Secure and Efficient Certificateless Homomorphic Signature Scheme for Network Coding

Hao Huang*, Xiaofen Wang*, Man Ho Allen Au†, Sheng Cao*, Qinglin Zhao‡, Xiaosong Zhang*
*University of Electronic Science and Technology of China, Chengdu, China
Email: hh23571113@163.com, xfwang@uestc.edu.cn, caosheng@uestc.edu.cn, johnsonzxs@uestc.edu.cn
†The Hong Kong Polytechnic University, Hung Hom, China. Email: mhaau@polyu.edu.hk
‡Macau University of Science and Technology, Macau, China. Email: qlzhao@must.edu.mo

***Abstract*—Network coding, as a routing technology to improve network throughput and robustness, is widely used in various scenarios. However, network coding is vulnerable to pollution attacks where nodes may maliciously modify transmitted data packets. Recently, certificateless linearly homomorphic signature schemes have been proposed to resist pollution attacks in network coding, which avoids burdensome certificate management and the key-escrow issue. In this paper, we show that Wu et al.'s certificateless homomorphic network coding signature (CHNCS) scheme, Chang et al.'s CHNCS scheme, and Li et al.'s CHNCS are not secure against pollution attacks in network coding. Then we present a secure and efficient CHNCS scheme and prove it is unforgeable against adaptive chosen identity-and-subspace attacks under two types of adversaries. Finally, performance analysis illustrates that our scheme is more efficient in practical application.**

***Index Terms*—network coding, pollution attacks, certificateless, homomorphic signature**

I. Introduction

Network coding [1] is a new transmission paradigm in which the incoming data packets are encoded at intermediate nodes and relayed to the other nodes. More precisely, data packets are seen as vectors in a linear space over some field and the encoding are linear combinations of these vectors. Compared with the traditional forwarding mechanisms which simply store and forward data packets, network coding can improve network throughput [2], robustness [3], and energy efficiency [4]. Because of these advantages, network coding has been applied in many fields, such as wireless networks, distributed data storage, peer-to-peer systems, etc.

Unfortunately, network coding is vulnerable to pollution attacks, that is, when intermediate nodes encode the input packets containing invalid data packets, the output packet is also invalid. Since corrupted packets will be propagated to the entire network, it wastes network resources, and destination nodes cannot decode the received packets to recover the original data. To resist pollution attacks, Boneh et al. [5] first introduced the formal definition of linearly homomorphic signature and proposed a concrete scheme over bilinear groups, which can authenticate valid packets and discard corrupted packets. Since their scheme is the traditional cryptosystem, the generation, validation, and management of public key certificates were involved. Thus, identity-based linearly homomorphic signature schemes [6], [7] were proposed to avoid certificate management. However, these schemes introduce the key-escrow issue, and Zhou et al. [8] found that the scheme [6] cannot resist pollution attacks for network coding.

To avoid burdensome certificate management and the key-escrow issue, certificateless linearly homomorphic signature has been recently proposed to resist pollution attacks in network coding. In 2020, Chang et al. [9] first proposed a concrete certificateless homomorphic network coding signature (CHNCS) scheme to resist pollution attacks, which is constructed on an identity-based homomorphic signature scheme proposed by Lin et al. [7] in 2018. Their solution combined a CHNCS scheme and a certificateless signature (CLS) scheme, resulting in high computational complexity. Later in 2021, Wu et al. [10] proposed a concrete CHNCS scheme without using the CLS scheme. In 2023, Bian et al. [11] designed a certificateless network coding signature scheme based on the certificateless public auditing protocol. To improve efficiency, Li et al. [12] proposed a lightweight certificateless linearly homomorphic network coding signature scheme for the electronic health system. After that, Yu et al. [13] proposed a certificateless homomorphic network coding signature scheme without bilinear pairing. Unfortunately, these schemes [10]–[13] cannot resist pollution attacks in network coding.

In this paper, we launch a Type-I adversary attack on Wu et al.'s CHNCS scheme [10], a Type-II adversary attack on Chang et al.'s CHNCS scheme [9], and a universal attack (i.e., Type-I or Type-II adversary) on Li et al.'s CHNCS scheme [12]. Specifically, an outside adversary can replace any user's public key to generate a valid signature on the corrupted data packet in Wu et al.'s CHNCS scheme [10], a malicious-but-passive key generation center (KGC) can obtain the private key of the victim in Chang et al.'s CHNCS scheme [9], and any adversary can forge a valid signature on any corrupted data packet in Li et al.'s CHNCS scheme [12]. Furthermore, Bian et al.'s CHNCS scheme [11] and Yu et al.'s CHNCS scheme [13] are also vulnerable to similar pollution attacks. Next, we propose a secure and efficient CHNCS scheme to resist pollution attacks in network coding, and prove it is unforgeable against adaptive chosen identity-and-subspace attacks under

This work is supported by the National Key Research and Development Program of China (2023YFB3105901) and the National Natural Science Foundation of China (62372092, U2336204). (Corresponding author: Xiaofen Wang.)

the above two types of adversaries in the random oracle model (ROM). Finally, the performance analysis illustrates that our scheme is more efficient for network coding.

The remainder of this paper is organized as follows: Section II describes preliminary knowledge related to this work. In Section III, we give concrete attacks on Wu et al's CHNCS scheme, Chang et al.'s CHNCS scheme, and Li et al.'s CHNCS scheme respectively. Section IV presents our new CHNCS scheme and its security analysis. In Section V, the communication costs and the computation costs of our scheme are analyzed. Finally, conclusions are made in Section VI.

