# Insecure "Provably Secure Network Coding" and Homomorphic Authentication Schemes for Network Coding

Yongge Wang
UNC Charlotte, USA
yonwang@uncc.edu

January 30, 2010

**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). Furthermore, we will propose a secure homomorphic authentication scheme for network coding.

## 1 Introduction

Maximum flow minimum cut (MFMC) theory [23] has been one of the most important principles for network traffic routing. However, MFMC theorem works only for the case that 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 [3] has been designed to overcome these problems and it has been shown that network coding can maximize network throughput [3, 33, 39], while traditional copy-and-forward networking technology cannot. In particular, it has been shown [30, 33, 39, 42] that random linear code can be used to broadcast a message to multiple recipients with maximum network capacity and probabilistic reliability. Deterministic polynomial time network coding schemes have also been designed to achieve maximum network capacity [33, 39, 42]. Since these seminal works, network coding techniques have been extensively used in other applications such as wireless networks [21, 36, 37, 43], energy [51], content distribution [27], and distributed storage [34].

Though network coding techniques have been extensively studied and mature techniques are now available for practical network coding, secure network coding techniques have relatively been less addressed. Without efficient techniques for reliable and private network coding, it is infeasible to widely deploy network coding techniques.

A malicious node in a network coding environment may inject/forward corrupted packets into the information flow. Since network coding makes the intermediate node mix received packets, a

single corrupted packet can corrupt the entire information reaching the destination. This kind of attack is commonly known as the *pollution attack*. Several researchers have tried to address this problem in a series of papers with important contributions. However, our analysis below shows that several of them are not suitable and several others could be easily broken.

Cai and Yeung [12, 52, 14] have proposed a general framework and obtained theoretical bounds for network error correction. Based on these theoretical bounds, Cai and Yeung [15] have designed algorithms for achieving network coding based information theoretic secure communication against passive adversaries (wire tappers). Several other papers [14, 8, 29, 31, 32, 52] studied network coding based information theoretic secure communication techniques against Byzantine/active adversaries. It should be noted that in these papers, the adversary model is based on the threshold 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 some wire based networks, it is unrealistic in wireless networks [21] or overlay networks such as peer-to-peer networks where the participants are open to the public. Thus the scope of these results could be limited.

It should be noted that non-network-coding based perfectly secure (information theoretic) message transmission techniques have been extensively studied in a series of papers (see, e.g., [19, 20, 24, 25, 48]). For example, Wang and Desmedt [48, 49] have designed information theoretic secure message transmission techniques against Byzantine adversaries in the non network coding based environment.

Cryptographic (or probabilistic) techniques have also been designed by researchers to protect network coding security against pollution attacks. It should be noted that traditional digital signature approaches are not suitable for network coding process since each intermediate node needs to mix the incoming packets and then forward it to the next node. This process destroys the sender's original signature. In order to address this challenge, several homomorphic cryptographic schemes have been proposed for network coding. The examples are: homomorphic hashing [28, 26, 40], homomorphic digital signatures [53, 55, 26, 16, 10], homomorphic MAC [2, 5, 47, 41, 44], and other schemes [38, 21].

In this paper, we will show that several of the existing cryptographic protocols for secure network coding could be easily broken and several others are impractical for network coding. This paper will also propose a secure and efficient homomorphic authentication scheme for network coding.

## 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 [3]. 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.

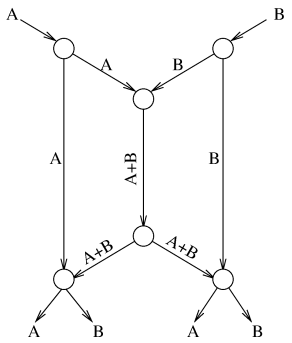


Figure 1: Network coding

Network coding has been extensively studied by researchers in the past few years. The works in [30, 33, 39, 42] 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 transmit the values $M_1, \ldots, M_t$ to the source node. So the source node transmits random linear combinations of these messages on 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 $v_1, \ldots, v_t$, then with high probability she could

recover the original message as

$$\begin{pmatrix} M_1 \\ M_2 \\ \cdots \\ M_t \end{pmatrix} = \begin{pmatrix} \beta_{1,1} & \cdots & \beta_{1,t} \\ \cdots & \cdots & \cdots \\ \beta_{t,1} & \cdots & \beta_{t,t} \end{pmatrix}^{-1} \begin{pmatrix} v_1 \\ v_2 \\ \cdots \\ v_t \end{pmatrix}$$

