This full text paper was peer reviewed at the direction of IEEE Communications Society subject matter experts for publication in the IEEE Globecom 2011 proceedings.

An Efficient Error-Detection and Error-Correction (EDEC) Scheme for Network Coding

Wenbo Qiao Jian Li Jian Ren
Department of Electrical and Computer Engineering
Michigan State University
East Lansing, MI 48824
Email: {qiaowenb, lijian6, renjian}@egr.msu.edu

***Abstract*—Network coding is being viewed to have the potential for significant throughput improvement in network environment. However, these expected benefits are very fragile to malicious attacks, including message block content corruption and node compromise attacks. To solve these problems, both pollution detection and pollution correction based schemes have been proposed. These schemes are only effective in some limited scenarios. In this paper, we propose a new scheme that combines the benefits of the existing error-detection and error-correction (EDEC) schemes. The proposed scheme is similar in structure to the existing error-control based schemes. However, by appropriately modifying the rate of the underlying error-control scheme, we can improve the network throughput and robustness significantly. Our scheme can detect the malicious attacks by computing whether the syndromes are all zeros. By collecting all the non-zero syndromes, the malicious attacks within the error-decoding capacity of the underlying linear network coding can be removed and the original message can be recovered. Our theoretical analysis and simulation results demonstrate that the proposed EDEC scheme can improve the overall network performance dramatically with only a very moderate increase of the computational overhead.**

I. Introduction

Network coding is a new communication diagram that is designed to improve the throughput and robustness in network environment. The core notation of network coding is that it allows the participating nodes to code incoming messages at intermediate network nodes in a way that when a receiver receives the messages, it can recover the originally messages. Network coding provides a trade-off between maximum multicast flow rate and computational complexity in directed networks.

Network coding was first introduced in the seminal paper by Ahlswede et al. [1]. For network coding to achieve the expected benefits, all the participating nodes in the network should be free of network pollution and malicious attacks. Currently the researches on combating network pollution and malicious attacks from the network nodes are mainly focused on linear network coding, which is introduced by Li et al. [2]. Koetter and Medard [3] have shown that linear codes are sufficient to achieve the multicast capacity by coding on a large enough field. Ho et al. [4] have shown that the use of random linear network coding is a more practical way to design linear codes. Gkantsidis and Rodriguez [5] have applied the principles of random network coding to the context of peer-to-peer (P2P) content distribution, and have shown that file downloading times can be reduced.

The linear network coding based schemes can largely be divided into error-detection based schemes and error-correction based schemes. For error-detection based schemes, the errors are normally detected at the intermediate forwarding nodes, while for error-correction based schemes, the errors are generally corrected at the sink node. While the error-correction based schemes seem to be more appealing, the complexity for encoding and decoding are relatively complex [6]. It also comes with high computational overhead. These schemes are generally designed based on the knowledge of the network topology, which makes these schemes less flexible to the current networks.

The limitations of error-correction based schemes make error-detection may seem to be attractive in some network scenarios. Krohn et al. propose to use homomorphic hash functions [7] to guarantee the correctness of network flow. The main idea is that each intermediate node will check the correctness of the messages. If a packet does not pass the check at an intermediate node, it will be discarded. This approach can reduce the communication overhead and can be used in random network coding. However, the computational complexity is still very high. When the network scale is large, computing too many hash values also creates high delay. To address all these defects, Kehdi and Li [8] developed a simple error-detection based *Null Keys* scheme. The main idea is to partition the n-dimensional linear space over $GF(q^n)$ into two orthogonal subspaces of dimension k (symbol subspace) and $n-k$ (null key space). Comparing to the homomorphic hash function, the Null Keys scheme is much more efficient and has virtually no message delay. Unfortunately, these schemes all have one weakness: all corrupted packets will be discarded. In packetized network, a large packet is divided into small fragments to transmit. As long as a malicious node can corrupt one fragment in the whole packet, according to the approach, this fragment will be discarded. In this way, the net transmission efficiency can be close to zero.

