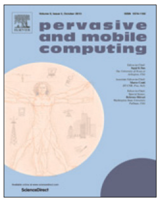
Contents lists available at ScienceDirect

Pervasive and Mobile Computing

journal homepage: www.elsevier.com/locate/pmc



# HCALA: Hyperelliptic curve-based anonymous lightweight authentication scheme for Internet of Drones

Aymen Dia Eddine Berini [*], Mohamed Amine Ferrag, Brahim Farou, Hamid Seridi

*LabSTIC Laboratory, 8 May 1945 University, Guelma, 24000, Algeria*

ARTICLE INFO

*Article history:*
Received 17 December 2022
Received in revised form 8 April 2023
Accepted 24 April 2023
Available online 25 April 2023

*Keywords:*
Drones
User authentication
Blockchain
Security
Cryptography
AVISPA

ABSTRACT

A drone is often called an "Unmanned Aerial Vehicle" or UAV. It is utilized in various civilian and military applications, including agriculture, surveillance, and delivery of packages. A promising idea for enhancing the safety and quality of drone flight is to build the Internet of Drones (IoD), in which drones are used to collect sensitive data, which is then communicated in real-time to external user ($U_i$) through the Ground Station Server (*GSS*). Before deployment, the GSS and all drones are registered with Control Room (*CR*), a central authority, which is a trusted authority. To ensure secure and reliable communications, an efficient and secure authentication scheme is needed to enable users and drones to authenticate each other and share a session key. Furthermore, because drones generally have small batteries and limited memory capacity, efficient and lightweight security techniques are suitable for them. Many schemes to secure IoD environments have been proposed recently; however, some were proven as insecure, and some degraded efficiency. In this work, we focus on developing a novel blockchain-based authentication scheme, called HCALA, on protecting the communication between an external user and drone utilizing Hyperelliptic Curve Cryptography (HECC). To evaluate the viability and efficacy of HCALA, We employ the extensively used Random Oracle Model (ROM) and formal security verification using a software tool called AVISPA, which is used to validate the internet security protocols. HCALA is also examined by utilizing informal security analysis techniques, demonstrating that the proposed protocol can withstand several well-known active and passive adversary attacks. It also shows that HCALA is more efficient regarding different parameters, according to the performance comparison. Compared to previous similar schemes, the security and functionality aspects are improved, and the computation, communication costs, and energy consumption are reduced.

## 1. Introduction

The IoD is a system in which a group of drones forms an aerial mesh network to communicate and collect sensitive information through sensors [1,2]. The information is then sent to the GSS. An external user can monitor and access some drones through their mobile devices in specific flying zones through the GSS, provided they are authorized to access the drones. The GSS is considered a trusted party, meaning attackers cannot compromise it. The drones can communicate with

* Corresponding author.
*E-mail addresses:* berini.aymen@univ-guelma.dz (A.D.E. Berini), ferrag.mohamedamine@univ-guelma.dz (M.A. Ferrag), farou.brahim@univ-guelma.dz (B. Farou), seridi.hamid@univ-guelma.dz (H. Seridi).

https://doi.org/10.1016/j.pmcj.2023.101798
1574-1192/© 2023 Elsevier B.V. All rights reserved.

each other, with users, and among themselves and users through the trusted GSS. The wireless nature of communication in IoD systems poses a risk to the privacy and security of the entities involved, such as drones, users, and GSS [3,4]. Different attacks can occur, such as impersonation, man-in-the-middle, drone capture, password guessing, replay, and insider attacks [5,6]. To prevent these risks, we need a way to ensure that only authorized and registered entities are securely communicating with each other. In particular, mutual authentication can be used to verify the identities of the communication parties before they exchange secrets and sensitive information over an insecure channel [7–9].

Due to the resource constraints of the drone device, several essential factors must be considered when designing an Authentication and Key Agreement (AKA) scheme. First, the designed protocol must be resistant to a variety of attacks, including drone capture, replay attacks, impersonation, and stolen verifier [5]. In addition, as drones have limited energy resources and flight time, the authentication should be computationally efficient and incur minimal expenses. Consequently, typical public cryptosystems such as RSA are unsuitable for IoD contexts [10–12].

Compared to Rivest-Shamir-Adleman (RSA) and Elliptic Curve Cryptography (ECC), HECC is the superior solution because it can consume less bandwidth, energy, and processing resources. HECC with an 80-bit key provides the same level of security as ECC and RSA with 160-bit and 1024-bit keys, respectively [13]. HECC is ideally suited for lightweight authentication and should be meticulously developed to withstand all cryptographic threats. In addition, the security of public key cryptography is contingent upon the difficulty of solving discrete logarithm problems [14].

Furthermore, to our knowledge, the bulk of IoD authentication schemes are either susceptible to known attacks or fail to achieve critical security requirements such as user anonymity or perfect forward secrecy.

To address these issues, we develop an efficient AKA scheme for IoD applications that provides resilience against multiple threats and is far less computationally expensive than competing schemes. The proposed scheme is drone-independent, which means that it can work with any type of drone. Therefore, the system can be easily adopted regardless of the specific drone type used. The only requirement is that the computational capacity is sufficient to support the processing needs of the proposed scheme.

Specifically, our contributions consist of the following:

- We propose a lightweight AKA scheme to secure user-drone communication for IoD, called HCALA. The HCALA scheme utilizes a HECC, exclusive-OR operation (XOR), and a hash function (SHA-1). HCALA is supported by the blockchain solution.
- HCALA renders the revocation, or reissue phases, password update, and enables the deployment of a new drone whenever a drone is physically attacked by an adversary or runs out of power.
- We consider the (DY) threat model and (CK) adversary model provides the most capability to an opponent attempting to compromise the proposed scheme's security.
- Formal security verification utilizing the AVISPA tool was used to assure the HCALA scheme's security and demonstrate the protocol's robustness against several attacks [15]. In addition, formal security utilizing "ROM" and informal security analysis demonstrate the robustness of HCALA scheme against different active adversary attacks.
- Finally, HCALA is compared with similar existing schemes regarding security features, computational, communication costs, and energy consumption.

The remaining sections of this article are structured as follows. In Section 2, we review some relevant literature on existing schemes, and in Section 3, we define the preliminaries. The network model and security requirements that the HCALA scheme must meet are described in Section 4. The HCALA scheme is provided in Section 5, followed by security analysis and performance evaluation in Sections 6 and 7, respectively. Section 8 concludes the paper.

## 2. Related work

In recent years, research has focused on providing safe and efficient communication among drones. In particular, in IoD environment, private data is exchanged between drones over unsecured wireless communication channels, rendering them vulnerable to a variety of security risks. Authors in [16] introduced the first AKA-based method that assures key agreement between users and nodes without gateway node, according to the authors of [17]. The scheme is lightweight since it uses just hash functions and bit-wise XOR operations. Farash et al. [18] claim that the scheme [16] is vulnerable to several attacks including man-in-the-middle (MTM) attacks, node anonymity and traceability, and node impersonation attack. As a result, The enhanced protocol proposed by Farash et al. overcomes the security issues raised by Turkanović et al. scheme [16]. The scheme described in [18] is still vulnerable to offline password guessing, user impersonation, and smart card loss attacks. Furthermore, against gateway nodes, it is unable to preserve user anonymity or secure session key secrecy. The authors in [19] suggested an efficiency AKA scheme based on smart cards that is adaptable to multiple gateway scenarios to solve the shortcomings of the protocol in [18]. However, the scheme does not eliminate the dangers of denial-of-service attacks or smart card theft, nor does it guarantee user anonymity. Challa et al. [20] proposed a user authentication protocol that was based on ECC. However, Jia et al. [21] discovered significant weaknesses in the scheme [20] which left it susceptible to impersonation attacks. Additionally, the computational and communication costs associated with [20] were found to be prohibitively high, rendering it impractical for deployment in many real-world scenarios.

In the concept of IoD network, researchers have made efforts and proposed many security protocols to ensure secure communication. Wazid [3] provides a taxonomy of these protocols, which includes key management, access control, user authentication, identity privacy, and intrusion detection. In 2018, Wazid et al. [22] designed a lightweight AKA protocol for authenticating users and flying drones. Their scheme supports mutual authentication between the user and the drone. Because just hash functions and fuzzy extractors are used, the scheme is extremely lightweight, with minimal memory overhead and computational and communication expenses. While the authors cover a variety of security concerns, they fail to emphasize the need for forward and backward perfect secrecy, as well as non-repudiation, which are critical requirements for sensitive drone operations.

In [23] Chen et al. presented Mutual Authentication DAA (MA-DAA), an improved Direct Anonymous Attestation (DAA) cryptographic scheme that uses asymmetric bi-linear pairing for mutual authentication between network-connected UAVs and the GSS. In comparison to conventional DAA schemes, the MA-DAA system is ideally suited to the low bandwidth and computational capabilities of UAV networks. Their approach, on the other hand, is based on Trusted Platform Modules (TPMs), which are specialized, expensive security coprocessors that must be integrated into systems, resulting in increased costs. Moreover, the security of the scheme is not substantiated by formal proof. Tanveer et al. [24] introduced the Lightweight AKE Protocol for IoD Environment (LAKE-IoD), which employs the AEGIS authenticated encryption algorithm, bit-wise XOR, and a hash function (SHA256). Their scheme consists of phases for revocation or reissue, dynamic drone deployment, and password update. They used Burrows-Abadi-Needham (BAN) logic to formally analyze the security of their scheme, while the Scyther toolkit was used for simulation, and for informal analysis, they used mathematical assumptions. They claim that their scheme is safe against a variety of security vulnerabilities, including replay and MTM attacks.

In a software-defined UAV network, the PUF-based Authentication for Remote Hovering Devices (PARTH) scheme allows mutual authentication between three levels of entities [25]. For guaranteeing high security in UAVs' sensitive data transmission and authentication concerns, their system calculated two session keys. They claim that their scheme is resistant to various attacks such as MTM attacks, node capture attacks, and replay attacks. Another scheme for securing the IoD environment is Temporal Credential-Based Anonymous Lightweight Authentication Scheme (TCALAS) [26]. The authors combine a cryptographic, fuzzy extractor method, bit-wise XOR operation, and hash function in their scheme. According to the analysis in this article, their protocol for drone-to-drone key management is not considered, as it is limited to a single flying zone and cannot be scaled. Ali et al. [5] demonstrate that "TCALAS" is vulnerable to stolen verifier attacks and lacks untraceability . It is claimed that after obtaining the verifier, an attacker can spoof any of the GSS, drone, and user. They also proposed "iTCALAS", an enhanced scheme to overcome these concerns and provide scalability for IoD.

Cho et al. [27] developed SENTINEL (Secure and Efficient autheNTIcation for uNmanned aErial vehicLes) authentication framework to address different security concerns posed by unauthorized drones in IoD environment. Mutual authentication between drones and GSS is provided by the scheme. Even though their scheme reduces computational and communication overheads, it is vulnerable to "ESL attack under the CK-adversary model". Additionally, untraceability and anonymity are not preserved in their scheme. In [10], Ever presented a secure authentication framework based on ECC. In the hierarchical wireless sensor network architecture of this scheme, drones are regarded as mobile sinks. Sensor nodes, mobile sinks, and cluster heads can all have one-time user authentication (UAVs). However, the ESL attack under the CK-adversary model is a weakness of their scheme, and their scheme lacks the anonymity and untraceability characteristics that Cho et al. [27] provide.

