Distributed and Autonomous Group Management Supporting Group Fusion for UAVs

Fengqun Wang, Manting Gan, Hong Zhong, Qingyang Zhang, Jie Cui, Debiao He

***Abstract*—With increasingly complex tasks, cooperation among multiple unmanned aerial vehicle (UAV) groups has become more significant. However, in complex operational environments, UAVs may operate outside the communication coverage of the trusted authority (TA), making continuous online TA services unavailable. Under such circumstances, most existing group management methods have difficulty achieving group fusion and cannot flexibly update post-fusion member certificates. Therefore, we propose an autonomous UAV group management scheme based on mobile proactive secret sharing. First, the scheme achieves autonomous group fusion by updating the subsecrets of UAVs. Second, without the participation of a TA, the scheme supports the dynamic self-updating of certificates, ensuring secure communication in the new group and continuous availability of certificates. Security proofs and analyses show that the proposed scheme is secure under the random oracle model and can resist several common attacks. The experimental results demonstrate that the proposed scheme outperforms related schemes in computational performance and is suitable for secure and efficient UAV group management scenarios.**

***Index Terms*—Unmanned Aerial Vehicle (UAV), Authentication, Mobile Proactive Secret Sharing, Group Management.**

I. Introduction

WITH the recent rapid developments in unmanned aerial vehicle (UAV) technology, UAVs have been widely used in areas such as remote sensing [1], road transportation [2], and emergency rescue [3], [4], significantly improving the quality of life. The total value of the global UAV market is estimated to reach 72.1 billion dollars by 2030 [5]. Individual UAVs have limited resources and are easily restricted by factors such as operational range, making it difficult to conduct tasks independently and efficiently in complex scenarios. In UAV groups, several individual UAVs are organized to collaborate and achieve orderly distributed operations [6]. Compared with traditional manual task execution, UAV groups can reduce risks and costs while enhancing system robustness and reliability [7].

Cooperation between UAVs is generally used in outdoor or denied environments [8], [9]. A typical scenario involves a group of UAVs managed by different mobile stations, as illustrated in Fig. 1. Once successfully networking, the UAV group members aggregate at a certain location to cooperate

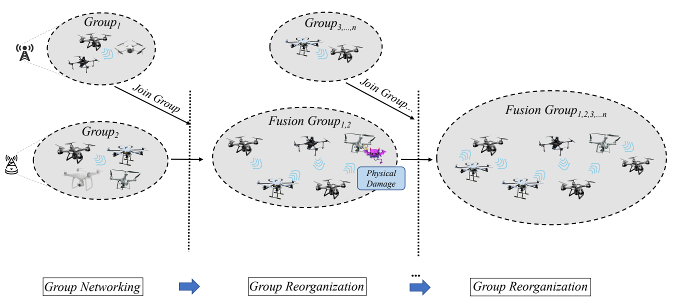


Fig. 1. Offline autonomous management of collaborative UAV groups in a denied environment.

on a task [7]. After completing the tasks in specific areas, these collaborative groups can continuously merge with other groups to cover broader regions and provide more resources. During mission execution, UAVs may operate beyond the communication range of a trusted authority (TA), making it impossible to rely on continuous online services provided by the TA. In addition, wireless channels are highly open and susceptible to network attacks [10]. Therefore, group collaboration presents several challenges for the security management of UAV groups, particularly in terms of reliable group fusion and secure certificate updating in offline TA scenarios.

Most existing group management schemes rely on the continuous online availability of TA for group fusion and certificate updating. These schemes can be divided into three main categories: centralized, blockchain-based, and multicenter. In centralized schemes [11]–[13], new members must apply for a certificate from the TA before joining and authenticating themselves with the group members. In blockchain-based schemes [14]–[16], when a new member joins the group, the TA must store the identity information on the blockchain. The group members then retrieve the new member's public key from the blockchain and verify the member's legitimacy. However, blockchain is expensive to maintain, and membership is managed by the TA. In multicenter schemes [17]–[19], the TA initially divides the entire system into multiple subgroups and selects a cluster head for each subgroup. These cluster heads are responsible for managing the identity information of group members and distributing certificates to new members. However, if a cluster head encounters a single point of failure, it cannot provide certification to new members. Therefore, the three aforementioned methods are difficult to implement for

F. Wang, M. Gan, H. Zhong, Q. Zhang and J. Cui are with the Key Laboratory of Intelligent Computing and Signal Processing of Ministry of Education, School of Computer Science and Technology, Anhui University, Hefei 230039, China, and the Anhui Engineering Laboratory of IoT Security Technologies, Anhui University, Hefei 230039, China (e-mail: zhongh@ahu.edu.cn).

D. He is with the School of Cyber Science and Engineering, Wuhan University, Wuhan 430072, China and the Shanghai Key Laboratory of Privacy Preserving Computation, MatrixElements Technologies, Shanghai 201204, China (email: hedebiao@163.com).

UAV group fusion when the TA is offline.

For certificate updates, most existing schemes renew public or pseudonymous certificates through a TA [14], [16], [20], [21], requiring the TA to provide online assistance when updating certificates. For example, in the scheme [20] proposed by Zhang *et al.*, the TA must compute and update the domain key parameter when a member joins a group. All group members calculate the domain key based on this domain key parameter to ensure the security of subsequent intragroup communications. In the scheme [21] proposed by Liu *et al.*, the TA must distribute pseudonym certificates to the new members. To ensure the long-term communication security of the entire group, the TA must update the pseudonym certificates of the original group members. These certificates are uploaded to the blockchain by the TA. However, when the TA is offline, the above schemes cannot ensure the continuous availability of certificates after group fusion.

A. Our Motivation

As the scope of tasks performed by UAV groups continues to expand, the signal coverage of trusted authorities is limited, making it impossible to provide continuous online authentication and management services for UAVs executing remote missions. Consequently, it is difficult to establish or update keys in a timely manner. Therefore, the research motivation of this paper is to design a scheme that can support flexible fusion of UAV groups and autonomous certificate update in scenarios where trusted authorities are offline, thereby realizing secure communication and autonomous management of UAV groups.

B. Our Contribution

Driven by the above research motivation, we propose a secure and efficient autonomous management scheme for UAV groups. Specifically, we implement the update of UAVs' subsecrets and the fusion of two groups based on mobile proactive secret sharing technology. New group members formed after fusion can autonomously update their original certificates, ensuring their identity's legitimacy without a TA's assistance. To the best of our knowledge, this is the first scheme to use mobile proactive secret sharing technology to address the secure management of UAV groups when the TA is offline. The contributions of this study are as follows:

- A secure and distributed management scheme for UAV groups is proposed to achieve group fusion when the TA is offline. This scheme updates the UAVs' subsecrets based on mobile proactive secret sharing and uses the homomorphism of secret sharing to achieve group fusion. Additionally, to ensure communication security during fusion, a threshold signature is used for batch authentication of UAVs' identities before fusion.
- An efficient certificate self-update method is designed to allow the continuous validity of the certificate. Each UAV only performs a few lightweight operations to update its certificate. Using the certificate chain principle, trust propagation ensures the legitimacy of the origin of the UAV group.
- A formal security proof using the real-or-random (ROR) model and informal security analysis are conducted to verify the correctness and security of the proposed scheme. The experimental results demonstrate that our scheme achieves better performance in terms of computational cost and enhanced security functionalities.

C. Organization of the Rest Paper

The remainder of this paper is organized as follows. In Section II, related studies on group management schemes are reviewed. Section III introduces some prerequisites relevant to the proposed scheme. Section IV describes the system model and the security objectives of the design. The details of the proposed scheme are presented in Section V. Section VI provides a security analysis for the proposed scheme. Section VII reports on experiment construction and the results. Finally, Section VIII concludes the paper by providing an outlook on future research.

II. Related Works

In this section, we will analyze the work related to group management regarding group fusion and certificate update, respectively.

A. Group Fusion

As cooperation among groups increases, dynamic group management has attracted widespread attention. A series of works has been proposed to achieve secure and efficient member joining.

Khan *et al.* [11] proposed an authentication and key agreement protocol based on hyperelliptic curve cryptography (HECC), which supports the dynamic joining of UAVs. However, the UAV joining phase has public-private key pairs and certificates issued by a ground control unit (GCU). This phase does not demonstrate a detailed fusion process.

Tan *et al.* [15] proposed a lightweight authentication scheme for industrial UAVs. They designed an identity management mechanism by combining it with blockchain technology. However, every two UAVs must obtain each other's public key from the blockchain before establishing a session key. When a new member joins, it needs to rely on a ground control station (GCS) to store the member's public key in the blockchain.

Bansal *et al.* [13] designed an authentication protocol suitable for UAV groups using PUF. This protocol realizes the identity authentication between UAVs and the base station through Physical Unclonable Function (PUF), and uses the minimum spanning tree algorithm to ensure the dynamic topology of the UAV group. However, both the authentication of UAVs and the establishment of session keys require the real-time participation of the base station (BS).

Tan *et al.* [17] proposed a blockchain-based heterogeneous flight autonomous networking management scheme. The scheme divides UAVs into subgroups with a cluster header in each subgroup. The scheme claims UAVs can autonomously distribute group keys and update their public-private key pairs. However, when a UAV joins a new group, the cluster header needs to perform these operations.

TABLE I
EXISTING GROUP MANAGEMENT SCHEMES: A COMPARATIVE SUMMARY

Schemes	Type	Technologies applied	Advantages/Descriptions	Drawbacks/Limitations
Khan *et al.* [11]	Authentication / Join	* HECC	* Shorter key length * Against secret key leakage	* Require a GCU * Joining is inflexible
Tan *et al.* [15]	Authentication / Join	* Blockchain * Smart contracts	* Distributed and lightweight * Tolerate UAV damage	* Require a GCS * High update cost
Bansal *et al.* [13]	Authentication / Join	* PUF * Spanning tree	* Ensure physical security * Support dynamic topologies	* Require a real-time BS * Joining is relatively complex
Tan *et al.* [17]	Key management / Join	* Blockchain	* Autonomously distribute keys * Migrate between clusters	* The cluster head UAV has a single point of failure
Tian *et al.* [18]	Key management / Join	* Blockchain	* Propose an effective and efficient key management scheme	* Members' states require sensors to write to the blockchain
Liu *et al.* [21]	Pseudonym update	* Blockchain * CRT	* Manage pseudonyms and group keys in a decentralized manner	* Update depends on OAs * High consensus cost
Ma *et al.* [25]	Key management / Update	* Binary polynomials * Blockchain	* Automatically update and revoke the user's public key	* Rely on a VSP to invoke the smart contract
Tanveer *et al.* [26]	Key management / Update	* ECC	* Support the revocation and update of UAVs' identities	* High computational and storage cost on UAVs
Zhang *et al.* [20]	Group key update	* TPD * CRT	* Dynamically assist the TAs while generating new group keys	* Require an ideal TPD * Doesn't support multi-groups
Zhang *et al.* [5]	Group key update	* TPD * Shamir's secret sharing	* Efficient group key update * High fault tolerance	* Require an ideal TPD * Doesn't support multi-groups
Our proposed	Group management / Certificate update	* MPSS * Threshold signature	* Autonomous group fusion * Efficient certificate update	* Communication overhead increases with larger group sizes

Tian *et al.* [18] proposed a trusted and secure blockchain-based key management scheme. However, when a member joins a new group, sensors within this group are required to notify other sensors of the change in the member's state. At the same time, the sensors within this group modify the list of legitimate identities and write them to the blockchain.

Yıldız *et al.* [22] proposed a lightweight group authentication and key distribution protocol based on PUF. However, when a new member joins the group, a new group key needs to be calculated based on the information broadcast by the server.

Roychoudhury *et al.* [23] proposed a mutual authentication and key agreement scheme for packet Device-to-Device communication. However, members need to be authenticated by a TA when they apply to join the group. After authentication, members update the key parameters based on the information sent by the TA.

Lee *et al.* [24] proposed an anonymous dynamic group authentication key agreement based on PUF. When a new member joins, other members in the group do not have to perform all the steps in the authentication and group key negotiation phases. However, the scheme requires a TA to assist in the mutual authentication between the new member and the group members.

Therefore, most group management schemes rely on trusted institutions when members join, and lack a secure and efficient group fusion mechanism. We list several differences of the existing group management schemes in Table I.

B. Certificate Update

Some schemes have been proposed to solve the problem of authenticating the legitimacy of group membership after fusion.

Xu *et al.* [16] proposed a blockchain-based anonymous authentication and dynamic group key agreement scheme, which achieves effective updating of pseudonym certificates. However, when a new member joins the group, it needs to rely on the auxiliary parameter provided by the TA to update the pseudonym certificate.

