Insecure “Provable Secure Network Coding”

Yongge Wang
UNC Charlotte, USA
yonwang@uncc.edu

October 18, 2009

Abstract

Network coding allows the routers to mix the received information before forwarding them to the next nodes. Though this information mixing has been proven to maximize network throughput, it also introduces security challenges such as pollution attacks. A malicious node could insert a malicious packet into the system and this corrupted packet will propagate more quickly than in traditional copy-and-forward networks. Several authors have studied secure network coding from both information theoretic and probabilistic viewpoints. In this paper, we show that there are serious flaws in several of these schemes (the security “proofs” for these schemes were presented in these publications).

1 Introduction

Maximum flow minimum cut (MFMC) theory has been one of the most important principles for network traffic routing. However, MFMC theorem only works if there is one sender and one receiver. When there are multiple receivers (multicast scenario), the maximum flow problems become NP-hard and there is no efficient way to multicast the same message to all receivers with maximum network capacity. Network coding [2] has been designed to overcome these problems and it has been shown that networking code can achieve better performance than traditional copy and forward networking technology. In particular, it has been shown that random linear code can be used to broadcast a message to multiple recipients with maximum network capacity with probabilistic reliability. Deterministic polynomial networking code has also been designed to achieve maximum network capacity.

Though network coding techniques have been extensively studied and mature techniques are now available for practical network coding, secure network coding has been poorly addressed. Without efficient techniques for reliable and private network coding, it is infeasible to widely deploy network coding techniques. Cai and Yeung [5, 7, 8] have proposed a general framework and obtained theoretical bounds for network error correction. Based on these theoretical bounds, Cai and Yeung [9] have designed algorithms for achieving network coding based secure communication against passive adversaries (wire tappers). Later, Jaggi, Langberg, Katti, Ho, Katabi, and Medard [12] studied network coding based secure communication techniques against Byzantine adversaries. It should be noted that in these two papers, the adversary model is based on the number of communication links that could be controlled by the adversary. This is very different from the more powerful model based on the number of nodes that could be controlled by the adversary. The link based adversary model may be realistic in wire based networks, it is unrealistic in wireless networks such as sensor networks [10]. Thus the application of these results could be limited.

Cryptographic techniques have also been used by researchers to protect network coding security against pollution attacks. For example, the following schemes have been proposed: Yu, Wei, Ramkumar, and Guan [18], Zhao, Kalker, Medard, and Han [19], Charles, Jain, and Lauter [6], and Boneh, Freeman, Katz and Waters [4]. In this paper, we will show that all these protocols except the last one are either completely insecure or non-practical for network coding.

Non-network-coding based perfectly secure message transmission techniques have been extensively studied in a series of papers. For example, Wang and Desmedt [17] have recently designed techniques for secure point-to-point communication in general networks with feedback channels against Byzantine adversaries.

2 Random Linear Network Coding

In this section, we briefly discuss the concept and notations of network coding. The network is modelled by a directed graph. There is a source node and several sink nodes. In network coding, the source node generates the data packets that she wishes to deliver to the sink nodes over the network. To do so, the source node encodes her data and transmits the encoded data via its outgoing edges, according to some encoding algorithm that we will discuss later. Each intermediate node receives data packets from its incoming edges, combines them by some encoding algorithms, and transmits the encoded data via its outgoing edges. Note that the node may transmits different data packets on different outgoing edges. The advantage of network coding is showed in the Figure 1 from [2]. In this Figure, we assume that the source node has two data packets A and B and wants to deliver them to the two sink nodes at the bottom. Assuming that all links have a capacity of one packet per unit of time, for traditional copy-and-forward network communication, there is no possibility for the source node to deliver these two packets to the two sink nodes in one unit time. However, if the upper intermediate node XORs the received packets and forward $A \oplus B$ to the middle link, both sink nodes obtain two distinct packets in every unit of time.

