Frontiers of Information Technology & Electronic Engineering
www.jzus.zju.edu.cn; engineering.cae.cn; www.springerlink.com
ISSN 2095-9184 (print); ISSN 2095-9230 (online)
E-mail: jzus@zju.edu.cn

Certificateless broadcast multi-signature for network coding*

Huifang YU†‡, Zhewei QI

School of Cyberspace Security, Xi'an University of Posts and Telecommunications, Xi'an 710121, China

†E-mail: yuhuifang@xupt.edu.cn

Received June 22, 2022; Revision accepted July 19, 2022; Crosschecked Aug. 9, 2022

Abstract: Network coding can save wireless network resources and is very fast in comparison with traditional routing. In real application scenarios, network coding is vulnerable to pollution attacks and forgery attacks. To solve these problems, the certificateless broadcast multi-signature for network coding (NC-CLBMS) method is devised, where each source node user generates a multi-signature about the message vector, and the intermediate node linearly combines the received data. NC-CLBMS is a multi-source multi-signature method with anti-pollution and anti-forgery advantages; moreover, it has a fixed signature length and its computation efficiency is very high. NC-CLBMS has extensive application prospects in unmanned aerial vehicle (UAV) communication networks, fifth-generation wireless networks, wireless sensor networks, mobile wireless networks, and Internet of Vehicles.

Key words: Network coding; Certificateless multi-signature; Linear combination; Homomorphic hash function
https://doi.org/10.1631/FITEE.2200271 **CLC number:** TP309

1 Introduction

Traditional routing technology is unable to satisfy the increasing demand for network communication. Improving network utilization and efficiency with limited network resources has become the key problem in the field of information theory research.

In theory, network coding (Ahlswede et al., 2000) can make actual network transmission reach the maximum capacity. Network coding can improve transmission reliability (Li P et al., 2012; Papailiopoulos et al., 2012), network transmission efficiency (Li ZP et al., 2009), and network robustness (Al-Kofahi and Kamal, 2009) because of integrating coding and routing. Traditional routing technology allows only intermediate nodes to save and forward received data. Network coding can use the intermediate nodes to combine and encode the received data packets, and improve the overall network performance. However, network coding also has many security problems. Its unique topology (Xu J et al., 2016) makes it more vulnerable to pollution attacks. Malicious nodes can tamper with or forge data information; if this data is used for encoding with other unpolluted data, the polluted messages will spread to an entire network and cause unnecessary losses.

Scholars have devised some secure network coding schemes to handle security problems, such as elliptic curve encryption for network coding (Wang L et al., 2019) and network coding signature schemes (Peng et al., 2015; Li SH and Mei, 2016; Wang HP and Mei, 2016; Zhou and Xu, 2016; Wang L et al., 2019; Yu and Li, 2019, 2020; Niu et al., 2020; Xu CD and Wang, 2021; Yu and Wang, 2021). In the network coding environment, the intermediate nodes combine and encode the information data and send the combination result to downstream nodes. The validity of the traditional signature is destroyed when the intermediate nodes encode the data.

The followings are the characteristics of broadcast multi-signature: (1) the signature length has nothing

‡ Corresponding author
* Project supported by the Key Project of Natural Science Basis Research Plan of Shaanxi Province, China (No. 2020JZ-54)
ORCID: Huifang YU, https://orcid.org/0000-0003-4711-3128

to do with the number of users; (2) the public key is used to verify the signature; (3) signers sign the messages out of sequence to obtain valid multi-signature; (4) it is impossible to obtain valid multi-signature without all signers working together. Until now, there is no certificateless multi-source broadcast multi-signature scheme, because the traditional certificateless broadcast multi-signature cannot be directly used in network coding. The construction of the secure network coding scheme is an open problem.

In this work, we construct a certificateless broadcast multi-signature for network coding (NC-CLBMS) method, in which each source node outputs a final broadcast multi-signature result and the intermediate node linearly combines the data information from different links to transfer the combination result to downstream nodes. NC-CLBMS creates multi-source multi-signatures that are out of sequence and the signature length is fixed. The hash function in this scheme can ensure the homomorphism of the signature process. NC-CLBMS is helpful when multiple users need to sign the same message, and can resist pollution and forgery attacks.

2 Preliminaries

2.1 Notations

Notations and their meanings are described in Table 1.

Table 1 Symbols and their meanings

Notation	Meaning
1^{ρ}	ρ-bit security parameter
s	Master key
ID_i	User identity
G_1	Addition cyclic group
$\mathcal{Y}_{\mathrm{pub}}$	System public key
$\mathcal{Y}_i$	User public key
d_i	Partial private key of the user
$\boldsymbol{v}_i$	Message vector
P	Generator of addition cyclic group G_1
G_2	Multiplication cyclic group
H_i	Secure hash functions
σ	Multi-signature
$\boldsymbol{\alpha}$	Local coding vector
$\boldsymbol{\beta}$	Global coding vector

2.2 Multi-source transmission model

Multi-source network coding (Yu and Gao, 2019) can use node computing power to improve network bandwidth utilization. Multi-source nodes are essential in the transmission model as shown in Fig. 1. The multi-source network is viewed as an acyclic directed graph $G=(M, E)$, where E is the edge set and M is the node set. In the multi-source network, the set of source nodes is $u=\{u_1, u_1, \cdots, u_m\}\in M$, and the set of sink nodes is $d=\{d_1, d_1, \cdots, d_m\}\in M$.

