Hindawi
Security and Communication Networks
Volume 2021, Article ID 1609873, 10 pages
https://doi.org/10.1155/2021/1609873

Research Article

Certificateless Multisignature Scheme Suitable for Network Coding

Huifang Yu,[1,2] Zhewei Qi,[1] Danqing Liu,[3] and Ke Yang[1]

[1] *School of Cyberspace Security, Xi'an University of Posts & Telecommunications, Xi'an 710121, China*
[2] *School of Information Engineering, Qinghai Communications Technical College, Xining 810003, China*
[3] *School of Computer, Qinghai Normal University, Xining 810008, China*

Correspondence should be addressed to Huifang Yu; yuhuifang@qhnu.edu.cn

Received 7 August 2021; Revised 30 September 2021; Accepted 20 October 2021; Published 16 November 2021

Academic Editor: Weizhi Meng

Network coding can save the wireless network resources and improve the network throughput by combining the routing with coding. Traditional multisignature from certificateless cryptosystem is not suitable for the network coding environment. In this paper, we propose a certificateless multisignature scheme suitable for network coding (NC-CLMSS) by using the sequential multisignature and homomorphic hash function. NC-CLMSS is based on the CDH and ECDL problems, and its security is detailedly proved in the random oracle (RO) model. In NC-CLMSS, the source node generates a multisignature for the message, and the intermediate node linearly combines the receiving message. NC-CLMSS can resist the pollution and forgery attacks, and it has the fixed signature length and relatively high computation efficiency.

1. Introduction

As the network information interaction technology, the network coding [1] has routing and coding functions and allows the router to encode the received data. Network coding has the merits of high transmission efficiency, fast speed, strong robustness, and good stability, but it is vulnerable to the pollution attacks in the data transmission process. In recent years, the researchers have proposed a series of network-coding signature schemes [2–6] to solve the network coding contamination, where the schemes in [4, 5] effectively solved the replay attacks by using the time stamps; the certificateless network-coding homomorphism signature [6] is designed by using the homomorphic hash function; it can resist the replay attacks with forgery attacks at the same time and has lower computational overhead with the communication cost.

In real scenario, there are many applications to use the signature technology. With the development of communication technology, the scholars proposed many signature varieties (including multisignature) suitable for various application scenarios, such as medical field [7–10], privacy security [11], vehicle-mounted network [12, 13], multicast network [14, 15], e-government [16], e-commerce [17], and campus management facilities [18]. Multisignature first generates the partial signature of the same message, and then, the signature collector integrates the partial signatures into a signature. In terms of the order of partial signatures, multisignature can be divided into sequential multisignature [19] and broadcast multisignature [20, 21]. Compared with ordinary multisignature, the sequential multisignature has the following characteristics: (1) the signature length has nothing to do with the number of signatures; (2) instead of using the public key of each signer, the group public key can be used to verify the signature; (3) signers sign the messages in a concrete order, otherwise a valid multisignature cannot be obtained; (4) it is not computationally feasible to obtain the valid signatures without the joint operation of all signers. From now on, there is no sequential multisignature suitable for network coding, as described in Figure 1, so we will

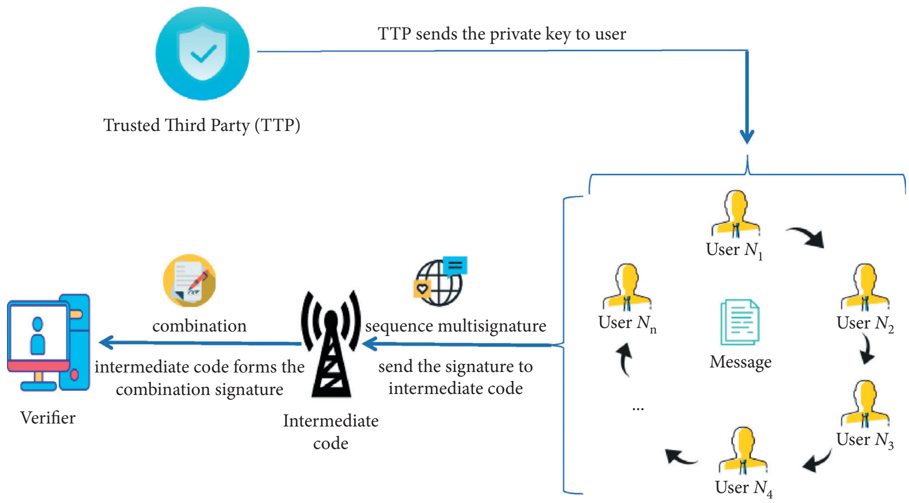

Figure 1: Model for sequential multisignature suitable for network coding.

devise such a scheme to resist the pollution and forgery attacks in wireless networks.

1.1. Contributions. For the above reasons, a new certificateless multisignature scheme for the network coding (NC-CLMSS) is devised by combining the certificateless public key with sequential multisignature. In NC-CLMSS, the users at the source node generate the sequential multisignatures for the messages in a fixed order and transfer the signed messages from the router to the intermediate node. Intermediate node performs the linear combination of received information. Meanwhile, the destination node can verify the correctness of the signature without knowing the signer private key. Destination node filters out the contaminated information and forwards the validated data to the next receiving node. NC-CLMSS overcomes the key escrow and certificate management issues; moreover, it can resist the forgery attacks with pollution attacks in the multisource network-coding environment and has relatively better transmission efficiency.

2. Preliminaries

2.1. Bilinear Pairing. Assume G_1 and G_2 are additive and multiplication cyclic groups with the prime order q, respectively. P is a generator of the cyclic group G_1. e: $G_1 \times G_1 \longrightarrow G_2$ is an admissible bilinear pairing if e is a map with the following properties: $e(aP, bP) = e(P, P)^{ab}$, for any $a, b \in z_q^*$, and $P \in G_1$; $e(P, P) \neq 1$; there exists an efficient algorithm to compute $e(P, Q)$, for any $P, Q \in G_1$.

Definition 1. (ECDL problem). Given $(P, aP) \in G_1$, for any $a \in z_q^*$, the ECDL (elliptic curve discrete logarithm) problem is to calculate $a \in z_q^*$.

Definition 2. (CDH problem). Given $(P, aP, bP) \in G_1$, for any $a, b \in z_q^*$, the CDH (computational Diffie–Hellman) problem is to calculate $abP \in G_1$.

2.2. Multisource Network Model. Multisource network coding [22] has a set of source nodes. In the multisource model, each encoding message has a uniformly assigned two-dimensional index. Model for multisource transmission network is shown in Figure 2.