II. Preliminaries

A. Bilinear Groups and Assumption

Definition 1 (Bilinear Groups)**.** *Let $\mathbb{G}_1$ and $\mathbb{G}_2$ be two cyclic groups with the same prime order p, and $e : \mathbb{G}_1 \times \mathbb{G}_1 \to \mathbb{G}_2$ be a bilinear map which can be efficiently computable with the following properties.*

1) *Bilinearity: For any $g_1, g_2 \in \mathbb{G}_1$, and $a, b \in \mathbb{Z}_p$, $e(g_1^a, g_2^b) = e(g_1, g_2)^{ab}$.*
2) *Non-degeneracy: If any two generators $g_1, g_2 \in \mathbb{G}_1$, then $e(g_1, g_2)$ is a generator of $\mathbb{G}_2$.*

Definition 2 (Computational Diffie-Hellman Assumption)**.** *Assume $\mathbb{G}_1, \mathbb{G}_2$ are cyclic groups of the same prime order p, g is a generator of $\mathbb{G}_1$. The Computational Diffie-Hellman (CDH) assumption holds if for any probabilistic polynomial time (PPT) adversary $\mathcal{A}$, given a tuple $(g, g^a, g^b) \in \mathbb{G}_1^3$ as input, the probability to output g^{ab}*

$$Pr[\mathcal{A}(1^\lambda, g, g^a, g^b) \to g^{ab}] \leq negl(\lambda)$$

holds for arbitrary security parameter λ, where $negl(\cdot)$ is a negligible function.

B. The Formal Definition of Certificateless Homomorphic Network Coding Signature Scheme

A certificateless homomorphic network coding signature scheme is defined by the following PPT algorithms.

Setup$(\lambda, N, m) \to (msk, params)$: Taking as input a security parameter λ and two positive integers N, m, this algorithm outputs a master key msk and the public parameters $params$.

Partial-Private-Key-Extract$(params, msk, ID) \to (d_{ID})$: Taking as input $params$, the master key msk, and a user identity ID, this algorithm outputs the partial private key d_{ID}.

Set-Secret-Value$(params, ID) \to (x_{ID})$: Taking as input $params$ and a user identity ID, this algorithm outputs a secret value x_{ID}.

Set-Private-Key$(params, d_{ID}, x_{ID}) \to (SK_{ID})$: Taking as input $params$, a partial private key d_{ID}, and a secret value x_{ID}, this algorithm outputs the private key SK_{ID}.

Set-Public-Key$(params, ID, x_{ID}) \to (PK_{ID})$: Taking as input $params$, a user identity ID, and a secret value x_{ID}, this algorithm outputs the public key PK_{ID}.

Sign$(params, ID, SK_{ID}, id, \mathbf{v}) \to (\sigma)$: Taking as input $params$, a user identity ID, a private key SK_{ID}, a file identifier id, and a vector $\mathbf{v}$, this algorithm outputs a signature σ of the vector $\mathbf{v}$.

Combine$(params, ID, PK_{ID}, id, \{(c_i, \mathbf{v}_i, \sigma_i)\}_{i=1}^{l}) \to (\mathbf{v}, \sigma)$: Taking as input $params$, a user identity ID, a public key PK_{ID}, a file identifier id, and a set of tuples $\{(c_i, \mathbf{v}_i, \sigma_i)\}_{i=1}^{l}$ $(l{>}0)$ with $c_i \in \mathbb{Z}_p$, this algorithm outputs a vector/signature pair $(\mathbf{v}, \sigma)$. If each σ_i is a valid signature on the vector $\mathbf{v}_i$, σ is a valid signature on $\mathbf{v} = \sum_{i=1}^{l} c_i \mathbf{v}_i$.

Verify$(params, ID, PK_{ID}, id, \mathbf{v}, \sigma) \to 1/0$: Taking as input $params$, a user identity ID, a public key PK_{ID}, a file identifier id, a vector $\mathbf{v}$, and the corresponding signature σ, this algorithm outputs either 1 (accept) or 0 (reject).

C. System Model

The system model of CHNCS-based secure network coding is shown in Fig. 1. In the system, there are the KGC and three types of nodes including the source nodes, intermediate nodes, and destination nodes. The system procedure is as follows.

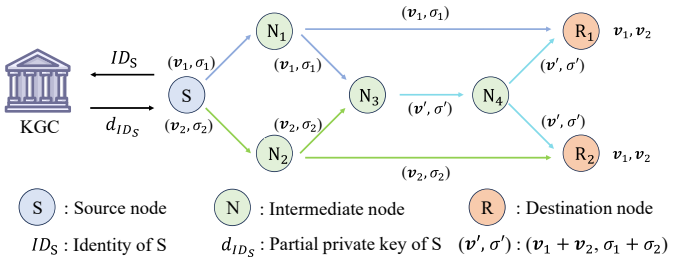

Fig. 1. CHNCS-based secure network coding system

- The source node completes the system registration by sending identity to the KGC, and computes its private key and public key with the help of the KGC.
- The original file transmitted in the system is treated as a sequence of n-dimensional vectors $\bar{\mathbf{v}}_1, \cdots, \bar{\mathbf{v}}_m \in \mathbb{Z}_p^n$, where p is prime and typically $m \ll n$. Before transmission, the source node creates m augmented vectors $\{\mathbf{v}_i\}_{i=1}^{m}$ such that the last m positions of each $\mathbf{v}_i$ can store the coding coefficients generated by any node, namely,

$$\mathbf{v}_i = \left(\bar{\mathbf{v}}_i, \overbrace{\underbrace{0, \cdots, 0, 1}_{i}, 0, \cdots, 0}^{m}\right) \in \mathbb{Z}_p^{n+m}.$$