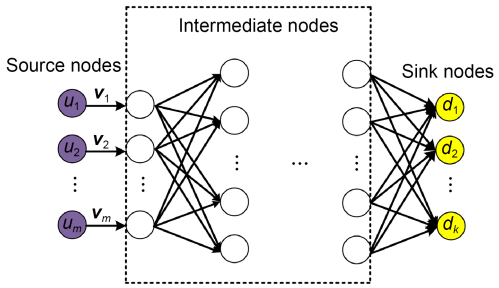

Fig. 1 Multi-source transmission network model

Each message vector $\boldsymbol{v}_i=(v_{i,1}, v_{i,2},\cdots, v_{i,m})$, where $\boldsymbol{v}_i\in\{\boldsymbol{v}_1, \boldsymbol{v}_2,\cdots, \boldsymbol{v}_m\}$. If the destination node can decode the encoded message, source node u_i needs to extend an n-dimensional coefficient vector to message vector $\boldsymbol{v}_i$ and the extended message vector is $\boldsymbol{v}_i=(v_{i,1}, v_{i,2}, \cdots, v_{i,n}, v_{i,n+1}, \cdots, v_{i,n+m})\in F^{n+m}$. Hence, $\boldsymbol{w}_i=(w_{i,1}, w_{i,2}, \cdots, w_{i,m}, \beta_{i,m+1}, \cdots, \beta_{i,m+n})\in F^{m+n}$.

The intermediate node receives the coding message vectors $(\boldsymbol{w}_1, \boldsymbol{w}_2, \cdots, \boldsymbol{w}_m)\in F^{n+m}$ from m links and obtains $\boldsymbol{w}=\sum_{i=1}^{m}\boldsymbol{\alpha}_i\boldsymbol{w}_i$, where $(\boldsymbol{\alpha}_1, \boldsymbol{\alpha}_2, \cdots, \boldsymbol{\alpha}_m)$ are local coding vectors; $\boldsymbol{w}$ received by any network node is also a linear combination of the original message $\boldsymbol{v}_i$: $\boldsymbol{w}=\sum_{i=1}^{m}\boldsymbol{\beta}_i\boldsymbol{v}_i$, where $\boldsymbol{\beta}_i$ $(i=1,2,\cdots,m)$ are global coding vectors.

Any destination node can receive m linearly independent $(\boldsymbol{w}_1, \boldsymbol{w}_2, \cdots, \boldsymbol{w}_m)$, where $\boldsymbol{w}_i$ $(i=1,2,\cdots,m)$ are denoted as

$$\begin{bmatrix}\boldsymbol{w}_1\\ \vdots\\ \boldsymbol{w}_m\end{bmatrix}=\mathcal{X}\begin{bmatrix}\boldsymbol{v}_1\\ \vdots\\ \boldsymbol{v}_m\end{bmatrix},\ \mathcal{X}=(\boldsymbol{\beta}_1,\boldsymbol{\beta}_2,\cdots,\boldsymbol{\beta}_m)^{-1}.$$

Because the matrix $\mathcal{X}$ is invertible and of full rank, the original message can be obtained after decoding as follows:

$$\begin{bmatrix}\boldsymbol{v}_1\\ \vdots\\ \boldsymbol{v}_m\end{bmatrix}=\mathcal{X}^{-1}\begin{bmatrix}\boldsymbol{w}_1\\ \vdots\\ \boldsymbol{w}_m\end{bmatrix}.$$

2.3 Bilinear pairing

G_1 and G_2 are two cyclic groups with prime order q and a generator of G_1 is P, where G_1 is an additive group and G_2 is a multiplicative group. Bilinear map e: $G_1\times G_1\to G_2$ satisfies three properties as follows: (1) $e(aP,bP)=e(P,P)^{ab}$, $\forall a,b\in Z_q^*$, $P\in G_1$; (2) $e(P,P)\neq 1$; (3) $\forall P,Q\in G_1$, there exists an efficient algorithm to calculate $e(P,Q)$.

Definition 1 (Computational Diffie-Hellman (CDH) problem) Given $(aP,bP)\in G_1$, $\forall a,b\in Z_q^*$, the CDH problem is to calculate $abP\in G_1$.

3 Formation definition

3.1 Algorithm definition

NC-CLBMS is defined by six polynomial time algorithms as follows:

Setup: Input the security parameter 1^ρ, and this algorithm outputs the system parameter set μ and master key s.

KeyGen: Input (μ,ID_i), and this algorithm outputs the user's public-private pair $(\mathcal{Y}_i,x_i)$, where ID_i is the user's identity.

Extract: Input (μ,ID_i,s), and this algorithm outputs the partial private key d_i of the user.

MultiSIG: Input $(\mu,\{\mathrm{ID}_1,\mathrm{ID}_2,\cdots,\mathrm{ID}_n\},x_i,d_i)$, and this algorithm outputs a multi-signature c_i.

Combine: Input $(c_i,\boldsymbol{v}_i)$, and this algorithm outputs a combination result γ.

Verification: Input $(\mu,c_i,\boldsymbol{v}_i,\gamma)$, and this algorithm outputs a verification result.

3.2 Security model

NC-CLBMS must be unforgeable against the adaptive chosen message attacks (UF-CMAs). The UF-CMA security model relies on Game1 and Game2, where A_1 cannot obtain the master key but can change any public key, and A_2 knows the master key but cannot change any public key.

Game1 (Game2): $\mathcal{O}_C^{\mathrm{setup}}(1^\rho)\xrightarrow{\mu}A_1$, $\mathcal{O}_C^{\mathrm{setup}}(1^\rho)\xrightarrow{(\mu,s)}A_2$, Then, $A_1(A_2)$ makes adaptive queries as follows:

$$\mathcal{O}_C^{\text{Public key}}(\mathrm{ID}_i)\xrightarrow{\mathcal{Y}_i}A_1(A_2),$$

$$\mathcal{O}_C^{\text{Partial private key}}(\mathrm{ID}_i)\xrightarrow{D_i}A_1(A_2),$$

$$\mathcal{O}_C^{\text{Private key}}(\mathrm{ID}_i)\xrightarrow{x_i\text{ if }\mathcal{Y}_i\text{ is not replaced}}A_1,$$

$$\mathcal{O}_C^{\text{Private key}}(\mathrm{ID}_i)\xrightarrow{x_i}A_2,$$

$$\mathcal{O}_C^{\text{Replace public key}}(\mathrm{ID}_i)\xrightarrow{\mathcal{Y}_i'}A_1,$$

$$\mathcal{O}_C^{\text{MultiSIG}}(\boldsymbol{v}_i)\xrightarrow{c_i}A_1(A_2),$$

$$\mathcal{O}_C^{\text{Combine}}(\boldsymbol{v}_i)\xrightarrow{c}A_1(A_2),$$

$$\mathcal{O}_C^{\text{Verfication}}(\boldsymbol{c},\sigma)\xrightarrow{\boldsymbol{v}_i\text{ or }\perp}A_1(A_2).$$

Finally, A_1 (A_2) wins in Game1 (Game2) if and only if the signature of ID_i is valid. Here, A_1 cannot query a full private key of ID_i whose public key cannot be replaced, and forgery signature is not returned by multi-signature oracle from A_1 (A_2).

