertex with a higher label;
- Continue until we reach a vertex where there are no more possibilities for connections.

This algorithm generates a complete acyclic directed graph with one source, one sink and $|E| = n(n-1)/2$ edges, since the total degree of each vertex is $n - 1 = \delta_I(v) + \delta_O(v)$. The source and the sink are naturally determined as those nodes that have only outgoing edges or only incoming edges, respectively. The ordering ensures that this algorithm always generates an acyclic directed graph, conferring the graphs generated in this way specific properties such as the distribution of the in and out-degrees. These properties can be determined directly from the order of the vertex using $\delta_O(v) = n - order(v)$ and $\delta_I(v) = n - \delta_O(v) - 1 = order(v) - 1$.

Before proving our next theorem, we introduce the following lemmas.

Lemma 4: In complete acyclic directed graphs, a node that receives R symbols, receives w.h.p. a partial transfer matrix with rank equal to $\min(R, K)$.

Proof: See the *Appendix*. ∎

Lemma 5: For the complete directed acyclic graph, w.h.p.,

$$\Delta_S(v) = \frac{K - \min(K, \text{order}(v))}{K}.$$

Proof: See the *Appendix*. ∎

Theorem 2: Let ϕ_S be the *secure max-flow*, defined as the maximum number of symbols that may be secured in a transmission by using random linear network coding. For a complete acyclic directed graph with n nodes, the secure max-flow equals the max-flow min-cut capacity of the network and is $n-1$. Conversely, the minimum numbers of required symbols for secured transmission is $n-1$ symbols.

Proof:

Suppose, by contradiction, that $K = n-1$ is the max-flow min-cut capacity of the complete directed acyclic graph. The maximum order of an intermediate node v is $n-2$, thus by *Lemma 5* we have $\Delta_S(v) = 1/(n-1)$. It follows that the secure max-flow of the complete acyclic directed graph equals the capacity of the graph.

By contradiction, let the minimum number of required symbols for secured transmission be $m_s \leq n-2$. There exists an intermediate node v such that $\text{order}(v) = n-1$, and consequently, $\Delta_S(v) = 0$. Then the minimum number of required symbols for secure transmission is $m_s = n-1$.

∎

It follows that the way to secure this class of complete graphs is to transmit at the max-flow min-cut capacity, if necessary by adding "dummy" symbols.

V. Conclusions

Intrigued by the security potential inherent to random linear network coding, we developed a specific algebraic security criterion, for which we proved a set of key properties. Perhaps one of the most striking conclusions of our analysis is that algebraic security with network coding is very dependent on the topology of the network. As an example, we focused on complete acyclic directed graphs, and determined the secure max-flow, as well as the minimum number of symbols required for algebraic security. As part of our ongoing work, we are extending this analysis to other more general network models. Ultimately, we would like to develop secure communication protocols capable of exploiting random linear network coding as an *almost free* cypher.

Acknowledgements

The authors gratefully acknowledge insightful discussions with Rui A. Costa (Univ. of Porto).

References

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

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

[3] N. Cai and RW Yeung, "Secure network coding," *Proceedings of the IEEE International Symposium on Information Theory*, Lausanne, Switzerland, 2002.

[4] J. Feldman, T. Malkin, C. Stein, and RA Servedio, "On the capacity of secure network coding," *Proc. of the 42nd Annual Allerton Conference on Communication, Control, and Computing*, 2004.

[5] K. Bhattad and K.R. Narayanan, "Weakly secure network coding," *Proc. of the First Workshop on Network Coding, Theory, and Applications (NetCod)*, Riva del Garda, Italy, 2005.

[6] K. Jain, "Security based on network topology against the wiretapping attack," *IEEE Wireless Communications*, vol. 11, no. 1, pp. 68–71, 2004.

[7] T. Ho, R. Koetter, M. Medard, D.R. Karger, and M. Effros, "The benefits of coding over routing in a randomized setting," *Proc. of the IEEE International Symposium on Information Theory (ISIT)*, Yokohama, Japan, June/July 2003.

[8] T. Ho, M. Medard, J. Shi, M. Effros, and D.R. Karger, "On randomized network coding," *Proceedings of the 41st Annual Allerton Conference on Communication, Control, and Computing*, 2003.

[9] D.S. Lun, M. Medard, R. Koetter, and M. Effros, "On Coding for Reliable Communication over Packet Networks," *Arxiv preprint cs.IT/0510070*, 2005.

[10] T. Ho, B. Leong, R. Koetter, M. Medard, M. Effros, and DR Karger, "Byzantine modification detection in multicast networks using randomized network coding," *Proceedings of the International Symposium on Information Theory*, Yokohama, Japan, June/July 2003.

[11] S. Jaggi, M. Langberg, T. Ho, and M. Effros, "Correction of adversarial errors in networks," *Proceedings of the International Symposium on Information Theory*, Adelaide, Australia, September 2005.

[12] S. Jaggi, M. Langberg, S. Katti, T. Ho, D. Katabi, and M. Medard, "Resilient Network Coding In the Presence of Byzantine Adversaries," IEEE Infocom, 2006.

[13] Sidharth Jaggi, *Design and Analysis of Network Codes*, Ph.D. thesis, California Institute of Technology, 2005.

[14] J. Tan and M. Medard, "Secure Network Coding with a Cost Criterion," *Proc. 4th International Symposium on Modeling and Optimization in Mobile, Ad Hoc and Wireless Networks (WiOpt'06)*, Boston MA, April, 2006.

[15] C.E. Shannon, *Communication Theory of Secrecy Systems*, Bell Systems Technical Journal, Vol. 28, pp. 656-715, October 1949.

APPENDIX

Proof of Lemma 1

We will prove this constructively in terms of the ranks of parts of the transfer matrix. The auxiliary encoding vector in each intermediate node v is given by

$$M'_{\Gamma_I(v)} = (A(I-F)^{-1})_{\Gamma_I(v)},$$

where $M'_{\Gamma_I(v)}$ denotes the columns of the matrix corresponding to the incoming edges of v. The dimension of $M'_{\Gamma_I(v)}$ is $K \times \delta_I(v)$, with $\delta_I(v) < |E|$.