# 3 Information Theoretic Approach to Network Coding

Cai and Yeung [12, 52, 14] have proposed a general framework and obtained theoretical bounds for network error correction. Based on these theoretical bounds, Cai and Yeung [15] designed algorithms for achieving network coding based secure communication against passive adversaries (wire tappers).

Several other papers [14, 8, 29, 31, 32, 52] studied network coding based information theoretic secure communication techniques against Byzantine/active adversaries. It should be noted that in these papers, the adversary model is based on the threshold 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 some wire based networks, it is unrealistic in wireless networks [21] or overlay networks such as peer-to-peer networks where the participants are open to the public.

For example, in Figure 2 (from Cai and Yeung [15]), 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 [15] is not private against one single eavesdropping node.

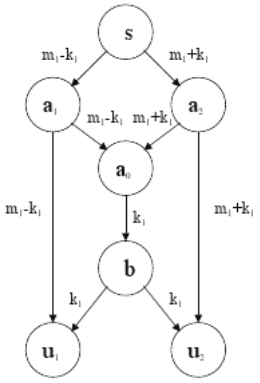


Figure 2: Cai and Yeung's private network coding

The same model is used by other researchers (see, e.g., [14, 8, 29, 31, 32, 52]) 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 network coding message transmission protocols are NOT secure against an adversary who*

*controls one single node and generates $k$-outgoing corrupted messages* where $k$ is the threshold bounds in these protocols. Thus the scope of these network coding transmission protocols could be limited. Note that the authors in [50] propose to reduce the adversary's capability by only allowing nodes to broadcast at most once. This model requires trusted nodes and are impractical for open systems such as practical wireless networks [22] or peer-to-peer networks.

Though the link based adversary models may be sufficient in some applications where it is hard for the adversary to control one single node with several output channels, these models are not valid in many applications where the adversary could control several nodes. For example, in peer-to-peer networks (indeed, one of the major applications of network coding is peer-to-peer networks) or wireless networks [21]. 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 show that these efforts are far from suitable solutions.

## 4.1 Signature Scheme from Infocom 08

Yu, Wei, Ramkumar, and Guan [53] 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 and, 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.

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 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:

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

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 [53] provide a security proof for the above signature scheme. In the following, we describe several 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 [53]. 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 $\beta_1, \beta_2$, the adversary can generate the "digital signature" $S(M)$ on the coined 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 for these applications (compared to traditional copy-and-forward techniques) since the useful network capacity may be significantly reduced. Our next attack shows that even if one adds some kind of session identification to the signatures, 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 sessions, it still does not work!

Recently, the same group of researchers have proposed an efficient scheme [54] for XOR based network coding. The scheme relies solely on symmetric key encryption schemes by avoiding using expensive public key cryptographic primitives, and may not be extended to general random linear network coding.

### 4.2 Digital signature on orthogonal vectors from ISIT 07

Zhao, Kalker, Medard, and Han [55] 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 $\mathbf{u}$ which is orthogonal to all these messages (that is, $M_i \cdot \mathbf{u} = 0$ for all $1 \leq i \leq t$) and digitally sign $\mathbf{u}$. The intermediate and receiver nodes will accept a received message $M$ if and only if $M \cdot \mathbf{u} = 0$.