Multisource network coding is regarded as a directed acyclic graph $R = (E', V)$, where E' is the set of edges in the network and V is the set of all nodes. $U = \{u_1, u_2, \ldots, u_m\} \subset V$ is the set of the source nodes and $D = \{d_1, d_2, \ldots, d_k\} \subset V$ is the set of the sink nodes; m multicast messages are expressed by $v = (v_1, v_2, \ldots, v_m)$; the source nodes' set U sends $v = (v_1, v_2, \ldots, v_m)$ to the sink nodes D, where each message vector v_i is composed of n elements over finite field F, where v_i is written as

$$v_i = \left(v_{i,1}, v_{i,2}, \ldots, v_{i,n}\right) \in F, \quad 1 \le i \le m. \tag{1}$$

Let j be the unique index uniformly assigned to each message, and the same multicast message sent by different source nodes has the same index. Each packet $w = (w_1, w_2, \ldots, w_l)$ can be sent by arbitrary intermediate node in network, and w is the linear combination of l messages received by this node.

2.3. Symbol Descriptions. In Table 1, the readers can see the meaning of notations relevant to this article.

3. Formal Definition

3.1. Algorithm Definition. A NC-CLMSS is defined by six polynomial time algorithms as follows.

Setup: input a security parameter ρ and finally output the master key s with a system parameter set μ.

Extract: input μ with the user identity ID_i and finally output a pair (R_i, D_i) of partial public/private keys.

KeyGen: input μ with the user identity ID_i and finally output a pair (x_i, P_i) of public/private keys.

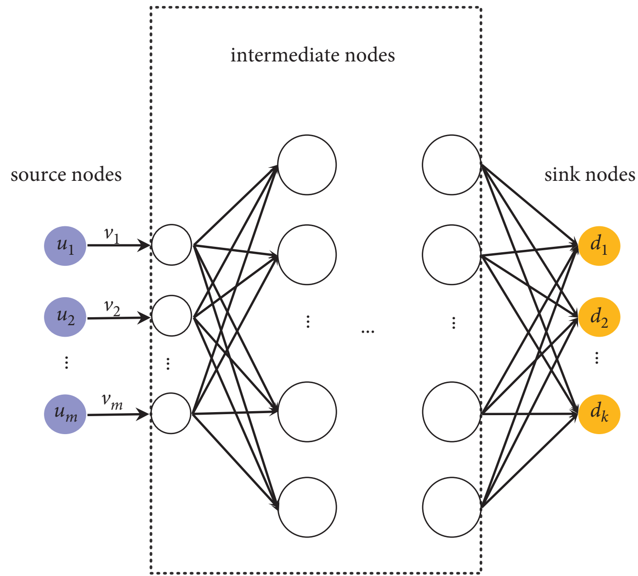

Figure 2: Model for multisource transmission network.

Table 1: Meaning of the various notations.

Notations	Meaning
KGC	The key generation center
ρ	A security parameter
η	The public parameter set
ID_i	Some user identity
N_i	User N_i $(i \in \{1, 2, \ldots, n\})$
S	The master secret key
s_i	The full private key of a user
PK_i	The full public key of a user
D_i	The partial private key of a user
R_i	The partial public key of a user
P_i	The user public key
P_{pub}	The system public key
H_i $(i = 0, 1, 2)$	Hash function $(i = 0, 1, 2)$
σ_i	A signature of the message v_t
σ	Multisignature

Multisignature: input μ, the master key s, the message v_t, the private key (D_i, x_i), and public key (R_i, P_i) and finally output a signature $\boldsymbol{\sigma}_i$.

Combination: input the message vector $w_1, \ldots, w_m$ and finally output a combined signature $\boldsymbol{\sigma}$.

Verification: input μ, $\boldsymbol{\sigma}_i$, and $\boldsymbol{\sigma}$, the public key (R_i, P_i), and the message v_t; the verifier outputs a result based on the verification case.

3.2. Security Model. A NC-CLMSS must meet the existential unforgeability against the adaptive chosen-message attacks (UF-CMA). For the UF-CMA security model of NC-CLMSS, we think about the game EXP1/EXP2 between a challenger C and a polynomial time adversary A_1 or A_2.

Firstly, $\mathcal{O}_C^{\text{setup}}(\rho) \xrightarrow{\mu} A_1$ and $\mathcal{O}_C^{\text{setup}}(\rho) \xrightarrow{\mu, s} A_2$, where A_1 is a malicious user who can change any user public key but cannot know the master private key; A_2 is a malicious KGC who knows the system master key but cannot change any user public key. After that, A_1 or A_2 carries out the adaptive queries as follows:

$$\mathcal{O}_C^{\text{partialprivatekey}}(ID_i) \xrightarrow{D_i} \frac{A_1}{A_2},$$

$$\mathcal{O}_C^{\text{publickey}}(ID_i) \xrightarrow{(R_i, P_i)} \frac{A_1}{A_2},$$

$$\mathcal{O}_C^{\text{privatekey}}(ID_i) \xrightarrow{x_i \text{ if } P_i \text{ was not replaced}} A_1,$$

$$\mathcal{O}_C^{\text{privatekey}}(ID_i) \xrightarrow{x_i} A_2,$$

$$\mathcal{O}_C^{\text{replacepublickey}}(ID_i) \xrightarrow{(R_i', P_i')} A_1,$$

$$\mathcal{O}_C^{\text{multisignature}}(v_t) \xrightarrow{\sigma_i} \frac{A_1}{A_2},$$

$$\mathcal{O}_C^{\text{combination}}(\boldsymbol{\sigma}_i) \xrightarrow{\sigma} \frac{A_1}{A_2},$$

$$\mathcal{O}_C^{\text{verification}}(\boldsymbol{\sigma}, \boldsymbol{\sigma}_i) \xrightarrow{\text{message or } \perp} \frac{A_1}{A_2}. \tag{2}$$

Finally, A_1/A_2 outputs a forged signature σ^*. In the adaptive queries, A_1 should not request the full private key of ID_s; A_2 cannot request the private key of ID_s. In addition, σ^* should not be returned by any multisignature oracle. A_1/A_2 wins in EXP1/EXP2 if $\mathcal{O}_C^{\text{verification}}(\sigma, \sigma_i) \longrightarrow^{\text{not} \perp} A_1/A_2$.

Assume Adv (ρ) denotes the adversary advantage in EXP1/EXP2; then, Adv (ρ) is defined as the probability which A_1/A_2 succeeds in EXP1/EXP2.

Definition 3. A NC-CLMSS is said to be UF-CMA secure if no polynomial time adversary A_1/A_2 succeeds in EXP1/EXP2 with a non-negligible advantage.