Authors in [28] designed a blockchain-based access control protocol in the IoD environment. They used an ECC-based Diffie–Hellman key exchange to create a session key for their first authentication mechanism, which is for drone-to-drone authentication; their second authentication method is for drone-to-GSS authentication. More recent work by Bera et al. [29] designed (BSD2C-IoD), a blockchain-based secure data delivery and collecting scheme for 5G-enabled IoD environments that enables authentication between drones and their associated GSS in each flying region. The authors claim that the suggested framework is secure against many IoD attacks. On the other hand, the scheme has a high computational cost. In [30], a lightweight authentication protocol for the IoD was proposed by nikooghadam et al. They asserted that their protocol is safe and resistant to numerous threats. On the other hand, their scheme is vulnerable to control server impersonation, user impersonation, privileged insider attacks, secret parameter leaking, drone impersonation, and does not guarantee user anonymity. Authors in [31] suggested a three-party authentication scheme in an IoD environment. This scheme utilizes symmetric encryption and a one-way hash function. However, we discover that the scheme [31] is susceptible to privileged insider attacks, drone capture attacks, and impersonation attacks.

To implement an authentication scheme for the IoD system, Tanveer et al. [32] used a combination of elliptic curve cryptography, a hash function, and a dedicated authenticated encryption algorithm. First, the user's identity is validated across seven steps, and then, for subsequent communications, a secret key is established between the user and the drone. The authors assert that their proposed security scheme accomplishes better performance and satisfies the defined security requirements. However, the proposed security scheme does not ensure dynamic privacy protection. Tanveer et al. [33] developed the AKA scheme to secure communication between a remote user and a drone, named RUAM-IoD, based on the AES-CBC-256 encryption, hash function(SHA-256), ECC, and XOR operation. According to the authors, the scheme is resistant to various security concerns, including stolen smart device, biometric and password change, drone capture, replay and MTM attacks. On the other hand, communication and computing expenses are somewhat high.

In recent work, Javed et al. [34] abandon the blockchain-based authentication protocol and HEC for IoT drones. The blockchain functions as a certification authority in this scheme and transactions are defined as certificates. This measure

**Table 1**
A comparative study of the most relevant AKA schemes by year, cryptography techniques, limitations, and characteristics.

| Scheme | Year | Cryptography techniques | Limitations & Characteristics |
|---|---|---|---|
| Challa et al. [20] | 2017 | ■ ECC<br>■ Bit-wise XOR operation<br>■ Hash function(SHA160) | ■ Exposed to privilege insider and stolen device. |
| Wizid et al. [22] | 2018 | ■ Bit-wise XOR operation<br>■ Hash function(SHA160) | ■ It is vulnerable to stolen-verifier attacks, user impersonation, and drone impersonation.<br>■ Exposed to session key leakage attack, server broadcasting and traceability issues. |
| Srinivas et al. [26] | 2019 | ■ Hash function<br>■ Biometric fuzzy extractor | ■ Vulnerable to user impersonation, identity guessing, and device impersonation attacks.<br>■ The cost of computation is slightly more expensive. |
| Tanveer et al. [24] | 2020 | ■ The authenticated encryption scheme (AEGIS)<br>■ Bit-wise XOR operation<br>■ Hash function (SHA256) | ■ The cost of computation is a little high.<br>■ Their scheme lacks support for blockchain solutions. |
| Ali et al. [5] | 2020 | ■ Advanced encryption standard (AES)<br>■ Bit-wise XOR operation<br>■ Hash function(SHA160) | ■ Exposed to forgery, Privilege Insider, Stolen Smart Device, Server Impersonation, and Denial-of-Service (DoS) attacks.<br>■ Perfect Forward Secrecy and key freshness features are not rendered. |
| Ever et al. [10] | 2020 | ■ One-way hash functions.<br>■ Bilinear pairings.<br>■ ECC.<br>■ Symmetric key encryption | ■ Susceptible to ESL attack under the CK-adversary model.<br>■ Absence of untraceability and anonymity preservation properties.<br>■ High costs of communication and computing. |
| Bera et al. [28] | 2020 | ■ ECC<br>■ Hash function(SHA256)<br>■ Symmetric key encryption<br>■ Blockchain consensus algorithms | ■ Vulnerable to user anonymity attack.<br>■ Computation is high. |
| Nikooghadam et al. [30] | 2021 | ■ One-way hash functions<br>■ Bit-wise XOR operations<br>■ ECC | ■ Vulnerable to control server impersonation, drone impersonation, user impersonation attack, privileged insider attack.<br>■ Does not offer user anonymity and untraceability. |
| Hussain et al. [31] | 2021 | ■ One-way hash functions<br>■ Symmetric key encryption | ■ Exposed to privileged insider attacks.<br>■ Vulnerable to impersonation attacks.<br>■ Not able to fend off drone capture attempts. |
| Tanveer et al. [32] | 2021 | ■ ECC<br>■ One-way hash functions<br>■ AEGIS<br>■ Bit-wise XOR operations | ■ Does not ensure dynamic privacy protection.<br>■ The use of blockchain technology is not endorsed. |
| Tanveer et al. [33] | 2022 | ■ Hash function (SHA-256)<br>■ Bit-wise XOR operations<br>■ ECC<br>■ AES-CBC-256 encryption | ■ Communication and computing expenses are somewhat high.<br>■ The blockchain solution is not supported. |
| HCALA scheme | / | ■ HEC<br>■ One-way hash functions<br>■ Bit-wise XOR operation<br>■ Symmetric key encryption<br>■ Blockchain consensus algorithms | ■ It provides protection against various known attacks.<br>■ It provides formal security analysis and communication cost is very low. |

is intended to reduce maintenance expenses while maintaining a high level of communication security. The authors determined that the proposed protocol offers protection against prevalent attacks in drone IoT networks while being more efficient than competing solutions regarding computational and communication overhead.

Previous research motivates us to address various security vulnerabilities in existing protocols. To achieve this objective, we want to create (HCALA), a new secure, lightweight user authentication scheme suitable for the IoD environment. The protocol is supported by the blockchain solution, based on a hyperelliptic curve, and is significantly more secure and efficient. Another notable characteristic is the small key size (80 bits), which is half of what the elliptic curve requires (160 bits). Table 1 provides a summary of competing for existing authentication/access control schemes and the proposed scheme for the IoD environment in terms of their cryptographic techniques, advantages, and properties.

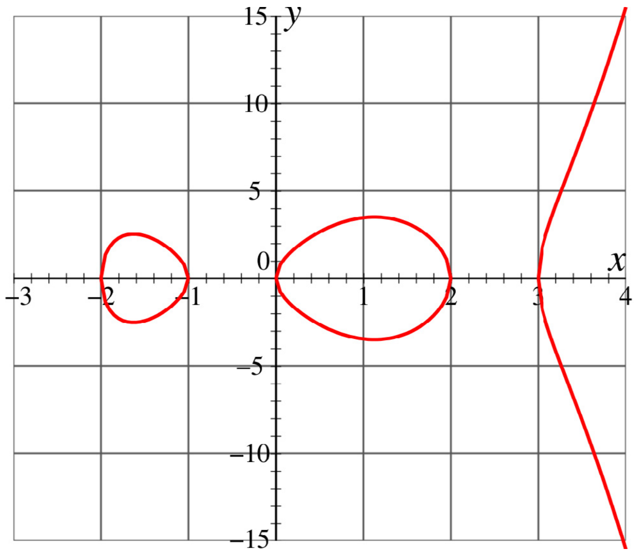


**Fig. 1.** Hyperelliptic curve of genus 2 [39].

## 3. Preliminaries

### 3.1. Hyperelliptic Curve (HEC)

The HEC is a class of algebraic curves [35] considered by [36] as a generalized version of elliptic curves (EC). However, HEC points cannot be obtained with a group. The additive Abelian group is obtained from a divisor or computed by the HEC. The advantage of HEC regarding RSA, EC, and bilinear pairing is that it can keep the same security level with a smaller parameter size [37].

EC refers to a curve with a genus value of 1. Fig. 1 shows a HEC with a genus higher than 1. Similarly, for the (genus $D = 1$), the group order of the finite field ($Fq$) needed 160-bits long operands, requiring at least $g.\log_2(q) = 2^{160}$. Similarly, curves with genus 2 require 80-bit long operands, while curves with genus 3 require 54-bit long operands [38].

A hyper elliptic curve "$C$" of genus $g$ ($g > 1$) over $F$ is a collection of solutions $(x, y) \in F \times F$ to the following equation:

$$C : y^2 + h(x)y = f(x) \tag{1}$$

The $D$ is a divisor of an HEC, which is a finite sum of points, and it is written as:

$$D = \sum_{p_i \in C} m_i p_i,\ m_i \in Z \tag{2}$$

### 3.2. Complexity assumptions

#### 3.2.1. Assumptions of Hyperelliptic Curve Discrete Logarithm Problem (HECDLP)

For HECDLP, we make the following suppositions:

- $\eta$ belongs to $\{1, 2, 3, \ldots\ldots\ldots, q-1\}$.
- The probability calculation $\eta$ from $R = \eta.D$ is negligible.

#### 3.2.2. Computational Diffie–Hellman assumption of Hyperelliptic Curve (HCCDHP)

We assume:

- $\eta$ and $\vartheta$ belong to $\{1, 2, 3, \ldots\ldots\ldots\ldots, q-1\}$.
- The probability computation of $\eta$ and $\vartheta$ from $\Gamma = \eta.\vartheta.D$ is negligible.

### 3.3. Encryption and Decryption Using HEC Diffie–Hellman(HEC-DH)

In the encryption and decryption process of this scheme [40], the two parties first agree on the HEC parameters, which include a large prime number (p) and an HEC over a finite field (Fq) denoted as C(Fq), both of which are publicly known.

In addition to these standard parameters, the parties also agree on other parameters $\{q, Fq, C(F_q), D, n\}$. The message to be encrypted is denoted as (MSG).

After agreeing on the parameters, the two parties now share a common key $K = (ab)D$, which is the reduced divisor of the Jacobian $JC(F_q)$. Party A uses this key to encrypt the message (MSG) to produce the ciphertext $(C_T)$ and sends it to Party B. Upon receiving the ciphertext $(C_T)$, and Party B uses the shared key to decrypt it and obtain the original message $(MSG)$ as follows:

**Algorithm 1:** HEC based encryption (Party A):

```
   Input: Message (MSG), domain parameters and public key of party B ($PK_B$)
   Output: Cipher text ($C_T$)
1  Choose a random integer $a \in [1, q- 1]$
2  Calculate: $a.PK_B$
3  Return the cipher text as $C_T = MSG + a.PK_B$
```

**Algorithm 2:** HEC based decryption (Party B):

```
   Input: Cipher text ($C_T$), domain parameters, public key of party A ($PK_A$), and private key of Party B ($PR_B$)
   Output: Message ($MSG$)
1  Calculate: $PR_B.PK_A$
2  Return the cipher text as $MSG = C_T - PR_B.PK_A$
```

The time complexity of HEC-DH is given by:

$$T_{HEC-DH}(m) = C_e 2^2(2m^2 - 1) = O(2^{m^2})$$

$C_e$ is the time needed to carry out single multiple addition operation, and $m$ is the public keys shared between parties A and B.