To determine the rank of the partial transfer matrix, we note that the transfer matrix $M = A(I-F)^{-1}B^T$ for the network must be invertible, and hence, $\text{rank}(M) = K$. On the other hand, to determine the rank of $A(I-F)^{-1}$ we use the fact that $(I-F)^{-1}$ is invertible and thus $\text{rank}((I-F)^{-1}) = |E|$. We also have

$$\text{rank}(A(I-F)^{-1}) \leq |E|,$$

because the dimension of $A(I-F)^{-1}$ is $K \times |E|$. But, since

$$\text{rank}(A(I-F)^{-1}B^T) = K = \min(\text{rank}(A(I-F)^{-1}), B)$$

holds and $K < |E|$ (true because K must be less than the minimum cut in the network) we conclude that

$$\text{rank}(A(I-F)^{-1}) = K.$$

We now consider $\Delta_S(v)$ at some vertex v. For that, we can consider two distinct cases: the first one is if $K < \delta_I(v)$. In this case, we cannot assume anything about $\Delta_S(v)$, since the rank of the matrix $M'_{\Gamma_I(v)}$ will be dependent on the topology of the network. As for the second case, $\text{rank}(M'_{\Gamma_I(v)}) < K \Rightarrow \Delta_S(v) < 0$. ■

Proof of Lemma 2

Contrary to the sum, the product of independent and uniformly distributed values in $\mathbb{F}_q$ is *not* independent and uniformly distributed. In fact, there are two ways to obtain a zero in a multiplication in $\mathbb{F}_q$: (1) by multiplication between an element $a \in \mathbb{F}_q$ and 0, and (2) by multiplication over two elements $a \in \mathbb{F}_q$ and $b \in \mathbb{F}_q$, such that $a \neq 0$ and $b \neq 0$, but $ab = 0$. Now, the total number of entries of the multiplicative table between q elements of $\mathbb{F}_q$ is q^2, and there are at most $2q$ instances of the first case: q instances of $ab = 0$, $a = 0$ and $b \neq 0$, and q instances of $ab = 0$, $a = 0$ and $b \neq 0$. As for the second case, it is possible to prove by contradiction that the number of zeros obtained this way is strictly less than q^2: if this was not the case, all products of elements of $\mathbb{F}_q$ would be zero, and that is absurd. Since this is true for any q, the number of zeros grows $O(h(q)) < O(q^2)$. Thus, we have

$$P(X_{lin} = 0) \leq \frac{2q + h(q)}{q^2}.$$

Since for large enough q we have $(2+h(q))/q < 1$, it follows that

$$P(X_{lin} = 0)_{q \to \infty} = 0.$$

■

Proof of Lemma 3

Each position of a line of the transfer matrix M is a linear combination of independently and uniformly chosen values in $\mathbb{F}_q$, and thus, the probability of obtaining a zero in a position is given by *Lemma 2*. The result follows by considering all the combinations of the possible positions in which the Y zeros may occur. ■

Proof of Lemma 4

Suppose that a given intermediate node receives $R = K + \theta$ symbols, $\theta \geq 0$. It is clear that the maximum possible rank is K and thus there is a way to remove θ columns s.t. the rank of the resulting set will still be at maximum K. Now consider the case in which vertex v receives at most K symbols. If the columns are linearly dependent, the condition

$$\{x_{h_1}\underline{c}_{h_1} + x_{h_2}\underline{c}_{h_2} + ... + x_{h_n}\underline{c}_{h_n} = (0...0)^T\},$$

such that $x_{h_1}, x_{h_2}, ..., x_{h_n}$ not all $0, \in \mathbb{F}_q$ and $h_1, h_2, ..., h_n$ represent the columns $\in \Gamma_I(v)$, will be satisfied. Since the linear combination of lines of the transfer matrix is again a linear combination of independent and uniformly distributed values in $\mathbb{F}_q$, it follows from *Lemma 3* that the probability of obtaining $(0...0)^T$ tends to 0 when $q \to \infty$ and $K \to \infty$, and thus, the columns $h_1, h_2, ..., h_n \in \Gamma_I(v)$ are linearly independent w.h.p. ■

Proof of Lemma 5

It follows from *Lemma 4* that w.h.p., the number of symbols received by a vertex is the rank of the partial transfer matrix received (and at most K) and thus

$$\Delta_S(v) = \frac{K - \min(K, \delta_I(v))}{K} = \frac{K - \min(K, \text{order}(v) - 1)}{K}$$

■

A Rank-Metric Approach to Error Control in Random Network Coding

Danilo Silva and Frank R. Kschischang
Department of Electrical and Computer Engineering
University of Toronto
Toronto, Ontario M5S 3G4, Canada
{danilo, frank}@comm.utoronto.ca

Ralf Koetter
Institute for Communications Engineering
TU Munich
D-80333 Munich, Germany
ralf.koetter@tum.de

Abstract—**The problem of error control in random network coding is considered, and a formulation of the problem is given in terms of rank-metric codes. This formulation allows many of the tools developed for rank-metric codes to be applied to random network coding. A random network code induces a generalized decoding problem for rank-metric codes in which the channel may supply partial information about the error in the form of erasures (knowledge of an error location not its values) and deviations (knowledge of an error value but not its location).**

I. Introduction

While random network coding [1] is an effective technique for information dissemination in communication networks, it is highly susceptible to errors. The insertion of even a single corrupt packet has the potential, when linearly combined with legitimate packets, to affect all packets gathered by an information receiver. The problem of error control in random network coding is therefore of great interest.

In this paper, we focus on end-to-end error control coding. Internal network nodes are assumed to be unaware of the presence of an outer code; they simply create outgoing packets as random linear combinations of incoming packets in the usual manner of random network coding. Unlike some approaches to error control in network coding (e.g., [2], [3]) we assume that the source and destination nodes have no knowlege—or at least make no effort to exploit knowledge—of the topology of the network or of the particular network code used in the network.