4. NC-CLMSS Instance

4.1. Setup. Given a security parameter ρ, KGC (key generation center) chooses cyclic groups G_1 and G_2 with the prime order q, as described in Section 2.1. P is a generator of G_1 and e: $G_1 \times G_1 \longrightarrow G_2$. KGC selects secure hash functions: H_0: $\{0,1\}^* \times G_1 \longrightarrow Z_q^*$, H_1: $\{0,1\}^* \longrightarrow Z_q^*$, H_2: $\{0,1\}^* \longrightarrow G_1$. KGC chooses a master key $s \in_R Z_q^*$ and maintains its secret and then calculates the system public key $P_{\text{pub}} = sP$. Finally, KGC publishes the system parameter set: $\mu = \{G_1, G_2, q, P, e, P_{\text{pub}}, H_0, H_1, H_2\}$.

4.2. Extract. Given the identity ID_i of the user N_i and μ, KGC randomly chooses $r_i \in Z_q^*$ and calculates $R_i = r_i P$, $h_i = H_0(ID_i, R_i)$, and $D_i = r_i + h_i s$, where $ID_i \in \{ID_1, ID_2, \ldots, ID_n\}$, D_i is the partial private key of N_i, and R_i is the partial public key of N_i.

4.3. KeyGen. Given the identity ID_i of the user N_i and μ, this user N_i ($i \in \{1, 2, \ldots, n\}$) randomly chooses a secret value $x_i \in Z_q^*$ and calculates the public key $P_i = x_i P$. Note that $ID_i \in \{ID_1, ID_2, \ldots, ID_n\}$, $PK_i = (P_i, R_i)$ is the full public key of N_i, and $s_i = (x_i, D_i)$ are the full private key of N_i.

4.4. Multisignature. Assume $L = \{ID_1, ID_2, \ldots, ID_n\}$ is an identity set of n users and $N_1 \longrightarrow N_2 \longrightarrow \ldots \longrightarrow N_n$ denotes the signature sequence of n users. In other words, the user N_i ($i \in \{1, 2, \ldots, n\}$) signs the message v_t with the sequence $N_1 \longrightarrow N_2 \longrightarrow \ldots \longrightarrow N_n$. Firstly, N_1 calculates

$$\begin{aligned} l_1 &= H_1(ID_1, L, P_1, R_1, v_t), \\ T &= H_2(v_t, L, P_{\text{pub}}), \\ \sigma_1 &= \text{SIGN}_1 \\ &= (l_1 x_1 + D_1)T, \end{aligned} \tag{3}$$

where $\boldsymbol{\sigma}_1$ is the partial signature of the message v_t from the user N_1. User N_1 delivers (v_t, σ_1) to the user N_2. After receiving (v_t, σ_{i-1}), the user N_i ($i = 2, 3, \ldots, n$) calculates

$$\begin{aligned} l_j &= H_1(ID_j, L, P_j, R_j, v_t), \\ h_j &= H_0(ID_j, R_j), \quad 1 \le j \le i-1, \\ T &= H_2(v_t, L, P_{\text{pub}}). \end{aligned} \tag{4}$$

If the equality $e(\sigma_{i-1}, P) = e(T, \sum_{j=1}^{i-1}(l_j P_j + R_j + h_j P_{\text{pub}}))$ holds, the user N_i calculates

$$\begin{aligned} l_i &= H_1(ID_i, L, P_i, R_i, v_t), \quad (1 \le i \le n), \\ \sigma_i &= \sigma_{i-1} + \text{SIGN}_i \\ &= \sigma_{i-1} + (l_i x_i + D_i)T. \end{aligned} \tag{5}$$

Then, the signature of the user N_n is $\sigma_n = \sum_{i=1}^n \text{SIGN}_i$. Finally, $\boldsymbol{\sigma} = (\sigma_1, \sigma_2, \ldots, \sigma_d)$ and v_t are sent to the intermediate code and sink node.

4.5. Combination. Given the local coding vector $\alpha = (\alpha_1, \ldots, \alpha_m)$ and global vector $\beta = (\beta_1, \ldots, \beta_m)$, the intermediate node combines the message vector as follows:

$$w = \sum_{i=1}^{m} \alpha_i w_i. \tag{6}$$

Then, the message vector v_t is also denoted as $w = \sum_{j=1}^m \beta_j v_j$, and the signature corresponding to the message vector w is $\boldsymbol{\sigma} = (\sigma_1, \sigma_2, \ldots, \sigma_d)$. Signature process corresponding to w is $\sigma_j = \sum_{i=1}^m \sigma_{i,j}^{\alpha_i}$, where $\sigma_{i,j}$ $(1 \le i \le m, 1 \le j \le l)$ represents the j-th element of $\boldsymbol{\sigma}_i$. Finally, the intermediate node outputs the combined signature $\sigma = \prod_{i=1}^m \boldsymbol{\sigma}_i^{\alpha_i}$.

4.6. Verify. After receiving the multisignature and combination signature, the verifier calculates $l_i = H_1(ID_i, L, P_i, R_i, v_t)$ $(1 \le i \le n)$ and $T = H_2(v_t, L, P_{\text{pub}})$.

If the equality $e(\sigma_n, P) = e(T, \sum_{i=1}^n (l_i P_i + R_i + h_i P_{\text{pub}}))$ holds, the multisignature is valid and invalid otherwise.

5. Correctness Analysis

5.1. Single Signature Verification. Given the signature $\boldsymbol{\sigma}_i$ of the message v_t, then the signature verification process of the user N_i ($i \in \{1, 2, \ldots, n\}$) is as follows:

$$\begin{aligned} e(\sigma_n, P) &= e\left(\sum_{i=1}^{n}(l_i x_i + D_i)T, P\right) \\ &= e\left(T, \sum_{i=1}^{n}(l_i x_i P + D_i P)\right) \\ &= e\left(T, \sum_{i=1}^{n}\left(l_i P_i + R_i + h_i P_{\text{pub}}\right)\right). \end{aligned} \tag{7}$$