Advantage of A_1 (A_2) is the probability that A_1 (A_2) succeeds in Game1 (Game2).

Definition 2 (Unforgeability) A NC-CLBMS scheme is said to be UF-CMA secure if no A_1 (resp. A_2) can win in Game1 (resp. Game2) with a non-negligible advantage.

4 Concrete NC-CLBMS

4.1 Setup

The key generation center (KGC) chooses G_1 and G_2 with ρ–bit prime order q as in Section 2.3, and e: $G_1\times G_1\to G_2$ is a bilinear map. KGC chooses a master key $s\in_R Z_q^*$ and calculates a system public key $\mathcal{Y}_{\mathrm{pub}}=sP$, where P is a generator of G_1. KGC chooses two secure hash functions H_0:$\{0,1\}^*\to Z_q^*$ and H_1: $\{0,1\}^*\to G_1$.

Finally, KGC keeps s to itself and publishes system parameters $\mu=(G_1,G_2,e,P,\mathcal{Y}_{\mathrm{pub}},H_0,H_1)$.

4.2 KeyGen

The user with identity ID_i randomly chooses a secret value $x_i\in Z_q^*$ and calculates the public key $\mathcal{Y}_i=x_iP$.

4.3 Extract

KGC randomly chooses $r_i\in Z_q^*$ and calculates $R_i=r_iP$. Then, KGC continues to calculate $\psi_i=H_0(\mathrm{ID}_i,\mathcal{Y}_i,R_i)$, $d_i=r_i+s\psi_i$, and delivers d_i to the user with identity ID_i, where d_i is the partial private key of this user and the set $r\leftarrow\{R_1,R_2\cdots,R_n\}$ is published.

4.4 MultiSIG

The source node user broadcasts $\gamma_i = H_1(\mathrm{id}, \boldsymbol{v}_i)$, $\sigma_i = (x_i + d_i)\gamma_i$, where the packets with same id must be encoded together and id is assigned to each packet. The combiner calculates $c_i = \sigma = \sum_{i=1}^{n}\sigma_i$. Finally, $(\boldsymbol{v}_i, c_i)|_{i=1}^{m}$ from each source node are sent to intermediate nodes and sink nodes.

4.5 Combine

After $(\boldsymbol{v}_i, c_i)|_{i=1}^{m}$ arrive at the intermediate node, the intermediate node calculates $\boldsymbol{w}=\sum_{i=1}^{m}\boldsymbol{\beta}_i\boldsymbol{v}_i$, where the coding message vector $\boldsymbol{w}=(\boldsymbol{w}_1,\boldsymbol{w}_2,\cdots,\boldsymbol{w}_m)$ and global coding vector $\boldsymbol{\beta}=(\boldsymbol{\beta}_1, \boldsymbol{\beta}_2, \cdots, \boldsymbol{\beta}_m)$. Then, the intermediate node calculates $\gamma = \prod_{i=1}^{m}\gamma_i^{\boldsymbol{\beta}_i}$, $\xi_1 = \prod_{i=1}^{m}e(c_i, P)^{\boldsymbol{\beta}_i}$, where $\gamma_i = H_1(\mathrm{id}, \boldsymbol{v}_i)$. Finally, the intermediate node returns the combination result $\boldsymbol{c} \leftarrow (\gamma, \xi_1)$.

4.6 Verification

After $(\boldsymbol{v}_i, c_i)|_{i=1}^{m}$, $\boldsymbol{c} \leftarrow (\gamma, \xi_1)$ arrive at the sink node, this verification algorithm is run as follows:

1. Calculate $\gamma_i = H_1(\mathrm{id}, \boldsymbol{v}_i)$.

2. Calculate $\xi_2 = e\left(\gamma,\ \mathcal{Y}_{\mathrm{pub}}\sum_{i=1}^{n}\psi_i + \mathcal{Y} + R\right)$, where $\mathcal{Y}=\sum_{i=1}^{n}\mathcal{Y}_i$, $R = \sum_{i=1}^{n}R_i$.

3. The signature of $\boldsymbol{v}_i$ is valid if $\xi_1 = \xi_2$, and invalid otherwise.

5 Correctness analysis

Signed packet verification is shown as follows. Note that $H_1(\mathrm{id}, \boldsymbol{v}_i)$ is an essential homomorphism hash function, for which $\gamma = \prod_{i=1}^{m}H_1(\mathrm{id}, \boldsymbol{v}_i)^{\boldsymbol{\beta}_i} = H_1(\mathrm{id}, \sum_{i=1}^{m}\boldsymbol{\beta}_i\boldsymbol{v}_i) = H_1(\mathrm{id}, \boldsymbol{w})$. So, we can obtain