### 3.4. Blockchain technology

The blockchain is a collection of blocks connected by their hash values, with each block consisting of a block header and a block payload. The permission model can be used to categorize a blockchain network and determine who can maintain it (e.g., publish blocks). Permissionless indicates that anyone can create a new block. It is permissioned if only certain people are permitted to publish blocks. A permissioned blockchain network resembles a corporate intranet, while a permissionless blockchain network resembles the public internet, where anyone can participate [6,41,42]. As shown in Fig. 2, the following three technologies are used to build the blocks of a blockchain-based system: a consensus mechanism, digital signatures, and cryptography hash functions. The concept of blockchain technology refers to a multi-user system that enables users across a network to share confidential data and communicate authentication results with users in different locations. These users decide on an exchange protocol known as the consensus algorithm, which frees them from needing an intermediary and enables them to validate peer-to-peer (P2P) transactions [43,44].

### 3.5. Consensus algorithms

In a distributed P2P network, a consensus algorithm is required for the addition of a block to the blockchain. A consensus algorithm is a decision-making mechanism in a trustless environment among untrustworthy nodes. Consensus is a situation in which all nodes in a distributed network have reached a precise agreement. It is quite difficult to reach a consensus in such an environment. Because the blockchain network is distributed, it is also a difficult job for blockchain technology. Because blockchain is decentralized, there is no central node to assure that all distributed nodes are trustworthy. As a result, some consensus mechanisms are necessary to guarantee that the ledgers in different nodes are consistent. In the blockchain, there are various consensus mechanisms, and some of them are listed below:

- Proof of Work (PoW).
- Proof of Stake (PoS).
- Delegated Proof of Stake (DPoS).
- Ripple Protocol Consensus Algorithm (RPCA).
- Byzantine Fault Tolerance (BFT).
- Practical Byzantine Fault Tolerance (PBFT).

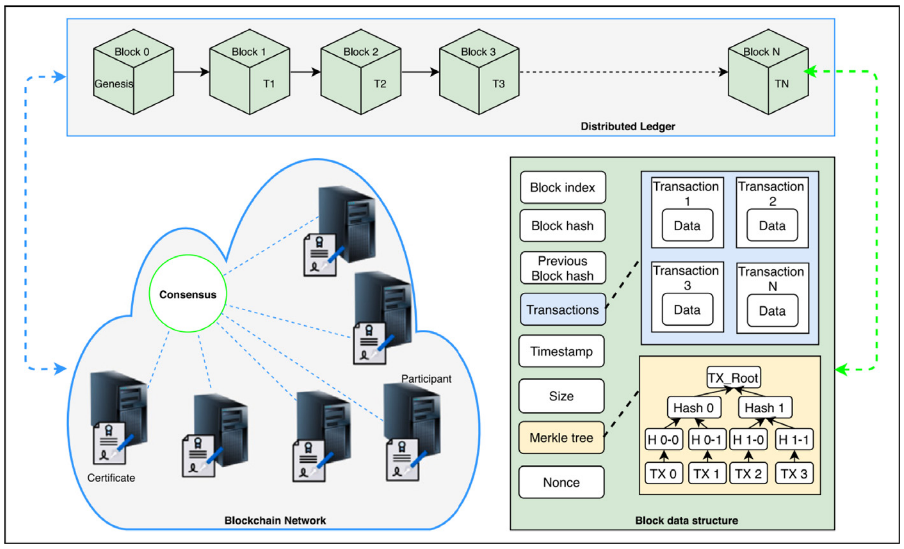


**Fig. 2.** Structure of the blockchain [43].

## 4. System models

In this section, we describe two models (network model and threat model) that are necessary for explaining the working and usability of the designed protocol.

### 4.1. Network model

Fig. 3 depicts the network model of the HCALA scheme for the IoD environment. In this architecture, the ($GSS$) is the trustworthy registration authority in charge of registering all drones and users. The drone is deployed in a specific flying zone to gather information or data from its surroundings. To monitor an IoD environment, an internal user is often seated in the ($CR$). Consider the situation when an external registered user ($U_i$) (for instance, an ambulance driver) would like to know the traffic situation in a certain flying zone. To acquire this data as quickly as possible. The user must connect to the $GSS$ through the Internet and use the $GSS$ to submit a request to the drone deployed in this region. The $GSS$ is used by an $U_i$ and a drone to authenticate each other. The drone and the $U_i$ can establish the session key (secret-key) after authentication and start future communication securely.

### 4.2. Threat model

We consider two threat models in this paper, which are briefly explained below:

#### 4.2.1. Dolev–Yao threat model

In the IoD environment, it is assumed under the well-known (DY) threat model [45] that any messages sent and received through insecure communication channels can be intercepted by an adversary $A$. In addition to that, $A$ has the capability to remove, alter, or insert false messages into the communication channel. Moreover, under this model, the communication entities at the endpoints (the drones) are not trusted in this network.

#### 4.2.2. Canetti and Krawczyk (CK)-adversary model

To further strengthen our user authentication technique, we also apply the broadly Canetti and Krawczyk's adversary model [46], which is more effective than the DY threat model used by existing user authentication schemes in the literature. Following the CK-adversary model, in addition to the capabilities accounted for in the DY threat model, $A$ is additionally able to obtain session states and secret information, including secret keys. Additionally, it is anticipated that the attacker $A$ will be able to seize certain drones physically. Then, by employing the power analysis attack, $A$ will be able to acquire all the secret credentials kept on the captured drones. Consequently, there is a risk of ESL and physical drone capture attacks.

It is presumed that the $GSS$, which provides registration services to other communicating entities, is a trusted network component. Similarly, servers that perform blockchain mining are seen as trustworthy.

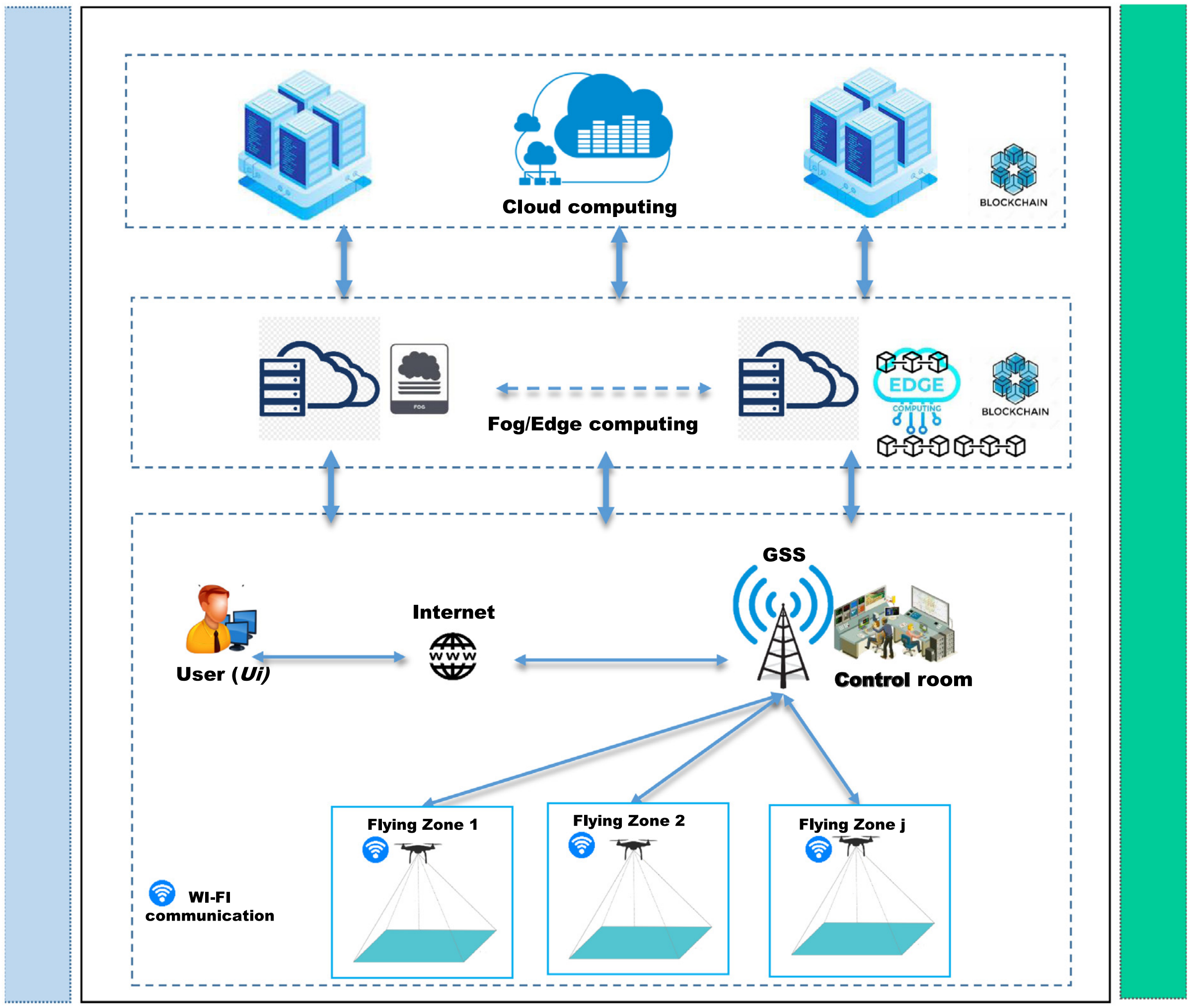


**Fig. 3.** Network model.

**Table 2**
Symbols and their descriptions.

| Symbol | Descriptions |
|---|---|
| $ID_x$ | Real identity of $x$ |
| $PR_x$ | Private key of $x$ |
| $PK_x$ | Public key of $x$ |
| $MID_x$ | Masked identity of $x$ |
| $h(.)$ | hash function |
| $Tw_x$ | Time window generated by $X$ |
| $\oplus$ | Bitwise XOR operation |
| $\Delta T$ | Threshold value for the timestamp |
| $\|$ | Concatenation operation |
| $Rn_2, Rn_3$ | Random numbers |
| $SK_{Dr,i}$ | Session key between user and drone |
| $SK_{i,Dr}$ | Session key between drone and user |
| $D$ | Divisor of hyperelliptic curve |

## 5. Proposed scheme

The proposed scheme (HCALA) is comprised of six parts: (i) setup phase, (ii) registration phase, (iii) login and authentication phase, (iv) password update phase, (v) revocation phase, and (vi) dynamic drone addition phase. Table 2 summarizes the abbreviations and notations used in HCALA.

### 5.1. Setup phase

During this phase, the ($GSS$) that serves as a certificate authority generates both the public parameters of the HCALA scheme and its own private key. In particular, the following are the steps that $GSS$ takes:

- Step 1: chooses a random number $PR_{GSS} \in \{1, 2, \dots, n-1\}$ as his private key.
- Step 2: The GSS computes the public key in the following way: $PK_{GSS} = PR_{GSS}.D$, where D is the divisor on a hyperelliptic curve.
- Step 3: The GSS chooses $h(.)$ as a secure one-way cryptographic hash function. Finally, the parameter set $\{PK_{GSS}, D, n = 2^{80}, h(.)\}$ are published publicity.

