http://ansinet.com/itj

ITJ

ISSN 1812-5638

INFORMATION TECHNOLOGY JOURNAL

ANSInet
Asian Network for Scientific Information
308 Lasani Town, Sargodha Road, Faisalabad - Pakistan

Information Technology Journal 11 (9): 1175-1183, 2012
ISSN 1812-5638 / DOI: 10.3923/itj.2012.1175.1183

UC Secure Network Coding Against Pollution Attacks

Feng Tao, Wang Shuang and Yuan Zhan-Ting
School of Computer Communication of Lanzhou University of Technology,
No. 287, Langongping Road, Qilihe District, Lanzhou, Gansu, People's Republic of China

Abstract: Considering the pollution attacks in network coding, in the single-source multi-sinks directed acyclic network, we present a Universally Composable (UC) secure random network coding scheme based on All-Or-Nothing Transform (AONT) encryption and homomorphic Network Coding Signature (NCS_1). At the source node, by means of AONT encryption an eavesdropper is unable to get any meaningful information no matter how many channels are wiretapped. To prevent malicious modification of data packets, in this study, we adopt the signature scheme NCS_1 for authentication the integrity of the transmitted data and the node identity. Lastly, we have proved the novel scheme is secure in the UC security model. By the security proof and comprehensive analysis, it is showed that the proposed scheme not only has the ability of resistance eavesdropper and the Byzantine attacker but also achieves the maximum flow of information transmission.

Key words: Pollution attacks, AONT encryption, NCS_1, UC security model

INTRODUCTION

Network coding has been proven to maximize network throughput (Ahlswede *et al*., 2000) by mixing the received information before forwarding them to the next nodes, however, network coding is vulnerable to pollution attacks by malicious nodes in the network. Once a packet is corrupted, a single error further will cause pollution of downstream nodes like the plague spread on the network. Thus, even a faulty transmission on a single edge will eventually cause almost all messages being forwarded in the network to be incorrect and will thus prevent reconstruction of even a portion of the messages.

The algorithms of detecting misbehaved wireless nodes have been described by Raja *et al*. (2008) and Jun and Wei-Hua (2010). Some network error correcting algorithms (Cai and Yeung, 2002; Zhang, 2008; Wei *et al*., 2011) based on different techniques have been proposed. Zhou *et al*. (2010) have designed a network coded non-orthogonal Interleave-division Multiple-access (IDMA) cooperation system. The design of secure network coding is mainly against two types of pollution attacks, including wiretapping attacks (passive attacks) (Meng, 2011) and Byzantine attacks (active attacks) (Lamport *et al*., 1982; Wang *et al*., 2003; Yan and Wang, 2004; Meng, 2011). A Byzantine attacker is a malicious adversary hidden in a network, capable of intercepting, tampering or inserting packets in the original information flow. We have surveyed some techniques for combating data pollution when network coding is used. The literatures (Silva and Kschischang, 2008; Chanl and Grant, 2008; Zhou *et al*., 2009) studied different encoding methods for preventing eavesdropping attackers. Ho *et al*. (2004) have proposed a Byzantine attack detection model that based on the attacker did not know the random coding coefficients. The first efficient scheme to design and implement error-correction against Byzantine adversaries has been used in the distributed network coding setting (Jaggi *et al*., 2008). 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. Yu *et al*. (2008) have designed a RSA homomorphic signature scheme for network coding against pollution attacks but it is pointed out in a recent paper (Gennaro *et al*., 2010), that this protocol is incorrect indeed. Agrawal and Boneh (2009) have proposed a homomorphic MAC mechanism but only at the receiving node could detect malicious packets; intermediate nodes could not detect them. RIPPLE mechanism (Li *et al*., 2010) has been designed to prevent pollution attacks. Recently, Wang (2010) shows that there are serious flaws in these schemes (Jaggi *et al*., 2008; Yu *et al*., 2008; Agrawal and Boneh, 2009; Gennaro *et al*., 2010; Li *et al*., 2010).

Our previous study (Feng *et al*., 2011) have presented a security random network coding model against Byzantine attack based on CBC (Hayat *et al*., 2004; Zaidan *et al*., 2010). In the model, the source adopts CBC packet encryption, intermediate nodes use ElGamal signature scheme. However, the security of the scheme is

Corresponding Author: Feng Tao, School of Computer Communication of Lanzhou University of Technology, No. 287, Langongping Road, Qilihe District, Lanzhou, Gansu, People's Republic of China

also dependent on limiting the numbers of eavesdropping links and this signature technology can not be combined at downstream nodes. To solve the problems, this study proposes a UC secure network coding model in view of the single-source multi-sinks directed acyclic network. We adopt AONT encryption instead of previous CBC encryption, no matter how many channels are wiretapped; the attacker can not get any meaningful information of the source messages. In order to ensure the participation of legitimate nodes, the source and intermediate nodes sign encoded message based on homomorphic signature scheme NCS_1 by Boneh *et al*. (2009), intermediate nodes and sinks can verify the integrity and legitimacy of the received information. NCS_1 signature scheme has the advantage that signatures can be associated with individual vectors rather than an entire subspace. Both the public key and per-vector signatures in this scheme have constant size, making the scheme ideally suited for network coding. This scheme also supports the transmission of streaming data, in the sense that the sender need not be aware of the entire file before computing the signature first packet. The novel approach to solve the problem of concurrent network coding against pollution attack within the framework of Universally Composable (UC) security is described by Meng (2011). The most outstanding nature of UC framework is its modular design concept: may alone design a protocol, so long as the protocol satisfies the UC security, it can be guaranteed secure while running concurrently with other protocols. By the security proof and comprehensive analysis, it is showed that the proposed scheme has greatly improved security and efficiency compared to the previous schemes.