5.2. Combination Verification. Given the message $(v_1, \ldots, v_m)$ and global coding vector $\beta = (\beta_1, \ldots, \beta_m)$, $w = \sum_{j=1}^{m} \beta_j v_j$ is the message vector received by the intermediate node, and $\boldsymbol{\sigma} = (\sigma_1, \sigma_2, \ldots, \sigma_t)$ is the multisignature corresponding to w. In the verification phase, it is necessary to check the correctness of the following equality:

$$e(\sigma_n, P) = e\left(T, \sum_{i=1}^{n}\left(l_i P_i + R_i + h_i P_{\text{pub}}\right)\right), \tag{8}$$

where $h_i = H_0\,(ID_i, R_i)$. In the multisource network coding, the intermediate nodes combine the messages from different source nodes and form a combination signature. Different source nodes may send the same message. In order to distinguish the possible combination of the same message vector, the global coding vector is expressed as $\beta_j = \sum_{k=1}^{d} \beta_j(u_k)$, where the global coding vector $\beta_j(u_k) \in \left\{\beta_j(u_1), \ldots, \beta_j(u_d)\right\}$ and source node user $u_k \in \{u_1, u_2, \ldots, u_d\}$. Then, the message vector is expressed as $w = \sum_{j=1}^{m}\sum_{k=1}^{d} \beta_j(u_k) v_j$. Hence, the multisignature of message vector w can be expressed as $\boldsymbol{\sigma} = \prod_{j=1}^{m}\prod_{k=1}^{d} (\sigma_j(u_k))^{\beta_j(u_k)}$, and then, the i-th component in the multisignature can be expressed as $\boldsymbol{\sigma}_i = \prod_{j=1}^{m}\prod_{k=1}^{d} (\sigma_{j,i}(u_k))^{\beta_j(u_k)}$, $(1 \le i \le n)$, where $\sigma_{j,i}(u_k)$ is the i-th component of multisignature $\sigma_j(u_k)$. Then, the relevant equality is verified as follows:

$$\begin{aligned} &e(\sigma_n, P) \\ &= e\left(\sum_{i=1}^{n}\left[\prod_{j=1}^{m}\prod_{k=1}^{d} l_i^{\boldsymbol{\beta}_j} \cdot x_i + D_i\right] \cdot \prod_{j=1}^{m}\prod_{k=1}^{d} T^{\boldsymbol{\beta}_j}, P\right) \\ &= e\left(\prod_{j=1}^{m}\prod_{k=1}^{d} T^{\boldsymbol{\beta}_j}, \sum_{i=1}^{n}\left[\prod_{j=1}^{m}\prod_{k=1}^{d} l_i^{\boldsymbol{\beta}_j} \cdot x_i + D_i\right] \cdot P\right) \\ &= e\left(\prod_{j=1}^{m}\prod_{k=1}^{d} T^{\boldsymbol{\beta}_j}, \sum_{i=1}^{n}\left[\prod_{j=1}^{m}\prod_{k=1}^{d} l_i^{\boldsymbol{\beta}_j} \cdot x_i \cdot P + D_i \cdot P\right]\right) \\ &= e\left(\prod_{j=1}^{m}\prod_{k=1}^{d} T^{\boldsymbol{\beta}_j}, \sum_{i=1}^{n}\left[\prod_{j=1}^{m}\prod_{k=1}^{d} l_i^{\boldsymbol{\beta}_j} \cdot P_i + r_i + h_i P_{\text{pub}}\right]\right). \end{aligned} \tag{9}$$

From the verification process of single message, we know $e(\sigma_n, P) = e(T, \sum_{i=1}^{n}(l_i P_i + R_i + h_i P_{\text{pub}}))$. Then, the verification process is denoted as

$$\begin{aligned} &e\left(\prod_{j=1}^{m}\prod_{k=1}^{d} T^{\beta_j}, \sum_{i=1}^{n}\left[\prod_{j=1}^{m}\prod_{k=1}^{d} l_i^{\boldsymbol{\beta}_j} \cdot P_i + R_i + h_i P_{\text{pub}}\right]\right) \\ &= e\left(T, \sum_{i=1}^{n}\left(l_i P_i + R_i + h_i P_{\text{pub}}\right)\right). \end{aligned} \tag{10}$$

6. Security Analysis

Theorem 1. *In the RO model, if the polynomial time adversary A_1 can break the UF-CMA-I security of NC-CLMSS, a challenge algorithm C can solve the CDH (computational Diffie–Hellman) problem.*

Proof. C receives a random instance $(P, aP, bP) \in G_1$ of CDH problem, and its aim is to use A_1 (the subroutine of C) to calculate $abP \in G_1$. C maintains the initially empty lists L_0, L_1, L_2, and L_3 to store the query-answer values of several oracles. Firstly, $\mathcal{O}_C^{\text{setup}}(\rho) \longrightarrow^{\mu} A_1$, where $P_{\text{pub}} = aP$. Then, A_1 adaptively issues the polynomial time queries as follows.

H_0 queries: A_1 issues an H_0 query. C outputs h_i to A_1 if the relevant tuple is in the list L_0; otherwise, C returns a random $h_i \in_R Z_q^*$ and stores (ID_i, R_i, h_i) in L_0.

H_1 queries: A_1 issues an H_1 query. C returns l_i if a matching tuple is in the list L_1; otherwise, C returns $l_i \in_R Z_q^*$ and stores $(ID_i, L, P_i, R_i, v_t, l_i)$ in L_1.

H_2 queries: A_1 issues an H_2 query. If it is not the θ-th query ($\theta \in \{1, 2, \ldots, q_0\}$ (q_0 is the query times relevant to the H_0 oracle) and a matching tuple is in the list L_2, C outputs $T = lP$ ($l \in_R Z_q^*$) and stores $(v_t, l_i, P_{\text{pub}}, l, T)$ in L_2; otherwise, C returns $T = bP$ and stores $(v_t, l_i, P_{\text{pub}}, -, T)$ in L_2.

Partial private key queries: A_1 requests a partial private key of ID_i. If it is not the θ-th query, C chooses $r_i \in_R Z_q^*$ to calculate $R_i = r_i P$ such that D_i satisfies $D_i P = R_i + h_i P_{\text{pub}}$ and finally returns D_i as the answer and stores $(ID_i, r_i, R_i, D_i, -, -)$ in the list L_3; otherwise, C fails and aborts the game.

Public key queries: A_1 requests a public key of ID_i. C calculates $P_i = x_i P$ ($x_i \in_R Z_q^*$) and finally returns $PK_i = (R_i, P_i)$ and updates the list L_3 with $(ID_i, r_i, R_i, D_i, x_i, P_i)$.

Secret value queries: A_1 requests a secret value of ID_i. C returns x_i from L_3 if the corresponding public key has not been replaced.

Public key replacement: if it is not the θ-th query, the public key of ID_i is replaced by A_1; otherwise, C fails and aborts the game.