Cheng *et al.* [14] proposed a conditional privacy-preserving multi-domain authentication and pseudonym management scheme. The scheme efficiently manages pseudonym certificates based on dynamic and sparse Merkle trees.

Liu *et al.* [21] proposed a decentralized group key management scheme. The scheme uses a blockchain consensus mechanism to manage the creation and update of membership keys. However, similar to the scheme proposed by Cheng *et al.* [14], the updated pseudonym certificates need to be uploaded to the blockchain by a TA in scheme [21].

Ma *et al.* [25] proposed a lightweight authentication and key management mechanism combined with binary polynomials to achieve the update of public certificates. However, certificates must be registered, updated, and revoked by invoking smart contracts through a TA.

Mansour *et al.* [12] proposed a group key management protocol based on the Chinese Remainder Theorem (CRT). The scheme decouples the initialization and group key computation phases and improves the efficiency of group key computation.

Tanveer et al. [26] proposed an authentication key management protocol suitable for IoD. This scheme employs authenticated encryption primitives, Elliptic Curve Cryptography (ECC), and hash functions to execute the protocol process. After the user's identity is authenticated, a session key can be established with a specific UAV. Meanwhile, the scheme supports the revocation and update of UAV identities. However, the computational operations are relatively complex, which brings substantial consumption to the computing and storage resources of UAVs.

Zhang *et al.* [20] proposed an authentication and key agreement protocol with conditional privacy preservation in the vehicle networking environment. This scheme does not require an ideal tamper-proof device (TPD) and avoids the whole system being at fault due to the TPD of a particular vehicle being compromised. However, the above two schemes still require the participation of a TA in the computation and updating of group public key certificates when vehicles are added to the batch.

Zhang *et al.* [5] proposed a group key negotiation scheme for threshold authentication. The scheme supports the updating of keys and certificates. Each UAV deploys a TPD. During the initialization phase, the TA stores the key update seed on the TPDs of each UAV and updates the subkey periodically. However, the TPDs are ideal. If a UAV is compromised, the key to the whole system will be leaked. In addition, the scheme application scenario is a single group and does not consider the management of multiple groups.

Therefore, the above schemes need to rely on TAs to update certificates. It is difficult to achieve autonomous update of keys and continuous availability of certificates in dynamic groups. Additionally, Table I lists several differences we have noted regarding existing certificate update schemes.

In summary, most group management schemes have proposed some constructive approaches to meet the needs of different scenarios. However, these schemes still require TAs to participate in group management online. Therefore, it is necessary to design a group management scheme that supports flexible group fusion and efficient certificate self-update for scenarios where TAs are offline.

III. Prelimilary Knowledge

A. Mobile Proactive Secret Sharing

David Schultz, Barbara Liskov, and Moses Liskov proposed the Mobile Proactive Secret Sharing (MPSS) scheme [27] in 2010 for the first time. Building upon proactive secret sharing [28], mobile proactive secret sharing provides mobility, allowing the group of nodes holding secret shares to change. This scheme combines share update and share recovery into a single step, enabling membership changes within the group. Nodes regularly execute a switching protocol, generating a new set of shares for different node groups while keeping the secret unchanged, and share these new shares with potentially different new node groups. The algorithm is briefly outlined below.

- Initialization phase: Assuming the secret distributor D uses randomly chosen Shamir's polynomial $P(x) = s + a_1x + a_2x^2 + ... + a_{t-1}x^{t-1}$ to split the secret into many fragments $x_1, x_2, ..., x_n$, and distributes them to n members $p_1, p_2, ..., p_n$ respectively. Each member holds a secret fragment, also known as a secret share or sub-secret. Here, $a_1, a_2, ..., a_{t-1}$ are random numbers, t is the threshold, and each participant p_i holds the share $x_i = P(i)$, where i is a unique identifier for each participant.
- Updating phase: First, the old node i selects a random polynomial $Q_i(x)$ for updating shares, such that $Q_i(0) = 0$. Then, for each new node k, a random polynomial $R_{i,k}(x)$ is chosen by i for obfuscation, satisfying $R_{i,k}(k) = 0$. The old node i secretly sends the value of $P(i) + (Q_i(k) + R_{i,k}(k))$ to the new node k. Once at least t legitimate values are received, the new node k can interpolate to obtain the polynomial $P(x) + (Q_i(x) + R_{i,k}(x))$. Computing the value of this polynomial at $x = k$ yields its new share $x_{k_{new}}$. Since $R_{i,k}(x)$ is random everywhere except at $x = k$, the share information provided by the polynomial is only effective at $x = k$. Apart from the new node k, no one else can recover the correct share $x_{k_{new}}$.

B. Threshold Signature

The (t, n) threshold signature refers to a signature scheme where a group of n members collectively form a signing group. Any subset of the group consisting of at least t legitimate members can collectively use the group's private key to generate a signature. Anyone can verify the signature using the group's public key. Here, t is the threshold value, and n is the total number of members. Any $t - 1$ or fewer members in the group cannot collectively sign on behalf of the group. No member can impersonate another member for signing. Threshold signature is a type of distributed multi-party signature protocol, which includes three algorithms: distributed key generation, signature generation, and signature verification.

- Distributed key generation: A TA randomly selects a (t, n) Shamir's polynomial and distributes the secret shares to each member i.
- Signature generation: Each member i generates an individual signature s_i for the message m using their secret share as a private key. Then member i sends s_i to the signature aggregator. The signature aggregator verifies the legitimacy of individual signatures. If all of them are legitimate, the signature aggregator computes the aggregate signature $s = \sum_{i=1}^{t} s_i$ and sends the signature message (m, s) to the verifier.
- Signature verification: After receiving the message, the verifier verifies the validity of the aggregate signature s with the public key corresponding to the secret in Shamir's polynomial. If the verification succeeds, the threshold signature is deemed legitimate.

IV. System Model

This section introduces the proposed scheme's system model and shows the goals to be achieved.

A. System Model

As shown in Fig. 2, our scheme consists of three entities: trusted authority (TA), mobile station (MS), and unmanned aerial vehicles (UAVs).

1) **TA**: TA is a trusted authority that all participating entities must first register with it. TA generates the system's public parameters, creates public-private key pairs, and issues certificates for all participating entities.

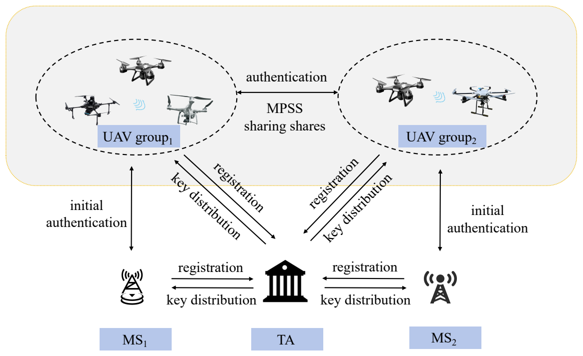


Fig. 2. System model.

In this scheme, TA only participates during the initialization and registration phases. Subsequent processes are conducted offline.

2) **MS**: MS is a mobile station with sufficient computing and storage resources. When a UAV is successfully deployed in an area managed by this MS, the MS is responsible for verifying the UAV's identity and distributing a secret share to it. Similar to TA, the MS only participates in the initialization authentication phase. Once the UAV receives its share, the MS goes offline and does not participate in subsequent processes.
3) **UAVs**: UAVs are the primary communication entities with limited computing and storage resources. Each UAV in the group has equal status, and our proposed scheme has no cluster headers. Considering the existence of flight loss, some UAVs in the group may experience physical damage. The authentication, fusion, and certificate update of the group should be done without the involvement of TA and MS.

B. Attack Model

This scheme assumes entities communicate over an insecure physical open channel, and adversary $\mathcal{A}$ has both active and passive attack capabilities, such as impersonation attacks, tampering attacks, and man-in-the-middle attacks. Adversary $\mathcal{A}$ is a legitimate network user with the ability to access any message transmitted in the network, interact with other users, receive messages sent by principals, and impersonate principals to send messages to other principals. However, adversary $\mathcal{A}$ finds it difficult to correctly guess random numbers in a sufficiently large finite field and to solve certain mathematical problems within polynomial time, such as the factorization problem and the discrete logarithm problem. Without the correct key, adversary $\mathcal{A}$ cannot construct the correct ciphertext from a given plaintext or recover the correct plaintext from a given ciphertext.

C. Design Goals

We aim to achieve the following security objectives in the proposed scheme:

1) **Mutual authentication**: To prevent malicious entities in the network, entities participating in communication must be able to authenticate each other's identities before transmitting sensitive information.
2) **Session key negotiation**: After the authentication of the entities has been passed, a session key needs to be negotiated to encrypt the communication.
3) **Fairness**: Fairness refers to the property that no particular group member has additional influence or advantage over the group management process. To ensure fairness, all group members should be considered equal.
4) **High fault tolerance**: UAVs may experience physical damage during flight. Even if some UAVs cannot participate in the scheme, it will not affect the group management process, e.g., group authentication, group fusion, and certificate update.
5) **Autonomous fusion**: Most group management schemes require TA to participate online. Due to the instability of wireless channels, TAs may go offline and cannot provide continuous online services for UAV groups. Therefore, autonomous fusion of UAV groups needs to be implemented.
6) **Certificate self-update**: After group fusion, certificates previously issued by MS (i.e., threshold signatures generated by UAVs) are insufficient to verify the identity of UAVs in the new group. Therefore, UAVs need to update their certificates autonomously.
7) **Confidentiality**: Confidentiality requires that adversaries never know the secret. Even if they are aware of the old share of the UAV for a certain period, they cannot obtain information about the shared share for the next period.
8) **Resisting some attacks**: To ensure the security of the entire network, the designed scheme should be resilient to some common attacks, such as impersonation attack, tampering attack, and man-in-the-middle attack.

TABLE II
EXPLANATIONS ON NOTATIONS IN THE SCHEME.

Notations	Descriptions
tsk, TPK	Master key of the system
UAV_i, MS_j	i-th UAV, j-th MS
UID_i, MID_j	ID of i-th UAV, and j-th MS
usk_i, upk_i	Public-private key pair of i-th UAV
msk_j, mpk_j	Public-private key pair of j-th MS
mk_j, MK_j	Secret and corresponding public key of j-th group
zsk_j	Secret key of j-th group
$key_{i,j}$	Session key between i-th and j-th UAV
$UCert_i, MCert_j$	Certificate of i-th UAV, and j-th MS
$HMAC(m, sk)$	Hash code generated by message m and key sk
t	The threshold of $group_1$ and $group_2$
t_{new}	The threshold of the new group after fusion
$P(\cdot), Q(\cdot), R(\cdot)$	Shamir's polynomial of degree $t-1$
x_i	Share corresponding to i-th UAV
s_i	Signature of i-th UAV
s	Aggregation signature of a group
L_i	Lagrange interpolation coefficients
$V_{i,j}, V_{j,k}$	Partial share of UAVs
$Enc_{key}(\cdot), Dec_{key}(\cdot)$	Encryption and decryption operations

V. Proposed Scheme

This section focuses on the proposed distributed management scheme for UAV groups based on mobile proactive secret sharing. The scheme is divided into five parts: system initialization, registration, initial authentication, group authentication, and group management.

In the system initialization phase, the TA generates and publishes the system's public parameters. Both the MS and UAVs participating in communication register with the TA during the registration phase and obtain information such as keys and certificates. In the initial authentication phase, UAVs join the MS and authenticate each other's identities. Due to the limited signal coverage of each MS, the signal coverage area is defined as a management domain. The UAVs within each MS's management domain form a group, and the MS distributes initial secret shares to each UAV in the domain. In the group authentication phase, UAVs, as a group, authenticate the identities of UAVs from another group. By the end of this phase, each pair of UAVs establishes a temporary session key to facilitate encryption and decryption of shared shares in the next phase. The group management phase mainly includes group fusion and certificate update. UAVs from the two groups calculate the new shares after fusion by sharing shares, and at the same time, update certificates autonomously.

To better understand the scheme, we summarize some important notations in Table II. Next, we detail the specific steps of each phase.

A. *System Initialization*

TA has sufficient computing and storage abilities, which is responsible for generating system parameters.

1) TA chooses an additive group G of order q, consisting of points on the elliptic curve and at infinity $\mathcal{O}$. q is a large prime number, P is a generator of G. In addition, TA selects 3 hash functions, namely h_1, h_2 and h_3.
2) TA generates the private key tsk, then calculates the corresponding public key $TPK = tsk \cdot P$.
3) Finally, TA publishes the system public parameters $Para = \{G, P, q, h_1, h_2, h_3, TPK\}$ in the network.