  These augmented vectors $\mathbf{v}_1, \cdots, \mathbf{v}_m$ form the basis of a subspace $V \subset \mathbb{Z}_p^{n+m}$, i.e., $V = span\{\mathbf{v}_1, \cdots, \mathbf{v}_m\}$. Finally, the source node runs the *Sign* algorithm to compute the corresponding signatures $\sigma_1, \cdots, \sigma_m$, and sends each vector/signature pair $(\mathbf{v}_i, \sigma_i)$ $(1 \leq i \leq m)$ (i.e., data packet) multiple times to the intermediate nodes.
- The intermediate node receives $(\mathbf{v}_{i_1}, \sigma_{i_1}), \cdots, (\mathbf{v}_{i_l}, \sigma_{i_l})$ on its l incoming edges, checks the validity of all vector/signature pairs, and discards corrupted vector/signature pairs by the *Verify* algorithm. Then the intermediate node generates random coefficients $\{c_i\}_{i=1}^{l}$ and executes the *Combine* algorithm to generate a new pair $(\mathbf{v}', \sigma')$ and transmits it on the outgoing edges.

- The destination node (i.e., receiver) checks the received vector/signature pairs and discards corrupted vector/signature pairs by the *Verify* algorithm. To recover the original file, the destination node needs to collect m linearly independent and valid vectors $\mathbf{v}'_1, \cdots, \mathbf{v}'_m$. For $i \in [1, m]$, $\mathbf{L}_i$ denotes the left-most n positions of $\mathbf{v}'_i$, and $\mathbf{R}_i$ denotes the right-most m positions of $\mathbf{v}'_i$. Finally, the original file can be recovered by computing

$$\begin{pmatrix} \bar{\mathbf{v}}_1 \\ \vdots \\ \bar{\mathbf{v}}_m \end{pmatrix} = \begin{pmatrix} \mathbf{R}_1 \\ \vdots \\ \mathbf{R}_m \end{pmatrix}^{-1} \cdot \begin{pmatrix} \mathbf{L}_1 \\ \vdots \\ \mathbf{L}_m \end{pmatrix}.$$

D. Security Model

For a CHNCS scheme, we consider two types of adversaries $\mathcal{A}_I$ and $\mathcal{A}_{II}$ as follows.

- Type-I adversary $\mathcal{A}_I$ can replace the public key PK_{ID} of the victim ID, but it is not given the master key msk or the partial private key d_{ID} of the victim.
- Type-II adversary $\mathcal{A}_{II}$ is a malicious-but-passive KGC proposed by Au et al. [14], who may not generate the master key and public parameters honestly. However, it cannot replace the user's public key or obtain the user's secret value.

If it is difficult to generate a signature of any vector $\mathbf{v}^* \notin V = span\{\mathbf{v}_1, \cdots, \mathbf{v}_m\}$ and to generate a signature of any new file identifier id^*, the CHNCS scheme is secure (i.e., unforgeable against adaptive chosen identity-and-subspace attacks). In the security model, we adopt two games **Game-**I and **Game-**II proposed by Chang et al. [9] to specify the formal security of the CHNCS scheme, except that **Setup-**II of **Game-**II is executed by the adversary.

III. Pollution Attacks on the Network Coding Signature Schemes

A. A Type-I Adversary Attack on Wu et al.'s Scheme

We first briefly review Wu et al.'s CHNCS scheme [10].

Setup: Given the security parameter λ, and two positive integers N, m, the KGC performs the following steps.

- Generate a bilinear group tuple $(p, \mathbb{G}_1, \mathbb{G}_2, g, e)$, where $\mathbb{G}_1, \mathbb{G}_2$ are the groups of the same prime order p, g is a generator of $\mathbb{G}_1$, and $e : \mathbb{G}_1 \times \mathbb{G}_1 \to \mathbb{G}_2$.
- Choose a random number $s \in \mathbb{Z}_p^*$ as the master key msk, and compute $P_{pub} = g^s$.
- Select four different cryptographic hash functions H, H_1, H_2, and H_3, each of which maps $\{0,1\}^*$ to $\mathbb{G}_1$.

Finally, the KGC outputs the public parameters $params = (p, \mathbb{G}_1, \mathbb{G}_2, g, e, H, H_1, H_2, H_3)$ and keeps $msk = s$ in secret.

Partial-Private-Key-Extract: Given $params$, the master key msk, and a user identity ID, the KGC computes $Q_{ID} = H(ID)$ and outputs the partial private key $D_{ID} = (Q_{ID})^s$.

Set-Secret-Value: Given $params$ and an identity ID, the user ID randomly chooses $x_{ID} \in \mathbb{Z}_p^*$ as its secret value.

Set-Private-Key: Given $params$, the partial private key D_{ID}, and the secret value x_{ID}, the user ID outputs the full private key $SK_{ID} = (D_{ID}, x_{ID})$.

Set-Public-Key: Given $params$ and the secret value x_{ID}, the user ID outputs the public key $PK_{ID} = g^{x_{ID}}$.

Sign: Given $params$, an identity ID, the private key SK_{ID}, an identifier $\tau \in \{0,1\}^\lambda$, and a vector $\mathbf{v} = (v_1, v_2, \cdots, v_N) \in \mathbb{Z}_p^N$, the user ID does as follows.