$$
\begin{aligned}
\xi_1 &= e(\boldsymbol{c}, P)\\
&= \prod_{i=1}^{m}e(c_i, P)^{\boldsymbol{\beta}_i}\\
&= \prod_{i=1}^{m}e\left(\gamma_i^{\boldsymbol{\beta}_i}, \sum_{i=1}^{n}(x_i + d_i + r_i)P\right)\\
&= e\left(\prod_{i=1}^{m}\gamma_i^{\boldsymbol{\beta}_i}, \sum_{i=1}^{n}\mathcal{Y}_i + R_i + h_i\mathcal{Y}_{\mathrm{pub}}\right)\\
&= e\left(\prod_{i=1}^{m}H_1(\mathrm{id}, \boldsymbol{v}_i)^{\boldsymbol{\beta}_i}, \mathcal{Y} + R + \sum_{i=1}^{n}\psi_i\mathcal{Y}_{\mathrm{pub}}\right)\\
&= e\left(H_1\left(\mathrm{id}, \sum_{i=1}^{n}\boldsymbol{\beta}_i\boldsymbol{v}_i\right), \mathcal{Y} + R + \mathcal{Y}_{\mathrm{pub}}\sum_{i=1}^{n}\psi_i\right)\\
&= e\left(H_1(\mathrm{id}, \boldsymbol{w}), \mathcal{Y} + R + \mathcal{Y}_{\mathrm{pub}}\sum_{i=1}^{n}\psi_i\right)\\
&= e\left(\gamma, \mathcal{Y} + R + \mathcal{Y}_{\mathrm{pub}}\sum_{i=1}^{n}\psi_i\right)\\
&= \xi_2.
\end{aligned}
$$

6 Security analysis

6.1 Anti-forgery attacks

Theorem 1 (Unforgeability-I) If an attacker A_1 can break the UF-CMA-I security with advantage ε in a random oracle model, a challenger C can solve the CDH problem with advantage $\varepsilon' \geq \varepsilon/[ne(q_{\mathrm{r}}+q_{\mathrm{s}}+q'_{\mathrm{s}})]$, where q_{r}, q_{s}, and q'_{s} are the times to query the public key replacement oracle, secret value oracle, and partial private key oracle, respectively.

Proof Assume that the remaining $n-1$ users are corrupted except ID_i, where there are n users in NC-CLBMS and $\mathrm{ID}_i \in \{\mathrm{ID}_1, \mathrm{ID}_2, \cdots, \mathrm{ID}_n\}$. C is a challenger to the CDH problem instance $(P, aP, bP) \in G_1$, and its aim is to use A_1 (the subroutine of C) to obtain $abP \in G_1$. Initially, empty lists L_0, L_1, and L_2 track various queries from A_1. ID_τ is the target identity and $\tau \in \{1, 2, \cdots, \delta\}$, where δ is the time to query H_0 oracle.

$\mathcal{O}_C^{\mathrm{setup}}(1^\rho) \xrightarrow{\mu(\mathcal{Y}_{\mathrm{pub}} = aP)} A_1$. Then, A_1 makes a series of adaptive queries as follows:

H_0 queries: C receives an H_0 query of ID_i from A_1. C checks whether L_0 contains a matching tuple $(\mathrm{ID}_i, \mathcal{Y}_i, R_i, \psi_i)$. If yes, C returns ψ_i to A_1; otherwise, C returns ψ_i satisfying $\psi_i P = bP$ and stores $(\mathrm{ID}_i, \mathcal{Y}_i, R_i, \psi_i)$ in L_0.

H_1 queries: C receives an H_1 query of ID_i from A_1. C checks whether L_1 contains a related tuple $(\mathrm{id}, \boldsymbol{v}_i, \gamma_i, \lambda_i)$. If yes, C return γ_i to A_1; otherwise, C returns $\gamma_i \leftarrow \lambda_i P$ to A_1 and records $(\mathrm{id}, \boldsymbol{v}_i, \gamma_i, \lambda_i)$ in L_1, where $\lambda_i \in_{\mathrm{R}} Z_q^*$.

Public key queries: C receives a public key query of ID_i from A_1. C returns $\mathcal{Y}_i = x_i P$ and stores $(\mathrm{ID}_i, r_i, -, x_i, \mathcal{Y}_i)$ in L_2, where $x_i \in_{\mathrm{R}} Z_q^*$.

Partial private key queries: C receives a partial private key query of ID_i from A_1. C chooses $r_i \in_{\mathrm{R}} Z_q^*$ to calculate $R_i = r_i P \in G_1$, returns d_i satisfying $d_i P = R_i + l_i\mathcal{Y}_{\mathrm{pub}}$, and updates L_2 with $(\mathrm{ID}_i, r_i, d_i, x_i, \mathcal{Y}_i)$ if $\mathrm{ID}_i \neq \mathrm{ID}_\tau$, where l_i is from H_0 oracle; otherwise, C fails and aborts.

Secret value queries: C receives a secret value query of ID_i from A_1. C returns x_i from L_2 to A_1 if $\mathrm{ID}_i \neq \mathrm{ID}_\tau$; otherwise, C fails and aborts.

Public key replacement: C receives a public key replacement of ID_i from A_1. If $\mathrm{ID}_i \neq \mathrm{ID}_\tau$, C replaces $\mathcal{Y}_i$ with $\mathcal{Y}_i'$ from A_1 and updates L_2 with $(\mathrm{ID}_i, r_i, d_i, -, \mathcal{Y}_i')$; otherwise, C fails and aborts.

Multi-signature queries: C receives a query of multi-signature about $\boldsymbol{v}_i$. C calls the actual multi-signature algorithm to return a result if $\mathrm{ID}_i \neq \mathrm{ID}_\tau$; otherwise, C calculates $\gamma_i = H_1(\mathrm{id}, \boldsymbol{v}_i)$, $\sigma_i = \lambda_i\left(\mathcal{Y}_i + R_i + l_i\mathcal{Y}_{\mathrm{pub}}\right)$. Finally, C calculates $c_i = \sigma = \sum_{i=1}^{n}\sigma_i$ and returns $(\boldsymbol{v}_i, c_i)$ to A_1.

Combination queries: C receives a combination query from A_1. C calls the actual combination algorithm to return a result if it is not the τ^{th} query; otherwise, for $\boldsymbol{\beta} = (\boldsymbol{\beta}_1, \boldsymbol{\beta}_2, \cdots, \boldsymbol{\beta}_m)$ and $\boldsymbol{w} = (\boldsymbol{w}_1, \boldsymbol{w}_2, \cdots, \boldsymbol{w}_m)$, C calculates the combined message $\boldsymbol{w} = \sum_{i=1}^{m}\boldsymbol{\beta}_i\boldsymbol{v}_i$, and C returns the combination results $\gamma \leftarrow \prod_{i=1}^{m}\gamma_i^{\boldsymbol{\beta}_i}$, $\xi_1 \leftarrow \prod_{i=1}^{m}e\left(c_i, P\right)^{\boldsymbol{\beta}_i}$.