The intuition for this scheme is that a received message $M$ should be accepted if and only if it belongs to the linear space spanned by the vectors $M_1, \cdots, M_t$. The authors [55] think that this is "equivalent" to the fact that $M \cdot \mathbf{u} = 0$. Unfortunately, this argument is not valid. There are many vectors $M'$ with the property $M' \cdot \mathbf{u} = 0$ 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 consist 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 $M_i = (m_{i,1}, \cdots, m_{i,n})$ $(1 \leq i \leq t)$, 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 for the message space is $\mathbf{x} = (u_1\alpha_1^{-1}, \cdots, u_n\alpha_n^{-1})$ together with a standard digital signature on $\mathbf{x}$.

To verify whether $\mathbf{x}$ 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 [55] 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 described in [55]

This attack was noticed by the authors in their original paper [55]. 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 $M = (m_1, \ldots, m_n)$ 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 coined 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 $M_i = (m_{i,1}, \cdots, m_{i,n-t}, 0, \cdots, 0, 1, 0, \ldots, 0)$. 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.

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

Now let $M_i' = (m_{i,1}', \cdots, m_{i,n-t}', 0, \cdots, 0, 1, 0, \ldots, 0)$ 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** is 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.

Recently, Kehdi and Li [38] designed a different scheme based on the orthogonal space. In their scheme, the vectors of the orthogonal spaces (that is, orthogonal to the linear space spanned by the message vectors) are distributed to all intermediate nodes (using a scheme that is similar to the network coding). There are several challenges for this scheme to be practical. First, the adversary may try to attack the orthogonal space distribution phase. The authors in [38] propose to use digital signature schemes to protect this phase. But then we have the egg and chicken problem.

### 4.3 Charles, Jain, and Lauter's signature scheme

Charles, Jain, and Lauter [16] 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 [35] and Boneh and Franklin [9].

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 [35, 9]. 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 [16].

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 a secret key $(s_1, \cdots, s_n)$. 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 a 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 [16] 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 (by assuming the values are not securely distributed, we present an attack).

- 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 $\beta$ such that $m = m_{i,1}\beta^{-1}$. Then $\big(g^{\beta s_1}, \cdots, g^{s_n}, S(M_i)\big)$ is a signature for $M'$. Thus 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 [10] 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 $\beta_1$ and $\beta_2$, 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 [16] requires a secure channel for each session. This may be achieved by letting the source node digitally sign the value $(g^{s_1}, \cdots, g^{s_n})$ using a traditional digital signature scheme.

In a summary, the signature scheme from [16] is not suitable for network coding for the following reasons:

1. *High computational overhead*: bilinear operations are very inefficient
2. *Bandwidth non-efficiency*: for the distribution of each file (session), the source node needs to securely broadcast $(g^{s_1}, \cdots, g^{s_n})$ to all nodes.

## 4.4 Other digital signature schemes, homomorphic hashing schemes, and network coding message authentication codes (MAC)

In previous sections, we show that all these signature schemes are either non-secure or non-practical. Recently, Boneh, Freeman, Katz, and Waters [10] designed two provably 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 most applications (see, e.g., [21] for some discussions).

The second digital signature scheme NCS2 from [10] requires longer signatures to be delivered for each session, which will reduce the advantage of the network coding by using much of the bandwidth for signature delivery. Furthermore, the scheme NCS2 requires each intermediate node to compute $n$ expensive public key exponentiation operations which could be impractical for many applications.

Gennaro, Katz, Krawczyk, and Rabin [26] proposed a RSA based homomorphic signature scheme and a homomorphic hashing scheme for linear network coding over integers. Though

these two schemes are good in several aspects, they still need several exponentiation operations for intermediate node which is too expensive for many applications.

Based on a classic MAC system due to Carter and Wagman [13], homomorphic MAC schemes have been introduced for different purposes [2, 5, 47, 41]. In Agrawal and Boneh's homomorphic MAC scheme [2], the source node and the receiver node share the secret $(k_1, k_2)$. In order for the source node to generate an MAC tag on a message $M_i \in F_p^n$ in session id, it first uses pseudo-random functions to generate $u = (u_1, \ldots, u_n) \in F_p^n$ from $k_1$ and generate $b \in F_p$ from (id, $i$, $k_2$). The MAC tag on $M_i$ is then defined as $u \cdot M_i + b$. It is clear that the intermediate node could generate the MAC tag for a linear combination of MACed messages based on their MAC tags. However, a node could verify the MAC tag only if it knows the value of the key $(k_1, k_2)$. Thus the scheme in [2] will only help the receiver node (but not the intermediate nodes) to detect malicious packets. In another word, the scheme will not be able to defeat pollution attacks.