Multisignature queries: for a multisignature query of message v_t, C runs the relevant algorithm and returns a result if it is not the θ-th query; otherwise, C signs v_t

with the sequence $N_1 \longrightarrow N_2 \longrightarrow \ldots \longrightarrow N_n$. Firstly, C calculates for N_1 as follows:

$$\begin{aligned} l_1 &= H_1(ID_1, L, P_1, R_1, v_t), \\ T &= H_2(v_t, L, P_{\text{pub}}), \\ \sigma_1 &= \text{SIGN}_1 \\ &= (l_1 x_1 + D_1)T, \end{aligned} \tag{11}$$

where $\boldsymbol{\sigma}_1$ is the partial signature of v_t for N_1. Then, C calculates for N_i $(i \in \{1, 2, \ldots, n\})$ relevant to (v_t, σ_{i-1}) as follows:

$$\begin{aligned} l_j &= H_1(ID_j, L, P_j, R_j, v_t), \\ h_j &= H_0(ID_j, R_j), \\ &1 \leq j \leq i-1, \\ T &= H_2(v_t, L, P_{\text{pub}}). \end{aligned} \tag{12}$$

If $e(\sigma_{i-1}, P) = e(T, \sum_{j=1}^{i-1}(l_j P_j + R_j + h_j P_{\text{pub}}))$ holds, C calculates for N_i as follows:

$$\begin{aligned} l_i &= H_1(ID_i, L, P_i, R_i, v_t), \quad (1 \leq i \leq n), \\ \sigma_i &= \sigma_{i-1} + \text{SIGN}_i \\ &= \sigma_{i-1} + (l_i x_i + D_i)T. \end{aligned} \tag{13}$$

Finally, C calculates $\sigma_n = \sum_{i=1}^{n} \text{SIGN}_i$ and delivers $\boldsymbol{\sigma} = (\sigma_1, \sigma_2, \ldots, \sigma_d)$ which is sent to A_1.

Combination queries: A_1 requests a combination query. For the local coding vector $\alpha = (\alpha_1, \ldots, \alpha_m)$, global vector $\beta = (\beta_1, \ldots, \beta_m)$, and message vector $(w_1, w_2, \ldots, w_m)$, C combines the message vector $w = \sum_{i=1}^{m} \alpha_i w_i$. Then, the message vector is also denoted as $w = \sum_{j=1}^{m} \beta_j v_j$, and the signature process relevant to w is $\sigma_j = \sum_{i=1}^{m} \sigma_{i,j}^{\alpha_i}$, where $\sigma_{i,\,j}$ $(1 \leq i \leq m$ and $1 \leq j \leq l)$ denotes the j-th element of $\boldsymbol{\sigma}_i$. Finally, C outputs a combined signature $\sigma = \prod_{i=1}^{m} \boldsymbol{\sigma}_i^{\alpha_i}$.

Verification queries: A_1 requests a verification query. C runs the verification algorithm and returns a result if it is not the θ-th query; otherwise, C calculates $l_i = H_1(ID_i, L, P_i, R_i, v_t)$ $(1 \leq i \leq n)$ and $T = H_2(v_t, L, P_{\text{pub}})$. If the equality $e(\sigma_n, P) = e(T, \sum_{i=1}^{n}(l_i P_i + R_i + h_i P_{\text{pub}}))$ holds, C returns $\boldsymbol{\sigma}$ and $\perp$ otherwise.

Finally, A_1 outputs a forgery signature σ^*. In the adaptive queries, A_1 cannot request a full private key of ID_i, and σ^* is not returned by any multisignature oracle. If it is not the θ-th query, C fails and aborts the game; otherwise, C calls the H_0, H_1, and H_2 oracles and then searches the list L_3. Finally, C verifies the following equality:

$$\begin{aligned} &e(\sigma_n^*, P) \\ &= e\left(T^*, \sum_{i=1}^{n}\left(l_i^* P_i + R_i^* + h_i^* P_{\text{pub}}\right)\right) \\ &= e\left(bP, \sum_{i=1}^{n}(l_i^* P_i^* + R_i^*) + \sum_{i=1}^{n} h_i^* aP\right) \\ &= e\left(\sum_{i=1}^{n}(l_i^* x_i^* bP + r_i^* bP) + \sum_{i=1}^{n} h_i^* abP, P\right). \end{aligned} \tag{14}$$

From the above equality, we can obtain the solution of CDH problem:

$$\begin{aligned} \sigma_n^* &= \sum_{i=1}^{n}(l_i^* x_i^* bP + r_i bP) + \sum_{i=1}^{n} h_i^* abP, \\ \Rightarrow abP &= \frac{\sigma_n^* - \sum_{i=1}^{n}(l_i^* x_i^* bP + r_i bP)}{\sum_{i=1}^{n} h_i^*}. \end{aligned} \tag{15}$$

□

6.1. Probability Estimation. Probability that C succeeds in the above-mentioned game is estimated as follows. Here, it is necessary to think about three events:

E_1 is the event that C does not abort the game

E_2 is the event that A_1 successfully forge a signature

E_3 is the event that there exists at least one record of nontarget identity in successful forgery case

In E_1, there exists one time not querying the target identity, and then, $\Pr[E_1] \geq 1/(l_s + l_r)$, where l_s is the times of secret value query and l_r is the query times of public key replacement, E_2 denotes that A_1 wins in the game, then $\Pr[E_2 | E_1] \geq \varepsilon$, and E_3 at least occurs once time in n queries, then $\Pr[E_3 | E_1 \wedge E_2] \geq 1/n$. Hence, the success probability that C solves the CDH problem is

$$\begin{aligned} \varepsilon' &= \Pr[E_1 \wedge E_2 \wedge E_3] \\ &= \Pr[E_1] \cdot \Pr[E_2 | E_1] \Pr[E_3 | E_1 \wedge E_2] \\ &\geq \frac{\varepsilon}{n \cdot (l_s + l_r)}. \end{aligned} \tag{16}$$

Theorem 2. *In the RO model, if the polynomial time adversary A_2 can break the UF-CMA-II security of NC-CLMSS, a challenge algorithm C must be able to solve the CDH problem.*

Proof. C receives a random instance $(P, aP, bP) \in G_1$ of CDH problem, and its aim is to utilize A_2 (the subroutine of C) to determine the value of $abP \in G_1$. C maintains the initially empty lists L_0, L_1, L_2, and L_3 to save the query-answer values of several oracles. Firstly, $\mathcal{O}_C^{\text{setup}}(\rho) \longrightarrow^{\mu}, s\ A_2$.

Then, A_2 adaptively performs the polynomial time queries as below:

H_0 queries: for an H_0 query, if (ID_i, R_i, h_i) is in the list L_0, C returns h_i; otherwise, C returns $h_i \in_R Z_q^*$ and stores (ID_i, R_i, h_i) in L_0.

H_1 queries: for an H_1 query, if the matching tuple is in the list L_1, C returns l_i; otherwise, C returns $l_i \in_R Z_q^*$ and stores (ID_i, L, P_i, R_i, v_t, l_i) in L_1.