In this paper, we propose a new scheme that combines error-detection and error-correcting (EDEC) to combat network pollution attacks. According to the network condition (number

978-1-4244-9268-8/11/$26.00 ©2011 IEEE

of errors per symbol), we choose a proper coding parameter (n, k) for message encoding. When an intermediate node detects an error, instead of discarding the packet, it will continue to forward the message together with the non-zero syndromes. As long as the errors are within the decoding capacity, after gathering enough information, the subsequent nodes will be able to recover the corrupted packet. In this case, our proposed EDEC scheme can continuously handle small pollution attacks for the same packet, while Kehdi and Li's scheme cannot.

The major contributions of this paper are the following:

1) Our proposed EDEC scheme can improve the throughput and robustness over the Null Keys scheme significantly, while maintaining the security advantage of the Null Keys scheme.
2) The computational overhead of our proposed scheme is very moderate.
3) We provide comprehensive theoretical analysis of the proposed EDEC scheme and extensive simulation results to compare the proposed EDEC scheme with the Null Keys scheme.

The rest of this paper is organized as follows: Section II presents related work. Section III gives an overall of the preliminary. The proposed EDEC scheme is presented in Section IV. Section V offers performance analysis and simulation results. We conclude in Section VI.

II. Related Work

As we mentioned above, the related work in the network coding for pollution attack can be divided into error detection at the intermediate forwarding nodes and error-correction at the sink node. Cai and Yeung [6] propose to correct errors at sink nodes using error correcting network coding. They derived the Hamming bound and the Gilbert-Varshamov bound. Jaggi et al. [9] develop a two-part rate-region for their codes based on BEC channel codes. Although these approaches only require the sink node to correct the error, they cannot stop the pollution spreading in network, which could reduce the network throughput and also increase the computational overhead. Krohn et al. [7] propose to verify the messages at the intermediate nodes using homomorphic hash functions. Charles et al. [10] use the cryptographic idea to discard the corrupted packets. Kehdi and Li [8] propose the null keys algorithm. There are two orthogonal subspaces in their model, one for the legitimate symbols and the other for the illegitimate. Illegitimate symbols (called null keys) are distributed randomly in all the nodes. According to linear algebra, when we multiply two vectors from the two orthogonal subspaces, we can get a zero. So every node checks the incoming symbols by multiplying them with the null keys stored in it. If the incoming symbols are illegal, the results will not be zero. They also prove that the probability that a malicious node can forge a bogus symbol is very small under the randomized null key distribution. However, it is only an error detection approach, and if error always happens, it will fail in transmitting a packet. In our approach, we also use two orthogonal subspaces in the model to implement a certain error correction code. But we intend to correct the error according to the multiplication results (called syndrome in error control coding) of received symbols and "null keys". After we successfully collect enough syndromes from a series of intermediate nodes, we can recover the corrupted message if the errors are less than the amount we designed to correct.This can significantly compensate null keys algorithm's defect.

III. Overview of the Error-Detection and Error-Correction

In this section, we propose an error-correction based scheme. We use (n, k, d) to represent a binary linear code, with a generating matrix $G_{k\times n}$, where k is the dimension (also knows as the rank) of the codeword, and d is the minimum distance, defined as the minimum *Hamming distance* for any two distinct codewords $\mathbf{x}$ and $\mathbf{y}$ of C:

$$d_{\min} = \min\{d(\mathbf{x}, \mathbf{y}) | \mathbf{x}, \mathbf{y} \in C\}.$$

where

$$d(\mathbf{x}, \mathbf{y}) = \min |\{x_i \neq y_i | \forall\, \mathbf{x}, \mathbf{y} \in C\}|.$$

The number of total codewords is 2^k, which form a subspace of dimension k over the n dimensional space. All codewords are at least $d_{\min}$ distance apart. All the codewords $\mathbf{c}$ can be generated from a k-tuple message $\mathbf{m}, \mathbf{c} = \mathbf{m} \cdot G$.