The authors in [2] further extended their scheme to broadcast homomorphic MACs by pre-distributing some keys to the intermediate nodes (based on the cover free family concept) so that intermediate nodes could verify the MAC tags (thus avoiding pollution attacks). However, this scheme is only $c$-collusion resistant for some pre-determined $c$. Furthermore, when $c$ becomes larger, the scheme will become impractical (e.g., for the key pre-distribution).

Based on the multi-receiver/multi-sender authentication scheme [18], Oggier and Fathi [44] recently designed a message authentication scheme for network coding. However, due to the complicated pre-key distribution scheme, the scheme in [44] has limited applications.

Gkantsidis and Rodriguez [28] introduces the homomorphic hashing for network coding which requires expensive exponentiation computations for each intermediate nodes. In particular, the hashing output for a message $M_i = (m_{i,1}, \ldots, m_{i,n})$ is defined as $h(M_i) = \prod_{k=1}^{n} g_k^{m_{i,k}}$. Thus it is not practical for many applications [22].

Dong, Curtmola, and Nita-Rotaru [21] recently proposed an elegant TESLA-like scheme to defend against pollution attacks in intra-flow network coding for wireless mesh networks. In their scheme, the source node periodically computes and disseminates a digitally signed random checksum packet $(\text{CHK}_s, s, l)$ where $\text{CHK}_s$ is a $t \times b$ matrix and $b$ is the security parameter. This protocol works fine for wireless mesh networks with sufficient generations of packets. However, for general networks (such as peer-to-peer networks), there are several limitations for this protocol. In particular, the frequent broadcast of the large size checksum $\text{CHK}_s$ will use up much of the network bandwidth and each intermediate node needs to verify the public key digital signature on each checksum.

# 5 Secure message authentication code for network coding

In this section, we propose an secure message authentication code for network coding based on delayed key release (TESLA-like) schemes.

Typical authenticated broadcast channels require asymmetric cryptographic techniques, otherwise any compromised receiver could forge messages from the sender. Cheung [17] proposed a symmetric cryptography based source authentication technique in the context of authenticating communication among routers. Cheung's technique is based on delayed disclosure of keys by the sender. It was used in the Guy Fawkes protocol [4] for interactive unicast communication, and in [6, 7, 11, 45, 46] for streamed data multicast. Later, Perrig, Szewczyk, Tygar, Wen, and Culler adapted delayed key disclosure based TESLA protocols [45, 46] to sensor networks for sensor

broadcast authentication (the new adapted protocol is called $\mu$TESLA).

The delayed checksum release idea has also been proposed for network coding in wireless mesh networks [21]. As we have mentioned in previous sections, this scheme is suitable to wireless mesh networks and requires frequent broadcast of large amount of checksum packets, and it has disadvantages in other environments such as peer-to-peer networks.

Assume that the maximum network hops for each packet to arrive its destination is bounded by $T$. For each session with identification number id, the source node chooses a random seed $s$ and computes $b^u = H_1^u(s, \text{id}) \in F_p$ and $a_j^u = H_2(j, b^u) \in F_p$ for $j = 0, \cdots, n$ and $u = 1, \cdots, T$, where $H_1$ and $H_2$ are a pseudo-random functions and $H_1^u(\cdot)$ means to apply $u$ times of the function $H_1$ to the input. That is, $H_1^u(\cdot) = H_1(H_1(\cdots H_1(\cdot)))$. The source node can generate the MAC tag for the message $M_i = (m_{i,1}, m_{i,2}, \cdots, m_{i,n})$ as follows:

$$\text{MAC}_u(M_i) = (a_1^u, \ldots, a_n^u) \cdot M_i + a_0^u(m_{i,n-t+1} + \cdots + m_{i,n}).$$

It is clear that this is a homomorphic MAC scheme and an intermediate node can generate the MAC tag $\text{MAC}_u(M)$ for a linearly combined message $M$ of the messages $M_1, \cdots, M_t$ from the MAC tags $\text{MAC}_u(M_1), \cdots, \text{MAC}_u(M_t)$.