H_2 queries: for an H_2 query, if it is not the θ-th query ($\theta \in \{1, 2, \ldots, q_0\}$ (q_0 is the query times relevant to H_0 oracle) and the relevant tuple is in the list L_2, C randomly outputs $T = lP \in G_1$ ($l \in_R Z_q^*$) as the answer; after that, C stores (v_t, l_i, P_{pub}, l, T) in L_2, otherwise, C returns $T = bP \in G_1$ and stores (v_t, l_i, P_{pub},-,T) in L_2.

Partial private key queries: for a partial private key query for identity ID_i. C calculates $R_i = r_iP$, $D_i = r_i + h_is$ ($r_i \in_R Z_q^*$) and returns D_i and stores (ID_i, r_i, R_i, D_i, -, -) in the list L_3.

Public key queries: for a public key query for identity ID_i, if it is not the θ-th query, C calculates $P_i = x_iP$ ($x_i \in_R Z_q^*$) and finally returns $PK_i = (R_i, P_i)$ and updates L_3 with (ID_i, r_i, R_i, D_i, x_i, P_i); otherwise, C returns $PK_i = (R_i, P_i \longleftarrow aP)$ and updates L_3 with (ID_i, r_i, R_i, D_i, -, P_i).

Signature queries: A_2 issues a multisignature query for message v_t. If it is not the θ-th query, C runs the multisignature algorithm to output a result; otherwise, C signs v_t with the sequence $N_1 \longrightarrow N_2 \longrightarrow \ldots \longrightarrow N_n$. Firstly, C calculates for N_1 as follows:

$$\begin{aligned} l_1 &= H_1(ID_1, L, P_1, R_1, v_t), \\ T &= H_2(v_t, L, P_{\text{pub}}), \\ \sigma_1 &= \text{SIGN}_1 \\ &= l_1 lP_1 + D_1 lP, \end{aligned} \tag{17}$$

where $\boldsymbol{\sigma}_1$ is the partial signature of v_t for N_1. Then, C calculates for N_i ($i \in \{1, 2, \ldots, n\}$) relevant to (v_t, σ_{i-1}) as follows:

$$\begin{aligned} l_j &= H_1(ID_j, L, P_j, R_j, v_t), \\ h_j &= H_0(ID_j, R_j), \quad (1 \le j \le i-1), \\ T &= H_2(v_t, L, P_{\text{pub}}). \end{aligned} \tag{18}$$

If $e(\sigma_{i-1}, P) = e(T, \sum_{j=1}^{i-1}(l_jP_j + R_j + h_jP_{\text{pub}}))$ holds, C calculates for N_i as follows: $l_i = H_1(ID_i, L, P_i, R_i, v_t)$, $\sigma_i = \sigma_{i-1} + \text{SIGN}_i = \sigma_{i-1} + (l_ilP_i + D_ilP)$. C calculates $\sigma_n = \sum_{i=1}^{n} \text{SIGN}_i$ and returns $\boldsymbol{\sigma} = (\sigma_1, \sigma_2, \ldots, \sigma_d)$.

Combination queries: A_2 issues a combination query. For the local coding vector $\alpha = (\alpha_1, \ldots, \alpha_m)$, global vector $\beta = (\beta_1, \ldots, \beta_m)$ and message vector $(w_1, w_2, \ldots, w_m)$, C combines the message vector $w = \sum_{i=1}^{m} \alpha_i w_i$. Then, the message vector is also denoted as $w = \sum_{j=1}^{m} \beta_j v_j$, and the signature corresponding to w is $\sigma_j = \sum_{i=1}^{m} \sigma_{i,j}^{\alpha_i}$, where $\boldsymbol{\sigma}_{i,j}$ ($1 \le i \le m$ and $1 \le j \le l$) denotes the j-th element of $\boldsymbol{\sigma}_i$. Finally, C outputs a combined signature $\sigma = \prod_{i=1}^{m} \boldsymbol{\sigma}_i^{\alpha_i}$.

Verification queries: for a verification query, C runs the verification algorithm and returns a result if it is not the θ-th query; otherwise, C calculates $l_i = H_1(ID_i, L, P_i, R_i, v_t)$ $(1 \le i \le n)$ and $T = H_2(v_t, L, P_{\text{pub}})$. If $e(\sigma_n, P) = e(T, \sum_{i=1}^{n}(l_iP_i + R_i + h_iP_{\text{pub}}))$ holds, C returns $\boldsymbol{\sigma}$ and $\perp$ otherwise.

Finally, A_2 outputs a forgery signature σ^*. In queries, A_2 cannot query the secret value of ID_i, and σ^* is not returned by signature oracle. If it is not the θ-th query, C fails and aborts the game; otherwise, C calls the H_0, H_1, and H_2 oracles and then searches the list L_3 and then verifies as follows:

$$\begin{aligned} e(\sigma_n^*, P) &= e\left(T^*, \sum_{i=1}^{n}\left(l_i^* P_i^* + R_i^* + h_i^* P_{\text{pub}}\right)\right) \\ &= e\left(bP, \sum_{i=1}^{n}\left(l_i^* P_i^* + R_i^* + h_i^* P\right)\right) \\ &= e\left(\sum_{i=1}^{n}\left(l_i^* abP + r_i^* bP + h_i^* sbP\right), P\right). \end{aligned} \tag{19}$$

CDH problem solution can be obtained from equation (19):

$$\begin{aligned} \sigma_n^* &= \sum_{i=1}^{n}\left(l_i^* abP + r_i^* bP + h_i^* sbP\right), \\ \Rightarrow abP &= \frac{\sigma_n^* - \sum_{i=1}^{n}\left(r_i^* bP + h_i^* sbP\right)}{\sum_{i=1}^{n} l_i^*}. \end{aligned} \tag{20}$$

□

6.2. Probability Analysis. Probability that C succeeds in the above game is analyzed as follows. Here, we need to think about three events: E_1 is the event that C does not abort the game. In E_1, there exists one time not querying the target identity, and then, $\Pr[E_1] \ge 1/l_s$, where l_s is the times of secret value query. E_2 is the event that A_2 successfully forge a signature. E_2 denotes A_2 wins in the game, then $\Pr[E_2|E_1] \ge \varepsilon$. E_3 is the event that there exists at least one record of nontarget identity in successful forgery case. E_3 at least occurs once time in n queries, then $\Pr[E_3|E_1 \wedge E_2] \ge 1/n$. Hence, the probability that C solves the CDH problem is

$$\begin{aligned} \varepsilon' &= \Pr[E_1 \wedge E_2 \wedge E_3] \\ &= \Pr[E_1] \cdot \Pr[E_2 \mid E_1] \Pr[E_3 \mid E_1 \wedge E_2] \\ &\ge \frac{\varepsilon}{n \cdot l_s}. \end{aligned} \tag{21}$$

Theorem 3. *Our NC-CLMSS can prevent the pollution attacks in the multisource network coding environment.*

Proof. In NC-CLMSS, the signature process takes place at the source node and intermediate node. For the source node, the attacker can capture any node in the network and uses it

Table 2: The time complexity of cryptographic operations.