PRELIMINARIES

Encoding kernel: The pair $G = (V_G, E_G)$ is called a single-source acyclic network, where V_G and E_G are the node set and the edge set. The node $S \in V_G$ called the source node, $T = \{t_1, t_2, \ldots t_m\}$ called the sink set. For every node v, let In(v) denote the set of incoming channels to v and Out(v) the set of outgoing channels from v. Meanwhile, let S′ denote an imaginary source at the upstream of S.

Local description of a linear network code on an acyclic network: Let F be a finite field and ω a positive integer. An ω-dimensional F-valued linear network code on an acyclic communication network consists of a scalar $k_{d,e}$, called the local encoding kernel, for every adjacent pair (d, e). Meanwhile, the local encoding kernel at the node v means: the $|In(v)|\times|Out(v)|$ matrix $K_v = [k_{d,e}]_{d\in In(v),e\in Out(v)}$.

Global description of a linear network code on an acyclic network: Let F be a finite field and ω a positive integer. An ω-dimensional F-valued linear network code on an acyclic communication network consists of a scalar $k_{d,e}$ for every adjacent pair (d, e) in the network as well as an ω-dimensional column vector f_e for every channel e such that:

- $f_e = \sum_{d\in In(v)} k_{d,e} f_d$, where, $e \in Out(v)$
- The vector f_e for the ω imaginary channels $e \in In(S)$ form the natural basis of the vector space F^{ω}. The vector f_e is called the global encoding kernel for the channel e

Let the source generate a message x in the form of an ω-dimensional row vector. A node v receives the symbols $x.f_d$, $d \in In(v)$, from which it calculates the symbol $x.f_e$ for sending onto each channel $e \in Out(v)$ via the linear Eq. 1:

$$x.f_e = x.\sum_{d\in In(v)} k_{d,e} f_d = \sum_{d\in In(v)} k_{d,e}(x.f_d) \quad (1)$$

UC framework: Universally composable security (UC security) is a kind of computational complexity model which was proposed by Canetti (2001) to represent and analyze cryptographic protocols under concurrent circumstances. The framework provides a rigorous method for defining the security of cryptographic tasks. A protocol is represented as a system of probabilistic Interactive Turing Machines (ITMs), where each ITM represents the program to be run within a different party. Adversarial entities are also modeled as ITMs. An additional adversarial entity called the environment Z is introduced. This environment generates the inputs to all parties, reads all outputs and in addition interacts with the adversary in an arbitrary way throughout the computation.

The real-life model: A probabilistic polynomial-time adversary. A participates in a run of the interactive protocol π with a set of parties $P_1, \ldots, P_l$. All parties are connected through point-to-point communication channels. The channels are public; the adversary can read all data transmitted between parties. The adversary is also responsible for delivery of messages. Each party is initially honest and follows the predetermined program of π. The adversary may corrupt parties, either at the outset only (non-adaptive or static adversaries) or at any point during the execution (adaptive adversaries). Once a party is corrupted by A, the party hands over all internal data including its input, previous incoming and outgoing

communication and the content of the random tape to the adversary. If a party becomes corrupted by the adversary, the party follows the adversary's instructions from then on. Let $REAL_{\pi,A,Z}$ denote the output of environment Z when interacting with adversary A and parties running protocol π.

The ideal process: A probabilistic polynomial-time adversary S (also called simulator) participates in an execution of (dummy) parties $P_1',\ldots,P_1'$ with some ideal and trustworthy functionality F. All parties are only connected to the functionality by secure channels and the simulator can not read the content of transmissions. Once an honest dummy party receives some input, it immediately forwards this input to the functionality, at some point which may reply with output for some parties (including the simulator S). Corruptions are dealt with as in the real-life setting. Let $IDEAL_{F,S,Z}$ denote the output of environment Z after interacting in the ideal process with adversary S and ideal functionality F.

In both settings an interactive distinguisher, the probabilistic polynomial-time environment Z, is present. This environment can interact with honest parties by determining the inputs and reading the output of these parties. Additionally, Z can communicate with the adversary A or S, respectively.

We say that a protocol π UC realizes an ideal functionality F if for any real-life adversary A there exists an ideal-process adversary *S* such that no environment Z, on any input, can tell with non-negligible probability whether it is interacting with A and parties running π in the real-life process or with S and F in the ideal process. We have the following:

Definition 1: Let $l\epsilon L$. Let F be an ideal functionality and let π be an l-party protocol. We say that π securely realizes F if for any adversary A there exists an ideal-process adversary S such that for any environment Z, we define Eq. 2 as follows:

$$IDEAL_{F,S,Z} \approx REAL_{\pi,A,Z} \quad (2)$$

The hybrid model. In order to state the composition theorem and in particular in order to formalize the notion of a real-life protocol with access to an ideal functionality, the hybrid model of computation with access to an ideal functionality F is formulated. The parties of this model may send messages to and receive messages from an unbounded number of copies of F. Each copy of F is identified via a unique session identifier (sid); all messages addressed to this copy and all message sent by this copy carry the corresponding sid. (The sids are chosen by the protocol run by the parties).

SECURE NETWORK CODING SCHEME

In this study, we present a UC secure network coding scheme in view of single-source multi-sinks directed acyclic network. The source messages are divided into m+1 packets, every packet has the same message length. Assume that the network maximum transmission rate is m+1 and each link throughput is 1. We obtain new m+1 packets $\bar{x}_1, \bar{x}_2, \ldots, \bar{x}_m, \bar{x}_{m+1}$ by AONT encryption. Furthermore, the source node transforms $\bar{x}_1, \bar{x}_2\ldots, \bar{x}_m$ into m augmented vectors $x_1, x_2\ldots, x_m$. Finally, the source makes initial sub-signatures for $x_1, x_2\ldots, x_m$ based on the signature mechanism NCS_1 and sends them to participate in encoding at the downstream nodes. The (m+1)th packet $\bar{x}_{m+1}$ is sent directly to the sinks by a shared secret channel between the source and every sink rather than participate in encoding. The intermediate nodes combine and encode the received information and sign the encoded message based on signature mechanism NCS_1, after that, the intermediate nodes and sinks can verify the receiving information, thereby effectively prevent pollution caused by the attacker and ultimately the sinks can decode and decrypt correctly.