- Maintain a list L to record the identifier τ and its related information. If τ does not appear in L, randomly choose $r \in \mathbb{Z}_p^*$, compute $U = g^r$, and add (τ, r, U) to L. Otherwise, retrieve (τ, r, U) from L.
- Compute $T_i = H_1(ID, P_{pub}, \tau, i)$,

$$T'_i = H_2(ID, PK_{ID}, \tau, i),\ T = H_3(ID, PK_{ID}),$$

$$W = (D_{ID})^{\sum_{i=1}^{N} v_i} (\prod_{i=1}^{N} T_i^{v_i})^r (\prod_{i=1}^{N} (T'_i)^{v_i} \cdot T^{\sum_{i=1}^{N} v_i})^{x_{ID}}.$$

Finally, the user ID outputs the signature $\sigma = (U, W)$.

Combine: Given $params$, an identity ID, the public key PK_{ID}, an identifier τ, and the tuples $(c_1, \mathbf{v}_1, \sigma_1), \cdots, (c_l, \mathbf{v}_l, \sigma_l)$, where $c_i \in \mathbb{Z}_p^*$ and $\sigma_i = (U, W_i)$, this algorithm computes $\mathbf{v} = \sum_{i=1}^{l} c_i \mathbf{v}_i$, $W = \prod_{i=1}^{l} (W_i)^{c_i}$, and outputs a vector/signature pair $(\mathbf{v}, \sigma)$, where $\sigma = (U, W)$.

Verify: Given an identity ID, the public key PK_{ID}, an identifier τ, a vector $\mathbf{v}$, and the corresponding signature $\sigma = (U, W)$, this algorithm checks the equation

$$e(W, g) = e(Q_{ID}^{\sum_{i=1}^{N} v_i}, P_{pub}) \cdot e(\prod_{i=1}^{N} T_i^{v_i}, U) \cdot e(\prod_{i=1}^{N} (T'_i)^{v_i} \cdot T^{\sum_{i=1}^{N} v_i}, PK_{ID}).$$

If it holds, outputs 1 (accept). Otherwise, outputs 0 (reject).

The Type-I adversary $\mathcal{A}_I$ can generate a valid signature on a corrupted vector $\mathbf{v}^*$ as follows.

- Pick a vector $\mathbf{v}^* = (v_1^*, \cdots, v_N^*)$ such that $\sum_{i=1}^{N} v_i^* = 0$, an identifier $\tau^* \in \{0,1\}^\lambda$, $r^* \in \mathbb{Z}_p^*$, and a random secret value $x^* \in \mathbb{Z}_p^*$. Set $PK_{ID^*} = g^{x^*}$ as the public key of any user ID^*. If the identity ID^* exists, replace the original public key with PK_{ID^*}.
- Compute $\tilde{T}_i = H_1(ID^*, P_{pub}, \tau^*, i)$,

$$\tilde{T}'_i = H_2(ID^*, PK_{ID^*}, \tau^*, i),\ U^* = g^{r^*},$$

$$W^* = (\prod_{i=1}^{N} \tilde{T}_i^{v_i^*})^{r^*} (\prod_{i=1}^{N} (\tilde{T}'_i)^{v_i^*})^{x^*},$$

and set $\sigma^* = (U^*, W^*)$.

Obviously, as $\sum_{i=1}^{N} v_i^* = 0$, we have

$$W^* = (D_{ID^*})^{\sum_{i=1}^{N} v_i^*} (\prod_{i=1}^{N} \tilde{T}_i^{v_i^*})^{r^*} (\prod_{i=1}^{N} (\tilde{T}'_i)^{v_i^*} \cdot (T^*)^{\sum_{i=1}^{N} v_i^*})^{x^*},$$

where $D_{ID^*} = H(ID^*)^s$ and $T^* = H_3(ID^*, PK_{ID^*})$. Thus, $\sigma^* = (U^*, W^*)$ is a valid signature on the vector $\mathbf{v}^*$ for the user ID^* with the corresponding public key PK_{ID^*}. Therefore, Wu et al.'s scheme [10] is not secure as $\mathcal{A}_I$ can forge a valid signature on the desired vector. Similarly, Bian et al.'s CHNCS scheme [11] is also vulnerable to this Type-I adversary's forgery attack.

B. A Type-II Adversary Attack on Chang et al.'s Scheme

We first briefly review Chang et al.'s CHNCS scheme [9].

Setup: Given the security parameter λ, the KGC generates a bilinear group tuple $(p, \mathbb{G}_1, \mathbb{G}_2, g, e)$, where $\mathbb{G}_1, \mathbb{G}_2$ are groups of prime order p, g is a generator of $\mathbb{G}_1$, and $e : \mathbb{G}_1 \times \mathbb{G}_1 \to \mathbb{G}_2$. Then the KGC chooses $x \in \mathbb{Z}_p^*$ and sets $h = g^x$. H_1 and H_2 are two hash functions from $\{0,1\}^*$ to $\mathbb{G}_1$. Finally, the KGC outputs the public parameter $params = (p, \mathbb{G}_1, \mathbb{G}_2, g, e, h, H_1, H_2)$ and keeps the master key $msk = x$ in secret.

Partial-Private-Key-Extract: Given the master key msk and a user identity ID, the KGC outputs a partial private key $PP_{ID} = H_1(ID)^x$.

Set-Secret-Value: Given an identity ID, the user ID randomly chooses $y \in \mathbb{Z}_p^*$ and outputs a secret value $s_{ID} = y$.

Set-Private-Key: Given the partial private key PP_{ID} and the secret value s_{ID}, the user ID computes the full private key $SK_{ID} = PP_{ID}^{s_{ID}} = H_1(ID)^{xy}$ and outputs it.

Set-Public-Key: Given the secret value s_{ID}, the user ID computes the public key $PK_{ID} = h^{s_{ID}} = h^y$ and outputs it.