For the generating matrix G, there exists a parity-check matrix, called $H_{(n-k)\times n}$, whose rows $\mathbf{h}$s are linearly independent and satisfying $c \cdot \mathbf{h} = \mathbf{0}$. The rows of H are called *Null Keys* in [8]. The set of linear independent Null Keys is $\{\mathbf{h}_1, \mathbf{h}_2, \cdots, \mathbf{h}_{n-k}\}$. For each codeword $\mathbf{c}$, we have

$$\mathbf{c} \cdot H^T = \mathbf{0}. \tag{1}$$

By checking messages at every node with equation (1), there is a high probability that the Null Keys scheme can detect the polluted messages after a few hops of transmission. However, the "check-and-dump" setup may have a very low efficiency when smart pollution exists. In this case, the communication efficiency may become very low.

Suppose $\mathbf{c}$ is the transmitted codeword, and $\mathbf{r} = \mathbf{c} + \mathbf{e}$ is the received codeword, where $\mathbf{e}$ is a n-tuple error message generated by a malicious node. For received word $\mathbf{r}$, according to equation (1), $\mathbf{c}$ is orthogonal to H, therefore, we have:

$$\mathbf{r} \cdot H^T = (\mathbf{c} + \mathbf{e}) \cdot H^T = \mathbf{0} + \mathbf{e} \cdot H^T = \mathbf{e} \cdot H^T. \tag{2}$$

Equation (2) is called the *syndrome* of error pattern $\mathbf{e}$, denoted as s. It is clear that $\mathbf{r}$ is a codeword if and only if $s = 0$. When $s \neq 0$, we have $e \neq 0$. The task of maximum likelihood decoding is to find the minimum weight error pattern $\mathbf{e}$ such that $\mathbf{r} \cdot H^T = \mathbf{e} \cdot H^T$. In this case, the received $\mathbf{r}$ is corrected to $\mathbf{r} + \mathbf{e} = \mathbf{c}$.

In the network, each node carries one to several null keys from H. For each null key $\mathbf{h}_i, i = 1, 2, \cdots, n-k$, we can derive its corresponding syndrome bits $s_i = \mathbf{r} \cdot \mathbf{h}_i$ for the received word $\mathbf{r}_i$ based on its own Null Keys. The node then forwards the non-zero syndrome(s) to the subsequent relay

nodes. Eventually, when an intermediate node is able to collect all the syndromes, based on the initial code selection, it is able to correct $\lfloor (d-1)/2 \rfloor$ errors using error-correcting techniques [11], and then identify the corresponding node(s) that have been polluted.

IV. Proposed Error-Detection and Error-Correction (EDEC) Scheme for Network Coding

In this paper, we use the notations of [3], [8]. We model the network as a directed random graph $G = (V, E)$, where V is the set of nodes in the network, and E is the set of connections. In the proposed algorithm, we assume that v_i is a node from V. For key distribution, homomorphic hashes are used to provide a high level of security. Because homomorphic hashes are only used for secure key distribution, the computation overhead for the system is very limited. The assigning keys process is totally random. Each node v_i will get m keys on average.

We consider a network where a source node S selects messages from an (n, k, d) code subspace C. We assume the generating matrix is G. C is closed to random linear combination of vectors from C.

The null space of the code subspace C, denoted as $C^{\perp}$, is the set of all vectors $\mathbf{h}$ for which $G \cdot \mathbf{h} = 0$. We have

$$dim(C) + dim(C^{\perp}) = n.$$

For a (n, k, d) linear block code C, there are 2^k codewords, which form a k-dimensional subspace of the vector space of all the n-tuples over $GF(2)$. A decoding scheme is a rule at to partition the 2^n possible received vectors into 2^k disjoint subsets. The partition is based on the linear structure of the code so that each subset will consist of 2^{n-k} vectors. In this way, we have the following results.