Two previous "noncoherent" or "channel-blind" approaches to data transmission in coded networks are closely related to the work of this paper. Jaggi *et al.* [4] provide polynomial-time rate-optimal network codes that combat Byzantine adversaries. Their approach is based on probabilistic assumptions that require both the field size and the packet length to be sufficiently large. In contrast, Koetter and Kschischang [5] take a more combinatorial approach to the problem. Their key observation is that a random coded network may be regarded as a noncoherent multiple-input multiple-output system (over a finite field). In this context, an appropriate encoding of information is the choice by the transmitter of a suitable *vector space* V, rather than a vector as in classical coding theory. The choice of the space V is signalled by insertion into the network of a basis for V, where each basis vector corresponds to a transmitted packet. As V is closed under linear combinations of vectors, random network coding will (in the absence of noise) preserve the space, even as it mixes the transmitted vectors themselves. A receiver collects packets, which are assumed to form a basis for a received space U. Correct reception is possible provided that U and V intersection in a space of sufficiently large dimension. By defining an appropriate metric on subspaces, a generalization of classical coding theory in the Hamming metric becomes possible. This approach works for any given field and imposes virtually no constraints on packet size.

This paper is motivated by the results of [5] and addresses the construction of practical codes. Our main contribution is to show that, for a large class of codes, the subspace distance metric of [5] and the rank metric (e.g., [6], [7]) are strongly related, allowing many of the tools from the theory of rank-metric codes to be naturally applied to random network coding. We note, however, that our approach is not a straightforward application of rank-metric codes. Under random network coding, two phenomena can occur, called here erasures and deviations, that depart from the conventional notion of rank errors. Erasures and deviations are dual to each other and correspond to partial information about the error matrix, akin to the role played by symbol erasures in the Hamming metric. In our context, an erasure corresponds to the knowledge of an error location but not its value, while a deviation correspond to the knowledge of an error value but not its location. These concepts generalize similar concepts found in the rank-metric literature under the terminology of "row and column erasures" [8]–[10]. In related work [11], we provide an efficient decoding algorithm for Gabidulin codes [7] that takes into account erasures and deviations. However, space limitations make it impossible to describe these results here.

The remainder of this paper is organized as follows. In Section II, we provide a brief description of the rank metric and its properties. In Section III, we provide a formulation of the problem of error control in random network coding. Finally, in Section IV, we demonstrate the strong connection between certain decoding problems in the rank metric and the subspace decoding approach of [5]. Proofs have been omitted

1-4244-1200-5/07/$25.00 ©2007 IEEE.

throughout; however, see [11].

II. Preliminaries

Let $q \geq 2$ be a power of a prime. In this paper, all vectors and matrices are defined over the finite field $\mathbb{F}_q$, unless otherwise mentioned. We denote $\mathbb{F}_q^{n\times m}$ as the set of all $n \times m$ matrices over $\mathbb{F}_q$ and we set $\mathbb{F}_q^n = \mathbb{F}_q^{n\times 1}$. Thus, $v \in \mathbb{F}_q^n$ is a column vector, whereas $v \in \mathbb{F}_q^{1\times m}$ is a row vector.

If v is a vector, then the symbol v_i denotes the ith entry of v. For matrices, the notation varies according to how the matrix is defined: $A_1, \ldots, A_k$ could be either the rows of a matrix $A = \begin{bmatrix} A_1 \\ \vdots \\ A_k \end{bmatrix}$ or the columns of a matrix $A = [A_1, \ldots, A_k]$. The distinction will always be clear from context. In either case, the symbol A_{ij} always refers to the entry in the ith row and jth column of A.

The $k \times k$ identity matrix is denoted by $I_{k\times k}$. If $I = I_{k\times k}$, then the notation I_i refers to the ith column of I.

We will make extensive use of the column vector variables $L_1, L_2, \ldots \in \mathbb{F}_q^n$ and the row vector variables $V_1, V_2, \ldots \in \mathbb{F}_q^{1\times m}$. For these variables only, we introduce the notations $L_s^t = [L_s, \ldots, L_t]$ and $V_s^t = \begin{bmatrix} V_s \\ \vdots \\ V_t \end{bmatrix}$. Note that the product $L_s^t V_s^t$ can be expanded as

$$L_s^t V_s^t = \sum_{k=s}^{t} L_k V_k \tag{1}$$

since $(L_s^t V_s^t)_{ij} = \sum_{k=s}^{t} L_{ik} V_{kj}$.

If $\mathcal{S} \subseteq \{1, \ldots, k\}$ and $A \in \mathbb{F}_q^{n\times k}$, then $A_{\mathcal{S}} = [A_i, i \in \mathcal{S}]$ will refer to the matrix consisting of the columns of A that are indexed by $\mathcal{S}$, placed in natural (increasing) order.

Let $I = I_{n\times n}$, $\mathcal{U} \subseteq \{1, \ldots, n\}$ and $\mathcal{U}^c = \{1, \ldots, n\} \setminus \mathcal{U}$. The matrices $I_{\mathcal{U}}$ and $I_{\mathcal{U}^c}$ will be extensively used in Section IV to simplify notation. For any $A \in \mathbb{F}_q^{n\times k}$ (respectively, $A \in \mathbb{F}_q^{k\times n}$), the matrix $I_{\mathcal{U}}^T A$ (resp., $A I_{\mathcal{U}}$) extracts the rows (resp., columns) of A indexed by $\mathcal{U}$. Conversely, for any $B \in \mathbb{F}_q^{|\mathcal{U}|\times k}$ (resp., $B \in \mathbb{F}_q^{k\times|\mathcal{U}|}$) the matrix $I_{\mathcal{U}} B$ (resp., $B I_{\mathcal{U}}^T$) reallocates the rows (resp., columns) of B to the positions indexed by $\mathcal{U}$, where all-zero rows (resp., columns) are inserted at the positions indexed by $\mathcal{U}^c$. Finally, observe that $I_{\mathcal{U}}$ and $I_{\mathcal{U}^c}$ satisfy the following properties:

$$I = I_{\mathcal{U}} I_{\mathcal{U}}^T + I_{\mathcal{U}^c} I_{\mathcal{U}^c}^T, \tag{2}$$

$$I_{\mathcal{U}}^T I_{\mathcal{U}} = I_{|\mathcal{U}|\times|\mathcal{U}|} \tag{3}$$

$$I_{\mathcal{U}}^T I_{\mathcal{U}^c} = 0. \tag{4}$$

Let $X \in \mathbb{F}_q^{n\times m}$. We use $\langle X \rangle$ and $\mathsf{rank}\, X$ to denote, respectively, the row space and the rank of X. By definition, $\mathsf{rank}\, X = \dim \langle X \rangle$. An equivalent definition of rank is the following:

$$\mathsf{rank}\, X = \min_{\substack{\tau, L_1^\tau V_1^\tau : \\ X = L_1^\tau V_1^\tau}} \tau. \tag{5}$$