B. *Registration*

At this phase, entities participating in communication must register with TA to obtain the corresponding key materials. The registration process needs to be conducted over a secure channel. Registration is divided into two phases, namely, UAVs registered with TA and MSs registered with TA. The entity interaction in the registration phase is shown in Fig. 3.

1) UAVs registered with TA: TA selects a unique real identity UID_i, a random number a_i and a private key usk_i for each UAV_i, calculates $A_i = a_i \cdot P$, public key $upk_i = usk_i \cdot P$, certificate $UCert_i = a_i + h_1(UID_i \parallel upk_i \parallel A_i) \cdot tsk$. Then, TA sends $\{UID_i, usk_i, upk_i, UCert_i, A_i\}$ to UAV_i through the secure channel.
2) MSs registered with TA: TA selects a unique real identity MID_j, two random numbers b_j^1, b_j^2, and a private key msk_j for each MS_j. Then TA computes the group secret $mk_j = b_j^2 + tsk$, $MK_j = mk_j \cdot P$, $B_j^1 = b_j^1 \cdot P$, $B_j^2 = b_j^2 \cdot P$, public key $mpk_j = msk_j \cdot P$, certificate $MCert_j = b_j^1 + h_2\left(MID_j \parallel mpk_j \parallel B_j^1 \parallel MK_j\right) \cdot tsk$, and sends $\left\{MID_j, msk_j, mpk_j, MCert_j, mk_j, MK_j, B_j^1, B_j^2\right\}$ to MS_j through a secure channel.

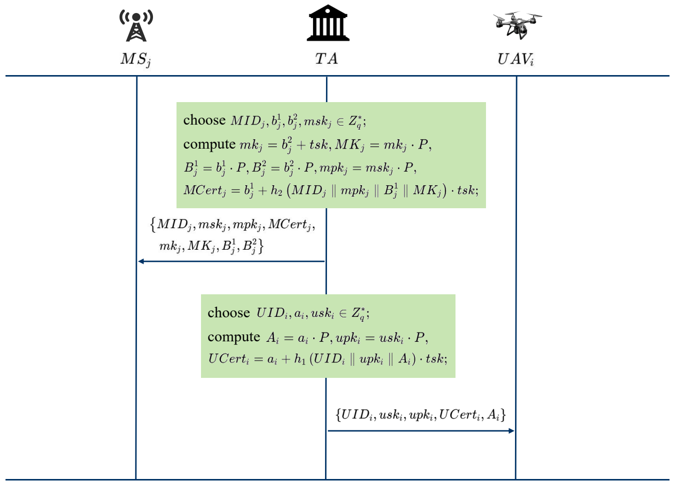

MS_j
TA
UAV_i
choose $MID_j, b_j^1, b_j^2, msk_j \in Z_q^*$;
compute $mk_j = b_j^2 + tsk, MK_j = mk_j \cdot P$,
$B_j^1 = b_j^1 \cdot P, B_j^2 = b_j^2 \cdot P, mpk_j = msk_j \cdot P$,
$MCert_j = b_j^1 + h_2\left(MID_j \parallel mpk_j \parallel B_j^1 \parallel MK_j\right) \cdot tsk$;
$\{MID_j, msk_j, mpk_j, MCert_j, mk_j, MK_j, B_j^1, B_j^2\}$
choose $UID_i, a_i, usk_i \in Z_q^*$;
compute $A_i = a_i \cdot P, upk_i = usk_i \cdot P$,
$UCert_i = a_i + h_1(UID_i \parallel upk_i \parallel A_i) \cdot tsk$;
$\{UID_i, usk_i, upk_i, UCert_i, A_i\}$

Fig. 3. Flowchart of the registration phase.

C. *Initial Authentication*

Mutual authentication must be completed when UAV_i joins the MS_j. MS_j sends the secret share and group's private key encrypted by the negotiated session key to UAV_i. All the UAVs that join MS_j form a group. The entity interaction in this phase is shown in Fig. 4.

1) Intra-group session key negotiation:

1) UAV_i sends $\{UID_i, upk_i, UCert_i, A_i\}$ to MS_j.
2) MS_j first checks if $UCert_i \cdot P = A_i + h_1(UID_i \parallel upk_i \parallel A_i) \cdot TPK$ holds. If it does, it computes the session key $sk_{j,i} = msk_j \cdot upk_i$, and sends $\{MID_j, mpk_j, MCert_j, B_j^1, MK_j\}$ to UAV_i.
3) UAV_i first checks whether $MCert_j \cdot P = B_j^1 + h_2\left(MID_j \parallel mpk_j \parallel B_j^1 \parallel MK_j\right) \cdot TPK$ holds. If it does, $MCert_j$ is stored, and the session key $sk_{i,j} = usk_i \cdot mpk_j$ is calculated.

2) Distribution of secret shares:

1) MS_j chooses $t-1$ random numbers $a_1, a_2, ..., a_{t-1}$ and generates a polynomial:
$$P_j(x) = mk_j + a_1 x + a_2 x^2 + ... + a_{t-1} x^{t-1} \quad (1)$$
2) MS_j computes the secret share for each UAV_i within the group: $x_i = P_j(UID_i)$, where the public key $X_i = x_i \cdot P$.
3) MS_j selects a random number zsk_j as the group key and unicasts $Enc_{sk_{j,i}}(x_i, zsk_j)$ to UAV_i.
4) After receiving unicast messages from MS_j, each UAV_i decrypts using the key $sk_{i,j}$ to obtain its own share x_i and the group key zsk_j.

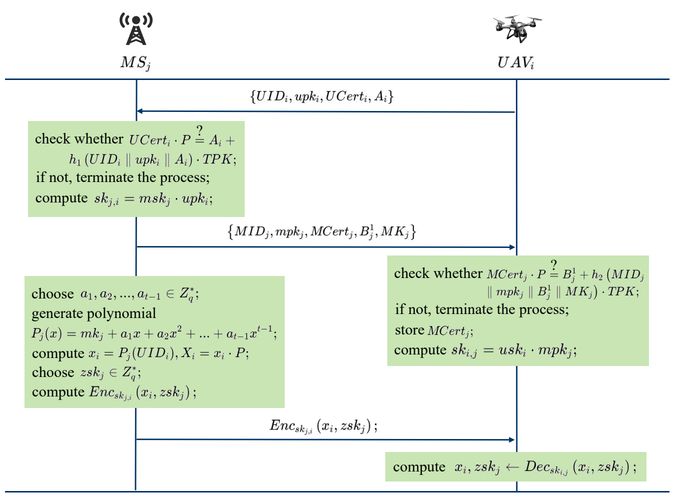

MS_j
UAV_i
$\{UID_i, upk_i, UCert_i, A_i\}$
check whether $UCert_i \cdot P \overset{?}{=} A_i + h_1(UID_i \parallel upk_i \parallel A_i) \cdot TPK$;
if not, terminate the process;
compute $sk_{j,i} = msk_j \cdot upk_i$;
$\{MID_j, mpk_j, MCert_j, B_j^1, MK_j\}$
check whether $MCert_j \cdot P \overset{?}{=} B_j^1 + h_2(MID_j \parallel mpk_j \parallel B_j^1 \parallel MK_j) \cdot TPK$;
if not, terminate the process;
store $MCert_j$;
compute $sk_{i,j} = usk_i \cdot mpk_j$;
choose $a_1, a_2, ..., a_{t-1} \in Z_q^*$;
generate polynomial
$P_j(x) = mk_j + a_1x + a_2x^2 + ... + a_{t-1}x^{t-1}$;
compute $x_i = P_j(UID_i), X_i = x_i \cdot P$;
choose $zsk_j \in Z_q^*$;
compute $Enc_{sk_{j,i}}(x_i, zsk_j)$;
$Enc_{sk_{j,i}}(x_i, zsk_j)$;
compute $x_i, zsk_j \leftarrow Dec_{sk_{i,j}}(x_i, zsk_j)$;

Fig. 4. Flowchart of the initial authentication phase.

D. *Group Authentication*

Before group fusion, all UAVs need to authenticate the identity of the group to be joined. After authentication, every two UAVs negotiate a session key to encrypt the shared shares for the next phase. Assuming there are two UAV groups, one from MS_1 and the other from MS_2. Considering the possibility of physical damage to UAVs during flight, the final numbers of UAVs eligible for group authentication are denoted as n_1 and n_2, forming the signer sets $group_1$ and $group_2$, respectively. The authentication for both groups is carried out simultaneously. To avoid redundant descriptions, this section takes $group_1$'s intra-group authentication and $group_2$ authenticates the legitimacy of $group_1$ as an example. The entity interaction in this phase is shown in Fig. 5.

1) Intra-group session keys negotiation:

1) Each UAV_i in $group_1$ selects a random number k_i and computes $K_i = k_i \cdot P$. Then, UAV_i broadcasts $\{UID_i, K_i, HMAC_i((UID_i \parallel K_i), zsk_1)\}$.
2) After receiving the message from UAV_j, UAV_i calculates $HMAC_j^{'}((UID_j \parallel K_j), zsk_1)$. Then UAV_i verifies whether $HMAC_j^{'}((UID_j \parallel K_j), zsk_1) = HMAC_j((UID_j \parallel K_j), zsk_1)$ holds. If it is true, UAV_i computes the session key with UAV_j as $key_{i,j} = k_i \cdot K_j$.

2) Cross-group session keys negotiation:

1) Individual signature generation: Each UAV_i in $group_1$ computes the Lagrange interpolation coefficient $L_i = \prod_{j=1,j\neq i}^{n_1} \frac{UID_j}{UID_j - UID_i} (mod\ q)$. Let $mess_1 = \{(UID_1, K_1), (UID_2, K_2), ..., (UID_{n_1}, K_{n_1})\}$, $mess_2 = \{MCert_1, B_1^1, MID_1, mpk_1, MK_1\}$. UAV_i computes the individual signature $s_i = k_i + h_3(mess) \cdot x_i \cdot L_i (mod\ q)$ for the message $mess = \{mess_1, mess_2\}$. Note that s_i serves as the certificate issued by the mobile station to UAV_i. Then UAV_i broadcasts $\{mess_2, UID_i, s_i, X_i\}$.
2) Signature aggregation and verification: After receiving n_1 signatures from $group_1$, UAV_k in $group_2$ computes the aggregate signature $s = \sum_{i=1}^{n_1} s_i$, and $K = \sum_{i=1}^{n_1} K_i$. Subsequently, the verification of $s \cdot P = K + h_3(mess) \cdot MK_1$ is performed. If this equation holds, it indicates the successful validation of $group_1$. Conversely, if it does not hold, the individual signature can be verified by checking whether the equation $s_i \cdot P - K_i = h_3(mess) \cdot r \cdot L_i \cdot X_i$ holds, thus identifying the legitimacy of UAV_i.
3) Certificate chain verification: The above steps have completed the validation of the legality of $group_1$. Next, UAVs in $group_2$ verify the origin of $group_1$ using the public key TPK of the TA, i.e., the legitimacy of MS_1. Determine whether the equation $MCert_1 \cdot P = B_1^1 + h_2(MID_1 \parallel mpk_1 \parallel B_1^1 \parallel MK_1) \cdot TPK$ holds. If the equation holds, MS_1 is a legitimate base station authenticated by the TA. Then, the session key between UAV_i in $group_1$ and UAV_k in $group_2$ is $key_{i,k} = k_i \cdot K_k$. This step applies the principle of trust transfer. The mobile station guarantees the legitimacy of the UAVs' certificate. TA guarantees the legitimacy of the mobile station's certificate. Therefore, with both the TA and the mobile station offline, the group's identity can be legitimate if both verifications pass.