Theorem 1 ([11]). *Every (n, k) linear code is capable of detecting $2^n - 2^k$ error patterns. However, it is only capable of correcting 2^{n-k} error pattern.*

Theorem 2 (The Singleton bound [12]). *If C is an (n, k, d) linear code, then $k + d \leq n + 1$.*

For an (n, k, d) linear block code, any two distinct codewords with length n differ in d places. It is able to detect $d-1$ or fewer bits of errors.

We will now look at the relationships between minimum distance and the number of errors that can be corrected. Suppose X and Y are two adjacent codewords and the spheres centered at X and Y with radius γ are disjoint. This means $d(X, Y) \geq 2\gamma + 1$, where $d(X, Y)$ stands for the Hamming distance between codeword X and Y.

Assume that Z is a polluted vector of X and $d(X, Z) \leq \gamma$, then Z will be in the sphere centered at X with radius γ. Because of $d(X, Z) < \gamma$, we have

$$d(Z, Y) \geq 2\gamma + 1 - d(X, Z) > \gamma.$$

Therefore, all codewords with γ or less errors can be correctly decoded based on the minimum Hamming distance.

Theorem 3 ([11]). *A linear code is capable of correcting λ or fewer errors and simultaneously detecting τ $(\tau > \lambda)$ or fewer errors if its minimum distance $d_{\min} \geq \lambda + \tau + 1$.*

Proof: Since $\tau > \lambda$, we have $d_{\min} \geq \lambda + \tau + 1 > 2\lambda + 1$, we have $\lambda < \lfloor (d-1)/2 \rfloor$. So we can correct λ or fewer errors. Suppose the transmitted codeword is $\mathbf{c}$, the received word is $\mathbf{r}$, and the decoded codeword is $\mathbf{r}'$. If τ $(\tau > \lambda)$ errors occur, we have $d(\mathbf{c}, \mathbf{r}) = \tau$.

On the other hand, since the number of errors that can be corrected is at most λ, we have $d(\mathbf{r}, \mathbf{r}') = \tau$. $d(\mathbf{c}, \mathbf{r}') \leq d(\mathbf{c}, \mathbf{r}) + d(\mathbf{r}, \mathbf{r}') = \tau + \lambda$. Suppose the codeword is $\mathbf{w}$ when $d(\mathbf{c}, \mathbf{w}) = d_{\min}$. $d(\mathbf{r}', \mathbf{w}) \geq d(\mathbf{c}, \mathbf{w}) - d(\mathbf{c}, \mathbf{r}') \geq d_{\min} - \tau - \lambda$. If we want to successfully detect the error, we must have $d(\mathbf{r}', \mathbf{w}) > 0$. That is $d_{\min} - \tau + \lambda \geq 1$. So if $d_{\min} \geq \lambda + \tau + 1$, we can detect τ or fewer errors. ■

From Fig. 1 and above results, we can conclude that for a (n, k) block code C, the code C can correct error patterns up to γ bits if $d_{\min} \geq 2\gamma + 1$. Therefore, $\gamma \leq \lfloor (d_{\min} - 1)/2 \rfloor$.

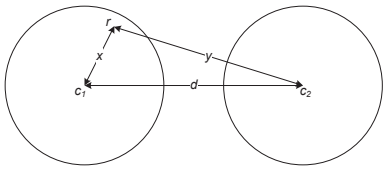

Fig. 1. An illustration of error correction and minimum distance.

Since we assume each node v_i will carry m keys on average, we are able to derive m syndromes at each node on average. If all the syndromes are zeros, the received codeword does not contain any error. In this case, our scheme works the same as the scheme proposed by Kehdl and Li in [8]. In case that some syndromes are non-zeros, our scheme differs from the Null Keys scheme in that the intermediate relay nodes need to forward the syndromes to the subsequent nodes for possible error-correction.

To minimize the message pollution from network propagating, we propose the error decoding to start right at the node when the last syndrome bit is derived. For practical purpose, we assume that the polluter has only limited power and can only corrupt a small network nodes. If the number of errors is less than or equal to $\lfloor (d-1)/2 \rfloor$, we are able to compute the minimum weight error pattern using maximum likelihood decoding algorithms [11] and recover the corrected transmitted codeword.