Verification queries: C receives a verification query from A_1. C calls the actual verification algorithm to return a result if $\mathrm{ID}_i \neq \mathrm{ID}_\tau$; otherwise, C responds as follows:

1. Calculate $\gamma_i = H_1(\mathrm{id}, \boldsymbol{v}_i)$.

2. Calculate $\xi_2 = e\left(\gamma, \mathcal{Y}_{\mathrm{pub}}\sum_{i=1}^{n}\psi_i + \mathcal{Y} + R\right)$, where $\mathcal{Y} = \sum_{i=1}^{n}\mathcal{Y}_i$, $R = \sum_{i=1}^{n}R_i$.

3. If $\xi_1 = \xi_2$ holds, C returns $\boldsymbol{v}_i$; otherwise, C returns $\perp$.

As above-mentioned queries are over, A_1 outputs a forgery signature c_i^*. In queries, A_1 cannot query a full private key of ID_i^*, and c_i^* is not returned by multi-signature oracle. If $\mathrm{ID}_i \neq \mathrm{ID}_\tau$, C fails and aborts; otherwise, C calculates

$$\begin{aligned} e\left(c_i^*, P\right) &= e\left(\sum_{i=1}^{n}\left(x_i^* + d_i^*\right)\gamma_i^*, P\right) \\ &= e\left(\sum_{i=1}^{n}\left(x_i^* + r_i^* + s\psi_i^*\right)\gamma_i^*, P\right) \\ &= e\left(\sum_{i=1}^{n}\left(x_i^*\gamma_i^* + r_i^*\gamma_i^* + s\psi_i^*\gamma_i^*\right), P\right) \\ &= e\left(\sum_{i=1}^{n}\left(x_i^* + r_i^*\right)\gamma_i^* + abP\sum_{i=1}^{n}\lambda_i^*, P\right). \end{aligned}$$

Then, C uses A_1 to solve the CDH problem as follows:

$$\begin{aligned} c_i^* &= \sum_{i=1}^{n}\left(x_i^* + r_i^*\right)\gamma_i^* + abP\sum_{i=1}^{n}\lambda_i^* \\ \Rightarrow abP &= \left(c_i^* - \sum_{i=1}^{n}\left(x_i^* + r_i^*\right)\gamma_i^*\right)\Big/\sum_{i=1}^{n}\lambda_i^*. \end{aligned}$$

Probability analysis: Advantage ε' of the CDH problem under forgery attacks is equal to the probability of simultaneous occurrence of three events as follows:

E_1: C cannot fail and abort the game.

E_2: A_1 successfully forges a multi-signature and $\Pr[E_2|E_1] \geq \varepsilon$.

E_3: There exists at least one record of non-target identity in a successful forgery case.

As described in Theorem 1, we can obtain

$$\begin{aligned} \Pr\left[E_1\right] &\geq \left(1 - \frac{1}{q_{\mathrm{s}} + q_{\mathrm{r}} + q_{\mathrm{s}}'}\right)^{\left(q_{\mathrm{s}} + q_{\mathrm{r}} + q_{\mathrm{s}}'\right)} \\ &\geq \frac{1}{\mathrm{e}\left(q_{\mathrm{s}} + q_{\mathrm{r}} + q_{\mathrm{s}}'\right)}, \\ \Pr\left[E_3|E_1 \cap E_2\right] &\geq 1/n. \end{aligned}$$

If A_1 can break the UF-CMA-I security of NC-CLBMS with advantage ε, C can solve the CDH problem with advantage ε', where

$$\begin{aligned} \varepsilon' &= \Pr\left[E_1 \cap E_2 \cap E_3\right] \\ &= \Pr\left[E_1\right]\Pr\left[E_2|E_1\right]\Pr\left[E_3|E_1 \cap E_2\right] \\ &\geq \frac{\varepsilon}{n\mathrm{e}\left(q_{\mathrm{s}} + q_{\mathrm{r}} + q_{\mathrm{s}}'\right)}. \end{aligned}$$

Theorem 2 (Unforgeability-II) If an attacker A_2 can break the UF-CMA-II security with advantage ε in a random oracle model, a challenger C can solve the CDH problem with advantage $\varepsilon' \geq \varepsilon/(n\mathrm{e}q_{\mathrm{s}})$, where q_{s} is the query time to secret value oracle.

Proof Assume that the remaining $n-1$ users are corrupted except ID_i, where there are n users in NC-CLBMS and $\mathrm{ID}_i \in \{\mathrm{ID}_1, \mathrm{ID}_2, \cdots, \mathrm{ID}_n\}$. C is a challenger to the CDH problem instance $(P, aP, bP) \in G_1$ and its aim is to use A_2 to obtain $abP \in G_1$. Initially, empty lists L_0, L_1, and L_2 record various query-answer values from A_2. ID_τ is the target identity, $\tau \in \{1, 2, \cdots, \delta\}$, and δ is the query time to H_0 oracle.

$\mathcal{O}_C^{\mathrm{setup}}\left(1^\rho\right) \xrightarrow{\mu \text{ and } s\left(\mathcal{Y}_{\mathrm{pub}} = sP\right)} A_2$. Then, A_2 submits a series of adaptive queries as follows:

H_0 queries: C receives an H_0 query of ID_i from A_1. C checks whether L_0 contains a related tuple $(\mathrm{ID}_i, \mathcal{Y}_i, R_i, \psi_i)$. If yes, C returns ψ_i; otherwise, C returns $\psi_i \leftarrow Z_q^*$ to A_1 and adds $(\mathrm{ID}_i, \mathcal{Y}_i, R_i, \psi_i)$ to L_0.