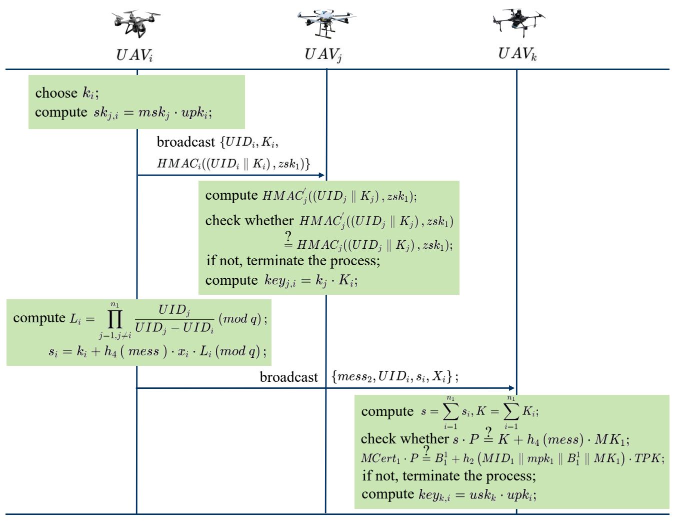

UAV_i
UAV_j
UAV_k
choose k_i;
compute $sk_{j,i} = msk_j \cdot upk_i$;
broadcast $\{UID_i, K_i, HMAC_i((UID_i \parallel K_i), zsk_1)\}$
compute $HMAC_j^{'}((UID_j \parallel K_j), zsk_1)$;
check whether $HMAC_j^{'}((UID_j \parallel K_j), zsk_1) \overset{?}{=} HMAC_j((UID_j \parallel K_j), zsk_1)$;
if not, terminate the process;
compute $key_{j,i} = k_j \cdot K_i$;
compute $L_i = \prod_{j=1,j\neq i}^{n_1} \frac{UID_j}{UID_j - UID_i} (mod\ q)$;
$s_i = k_i + h_4(mess) \cdot x_i \cdot L_i (mod\ q)$;
broadcast $\{mess_2, UID_i, s_i, X_i\}$;
compute $s = \sum_{i=1}^{n_1} s_i, K = \sum_{i=1}^{n_1} K_i$;
check whether $s \cdot P \overset{?}{=} K + h_4(mess) \cdot MK_1$;
$MCert_1 \cdot P \overset{?}{=} B_1^1 + h_2(MID_1 \parallel mpk_1 \parallel B_1^1 \parallel MK_1) \cdot TPK$;
if not, terminate the process;
compute $key_{k,i} = usk_k \cdot upk_i$;

Fig. 5. Flowchart of the group authentication phase.

E. *Group Management*

This phase merges two groups and updates UAVs' subsecrets by sharing the shares. The entity interaction in this phase is shown in Fig. 6.

1) Group fusion: Assuming that the verified UAVs in $group_1$ and $group_2$ want to share shares, UID_i and UID_j are the identities of the UAVs in the old group, UID_k is the identity of the UAVs in the new group. Take one of the parties as an example: the UAVs in $group_1$ share the shares with the UAVs in $group_2$. Here, $group_1$ is the old group, $group_1$ and $group_2$ forms a new group.

1) UAV_i in the old group selects $t-1$ random numbers $q_{i,1}, ..., q_{i,t-1}$ to generate polynomial:

$$Q_i(x) = q_{i,1}x + ... + q_{i,t-1}x^{t-1}. \tag{2}$$

Obviously, $Q_i(0) = 0$.

2) UAV_i in the old group selects $t-1$ random numbers $r_{i,k,1}, ..., r_{i,k,t-1}$ for UAV_k in the new group, generating polynomial:

$$R_{i,k}(x) = r_{i,k,1}x + r_{i,k,2}x^2 + ... + r_{i,k,t-1}x^{t-1}, \quad (3)$$

satisfying $R_{i,k}(k) = 0$.

3) UAV_i in the old group calculates the vector $V_{i,j} = \{Q_i(j)+R_{i,1}(j), Q_i(j)+R_{i,2}(j), ..., Q_i(j)+R_{i,k}(j)\}$, broadcasts the encrypted information $Enc_{key_{i,j}}(V_{i,j})$. (**Efficiency improvement**: computation and communication between UAVs of the old group can be done in advance, and the joining of new UAVs does not affect the share-sharing process of the old group.)

4) UAV_j decrypts $V_{i,j}$ to obtain all legitimate shared shares: $Q_i(j) + R_{i,1}(j), Q_2(j) + R_{i,2}(j), ..., Q_t(j) + R_{i,k}(j)$. Then UAV_j computes $v_{j,k} = P_1(j) + \sum_{i=1}^{t}(Q_i(j) + R_{i,k}(j))$, and broadcasts the encrypted information $Enc_{key_{j,k}}(v_{j,k})$.

5) Each UAV_k decrypts to obtain $v_{j,k}$. After obtaining at least t valid shares, UAV_k interpolates to recover the polynomial:

$$P_1(x)+Q(x)+R_k(x) = \sum_{k=1}^{t} v_{j,k} \prod_{k=1,k\neq j}^{t} \frac{x-UID_k}{UID_j-UID_k}. \quad (4)$$

and computes the shared share from $group_2$, that is $x_{k_{2,1}} = |P_1(x) + Q(x) + R_k(x)|_{x=k} = P_1(k) + Q(k)$.

Similarly, after the UAVs in $group_2$ share their shares with the UAVs in $group_1$, the shared share of each UAV_k is $x_{k_{2,1}}$. UAV_k calculates the final new share $x_{k_{new}} = x_{k_{1,2}} + x_{k_{2,1}}$. Due to the homomorphic property of Shamir's secret sharing, each point $(UID_k, x_{k_{new}})$ formed by the UAV_k satisfies the polynomial:

$$P(x) = mk_1 + mk_2 + e_1x + e_2x^2 + ... + e_{t-1}x^{t_{new}-1}. \quad (5)$$

Here, mk_1 and mk_2 are the original secrets of $group_1$ and $group_2$, and $e_1, e_2, ..., e_{t-1}$ are random numbers. Thus, the merged group $group_{1,2}$ is formed. Note that t_{new} is the threshold of the new group after fusion.

Remark: Assume that the total number of members in the new group after fusion is n, so $n = n_1 + n_2$. Let t be the threshold of each group before fusion, and t_{new} be the threshold of the new group after fusion. According to the system security requirements specified in the MPSS scheme [27], $n_1 \geq 3t+1$, $n_2 \geq 3t+1$, and $n \geq 3t_{new}+1$, which means $n_1 + n_2 \geq 3t_{new}+1$. Taking $n_1 = n_2 = 4t$, we have $8t \geq 3t_{new}+1$, so $t_{new} \leq \frac{8t-1}{3}$. Therefore, $t_{new} = 2t$ can be adopted, which satisfies the security requirements of the new group after fusion.

2) Certificate updates: In $group_{1,2}$, each UAV_i computes $L_{i_{new}} = \prod_{j=1,j\neq i}^{n_1+n_2} \frac{UID_j}{UID_j-UID_i} \,(mod\ q)$, and updates its certificate as follows:

$$s_{i_{new}} = k_i + h_3(mess_{new}) \cdot x_{i_{new}} \cdot L_{i_{new}} \pmod{q}. \quad (6)$$

Among them, $mess_{new} = \{mess_{1_{new}}, mess_{2_{new}}\}$, $mess_{1_{new}} = \{(UID_1, K_1), ..., (UID_{n_1+n_2}, K_{n_1+n_2})\}$, and $mess_{2_{new}} = \{MCert_1, B_1^1, MID_1, mpk_1, MK_1, MCert_2, B_1^2, MID_2, mpk_2, MK_2\}$. Additionally, this scheme supports the continuous fusion of UAV groups. Assuming the merged $group_{1,2}$ intends to join $group_3$, the UAVs in $group_3$ only need to verify the new certificate of $group_{1,2}$ using the public key $MK = (mk_1 + mk_2) \cdot P = MK_1 + MK_2$. The verification steps are as follows. The UAVs in $group_3$ calculate $s_{new} = \sum_{i=1}^{n_1+n_2} s_{i_{new}}$. If the equation $s_{new} \cdot P = K + h_3(mess_{new}) \cdot MK$ holds true, it indicates that the new merged group $group_{1,2}$ is legitimate.

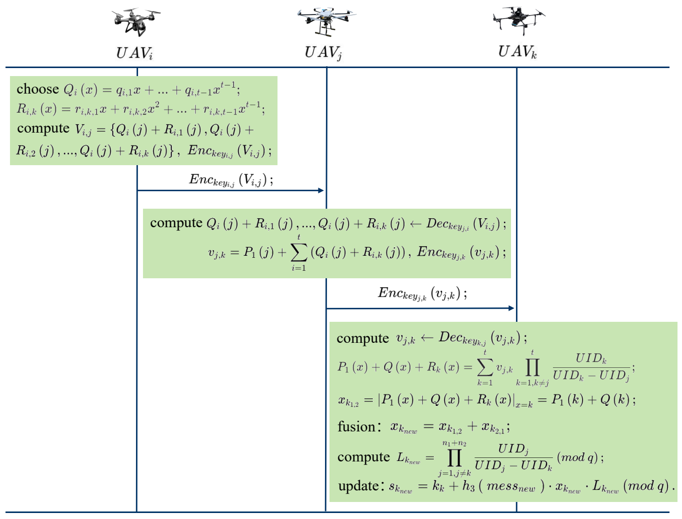

UAV_i
UAV_j
UAV_k
choose $Q_i(x) = q_{i,1}x + ... + q_{i,t-1}x^{t-1}$;
$R_{i,k}(x) = r_{i,k,1}x + r_{i,k,2}x^2 + ... + r_{i,k,t-1}x^{t-1}$;
compute $V_{i,j} = \{Q_i(j) + R_{i,1}(j), Q_i(j) + R_{i,2}(j), ..., Q_i(j) + R_{i,k}(j)\}$, $Enc_{key_{i,j}}(V_{i,j})$;
$Enc_{key_{i,j}}(V_{i,j})$;
compute $Q_i(j) + R_{i,1}(j), ..., Q_i(j) + R_{i,k}(j) \leftarrow Dec_{key_{j,i}}(V_{i,j})$;
$v_{j,k} = P_1(j) + \sum_{i=1}^{t}(Q_i(j) + R_{i,k}(j))$, $Enc_{key_{j,k}}(v_{j,k})$;
$Enc_{key_{j,k}}(v_{j,k})$;
compute $v_{j,k} \leftarrow Dec_{key_{k,j}}(v_{j,k})$;
$P_1(x) + Q(x) + R_k(x) = \sum_{k=1}^{t} v_{j,k} \prod_{k=1,k\neq j}^{t} \frac{UID_k}{UID_k - UID_j}$;
$x_{k_{1,2}} = |P_1(x) + Q(x) + R_k(x)|_{x=k} = P_1(k) + Q(k)$;
fusion: $x_{k_{new}} = x_{k_{1,2}} + x_{k_{2,1}}$;
compute $L_{k_{new}} = \prod_{j=1,j\neq k}^{n_1+n_2} \frac{UID_j}{UID_j - UID_k}(mod\ q)$;
update: $s_{k_{new}} = k_k + h_3(mess_{new}) \cdot x_{k_{new}} \cdot L_{k_{new}}(mod\ q)$.

Fig. 6. Flowchart of the group management phase.

VI. Security Analysis

This section will employ formal and informal analyses to assess the proposed scheme's correctness and security.

A. The Correctness of the Scheme

Assuming that the UAVs participating in group authentication in $group_1$ are denoted as $UAV_1, UAV_2, ..., UAV_{n_1}$, where $t \leq n_1 \leq N_1$. According to the Lagrange interpolation theorem, the secret $mk_1 = P_1(0) = \sum_{i=1}^{n_1} P_1(UID_i)L_i = \sum_{i=1}^{n_1} x_iL_i \,(mod\ q)$, where $L_i = \prod_{j=1,j\neq i}^{n_1} \frac{UID_j}{UID_j-UID_i} \,(mod\ q)$ represents the Lagrange interpolation coefficients. UAV_i forms an individual signature $s_i = k_i + h_3(mess\ ) \cdot x_i \cdot L_i \,(mod\ q)$.

$$\begin{aligned}\sum_{i=1}^{n_1} s_i &= \sum_{i=1}^{n_1} k_i + h_3(mess) \cdot \sum_{i=1}^{n_1} x_i \cdot L_i \\ &= \sum_{i=1}^{n_1} k_i + mk_1 \cdot h_3(mess)\end{aligned} \quad (7)$$

$$\begin{aligned}\left(\sum_{i=1}^{n_1} s_i\right) \cdot P &= \left(\sum_{i=1}^{n_1} k_i + mk_1 \cdot h_3(mess)\right) \cdot P \\ &= \sum_{i=1}^{n_1} k_i \cdot P + mk_1 \cdot h_3(mess) \cdot P \quad (8) \\ &= \sum_{i=1}^{n_1} k_i \cdot P + h_3(mess) \cdot MK_1\end{aligned}$$

Therefore, if $(\sum_{i=1}^{n_1} s_i)\cdot P = \sum_{i=1}^{n_1} k_i\cdot P + h_3\,(mess)\cdot MK_1$ holds, the threshold signature verification passes, indicating the correctness of the scheme.

B. Formal Security Proof using Random Oracle Model

Random Oracle Model (ROM) [29] is used to formally analyze security protocols, typically to prove whether the session key negotiation by the protocol is secure or if the generated signatures can be forged. Here, we employ ROM to verify the scheme's security under the Elliptic Curve Discrete Logarithm Problem (ECDLP).