### 5.2. Registration phase

In this phase, the ($GSS$) securely registers all drones ($Dr$) in offline mode prior to their deployment. In addition, the registration of the users is also executed securely by the ($GSS$). The following is a detailed overview of the enrolling phase for each particular drone ($Dr$) and user ($U_i$).

#### 5.2.1. Drone registration

The $GSS$ registers all drones prior to deployment in a certain area. Below is a step-by-step illustration of the drone registration procedure.

- Step 1: For each drone, $GSS$ chooses a unique identity $ID_{Dr}$ and calculates the corresponding masked-identity as: $MID_{Dr} = h(ID_{Dr} \parallel PR_{GSS})$.
- Step 2: $GSS$ saves the identity $MID_{Dr}$ in its own database and engraves $\{ID_{Dr_j}, MID_{Dr}\}$ in the memory of the respective drone ($Dr$).

#### 5.2.2. User registration

In an IoD environment, an external user ($U_i$) must register safely at the GSS, either in person or over a secure channel. After successful registration, $U_i$ obtains real-time data from the desired drone deployed in a specific flying zone. To finish registration, GSS and $U_i$ do the procedures below.

- Step 1: Initially, $U_i$ picks a unique identity $ID_i$ and password $Pw_i$ followed by the selection of a random number $\beta \in n$ to computes

  $$A_i = h(h(ID_i \parallel \beta) \oplus h(PW_i \parallel \beta))$$

  Finally, $U_i$ securely transmits the registration request message to $GSS$.
- Step 2: Following receipt of the message, GSS computes $MID_i$ And $B_i$ as

  $$MID_i = h(ID_i \parallel PR_{GSS})$$

  $$B_i = h(MID_i \parallel A_i)$$

  Next, GSS stores $\{ID_i, MID_i, B_i\}$ in its database, and sends $\{MID_i, B_i\}$ to $U_i$ across a secure channel.
- Step 3: Upon receipt from GSS, $U_i$ computes

  $$B'_i = h(MID_i \parallel PW_i) \oplus B_i$$

  $$MID'_i = h(ID_i \parallel PW_i) \oplus MID_i$$

  Finally, $U_i$ stores $\{\beta, B'_i, MID'_i\}$ in its own memory of the device to complete the registration process.

### 5.3. Login and authentication phase

A registered user $U_i$ invokes the login and authentication phase of the HCALA scheme to create a secure channel and become authorized by exchanging a secret key with a deployed drone. This section provides a more in-depth description of this phase:

- Step 1: $U_i$ must first provide his or her identification $ID_i$ and password $PW_i$ before the mobile device will perform the computation $A_i^m = h(h(ID_i \parallel \beta) \oplus h(PW_i \parallel \beta))$ $MID_i^m = h(ID_i, PK_{GSS})$ and $B_i^m = h(MID_i^m \parallel A_i^m)$. Then verifies ($B_i^m \overset{?}{=} B_i$). If the verification is unsuccessful, the process is aborted immediately. Otherwise $U_i$ generates $PR_u$ and a current time window $Tw_1$ to calculate the following:

  $$PK_{GSS} = PR_u.D$$

$$E_i = PR_u.PK_{GSS}$$

$$U_1 = MID_i \oplus h(MID_{GSS} \parallel Tw_1)$$

$$U_2 = MID_{Dr} \oplus h(MID_{GSS} \parallel Tw_1 \parallel E_i)$$

$$U_3 = h(MID_i \parallel MID_{GSS} \parallel MID_{Dr} \parallel E_i \parallel Tw_1)$$

Finally, the authentication request message $MSG1 = (U_1, U_2, U_3, PK_u, Tw_1)$ is sent to $GSS$ over a public channel to be analyzed later.

- Step 2: When the GSS receives the authentication request message $MSG1$ $(U_1, U_2, U_3, PK_u, Tw_1)$, it first checks the validation of the $Tw1$ by verifying $|Tw_c - Tw_1| \le \Delta T$, where $\Delta T$ is the maximum time threshold for receiving messages and $Tw_c$ is the currently received message time, and if the validation is successful, the $GSS$ will compute $E_{GSS} = PK_u.PR_{GSS}$. Using this value, the $GSS$ compute the following:
$MID_i^* = U_1 \oplus h(MID_{GSS} \parallel Tw_1)$
$MID_{Dr} = U_2 \oplus h(MID_i^* \parallel Tw_1 \parallel E_{GSS})$
$U_3^* = h(MID_i^* \parallel MID_{Dr}^* \parallel MID_{GSS}^* \parallel E_{GSS} \parallel Tw_1)$
$GSS$ checks if the condition $(U_3 \overset{?}{=} U_3^*)$ is valid. If it is invalid, $GSS$ rejects the authentication request. If not, $GSS$ can authenticate $U_i$ and continue the following steps.

$$G_1 = h(MID_{Dr}^* \parallel Tw_2) \oplus Rn_2$$

$$G_2 = MID_i^* \oplus h(MID_{Dr}^* \parallel MID_{GSS} \parallel Tw_2 \parallel Rn_2)$$

$$G_3 = h(MID_{Dr}^* \parallel MID_{GSS} \parallel MID_i^* \parallel Tw_2 \parallel Rn_2)$$

Finally, $GSS$ sends message $MSG_2 = (G_1, G_2, G_3, TW_2)$ to drone through a public channel.

- Step 3: Upon reception, the drone verifies the message's freshness by the condition $|Tw_c - Tw_2| \le \Delta T$, and on success, $Dr$ proceeds to calculate the following:

$$Rn_2^* = G_1 \oplus h(MID_{Dr} \parallel Tw_2)$$

$$MID_i^{'*} = G_2 \oplus h(MID_{Dr} \parallel MID_{GSS} \parallel Tw_2 \parallel Rn_2^*)$$

$$G_3^* = h(MID_{Dr} \parallel MID_{GSS} \parallel MID_i^* \parallel Tw_2 \parallel Rn_2^*)$$

It further checks whether $(G_3 \overset{?}{=} G_3^*)$ to authenticate GSS. If it fails, session is terminated immediately. Otherwise, it generates a random number $Rn_3$ using the current time window $Tw_3$, then proceeds to the next steps.

$$D_1 = h(MID_i^{'*} \parallel MID_{Dr} \parallel Tw_3) \oplus Rn_3$$

$$SK_{Dr,i} = h(MID_{Dr} \parallel MID_{GSS} \parallel MID_i^{*'} \parallel Tw_3 \parallel Rn_3)$$

$$Auth = h(SK_{Dr,i} \parallel Tw_3)$$

Finally, $Dr$ sends the message $MSG_3 = (D_1, Auth, Tw_3)$ directly to user $U_i$ through a public channel.

- Step 4: After getting the message $MSG_3 = (D_1, Auth, Tw_3)$, $U_i$ first checks time freshness by the condition $|Tw_c - Tw_3| \le \Delta T$. If the condition is valid, $U_i$ calculates $Rn_3^*$, session key and authentication value $(Sk_{i,Dr})$ as:

$$Rn_3^* = D_1 \oplus h(MID_i \parallel MID_{Dr} \parallel Tw_3)$$

$$SK_{i,Dr} = h(MID_{Dr} \parallel MID_{GSS} \parallel MID_i \parallel Tw_3 \parallel Rn_3^*)$$

$$Auth^* = h(SK_{i,Dr} \parallel Tw_3)$$

$U_i$ further verifies the validation of $Auth^* \overset{?}{=} Auth$. If they are equal, the user $U_i$ is regarded to have authenticated the drone $Dr$, and the computed session key is saved for future use in establishing secure communication. Otherwise, $U_i$ instantly terminates the session. Fig. 4 summarized the detailed illustration of the proposed scheme.