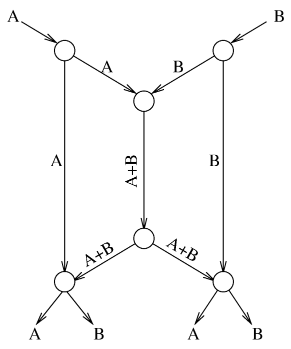

Figure 1: Network coding

Network coding has been extensively studied by researchers in the past few years. The works in [11, 13, 15, 16] show that simple random linear coding is sufficient for achieving maximum capacity bounds in multicast traffic. In the following, we introduce notations for the random linear coding.

Without loss of generality, we assume that the source node generates the messages $w_1, \ldots, w_t \in F_p^{n-t}$, where F_p is the finite field. In another word, each w_i consists of $n-t$ elements from F_p. First, the source node pads the messages with the $t \times t$ identity matrix I as follows:

$$\left(\begin{array}{c} M_1 \\ M_2 \\ \cdots \\ M_t \end{array}\right) = \left(\begin{array}{c|c} w_1 & \\ w_2 & \\ \cdots & I \\ w_t & \end{array}\right)$$

Thus we can consider the messages as $M_1, \ldots, M_t \in F_p^n$.

For one message transmission session, each node with k incoming edges receives $v_1, \cdots, v_k \in F_p^n$ from its k incoming edges respectively. For each outgoing edge, the node chooses random $\alpha_1, \ldots, \alpha_k \in F_p$ and transmits $\alpha_1 v_1 + \cdots + \alpha_k v_k$ on this outgoing edge.

Without loss of generality, we also assume that there are t virtual nodes which transmits the values $M_1, \ldots, M_t$ to the source node. So the source node transmits random linear combinations of these messages to its outgoing edges instead of the original messages.

Note that if one sink node receives $v_i = (u_{i,1}, \ldots, u_{i,n-t}, \beta_{i,1}, \ldots, \beta_{i,t})$, then we have the following property

$$v_i = (\beta_{i,1}, \ldots, \beta_{i,t}) \left(\begin{array}{c} M_1 \\ M_2 \\ \cdots \\ M_t \end{array}\right)$$

Thus if the receiver node could collect t packets $v_1, \ldots, v_t$, she could recover the original message as

$$\left(\begin{array}{c} M_1 \\ M_2 \\ \cdots \\ M_t \end{array}\right) = \left(\begin{array}{ccc} \beta_{1,1} & \cdots & \beta_{1,t} \\ \cdots & \cdots & \cdots \\ \beta_{t,1} & \cdots & \beta_{t,t} \end{array}\right)^{-1} \left(\begin{array}{c} v_1 \\ v_2 \\ \cdots \\ v_t \end{array}\right)$$

3 Information Theoretic Approach to Network Coding

Cai and Yeung [5, 7, 8] have proposed a general framework and obtained theoretical bounds for network error correction. Based on these theoretical bounds, Cai and Yeung [9] designed algorithms for achieving network coding based secure communication against passive adversaries (wire tappers). Jaggi, Langberg, Katti, Ho, Katabi, and Medard [12] studied network coding based secure communication techniques against Byzantine adversaries. It should be noted that in these two papers, the adversary model is based on the number of communication links that are controlled by the adversary. This is very different from the more powerful model based on the number of nodes that are controlled by the adversary. Thus the application could be limited. For example, Dong, Curtmola, and Nita-Rotaru [10] pointed out that this model is not realistic in sensor networks.

For example, in the Figure 2 from Cai and Yeung [9], the sender s generates a random key k_1 and sends the encrypted versions of the message $m_1 + k_1$ and $m_1 - k_1$ on the two outgoing links respectively. It is clear that any single link will not be able to recover the message m_1 nor the key k_1. Thus this message transmission protocol is secure against any single corrupted link. However, the node a_0 could easily recover the message by summing up the two received packets:

$(m_1 - k_1) + (m_1 + k_1) = 2m_1$. In another word, the protocol proposed by Cai and Yeung [9] is not secure against one single corrupted node.