1) Definition of Hard Problem: Let P and Q be generators of an additive group G of order q over the finite field Z_q, where a is an integer in the field Z_q. The adversary $\mathcal{A}$ knows P and Q and finds it computationally infeasible to calculate the value of a such that $Q = a\cdot P$, within probabilistic polynomial time (PPT). In other words, the advantage of the adversary $\mathcal{A}$ in computing a within PPT is considered negligible.

2) Security Model:

- **Participants**: Suppose $UAV_1^1, UAV_2^1, ..., UAV_{n_1}^1$ represents the participation in the first UAV group shared members, $UAV_1^2, UAV_2^2, ..., UAV_{n_2}^2$ represent the participants in the second UAV group shared members. Each participant has many different instances, each corresponding to a random oracle.
- **Oracle query**: The challenge scenario involves two entities: one is the challenger $\mathcal{C}$, and the other is the adversary $\mathcal{A}$ that runs in polynomial time. We define a series of queries between them to simulate the security model of the scheme. In the random oracle model, if the adversary $\mathcal{A}$ can break the scheme we designed in polynomial time, it is equivalent to the challenger $\mathcal{C}$ breaking the ECDLP. Of course, $\mathcal{A}$ can make queries using the oracle, and the supported queries are as follows:

  a) Setup Oracle: The challenger $\mathcal{C}$ first builds the system, forms system parameters, and then sends the system public parameters to the attacker $\mathcal{A}$.

  b) Hash Oracle: When the challenger $\mathcal{A}$ performs a hash query of a message if the hash result exists in the hash list, the output returns the hash result; if there is no hash value corresponding to the current message in the hash list, a random value is returned as the output result.

  c) Secret Key Oracle: The query performs a search for the private key, and the attacker $\mathcal{A}$ can obtain the corresponding private key by inputting the participant's UID_i, with the challenger $\mathcal{C}$ able to respond accordingly. It is assumed that all the private keys queried by $\mathcal{A}$ are present in the list $List_{sk}$.

  d) Sign Oracle: The attacker $\mathcal{A}$ returns a signature $\{s_i, K_i, UID_i\}$. If the signature is validated as input, returning result 1, and the corresponding private key is not in $List_{sk}$, it indicates a victory for the attacker $\mathcal{A}$.

3) Security Proof:

Theorem: If adversary $\mathcal{A}$ is able to breach the proposed scheme, then challenger $\mathcal{C}$ can solve the ECDLP.

Proof: If challenger $\mathcal{A}$ can successfully forge a signature $\{s_i, K_i, UID_i\}$, then challenger $\mathcal{C}$ can break the ECDLP in polynomial time, i.e., output x_i within PPT such that $X_i = x_i\cdot P$. $\mathcal{A}$ can make the following queries to $\mathcal{C}$:

1) Setup Oracle: The challenger $\mathcal{C}$ first forms the system parameters $\{G, q, P, TPK, h_3\}$, and then sends the system public parameters to the attacker $\mathcal{A}$.
2) h_3 Oracle: The challenger $\mathcal{C}$ maintains a hash list $List_{h_3}$, initially initialized to empty. When $\mathcal{A}$ queries the message $(MCert_j, MK_j)$, $\mathcal{C}$ first checks if the hash value x_{h_3} of the message exists in the list. If it does, $\mathcal{C}$ returns x_{h_3}; otherwise, $\mathcal{C}$ selects a random value x_{h_3}, inserts $(MCert_j, MK_j, x_{h_3})$ into $List_{h_3}$, and returns the random value x_{h_3} to $\mathcal{A}$.
3) Secret Key Oracle: When the attacker $\mathcal{A}$ utilizes UID_i for key query, $\mathcal{C}$ selects polynomial $P_{\mathcal{C}}(x)$, substitutes UID_i into the polynomial to compute xx_i, which corresponds to the sub-secret associated with UID_i. Finally, $\mathcal{C}$ sends (UID_i, xx_i) to $\mathcal{A}$.
4) Sign Oracle: $\mathcal{A}$ selects random numbers $k_i\,(i = 1, 2, ..., t)$ for UID_i, computes $K_i = k_i\cdot P$. Then, $\mathcal{A}$ forms the signature:

$$s_i = k_i + x_{h_3}\cdot xx_i\cdot L_i \tag{9}$$

Finally, $\mathcal{A}$ forms the signature $\{s_i, K_i, UID_i\}$. According to the forking lemma [30], by performing the above operations, $\mathcal{A}$ can obtain different outputs $x_{h_3}^{'}$ with the same input $(MCert_j, MK_j)$. Therefore, another valid forged signature $s_i^{'} = k_i + x_{h_3}^{'}\cdot xx_i\cdot L_i$ can be obtained, thus obtaining a in ECDLP:

$$xx_i = \frac{s_i^{'} - s_i}{\left(x_{h_3}^{'} - x_{h_3}\right)\cdot r'\cdot L_i} \tag{10}$$

We assume that q_{h_3} and q_s represent the number of times the h_3 Oracle and Secret Key Oracle are executed, respectively. Let E_1 denote the event of no conflict between the Hash Oracle and Secret Key Oracle. E_2 represents the event that $\mathcal{A}$ can output a valid forged signature. Therefore, $\mathcal{A}$ can return a forged signature with the probability:

$$acc = Pr\left[E_1E_2\right] = Pr\left[E_1\right]Pr\left[E_2\mid E_1\right] \tag{11}$$

Due to $\mathcal{A}$ performing $q_{h_3}+q_s$ queries, the probability of the event that $bad \longleftarrow true$ is $\frac{q_{h_3}+q_s}{q}$ for one secret key query, and the probability for q_s queries is $\frac{q_s(q_{h_3}+q_s)}{q}$. Therefore,

$$Pr\left[E_1\right] = 1 - Pr\left[bad = true\right] = 1 - \frac{q_s(q_{h_3}+q_s)}{q} \tag{12}$$

Then, $Pr[E_2\mid E_1]$ represents the probability that $\mathcal{A}$ returns a valid forgery when $\mathcal{C}$ does not terminate due to queries from $\mathcal{A}$. Assuming $\mathcal{A}$ can return a valid forgery with probability not less than ε, then:

$$Pr\left[E_2\mid E_1\right] = \frac{Pr\left[E_1E_2\right]}{Pr\left[E_1\right]} \geq \varepsilon \tag{13}$$

TABLE III
COMPARISON OF SECURITY PROPERTIES.

Security properties	TAGKA [5]	BBKM [17]	PPAS [11]	Ours
Mutual authentication	√	×	√	√
No online center	√	×	×	√
Fairness	√	×	√	√
High fault tolerance	√	×	×	√
Support multi-groups	×	√	×	√
Confidentiality	√	−	−	√
Autonomous fusion	√	×	×	√
Certificate self-update	−	×	×	√
No TPD required	×	√	√	√

√: Satisfying the security property.
×: Not satisfying the security property.
−: Not mentioned.

$$\begin{aligned} Pr\left[E_1E_2\right] \geq \varepsilon Pr\left[E_1\right] &= \varepsilon\left(1-\frac{q_s(q_{h_3}+q_s)}{q}\right) \\ &= \varepsilon - \varepsilon\frac{q_s(q_{h_3}+q_s)}{q} \\ &\geq \varepsilon - \frac{q_s(q_{h_3}+q_s)}{q} \end{aligned} \tag{14}$$

According to the General Forking Lemma [31], $\mathcal{C}$ utilizes $\mathcal{A}$ as a subroutine and can output two valid signatures with a probability of $frk \geq acc\left(\frac{acc}{\hat{q}} - \frac{1}{\hat{h}}\right)$, where $\hat{q}$ represents the number of hash queries and $\hat{h}$ represents the number of responses to random oracle queries. Therefore, the probability of $\mathcal{C}$ breaking the ECDLP is:

$$\left(\varepsilon - \frac{q_s(q_{h_3}+q_s)}{q}\right)\left(\frac{\varepsilon - \frac{q_s(q_{h_3}+q_s)}{q}}{q_{h_3}} - \frac{1}{q}\right) \tag{15}$$

Among them, ε cannot be ignored, which conflicts with the ECDLP. Therefore, it is proved that the scheme is safe under the random oracle model.

C. Informal Security Analysis

This section provides an informal analysis of the security objectives the proposed scheme addresses in Section IV. Table III compares the scheme's security properties with other related schemes.

1) **Mutual authentication**: During the initial authentication phase, UAVs and MSs mutually provide and verify the certificates issued by TA. During the intra-group authentication, UAVs verify the sender's identity and message using HMAC. During cross-group authentication, each UAV validates the threshold signature of another group and the certificate of the corresponding MS. Thus, the proposed scheme facilitates mutual authentication between communicating entities.
2) **Session key negotiation**: In the initial authentication phase, a session key is negotiated using a Diffie-Hellman exchange after being authenticated. Similarly, UAVs in two groups negotiate a session key after completing identity authentication.
3) **Fairness**: Each UAV's status is equal in this scheme, and the cluster header UAV is absent. Furthermore, all UAVs participating in the signing process are eligible for a shared sub-secret.
4) **High fault tolerance**: According to the principle of the threshold signature, it is known that at least t UAVs are required to generate a signature. Thus, physical damage to at most $n-t$ UAVs in each group can be tolerated. At least t UAVs in each group are required to participate in the scheme, and the secure fusion of groups and efficient updating of certificates can be finished.
5) **Autonomous fusion**: TA and MS only participate in the initialization phase. The UAVs autonomously handle group authentication and fusion, which is suitable for fully distributed scenarios. UAVs in the group will all undergo signature authentication before fusion. Only UAVs that pass the verification can participate in the session key agreement and use this key to encrypt shares. The autonomous fusion of the group is achieved through share sharing among UAVs.
6) **Certificate self-update**: UAVs can update certificates using a threshold signature calculated with the new shares after the group fusion. UAVs are honest after intra-group and cross-group key negotiation. The only issue is the loss of UAVs due to physical damage. During the key negotiation phase, unauthorized UAVs have been identified through signature verification. Therefore, the shares sent by the remaining legitimate UAVs are valid. Signatures generated from these valid shares will not be subject to the injection of malicious signatures, and thus, the certificate update process is secure. Considering the scenario where internal attackers send invalid shares, reference can be made to the member consensus mechanism provided in the MPSS scheme.
7) **Confidentiality**: The coefficients of the polynomials Q and R_k generated by the scheme are randomly chosen, independent of any information known to the adversary. Q and R_k can be updated over time, and the correct share of UAV_k can only be computed when $x = k$. Only at that point, $R_k(k) = 0$. Share information for other independent variable values is considered interference information, and the shares reconstructed from it cannot satisfy the final polynomial, making them incorrect shares.
8) **Protect against some attacks**:

   - **Impersonation attack**: Assuming a malicious UAV intends to masquerade as a legitimate one, it needs to generate a message $\{UID_i, A_i, s_i, K_i, x_i\}$ that satisfies $UCert_i \cdot P = A_i + h_1(UID_i \parallel upk_i \parallel A_i) \cdot TPK$ and $s_i \cdot P = K_i + h_3(mess) \cdot X_i \cdot L_i$. However, A_i, K_i, and MK_1 are computed based on random numbers a_i, k_i, and mk_1 respectively. According to the ECDLP, generating such a message in polynomial time is computationally infeasible. Therefore, the proposed scheme is resilient against impersonation attack.

- **Tampering attack**: If a malicious attacker modifies the message, the certificate cannot be verified, and therefore the correct session key will not be obtained. Then, the generated signature will not make the equation $s_i \cdot P = K_i + h_3(mess) \cdot X_i \cdot L_i$ equal. This is because the properties of the hash function ensure the integrity of the message, and any tampering with the message can be detected. Therefore, the proposed scheme can resist tampering attack.
- **Man-in-the-middle attack**: When a UAV receives a signed message from another UAV_i, signature verification takes place. Verifying the signature involves shares x_i, the secret value mk_j, and the correctness is ensured through the certificate issued by TA. Therefore, the proposed scheme can resist man-in-the-middle attack.

VII. Experimental Evaluation

In this section, based on computational and communication cost, we will conduct comparative experiments with the schemes [5], [11], [17].

A. Experimental Settings

1) Environment Setup: We tested the experimental data on an AmovLab Prometheus 600 [32], a UAV equipped with onboard computing units with NVIDIA Carmel ARM processors @ 1.4 GHz. The operating system is Ubuntu 18.04. The TA's simulation environment is a PC with an Intel(R) Core(TM) i7-11700 @ 2.50GHz 2.50 GHz processor and 16GB of RAM, and the operating system is Windows 10.

Fig. 7. Amov Lab P600.