The all-or-nothing transform: Rivest (1997) presented a model of encryption for block ciphers which is called All-Or-Nothing Transform (AONT). AONT is defined for information-theoretic security (Stinson, 2001).

Definition 2: Let F_q be a finite field and F^n_q be an n-dimensional space over F_q. Suppose that $\Phi: F^n_q \rightarrow F^n_q$, Φ is named as an (n, q)-AONT if Φ satisfies the following properties:

- Φ is a bijection
- If any m of the m+1 output values $\bar{x}_1, \bar{x}_2\ldots, \bar{x}_m, \bar{x}_{m+1}$ is fixed, then the value of any input value $w_i (1 \leq I \leq m+1)$ is completely undetermined

NCS_1 signature scheme: The signature scheme operates over a bilinear group tuple $\xi = (G_1, G_2, G_T, q, e, \Phi)$ with the following properties:

- G_1, G_2 and G_T are cyclic multiplicative groups of the same prime order q. The discrete logarithm problem is assumed to be computationally infeasible in these groups
- e: $G_1 \times G_2 \rightarrow G_T$ is an efficiently computable map satisfying the following:
  - Bilinearity: for any $g\epsilon G_1$, $h\epsilon G_2$ and a, $b\epsilon Z$, $e(g^a,h^b) = e(g,h)^{ab}$

- Non-degeneracy: if g generates G_1 and h generates G_2, then e(g,h) generates G_T
- φ: $G_2 \rightarrow G_1$ is an efficiently computable isomorphism

Given a security parameter 1^k and a positive integer N, N-m = n, do:

- Generate a bilinear group tuple $\xi = (G_1, G_2, G_T, q, e, \varphi)$, such that $q>2^k$, where k is the security parameter of the system. Generate N-m random numbers $g_1,\ldots,g_n \in G_1\setminus\{1\}$, Choose a generator $h \xleftarrow{R} G_2 \setminus \{1\}$
- Choose $\alpha \xleftarrow{R} F_P^*$ and set $u=h^{\alpha}$, hence $u^{\alpha^{-1}} = h$
- Let $H: \{0,1\}^* \times \{0,1\}^* \rightarrow G_1$ be a hash function
- Output q, the public key PK:= (ξ, H, h, u) and the private key SK:=α

In the following section, We will discuss in detail how NCS_1 signature is applied to the network coding.

NETWORK CODING SCHEME

Source node: In the finite field F_q, we construct a linear ANOT according to the literature (Rivest, 1997), such that $\Phi(w) = wM^{-1}$,where M is an invertible (m+1)-by-(m+1) matrix, function Φ is a linear ANOT. As in definition 2, a file to be transmitted is viewed as an ordered sequence of (m+1)-dimensional vectors $w_1, w_2 \ldots, w_m, w_{m+1} \in F^n_q$. In another word, individual vectors refer to as packets, each w_i consists of n elements from F_q. If the source messages have not enough length, the source node pads the messages with corresponding 0 elements. Through the AONT encryption, we get the new ciphertext as $\bar{x}_1, \bar{x}_2 \ldots, \bar{x}_m, \bar{x}_{m+1}$. The transformation is defined by Eq. 3:

$$\phi(w_1, w_2 \ldots, w_m, w_{m+1}) = (w_1, w_2 \ldots, w_m, w_{m+1})M^{-1} = (\bar{x}_1, \bar{x}_2 \ldots, \bar{x}_m, \bar{x}_{m+1}) \quad (3)$$

Let $q = p^k$, where p is prime and k is a positive integer. Let $\lambda \in F_q$ be such that $\lambda \notin \{m \bmod p, m-1 \bmod p\}$ (this can be done since q>2). We can define $\gamma = (m-\lambda)^{-1}$, define M to be the following symmetric matrix by Eq. 4:

$$M = \begin{pmatrix} 1-\gamma & -\gamma & -\gamma & \cdots & -\gamma & \gamma \\ -\gamma & 1-\gamma & -\gamma & \cdots & -\gamma & \gamma \\ -\gamma & -\gamma & 1-\gamma & \cdots & -\gamma & \gamma \\ \vdots & \vdots & \vdots & \ddots & \vdots & \vdots \\ -\gamma & -\gamma & -\gamma & \cdots & 1-\gamma & \gamma \\ \gamma & \gamma & \gamma & \cdots & \gamma & -\gamma \end{pmatrix} \quad (4)$$

It is straightforward to verify that M is invertible; indeed, M^{-1} is indicated by Eq. 5:

$$M^{-1} = \begin{pmatrix} 1 & 0 & 0 & \cdots & 0 & 1 \\ 0 & 1 & 0 & \cdots & 0 & 1 \\ \vdots & \vdots & \vdots & \ddots & \vdots & \vdots \\ 0 & 0 & 0 & \cdots & 1 & 1 \\ 1 & 1 & 1 & \cdots & 1 & \lambda \end{pmatrix} \quad (5)$$

Next, we start to encode the first packets $\bar{x}_1, \bar{x}_2 \ldots, \bar{x}_m \in F^n_q$, $F^n_q \times F^m_q \rightarrow F^{n+m}_q$ to the input information $\bar{x}_1 \in F^n_q$ and some randomly chosen $r \in F^m_q$. According to the AONT properties, the attacker is unable to get any meaningful information about source even if eavesdropping all of the m coding channel. The source node pads the messages with the m×m identity matrix I by Eq. 6:

$$X = \begin{pmatrix} x_1 \\ x_2 \\ \ldots \\ x_m \end{pmatrix} = \left(\begin{array}{c|c} \bar{x}_1 & \\ \bar{x}_2 & \\ \ldots & \\ \bar{x}_m & \end{array}\right. \left.\begin{array}{c} \\ \\ \\ I \end{array}\right) \quad (6)$$

where, N = n+m and the m augmented vectors $x_1, x_2, \ldots, x_m$ are given by Eq. 7:

$$x_i = (-\bar{x}_i -, \overbrace{0,\ldots,1,0,\ldots}^{m}, 0) \in F^N_q \quad (7)$$

that is, each original vector $\bar{x}_1$ is appended with the vector of length m containing a single '1' in the ith position. The source makes initial sub-signature σ_i for x_i $(1 \le i \le m)$, respectively based on the signature mechanism NCS_1 as follows.