It is well-known that, for any $X, Y \in \mathbb{F}_q^{n\times m}$, we have

$$\mathsf{rank}(X + Y) \leq \mathsf{rank}\, X + \mathsf{rank}\, Y. \tag{6}$$

Proposition 1: For any $X \in \mathbb{F}_q^{n\times m}$ and $Y \in \mathbb{F}_q^{k\times m}$,

$$\mathsf{rank} \begin{bmatrix} Y \\ X \end{bmatrix} = \mathsf{rank}\, X + \mathsf{rank}\, Y - \dim \langle X \rangle \cap \langle Y \rangle \tag{7}$$

$$= \mathsf{rank}\, X + \min_{A \in \mathbb{F}_q^{k\times n}} \mathsf{rank}(Y - AX). \tag{8}$$

Proposition 2: For any $X \in \mathbb{F}_q^{n\times m}$, $Y \in \mathbb{F}_q^{k\times m}$ and $r \leq k$,

$$\min_{\substack{A \in \mathbb{F}_q^{k\times n}: \\ \mathsf{rank}\, A \geq r}} \mathsf{rank}(Y - AX) = \mathsf{rank} \begin{bmatrix} Y \\ X \end{bmatrix} - \mathsf{rank}\, X + c_r(X, Y) \tag{9}$$

where $c_r(X, Y) = \max\{r - n + \mathsf{rank}\, X - \mathsf{rank}\, Y, 0\}$.

A *matrix code* $\mathcal{C}$ is defined as any subset of $\mathbb{F}_q^{n\times m}$. A matrix code is also commonly known as an *array code* when it forms a linear space over $\mathbb{F}_q$ [12]. It follows from (6) that the following distance function is a metric [7]:

Definition 1: The *rank distance* between matrices $\boldsymbol{a}, \boldsymbol{b} \in \mathbb{F}_q^{n\times m}$ is defined as $d_R(\boldsymbol{a}, \boldsymbol{b}) = \mathsf{rank}(\boldsymbol{b} - \boldsymbol{a})$.

A *rank-metric code* is any matrix code used in the context of the rank distance metric. The minimum distance of a rank metric code is the minimum rank distance of the code among all pairs of codewords.

Consider a rank-metric code $\mathcal{C} \subseteq \mathbb{F}_q^{n\times m}$ and let $\boldsymbol{c}$ be a codeword in $\mathcal{C}$. Suppose an error matrix $\boldsymbol{e} \in \mathbb{F}_q^{n\times m}$ is added to $\boldsymbol{c}$ and the matrix $\boldsymbol{r} = \boldsymbol{c} + \boldsymbol{e}$ is received. The standard rank decoding problem is to find a codeword $\hat{\boldsymbol{c}} \in \mathcal{C}$ that minimizes the rank distance between $\boldsymbol{r}$ and $\hat{\boldsymbol{c}}$, that is,

$$\hat{\boldsymbol{c}} = \operatorname*{argmin}_{\boldsymbol{c} \in \mathcal{C}} \mathsf{rank}(\boldsymbol{r} - \boldsymbol{c}). \tag{10}$$

Now, if the error matrix $\boldsymbol{e} = \boldsymbol{r} - \boldsymbol{c}$ has rank τ, we can use (5) and (1) to write

$$\boldsymbol{e} = L_1^\tau V_1^\tau = \sum_{j=1}^{\tau} L_j V_j \tag{11}$$

where the vectors $L_1, \ldots, L_\tau$ and $V_1, \ldots, V_\tau$ are as defined above.

Vectors V_j and L_j can be thought of as giving the *value* and the *location*, respectively, of the jth error component. Namely, the error value V_j appears (multiplied by the coefficient L_{ij}) in every row i of $\boldsymbol{e}$ for which L_{ij} is nonzero.

Note that if $m = 1$ and L_j is a unit vector (that is, a zero/one vector with a single one) for all j, then this rank-metric description of the error naturally reduces to that of the Hamming metric. Thus, rank decoding can be seen as a generalization of minimum Hamming distance decoding. Unlike the classical case, however, the description of the error matrix in terms of general L_j and V_j is not unique. Namely, if $\boldsymbol{e} = L_1^\tau V_1^\tau$, then we also have $\boldsymbol{e} = (L_1^\tau T)(T^{-1} V_1^\tau)$ for any nonsingular matrix $T \in \mathbb{F}_q^{\tau\times\tau}$.

III. Error Correction in Random Network Coding

We consider a point-to-point communication network with a single source node and single destination node. The source node selects a message w from a set $\mathcal{W}$ and encodes this message into n packets $X_1, \ldots, X_n \in \mathbb{F}_q^M$, which are regarded as the incoming packets to the source node. Each node in the network, including the source, performs standard distributed network coding [1]: whenever a node has a transmission opportunity, it produces an outgoing packet as a random $\mathbb{F}_q$-linear combination of all the incoming packets it has until then received. The destination node collects N packets $Y_1, \ldots, Y_N \in \mathbb{F}_q^M$ and decodes these packets into a estimate $\hat{w} \in \mathcal{W}$ of the original message. The decoding is successful whenever $\hat{w} = w$.

Let X be an $n \times M$ matrix whose rows are the transmitted packets $X_1, \ldots, X_n$ and, similarly, let Y be an $N \times M$ matrix whose rows are the received packets $Y_1, \ldots, Y_N$. Since all packet operations are linear over $\mathbb{F}_q$, then, regardless of the network topology, the transmitted and received packets can be related by the following expression:

$$Y = AX, \tag{12}$$

where A is an $N \times n$ matrix corresponding to the overall linear transformation applied by the network (note that any linear packet operations performed at the source node are considered part of the network).

Before proceeding, we note that this model encompasses a variety of situations:

- The network may have cycles or delays. Since the overall system is linear, expression (12) will be true regardless of the network topology.
- The network could be wireless instead of wired. In this case, we simply constrain each intermediate node to send exactly the same packet in each of its outgoing links.
- The source node may want to transmit more than one message. In this case, we assume that each packet carries an index of the message to which it corresponds and that packets with different message indices are processed separately throughout the network [13].
- The network topology may exhibit time-variance as nodes join and leave and connections are established and lost. In this case, the model can still be preserved by considering each link to be the instantiation of a successful packet transmission.
- The network may operate in multicast mode, i.e., there may be more than one destination node. Again, expression (12) still applies, where the matrix A may be different for each destination.