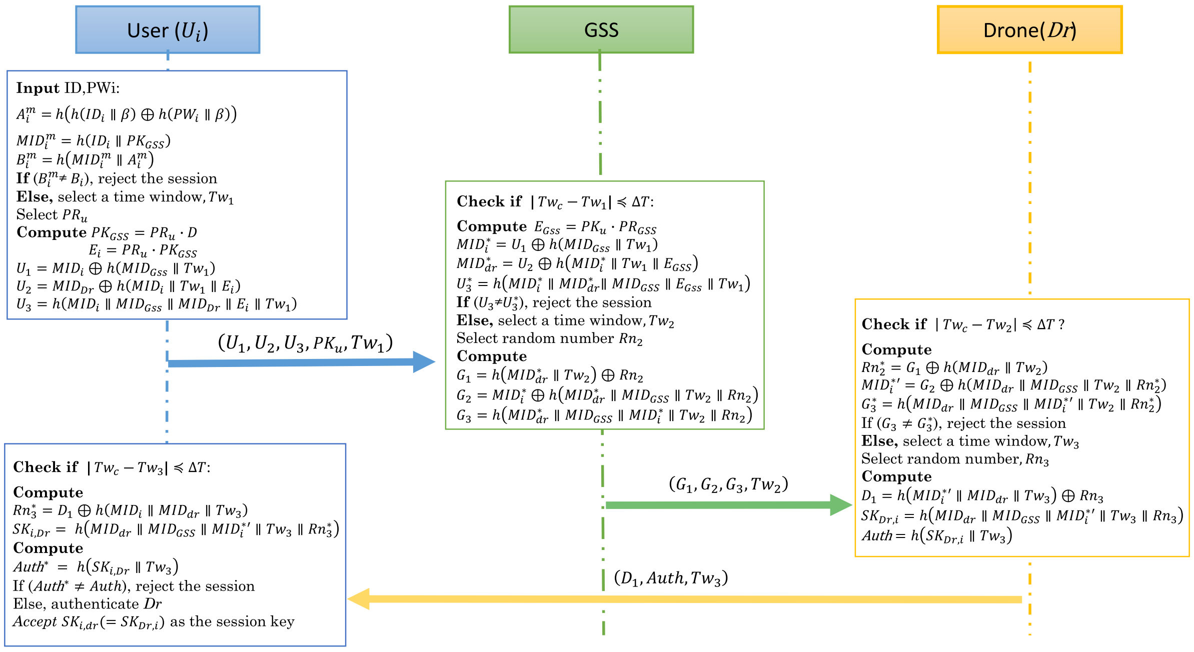

User ($U_i$)
GSS
Drone($Dr$)
**Input** ID,PWi:
$A_i^m = h(h(ID_i \| \beta) \oplus h(PW_i \| \beta))$
$MID_i^m = h(ID_i \| PK_{GSS})$
$B_i^m = h(MID_i^m \| A_i^m)$
**If** ($B_i^m \neq B_i$), reject the session
**Else,** select a time window, $Tw_1$
Select $PR_u$
**Compute** $PK_{GSS} = PR_u \cdot D$
$E_i = PR_u \cdot PK_{GSS}$
$U_1 = MID_i \oplus h(MID_{GSS} \| Tw_1)$
$U_2 = MID_{Dr} \oplus h(MID_i \| Tw_1 \| E_i)$
$U_3 = h(MID_i \| MID_{GSS} \| MID_{Dr} \| E_i \| Tw_1)$
$(U_1, U_2, U_3, PK_u, Tw_1)$
**Check if** $|Tw_c - Tw_1| \leqslant \Delta T$:
**Compute** $E_{GSS} = PK_u \cdot PR_{GSS}$
$MID_i^* = U_1 \oplus h(MID_{GSS} \| Tw_1)$
$MID_{dr}^* = U_2 \oplus h(MID_i^* \| Tw_1 \| E_{GSS})$
$U_3^* = h(MID_i^* \| MID_{dr}^* \| MID_{GSS} \| E_{GSS} \| Tw_1)$
**If** ($U_3 \neq U_3^*$), reject the session
**Else,** select a time window, $Tw_2$
Select random number $Rn_2$
**Compute**
$G_1 = h(MID_{dr}^* \| Tw_2) \oplus Rn_2$
$G_2 = MID_i^* \oplus h(MID_{dr}^* \| MID_{GSS} \| Tw_2 \| Rn_2)$
$G_3 = h(MID_{dr}^* \| MID_{GSS} \| MID_i^* \| Tw_2 \| Rn_2)$
$(G_1, G_2, G_3, Tw_2)$
**Check if** $|Tw_c - Tw_2| \leqslant \Delta T$ ?
**Compute**
$Rn_2^* = G_1 \oplus h(MID_{dr} \| Tw_2)$
$MID_i^{*\prime} = G_2 \oplus h(MID_{dr} \| MID_{GSS} \| Tw_2 \| Rn_2^*)$
$G_3^* = h(MID_{dr} \| MID_{GSS} \| MID_i^{*\prime} \| Tw_2 \| Rn_2^*)$
If ($G_3 \neq G_3^*$), reject the session
**Else,** select a time window, $Tw_3$
Select random number, $Rn_3$
**Compute**
$D_1 = h(MID_i^{*\prime} \| MID_{dr} \| Tw_3) \oplus Rn_3$
$SK_{Dr,i} = h(MID_{dr} \| MID_{GSS} \| MID_i^{*\prime} \| Tw_3 \| Rn_3)$
$Auth = h(SK_{Dr,i} \| Tw_3)$
$(D_1, Auth, Tw_3)$
**Check if** $|Tw_c - Tw_3| \leqslant \Delta T$:
**Compute**
$Rn_3^* = D_1 \oplus h(MID_i \| MID_{dr} \| Tw_3)$
$SK_{i,Dr} = h(MID_{dr} \| MID_{GSS} \| MID_i^{*\prime} \| Tw_3 \| Rn_3^*)$
**Compute**
$Auth^* = h(SK_{i,Dr} \| Tw_3)$
If ($Auth^* \neq Auth$), reject the session
Else, authenticate $Dr$
Accept $SK_{i,dr} (= SK_{Dr,i})$ as the session key

**Fig. 4.** Authentication process of the HCALA scheme.

## 5.4. *Password update phase*

A password updating process should be available in a secure authentication scheme. So authorized user $U_i$ using a mobile device can change the current password $PW_i$ with a new password $PW_i^{new}$. $U_i$ must complete the following steps:

- Step1: $U_i$ inputs his login credentials identity $ID_i$ and password $Pw_i$. Then, the mobile device calculates the following operations:

$$A_i^m = h(h(ID_i \parallel \beta) \oplus h(PW_i \parallel \beta))$$

$$MID_i^m = h(ID_i, PK_{GSS})$$

$$B_i^m = h(MID_i^m \parallel A_i^m)$$

  The mobile device then checks if the condition ($B_i^m \overset{?}{=} B_i$) is valid, and if it is not, the procedure is aborted. Otherwise, it will ask $U_i$ to provide a new password to complete the process.
- Step2: $U_i$ selects a new password $PW_i^{new}$ and sends it. The mobile device calculates the following:

$$A_i^{new} = h(h(ID_i \parallel \beta) \oplus h(PW_i^{new} \parallel \beta))$$

$$MID_i = h(ID_i^{new}, PK_{GSS})$$

$$B_i^{new} = h(MID_i \parallel A_i^{new})$$

- Step3: Finally, $U_i$ replaces $B_i^{new}$ with $B_i$ in the mobile device.

To maximize the security system, it is important to remember that the password for user $U_i$ may be updated periodically.

## 5.5. *Revocation and reissue phase*

In the case that an authorized user $U_i$'s mobile device is stolen or lost, $U_i$ can replace it and follow the instructions listed below.

- Step 1: $U_i$ keeps his $ID_i$ identity but chooses $PW_i^{new}$ as his new password. Then, using a random number $\beta'$, $U_i$ calculates

$$A_i^{new} = h(h(ID_i \parallel \beta') \oplus h(PW_i^{new} \parallel \beta'))$$

and sends $\{ID_i, A_i^{new}\}$ to the $GSS$ across secure channel.
- Step 2: $GSS$ computes $MID_i$ and $B_i$ after receiving the message, as follows:

$$MID_i = h(ID_i \parallel PR_{GSS})$$

$$B_i^{new} = h(MID_i \parallel A_i^{new})$$

Next, GSS stores $\{MID_i, B_i^{new}\}$ in its database, and sends $\{MID_i, B_i^{new}\}$ to $U_i$ through a safe channel.
- Step 3: Following receipt from $GSS$, $U_i$ computes

$$B_i^{new^*} = h(MID_i \parallel A_i^{new}) \oplus B_i^{new}$$

$$MID_i^* = h(ID_i \parallel PW_i^{new}) \oplus MID_i$$

Finally, $U_i$ replaces $B_i$ with $B_i^{new^*}$, and stores $\{B_i^{new^*}, MID_i^*\}$ in its own memory of device. $U_i$ also deletes $B_i'$ from the memory of the device to complete the revocation and reissue process.

### 5.6. Dynamic drone addition phase

It is critical to immediately deploy another drone in the same flight area. in the event of an accident, such as a low battery or physical capture of the drone by an attacker. To accomplish this, the HCALA scheme allows for the addition of new drones to the network at any moment. This phase is very similar to the phase of drone registration. Below is a description of this phase in more detail:

- Step1: For a new unregistered drone $Dr$ to be used in a certain flying area, the $GSS$ picks a unique identity $ID_{Dr}^{new}$ and computes the corresponding masked-identity as:

$$MID_{Dr}^{new} = h(ID_{Dr}^{new} \parallel PR_{GSS})$$

- Step2: The $GSS$ stores $\{ID_{Dr}^{new}, MID_{Dr_j}^{new}\}$ in the drone's memory before deploying it in the field, $GSS$ also keep $\{ID_{Dr}^{new}\}$ in its own database.

### 5.7. Block creation and addition in blockchain

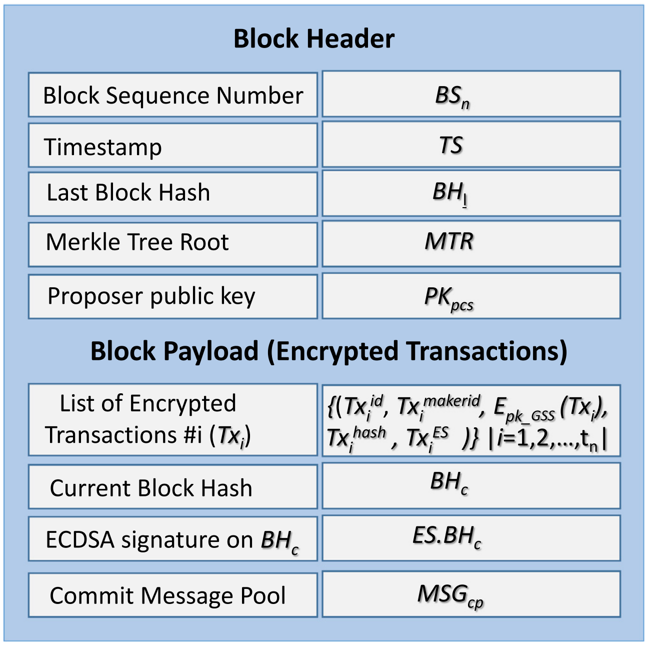
We presume that the data gathered by drones are private and confidential under the HCALA scheme. As a result, we want to store this information in a private blockchain managed by the P2P CS network. The computing power of drones is constrained. So, assigning these entities the responsibility of creating transactions for the blockchain might prove to be very taxing. For this reason, allowing the $GSS$ to construct the transactions of the acquired data to be added to the blockchain is considered to be more computationally efficient. Once a block $Block_m$ is received by a cloud server $CS$ from the $GSS$, When the threshold is reached for the pool's number of transactions ($Tran_{sh}$), the received cloud server $CS$ generates a transaction pool containing securely received transactions, $CS$ will create a block ($Block_m$), the formation of ($Block_m$) consists the transactions $Tx_1$, $Tx_2$, $Tx_3$... as shown in Fig. 5. Utilizing a voting-based consensus mechanism, such as the "PBFT" algorithm, $CS$ adds the transactions to the blockchain. Algorithm 3 gives a detailed description of this process.

## 6. Security analysis

This part analyzes the security features of the HCALA scheme. First, we show that HCALA is secure using the "ROM" and the AVISPA tool [15]. The security features are then evaluated to ensure that HCALA is highly resistant to a variety of potential attacks. Table 4 compares the security and functionality properties between HCALA and other schemes.

### 6.1. Formal security verification using (ROM)

The ROM model involves an attacker, denoted as $A$, who engages with the $i$th instance of a participant executing the protocol, denoted as $\Pi^i$. In the context of our scheme, this participant could be a legitimate user represented by $U_i$, a drone represented by $DR_j$ or the ($GSS$). In addition, the ROM model assumes that several queries, such as $Extract(.)$, $Execute(.)$, $Test(.)$, and $Reveal(.)$, are used to simulate an actual attack, as mentioned in Table 3. Additionally, the instances of each entity, as well as $A$, can access a collision-resistant one-way hash function $h(.)$.