At the beginning of the session, the source node digitally signs and broadcasts $b^T$ to all nodes in the network, using a traditional cryptographic digital signature scheme. At time $u$, the source node broadcasts $b^{T-u}$ (no signature needed).

The source node initiates the message transmission by transferring a message $M$ (note that according to our discussion in previous sections, we assume that $M$ is a linear combination of the messages $M_1, \ldots, M_t$) together with its MAC tags $\text{MAC}_T(M)$, $\cdots$, $\text{MAC}_1(M)$. When an intermediate node at hop $u$ receives packets with MAC tags $\text{MAC}_{T-u}$, $\cdots$, $\text{MAC}_1$, it will hold the packets and wait for the value $b^{T-u}$ from the source node. After it receives this value, it can verify the validity of $b^{T-u}$ by comparing whether $H_1^u(b^{T-u}) = b^T$. If the equation holds, it continues to verify whether the MAC tag $\text{MAC}_{T-u}$ is valid by generating $(a_0^{T-u}, \cdots, a_n^{T-u})$ from $b^{T-u}$. If the MAC tag is not valid, it will discard the packet. Otherwise, it will continue with the network coding protocol by generating and attaching the MAC tags $\text{MAC}_{T-u-1}$, $\cdots$, $\text{MAC}_1$ for its outgoing messages.

# References


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

[2] S. Agrawal and D. Boneh. Homomorphic MACs: MAC-Based Integrity for Network Coding. In *ACNS 2009*:292-305, Springer Verlag.

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

[4] R. Anderson, F. Bergadano, B. Crispo, J. Lee, C. Manifavas, and R. Needham. A new family of authentication protocols. *Operating Systems Review*, **32**(4):9–20, 1998.

[5] G. Ateniese, R. Burns, R. Curtmola, J. Herring, L. Kissner, Z. Peterson, and D. Song. Provable data possession at untrusted stores. In *Proc. ACM CCS 2007*.

[6] F. Bergadano, D. Cavagnino, and B. Crispo. Chained stream authentication. In: *Selected Areas in Cryptography*, Waterloo, Canada, 2000.

[7] F. Bergadano, D. Cavagnino, and B. Crispo. Individual single source authentication on the mbone. In: *ICME 2000*, August 2000.

[8] K. Bhattad and K.R. Narayanan. Weakly Secure Network Coding. In *NETCOD 2005*, Italy, April 2005.

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

[10] 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.

[11] B. Briscoe. FLAMeS: Fast, Loss-Tolerant Authentication of Multicast streams. Technical report 2000. http://www.labs.bt.com/people/briscorj/papers.html

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

[13] L. Carter and M. Wegman. Universal classes of hash functions. *J. Computer and System Sciences*, 18(2):143–154, 1979.

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

[15] 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

[16] 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

[17] S. Cheung. An efficient message authentication scheme for link state routing. In: *13th Annual Computer Security Applications Conference*, 1997.

[18] Y. Desmedt, Y. Frankel, and M. Yung. Multi-receiver/multi-sender network security: efficient authenticated multicast/feedback. In *IEEE Infocom 1992*.

[19] D. Dolev. The Byzantine generals strike again. *J. of Algorithms*, 3:14–30, 1982.

[20] D. Dolev, C. Dwork, O. Waarts, and M. Yung. Perfectly secure message transmission. *J. of the ACM*, 40(1):17–47, 1993.

[21] 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.

[22] J. Dong, R. Curtmola, and C. Nita-Rotaru. Secure network coding for wireless mesh networks: Threats, challenges, and directions. Computer Communications, 32(17):1790–1801, 2009.