The error pattern also contains the location of the pollution. Based on the statistical analysis of the location of pollution, we distinguish an enroute pollution from a source node compromise pollution. In this way, we can eliminate the polluted source nodes so that the communication efficiency can be improved.

The key distribution, syndrome computation, and error-correction of the proposed EDEC scheme are described in Algorithm 1 below.

Algorithm 1 Proposed *EDEC* Scheme

Algorithm Key Distribution
At source node S
for $v_i \in V$ **do**
  $h_j^T \leftarrow rowvector(H)$
  for $j = 1$ **to** $n - k$ **do**
    if $rand(1) < m/(n-k)$ **then**
      $v_i \leftarrow h_j^T$
    end if
  end for
end for

Algorithm Error Correction
At each node v_i
for all *incoming message* r **do**
  $r \cdot \{h^T\} = \{s\}$
  if *the size of* $\{s\}$ *is* $n - k$ **then**
    $u' = decode(r)$
  end if
end for

V. Performance Evaluation and Simulation Results of Our Proposed EDEC Scheme

In this section, the efficiency of the proposed scheme will be analyzed using error-control techniques.

By adding redundancy to the k-bit long messages, the Null Keys scheme is able to detect errors of network coding. For an (n, k, d) linear block code, k/n is the code rate of the error-control code. A high k/n means a high code rate and a high communication efficiency. However, due to Theorem 1 and Theorem 2, when k is large, the minimum distance d and the number of errors that can be detected or corrected is relatively small. In fact, the number of errors that can be detected or corrected is $d - 1$ or $\lfloor (d-1)/2 \rfloor$, respectively.

We present our simulation results to test the performance and efficiency of *Error-Detection and Error-Correction (EDEC)* scheme. We compare the performance of our proposed scheme with the scheme proposed by Kehdi and Li in [8].

A. Illustrative Examples

We give two illustrative examples below to compare these two schemes, and also show how k impacts the overall transmission efficiency.

Example 1. *For a* $(6, 4)$ *code, if* $d_{\min} = 2$*, then it can detect all single error. Suppose we are transmitting 100 messages with 90% probability that 1-bit is polluted. From a probability point of view, 90% of the messages will be dropped by using Null Keys scheme. The overall efficiency of the Null Keys scheme is, therefore,* $4/6 \times 10\% = 6.67\%$.

If we add one more parity-check bit to the codeword, then the code become a $(7, 4, 3)$ *linear code. Then using our proposed EDEC scheme, we can detect any 2-bit errors, and correct any single bit error. If we are still transmitting 100 messages with 90% probability that 1-bit is polluted. Then, no message will be dropped at the sink node. Therefore, the overall efficiency is* $4/7 = 57.1\%$*. In this way, the overall efficiency can be increased by 8.57 times.*

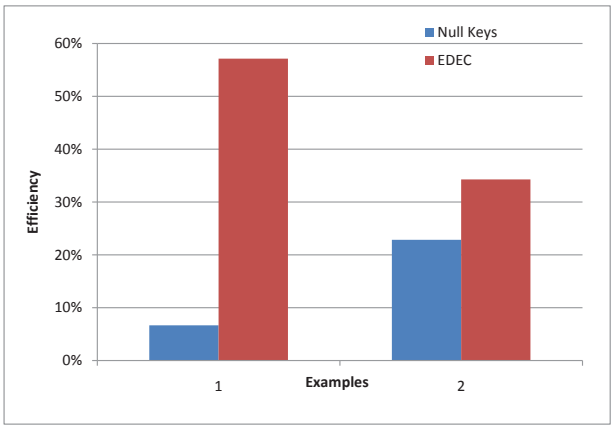

Fig. 2. Transmission Efficiency Comparison

Example 2. *Suppose we will use Null Keys scheme and our proposed EDEC scheme to transmit messages using a* $(8, 4, 4)$ *linear block code. Assume that during message transmission, 20% messages are polluted with 1-bit error, and 20% with 2-bit errors. The Null Keys scheme will drop all the 40% messages that have pollution, which makes the overall efficiency* $4/8 \times 60\% = 30\%$.