**Block Header**

| | |
|---|---|
| Block Sequence Number | $BS_n$ |
| Timestamp | $TS$ |
| Last Block Hash | $BH_l$ |
| Merkle Tree Root | $MTR$ |
| Proposer public key | $PK_{pcs}$ |

**Block Payload (Encrypted Transactions)**

| | |
|---|---|
| List of Encrypted Transactions #i ($Tx_i$) | $\{(Tx_i^{id}, Tx_i^{makerid}, E_{pk\_GSS}(Tx_i), Tx_i^{hash}, Tx_i^{ES})\} \mid i=1,2,\ldots,t_n \mid$ |
| Current Block Hash | $BH_c$ |
| ECDSA signature on $BH_c$ | $ES.BH_c$ |
| Commit Message Pool | $MSG_{cp}$ |

**Fig. 5.** Structure of a block $Block_m$.

**Algorithm 3:** Consensus for block validation and addition

```
Input: $Tans_p$: A pool of transactions, $N$ : number of P2P nodes, $Tans_sh$: transaction threshold, $App_t$: approval
       threshold, where $App_t = 2 * (N - 1)/3 + 1$
Output: After successful validation, the block $Block_m$ is committed and added to the blockchain.
1  if ($|Tans_p| = Tans_{sh}$) then
2      A leader $CS_l$ is chosen in a round-robin manner from the P2P $CS$ network for voting requests.
3      ($CS_l$) constructs a block $Block_m$ depicted in Fig 5, sets $MSG_{cp} \leftarrow \emptyset$ (empty) and broadcasts $Block_m$ to the P2P
        network for voting request
4      The follower receives $Block_m$ and validates it with the transaction pool
5      for "each follower" $CS_j$ do
6          Verify $Tx_i^{hash}$, $Tx_i^{ES}$, $MTR$, $BH_c$, $ES.BH_c$
7          If all are validated successfully, $CS_j$ puts a valid vote reply to $MSG_{cp}$
8      end
9      Let $VT_{count}$ denotes the number of valid votes in the pool, $MSG_{cp}$
10     Set $VT_{count} \leftarrow 0$
11     for "each valid vote reply in $MSG_{cp}$" do
12         Set $VT_{count} = VT_{count}+1$
13     end
14     if ($App_t \leq VT_{count}$ ) then
15         Add block $Block_m$ to the blockchain
16         Broadcast commitment response to all followers
17     end
18 end
```

**Definition** (*Semantic Security*)**.** The indistinguishability of real $SK$ from a random number guessed by $A$ is the basis for the security of $SK$, which is established between $U_i$ and $DR_j$ under the ROM. $A$ has the probability of breaking the security of the HCALA scheme to derive the $SK$. Let $\Omega$ represent the correct bits and $\Omega'$ represent $A$'s guess. $A$ will win the game if $\Omega = \Omega'$. The advantage of $A$ is described by:

$$Adv_A^{protocol} = |2.Prob[\Omega = \Omega'] - 1|.$$

where $Pro[\Omega = \Omega']$ represents the probability of success. If $Adv_A^{protocol}$ is negligible under the ROM, our Scheme is secure.

**Table 3**
Queries and their purposes.

| Query | Purpose |
|---|---|
| $Corrupt(\Pi^t, U_i)$ | Applying this query, the adversary $A$ can utilize a power analysis attack to corrupt a legitimate user and obtain the sensitive credentials. |
| $Send(\Pi^t, msg)$ | This means that $A$ can initiate an active attack by sending the message $msg$ to $\Pi^t$ and receiving message in return from $\Pi^t$. $A$ can start a new instance of $\Pi^t$ by sending Send($\Pi^t$, start) to the oracle. |
| $Reveal(\Pi^t)$ | Applying this query, $A$ can reveal a session key $Sk$ between $\Pi^t$ and its partner. |
| $Execute(U_i, V_j)$ | This query enables $A$ to launch passive attacks. This query has the potential to eavesdrop on any messages sent over the public channel. It outputs the exchanged messages among participants. |
| $Test(\Pi^t)$ | By executing this query, $A$ may confirm if the established $SK$ is actual or a probabilistic random result of a coin flip. This query can only be executed once by $A$. if $b = 1$, $C$ returns a valid $SK$ to $A$ ; else, ($b = 0$) returns a random, equal-sized secret key. |

**Table 4**
Comparison of functionality features between HCALA scheme and related authentication schemes.

| | | [32] | [10] | [20] | [22] | [31] | HCALA |
|---|---|---|---|---|---|---|---|
| Security goals | ■ Privacy and anonymity | ● | ● | ● | ● | ● | ● |
| | ■ Un-traceability | ○ | ○ | ● | ● | ● | ● |
| | ■ Mutual Authentication | ● | ● | ● | ● | ● | ● |
| | ■ Session Key Agreement | ● | ● | ● | ● | ● | ● |
| | ■ Integrity | ○ | ○ | ○ | ○ | ○ | ● |
| Resistance to | ■ Replay Attacks | ● | ● | ● | ● | ● | ● |
| | ■ Denial-of-Service Attack | ○ | ○ | ● | ● | ● | ● |
| | ■ MTM Attack | ● | ● | ● | ● | ● | ● |
| | ■ Modification Attack | ○ | ○ | ○ | ○ | ○ | ● |
| | ■ Physical Drone Capture Attack | ○ | ● | ● | ● | ● | ● |
| | ■ Known session key Attack | ○ | ○ | ● | ○ | ● | ● |
| | ■ Stolen Smart Device Attack | ● | ● | ● | ● | ● | ● |
| | ■ GSS impersonation Attack | ○ | ● | ○ | ○ | ○ | ● |
| | ■ User Impersonation Attack | ○ | ● | ● | ● | ● | ● |
| | ■ Drone Impersonation Attack | ● | ○ | ○ | ● | ○ | ● |
| Network components | ■ Cloud computing | ○ | ○ | ○ | ○ | ○ | ● |
| | ■ Blockchain | ○ | ○ | ○ | ○ | ○ | ● |
| Security analysis | ■ Formal: ROM | ● | ○ | ○ | ○ | ○ | ● |
| | ■ Formal: AVISPA tool | ○ | ○ | ● | ● | ○ | ● |
| | ■ Informal analysis | ● | ● | ● | ● | ● | ● |

*Notes:* ●: indicates that the feature is available; ○: indicates that the feature is not available.

**Theorem 1.** *Suppose that attacker $A$ tries to compromise a secret key's security in polynomial time $T$. If $Q_{Hash}$, $|Hash|$, and $Adv_A^{HECDLP}(P_t)$, denotes the number of hash queries, the size of the one-way collision-resistant hash function $h(.)$, and the advantage of breaking the (HECDLP) for A, respectively. The approximate advantage of A in breaching the security of HCALA in order to obtain SK between $U_i$ and $D_r$ is as follows:*

$$Adv_A^{HCALA}(P_t) \leq \frac{Q_{Hash}^2}{2|Hash|} + Adv_A^{HECDLP}(P_t).$$

**Proof.** The security of $SK$ is demonstrated proved in the following three games, namely $Game_i^A (i = 1, 2, 3)$, using the queries provided in Table 3. In the game $Game_i^A$, let $Success_{game_i}^A$ be an event in which $A$ successfully guesses a random bit $\Omega$. Thus, the advantage (success probability) of $A$ to win the game $Game_i^A$ is $Adv_{A,Game_i}^{HCALA} = Prob[success_{Game_i}^A]$. The following is a detailed description of each game.

$Game_1^A$ : $A$ plays a real attack in this game against the suggested scheme under ROM. At the beginning of the game $Game_1^A$, $A$ must guess the bit $\Omega$. Then, we have:

$$Adv_A^{HCALA}(P_t) = |2.Adv_{A,Game_1}^{HCALA} - 1|. \tag{3}$$

$Game_2^A$ : An eavesdropping attack is simulated in this game, in which $A$ can intercept all communicated messages $MSG1 = \{U_1, U_2, U_3, PK_u, TW_1\}$, $MSG2 = \{G_1, G_2, G_3, TW_2\}$, and $MSG3 = \{D1, Auth, TW_3\}$ during the login and authentication phase by utilizing the *Execute* query provided in Table 3 to execute the designed protocol. $A$ executes the *Reveal* and *Test* queries at the end of the game to determine whether the generated $SK$ is valid or a random key. The session key created between user $U_i$ and accessible drone $DR_j$ is $SK_{i,Dr} = h(MID_{Dr} \parallel MID_{GSS} \parallel MID_i\ parallel Tw_3 \parallel Rn_3^*)$. Because all the temporal secrets($Rn_3^*$) and long term secrets ($MID_{Dr}, MID_{GSS}, MID_i$) are safeguarded by $h(.)$ and unknown by the attacker $A$. As a result, the chance of successfully obtaining the session key $SK_{i,Dr}(= SK_{Dr,i})$ will not be increased by eavesdropping on the messages $MSG1$, $MSG2$ and $MSG3$. Hence, under the eavesdropping attack, the $Game_2^A$ and $Game_1^A$ are no longer distinguishable. This leads to the following:

$$Adv_{Game_2,A}^{HCALA} = Adv_{Game_1,A}^{HCALA}. \tag{4}$$

$Game_3^A$ : In this game, the attacker $A$ performs $Corrupt(\Pi^t, U_i)$ query to extricate the data kept in the $U_i$'s memory (i.e., $\{B_i', MID_i'\}$) by utilizing the power analysis attack [45]. The hash function protects variables like $ID_i$, $PW_i$ and $A_i$. It is worth noting that even if the attacker successfully captures $ID_i$, $PW_i$ and $A_i$, generating the authentication message $MSG1 = (U_1, U_2, U_3, PK_u, Tw_1)$ would be impossible. Because of the following two reasons, performing the aforementioned computations is difficult:

- Complicated HECDLP computations are required to determine the values of the related variables, such as $PK_u$ and $E_i$.
- An effort to find the values of $MID_{GSS}$ and $MID_{Dr}$ from $U_1$, $U_2$ and $U_3$ is thwarted by one-way collision resistant hash function property.

No collision happens even if the attacker $A$ performs the hash query. Additionally, $Game_2^A$ and $Game_3^A$ are difficult to differentiate. Consequently, due to HECDLP and the birthday paradox concept, the following result is obtained:

$$|Adv_{A,Game_2}^{HCALA} - Adv_{A,Game_3}^{HCALA}| \le \frac{Q_{Hash}^2}{2|Hash|} + Adv_A^{HECDLP}(P_t). \tag{5}$$

It is sufficient to predict the bit $c$ to gain the *Game* once the *Test* query has been performed, and therefore we obtain the following output:

$$Adv_{A,Game_3}^{HCALA} = \frac{1}{2}. \tag{6}$$

Eq. (3) gives:

$$\frac{1}{2}Adv_A^{HCALA}(p_t) = |Adv_{A,Game_1}^{HCALA} - \frac{1}{2}|. \tag{7}$$

Simplifying the Eqs. (3)–(5), and using triangular inequality result in the following derivation from Eq. (7)

$$\frac{1}{2}Adv_A^{HCALA}(p_t) = |Adv_{A,Game_1}^{HCALA} - Adv_{A,Game_3}^{HCALA}| \tag{8}$$

$$= |Adv_{A,Game_2}^{HCALA} - Adv_{A,Game_3}^{HCALA}| \tag{9}$$

$$\le \frac{Q_{Hash}^2}{2|Hash|} + Adv_A^{HECDLP}(p_t).$$

The final result is obtained by multiplying both sides of the equation Eq. (9) by "2" as follows:

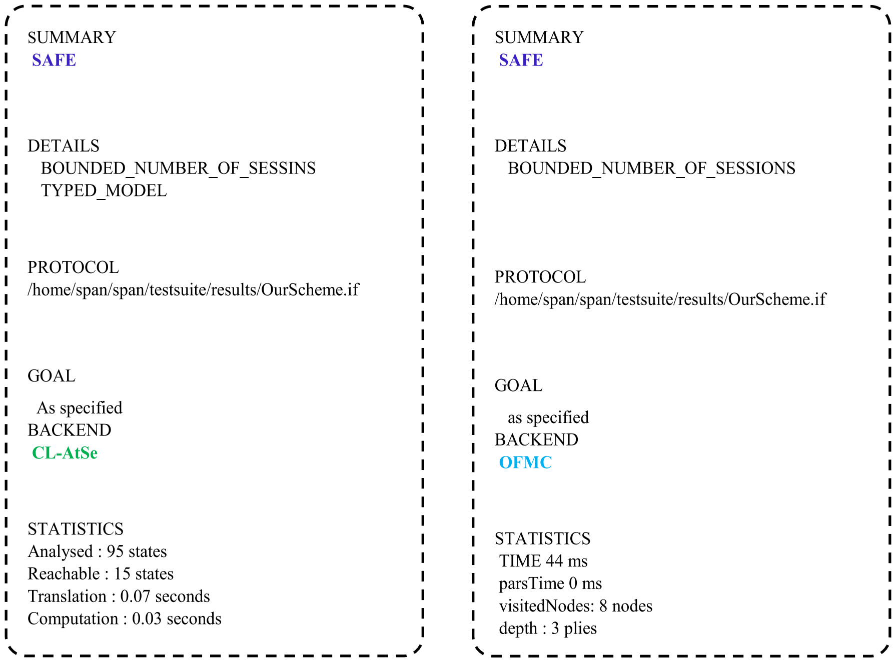
$$Adv_A^{HCALA}(P_t) \le \frac{Q_{Hash}^2}{|Hash|} + 2Adv_A^{HECDLP}(P_t) \quad \blacksquare$$

### 6.2. Formal security verification

To validate the security of the designed protocol (HCALA), we used "AVISPA" tool to provide the formal security verification [15]. The use of automated software in formal security verification has grown in popularity among security researchers in recent years. AVISPA [15], ProVerif [47], Casper/FDR [48], and Scyther [49] are some of the formal security verification techniques available. AVISPA is equipped with several cutting-edge methods for performing an automatic security analysis of a security scheme. The four back-ends that AVISPA integrates with are CL-AtSe, SATMC, OFMC, and TA4SP [15]. The modular language must be used to implement the tested protocols "High-Level Protocol Specification Language (HLPSL)", which facilitates the modeling of complicated security properties. The output format (OF) is created by converting the HLPSL code to the intermediate Format (IF) and then feeding it into one of the four available back-ends.