H_1 queries: C receives an H_1 query of ID_i from A_2. C checks whether there is $(\mathrm{id}, \boldsymbol{v}_i, \gamma_i)$ in L_1. If yes, C return γ_i to A_2; otherwise, C outputs $\gamma_i \leftarrow aP$ and updates L_1.

Public key queries: C receives a public key query of ID_i from A_2. If $\mathrm{ID}_i \neq \mathrm{ID}_\tau$, C returns $\mathcal{Y}_i \leftarrow x_i P$ to A_1 and stores $(\mathrm{ID}_i, r_i, -, x_i, \mathcal{Y}_i)$ in L_2, where $x_i \in_{\mathrm{R}} Z_q^*$; otherwise, C returns $\mathcal{Y}_i \leftarrow bP$ and stores $(\mathrm{ID}_i, r_i, -, -, \mathcal{Y}_i)$ in L_2.

Partial private key queries: C receives a partial private key query of ID_i from A_2. If $\mathrm{ID}_i \neq \mathrm{ID}_\tau$, C selects $r_i \in_{\mathrm{R}} Z_q^*$ to calculate $R_i = r_i P \in G_1$, and then returns $d_i \leftarrow r_i + s\psi_i$ to A_2 and updates L_2 with $(\mathrm{ID}_i, r_i, d_i, x_i, \mathcal{Y}_i)$; otherwise, C fails and aborts.

Secret value queries: C receives a secret value query of ID_i from A_1. C returns x_i from L_2 if $\mathrm{ID}_i \neq \mathrm{ID}_\tau$; otherwise, C fails and aborts.

Multi-signature queries: C receives a query of multi-signature about $\boldsymbol{v}_i$. C calls the actual multi-signature algorithm to return a result if $\mathrm{ID}_i \neq \mathrm{ID}_\tau$; otherwise, C calculates $\gamma_i = H_1(\mathrm{id}, \boldsymbol{v}_i)$, $\sigma_i = \lambda_i\left(\mathcal{Y}_i + R_i + l_i \mathcal{Y}_{\mathrm{pub}}\right)$. C calculates $c_i = \sigma = \sum_{i=1}^{n} \sigma_i$ and returns $(\boldsymbol{v}_i, c_i)$ to A_2.

Combination queries: C receives a combination query from A_2. C calls the actual combination algorithm to return a result if it is not the τ^{th} query; otherwise, for $\boldsymbol{\beta}=(\boldsymbol{\beta}_1, \boldsymbol{\beta}_2, \cdots, \boldsymbol{\beta}_m)$ and $\boldsymbol{w}=(\boldsymbol{w}_1, \boldsymbol{w}_2, \cdots, \boldsymbol{w}_m)$, C calculates the combined message $\boldsymbol{w}=\sum_{i=1}^{m} \boldsymbol{\beta}_i \boldsymbol{v}_i$ and returns the combination results $\gamma \leftarrow \prod_{i=1}^{m} \gamma_i^{\boldsymbol{\beta}_i}$, $\xi_1 \leftarrow \prod_{i=1}^{m} e\left(c_i, P\right)^{\boldsymbol{\beta}_i}$.

Verification queries: C receives a verification query from A_2. C calls the verification algorithm to return a result if $\mathrm{ID}_i \neq \mathrm{ID}_\tau$; otherwise, C answers as follows:

1. Calculate $\gamma_i = H_1(\mathrm{id}, \boldsymbol{v}_i)$.

2. Calculate $\xi_2 = e\left(\gamma, \mathcal{Y}_{\mathrm{pub}} \sum_{i=1}^{n} \psi_i + \mathcal{Y} + R\right)$, where $\mathcal{Y} = \sum_{i=1}^{n} \mathcal{Y}_i$, $R = \sum_{i=1}^{n} R_i$.

3. If $\xi_1 = \xi_2$ holds, C returns $\boldsymbol{v}_i$; otherwise, C returns $\perp$.

After above-mentioned queries, A_2 outputs a forgery signature c_i^*. In adaptive queries, A_2 cannot query the secret value of ID_i^* and c_i^* is not returned by multi-signature oracle. If $\mathrm{ID}_i \neq \mathrm{ID}_\tau$, C fails and aborts; otherwise, C calculates

$$\begin{aligned} e\left(c_i^*, P\right) &= e\left(\sum_{i=1}^{n}\left(x_i^* + d_i^*\right)\gamma_i^*, P\right) \\ &= e\left(\sum_{i=1}^{n}\left(x_i^*\gamma_i^* + d_i^*\gamma_i^*\right), P\right) \\ &= e\left(\sum_{i=1}^{n}\left(abP + d_i^*\gamma_i^*\right), P\right) \\ &= e\left(nabP + \sum_{i=1}^{n} d_i^*\gamma_i^*, P\right). \end{aligned}$$

C uses A_2 to successfully solve the CDH problem, and the solution to the CDH problem is as follows:

$$\begin{aligned} \boldsymbol{c}_i &= nabP + \sum_{i=1}^{n} d_i^*\gamma_i^* \\ \Rightarrow abP &= \frac{1}{n}\left(c_i - \sum_{i=1}^{n} d_i^*\gamma_i^*\right). \end{aligned}$$

Probability analysis: Advantage ε' of the CDH problem under forgery attacks is equal to the probability of simultaneous occurrence of three events as follows:

E_1: C cannot fail and terminate the game.

E_2: A_2 successfully forges a multi-signature and $\Pr[E_2|E_1] \geq \varepsilon$.

E_3: There exists at least one record of the non-target identity in a successful forgery case.