Source sub-signatures (SK, id, v) the source selects randomly a private key SK:= $\alpha \in F_q$, given an identifier $id \in \{0,1\}^k$, the source computes signature σ_i on vector $x_i = (x_{i,1}, \ldots, x_{i,N}) \in F^N_q$ by Eq. 8:

$$\sigma_i = (\prod_{i=1}^{m} H(id,i)^{x_{i,n+i}} \prod_{j=1}^{n} g_j^{x_{i,j}})^{\alpha} \quad (8)$$

Intermediate nodes: First verify the signatures on the received signed packets, then produce linear combinations of the verified packets and compute a signature for the encoded packet using the received signatures without the need to access the private keys.

At this point, the combining process has to check if all input packets that contain a signature for an x_i contain the same value of σ_i. If not, the combining process fails

and an adversary is recognized. Setting y is the encoded payload and y is defined by Eq. 9:

$$y = \beta_{i,1}x_1 + \beta_{i,2}x_2 + \ldots + \beta_{i,m}x_m = (\beta_{i,1},\ldots,\beta_{i,m})\begin{pmatrix} x_1 \\ \ldots \\ x_m \end{pmatrix} \quad (9)$$

$\beta_i = (\beta_{i,1},\ldots, \beta_{i,m})$ is the corresponding randomly chosen global encoding kernel. All intermediate nodes in the network to sign and verify received packets as follows:

- Combine(PK, id, $\{(\beta_i, \sigma_i)\}_{i=1}^{l}$): Given a public key PK, an identifier id and $\{(\beta_i, \sigma_i)\}_{i=1}^{l}$ with $\beta_i \epsilon F_q$, this algorithm outputs:

$$\sigma = \prod_{i=1}^{m}(\sigma_i)^{\beta_i} \quad (10)$$

- Verify (PK,id,y,σ): Given a public key PK:= (ξ, H, h, u), an identifier id, a signature σ and a vector y = $(y_1,y_2,\ldots,y_N)\epsilon F$, we describe Eq. 11 and 12, respectively:

$$\Upsilon_1(PK,\sigma) \overset{def}{=} e(\sigma,h) \quad (11)$$

$$\Upsilon_2(PK,id,y) \overset{def}{=} e(\prod_{i=1}^{m}H(id,i)^{y_{m+i}}\prod_{j=1}^{n}g_j^{y_j},u) \quad (12)$$

If $\Upsilon_1(PK, \sigma) = \Upsilon_2(PK, id, y)$, this algorithm outputs 1; otherwise it outputs 0.

Sink node: The sinks receive the former level transmitted message and verify the signature as the above method. The verified message can be ultimately decoded and decrypted. If $t_i \epsilon T$, the sink t_i final received message is message Y that has a valid signature. Let A_m be system transfer matrix of input vector X and the output vector Y, where $Y=A_m$. X given by Eq. 13:

$$Y = \begin{pmatrix} f_{1,1},\ldots,f_{1,m} \\ \ldots \\ f_{m,1},\ldots,f_{m,m} \end{pmatrix}\cdot\begin{pmatrix} x_1 \\ \ldots \\ x_m \end{pmatrix} = A_m\cdot\begin{pmatrix} x_1 \\ \ldots \\ x_m \end{pmatrix} \quad (13)$$

If the sink t_i receives correct information from at least m incoming channels and global encoding kernel of the m incoming channels are linearly independent, only in this way can t_i recovery source information. Koetter R (Koetter and Medard, 2003) has proved a coding scheme to be secure, as long as the system transfer matrix $A_m=A(I-F)^{-1}B^T$ is full rank, the destination can decode correctly and output $X = A_m^{-1}.Y$. The vector X is column vector and we write X^T to denote the corresponding row vector, that is $X^T = (x_1,..,x_m)$,obviously we can obtain $\bar{x}_1$, $\bar{x}_2\ldots$, $\bar{x}_m$, then we have $\bar{x}_1$, $\bar{x}_2\ldots$, $\bar{x}_m$, $\bar{x}_{m+1}$, where, $\bar{x}_{m+1}$ has been transmitted through a secret channel. Finally, we will obtain source original information by AONT inverse transformation matrix M. The process can be described by Eq. 14:

$$W = (w_1, w_2\ldots, w_m, w_{m+1}) = (\bar{x}_1, \bar{x}_2\ldots, \bar{x}_m, \bar{x}_{m+1})M \quad (14)$$

PROOF SECURITY OF CODING SCHEME

Our protocols are based on the following cryptographic assumptions. Adversaries can adaptively corrupt as many parties as they wish. The protocol is secure against adaptive adversaries in the so-called non-erasing model, where the parties are not allowed to erase any information. Our protocols need UC security even in non-erasing model.

Construction of UC secure network coding protocol: In this section, we formulate a universally composable network coding scheme π_{coding} in (F_{CPKE}, F_{sig})-hybrid model. Here, F_{CPKE} is the encryption ideal functionality and F_{sig} is the signature ideal functionality. In the following graphs, the definition of function Φ, matrix M, M^{-1}, y and Y as above. We start by presenting F_{CPKE} and F_{sig}. Next we present the real protocol π_{coding}. According to the UC secure framework (Canetti, 2004), we modify the definition of F_{CPKE} (Canetti and Herzog, 2006), as indicated in Fig. 1.