[23] L.R. Ford and D. R. Fulkerson. *Flows in Networks*. Princeton Univ. Press, 1962.

[24] M. Franklin and R. Wright. Secure communication in minimal connectivity models. *J. Cryptology*, 13(1):9–30, 2000.

[25] M. Franklin and M. Yung. Secure hypergraphs: privacy from partial broadcast. In: *Proc. ACM STOC 1995*, pages 36–44.

[26] R. Gennaro, J. Katz, H. Krawczyk, and T. Rabin. Secure network coding over the integers. Cryptology ePrint Archive Report 2009/559. http://eprint.iacr.org/2009/569

[27] G. Gkantsidis and P. Rodriguez. Network coding for large scale content distribution. In *IEEE Infocom 2005*.

[28] G. Gkantsidis and P. Rodriguez. Cooperative security for network coding file distribution. In *IEEE INFOCOM 2006*.

[29] T. Ho, B. Leong, R. Koetter, and M. Medard. Byzantine Modification Detection in Multicast Networks using Randomized Network Coding. In *IEEE Proc. ISIT 2004*.

[30] 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.

[31] S. Jaggi, M. Langberg, T. Ho, and M. Effros. Correction of adversarial errors in networks. In: *IEEE ISIT 2005*, pages 1455-1459.

[32] 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.

[33] 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.

[34] A. Jiang. Network coding for joing storage and transmission with minimum cost. In *ISIT 2006*.

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

[36] S. Katti, D. Katabi, W. Hu, H. Rahul, and M. Medard. The importance of being opportunistic: practical network coding for wireless environments. In *43rd Annual Allerton Conf. Communication, Control and Computing*, 2005.

[37] S. Katti, H. Rahul, D. Katabi, M. Medard, and J. Crowcroft. XORs in the air: practical network coding. In *ACM SIGCOMM*, 2006.

[38] E. Kehdi and B. Li. Null keys: limiting malicious attacks via null space properties of network coding. In *IEEE INFOCOM 2009*.

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

[40] M. Krohn, M. Freedman, and D. Mazieres. On-the-Fly Verification of Rateless Erasure Codes for Efficient Content Distribution. In *IEEE Security and Privacy 2004*.

[41] Q. Li, D. Chiu, and J. C. S. Lui. On the Practical and Security Issues of Batch Content Distribution Via Network Coding. In *IEEE ICNP 2006*.

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

[43] D. Lun, M. Medard, and R. Koetter. Efficient operation of wireless packet networks using network coding. In *IWCT*, 2005.

[44] F. Oggier and H. Fathi: An Authentication Code against Pollution Attacks in Network Coding. CoRR abs/0909.3146, 2009

[45] A. Perrig, R. Canetti, D. Song, and J. Tygar. Efficient and secure source authentication for multicast. In: *NDSS 2001*.

[46] A. Perrig, R. Canetti, J. Tygar, and D. Song. Efficient authentication and signing of multicast streams over lossy channels. In: *IEEE Symp. Security and Privacy*, 2000.

[47] H. Shacham and B. Waters. Compact proofs of retrievability. In *Asiancrypt 2008*, LNCS 5350, pages 90–107, 2008.

[48] Y. Wang and Y. Desmedt. Secure communication in multicast channels: the answer to Franklin and Wrights question. *J. of Cryptology*, 14(2):121–135, 2001. An earlier version of this paper appeared in Eurocrypt 1999.

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

[50] D. Wang, D. Silva, and F. R. Kschischang. Constricting the adversary: A broadcast transformation for network coding. In *Allerton 2007*.

[51] J. Wieselthier, G. Nguyen, and A. Ephremides. On the construction of energy-efficient broadcast and multicast trees in wireless networks. In *IEEE INFOCOM 2000*.

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

[53] 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*, 2008.

[54] Z. Yu, Y. Wei, B. Ramkumar, and Y. Guan. An efficient scheme for securing xor network coding against pollution attacks. In *IEEE INFOCOM 2009*.

[55] 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.