HCALA scheme has been implemented for three basic roles: User ($U_i$), $GSS$, and $Dr_j$. Additionally, HCALA defines the session's mandatory roles, as well as the composite roles (session, goal, and environment). The designed scheme utilizes OFMC and CL-AtSe backends. Because they do not allow bitwise XOR operations, other back-ends (TA4SP and SATMC) were not considered. To determine whether a protocol is vulnerable to a replay attack, the back-ends determine whether the tested scheme may be implemented by approved agents to identify an intruder (passive adversary). The back-ends offer information to the intruder regarding normal sessions held by authorized agents.

HCALA scheme was then simulated utilizing "Security Protocol ANimator for AVISPA (SPAN)" tool [15]. Fig. 6 depicts the simulation results in detail.



```
SUMMARY
 SAFE

DETAILS
  BOUNDED_NUMBER_OF_SESSINS
  TYPED_MODEL

PROTOCOL
/home/span/span/testsuite/results/OurScheme.if

GOAL
 As specified
BACKEND
 CL-AtSe

STATISTICS
Analysed : 95 states
Reachable : 15 states
Translation : 0.07 seconds
Computation : 0.03 seconds
```

```
SUMMARY
 SAFE

DETAILS
  BOUNDED_NUMBER_OF_SESSIONS

PROTOCOL
/home/span/span/testsuite/results/OurScheme.if

GOAL
  as specified
BACKEND
 OFMC

STATISTICS
 TIME 44 ms
 parsTime 0 ms
 visitedNodes: 8 nodes
 depth : 3 plies
```

**Fig. 6.** Results of the AVISPA simulation using the CL-AtSe and OFMC backends.

## 6.3. *Informal security analysis*

### 6.3.1. *Privacy and anonymity*

In order to protect privacy and anonymity, the proposed scheme must assure that, once our system is in place, no attacker can extract real identities. As stated earlier, our system can protect the privacy of all transmitted and received messages, such as $MSG_1$, $MSG_2$, $MSG_3$. Furthermore, fresh time windows and random numbers are used to produce these messages. Due to this reason, it is difficult for attackers to get real identity and private data from users, GSS, or drones, which is a substantial advantage. Consequently, the HCALA scheme ensures privacy and anonymity.

### 6.3.2. *Un-traceability*

During the authentication step, random nonce $Rn_2$, $Rn_3$, and current time-window are selected differently for each session, resulting in messages sent by each participant is unique. The adversary $A$ is unable to correlate the messages transmitted by the GSS, drone, and user. Likewise, the sender cannot be tracked. Furthermore, a safe one-way collision-resistant hash function contains the real identities or masked identities ($ID_x$, $MID_x$). Thus, the HCALA scheme is capable of achieving un-traceability.

### 6.3.3. *Session key agreement*

After confirming mutual authentication between the registered user $U_i$ and drone $Dr_j$ during the login and authentication phase, the user and drone calculate a common session key $SK_{Dr,i} = h(MID_{Dr} \parallel MID_{GSS} \parallel MID_i^{*'} \parallel Tw_3 \parallel Rn_3)\ (= SK_{i,Dr})$ and utilize it in future communication. Consequently, the HCALA scheme ensures the session key agreement.

### 6.3.4. *Perfect forward secrecy (PFS)*

PFS property guarantees that the leakage of a long-term key does not result in the leakage of the session key used in the previous communication. Participants are required to produce a new session key ($Sk$) for each session under the HCALA scheme, which includes a random number ($Rn$) that cannot be easily computed or guessed by an attacker $A$. In addition, the security of previous sessions is not compromised even if an attacker obtains a secret component key because the protocol includes a time window $TW$ that verifies recent sessions. This means that our protocol ensures perfect forward secrecy.

### 6.3.5. *Integrity*

Integrity is the guarantee that no opponent can modify the exchanged messages, and if they do, the system will identify and report the change. Firstly, based on the *HECDLP*, it is tough for adversaries to deduce the corresponding $Sk_x$. Second, after exchanging a message in each phase, nodes perform an integrity check using the one-way hash function. Consequently, the HCALA scheme is far more secure in terms of maintaining integrity.

## 6.4. Resistance against potential attacks

### 6.4.1. Resistance against replay attacks

Based on the time windows and random numbers (or nonces) involved in the communicated messages, When a message is received, the time window associated with the message must first be extracted and compared to the one that is encrypted within the message as the first step. If the values do not match, the receiver will determine that the message is old or has been altered. As a result, our protocol can Withstand replay attacks.

### 6.4.2. Resistance against denial-of-service attack

If a registered user $U_i$ enters an invalid $ID_i$ and/or $PW_i$ during the login phase or the password update phase, it is locally checked by verifying the condition $A1 = A$ or $A_{old} = A_{new}$. After being verified successfully, the user $Ui's$ login request is forwarded to the $GSS$. Furthermore, the password is only updated when the old password has been successfully verified in the update process. Consequently, the HCALA scheme is resistant to such kinds of denial-of-service attacks.

### 6.4.3. Resistance against MTM attacks

utilizing hash function $h(.)$ and time windows in with authentication tokens renders the MTM attack ineffective. He tries to capture and modify the messages $MSG_1, MSG_2, MSG_3$ so that the participants believe they are receiving messages from actual legitimate participants. This attempt fails because he was unable to generate or confirm authentication tokens, and he was unable to modify or delay the transmitted messages due to the fact that the hash function is employed to control both the integrity and the freshness of messages. The proposed protocol can therefore resist MTM attack.

### 6.4.4. Resistance to drone impersonation attack

If an attacker wants to impersonate a registered drone $Dr_j$, he must produce valid messages $Auth = h(sk_{Dr_j} \parallel TW_3)$ and send it to $U_i$ in a manner that passes the comparison. Nevertheless, $Auth_j$ contains the session key $SK_{Dr_j}$, which cannot be accessed by the attacker. When $U_i$ receives the message $Auth$, it calculates $Auth^*$ and compares it to $Auth$ to see whether they are equivalent. As a result, $U_i$ can differentiate between an impersonated drone and a legitimate drone, indicating that the HCALA scheme is secure from drone impersonation attacks.

### 6.4.5. Resistance to user impersonation attack

According to what was said in the second step of the login and authentication phase, the GSS authenticates user $U_i$ by computing $U_3*$ and comparing it with $U_3$ received from $U_i$. To impersonate $U_i$, one way for the attacker is to generate valid messages $U_3 = h(MID_i \parallel MID_{GSS} \parallel MID_{Dr} \parallel E_i \parallel Tw_1)$ and send them to $GSS$. The attacker can generate its own timestamp $TW_A$ but he is unable to access the secret parameters, which include $E_i$ and private key $PR_{GSS}$ of GSS, which are also unavailable to the attacker. Thus, the attacker is unable to produce a valid $U_3$, as a result, GSS can distinguish the impersonated user from the real user.

### 6.4.6. Resistance to GSS impersonation attack

The adversary assumes the role of a legal register $GSS$ in this attack, he intercepts the authentication message $MSG_2$ between the $GSS$ and the drone $Dr_j$. To prove his authenticity, the adversary may create modified or fake messages by extracting sensitive data from the $GSS$. To accomplish so, the attacker must successfully generate a valid message $MSG_2$ in polynomial time by creating timestamp $TW_2$ and the fresh random number $Rn_2$. The attacker is unable to compute $G_1, G_2, G_3$ or modify $MSG_3$ due to a lack of information about $MID_{dr}$, $MID_{GSS}$ and $MID_i$. As a result, it assures that the attacker cannot forge or tamper the $GSS's$ deceived message in polynomial time. HCALA scheme is protected from GSS impersonation attacks.

### 6.4.7. Resistance to stolen smart device attack

Suppose that the smart device of a registered user $U_i$ is stolen or lost by an attacker. All information $\{B'_i, MID'_i\}$ in the device's memory can be extracted using power analysis attacks. in which $B'_i = h(MID_i \parallel A_i) \oplus B_i$ $MID'_i = h(ID_i \parallel PW_i) \oplus MID_i$. From the information gathered, the attacker cannot correctly guess $ID_i$ and $PW_i$ from the extracted information without the knowledge about secret parameter $A_i$. In addition, a one-way hash function prevents an attacker from retrieving both the password and identity concurrently. However, He is unable to obtain $U_i$'s secret parameters. Consequently, the HCALA scheme is safe against mobile device stolen attacks.

### 6.4.8. Resistance to known session key attack

The session key $SK_{Dr,i} = h(MID_{Dr} \parallel MID_{GSS} \parallel MID'_{i*} \parallel Tw_3 \parallel Rn_3)$ contains the random numbers unique to the current session. The one-way collision-resistant hash function prevents the attacker from parsing the random numbers from $Sk$. As a result, an attacker who acquires an old session key will be unable to access the present session key. Thus, the HCALA scheme is resistant to the known session key attack.