Sign: Given an identity ID, the public key PK_{ID}, the private key SK_{ID}, an identifier id, and a vector $\mathbf{v} = (v_1, v_2, \cdots, v_N) \in \mathbb{Z}_p^N$, the user ID does as follows.

- Maintain a list L to record the identifier id and its related information. If id does not appear in L, randomly choose $r \in \mathbb{Z}_p^*$, compute $w = g^r$, and add (id, r, w, σ_1) to L, where σ_1 is a signature of (id, w) generated by another certificateless signature scheme, which is not specifically constructed in Chang et al.'s scheme [9]. Otherwise, retrieve (r, w, σ_1) from L. Note that as we focus on the security analysis of the CHNCS scheme, we suppose this certificateless signature scheme used here is secure.
- Randomly choose $s \in \mathbb{Z}_p^*$ and compute

$$\sigma_2 = (SK_{ID})^{\sum_{j=1}^{N} v_j} \times (H_1(ID)^s \cdot \sum_{j=1}^{N} H_2(id, ID, PK_{ID}, j)^{v_j})^r.$$

Finally, the user outputs $Q = (w, \sigma_1, \sigma_2, s)$ as the signature.

Combine: Given an identity ID, the public key PK_{ID}, an identifier id, and a set of tuples $\{c_i, \mathbf{v}_i, Q_i\}_{i=1}^{l}$, where $Q_i = (w^{(i)}, \sigma_1^{(i)}, \sigma_2^{(i)}, s^{(i)})$, $w^{(1)} = \cdots = w^{(l)}$, and $\sigma_1^{(1)} = \cdots = \sigma_1^{(l)}$, this algorithm computes $\sigma_2 = \prod_{i=1}^{l} (\sigma_2^{(i)})^{c_i}$, $s = \sum_{i=1}^{l} c_i s^{(i)}$ and outputs a vector/signature pair $(\mathbf{v}, Q)$, where $Q = (w^{(1)}, \sigma_1^{(1)}, \sigma_2, s)$.

Verify: Given an identity ID, the public key PK_{ID}, an identifier id, a vector $\mathbf{v}$, and the signature $Q = (w, \sigma_1, \sigma_2, s)$, this algorithm checks the equation

$$e(\sigma_2, g) = e(H_1(ID), PK_{ID})^{\sum_{j=1}^{N} v_j} \cdot e(H_1(ID)^s \cdot \prod_{j=1}^{N} H_2(id, ID, PK_{ID}, j)^{v_j}, w).$$

If it holds, outputs 1 (accept). Otherwise, outputs 0 (reject).

The malicious-but-passive KGC, who generates the master key and public parameters maliciously, can obtain the private key of a selected victim user as follows.

- Suppose the identity of a victim user is ID^* (not necessarily to be random), the KGC randomly chooses $\alpha \in \mathbb{Z}_p^*$ and computes $g = H_1(ID^*)^\alpha$. The rest follows the original **Setup**, **Partial-Private-Key-Extract**, and **Set-Secret-Value** algorithms.
- Suppose the public key of the user ID^* is $PK_{ID^*} = h^y = H_1(ID^*)^{xy}$. From the PK_{ID^*}, the KGC can obtain the private key of the user ID^* by computing $SK_{ID^*} = PK_{ID^*}^{\frac{1}{\alpha}} = h^{\frac{y}{\alpha}} = H_1(ID^*)^{xy}$.

From above, it can be found that the manipulation of g can help disclose the victim user's private key, and thus Chang et al.'s CHNCS scheme is not secure against the malicious-but-passive KGC.

C. Universal Attack to Li et al.'s Scheme

We briefly describe Li et al.'s CHNCS scheme [12].

Setup: Given the security parameter λ, the KGC generates a random $s \in \mathbb{Z}_p^*$ as the master key and a bilinear group tuple $(p, \mathbb{G}_1, \mathbb{G}_2, g, e)$, where $\mathbb{G}_1, \mathbb{G}_2$ are groups of prime order p, g is a generator of $\mathbb{G}_2$. Then the KGC chooses three secure hash functions H_1, H_2, H_3, each of which maps $\{0,1\}^*$ to $\mathbb{Z}_p^*$, computes $P_{pub} = g^s, \omega = e(g, g)$, outputs the public parameters $params = (p, \mathbb{G}_1, \mathbb{G}_2, g, e, P_{pub}, \omega, H_1, H_2, H_3)$ and keeps $msk = s$ in secret.

Partial-Private-Key-Extract: Given $(params, msk, ID)$, the KGC randomly selects $r_{ID} \in \mathbb{Z}_p^*$, computes $R_{ID} = g^{r_{ID}}$, $z_{ID} = r_{ID} + sH_1(ID, P_{pub}, R_{ID})$, and outputs the partial private key $D_{ID} = (z_{ID}, R_{ID})$.

Set-Secret-Value: Given $(params, ID)$, the user randomly selects $x_{ID} \in \mathbb{Z}_p^*$ as its secret value.

Set-Private-Key: Given $(params, D_{ID}, x_{ID})$, the user sets $SK_{ID} = (z_{ID}, x_{ID})$ as the full private key.

Set-Public-Key: Given $(params, D_{ID}, x_{ID})$, the user computes $U_{ID} = g^{x_{ID}}$ and sets $PK_{ID} = (U_{ID}, R_{ID})$ as the public key.