Let us now extend this model to incorporate packet errors. Suppose that at the input of each link, a corrupting packet may be added to the packet being transmitted at that link. Let $Z_i \in \mathbb{F}_q^M$ denote the corrupting packet applied at the input of link i, where we assume that the links are indexed from 1 to L. By linearity of the network, we can write

$$Y = AX + BZ \tag{13}$$

where Z is an $L \times M$ matrix whose rows are $Z_1, \ldots, Z_L$, and B is an $N \times L$ matrix corresponding to the overall linear transformation incurred by the corrupting packets until the destination.

Observe that this model can represent not only the occurrence of random link errors but also the action of malicious nodes, in the following way. We assume that each node, malicious or not, creates a prescribed outgoing packet as a random linear combination of incoming packets, as described above. A non-malicious node then simply transmits this prescribed packet as its outgoing packet. A malicious node may either operate as a non-malicious node or may replace this prescribed packet by one of the following: an arbitrary linear combination of the incoming packets or an arbitrary packet which may not be a linear combination of the incoming packets. Refusing to transmit a packet corresponds to sending the trivial linear combination, i.e., an all-zero packet. Note that all these operations can be represented in the model as the addition of a corrupting packet to the prescribed outgoing packet, thus (13) holds. The number of nonzero rows of Z gives exactly the number of "packet interventions" performed by a malicious node and thus give a sense of the "power" employed by this node towards jamming the network.

Let $\mathrm{Enc}\colon \mathcal{W} \to \mathbb{F}_q^{n \times M}$ be the encoding function applied by the source node. Consider the decoding operation at the destination node. Performing maximum-likelihood decoding of the message w given the received matrix Y would require knowledge of the statistics of A, B and Z, which may not be available or may be too hard to find. A more modest goal may be to find a message that minimizes the number of corrupting packets introduced in the network, owing to the assumption that adversarial power is somehow limited or costly.

If A and B are known to the destination node, then we may define the *network decoding* problem as the problem of finding

$$\hat{w} = \operatorname*{argmin}_{w \in \mathcal{W}} \min_{\substack{Z: \\ Y = AX + BZ \\ X = \mathrm{Enc}(w)}} \mathrm{wt}(Z) \tag{14}$$

where $\mathrm{wt}(Z)$ denotes the number of nonzero rows of Z. A typical situation where this problem arises is when both the network code and the network topology are deterministic and known to the destination node [2], [3].

Here, we are interested in encoding and decoding functions that operate under no assumptions on the matrices A and B (except possibly some constraint on the rank of A). Any valid explanation of the received Y in terms of X, A, B and Z is conceivable, and the decoder attempts to find one that minimizes $\mathrm{wt}(Z)$. Thus, we define the *random network decoding* problem as the problem of finding

$$\hat{w} = \operatorname*{argmin}_{w \in \mathcal{W}} \min_{\substack{A,B,Z: \\ Y = AX + BZ \\ X = \mathrm{Enc}(w) \\ \mathrm{rank}\, A \geq r}} \mathrm{wt}(Z) \tag{15}$$

where $r \leq N$ is some lower bound on the rank of A.

The ability to range over all possible A and B actually facilitates the problem, since we can now find the value of the inner minimization in (15).

Theorem 3: Let $X \in \mathbb{F}_q^{n\times M}$, $Y \in \mathbb{F}_q^{N\times M}$, $Z \in \mathbb{F}_q^{L\times M}$, $A \in \mathbb{F}_q^{N\times n}$ and $B \in \mathbb{F}_q^{N\times L}$, where n, N, M and $L \geq N$ are positive integers. Let $\mathcal{W}$ be a finite set and $\mathsf{Enc}\colon \mathcal{W} \to \mathbb{F}_q^{n\times M}$. The random network decoding problem (15) can be reformulated as

$$\hat{w} = \underset{w\in\mathcal{W}}{\operatorname{argmin}}\ \mathsf{rank}\begin{bmatrix}Y\\X\end{bmatrix} - \mathsf{rank}\, X + c_r(X,Y)\Big|_{X=\mathsf{Enc}(w)} \tag{16}$$

where $c_r(X,Y) = \max\{r - n + \mathsf{rank}\, X - \mathsf{rank}\, Y, 0\}$.

From Theorem 3, we observe that performing any elementary row operations in X or Y does not change the decoding problem. Thus, from the point of view of the decoder, what is transmitted is not the actual matrix X, but only the row space of X. Likewise, it is the row space of Y what is effectively received by the destination node. This observation provides a close connection between the random network decoding problem proposed in this paper and the coding theory for subspaces introduced in [5].

Observe also that, if $\mathsf{rank}\, X$ is a constant for all valid X, then the term $c_r(X,Y)$ does not depend on the transmitted message and can be omitted from the minimization.

From now on, we can assume that $\mathsf{rank}\, Y = N$, since only a basis for $\langle Y\rangle$ needs to be considered as the received matrix. In practice, this means that any linear dependent received packet may be safely discarded by the destination node.

The approach we propose is this paper is characterized by choosing the message set as a rank-metric code $\mathcal{W} = \mathcal{C} \subseteq \mathbb{F}_q^{n\times m}$ and setting $\mathsf{Enc}(\boldsymbol{x}) = [I \quad \boldsymbol{x}]$ for all $\boldsymbol{x} \in \mathcal{C}$, where we assume $M = n + m \geq n$.

Note that, in the error-free case, we may chose $\mathcal{C} = \mathbb{F}_q^{n\times m}$ and obtain exactly the standard random network coding approach [1], [13]. The vectors $x_1, \ldots, x_n$ may be interpreted as *data packets*, and each of the transmitted packets $X_1, \ldots, X_n$ is formed by appending a *header* at the beginning of the corresponding data packet, so that $X = [I \quad \boldsymbol{x}]$. The received matrix will then be given by $Y = [A \quad A\boldsymbol{x}]$, from which $\boldsymbol{x}$ can be recovered if A has rank n.