Functionality F_{CPKE}

F_{CPKE} proceeds as follows, when parameterized by message domain W, a encryption function E with domain W and range $\{0,1\}^*$ and a decryption function D of domain $\{0,1\}^*$ and range W error. The sid is assumed to consist of a pair sid = (pid, sid′), where pid is the identity of the owner of this instance.

Encryption

Upon receiving a value (Encrypt,sid,wi) from some party P proceeds as follows.

(1) If $w_i \notin W$, then return an error message to P.

(2) If $w_i \in W$, then:

If party pid is corrupted, sends w_i to the adversary S and receives a value $\bar{x}_1$, else, computes the value $\bar{x}_1 = E(w_i)$ and then records $(w_i, \bar{x}_1)$.

Returns x_i to P (If $(w_i, \bar{x}_1)$ was recorded from another party then ignore)

Decryption

Upon receiving a value (Decrypt, sid, $\bar{x}_1$) from the owner of this instance, proceeds as follows.

(1) If there is a recorded pair $(w_i, \bar{x}_1)$ for some w_i, then hands w_i to P.

(2) Otherwise, compute $w_i = D(\bar{x}_1)$ and hands w_i to P.

Fig. 1: The certified public-key encryption functionality

Functionality F_{sig}

Key generation

Upon receiving a value (KenGen,sid) from some party P, verifies that sid=(P, sid′) for some sid′. If not, then ignores the request. Else, hands (KenGen,sid) to the adversary S. Upon receiving (Verificationkey,sid,ZS,C,V) from the adversary S, outputs (Verificationkey,sid,V) to P.

Sub-signature generation

Upon receiving a value (Sign,sid,x_i) from P, verifies that sid = (P, sid′) for some sid′. If not, then ignores the request. Else, sends (Sign,sid,x_i) to the adversary S. Upon receiving (signature,sid,x_i,σ_i) from the adversary, where $\sigma_i = ZS(x_i)$, verifies that no entry (x_i,σ_i,0) is recorded. If it is, then outputs an error message to P and halt. Else, outputs (signature,sid,x_i,σ_i) to P and records the entry (x_i, σ_i,1).

Combination of Sub- signatures

Upon receiving a value (Encoding,Combine,x_1,σ_1,...,x_m,σ_m) from some party C, sends (Encoding, Combine,x_1,σ_1,...,x_m,σ_m) to the adversary S. Upon receiving a value (y,σ,C) from the adversary S, sends (y,σ,C) to C.

Signature Verification

Upon receiving a value (Verify,sid,y,σ,V) from some party V, hands (Verify,sid,y,σ,V′) to the adversary S. Upon receiving a value (Verified,sid,y,φ) from the adversary do:

(1) If V=V′ and the entry (y,σ,V,1) is recorded, then sets f=1.
(2) Else, if V=V′, the signer is not corrupted and no entry (y,σ′,V,1) for any σ′ is recorded, then sets f=0,and records the entry (y,σ,V,0).
(3) Else, if there is an entry (y,σ, V′, f ′) recorded, then lets f= f ′.
(4) Else, let f=Φ and records the entry (y,σ, V′,Φ)?
(5) Outputs (Verified,sid,y,f) to V.

Fig. 2: The functionality of network coding signature

According to the UC secure framework, we modify the definition of F_{sig} (Canetti, 2001) as indicated in Fig. 2. In Fig. 2, *ZS* denotes sub-signature function and C combination function.

Note that all our functionalities receive initially a session id sid=(P, sid′) where P is the set of players who participate in realizing the functionality and sid′ is a number identifying this particular instance of the functionality. Next we present the real protocol π_{coding} in Fig. 3.

Construction of network coding against pollution attacks ideal functionality F_{coding}: In this section, we present our formulation of an ideal functionality for network coding against pollution attacks in the UC framework. An exact definition of the ideal functionality, denoted F_{coding}, appears in Fig. 4.

SECURITY PROOF OF π_{CODING}

Theorem 1: Protocol π_{coding} securely realizes F_{coding} in the (F_{CPKE}, F_{sig})-hybrid model.

Proof: Let A be an adaptive adversary that interacts with the parties running π_{coding} in the (F_{CPKE}, F_{sig})-hybrid model. We construct an ideal adversary S such that any

Protocol π_{coding}

Source encryption

If source S obtains a value (Encrypt,sid,w_i) from dummy sourceS′, then sends (Encrypt,sid,w_i) to F_{CPKE} and F_{CPKE} returns $\bar{x}_i$ to S. S does the following transformation and get x_i (1=i=m+1).

$$x_i = (-\bar{x}_1 - , \underbrace{0,\ldots,1}_{i}, \overbrace{0,\ldots,0}^{m}) \in F_q^N \quad (a)$$

Source sub-signatures

Source S obtains a value (Sign,sid,x_i) with sid=(P, sid′), sends (Sign,sid,x_i) to F_{sig} and F_{sig} returns (signature,sid,x_i,σ_i) to S.

Intermediate nodes Encoding, Signature and Verification

F_{sig} obtains (Encoding,Combine,x_1,σ_1,...,x_m,σ_m) from some party C and returns (y,σ,C) to C. F_{sig} obtains (Verify,sid,y,σ,V) from verifier V, then sends (Verified,sid,y,f) to V.

Sinks decoding

Every sink t implements (Decoding,sid,Y) by computing formula (b) as follows and gets X, further gets $\bar{x}_i$ (1≤i≤m) according to formula (a). The (m+1)th packet $\bar{x}_{m+1}$ is sent directly to the sink by a shared secret channel between the source and the sink, then obtains $\bar{x}_1$, $\bar{x}_2$..., $\bar{x}_m$, $\bar{x}_{m+1}$.

$$X = \begin{pmatrix} x_1 \\ x_2 \\ \cdots \\ x_m \end{pmatrix} = A_m^{-1} \cdot Y \quad (b)$$

Sinks decryption

F_{CPKE} obtains a value (Decrypt,sid, $\bar{x}_i$), then:

(1) If there is a recorded pair (w_i, $\bar{x}_i$) for some w_i, then hands w_i to t.
(2) Otherwise, compute $w_i = D(\bar{x}_i)$ and hands w_i (1=i=m+1) to t.