2) Parameter Configuration: The computational cost is mainly measured by the time spent on the operations in each phase. Elliptic curve BLS12381 and SHA-384 hash functions are used for the experiments. The symmetric encryption and decryption algorithm used is the AES algorithm. We used the public cryptographic library Miracl Core [33] to implement the cryptographic primitives. For ease of presentation, we define some notations about the execution time of cryptographic operations. T_{pm}: execution time of point-multiplication on elliptic curves; T_{pa}: execution time of point-addition on elliptic curves; T_h: execution time of a hash operation; T_{hm}: execution time of computing a hash code; T_{enc}: execution time of AES for encrypting 48 bytes; T_{dec}: execution time of AES for decrypting 48 bytes; T_{sha}: execution time of computing a Shamir's polynomial; T_{pr}: execution time of recovering a Shamir's polynomial; T_{che}: execution time of computating a Chebyshev's polynomial; T_{sm}: execution time of big numbers' multiplication; T_{sa}: execution time of big numbers' addtion. T_{inv}: execution time of inversion operation. The execution time of each operation is shown in Table IV. The $Time$ column represents the execution time of UAV, and the $Time^{'}$ column represents the execution time of TA. Table V shows the T_{sha} and T_{pr} corresponding to different t.

TABLE IV
Execution Time of Miracl Library Password Operations.

Symbol	Description	$Time$ (ms)	$Time^{'}$ (ms)
T_{pm}	Point multiplication	0.670	0.197
T_{pa}	Point addition	0.005	0.001
T_h	One-way hash operation	0.007	0.002
T_{hm}	Calculate the hash code	0.014	0.004
T_{enc}	Encryption	0.016	0.009
T_{dec}	Decryption	0.013	0.009
T_{che}	Chebyshev polynomials	5.837	0.869
T_{sm}	Multiplication of big numbers	0.011	0.004
T_{sa}	Addition of big numbers	0.001	0.0002
T_{inv}	Inversion operation	0.067	0.009

$Time$: Represents runtime on a UAV.
$Time^{'}$: Represents runtime on a TA.

TABLE V
Execution Time of T_{sha} and T_{pr}.

t	1	2	3	4	5
T_{sha}(ms)	0.029	0.051	0.070	0.089	0.111
T_{pr}(ms)	0.012	0.123	0.311	0.596	0.989
t	6	7	8	9	10
T_{sha}(ms)	0.130	0.151	0.169	0.190	0.209
T_{pr}(ms)	1.449	1.966	2.586	3.382	4.163

t: Threshold.
T_{sha}: Calculate Shamir's secret shares on UAV.
T_{pr}: Lagrange interpolation on UAV.

For the communication cost, the size of the parameters involved in the scheme are set as follows: the size of the elements on the finite domain is 48 bytes; the size of the elliptic curve is 97 bytes; the size of UID and MID are 16 bytes; the output of the hash function is 48 bytes; the size of the random number is 48 bytes; and the size of the timestamp is 4 bytes.

B. ***Computational Cost***

In this scheme, during the intra-group session key negotiation phase, the computation for each UAV in $group_1$ includes n_1 multiplication operations over G and n_1 hash

TABLE VI
COMPUTATION AND COMMUNICATION COST ON EACH UAV OF THE PROPOSED SCHEME.

Each Phase	Computational Cost (ms)	Communication Cost (bytes)
Intra-group Keys Negotiation	$n_1T_{pm}+n_1T_{hm}$	161
Cross-group Keys Negotiation	$(n_2+4)T_{pm}+(n_2+1)T_{pa}+3T_h+(2n_1-1)T_{sm}+n_2T_{sa}+(n_1-1)T_{inv}$	516
Group Fusion	$n_1(n_1+n_2+1)T_{sha}+[\frac{5}{4}n_1(n_1+n_2)+1]T_{sa}+[n_1(n_1+n_2)-1](T_{enc}+T_{dec})+2T_{pr}$	$48[n_1(n_1+n_2)-1]$
Certificate Update	$T_h+(2n_1+2n_2-1)T_{sm}+T_{sa}+(n_1+n_2-1)T_{inv}$	0

code computation operations, with a computational cost of $n_1T_{pm}+n_1T_{hm}\approx 0.684n_1\ ms$.

During the cross-group session key negotiation phase, the computation for each UAV includes n_2+4 multiplication operations over G, n_2+1 addition operations over G, 3 hash operations, $2n_1-1$ multiplication operations over Z_q, n_2 addition operations over Z_q, and n_1-1 inverse operations over Z_q, with a computational cost of $(n_2+4)T_{pm}+(n_2+1)T_{pa}+3T_h+(2n_1-1)T_{sm}+n_2T_{sa}+(n_1-1)T_{inv}\approx 0.765n_1+2.628\ ms$.

During the group fusion phase, the computation for UAVs includes $n_1(n_1+n_2+1)$ Shamir polynomial operations, $[\frac{5}{4}n_1(n_1+n_2)+1]$ addition operations over Z_q, $[n_1(n_1+n_2)-1]$ encryption and decryption operations, and 2 Lagrange interpolation operations, with a computational cost of $n_1(n_1+n_2+1)T_{sha}+[\frac{5}{4}n_1(n_1+n_2)+1]T_{sa}+[n_1(n_1+n_2)-1](T_{enc}+T_{dec})+2T_{pr}\approx(0.0605+2T_{sha})n_1^2+n_1T_{sha}+2T_{pr}-0.028\ ms$.

During the certificate update phase, the computation for each UAV includes 1 hash operation, $(2n_1+2n_2-1)$ multiplication operations over Z_q, 1 addition operation over Z_q, and (n_1+n_2-1) inverse operations over Z_q, with a computational cost of $T_h+(2n_1+2n_2-1)T_{sm}+T_{sa}+(n_1+n_2-1)T_{inv}\approx 0.178n_1-0.07\ ms$.

The second column of Table VI presents the computational cost of each UAV in each phase of this scheme. When $n_1=n_2$, the computational costs of intra-group key negotiation, cross-group key negotiation, and certificate update are linearly related to n_1, while the computational cost of group fusion is quadratically related to n_1, resulting in higher complexity.

Fig. 8 shows the growth trend of the computational cost of UAVs in each phase as the number of group members changes. As $n_1=n_2$ increases, the computational cost of each phase rises. Among them, the cost of the group fusion phase grows the fastest and is the key factor affecting the total cost. The costs of intra-group key negotiation, cross-group key negotiation, and certificate update also increase to varying degrees, but their impact on the total cost is relatively small.

In the BBKM scheme proposed by Tan et al. [17], when a member joins, the computational cost of the cluster head is $(n_1+n_2)(3T_{pm}+T_{pa}+4T_h+T_{sm}+T_{sa})\approx 4.11n_1\ ms$. The computational cost of a UAV is $T_{pm}+2T_h+T_{sm}+T_{sa}\approx 0.696\ ms$. When the group members change, the cluster head redistributes the group key according to Algorithm $\eta(\cdot)$. Each UAV obtains the group key through the inverse algorithm $\eta^{-1}(\cdot)$ and uploads the updated private key to the blockchain. According to the scheme it cites [34], $T_{\eta(\cdot)}=19ms$ and

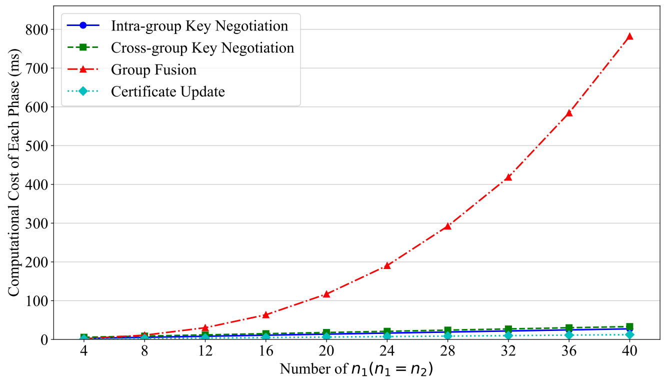


Fig. 8. Computational cost in each phase of our proposed scheme.

$T_{\eta^{-1}(\cdot)}=5.3ms$. The update cost of the cluster head is $(n_1+n_2)(3T_{pm}+T_{pa}+5T_h+6T_{dec}+T_{sm}+T_{sa})+(T_\eta+T_{pm}+2T_h+T_{sm}+T_{sa})\approx 4.28n_1+19.696\ ms$, and the update cost of a UAV is $T_{sm}+T_{sa}+T_{pa}+T_{\eta^{-1}}\approx 8.114\ ms$.

In Khan et al.'s [11] PPAS scheme, a single UAV must obtain a certificate from the GCU before joining a group. The joining phase in this scheme is equivalent to the registration process of a new UAV. To achieve group fusion similar to our scheme, the scheme [11] is extended such that, after obtaining the certificate, authentication by the original group members is required. When (n_1+n_2) UAVs join the group, the computational cost of the GCU is $(n_1+n_2)(T'_{pm}+T'_{pa}+T'_h+T'_{sm}+T'_{sa})\approx 0.4084n_1\ ms$, and the authentication cost of UAVs is $(n_1+n_2-1)(T_{pm}+T_{pa}+4T_{enc}+4T_{dec})+[2(n_1+n_2)-1]T_h+T_{sm}+T_{sa}\approx 1.61n_1-0.786\ ms$.

In Zhang et al.'s [5] TAGKA scheme, a new UAV needs to authenticate with two UAVs in the original group before joining, and negotiate a session key after successful authentication. However, the relationship among group members in this scheme is a chain structure, which makes it difficult to support batch joining of members. Therefore, when n_1+n_2 UAVs join the group, the computational cost of UAVs in the fusion phase is $(n_1+n_2)(8T_{che}+15T_h+4T_{enc}+4T_{dec})\approx 93.834n_1-46.917\ ms$. Due to changes in group members, each UAV must recalculate its key parameters. The computational cost of UAVs in the update phase is $3T_h+(n_1+n_2)T_{sm}+(n_1+n_2-1)T_{sa}+T_{pr}\approx 0.024n_1+0.02+T_{pr}\ ms$.

Table VII presents the computational cost of these schemes in the group fusion and certificate update phases. It can be seen that both Tan et al.'s [17] BBKM scheme and Khan et al.'s [11]

TABLE VII
COMPARISON OF COMPUTATIONAL COST OF DIFFERENT SCHEMES

Schemes	Group Fusion (ms)		Certificate Update (ms)	
	UAV_{header}/GCU	UAV_{join}	UAV_{header}/GCU	UAV_{join}
BBKM [17]	$(n_1+n_2)(3T_{pm}+T_{pa}+4T_h+T_{sm}+T_{sa})$	$T_{pm}+2T_h+T_{sm}+T_{sa}$	$(n_1+n_2)(3T_{pm}+T_{pa}+5T_h+6T_{dec}+T_{sm}+T_{sa})+(T_\eta+T_{pm}+2T_h+T_{sm}+T_{sa})$	$4T_{pm}+3T_h+6T_{enc}+T_{sm}+T_{sa}+T_{pa}+T_{\eta^{-1}}$
PPAS [11]	$(n_1+n_2)(T'_{pm}+T'_{pa}+T'_h+T'_{sm}+T'_{sa})$	$(n_1+n_2-1)(T_{pm}+T_{pa}+4T_{enc}+4T_{dec})+[2(n_1+n_2)-1]T_h+T_{sm}+T_{sa}$	$-$	$-$
TAGKA [5]	0	$(n_1+n_2-1)(8T_{che}+15T_h+4T_{enc}+4T_{dec})$	0	$3T_h+(n_1+n_2)T_{sm}+(n_1+n_2-1)T_{sa}+T_{pr}$
Ours	0	$n_1(n_1+n_2+1)T_{sha}+[\frac{5}{4}n_1(n_1+n_2)+1]T_{sa}+[n_1(n_1+n_2)-1](T_{enc}+T_{dec})+2T_{pr}$	0	$T_h+(2n_1+2n_2-1)T_{sm}+T_{sa}+(n_1+n_2-1)T_{inv}$

$-$: Not supported.

PPAS scheme require entities similar to trusted authorities to bear the main computational cost. Zhang et al.'s [5] TAGKA scheme and our scheme do not require the participation of trusted authorities, resulting in only a computational cost on the UAVs.

Fig. 9 compares the computational cost of our scheme with the comparison schemes. As the number of UAVs in a single group increases, the computational cost of each scheme in each phase becomes higher. It should be noted that the horizontal axis represents the number of members in a single group, and the total number of members in the system is twice that number (with the maximum value being 80 in the figures).

It can be observed that in the fusion phase, the better the computational performance of our scheme, the smaller the number of members in a single group. However, the BBKM scheme proposed by Tan et al. [17] involves time-consuming blockchain read and write operations, and the computational time consumption related to blockchain is not introduced here. The PPAS scheme proposed by Khan et al. [11] requires the participation of the TA in fusion, and a large amount of computational cost is concentrated on the TA. In the update phase, although the computational cost of the TAGKA scheme proposed by Zhang et al. [5] is lower than that of our scheme, its update highly depends on the ideal tamper-proof device (TPD) built into the UAV. The PPAS scheme proposed by Khan et al. [11] does not support certificate update.