Sign: Given $(params, ID, SK_{ID}, \mathbf{v}_i)$, the user generates an identifier vid, randomly selects $x \in \mathbb{Z}_p^*$, and computes $Y = \omega^x$, $b = H_2(ID, vid, Y)$. Then the user computes the tag $\tau = (\tau_0, Y)$ of the subspace V, where $\tau_0 = g^{(z_{ID} + bx_{ID})^{-1}}$. Finally, the user outputs a signature $\sigma_i = (\sigma_{i1}, \sigma_{i2})$ of the vector $\mathbf{v}_i$, where $\sigma_{i1} = \omega^{\frac{x \cdot x_{ID}}{x + x_{ID}} (\sum_{j=1}^{n} v_{i,j} H_3(ID, j, \tau_0))}$ and $\sigma_{i2} = \omega^{\frac{x^2}{x + x_{ID}} (\sum_{j=n+1}^{n+m} v_{i,j} H_3(ID, j, \tau_0))}$.

Combine: Given $(params, ID, PK_{ID}, \tau, \{(c_i, \mathbf{v}_i, \sigma_i)\}_{i=1}^{l})$ with the same identifier vid, this algorithm computes $\mathbf{v} = \sum_{i=1}^{l} c_i \mathbf{v}_i$, $\sigma_1 = \prod_{i=1}^{l} \sigma_{i1}^{c_i}$, $\sigma_2 = \prod_{i=1}^{l} \sigma_{i2}^{c_i}$, and outputs the combined vector $\mathbf{v}$ and the corresponding signature $\sigma = (\sigma_1, \sigma_2)$.

Verify: Given $(params, ID, PK_{ID}, vid, \tau, \mathbf{v}_i, \sigma_i)$, where $\tau = (\tau_0, Y)$ and $\sigma_i = (\sigma_{i1}, \sigma_{i2})$, this algorithm checks two equations $e(\tau_0, R_{ID} \cdot P_{pub}^{H_1(ID, P_{pub}, R_{ID})} \cdot U_{ID}^{H_2(ID, vid, Y)}) = \omega$ and $\sigma_{i1}^{\frac{1}{\sum_{j=1}^{n} v_{i,j} H_3(ID, j, \tau_0)}} \cdot \sigma_{i2}^{\frac{1}{\sum_{j=n+1}^{n+m} v_{i,j} H_3(ID, j, \tau_0)}} = Y$. If both

equations hold, outputs 1 (accept). Otherwise, outputs 0 (reject).

In the following, we show that Li et al.'s scheme [12] is not secure against pollution attacks, as the signature can be forged by any adversary $\mathcal{A}$:

- Capture a valid file $F = (ID, PK_{ID}, vid, \tau, \{\mathbf{v}_i, \sigma_i\}_{i=1}^m)$ by eavesdropping on the public channel. Let $V = span\{\mathbf{v}_1, \cdots, \mathbf{v}_m\}$.
- Choose a corrupted vector $\mathbf{v}^* \notin V$. Without loss of generality, $(\mathbf{v}^*, \sigma^*)$ can be forged from a valid $(\mathbf{v}_i, \sigma_i)$ in F. The forged signature $\sigma^* = (\sigma_1^*, \sigma_2^*)$ is computed as

$$\sigma_1^* = \sigma_{i1}^{\frac{\sum_{j=1}^{n} v_j^* H_3(ID,j,\tau_0)}{\sum_{j=1}^{n} v_{i,j} H_3(ID,j,\tau_0)}}, \sigma_2^* = \sigma_{i2}^{\frac{\sum_{j=n+1}^{n+m} v_j^* H_3(ID,j,\tau_0)}{\sum_{j=n+1}^{n+m} v_{i,j} H_3(ID,j,\tau_0)}}.$$

It can be verified that the forged signature σ^* is valid.

$\mathcal{A}$ can launch pollution attack by injecting the forged tuple $(ID, PK_{ID}, vid, \tau, \mathbf{v}^*, \sigma^*)$ into the network, making the recipients fail to recover the original file. Similarly, Yu et al.'s CHNCS scheme [13] is also vulnerable to the universal pollution attack.

IV. A New Scheme and Its Security Analysis

A. Our New CHNCS Scheme

In this section, we propose a new CHNCS scheme, which is composed of the following algorithms.

Setup: On input the security parameter λ, and two positive integers N, m, the KGC performs the following steps.

- Generate two groups $\mathbb{G}_1, \mathbb{G}_2$ of the same prime order p and a bilinear map $e : \mathbb{G}_1 \times \mathbb{G}_1 \to \mathbb{G}_2$.
- Choose a generator $g \in \mathbb{G}_1$, select a random number $s \in \mathbb{Z}_p^*$ as the master key msk, and compute $P_{pub} = g^s$.
- Select four different cryptographic hash functions $H_1, H_2 : \{0,1\}^* \to \mathbb{Z}_p^*$, and $H_3, H_4 : \{0,1\}^* \to \mathbb{G}_1$.
- For all $i \in [1, n]$, compute $g_i = H_4(P_{pub}, g, i) \in \mathbb{G}_1$.

Finally, the KGC outputs the public parameters $params = (\mathbb{G}_1, \mathbb{G}_2, e, p, g, P_{pub}, g_1, \cdots, g_n, H_1, H_2, H_3, H_4)$ and keeps $msk = s$ in secret.

Partial-Private-Key-Extract: On input $params$, the master key msk, and a user identity ID, the KGC chooses a random $y_{ID} \in \mathbb{Z}_p^*$, computes $Y_{ID} = g^{y_{ID}}$, $k_{ID} = y_{ID} + H_1(ID, Y_{ID}) \cdot s$, and outputs a partial private key $d_{ID} = (Y_{ID}, k_{ID})$.

Set-Secret-Value: On input $params$ and an identity ID, the source node ID randomly chooses $x_{ID} \in \mathbb{Z}_p^*$ and outputs it as a secret value.