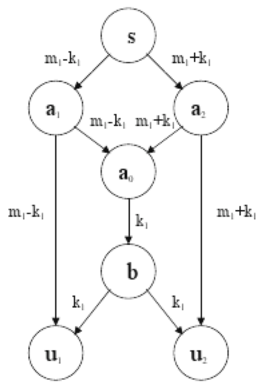

Figure 2: Cai and Yeung's private network coding

The same model is used by Jaggi, Langberg, Katti, Ho, Katabi, and Medard [12] to design secure message transmission protocols in network coding against active (Byzantine style) adversaries. Since the adversary model is based on the maximum number of links controlled by the adversary, these message transmission protocols are not secure against an adversary who controls only one node and is able to generate t-outgoing corrupted messages.

Though the link based adversary models used in [9] and [12] may be sufficient in some applications where it is hard for the adversary to control one single node, these models are not valid in many applications where the adversary could control several nodes. Thus it is preferred to study information theoretic message transmission network coding protocols in the node based adversary models.

4 Probabilistic approach to secure network coding

In addition to the information theoretic approaches for secure network coding that we have discussed in the previous section. Several efforts have been made to design cryptographic protocols for secure network coding. In this section, we describe these efforts and discuss their security.

4.1 Signature Scheme from Infocom 08

Yu, Wei, Ramkumar, and Guan [18] designed a homomorphic digital signature scheme for network coding against pollution attacks. The homomorphic signature scheme was designed with the purpose that:

- Each intermediate node can verify whether a received packet has a valid signature.
- Without access to the private key, each intermediate node can generate a random linear combination of the incoming messages together with a digital signature on the combined message. In another word, after the source node generates a digital signature on the messages, other nodes may be able to generate a digital signature on the linear space expanded by the original message vectors.

Specifically, their signature scheme is as follows. The source node has an RSA private key d, and public key (N, e). Without loss of generality, we assume that these parameters are chosen securely and meets the security requirements (e.g., N is at 1024 bits or 2048 bits). Furthermore, let p and q be two primes such that $q|(p-1)$ and $g_1, \cdots, g_n \in Z_p$ be randomly chosen numbers with order q (mod p). We assume that all nodes in the network have an authentic copy of the system parameters $(N, e), p, q, g_1, \cdots, g_n$.

In one session, the source node generates the digital signatures for messages $M_1, \ldots, M_t$ as follows. The signature on $M_i = (m_{i,1}, \cdots, m_{i,n})$ is computed as:

$$S(M_i) = \left(\prod_{j=1}^{n} g_j^{m_{i,j}}\right)^d \bmod N$$

Now assume that an intermediate node receives message-signature pair $(v, S(v))$, where $v = (u_1, \ldots, u_n)$, it can verify the digital signature as follows:

$$S(v)^e = \prod_{j=1}^{n} g_j^{u_j} \bmod p \qquad \bmod N$$

Furthermore, assume that the intermediate node receives $(v_1, S(v_1)), \cdots, (v_k, S(v_k))$ from its k incoming edges respectively, where $S(v_k)$ is the digital signature on v_k. For each outgoing edge, the node chooses random $\alpha_1, \ldots, \alpha_k \in F_p$ and computes the digital signature on the combined message $v = \alpha_1 v_1 + \cdots + \alpha_k v_k$ as follows:

$$S(v) = \prod_{j=1}^{n} S(v_j)^{\alpha_j}$$

The correctness of the signature scheme is straightforward and omitted here. The authors in [18] have provide a security proof for the above signature scheme. In the following, we show a few attacks on this signature scheme.

4.1.1 Attack 1

This attack is derived from the "batch verification" properties provided by the authors in their original paper [18]. Assume that the adversary observes a message-signature pair $(M', S(M'))$ from session one and a message-signature pair $(M'', S(M''))$ from session two. For any random numbers β_1, β_2, the adversary can generate the "digital signature" $S(M)$ on the message $M = \beta_1 M' + \beta_2 M''$ as $S(M')^{\beta_1} S(M'')^{\beta_2}$. It should be noted that this message M belongs neither to session one nor to session two. One may propose that a session ID could be embedded into the message space, but there is no easy way to do that. One may also propose that the system parameters $g_1, \cdots, g_n$ be changed for each session. That is, different sessions do not share the same parameters. But then the protocol will become very inefficient and one may wonder what is the advantage of network coding then (compared to traditional copy and forward techniques) since the network capacity may be significantly reduced. Our next attack shows that even if one adds some the session ID to the signature, it may still be easily broken.