As described in Theorem 2, we can obtain

$$\begin{aligned} &\Pr\left[E_1\right] \geq \left(1 - \frac{1}{q_{\mathrm{s}}}\right)^{q_{\mathrm{s}}} \geq \frac{1}{\mathrm{e}q_{\mathrm{s}}}, \\ &\Pr\left[E_3|E_1 \cap E_2\right] \geq 1/n. \end{aligned}$$

If A_2 can break the UF-CMA-II security of NC-CLBMS with advantage ε, C can solve the CDH problem with advantage ε', where

$$\begin{aligned} \varepsilon' &= \Pr\left[\mathrm{E}_1 \cap \mathrm{E}_2 \cap \mathrm{E}_3\right] \\ &= \Pr\left[\mathrm{E}_1\right]\Pr\left[\mathrm{E}_2|\mathrm{E}_1\right]\Pr\left[\mathrm{E}_3|\mathrm{E}_1 \cap \mathrm{E}_2\right] \\ &\geq \frac{\varepsilon}{n\mathrm{e}q_{\mathrm{s}}}. \end{aligned}$$

6.2 Anti-pollution attacks

Theorem 3 NC-CLBMS can resist the pollution attacks in a multi-source network coding environment.

Proof In a multi-source multi-signature, there are two kinds of pollution attacks: one is to generate the forged message; the other is to obtain a forged signature based on the combination result intercepted by the attacker.

In the first attack, the attacker pollutes the source node, the intermediate node receives the message to directly forge the message to transfer in the network, and the intermediate node combines the polluted message: $\boldsymbol{w}' = \sum_{i=1}^{m} \boldsymbol{\beta}_i \boldsymbol{v}_i$. However, the attack is invalid because the attacker cannot effectively sign the tainted message without a private key of signer.

In the second attack, the attacker generates a forged signature; that is, the attacker hopes to forge relevant signature $\sigma_i' = \left(x_i' + r_i' + s'\psi_i\right)\gamma_i$ based on the signature $\sigma_i = \left(x_i + r_i + s\psi_i\right)\gamma_i$. For $\sigma_k = \left(x_k + r_k + s\psi_k\right)\gamma_k$, we can obtain

$$\begin{aligned}\left(x_k + r_k + s\psi_k\right)\gamma_k &= \left(x_k' + r_k' + s'\psi_k\right)\gamma_k\\ \Rightarrow x_k' &= x_k + r_k - r_k' + (s - s')\psi_k\\ \Rightarrow r_k' &= x_k - x_k' + r_k + (s - s')\psi_k\\ \Rightarrow s' &= \frac{x_k - x_k' + r_k - r_k' + s\psi_k}{\psi_k}.\end{aligned}$$

Obtaining x_k', r_k', and s' from the above equations is equivalent to solving the elliptic curve discrete logarithm (ECDL) problem.

7 Efficiency analysis

In terms of computational complexity, we compare NC-CLBMS with ZX (Zhou and Xu, 2016), YG (Yu and Gao, 2019), WZZ (Wang L et al., 2019), YL (Yu and Li, 2019), and YW (Yu and Wang, 2021). The test platform is as follows:

Operating system: Win10, 64-bit; CPU: Intel® Core™ i5-8250U, 1.8 GHz; memory: 4 GB; operating platform: Matlab2016a.

Table 2 shows each cryptographic operation time. Table 3 shows the signature time and verification time of several schemes.

Table 2 Operation time of cryptographic algorithms

Notation	Meaning
C_{me}	Time of running an exponential operation: 6.85 ms
C_{mul}	Time of running a scalar multiplication: 0.75 ms
C_{mtp}	Time of running a hash operation: 19.60 ms
C_{par}	Time of running a bilinear operation: 22.73 ms
C_{ex}	Time of running a modular exponentiation operation: 34.20 ms

Comparison of the signature time and verification time of several schemes is as follows. The operation time of the hash function is ignored in the analysis. NC-CLBMS, YL, and YW are based on the certificateless cryptosystem, and they can resist the pollution attacks and forgery attacks; ZX, YG, and WZZ are not based on the certificateless cryptosystem and cannot resist the forgery attacks.

As shown in Table 3, there are many exponential operations in signature and verification algorithms of ZX, YG, WZZ, YL, and YW; thus, they result in the consumption of storage resources; NC-CLBMS uses only $2C_{par}$, but YL uses $3C_{par}$; in addition, NC-CLBMS does not need any modular exponentiation operation. Simulation curves of the signature time are shown in Fig. 2. Simulation curves of the verification time are shown in Fig. 3. Known from simulation results in Figs. 2 and 3, with the increase of the message vector dimension, the growth rate of

Table 3 Comparison of computational efficiency of several schemes

Scheme	Signature time (ms)	Verification time (ms)
ZX	$C_{mtp}+(m+n)C_{mul}+3nC_{me}+6C_{ex}$	$C_{mtp}+3nC_{me}+3C_{ex}$
YG	$2C_{par}+(2n+m)C_{mul}+2nC_{me}$	$2C_{mtp}+(2n+m)C_{mul}+nC_{me}+3C_{par}$
WZZ	$C_{par}+(m+n)C_{mul}+2nC_{me}+5C_{ex}$	$C_{mtp}+3nC_{me}+C_{ex}$
YL	$3C_{par}+(3n+m)C_{mul}+nC_{me}$	$3C_{mtp}+(4n+m)C_{mul}+nC_{me}$
YW	$2C_{par}+(m+n)C_{mul}+nC_{me}$	$3C_{mtp}+3(m+n)C_{mul}+nC_{me}+2C_{par}$
NC-CLBMS	$2C_{par}+(2(n+m)+1)C_{mul}$	$2C_{mtp}+(2(n+m)+1)C_{mul}+3C_{par}$