Cryptographic operations	Operation time (ms)
Time to perform an exponential operation: C_{me}	7.5
Time to perform a scalar multiplication: C_{mul}	1.56
Time to perform a hash operation: C_{mtp}	17.2
Time to perform a bilinear operation: C_{par}	19.7

Table 3: Computational efficiency comparison of several schemes.

References	Signature time (ms)	Verification time (ms)	Total time (ms)
Xu and Wang scheme	$3nC_{\mathrm{mtp}}+2nC_{\mathrm{mul}}+2nC_{\mathrm{par}}$	$3nC_{\mathrm{mtp}}+2nC_{\mathrm{par}}$	$6nC_{\mathrm{mtp}}+2nC_{\mathrm{mul}}+4nC_{\mathrm{par}}$
Niu et al. scheme	$(2n+1)\ C_{\mathrm{mtp}}+2nC_{\mathrm{mul}}+2nC_{\mathrm{par}}$	$(2n+1)\ C_{\mathrm{mtp}}+3nC_{\mathrm{par}}$	$2\ (2n+1)\ C_{\mathrm{mtp}}+2nC_{\mathrm{mul}}+5nC_{\mathrm{par}}$
Tanwar et al. scheme	$3nC_{\mathrm{mtp}}+4nC_{\mathrm{mul}}$	$3nC_{\mathrm{mtp}}+2nC_{\mathrm{mul}}$	$6nC_{\mathrm{mtp}}+6nC_{\mathrm{mul}}$
NC-CLMSS	$(2n+1)\ C_{\mathrm{mtp}}+2nC_{\mathrm{mul}}+2nC_{\mathrm{par}}$	$(2n+1)\ C_{\mathrm{mtp}}+2nC_{\mathrm{par}}$	$2\ (2n+1)\ C_{\mathrm{mtp}}+2nC_{\mathrm{mul}}+4nC_{\mathrm{par}}$

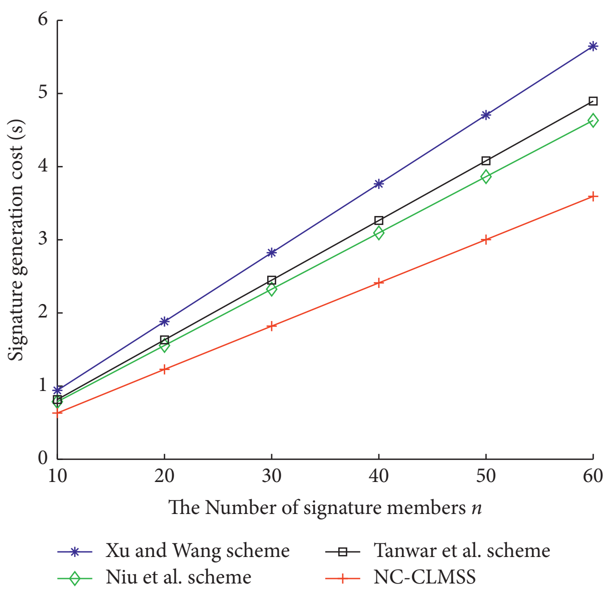

Figure 3: Comparison of the signature time.

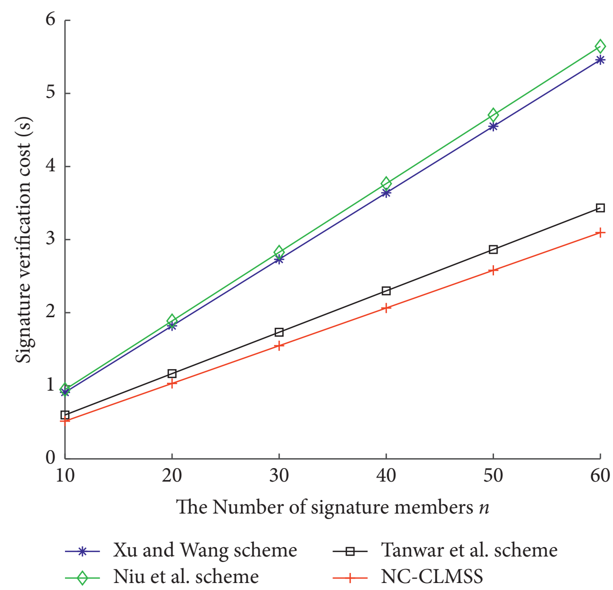

Figure 4: Comparison of the verification time.

to launch the attacks; this node sends the polluted information to the next node, but it is equivalent to solving the elliptic curve discrete logarithm (ECDL) problem for the attacker obtaining the signer private key from the public key. For the intermediate nodes, the attacker captures the signature from source node and tries to forge a signature; then, the attacker must own the user private key, and it is also equivalent to solving the ECDL problem.

NC-CLMSS can resist the pollution attacks in the network-coding environment because solving the ECDL problem is computationally infeasible. □

7. Performance Analysis

In this section, the performance comparison is made between NC-CLMSS and existing schemes in [19–21] based on the computational complexity. Schemes in [19–21] cannot resist the pollution attacks; the schemes in [20, 21] are not sequential multisignature. Our NC-CLMSS is a sequential multisignature and can resist pollution attacks. Table 2 describes the time complexity of main cryptography operations. Experimental environment for the performance analysis in this section: the processor is Intel (R) Core (TM) i7-6700HQ CPU @2.60GHz; the system type is the 64-bit operating system. Based on this system, we use C programming language, PBC library, and OpenSSL program to obtain the cryptography operation time, as shown in Table 2.

Table 3 describes the computational efficiency of several schemes. From Table 3, the computational complexity of NC-CLMSS is relatively lower than other schemes in [19–21].