In general, our scheme exhibits better computational performance compared to other schemes and can implement more security functions. In addition, the novelty of our scheme lies in the proposal of the concept of group fusion, which is not involved in other schemes.

Additionally, we assessed the computational cost of verifying the new group after merging groups in this scheme, as illustrated in Fig. 10. As the group size n_1 or n_2 changes, the green line in Fig. 10 represents the computational cost for certificate update, while the blue line denotes the computational cost required to verify the certificate of the new group. It can be observed that changes in group size have little impact on the computational cost of verifying the new certificate. This is because verifying the legitimacy of the new group only requires the use of the public key MK; thus, it is sufficient to verify the aggregated threshold signature just once.

C. *Communication Cost*

For this scheme, during the intra-group session key negotiation phase, each UAV sends the information $\{UID_i, K_i, HMAC_i((UID_i \parallel K_i), zsk_1)\}$, which is $|ID|+|G|+|Z_q| = 161\ bytes$. During the cross-group session key negotiation phase, each UAV sends the information $\{mess_2, UID_i, s_i, X_i\}$, which is $2|Z_q|+4|G|+2|ID| = 516\ bytes$. During the group fusion phase, each UAV needs to broadcast the encrypted messages $Enc_{key_{i,j}}(V_{i,j})$ and $Enc_{key_{j,k}}(v_{j,k})$, which are $(n_1-1)(n_1+n_2)|Z_q| = 48(n_1-1)(n_1+n_2)$ $bytes$ and $(n_1+n_2-1)|Z_q| = 48(n_1+n_2-1)\ bytes$ respectively. Therefore, the communication cost of each UAV is $48(n_1-1)n_2+48n_2 = 48[n_1(n_1+n_2)-1]\ bytes$. No interaction is required during the certificate update phase, so the communication cost is 0.

The third column of Table VI presents the communication cost of each UAV in each phase of this scheme. When $n_1 = n_2$, the communication costs of intra-group and cross-group key negotiation are relatively stable and independent of n_1. No communication is required during the certificate update phase, so the cost is 0. However, the communication cost of group fusion shows a quadratic function growth trend as n_1 increases.

Fig. 11 shows the communication cost of UAVs in each phase of this scheme as the numbers of n_1 and n_2 change. It can be seen that most of the total communication cost is incurred during the group fusion phase.

In Tan et al.'s [17] BBKM scheme, a single UAV needs to send a join request $participation = C_x, PK_{xy}, T, Sign_{sk_{xy}}(m)$ during the fusion phase, which is $|ID|+|G|+|T|+|Z_q| = 165\ bytes$. The size of the join transaction generated by the cluster head for (n_1+n_2) UAVs is $97(n_1+n_2)\ bytes$. During the update phase, the update request sent by a single UAV is $update = EN_{Gkey_x}\{PK_{xy}, PK_{xynew}, T, Sign_{sk_{xy}}(m_{new})\}$, which is 288 $bytes$. The communication cost of the cluster head when distributing the group key is 458 $bytes$, and the update transaction generated for UAVs is $362(n_1+n_2)\ bytes$, with a total cost of $362(n_1+n_2)+458\ bytes$.

In Khan et al.'s [11] PPAS scheme, a UAV sends an appli-

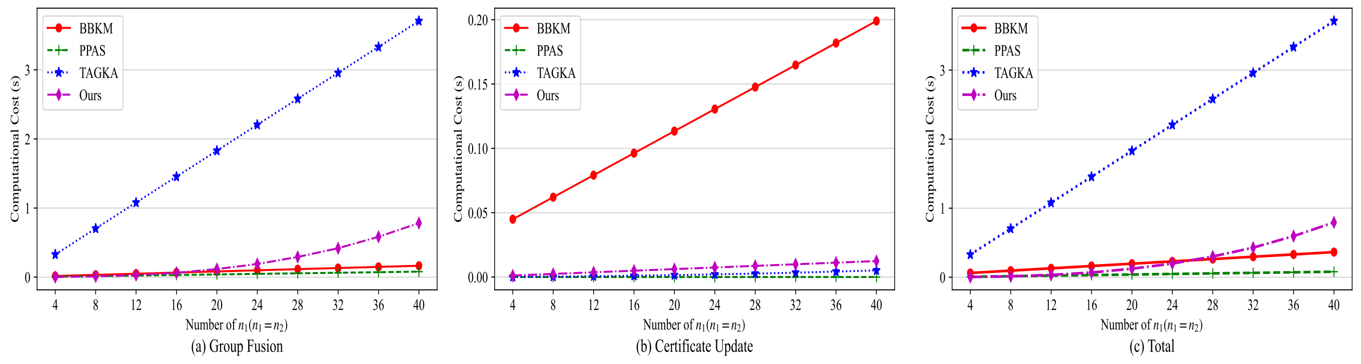


Fig. 9. Computational cost in each phase of our proposed scheme.

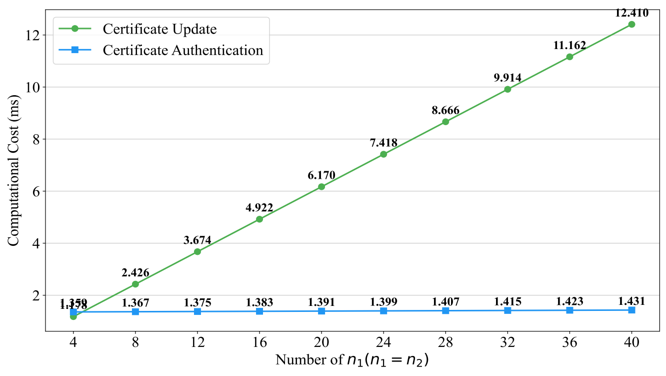


Fig. 10. Computational cost after group fusion in our proposed scheme.

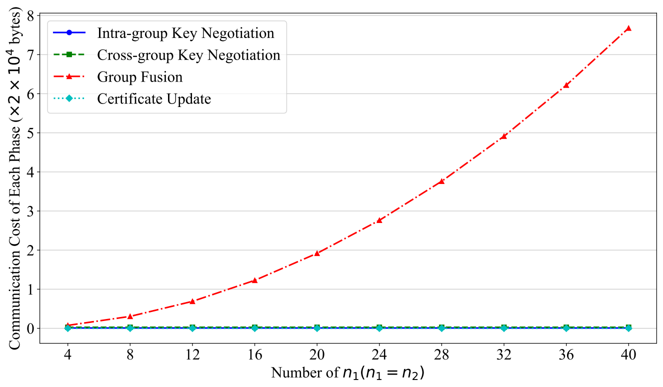


Fig. 11. Communication cost in each phase of our proposed scheme.

cation (l_{new}, ID_{new}) to the TA, which is $|ID| + |G| = 113$ *bytes*. The TA issues a certificate (C_{new}, s_{new}) to the UAV, which is $(|Z_q| + |G|) = 145$ *bytes*. During the authentication phase, UAVs send (ξ, H) to each other, which is $|Z_q| + |ID| + |G| + |T| + |Z_q| = 213$ *bytes*. Therefore, when $(n_1 + n_2)$ UAVs join the group, the communication cost of the TA is $145(n_1 + n_2)$ *bytes*, and the communication cost of UAVs is $213(n_1 + n_2 - 1) + 113 = 213(n_1 + n_2) - 100$ *bytes*.

In Zhang et al.'s [5] TAGKA scheme, a newly joined UAV sends $\{C_i, T_{r_i}(x), MAC_1, T_1\}$, $\{C_{i+1}, T_{r_{i+1}}(x), MAC_2, T_2\}$ and $\{MAC_3, T_3\}$ during the authentication process. The communication costs are $4|Z_q| + |T| = 196$ *bytes*, $4|Z_q| + |T| = 196$ *bytes*, and $|Z_q| + |T| = 52$ *bytes* respectively. Therefore, when $(n_1 + n_2)$ UAVs join the group, the communication cost in the fusion phase is $444(n_1 + n_2)$ *bytes*. During the update phase, each UAV broadcasts $\{A_i, AX_i, AY_i\}$, which is $3|Z_q| = 144$ *bytes*.

Table VIII summarizes the communication costs of these schemes in the group fusion and certificate update phases. Compared with the schemes [5], [11], [17] where communication costs grow linearly, the communication cost of our scheme grows in a quadratic function.

Fig. 12 shows the communication costs of our scheme and the comparison schemes. Here, we do not introduce the blockchain communication costs involved in Tan et al.'s [17] BBKM scheme. It can be considered that when the number of members in a single group is less than 8, our scheme has the lowest communication cost in the fusion phase. Regardless of the number of members, the communication cost of our scheme in the certificate update phase is 0. Zhang et al.'s [5] scheme relies on seeds initially deployed on TPDs during updates, which has high hardware requirements.

In the group fusion phase, a large number of encrypted shares are transmitted between UAVs, resulting in a relatively high overall communication cost of our scheme. However, in a fully distributed scenario, the communication cost caused by such interaction is unavoidable. With the development of communication technologies, 5G/B5G networks can transmit data at a faster speed and reduce transmission time. This means that more data can be transmitted within the same time, lowering the communication cost per unit of data. In general, compared with the group autonomous fusion and efficient certificate update that the scheme can achieve, the current communication cost is acceptable.

VIII. Conclusion

In this study, we propose a distributed management scheme for UAV groups to address the limitations of relying on a trusted authority during group fusion. First, by leveraging mobile proactive secret sharing and threshold signature, the proposed scheme enables UAVs to autonomously update their subsecrets, thereby achieving secure and seamless group fusion. Second, a lightweight certificate self-updating mechanism is designed to support efficient certificate updating

TABLE VIII
COMPARISON OF COMMUNICATION COSTS OF DIFFERENT SCHEMES

Schemes	Group Fusion (bytes)		Certificate Update (bytes)	
	UAV_{header}/GCU	UAV_{join}	UAV_{header}/GCU	UAV_{join}
BBKM [17]	$97(n_1+n_2)$	165	$362(n_1+n_2)+458$	288
PPAS [11]	$145(n_1+n_2)$	$213(n_1+n_2)-100$	−	−
TAGKA [5]	0	$444(n_1+n_2)$	0	144
Ours	0	$48[n_1(n_1+n_2)-1]$	0	0

−: Not supported.

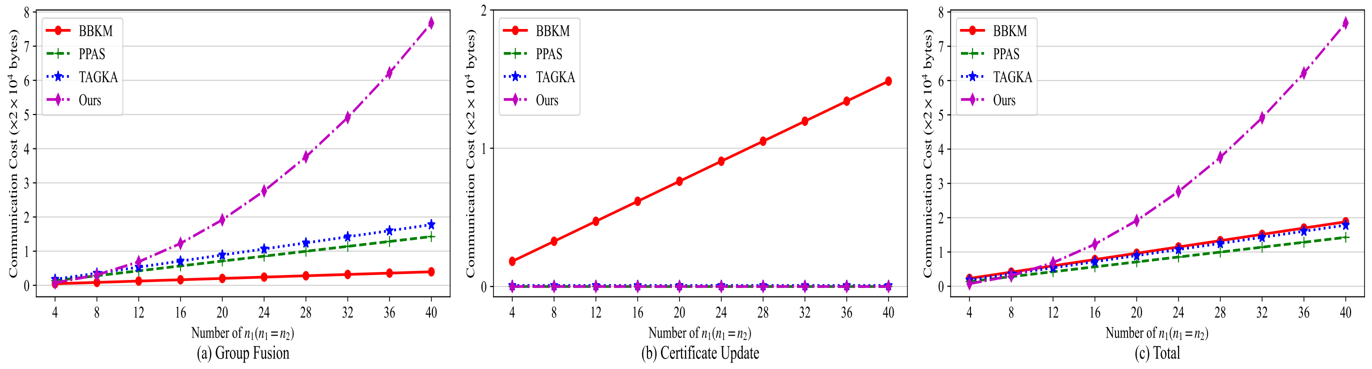


Fig. 12. Communication cost in each phase of our proposed scheme.

and ensure the continuous availability of certificates after group fusion. Formal analysis demonstrates the correctness and security of the scheme under the random oracle model, and experimental results show that it achieves superior computational efficiency compared with existing schemes.

However, the current scheme still has some limitations, and future work will be carried out from the following three aspects. First, to further expand the threat model and study the scenario involving internal attackers. Second, to design a more efficient protocol to reduce unnecessary cost in the process of share sharing. Third, to explore how to efficiently realize the separation of UAV groups so as to better adapt to the needs of flight tasks.