4.1.2 Attack 2

Assume that the adversary observes a digital signature $S(M)$ on the message $M = (m_1, m_2, \cdots, m_n)$. For any message $M' = (m, m_2, \cdots, m_n)$ chosen by the adversary, she may compute a number x such that $m = m_1 + e \cdot x$. Then the signature on the message M' is $S(M') = S(M) \cdot g_1^x$. The reason is due to the following fact:

$$S(M')^e = S(M)^e g_1^{e \cdot x} = g_1^{e \cdot x} \prod_{j=1}^{n} g_j^{m_j} = g_1^{m_1 + ex} \cdots g_n^{m_n}$$

Similarly, the adversary can generate a digital signature $S(M'')$ for any message $M'' = (m'_1, \cdots, m'_n)$ at her choice. This attack shows that even if the source node distributes different system parameters for different session, it still does not work!

4.2 Digital signature on orthogonal vectors from ISIT 07

Zhao, Kalker, Medard, and Han [19] introduce a different scheme to authenticate messages in network coding. Roughly speaking, their technique is based on the following ideas:

- In order for the source node to authenticate messages $M_1, \cdots, M_t$, the source node finds a vector u which is orthogonal to all these messages (that is, $M_i \cdot u = 0$ for all $1 \le i \le t$) and digitally sign u. The intermediate and receiver nodes will accept a received message M if and only if $M \cdot u = 0$.

The intuition for this scheme is that a received message M should be accepted if only if it belongs to the linear space spanned by the vectors $M_1, \cdots, M_t$. The authors [19] think that this is "equivalent" to the fact that $M \cdot u = 0$. Unfortunately, this argument is not valid. There are many vectors M' with the property $M' \cdot u = 0$ but M' does not belong to the linear space spanned by the vectors $M_1, \cdots, M_t$. In the following, we describe the signature scheme and simple attacks on the scheme.

The parameters for the system consists of a generator g for the group G of order p. The private key for the source node is n random elements $\{\alpha_1, \cdots, \alpha_n\}$ from F_p. The public key for the source node is $\{g^{\alpha_1}, \cdots, g^{\alpha_n}\}$. We assume that all nodes in the network have an authentic copy of the system parameters (g, G, p) and the public key of the source node.

For the source node to sign the messages $M_i = (m_{i,1}, \cdots, m_{i,n})$ $(1 \le i \le t)$, the source node finds a nonzero vector $\mathbf{u} \in F_p^n$ with the property that

$$\mathbf{u} \cdot M_i = 0 \quad i = 1, \cdots, t$$

Then the digital signature is $\mathbf{x} = (u_1 \alpha_1^{-1}, \cdots, u_n \alpha_n^{-1})$ together with a standard signature on $\mathbf{x}$

To verify whether **x** is a valid signature on a message $M = (m_1, \cdots, m_n)$, one needs to check whether

$$\prod_{i=1}^{n} (g^{\alpha_i})^{x_i m_i} = 1$$

The correctness of the signature scheme is straightforward and is omitted here. The authors in [19] have provided a security proof for the above signature scheme. In the following, we show a few attacks on this signature scheme.

4.2.1 Attack 1

This attack was noticed by the authors in their original paper [19]. Assume that $\mathbf{x}$ is the digital signature for session one (of file F) and $\mathbf{x}'$ is the digital signature for session two (of file F'). Furthermore, assume that the message $M = (m_1, \dots, m_n)$ is from session one. Then one can construct a message $M' = (m'_1, \cdots, m'_n)$ for session two, where $m'_i = x_i m_i / x'_i$. This is true since

$$\prod_{i=1}^{n} (g^{\alpha_i})^{x'_i m'_i} = \prod_{i=1}^{n} (g^{\alpha_i})^{x_i m_i} = 1.$$

Obviously, M' is not a valid message for session two. This attack shows that each session needs the secure deployment of a different public/private keys, which could use too much of the network coding capacity.