Fig. 3: The protocol π_{coding}

environment Z can not distinguish with a non-negligible probability whether it is interacting with A and π_{coding} in the (F_{CPKE}, F_{sig})-hybrid model (denoted $REAL_{UC\text{-}coding,A,Z}$) or it is interacting with S and F_{coding} in the ideal world (denoted $IDEAL_{Fcoding,S,Z}$).

Construction of the adversary S: The adversary S shown below runs a simulated copy of the adversary A, thus S is often called a simulator. Any input from Z is forwarded to A and any output of A is copied to the output of S.

Simulating the sender: When an uncorrupted party P is activated with input (Encrypt,sid,w_i), S obtains this value from F_{coding} and simulates for A the protocol π_{coding}.

Whenever, S obtains (Encrypt,sid,w_i) from F_{CPKE}, S sends the message (Encrypt,sid,w_i) to A with sid = (P, sid′), then forwards the response x_i (1≤i≤m+1)from A to F_{sig}.

Whenever, S receives a message (KenGen,sid) from F_{sig}, S sends to A the message (KenGen,sid) with sid = (P, sid′), then forwards the response (Verification key, sid, ZS, C, V) from A to F_{sig}.

Whenever, S receives a message (Sign, sid, x_i) from F_{sig}, S sends (Sign,sid,x_i) to A, then forwards the response (signature, sid, x_i, σ_i) from A to F_{sig}.

Functionality F_{coding}

Source encryption

If source S obtains a value (Encrypt,sid,w_i) from dummy source S′, then sends (Encrypt,sid,w_i) to the adversary S. Upon receiving x_i(1=i=m+1) from the adversary S, sends x_i (1=i=m+1) to S.

Source sub-signatures

Source S obtains a value (Sign,sid,x_i) with sid=(P, sid′), then sends (Sign,sid,x_i) to the adversary S. Upon receiving a value (signature,sid,x_i,σ_i) from S, outputs (signature,sid,x_i,σ_i) to S.

Intermediate nodes Encoding, Signature and Verification

Upon receiving a value (Encoding,Combine,x_1,σ_1,...,x_m,σ_m) from C, sends (Encoding, Combine, x_1,σ_1,...,x_m,σ_m) to the adversary S. Upon receiving (y,σ,C) from the adversary S, outputs (y,σ,C) to C. Upon receiving a value (Verify,sid,y,σ,V) from some party V, sends (Verify,sid,y,σ,V′) to the adversary S. Upon receiving a value (Verified, sid,y,Φ) from the adversary do:

(1) If V = V′ and the entry (y,σ,V,1) is recorded, then sets f = 1.

(2) Else, if V = V′,the signer is not corrupted and no entry (y,σ′,V,1) for any σ′ is recorded, then sets f = 0 and records the entry (y,σ,V,0).

(3) Else, if there is an entry (y,σ, V′, f ′) recorded, then lets f= f ′.

(4) Else, let f = φ and records the entry (y,σ, V′,Φ).

(5) Outputs (Verified,sid,y,f) to V.

Sinks decoding

Upon receiving a value (Decoding,sid,Y) after verification, sends (Decoding,sid,Y) to the adversary S. S return X, The (m+1)th packet $\bar{x}_{m+1}$ is sent directly to the sink by a shared secret channel between the source and the sink, then obtains $\bar{x}_1$, $\bar{x}_2$..., $\bar{x}_m$, $\bar{x}_{m+1}$.

Sinks decryption

Upon receiving a value (Decrypt,sid, $\bar{x}_i$) from the sink t, then:

(1) If there is a recorded pair (w_i, $\bar{x}_i$) for some w_i, then hands w_i to t.

(2) Otherwise, compute $w_i = D(\bar{x}_i)$ and hands w_i (1≤i≤m+1) to t.

Fig. 4: Secure network coding ideal functionality

Simulating the intermediate nodes: Whenever S receives a message (Encoding,Combine,x_1,σ_1,...,x_m,σ_m) from F_{sig}, S sends (Encoding,Combine,x_1,σ_1,...,x_m,σ_m) to A, then forwards the response(y,σ,C) from A to F_{sig}.

Simulating the recipients: When A delivers a message (y,σ,C) to an uncorrupted party t, S simulates for A the protocol π_{coding}.

Whenever, S receives a message (Verify,sid,y′,σ,V) from F_{sig}, S sends (Verify,sid,y′,σ,V) to A, where a sink receives a encoded message y′ at a previous level, then forwards the response (Verified,sid, y′, f) from A to F_{sig}.

If F_{CPKE} receives (Verify,sid, y′,σ,V,f=1) from F_{sig}, then records (x,V) and sends (y,σ,C) to t. Otherwise, do nothing.

If t receives (Decrypt, sid, $\bar{x}_i$) from F_{CPKE}, if there is a recorded pair (w_i, $\bar{x}$) then hands w $_i$to t. Otherwise, computes $w_i = D(\bar{x}_i)$ and hands w_i (1≤i≤m+1) to t.