For our proposed EDEC scheme, since it is able to correct all the single error patterns, we only drop 20% of the messages that have 2 errors. The overall efficiency is, therefore, $4/8 \times 80\% = 40\%$.

B. Simulation Results

In our simulation, the same settings are used for both the Null Keys scheme and our proposed EDEC scheme.

We measure the cost of both schemes based on the number of hops each polluted message has to be forwarded. We assume that each node carries m linear independent random vectors from the null key subspace.

It is clear that increasing of m will result in a shorter time delay in computing all the required syndromes. However, a larger m means higher computation overhead and energy consuming.

Fig. 3 and Fig. 4 compare the throughput efficiency of our proposed EDEC scheme with the Null Keys based on the number of null keys m each node carries. Fig. 3 uses a $(24, 12, 8)$ Golay linear code. We can observe that at about 10 hops, the Null Keys algorithm can detect and dump all the polluted messages. With only 5 more steps, using our proposed EDEC scheme, we can correct all the errors. Through EDEC scheme, the throughput can be improved from 0% to 70%.

Fig. 4 is a $(8, 2, 5)$ linear code. We can see a similar trend: with 10 more hops of message forwarding, we can

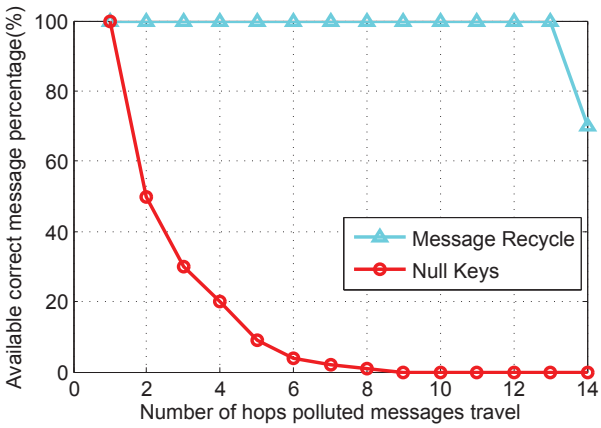

Fig. 3. Throughput efficiency comparison between Null Keys and EDEC for $(24, 12, 8)$ linear code: each node carries $m = 6$ null keys, 70% messages have been corrupted with 1 to 3 errors, and 30% with 4 error patterns.

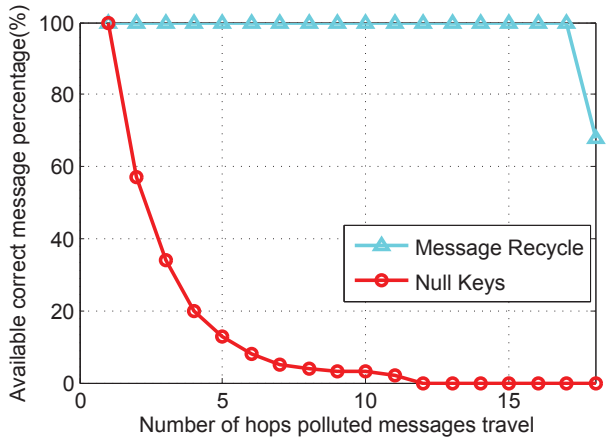

Fig. 4. Throughput efficiency comparison between Null Keys and EDEC for $(8, 2, 5)$ linear code: each node carries $m = 2$ null keys, 2/3 of the messages have been polluted with 1 error, and 1/3 with 2 or 3 errors.

eliminate all the pollution and recover the correct messages. The throughput can be improved from 0% to 66.7%.