Set-Private-Key: On input $params$, the partial private key d_{ID}, and the secret value x_{ID}, the source node ID computes $X_{ID} = g^{x_{ID}}$ and outputs the full private key $SK_{ID} = x_{ID} H_2(Y_{ID}, X_{ID}, P_{pub}) + k_{ID}$.

Set-Public-Key: On input $params$ and the secret value x_{ID}, the source node ID computes $X_{ID} = g^{x_{ID}}$ and outputs the full public key $PK_{ID} = (Y_{ID}, X_{ID})$.

Sign: On input $params$, an identity ID, the private key $SK_{ID} = x_{ID} H_2(Y_{ID}, X_{ID}, P_{pub}) + k_{ID}$, a random identifier $id \in \{0,1\}^\lambda$, and a vector $\mathbf{v} = (v_1, v_2, \cdots, v_N) \in \mathbb{Z}_p^N$, the source node ID computes

$$\sigma = (\prod_{i=1}^{n} g_i^{v_i} \prod_{j=1}^{m} H_3(id, u)^{v_{n+j}})^{x_{ID} H_2(Y_{ID}, X_{ID}, P_{pub}) + k_{ID}}$$

and outputs it as a signature of the vector $\mathbf{v}$.

Combine: On input $params$, an identity ID, the public key PK_{ID}, an identifier id, and a set of tuples $\{(c_i, \mathbf{v}_i, \sigma_i)\}_{i=1}^l$ (l>0), where $c_i \in \mathbb{Z}_p$, the source nodes or the intermediate nodes output a new vector/signature pair $(\hat{\mathbf{v}}, \hat{\sigma})$, where $\hat{\mathbf{v}} = \sum_{i=1}^l c_i \mathbf{v}_i$ and $\hat{\sigma} = \prod_{i=1}^l \sigma_i^{c_i}$.

Verify: On input an identity ID, the public key PK_{ID}, an identifier id, a vector $\mathbf{v}$, and the corresponding signature σ, the intermediate nodes or the destination nodes compute $h_1 = H_1(ID, Y_{ID}), h_2 = H_2(Y_{ID}, X_{ID}, P_{pub})$ and check the equation

$$e(\sigma, g) = e(\prod_{i=1}^{n} g_i^{v_i} \cdot \prod_{j=1}^{m} H_3(id, j)^{v_{n+j}}, X_{ID}^{h_2} \cdot Y_{ID} \cdot P_{pub}^{h_1}).$$

If the above equation holds, then outputs 1 (accept). Otherwise, outputs 0 (reject).

The correctness of this scheme can be easily verified.

B. Security Analysis

Theorem 1. *Our CHNCS scheme is unforgeable against adaptive chosen identity-and-subspace attacks in the random oracle model, assuming that solving the CDH problem in $\mathbb{G}_1$ is infeasible.*

Due to space limitation, the proof of Theorem 1 is provided at https://github.com/0racle2/IEEE-supplementary-material/.

TABLE I
Comparisons of communication costs.

Scheme	Private key / Partial private key	Signature
ID-LHS scheme [7]	$\|\mathbb{Z}_p\|+2\|\mathbb{G}_1\|$	$2\|\mathbb{Z}_p\|+4\|\mathbb{G}_1\|$
Our scheme	$\|\mathbb{Z}_p\|+\|\mathbb{G}_1\|$	$\|\mathbb{G}_1\|$

V. Performance Analysis

To the best of our knowledge, no CHNCS scheme for network coding is secure against attacks defined in Section II-D. In addition, identity-based linearly homomorphic signature (ID-LHS) also avoids cumbersome certificate management to reduce network overhead. Therefore, we compare our CHNCS scheme with a classical ID-LHS scheme [7] in terms of communication and computational costs. Note that Galindo and Garcia's identity-based signature scheme [15] is applied in the ID-LHS scheme [7].

Communication Costs: Since both the partial private key in our scheme and the private key in the ID-LHS scheme [7] are sent from the KGC to each source node, and the signature of each vector is transmitted between nodes in networks, we consider the size of the key sent from the KGC to each source node and the size of the signature. Let $|\mathbb{Z}_p|$ and $|\mathbb{G}_1|$ represent the size of elements in $\mathbb{Z}_p$ and $\mathbb{G}_1$ respectively. The

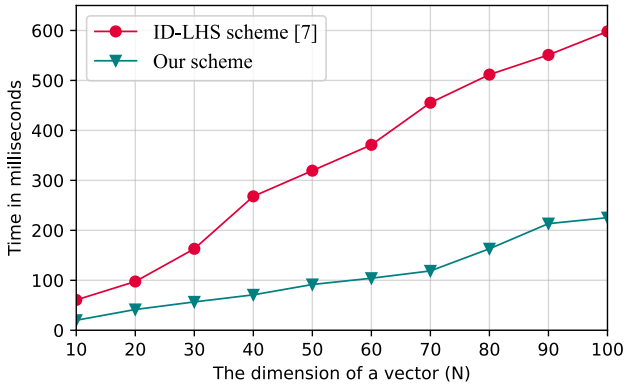

Fig. 2. Computational costs for signing different dimensional vectors.