Simulating party corruption: Whenever, a corrupts a party, *S* corrupts the same party and provides A with the internal state of the corrupted party. Whenever a corrupted party P receives (Encrypt, sid, w_i), S obtains x_i from F_{coding} and simulates for A only the interaction with F_{sig} and F_{CPKE}. Whenever S obtains (Encrypt,sid,w_i) from F_{CPKE}, S sends the message (Encrypt, sid, w_i) to A with sid = (P, sid′), then forwards the response x_i (1≤i≤m+1)from A to F_{sig}. Whenever, S receives a message (KenGen,sid) from F_{sig}, S sends to A the message (KenGen,sid) with sid = (P,sid′), then forwards the response (Verificationkey, sid, ZS, C, V′) from A to F_{sig}. (V′ may be different from V). Whenever, S receives a message (Sign, sid, x_i) from F_{sig}, S sends (Sign, sid, x_i) to A, then forwards the response (signature, sid, x_i, σ_i) from A to F_{sig}. Assume that π_{coding} can not securely realize F_{coding} then existing Z can distinguish with a non-negligible probability whether it is interacting with A and π_{coding} or it is interacting with S and F_{coding}. Assume that we can construct an adversary B who can forge signature. B can activate environment Z, when C is activated with input (Encoding,Combine,x_1,σ_1,...,x_m,σ_m) by Z and B computes the signature of x_i by sub-signatures combination algorithm and submits it to Z. When P is activated with input (Verify,sid,y,σ,V) by Z, B first checks whether (y,σ,C) is a forged signature or not. If it is a forged signature, then B outputs (y,σ,C) and stops operation. Next, we analysis the probability of success of B. We first denote D as the event where the receiver obtains (Verified,sid,y,f = 1),while party P is uncorrupted at the time when the message is delivered and has never sent (y,σ,C). However, according to the protocol and the logics of F_{CPKE} and F_{sig}, the event D would not happen. The reason being is that, firstly the receiver should obtain a valid verified key from F_{sig} (Otherwise (Verified,sid,y,f = 1) would not be sent by F_{sig}); secondly, if an uncorrupted P never sent (y,σ,C), then the message y is never signed by F_{sig}. Thus, P would always obtain (Verified,sid,y ,f = 0) from F_{sig}. Otherwise, this is incompatible with the existential unforgeability(EU-COMA) (Even *et al*., 1989; Canetti, 2001) property of F_{sig}. Therefore, the simulation above is perfect based on the fact that the event D will not occur. In another word, Z can not distinguish with a non-negligible probability whether it is interacting with A and π_{coding} or it is interacting with S and F_{coding} Thus, we realize indistinguishability of $REAL_{UC\text{-}coding,A,Z}$ and $IDEAL_{Fcoding,S,Z}$, in other words, π_{coding} securely realizes the functionality F_{coding} in the (F_{CPKE}, F_{sig})-hybrid model. (If we use y′ instead of y, the operation is in the same way).

COMPARISON WITH RELATED WORK

The scheme for network coding against pollution attacks in this article is compared with other representative schemes in Table 1, including security model, the adversary's capability model and complexity of computation. The literature (Li *et al*., 2010) gave a MAC-based approach supporting in-network verification and

Table 1: Comparison between the different schemes for secure network coding

	Li *et al.* (2010)	Feng *et al.* (2011)	Our scheme
Security model	computational hardness	computational hardness	UC security
Adversaries model	adaptive+erase	adaptive+ non-erase	adaptive + non-erase
Complexity of computation.	MAC: $L(n+m+\frac{L-1}{2})$ mults Combine: Verify: $n+m+\frac{L-1}{2}$	O(logN)+ $\log_g y^r r^s$	Sign: N exps, N-1 mults. Combine: mN exps, m (N-1) mults Verify: Υ_1: m mappings, m exps, m-1 mults. Υ_2: N modulo exps, N-1 mults, one mapping

In the table above, 'Mults' is the abbreviation of multiplications and 'exps' is the abbreviation of exponentiations

resisting an arbitrary number of collusions. The adversary does not have access to the randomness used by the source in order to produce the various cryptographic keys. The computation complexity includes three algorithms: generation MAC, combination of signatures and verification. The computation complexity of literature (Feng *et al.*, 2011) includes two parts: source encryption's complex is O (logN) and intermediate nodes's discrete logarithm complex $\log_g y^r r$.

Our protocol is secure against adaptive adversaries in the so-called non-erasing model, Existing equipment or software can achieve AONT encryption, so we need not consider the encryption computational complexity. The computational complexity is mainly reflected in the following aspects. Signing a raw packet requires N exponentiations and N-1 multiplications. Combining is extremely simple, m packets require mN exponentiations and m (N-1) multiplications. Verification is computationally more expensive, if m packets are combined together, computing Υ_1 requires m mappings, m exponentiations and m-1 multiplications, while Υ_2 requires N modulo exponentiations, N-1 multiplications and one mapping.

CONCLUSION

We presented a new random network coding scheme within the framework of Universally Composable (UC). In this work, we realized indistinguishability of wiretapping messages by means of ANOT encryption. In order to prevent malicious nodes, we adopted the signature mechanism NCS_1 to authenticate nodes identity and the transmitted data. Finally, according to the composition theorem, our construction using F_{CPKE} and F_{sig} was a secure network coding scheme. Through comparison between the different representative schemes, show that the new scheme having better performance. However, the scheme NCS_1 is based on bilinear maps which may require more powerful computing capabilities for the intermediate nodes. Our next work is to design protocol using more effective signature scheme in network coding.

ACKNOWLEDGMENTS

First and foremost, I am most grateful to my senior, Professor Mr Guo, whose useful suggestions, incisive comments and constructive criticism have contributed greatly to the completion of this paper. I am also greatly indebted to all my friends who have helped me directly and indirectly in my studies. Any progress that I have made is the result of their profound concern and selfless devotion. Among them the following require mentioning: Mr. Liang, Miss Chen and Mr. Jiao.

REFERENCES

Agrawal, S. and D. Boneh, 2009. Homomorphic MACs: MAC-based integrity for network coding. Applied Cryptography Network Security, 5536: 292-305.

Ahlswede, R., N. Cai, S.Y.R. Li and R.W. Yeung, 2000. Network infomation flow. IEEE Trans. Inform. Theor., 46: 1204-1216.