4.2.2 Attack 2

It is straightforward that if the adversary can collect t messages, then she will be able to recover the original message. At the same time, the adversary will also be able to compute the original orthogonal vector **u**. Here we assume that the implementation for computing **u** from the messages is public so **u** can be uniquely recovered. By the fact that $\mathbf{x} = (u_1\alpha_1^{-1}, \cdots, u_n\alpha_n^{-1})$, the adversary will be able to recover the private key of the source node. Thus she will be able to create any signature on any chosen message.

4.2.3 Attack 3

This attack shows that the digital scheme based on orthogonal vectors are completely infeasible for practical purposes.

Let $M_1, \cdots, M_t$ be the message vectors where $M_i = (m_{i,1}, \cdots, m_{i,n-t}, 0, \cdots, 0, 1, 0, \dots, 0)$. It is straightforward to check that the following vector **u** is orthogonal to all of these messages and satisfies the requirements for the orthogonal signature.

$$(1_1, \cdots, 1_{n-t}, -\sum_{j=1}^{n-t} m_{1,j}, -\sum_{j=1}^{n-t} m_{2,j}, \dots - \sum_{j=1}^{n-t} m_{t,j})$$

Now let $M'_i = (m'_{i,1}, \cdots, m'_{i,n-t}, 0, \cdots, 0, 1, 0, \dots, 0)$ where $m'_{i,1}, \cdots, m'_{i,n-t}$ is any permutation of $m_{i,1}, \cdots, m_{i,n-t}$. It is clear that M'_i is orthogonal to **u**. Thus it will be accepted as a valid message. However, M'_i is not a linear combination of the original messages.

The reason why this attack is successful is as follows:

- The linear space spanned by the original messages $M_1, \cdots, M_t$ is t-dimensional.
- Assume that **u** be any fixed vector which is orthogonal to the message space. Then **u** is orthogonal to a subspace of dimension $n - 1$ which contains the message space. Thus, for $n - 1 > t$, there is a huge room for the adversary to generate fake messages.

4.3 Charles, Jain, and Lauter's signature scheme

Charles, Jain, and Lauter [6] have designed a signature scheme for network coding. The scheme is based on bilinear maps which we will discuss first.

4.3.1 Bilinear maps and the bilinear Diffie-Hellman assumptions

In the following, we briefly describe the bilinear maps and bilinear map groups. The details could be found in Joux [14] and Boneh and Franklin [3].

1. G_1, G_2, and G_T are three (multiplicative) cyclic groups of prime order q.
2. g_1, g_2 are generators of G_1, G_2 respectively.
3. $\hat{e} : G_1 \times G_2 \to G_T$ is a bilinear map.

A bilinear map is a map $\hat{e} : G \times G \to G_1$ with the following properties:

1. bilinear: for all $x, y \in Z$, we have $\hat{e}(g_1^x, g_2^y) = \hat{e}(g_1, g_2)^{xy}$.
2. non-degenerate: $\hat{e}(g_1, g)_2 \neq 1$.

We say that G_1, G_2 are bilinear groups if the group action in G_1, G_2 can be computed efficiently and there exists a group G_T and an efficiently computable bilinear map $\hat{e} : G_1 \times G_2 \to G_T$ as above. Concrete examples of bilinear groups are given in [14, 3]. For convenience, throughout the paper, we view G_1, G_2, and G_T as multiplicative groups though the concrete implementation of G_1, G_2 could be additive elliptic curve groups.

4.3.2 The signature scheme

We first briefly discuss the network coding signature scheme by Charles, Jain, and Lauter [6].