REFERENCES

[1] C. Lin, G. Han, X. Qi, J. Du, T. Xu, and M. Martínez-García, "Energy-optimal data collection for unmanned aerial vehicle-aided industrial wireless sensor network-based agricultural monitoring system: A clustering compressed sampling approach," *IEEE Transactions on Industrial Informatics*, vol. 17, no. 6, pp. 4411–4420, 2021.

[2] R. Liu, A. Liu, Z. Qu, and N. N. Xiong, "An uav-enabled intelligent connected transportation system with 6g communications for internet of vehicles," *IEEE Transactions on Intelligent Transportation Systems*, vol. 24, no. 2, pp. 2045–2059, 2023.

[3] H. Tan, W. Zheng, and P. Vijayakumar, "Secure and efficient authenticated key management scheme for uav-assisted infrastructure-less iovs," *IEEE Transactions on Intelligent Transportation Systems*, vol. 24, no. 6, pp. 6389–6400, 2023.

[4] Z. Su, Y. Wang, Q. Xu, and N. Zhang, "Lvbs: Lightweight vehicular blockchain for secure data sharing in disaster rescue," *IEEE Transactions on Dependable and Secure Computing*, vol. 19, no. 1, pp. 19–32, 2022.

[5] Z. Zhang, X. Li, Y. Wang, Y. Miao, X. Liu, J. Weng, and R. H. Deng, "Tagka: Threshold authenticated group key agreement protocol against member disconnect for uanet," *IEEE Transactions on Vehicular Technology*, vol. 72, no. 11, pp. 14987–15001, 2023.

[6] W. Xu, L. Xiang, T. Zhang, M. Pan, and Z. Han, "Cooperative control of physical collision and transmission power for uav swarm: A dual-fields enabled approach," *IEEE Internet of Things Journal*, vol. 9, no. 3, pp. 2390–2403, 2022.

[7] S. Javaid, N. Saeed, Z. Qadir, H. Fahim, B. He, H. Song, and M. Bilal, "Communication and control in collaborative uavs: Recent advances and future trends," *IEEE Transactions on Intelligent Transportation Systems*, vol. 24, no. 6, pp. 5719–5739, 2023.

[8] P. Tedeschi, S. Sciancalepore, and R. Di Pietro, "Ppca-privacy-preserving collision avoidance for autonomous unmanned aerial vehicles," *IEEE Transactions on Dependable and Secure Computing*, vol. 20, no. 2, pp. 1541–1558, 2022.

[9] S. Wang, Y. Wang, X. Bai, and D. Li, "Communication efficient, distributed relative state estimation in uav networks," *IEEE Journal on Selected Areas in Communications*, vol. 41, no. 4, pp. 1151–1166, 2023.

[10] G. Bansal, Naren, V. Chamola, and B. Sikdar, "Shots: Scalable secure authentication-attestation protocol using optimal trajectory in uav swarms," *IEEE Transactions on Vehicular Technology*, vol. 71, no. 6, pp. 5827–5836, 2022.

[11] M. A. Khan, I. Ullah, A. Alkhalifah, S. U. Rehman, J. A. Shah, M. I. Uddin, M. H. Alsharif, and F. Algarni, "A provable and privacy-preserving authentication scheme for uav-enabled intelligent transportation systems," *IEEE Transactions on Industrial Informatics*, vol. 18, no. 5, pp. 3416–3425, 2022.

[12] A. Mansour, K. M. Malik, A. Alkaff, and H. Kanaan, "Alms: Asymmetric lightweight centralized group key management protocol for vanets," *IEEE Transactions on Intelligent Transportation Systems*, vol. 22, no. 3, pp. 1663–1678, 2021.

[13] G. Bansal and B. Sikdar, "S-maps: Scalable mutual authentication protocol for dynamic uav swarms," *IEEE Transactions on Vehicular Technology*, vol. 70, no. 11, pp. 12088–12100, 2021.

[14] G. Cheng, J. Huang, Y. Wang, J. Zhao, L. Kong, S. Deng, and X. Yan, "Conditional privacy-preserving multi-domain authentication and pseudonym management for 6g-enabled iov," *IEEE Transactions on Information Forensics and Security*, pp. 1–1, 2023.

[15] Y. Tan, J. Wang, J. Liu, and N. Kato, "Blockchain-assisted distributed and lightweight authentication service for industrial unmanned aerial vehicles," *IEEE Internet of Things Journal*, vol. 9, no. 18, pp. 16928–16940, 2022.

[16] Z. Xu, W. Liang, K.-C. Li, J. Xu, A. Y. Zomaya, and J. Zhang, "A time-

sensitive token-based anonymous authentication and dynamic group key agreement scheme for industry 5.0," *IEEE Transactions on Industrial Informatics*, vol. 18, no. 10, pp. 7118–7127, 2022.

[17] Y. Tan, J. Liu, and N. Kato, "Blockchain-based key management for heterogeneous flying ad hoc network," *IEEE Transactions on Industrial Informatics*, vol. 17, no. 11, pp. 7629–7638, 2021.

[18] Y. Tian, Z. Wang, J. Xiong, and J. Ma, "A blockchain-based secure key management scheme with trustworthiness in dwsns," *IEEE Transactions on Industrial Informatics*, vol. 16, no. 9, pp. 6193–6202, 2020.

[19] J. Wang, Z. Jiao, J. Chen, X. Hou, T. Yang, and D. Lan, "Blockchain-aided secure access control for uav computing networks," *IEEE Transactions on Network Science and Engineering*, pp. 1–14, 2023.

[20] J. Zhang, J. Cui, H. Zhong, Z. Chen, and L. Liu, "Pa-crt: Chinese remainder theorem based conditional privacy-preserving authentication scheme in vehicular ad-hoc networks," *IEEE Transactions on Dependable and Secure Computing*, vol. 18, no. 2, pp. 722–735, 2021.

[21] G. Liu, H. Li, N. Wang, T. Xiang, and Y. Liu, "Degkm: Decentralized group key management for content push in integrated networks," *IEEE Transactions on Dependable and Secure Computing*, pp. 1–17, 2024.

[22] H. Yıldız, M. Cenk, and E. Onur, "Plgakd: A puf-based lightweight group authentication and key distribution protocol," *IEEE Internet of Things Journal*, vol. 8, no. 7, pp. 5682–5696, 2021.

[23] P. Roychoudhury, B. Roychoudhury, and D. K. Saikia, "A secure device-to-device communication scheme for massive machine type communication," *Computers Security*, vol. 108, p. 102370, 2021.

[24] T.-F. Lee, X. Ye, and S.-H. Lin, "Anonymous dynamic group authenticated key agreements using physical unclonable functions for internet of medical things," *IEEE Internet of Things Journal*, vol. 9, no. 16, pp. 15336–15348, 2022.

[25] Z. Ma, J. Zhang, Y. Guo, Y. Liu, X. Liu, and W. He, "An efficient decentralized key management mechanism for vanet with blockchain," *IEEE Transactions on Vehicular Technology*, vol. 69, no. 6, pp. 5836–5849, 2020.

[26] M. Tanveer, A. U. Khan, N. Kumar, and M. M. Hassan, "Ramp-iod: A robust authenticated key management protocol for the internet of drones," *IEEE Internet of Things Journal*, vol. 9, no. 2, pp. 1339–1353, 2022.

[27] D. Schultz, B. Liskov, and M. Liskov, "Mpss: mobile proactive secret sharing," *ACM Transactions on Information and System Security (TISSEC)*, vol. 13, no. 4, pp. 1–32, 2010.

[28] A. Herzberg, S. Jarecki, H. Krawczyk, and M. Yung, "Proactive secret sharing or: How to cope with perpetual leakage," in *Advances in Cryptology — CRYPT0' 95* (D. Coppersmith, ed.), (Berlin, Heidelberg), pp. 339–352, Springer Berlin Heidelberg, 1995.

[29] M. Abdalla, P.-A. Fouque, and D. Pointcheval, "Password-based authenticated key exchange in the three-party setting," in *Public Key Cryptography - PKC 2005* (S. Vaudenay, ed.), (Berlin, Heidelberg), pp. 65–84, Springer Berlin Heidelberg, 2005.

[30] D. Pointcheval and J. Stern, "Security arguments for digital signatures and blind signatures," *Journal of cryptology*, vol. 13, pp. 361–396, 2000.

[31] M. Bellare and G. Neven, "Multi-signatures in the plain public-key model and a general forking lemma," in *Proceedings of the 13th ACM Conference on Computer and Communications Security*, CCS '06, (New York, NY, USA), p. 390–399, Association for Computing Machinery, 2006.

[32] Amovlab, "Prometheus autonomous uav opensource project." https://github.com/amov-lab/Prometheus.

[33] "Miracl core." https://github.com/miracl/core.

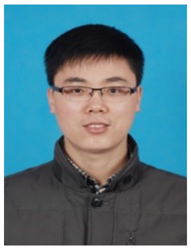
[34] P. Vijayakumar, M. Azees, A. Kannan, and L. Jegatha Deborah, "Dual authentication and key management techniques for secure data transmission in vehicular ad hoc networks," *IEEE Transactions on Intelligent Transportation Systems*, vol. 17, no. 4, pp. 1015–1028, 2016.

Fengqun Wang was born in Anhui Province, China, in 1996. He received his Ph.D. degree in computer science from Anhui University in 2024. He is currently a lecturer of the School of Computer Science and Technology at Anhui University. His research interests include IoT security, blockchain, and applied cryptography. He has multiple scientific publications in reputable journals (e.g., IEEE Transactions on Information Forensics and Security, IEEE Transactions on Dependable and Secure Computing, IEEE Transactions on Network and Service Management, IEEE Transactions on Industrial Electronics).

Manting Gan is now a research student in the School of Computer Science and Technology, Anhui University. Her research focuses on the security of the Unmanned Aerial Vehicles (UAVs).

Hong Zhong was born in Anhui Province, China, in 1965. She received her PhD degree in computer science from University of Science and Technology of China in 2005. She is currently a professor and Ph.D. supervisor of the School of Computer Science and Technology at Anhui University. Her research interests include applied cryptography, IoT security, vehicular ad hoc network, cloud computing security and software-defined networking (SDN). She has over 200 scientific publications in reputable journals (e.g. IEEE Journal on Selected Areas in Communications, IEEE Transactions on Parallel and Distributed Systems, IEEE Transactions on Mobile Computing, IEEE Transactions on Dependable and Secure Computing, IEEE Transactions on Information Forensics and Security), academic books and international conferences.

Qingyang Zhang was born in Anhui Province, China, in 1992. He received his B. Eng. degree and Ph.D. degree in computer science from Anhui University in 2021. He is currently an associate professor of School of Computer Science and Technology at Anhui University. His research interest includes edge computing, computer systems, and security. He has over 30 scientific publications in reputable journals (e.g. Proceedings of the IEEE, IEEE Transactions on Dependable and Secure Computing, IEEE Transactions on Information Forensics and Security, IEEE Transactions on Parallel and Distributed Systems, IEEE Transactions on Computers) and international conferences.



Jie Cui (Senior Member, IEEE) was born in Henan Province, China, in 1980. He received his Ph.D. degree in University of Science and Technology of China in 2012. He is currently a professor and Ph.D. supervisor of the School of Computer Science and Technology at Anhui University. His current research interests include applied cryptography, IoT security, vehicular ad hoc network, cloud computing security and software-defined networking (SDN). He has over 150 scientific publications in reputable journals (e.g. IEEE Transactions on Dependable and Secure Computing, IEEE Transactions on Information Forensics and Security, IEEE Journal on Selected Areas in Communications, IEEE Transactions on Mobile Computing, IEEE Transactions on Parallel and Distributed Systems, IEEE Transactions on Computers), academic books and international conferences.

Debiao He received his Ph.D. degree in applied mathematics from School of Mathematics and Statistics, Wuhan University, Wuhan, China in 2009. He is currently a professor of the School of Cyber Science and Engineering, Wuhan University, Wuhan, China and the Shanghai Key Laboratory of Privacy Preserving Computation, MatrixElements Technologies, Shanghai 201204, China. His main research interests include cryptography and information security, in particular, cryptographic protocols. He has published over 100 research papers in refereed international journals and conferences, such as IEEE Transactions on Dependable and Secure Computing, IEEE Transactions on Information Security and Forensic, and Usenix Security Symposium. He is the recipient of the 2018 IEEE Sysems Journal Best Paper Award and the 2019 IET Information Security Best Paper Award. His work has been cited more than 10000 times at Google Scholar. He is in the Editorial Board of several international journals, such as Journal of Information Security and Applications, Frontiers of Computer Science, and Human-centric Computing Information Sciences.