In Fig. 5 we provide simulation results for a special code with parameters $n = 2^\alpha - 1, k = 2^\alpha - 1 - \alpha = n - \alpha$, and $d_{\min} = 3$, where $\alpha = \{5, 6, 7, 8, 9\}$. This kind of code can correct $\gamma = 1$ error.

Based on Theorem 2, for an (n, k, d) linear code of this kind, a smaller $n - k = \alpha$ corresponds to a larger k, and a more efficient linear code. In the simulation, we use m to denote the number of null keys that each node carries. The simulation shows the relationship between m and the average number of hops for a polluted message is corrected. We can also see the number of hops for the polluted messages to be corrected in the network decreases with the number n.

VI. Conclusion

In this paper, we have proposed an error-detection and error-correction (EDEC) scheme to deal with malicious attacks and node compromising in network coding. By appropriately

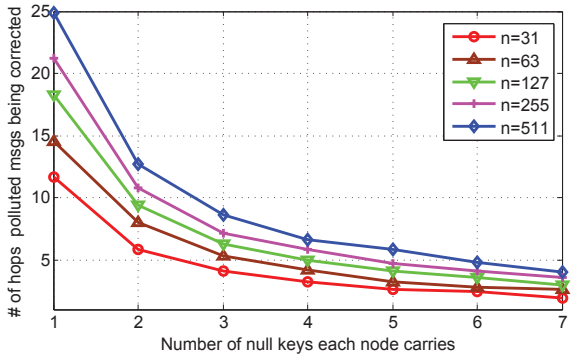

Fig. 5. Number of hops for polluted messages to be corrected for multiple code lengths.

selecting the linear network coding scheme, we can identify the network pollution to reduce the needs for messages to be dropped or resent, which can greatly improve the overall network performance. While significantly improving the network throughput and robustness, the computational complexity for pollution identification can be clearly computed based on the underlying error-correction algorithm of the linear code, which is very moderate. Our theoretical analysis and simulation results demonstrate that the proposed EDEC scheme can improve the throughput over the Null Keys dramatically in nearly all scenarios.

Acknowledgement

This research is was supported in part by the NSF under grants CNS-0845812, CNS-0848569 and CNS-1050326.

References

[1] R. Ahlswede, N. Cai, S.-Y. Li, and R. Yeung, "Network information flow," *IEEE Transactions on Information Theory*, vol. 46, pp. 1204–1216, July 2000.

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

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

[4] T. Ho, B. Leong, R. Koetter, M. Medard, M. Effros, and D. Karger, "Byzantine modification detection in multicast networks using randomized network coding," in *International Symposium on Information Theory (ISIT) 2004*, p. 144, July 2004.

[5] C. Gkantsidis and P. Rodriguez, "Cooperative security for network coding file distribution," in *IEEE INFOCOM 2006*, pp. 1–13, Apr. 2006.

[6] N. Cai and R. W. Yeung, "Network coding and error correction," in *Proc. of IEEE Information Theory Workshop (ITW 2002)*, pp. 119–122, 2002.

[7] M. Krohn, M. Freedman, and D. Mazieres, "On-the-fly verification of rateless erasure codes for efficient content distribution," in *IEEE Symposium on Security and Privacy 2004*, pp. 226–240, May 2004.

[8] E. Kehdi and B. Li, "Null keys: Limiting malicious attacks via null space properties of network coding," in *IEEE INFOCOM 2009*, pp. 1224–1232, Apr. 2009.

[9] S. Jaggi, M. Langberg, T. Ho, and M. Effros, "Correction of adversarial errors in networks," in *Proc. of International Symposium on Information Theory (ISIT 2005)*, pp. 1455–1459, 2005.

[10] D. Charles, K. Jain, and K. Lauter, "Signatures for network coding," in *Proc. of CISS'06*, pp. 857–863, 2006.

[11] S. Lin and D. J. Costello, *Error Control Coding*. Prentice Hall, 2nd ed., June 2004.

[12] F. J. MacWilliams and N. J. A. Sloane, *The Theorey of Error-Correcting Codes*. North-Holland, 1977.