The system parameter consists of the bilinear group $\mathcal{G} = \langle G, G, G_T, \hat{e} \rangle$ and $(n+1)$ elements $g_1, \cdots, g_n, g \in G$ that are chosen by the source node. Note that here we assume that $G = G_1 = G_2$ for the bilinear groups.

For each session, the source node chooses secret key $(s_1, \cdots, s_n)$. Then the signature on the message $M_i = (m_{i,1}, \cdots, m_{i,n})$ is $(g^{s_1}, \cdots, g^{s_n}, S(M_i))$ where

$$S(M_i) = \prod_{j=1}^{n} g_j^{m_{i,j} s_j}$$

Now assume that an intermediate node receives message-signature pair $(v, (h_1, \cdots, h_n, S(v)))$, where $v = (u_1, \ldots, u_n)$, it can verify the digital signature as follows:

$$\prod_{j=1}^{n} \hat{e}(g_j^{u_j}, h_j) = \hat{e}(S(v), g)$$

Furthermore, assume that the intermediate node receives $(v_1, (h_1, \cdots, h_n, S(v_1))), \cdots$, and $(v_k, (h_1, \cdots, h_n, S(v_k)))$ from its k incoming edges respectively, where $S(v_i)$ is the digital signature on $v_i = (u_{i,1}, \cdots, u_{i,n})$. For each outgoing edge, the node chooses random $\alpha_1, \ldots, \alpha_k \in F_p$ and computes the digital signature on the combined message $v = \alpha_1 v_1 + \cdots + \alpha_k v_k$ as $(h_1, \cdots, h_n, S(v))$ where

$$S(v) = \prod_{j=1}^{n} g_j^{s_j \sum_{i=1}^{t} \alpha_i u_{i,j}} = \prod_{i=1}^{t} \prod_{j=1}^{n} g_j^{\alpha_i s_j u_{i,j}} = \prod_{i=1}^{t} S(v_i)^{\alpha_i}$$

The correctness of the signature scheme is straightforward and omitted here. The authors in [6] have provide a security proof for the above signature scheme. In the following, we show a few attacks on this signature scheme.

4.3.3 Attacks

Our first analysis shows that the values $(g^{s_1}, \cdots, g^{s_n})$ have to be distributed to all nodes in a secure channel for each session. The reason is as follows.

- Assume that the adversary observes one signature $(g^{s_1}, \cdots, g^{s_n}, S(M_i))$ on the message $M_i = (m_{i,1}, m_{i,2}, \cdots, m_{i,n})$. For any message $M' = (m, m_{i,2}, \cdots, m_{i,n})$ chosen by the adversary, the adversary can compute a number β such that $m = m_{i,1}\beta^{-1}$. Then $\left(g^{\beta s_1}, \cdots, g^{s_n}, S(M_i)\right)$ is a signature for M'. It is straightforward for the adversary to generate signatures on any message M'' by modifying the values of $(g^{s_1}, \cdots, g^{s_n})$.

Boneh, Freeman, Katz and Waters [4] observed that, for this signature scheme, if both sessions share $(g^{s_1}, \cdots, g^{s_n})$, then the adversary can easily combine the signatures on messages from two sessions to generate a new signature on a fake message:

- Assume that M' is from session one and M'' is from session two. The signatures on the two messages are $h_1, \cdots, h_n, S(M')$ and $h_1, \cdots, h_n, S(M'')$.
- For any β_1 and β_2, one can compute the signature on the message $\beta_1 M' + \beta_2 M''$ as $S(M')^{\beta_1} S(M'')^{\beta_2}$.

Combining these attacks, it is clear that the network coding signature in [6] requires a secure channel for each session. This may be achieved by letting the source node digital sign the value $(g^{s_1}, \cdots, g^{s_n})$ using a traditional digital signature scheme. But this will certainly introduce extra overhead for the network traffic which may significantly reduce the performance of network coding.