Proposition 4: Let $\mathcal{W} = \mathcal{C} \subseteq \mathbb{F}_q^{n\times m}$ and $\mathsf{Enc}(\boldsymbol{x}) = [I \quad \boldsymbol{x}]$ for all $\boldsymbol{x} \in \mathcal{C}$. If $Y = [\hat{A} \quad \boldsymbol{y}]$, then the random network decoding problem (16) becomes the problem of finding

$$\hat{\boldsymbol{x}} = \underset{\boldsymbol{x}\in\mathcal{C}}{\operatorname{argmin}}\ \mathsf{rank}(\boldsymbol{y} - \hat{A}\boldsymbol{x}). \tag{17}$$

Note that if $\hat{A}$ is square and invertible, then $\mathsf{rank}(\boldsymbol{y} - \hat{A}\boldsymbol{x}) = \mathsf{rank}(\hat{A}^{-1}\boldsymbol{y} - \boldsymbol{x}) = \mathsf{rank}(\boldsymbol{r} - \boldsymbol{x})$, where $\boldsymbol{r} = \hat{A}^{-1}\boldsymbol{y}$. In this case, we obtain the conventional rank decoding problem of finding a codeword $\boldsymbol{x} \in \mathcal{C}$ that is closest in rank distance to $\boldsymbol{r}$. Thus, at least in this case, it is clear that the rank distance is the "right" metric for this problem, and that we should use a rank-metric code with large minimum rank distance.

In the case where $\hat{A}$ is not invertible, a general approach could be to define a new code $\mathcal{C}' = \hat{A}\mathcal{C} = \{\hat{A}\boldsymbol{x},\ \boldsymbol{x} \in \mathcal{C}\}$ and find a codeword $\boldsymbol{x}' \in \mathcal{C}'$ that has the smallest rank distance to $\boldsymbol{r}$. Then, any pre-image of $\boldsymbol{x}'$ in $\mathcal{C}$ will be a solution to the random network decoding problem. This approach, however, has the inconvenience that a new code would have to be used at each decoding instance, which may lead to decoding inefficiency (an efficient decoder for the code $\mathcal{C}'$ may not even be known). Instead, we would like to find a decoding problem where the structure of $\mathcal{C}$ could still be exploited. This is the subject of the next section.

IV. A Generalized Rank Decoding Problem

In this section we explore the case when $\hat{A}$ is not square and invertible.

Define μ and δ such that $\mathsf{rank}\, \hat{A} = n - \mu$ and $N = n - \mu + \delta$. Choose an $N \times N$ nonsingular matrix $T = \begin{bmatrix}T_1\\T_2\end{bmatrix}$, where T_1 and T_2 have $n - \mu$ and δ rows, respectively, such that $\mathsf{rank}\, T_1\hat{A} = n - \mu$ and $T_2\hat{A} = 0$. Such a matrix T can be found by performing Gaussian elimination on $\hat{A}$. Note that, since $\mathsf{rank}\, TY = N$, we must have $\mathsf{rank}\, T_2\boldsymbol{y} = \delta$.

We can now rewrite our objective function as

$$\mathsf{rank}(\boldsymbol{y} - \hat{A}\boldsymbol{x}) = \mathsf{rank}\, T(\boldsymbol{y} - \hat{A}\boldsymbol{x}) = \mathsf{rank}\begin{bmatrix}T_1\boldsymbol{y} - T_1\hat{A}\boldsymbol{x}\\ \hat{V}\end{bmatrix} \tag{18}$$

where $\hat{V} = T_2\boldsymbol{y}$.

Let us first examine the case where $\mu = 0$, i.e., $\mathsf{rank}\, \hat{A} = n$. We can choose T_1 such that $T_1\hat{A} = I$ and obtain

$$\mathsf{rank}\begin{bmatrix}T_1\boldsymbol{y} - T_1\hat{A}\boldsymbol{x}\\ \hat{V}\end{bmatrix} = \mathsf{rank}\begin{bmatrix}\boldsymbol{r} - \boldsymbol{x}\\ \hat{V}\end{bmatrix} \tag{19}$$

where $\boldsymbol{r} = T_1\boldsymbol{y}$.

From (8), we obtain

$$\mathsf{rank}\begin{bmatrix}\boldsymbol{r} - \boldsymbol{x}\\ \hat{V}\end{bmatrix} = \delta + \min_{L\in\mathbb{F}_q^{n\times\delta}} \mathsf{rank}(\boldsymbol{r} - \boldsymbol{x} - L\hat{V}). \tag{20}$$

Thus, we obtain a problem very similar to minimizing the rank distance between $\boldsymbol{r}$ and $\boldsymbol{x}$, except for the presence of the term $L\hat{V}$. Roughly speaking, this means that any rank difference that could be "explained away" by $\hat{V}$ is not counted in the rank distance.

Let us now proceed to the general case. Similarly as above, we would like to choose T_1 in such a way that the resulting decoding problem is as close as possible to the standard decoding problem in $\mathcal{C}$. Since, for $\mu > 0$, we cannot make $T_1\hat{A} = I$, we will choose T_1 so that $T_1\hat{A}$ is as close as possible to an identity matrix. Let $\mathcal{U}^c$ be the set of indices of some $n - \mu$ linear independent columns of $\hat{A}$, and let $\mathcal{U} = \{1, \ldots, n\}\backslash\mathcal{U}^c$. We choose T_1 such that the columns of $T_1\hat{A}$ indexed by $\mathcal{U}^c$ form an identity submatrix, i.e., $T_1\hat{A}I_{\mathcal{U}^c} = I_{(n-\mu)\times(n-\mu)}$. The remaining columns of $T_1\hat{A}$ are given in $T_1\hat{A}I_{\mathcal{U}}$. We record this information in the matrix

$$\hat{L} = I_{\mathcal{U}} - I_{\mathcal{U}^c}T_1\hat{A}I_{\mathcal{U}} \tag{21}$$

which possess the property $I_{\mathcal{U}}^T\hat{L} = I_{\mu\times\mu}$ (in particular, $\mathsf{rank}\, \hat{L} = \mu$). Using properties of the matrices $I_{\mathcal{U}}$ and $I_{\mathcal{U}^c}$,

it is easy to verify by direct substitution that $T_1\hat{A}$ can be recovered from $\hat{L}$ as