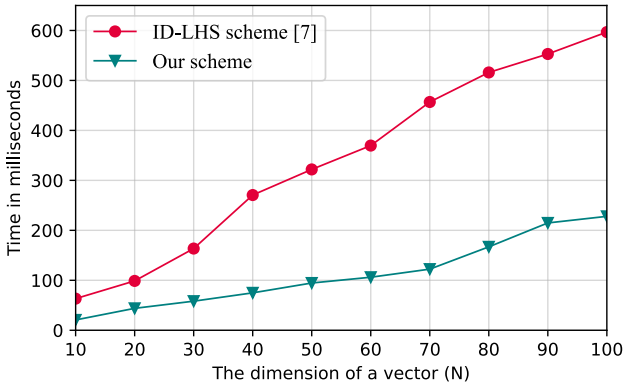

Fig. 3. Computational costs for verifying the signatures of different dimensional vectors.

comparisons of communication costs are shown in Table I, showing that our scheme has lower communication costs than the ID-LHS scheme [7].

Computational Costs: Simulation experiments are conducted on the Ubuntu 22.04.1 (4 GB memory, 4 processor cores) VMware 17.0.0 in a laptop running Windows 11 with 12-th Gen Intel(R) Core(TM) i5-12500H @ 2.50 GHz and 16.0 GB RAM. The codes are written using C programming language with GNU Multiple Precision Arithmetic library (GMP-6.2.1) and Pairing-Based Cryptography (PBC) library (PBC-0.5.14). For the security level of 80-bit, we choose the standard parameter settings $a.param$ of Type A curve in PBC library. Since the augmented vector introduces $\Theta(m^2)$ communication costs for each file, we typically choose $m \ll n$ in the simulation, specifically, $m = \frac{N}{10}$.

For the *Sign* and *Verify* algorithms, we set the dimension N of a vector increases from 10 to 100. The simulation results are obtained by repeating the experiments 1000 times. As shown in Fig. 2 and Fig. 3, when $N = 100$ (i.e., $n = 90$ and $m = 10$), the ID-LHS scheme [7] takes 597.969 milliseconds for signing a vector and 596.785 milliseconds for verifying the signature of a vector, but our scheme only requires 225.338 and 227.999 milliseconds correspondingly, reducing by 62.316% and 61.795%. Therefore, our scheme has lower computation costs than the ID-LHS scheme [7].

VI. Conclusion

In this paper, we have shown that a Type-I adversary can replace any user's public key to generate a valid signature on the corrupted data packet in Wu et al.'s CHNCS scheme [10], a Type-II adversary can obtain the private key of a selected victim in Chang et al.'s CHNCS scheme [9], and any adversary can forge a valid signature on the corrupted data packet in Li et al.'s CHNCS scheme [12]. To overcome the vulnerabilities in these schemes, we propose a novel CHNCS scheme, which is proved to be unforgeable against adaptive chosen identity-and-subspace attacks under two types of adversaries in the ROM. Thus, our CHNCS scheme can resist pollution attacks. Furthermore, the theoretical analysis and experiment results show that our scheme is more efficient for network coding.

References

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

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

[3] J. Widmer and J.-Y. Le Boudec, "Network coding for efficient communication in extreme networks," in *Proceedings of the 2005 ACM SIGCOMM workshop on Delay-tolerant networking*, 2005, pp. 284–291.

[4] Y. Wu, P. A. Chou, and S.-Y. Kung, "Minimum-energy multicast in mobile ad hoc networks using network coding," *IEEE Transactions on communications*, vol. 53, no. 11, pp. 1906–1918, 2005.

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

[6] Y. Li, F. Zhang, and X. Liu, "Secure data delivery with identity-based linearly homomorphic network coding signature scheme in iot," *IEEE Transactions on Services Computing*, vol. 15, no. 4, pp. 2202–2212, 2020.

[7] Q. Lin, H. Yan, Z. Huang, W. Chen, J. Shen, and Y. Tang, "An id-based linearly homomorphic signature scheme and its application in blockchain," *IEEE Access*, vol. 6, pp. 20 632–20 640, 2018.

[8] X. Zhou, T. Zhou, Y. Tian, W. Zhong, and X. Yang, "Linearly homomorphic signature scheme with high signature efficiency and its application in iot," *IEEE Internet of Things Journal*, 2024.

[9] J. Chang, Y. Ji, B. Shao, M. Xu, and R. Xue, "Certificateless homomorphic signature scheme for network coding," *IEEE/ACM Transactions on Networking*, vol. 28, no. 6, pp. 2615–2628, 2020.

[10] B. Wu, C. Wang, and H. Yao, "A certificateless linearly homomorphic signature scheme for network coding and its application in the iot," *Peer-to-Peer Networking and Applications*, vol. 14, pp. 852–872, 2021.

[11] G. Bian, M. Song, and B. Shao, "Certificateless network coding scheme from certificateless public auditing protocol," *The Journal of Supercomputing*, vol. 79, no. 3, pp. 2570–2602, 2023.

[12] Y. Li, F. Zhang, and Y. Sun, "Lightweight certificateless linearly homomorphic network coding signature scheme for electronic health system," *IET Information Security*, vol. 15, no. 1, pp. 131–146, 2021.

[13] H. Yu and J. Shi, "Certificateless homomorphism network coding signature scheme," *IEEE Sensors Journal*, vol. 22, no. 13, pp. 13 707–13 715, 2022.

[14] M. H. Au, Y. Mu, J. Chen, D. S. Wong, J. K. Liu, and G. Yang, "Malicious kgc attacks in certificateless cryptography," in *Proceedings of the 2nd ACM symposium on Information, computer and communications security*, 2007, pp. 302–311.

[15] D. Galindo and F. D. Garcia, "A schnorr-like lightweight identity-based signature scheme," in *Progress in Cryptology–AFRICACRYPT 2009: Second International Conference on Cryptology in Africa, Gammarth, Tunisia, June 21-25, 2009. Proceedings 2*. Springer, 2009, pp. 135–148.