4.4 Boneh, Freeman, Katz and Waters signature schemes

In previous sections, we show that all these signature schemes are either non-secure or non-practical. In this section, we briefly describe a recent effort by Boneh, Freeman, Katz and Waters [4]. They designed two provable secure digital signature schemes NCS1 and NCS2 for network coding. However, the first scheme NCS1 is based on bilinear maps, which may require more powerful computing capabilities for the intermediate nodes (could be routers). Thus it may be impractical for some applications (see, e.g., [10] for some discussions) though may be acceptable for systems with relatively powerful routers.

The signature scheme NCS2 requires longer signatures to be delivered for each session, which could reduce the advantage of the network coding if the session message is relatively smaller, which is often true in wireless communications such as sensor networks.

References

[1] http://www.ifp.illinois.edu/~koetter/NWC/

[2] R. Ahlswede, N. Cai, S.-Y. R. Li and R. W. Yeung. Network information flow. *IEEE Trans. on Information Theory* **46**:1204–1216, 2000.

[3] D. Boneh and M. Franklin. Identity-based encryption from the Weil pairing. *SIAM J. Computing* **32**(3):586–615, 2003.

[4] D. Boneh, D. Freeman, J. Katz, and B. Waters. Signing a Linear Subspace: Signature Schemes for Network Coding. In *Public Key Cryptography, PKC 2009*, pages 68–87, LNCS 5443, Springer Verlag, 2009.

[5] N. Cai and R. Yeung. Network coding and error correction. In: *Proc. 2002 IEEE Information Theory Workshop 2002*, pages 119–122.

[6] D. Charles, K. Jain, and K. Lauter. Signatures for network coding. *Int. J. Information and Coding Theory*, **1**(1):3–14. An abstract of this paper has appeared in: *40th Annual Conf. on Information Sciences and Systems 2006*. Microsoft has also a patent on this technique, see http://www.wipo.int/pctdb/en/wo.jsp?wo=2007056038

[7] R. Yeung and N. Cai. Network Error Correction, I: Basic Concepts and Upper Bounds. *Commun. Inf. Syst.* **6**(1):19–35, 2006

[8] N. Cai and R. Yeung. Network Error Correction, II: Lower Bounds. *Commun. Inf. Syst.* **6**(1):37–54, 2006

[9] N. Cai and R. Yeung. Secure network coding. In: *Proc. International Symposium on Information Theory (ISIT '02)*, June 2002. http://iest2.ie.cuhk.edu.hk/~whyeung/publications/secure.pdf

[10] J. Dong, R. Curtmola, and C. Nita-Rotaru. Practical defenses against pollution attacks in intra-flow network coding for wireless mesh networks. In: *Proc. ACM WiSec 2009*, ACM Press.

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

[12] S. Jaggi, M. Langberg, S. Katti, T. Ho, D. Katabi, M. Medard Resilient network coding in the presence of Byzantine adversaries. In *Proc. 26th Annual IEEE Conf. on Computer Commun., INFOCOM*, pages 616–624, 2007.

[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 Transaction on Information Theory*, 2003.

[14] A. Joux. A one round protocol for tripartite Diffie-Hellman. In: *Algorithmic number theory symposium, ANTS-IV*, LNCS 1838, pages 385–394, 2000.

[15] R. Koetter and M. Medard. An algebraic approach to network coding. *IEEE/ACM Transactions on Networking*, 2003.

[16] S. -Y. R. Li, R. W. Yeung, and N. Cai. Linear network coding. *IEEE Transaction on Information Theory*, 2003.

[17] Y. Wang and Y. Desmedt. Perfectly Secure Message Transmission Revisited. *IEEE Transactions on Information Theory* **54**:(6):2582–2595, 2008.

[18] Z. Yu, T. Wei, B. Ramkumar, and Y. Guan. An Efficient Signature-based Scheme for Securing Network Coding against Pollution Attacks. In *Proc. 27th Annual IEEE Conf. on Computer Commun., INFOCOM*, pages xxx-xxx, 2008.

[19] F. Zhao, T. Kalker, M. Medard, K. Han. Signatures for Content Distribution with Network Coding. In *Proc. 2007 IEEE International Symposium on Information Theory*, pages xxx, 2007.