$$T_1\hat{A} = I_{\mathcal{U}^c}^T(I - \hat{L}I_{\mathcal{U}}^T). \tag{22}$$

Define $\boldsymbol{r} = I_{\mathcal{U}^c}T_1\boldsymbol{y}$, so that $I_{\mathcal{U}^c}^T\boldsymbol{r} = T_1\boldsymbol{y}$ and $I_{\mathcal{U}}^T\boldsymbol{r} = 0$. It follows that $I_{\mathcal{U}^c}^T(I - \hat{L}I_{\mathcal{U}}^T)\boldsymbol{r} = T_1\boldsymbol{y}$. We can now rewrite (18) as

$$\mathsf{rank}\begin{bmatrix} T_1\boldsymbol{y} - T_1\hat{A}\boldsymbol{x} \\ \hat{V} \end{bmatrix} = \mathsf{rank}\begin{bmatrix} I_{\mathcal{U}^c}^T(I - \hat{L}I_{\mathcal{U}}^T)(\boldsymbol{r} - \boldsymbol{x}) \\ \hat{V} \end{bmatrix} \tag{23}$$

$$= \mathsf{rank}\begin{bmatrix} (I - \hat{L}I_{\mathcal{U}}^T)(\boldsymbol{r} - \boldsymbol{x}) \\ \hat{V} \end{bmatrix} \tag{24}$$

where the last equality follows from $I_{\mathcal{U}}^T(I - \hat{L}I_{\mathcal{U}}^T) = 0$.

We have obtained an expression that is very similar to a rank distance between $\boldsymbol{r}$ and $\boldsymbol{x}$. The precise relationship between (24) and the rank of $\boldsymbol{r} - \boldsymbol{x}$ will be established in the following definition and the subsequent lemma.

Definition 2: Let $\boldsymbol{e} \in \mathbb{F}_q^{n\times m}$, $\hat{L} \in \mathbb{F}_q^{n\times\mu}$ and $\hat{V} \in \mathbb{F}_q^{\delta\times m}$. The *rank* of $\boldsymbol{e}$ *given* $\hat{L}$ and $\hat{V}$, denoted by $\mathsf{rank}(\boldsymbol{e}|\hat{L},\hat{V})$, is defined as

$$\mathsf{rank}(\boldsymbol{e}|\hat{L},\hat{V}) \triangleq \min_{\substack{\tau, L_1^\tau, V_1^\tau: \\ \boldsymbol{e}=L_1^\tau V_1^\tau \\ L_1^\mu = \hat{L},\, V_{\mu+1}^{\mu+\delta} = \hat{V}}} \tau. \tag{25}$$

If $(\tau, L_1^\tau, V_1^\tau)$ is a solution to the above problem, then the decomposition $\boldsymbol{e} = L_1^\tau V_1^\tau$ will be called a *rank decomposition of* $\boldsymbol{e}$ *consistent with* $\hat{L}$ *and* $\hat{V}$.

Lemma 5: Let $\mathcal{U} \subseteq \{1,\ldots,n\}$, $\hat{L} \in \mathbb{F}_q^{n\times\mu}$ and $\hat{V} \in \mathbb{F}_q^{\delta\times m}$ be such that $|\mathcal{U}| = \mu$, $I_{\mathcal{U}}^T\hat{L} = I_{\mu\times\mu}$ and $\mathsf{rank}\,\hat{V} = \delta$. Then, for any $\boldsymbol{e} \in \mathbb{F}_q^{n\times m}$,

$$\mathsf{rank}\begin{bmatrix} (I - \hat{L}I_{\mathcal{U}}^T)\boldsymbol{e} \\ \hat{V} \end{bmatrix} = -\mu + \mathsf{rank}(\boldsymbol{e}|\hat{L},\hat{V}). \tag{26}$$

From Lemma 5, we see that the random network decoding problem (17) can be restated as

$$\hat{\boldsymbol{x}} = \underset{\boldsymbol{x}\in\mathcal{C}}{\operatorname{argmin}}\, \mathsf{rank}(\boldsymbol{r} - \boldsymbol{x}|\hat{L},\hat{V}). \tag{27}$$

If the error matrix $\boldsymbol{e} = \boldsymbol{r} - \boldsymbol{x}$ can be rank-decomposed into $\boldsymbol{e} = L_1^\tau V_1^\tau$ consistently with $\hat{L}$ and $\hat{V}$, then we will say that μ *erasures*, δ *deviations* and $\epsilon = \tau - \mu - \delta$ *errors* have occurred. The components L_jV_j, $j = 1,\ldots,\mu$, correspond to the *erasures*, the components L_jV_j, $j = \mu+1,\ldots,\delta$, correspond to the *deviations*, while the remaining components L_jV_j, $j = \mu+\delta+1,\ldots,\tau$, correspond to the (unknown) *errors*.

Proposition 6: A rank-metric code $\mathcal{C} \subseteq \mathbb{F}_q^{n\times m}$ of minimum rank distance d is able to correct any pattern of ϵ errors, μ erasures and δ deviations if and only if $2\epsilon + \mu + \delta \le d - 1$.

We summarize the results of this section in the following theorem, which is the main theorem of this paper.

Theorem 7: Let $\mathcal{C} \subseteq \mathbb{F}_q^{n\times m}$ be a code of minimum rank distance d, and let $\hat{A} \in \mathbb{F}_q^{N\times n}$ and $\boldsymbol{y} \in \mathbb{F}_q^{N\times m}$ be such that

$$\mathsf{rank}[\hat{A} \quad \boldsymbol{y}] = N. \tag{28}$$

Set $\mu = n - \mathsf{rank}\,\hat{A}$ and $\delta = N - (n - \mu)$. For any $T_1 \in \mathbb{F}_q^{(n-\mu)\times N}$, $T_2 \in \mathbb{F}_q^{\delta\times N}$ and $\mathcal{U} \subseteq \{1,\ldots,n\}$ such that

$$T_1\hat{A}I_{\mathcal{U}^c} = I_{(n-\mu)\times(n-\mu)} \tag{29}$$

$$T_2\hat{A} = 0 \tag{30}$$

$$\mathsf{rank}\begin{bmatrix} T_1 \\ T_2 \end{bmatrix} = N \tag{31}$$

define $\boldsymbol{r} = I_{\mathcal{U}^c}T_1\boldsymbol{y}$, $\hat{L} = (I - I_{\mathcal{U}^c}T_1\hat{A})I_{\mathcal{U}}$, and $\hat{V} = T_2\boldsymbol{y}$. Then a solution to the random network decoding problem (17) is given by

$$\hat{\boldsymbol{x}} = \underset{\boldsymbol{x}\in\mathcal{C}}{\operatorname{argmin}}\, \mathsf{rank}(\boldsymbol{r} - \boldsymbol{x}|\hat{L},\hat{V}). \tag{32}$$

This solution is guaranteed to be unique if $\mathsf{rank}(\boldsymbol{r} - \hat{\boldsymbol{x}}|\hat{L},\hat{V}) \le (d - 1 + \mu + \delta)/2$.