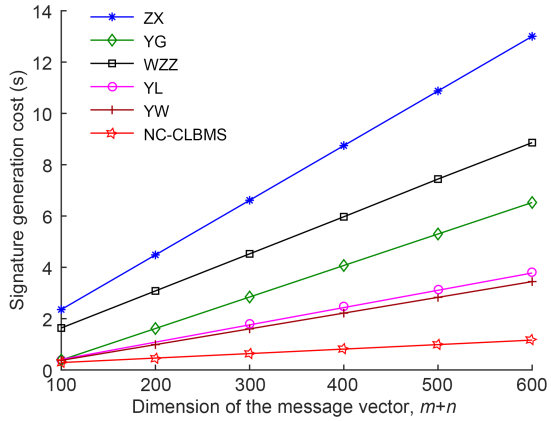

Fig. 2 Signature time of NC-CLBMS and other schemes

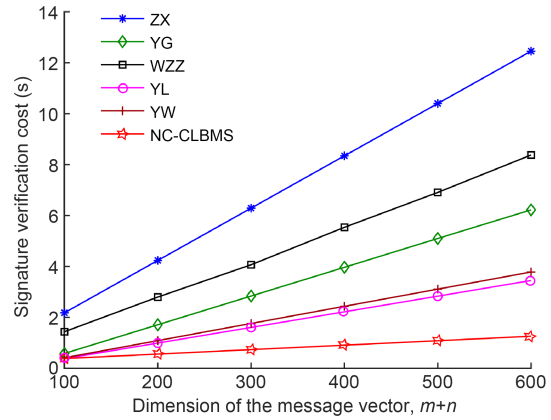

Fig. 3 Verification time of NC-CLBMS and other schemes

NC-CLBMS is lower than those of other schemes. Hence, NC-CLBMS is more efficient than other schemes.

8 Summary

NC-CLBMS has the advantages of anti-pollution and anti-forgery, and its security relies on the hardness of the elliptic curve discrete logarithm and computational Diffie-Hellman problems.

The homomorphic hash function enables the node to ensure that the $\boldsymbol{c}$ signature and verification processes are correct. NC-CLBMS is different from traditional signatures, which cannot be directly used in network coding. NC-CLBMS avoids the certificate use and key escrow, and its signature length is fixed. NC-CLBMS has strong robustness and low computation complexity, so it is suitable for applications in unmanned aerial vehicle (UAV) communication networks, wireless sensor networks, fifth-generation wireless networks, Internet of Things, and wireless mesh networks.

Contributors

Huifang YU devised the NC-CLBMS method and analyzed its security. Zhewei QI processed the data and analyzed the efficiency. Huifang YU drafted the paper. Huifang YU and Zhewei QI revised and finalized this paper.

Compliance with ethics guidelines

Huifang YU and Zhewei QI declare that they have no conflict of interest.

References

Ahlswede R, Cai N, Li SYR, et al., 2000. Network information flow. *IEEE Trans Inform Theory*, 46(4):1204-1216. https://doi.org/10.1109/18.850663

Al-Kofahi OM, Kamal AE, 2009. Network coding-based protection of many-to-one wireless flows. *IEEE J Sel Areas Commun*, 27(5):797-813. https://doi.org/10.1109/JSAC.2009.090619

Li P, Guo S, Yu S, et al., 2012. CodePipe: an opportunistic feeding and routing protocol for reliable multicast with pipelined network coding. Proc IEEE INFOCOM, p.100-108. https://doi.org/10.1109/INFCOM.2012.6195456

Li SH, Mei ZH, 2016. Homomorphic signature scheme for network coding against inter-generation pollution attacks. *Comput Technol Dev*, 26(10):73-76 (in Chinese). https://doi.org/10.3969/j.issn.1673-629X.2016.10.016

Li ZP, Li BC, Lau LC, 2009. A constant bound on throughput improvement of multicast network coding in undirected networks. *IEEE Trans Inform Theory*, 55(3):1016-1026. https://doi.org/10.1109/TIT.2008.2011516

Niu SF, Li WT, Wang CF, 2020. New efficient certificateless broadcast multi-signature scheme. *Appl Res Comput*, 37(8): 2464-2467 (in Chinese). https://doi.org/10.19734/j.issn.1001-3695.2019.02.0070

Papailiopoulos DS, Luo JQ, Dimakis AG, et al., 2012. Simple regenerating codes: network coding for cloud storage. Proc IEEE INFOCOM, p.2801-2805. https://doi.org/10.1109/INFCOM.2012.6195703

Peng TL, Shang T, Liu JW, 2015. Signature scheme for network coding against inter-generation pollution attacks. *J Beijing Univ Aeronaut Astronaut*, 41(4):721-726 (in Chinese). https://doi.org/10.13700/j.bh.1001-5965.2014.0478

Wang HP, Mei ZH, 2016. A scheme against pollution attacks based on secure network coding. *Comput Technol Dev*, 26(7):94-99 (in Chinese).

https://doi.org/10.3969/j.issn.1673-629x.2016.07.020

Wang L, Zhang Z, Zhang H, et al., 2019. A RSA-based secure network coding scheme against multiple attacks. *Comput Eng*, 45(11):166-171 (in Chinese). https://doi.org/10.19678/j.issn.1000-3428.0052845

Xu CD, Wang HQ, 2021. Sequential multi-signature scheme based on blockchain. *J Nanjing Univ Posts Telecommun (Nat Sci Ed)*, 41(2):85-94 (in Chinese). https://doi.org/10.14132/j.cnki.1673-5439.2021.02.010

Xu J, Liu YT, Xia GY, et al., 2016. Network coding based topology inference: a survey. *Comput Sci*, 43(S1):242-248, 264 (in Chinese).

Yu HF, Gao XZ, 2019. Homomorphic ring signature scheme technology for multi-source network coding. *Netinfo Secur*, (2):36-42 (in Chinese). https://doi.org/10.3969/j.issn.1671-1122.2019.02.005

Yu HF, Li W, 2019. Homomorphic signature schemes for single-source and multi-source network coding. *J Commun*, 40(11):112-121 (in Chinese). https://doi.org/10.11959/j.issn.1000-436x.2019219

Yu HF, Li W, 2020. A certificateless signature for multi-source network coding. *J Inform Secur Appl*, 55:102655. https://doi.org/10.1016/j.jisa.2020.102655

Yu HF, Wang WK, 2021. Certificateless network coding ring signature scheme. *Secur Commun Netw*, 2021:8029644. https://doi.org/10.1155/2021/8029644

Zhou ZB, Xu L, 2016. Pollution-resistant network coding scheme based on digital signature. *Comput Syst Appl*, 25(6):185-190 (in Chinese). https://doi.org/10.15888/j.cnki.csa.005192