Boneh, D., D. Freeman, J. Katz and B. Waters, 2009. Signing a linear subspace: Signature schemes for network coding. Proceedings of the 12th International Conference on Practice and Theory in Public Key Cryptography, March 18-20, 2009, Irvine, CA, USA., pp: 68-87.

Cai, N. and R.W. Yeung, 2002. Network coding and error correction. Proceedings of the 2002 IEEE Information Theory Workshop, October 20-25, 2002, Bangalore, India, pp: 119-122.

Canetti, R. and J. Herzog, 2006. Universally composable symbolic analysis of mutual authentication and key exchange protocols. Theory Cryptography, 3876: 380-403.

Canetti, R., 2001. Universally composable security: A new paradigm for cryptographic protocols. Proceedings of the 42nd IEEE Symposium on Foundations of Computer Science, October 14-17, 2001, Las Vegas, Nevada , pp: 136-145.

Canetti, R., 2004. Universally composable signature, certification and authentication. Proceedings of the 17th IEEE Computer Security Foundations Workshop, June 28-30, 2004, California, pp: 219-235.

Chanl, T. and A. Grant, 2008. Capacity bounds for secure network coding. Proceedings of the Australian Communications Theory Workshop, January 30-February 1, 2008, Christchurch, pp: 95-100.

Even, S., O. Goldreich and S. Micali, 1989. On-Line/Off-Line Digital Signatures. In: Advances in Cryptology-CRYPTO 89, Brassard, G. (Ed.), LNCS 435, Springer-Verlag, New York, pp: 263-277.

Feng, T., B.T. Zhang and J.F. Ma, 2011. Security random network coding model against byzantine attack based on CBC. Proceedings of the 4th International Conference on Intelligent Computation Technology and Automation, March 28-29, 2011, Shenzhen, China, pp: 1178-1181.

Gennaro, R., J. Katz, H. Krawczyk and T. Rabin, 2010. Secure network coding over the integers. Public Key Cryptography, 6056: 142-160.

Hayat, K.A., U.W. Anis and Tauseef ur Rahman, 2004. Password interception in a SSL/TLS channel. Inform. Technol. J., 3: 327-331.

Ho, T., B. Leong, R. Koetter, M. Medard, M. Effros and D.R. Karger, 2004. Byzantine modification detection in multicast networks using randomized network coding. Proceedings of the International Symposium on Information Theory, June 27-July 2, 2004, Chicago, IL., pp: 144.

Jaggi, S., M. Langberg, S. Katti, T. Ho, D. Katabi, M. Medard and M. Effros, 2008. Resilient network coding in the presence of byzantine adversaries. IEEE Trans. Inform. Theor., 54: 2596-2603.

Jun, D. and L. Wei-Hua, 2010. A security routing optimization scheme for multi-hop wireless networks. Inform. Technol. J., 9: 506-511.

Koetter, R. and M. Medard, 2003. An algebraic approach to network coding. IEEE/ACM Trans. Networking, 11: 782-796.

Lamport, L., R. Shostak and M. Pease, 1982. The byzantine generals problem. ACM Trans. Programm Languages Syst., 4: 382-401.

Li, Y., H. Yao, M. Chen, S. Jaggi and A. Rosen, 2010. ripple authentication for network coding. Proceedings of the 29th Conference on Information Communications, March 15-19, 2010, San Diego, USA., pp: 14-19.

Meng, B., 2011. A survey on analysis of selected cryptographic primitives and security protocols in symbolic model and computational model. Inform. Technol. J., 10: 1068-1091.

Raja, P.C.K., M. Suganthi and R. Sunder, 2008. Wireless node misbehavior detection using genetic algorithm. Inform. Technol. J., 7: 143-148.

Rivest, R.L., 1997. All-or-nothing Encryption and the package transform. Fast Software Encryption, 1267: 210-218.

Silva, D. and F.R. Kschischang, 2008. Security for wiretap networks via rank-metric codes. Proceedings of the IEEE International Symposium on Information Theory, July 6-11, 2008, Toronto, ON., pp: 176-180.

Stinson, D.R., 2001. Something about all or nothing (transforms). Designs Codes Cryptography, 22: 133-138.

Wang, S.C., K.Q. Yan and C.F. Cheng, 2003. Byzantine agreement under unreliable multicasting network. Inform. Technol. J., 2: 104-115.

Wang, Y., 2010. Insecure "Provably secure network coding" and homomorphic authentication schemes for network coding. IACR Eprint archive. http://www.iacr.org/cryptodb/data/paper.php?pubkey=22961

Wei, W., P. Qian, B. Zhang and B. Huang, 2011. Adaptive symbol-level network coding for broadcasting retransmission. Inform. Technol. J., 10: 1264-1267.

Yan, K.Q. and S.C. Wang, 2004. Detecting the faulty communication links under the loose coupled distributed system. Inform. Technol. J., 3: 275-282.

Yu, Z., Y. Wei, B. Ramkumar and Y. Guan, 2008. An efficient signature-based scheme for securing network coding against pollution attacks. Proceedings of the 27th Conference on Computer Communications, April 13-18, 2008, Phoenix, AZ, USA., pp: 1409-1417.

Zaidan, B.B., A.A. Zaidan, A.K. Al-Frajat and H.A. Jalab, 2010. On the differences between hiding information and cryptography techniques: An overview. J. Applied Sci., 10: 1650-1655.

Zhang, Z., 2008. Linear network error correction codes in packet networks. IEEE Trans. Inf. Theory, 54: 209-218.

Zhou, X., Y. Yang, H. Shan and Z. Wang, 2010. Performance study of a network coded non-orthogonal user cooperation system over Nakagami-m channels. Inform. Technol. J., 9: 1353-1360.

Zhou, Y.J., H. Li and J.F. Ma, 2009. Random network coding against the eavesdropping adversaries. J. Xidian Univ., 36: 696-701.