The problem (32) given in Theorem 7 will be called *rank decoding with errors, erasures and deviations*, or simply *rank decoding*. Note that it reduces to the standard rank decoding problem (10) when $\mu = \delta = 0$.

Finally we note that certain rank decoding problems with "row and column erasures" have been previously proposed in the literature [8]–[10]. and correspond, respectively, to the cases where $\hat{V}_{\mu+1},\ldots,\hat{V}_{\mu+\delta}$ are unit row vectors and where $\hat{L}_1,\ldots,\hat{L}_\mu$ are unit column vectors. Thus, the rank decoding problem we propose is a strict generalization of the previous ones.

References

[1] T. Ho, M. Medard, R. Koetter, D. Karger, M. Effros, J. Shi, and B. Leong, "A random linear network coding approach to multicast," *IEEE Trans. on Inform. Theory*, vol. 52, pp. 4413–4430, Oct. 2006.

[2] Z. Zhang, "Network error correction coding in packetized networks," in *Proc. of 2006 IEEE Inform. Theory Workshop*, 2006.

[3] S. Yang and R. W. Yeung, "Characterizations of network error correction/detection and erasure correction," in *Proc. NetCod*, 2007.

[4] S. Jaggi, M. Langberg, S. Katti, T. Ho, D. Katabi, and M. Médard, "Resilient network coding in the presence of Byzantine adversaries," in *Proc. 26th Annual IEEE Conf. on Computer Commun.*, Anchorage, AK, May 2007.

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

[6] P. Delsarte, "Bilinear forms over a finite field, with applications to coding theory," *J. of Comb. Theory: Series A*, vol. 25, pp. 226–241, 1978.

[7] E. M. Gabidulin, "Theory of codes with maximum rank distance," *Probl. Inform. Transm*, vol. 21, no. 1, pp. 1–12, 1985.

[8] E. Gabidulin and N. Pilipchuk, "A new method of erasure correction by rank codes," in *Proc. IEEE Int. Symp. Information Theory*, 29 June–4 July 2003, p. 423.

[9] G. Richter and S. Plass, "Error and erasure decoding of rank-codes with a modified Berlekamp-Massey algorithm," in *Proc. of ITG Conference on Source and Channel Coding*, Erlangen, Germany, January 2004.

[10] ——, "Fast decoding of rank-codes with rank errors and column erasures," in *Proc. IEEE Int. Symp. Information Theory*, 27 June-2 July 2004, pp. 398–398.

[11] D. Silva, F. R. Kschischang, and R. Koetter, "Using rank-metric codes for error correction in random network coding," 2007, in preparation.

[12] R. M. Roth, "Maximum-rank array codes and their application to crisscross error correction," *IEEE Trans. on Inform. Theory*, vol. 37, pp. 328–336, 1991.

[13] P. A. Chou, Y. Wu, and K. Jain, "Practical network coding," in *Allerton Conf. on Comm., Control, and Computing*, Monticello, IL, October 2003.

Network error correction coding provides information theoretic security against arbitrary non-ergodic errors in a network. As one part of this project we investigated network error correction for multiple-source multicast as well as non-multicast, generalizing previous results in the literature on single-source multicast.

For multiple-source multicast, we considered the coherent case where the network topology is known, as well as the noncoherent case where random linear coding is done over an unknown network topology. For both cases, we obtained the capacity region of reliable transmission rates under arbitrary errors on up to z links in the network, as

$$\sum_{i \in \mathcal{S}} r_i \leq m_{\mathcal{S}} - 2z \; \forall \text{ subsets of sources } \mathcal{S}$$

where r_i denotes the rate of source i, and $m_{\mathcal{S}}$ denotes the minimum cut capacity between the subset of sources $\mathcal{S}$ and any sink. For noncoherent coding, the region is asymptotically achievable with high probability over the random network code, as packet length grows. Unlike the single-source case where network coding distance arguments suffice to show achievability of the capacity, in the multiple-source case we additionally rely on the generic nature of the random network code in linearly combining packets from different sources.

For non-multicast, finding the capacity region of a general network even in the error-free case is an open problem. Thus, we considered the problem of constructing a network error correction code from a given error-free network code. Given any linear network code that achieves rate vector $(r_1, r_2, \ldots, r_n)$, where r_i is the transmission rate from source $i = 1, \ldots, n$ to its set of sink nodes $\mathcal{T}_i$, we can obtain a network code that achieves rate vector $(r_1 - 2z, r_2 - 2z, \ldots, r_n - 2z)$ under arbitrary errors on up to z links in the network.

Another problem investigated in this project considered network error and erasure correction coding under non-worst-case models of error and erasure locations, in contrast to existing worst-case models which only consider the number of errors and erasures. In the latter case, it is well-known that optimal worst-case performance is achievable with random linear coding at every node. On the other hand, for randomly located errors and erasures we showed that the relative benefit of coding versus routing in the network depends on the relative occurrence of errors and erasures and the network topology, through theoretical analysis of a family of simple network subgraphs consisting of multiple multi-hop paths, and simulation experiments on randomly generated hypergraphs. We also analyzed the relative benefit

of designing network codes for in-network decoding versus decoding at the sink, which depends on the network topology.

We also showed how back pressure routing/coding algorithms could be extended to various classes of network codes, including pairwise wireline network coding and one-hop wireless coding. Back pressure approaches make routing and coding decisions based on local queue length information, which provides robustness to networks that are changing ergodically or adversarially. We showed how to define virtual queues appropriately so as to efficiently optimize over different classes of network codes.