Simulation curves of signature time-consuming of comparison schemes are shown in Figure 3. Simulation curves of verification time-consuming comparison are shown in Figure 4. Simulation curves of total algorithm time comparison are shown in Figure 5. Assume the number n of signature members is 10, 20, 30, 40, 50, and 60, respectively. Experiment results show the running time of different schemes increases linearly with the increase of the number of signed members. As shown in Figure 3, in the signature

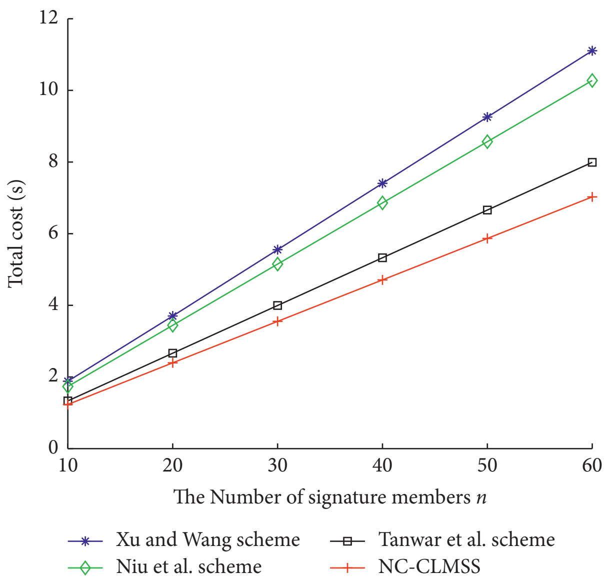

Figure 5: Total time comparison of several schemes.

phase, the growth rate of NC-CLMSS is relatively slower than other schemes. From Figure 4, the computational efficiency of NC-CLMSS is the highest. In terms of total time in Figure 5, NC-CLMSS takes the least time. Hence, NC-CLMSS is a relatively better cryptography algorithm in several schemes.

8. Summary

Network encoding cryptography has many merits, but there exists the inevitable problem how to resist the pollution attacks and forgery attacks in the message transmission process. By using the techniques of the certificateless multisignature and multisource network coding cryptosystem, we construct a certificateless multisignature scheme suitable for network coding (NC-CLMSS). Under the ECDL and CDH assumptions, this algorithm is proved to satisfy the UF-CMA security and can resist the pollution attacks; its computational complexity is relatively lower.

Data Availability

The data used to support the findings of this study are available from the corresponding author upon request.

Conflicts of Interest

The authors declare that they have no conflicts of interest.

Authors' Contributions

Huifang Yu worked on the security model, instance design, and security proof; Zhewei Qi worked on the instance design and simulation experiment; Danqing Liu worked on the introduction and formal algorithm definition; Ke Yang estimated the probability.

Acknowledgments

This work was supported by the Planning Subject for the 13th Five-Year Plan of National Education Science, under Grant EJA180476.

References

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

[2] Y. Wang, X. Lin, and H. Qu, "An effective hybrid network coding scheme against pollution attacks," *Netinfo Security*, vol. 19, no. 5, pp. 69–76, 2019.

[3] Z. Ning, W. Shi, L. Xiao, W. Liang, and T. Weng, "A novel approach for anti-pollution attacks in network coding," *Connection Science*, vol. 33, pp. 1–16, 2021.

[4] L. Wang, Z. Zhang, H. Zhang, and X. Zhang, "A RSA-based secure network coding scheme against multiple attacks," *Computer Engineering*, vol. 45, no. 11, pp. 166–171, 2019.

[5] Z. Zhou and L. Xu, "Anti-pollution network coding scheme based on digital signature," *Computer System Applications*, vol. 25, no. 6, pp. 185–190, 2016.

[6] H. F. Yu and W. Li, "A certificateless signature for multi-source network coding," *Journal of Information Security and Applications*, vol. 55, pp. 1–9, 2020.

[7] Z. Xu, D. He, P. Vijayakumar, K. R. Choo, and L. Li, "Efficient NTRU lattice-based certificateless signature scheme for medical cyber-physical systems," *Journal of Medical Systems*, vol. 44, no. 5, pp. 92–98, 2020.

[8] Z. Xu, M. Luo, N. Kumar, P. Vijayakumar, and L. Li, "Privacy-protection scheme based on sanitizable signature for smart mobile medical scenarios," *Wireless Communications and Mobile Computing*, vol. 2020, Article ID 8877405, 10 pages, 2020.

[9] Y. Y. Wang, M. Q. Shao, L. X. Cheng, X. Ma, and F. Y. Wang, "Application of multiple digital signatures on expert diagnose," *Journal of Shandong University of Technology*, vol. 30, no. 1, pp. 29–32, 2016.

[10] B. Geng, "Research on application of electronic medical record document multi-signature based on XML," *Computer and Modernization*, vol. 12, pp. 131–135, 2012.

[11] W. Kong, J. Shen, P. Vijayakumar, Y. Cho, and V. Chang, "A practical group blind signature scheme for privacy protection in smart grid," *Journal of Parallel and Distributed Computing*, vol. 136, pp. 29–39, 2020.

[12] C. Song, X. Gu, L. Wang, Z. Liu, and Y. Ping, "Research on identity-based batch anonymous authentication scheme for vehicular network," *Journal of Beijing University of Posts and Telecommunications*, vol. 42, no. 5, pp. 69–74, 2019.

[13] H. Z. Du, "Secure and efficient sequential multisignature scheme for VANET," *Application Research of Computers*, vol. 33, no. 10, pp. 3105–3108, 2016.

[14] C. Peng, J. Chen, M. S. Obaidat, P. Vijayakumar, and D. He, "Efficient and provably secure multireceiver signcryption scheme for multicast communication in edge computing," *IEEE Internet of Things Journal*, vol. 7, no. 7, pp. 6056–6068, 2019.

[15] P. Vijayakumar, S. Bose, and A. Kannan, "Improved harn batch digital signature algorithm for multicast authentication," *Journal of Discrete Mathematical Sciences and Cryptography*, vol. 17, no. 5-6, pp. 435–442, 2014.

[16] Z. Liu and S. Yang, "The application of multiple signatures based on ECC in E-government," *Journal of Guizhou University (Natural Sciences)*, vol. 29, no. 5, pp. 76–79, 2012.
[17] Q. Lei, "Sequential-multi digital signature application in electronic contract signing," *Computer and Modernization*, vol. 2, pp. 72–74, 2016.
[18] F. Zhang, H. Shao, and X. Z. Yuan, "Digital signature application in university management," *Software Engineering*, vol. 11, pp. 23-24, 2013.
[19] C. D. Xu and H. Q. Wang, "Order multiple signature scheme based on block chain," *Journal of Nanjing University of Posts and Telecommunications (natural science edition)*, vol. 2, pp. 90–99, 2021.
[20] S. Niu, W. Li, and C. Wang, "A new efficient certificateless broadcast multiple signature scheme," *Application Research of Computers*, vol. 37, no. 8, pp. 2464–2467, 2020.
[21] S. Tanwar, S. Badotra, M. Gupta, and A. Rana, "Efficient and secure multiple digital signature to prevent forgery based on ECC," *International Journal of Applied Science & Engineering*, vol. 18, no. 5, pp. 1–7, 2021.
[22] H. Yu and W. Wang, "Certificateless network coding ring signature scheme," *Security and Communication Networks*, vol. 2021, Article ID 8029644, 10 pages, 2021.