**Table 5**
Comparison of computation cost.

| Scheme | User side | Server side | Drone side/Sensing device side | Total (ms) |
|---|---|---|---|---|
| [32] | $6T_h + 3T_{ag} + 3T_{ecm} + T_{fe}(10.1628)$ | $2T_h + 3T_{ecm} + T_{ag}(3.4756)$ | $3T_h + 2T_{ecm} + 2T_{ag}(5.2889)$ | 18.9273 |
| [10] | $5T_h + 2T_b(10.8655)$ | $3T_h + 2T_b(10.8609)$ | $9T_h + 2T_b + 4T_{ecm}(19.7787)$ | 41.5051 |
| [20] | $5T_h + 5T_{ecm} + T_{fe}(13.3675)$ | $4T_h + 5T_{ecm}(11.1392)$ | $3T_h + 4T_{ecm}(26.712)$ | 51.2187 |
| [22] | $T_{fe} + 16T_h(2.2605)$ | $8T_h(0.0184)$ | $7T_h(0.0161)$ | 2.2973 |
| [31] | $15T_h + T_{fe}(2.2605)$ | $9T_h + 2T_{sym}(0.0299)$ | $7T_h(0.0161)$ | 2.3065 |
| HCALA | $9T_h + 2T_{Hec}(2.2467)$ | $6T_h + T_{Hec}(1.1268)$ | $6T_h(0.0138)$ | 3.3873 |

### 6.4.9. *Resistance to physical drone capture attack*

As previously stated, a physical drone capture by an attacker is a real possibility. Assume an attacker has captured drone $Dr$ and can obtain all stored credentials and communication information $\{ID_{Dr_j}, MID_{Dr_j}\}$. The private key $PR_{GSS}$ is contained in a one-way hash function, preventing an adversary from computing the next communication session key without knowledge of the masked identity and random numbers. Using the extracted information from a captured drone, an attacker cannot generate session keys for non-compromised drones and the *GSS* since the secret information is distinct and unique for each deployed drone. Consequently, the HCALA scheme can resist physical drone capture attacks.

### 6.4.10. *Modification attack*

An adversary can modify the authentication and reply packets. We utilize a one-way hash function to ensure that information is not modified. Moreover, It can be seen that the sent message $U_3(G_3)$ contains the receiver's(sender's) secret key $E_i$. The $GSS(Dr_j)$ can easily detect if the message has been updated by examining the equation $U_3 = U_3^*(G_3 = G_3^*)$. In addition, $U_i$ may identify any change to *Auth* by evaluating the equation $Auth = Auth^*$. Consequently, the HCALA scheme is resistant to modification attacks.

## 7. Performance evaluation

In this section, we assess the performance of our suggested scheme in terms of its computation and communication overheads and energy consumption. These metrics are crucial indicators of the efficiency and practicality of the HCALA scheme in real-world situations. To determine the effectiveness of the HCALA scheme, we compare it withTanveer et al.'s [32], Ever et al. [10], Challa et al.'s [20], Wazid et al.'s [22], and Hussain et al.'s [31] scheme. This comparison enables us to evaluate our scheme's strengths and weaknesses and demonstrate its advantages over existing schemes.

### 7.1. *Computational overhead*

The registration phase includes necessary operations such as XOR operations, hash functions, comparisons, ECC and HEC multiplicative operations, and concatenation operations. In comparison with other operations and functions, concatenation, XOR operation, and comparison are negligible. Let $T_h$, $T_{ecm}$, $T_{hcm}$, $T_{fe}$, $T_{sym}$, and $T_{ag}$ denote the time required to execute a secure hash function, HEC divisor multiplication, ECC point multiplication, fuzzy extractor function (Gen(·)/Rep(·)), symmetric encryption/decryption, and AEGIS (AEAD scheme), respectively. Using the results utilized in [26,50–52], we have $T_h \approx 0.0023$ ms, $T_{ecm} \approx 2.226$ ms, $T_{hcm} \approx 0.48$ ms, $T_{fe} \approx T_{ecm} \approx 2.226$ ms, $T_{sym} \approx 0.0046$ ms, and $T_{ag} \approx 0.415$ ms.

The comparative results of computing overheads among various related authentication schemes [10,20,22,31,32] reported in Table 5 as well as in Fig. 7. Table 5 and Fig. 7 clearly show that the HCALA scheme achieves significantly better performance than other related schemes [10,20,32], but incurs a higher computation cost than comparable schemes [22,31], However, the HCALA scheme provides enhanced security and functionality.

### 7.2. *Communication overheads*

Fig. 8 provides a comparison of the communication costs of some related authentication schemes [10,20,22,31,32] and HCALA scheme during the login and authentication phases. To estimate communication costs, we assume the bit-sizes of the identity, random number, timestamp, elliptic curve point, hyperelliptic curve and the digest of a hash function (using Secure Hash Standard "SHA-1") are 160, 32, 128,160, 80 and 160 bits, respectively. As shown in Fig. 8, our protocol requires less communication cost than the related protocols [10,20,22,31,32] in terms of bits needed to transmit the messages.

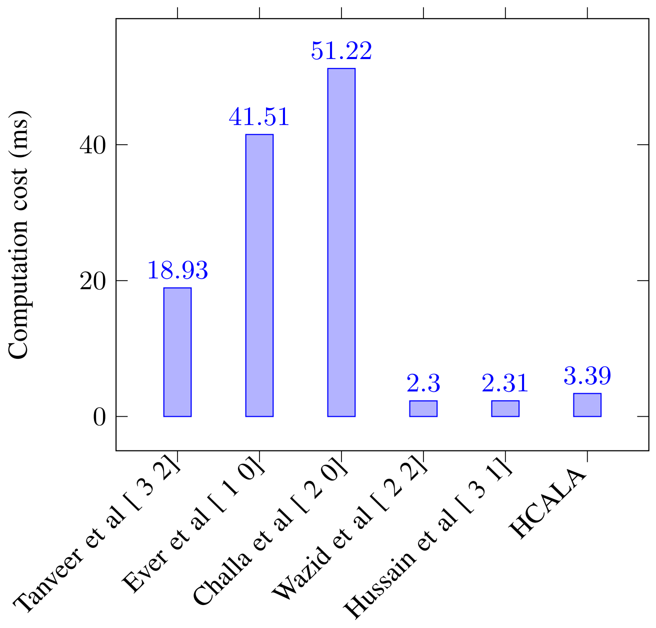


**Fig. 7.** Comparison of computation costs.

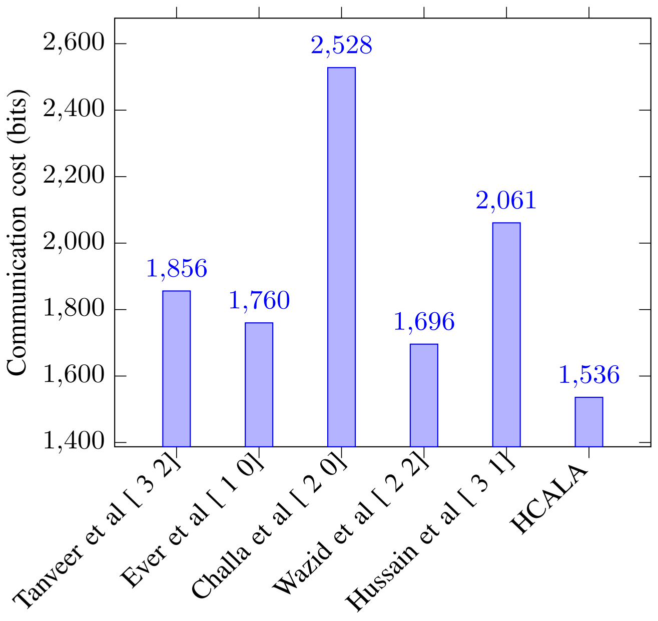


**Fig. 8.** Comparison of communication costs.

### *7.3. Energy consumption*

Energy consumption refers to the total amount of energy used to complete all algorithm operations [53]. The energy consumed in the communication process is measured in Joules and is determined by the number of messages exchanged [54].

Table 6 compares the energy consumption of several relevant schemes, including [10,20,22,31,32], and the HCALA scheme during the authentication process. The HCALA scheme and [20,22,31] demonstrate a similar energy consumption level of $(3.38 \times 10^{-4})$ and consume less energy than the other schemes $(6.76 \times 10^{-4})$ [10,32], indicating that HCALA is more energy-efficient.

**Table 6**
Comparison of communication energy consumption.

| Scheme | No. of messages | Energy consumption (Joule) |
|---|---|---|
| [32] | 6 | $6.76 \times 10^{-4}$ |
| [10] | 6 | $6.76 \times 10^{-4}$ |
| [20] | 3 | $3.38 \times 10^{-4}$ |
| [22] | 3 | $3.38 \times 10^{-4}$ |
| [31] | 3 | $3.38 \times 10^{-4}$ |
| HCALA | 3 | $3.38 \times 10^{-4}$ |

## 8. Conclusion and future work

We designed, in this article a hyperelliptic curve-based anonymous lightweight authentication (HCALA) scheme between users and drones with the aid of the GSS. Specifically, the HCALA scheme comprises the following phases: setup, registration, login and authentication, password update, revocation and reissue, and dynamic drone addition. Based on HECC, hash functions, XOR operation and blockchain technology, the HCALA scheme can provide privacy and anonymity, un-traceability, mutual authentication, session key agreement, integrity, and confidentiality. Furthermore, the HCALA scheme is resistant to replay attacks, denial-of-Service attacks, MTM attacks, modification attacks, physical drone capture attacks, known Session Key attacks, stolen smart device attacks, and impersonation attacks. An informal security analysis is performed to demonstrate the robustness and security of the scheme against several security attacks. Moreover, we have proved the secure authentication mechanism of HCALA using formal security analysis using "ROM" and formal security verification using the AVISPA tool. In addition, the comparison study reveals that the HCALA scheme performs better in terms of security and provides a better trade-off between efficiency and security for drones.

In the future, We would like to evaluate HCALA in a real-world environment. This will let us fine-tune HCALA as necessary to improve its security and performance in production environments.

## Declaration of competing interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

## Data availability

No data was used for the research described in the article

## References

[1] Y.-J. Chen, L.-C. Wang, Privacy protection for internet of drones: A network coding approach, IEEE Internet Things J. 6 (2) (2018) 1719–1730.
[2] S. Aggarwal, N. Kumar, Path planning techniques for unmanned aerial vehicles: A review, solutions, and challenges, Comput. Commun. 149 (2020) 270–299.
[3] M. Wazid, A.K. Das, J.-H. Lee, Authentication protocols for the internet of drones: taxonomy, analysis and future directions, J. Ambient Intell. Humaniz. Comput. (2018) 1–10.
[4] V. Cichella, R. Choe, B.S. Mehdi, E. Xargay, N. Hovakimyan, A.C. Trujillo, I. Kaminer, Trajectory generation and collision avoidance for safe operation of cooperating UAVs, in: AIAA Guidance, Navigation, and Control Conference, 2014, p. 0972.
[5] Z. Ali, S.A. Chaudhry, M.S. Ramzan, F. Al-Turjman, Securing smart city surveillance: A lightweight authentication mechanism for unmanned vehicles, IEEE Access 8 (2020) 43711–43724.
[6] M.A. Ferrag, L. Shu, X. Yang, A. Derhab, L. Maglaras, Security and privacy for green IoT-based agriculture: Review, blockchain solutions, and challenges, IEEE Access 8 (2020) 32031–32053.
[7] Q. Feng, D. He, Z. Liu, D. Wang, K.-K.R. Choo, Distributed signing protocol for IEEE P1363-compliant identity-based signature scheme, IET Inf. Secur. 14 (4) (2020) 443–451.
[8] D. He, Y. Zhang, D. Wang, K.-K.R. Choo, Secure and efficient two-party signing protocol for the identity-based signature scheme in the IEEE P1363 standard for public key cryptography, IEEE Trans. Dependable Secure Comput. 17 (5) (2018) 1124–1132.
[9] Y. Zhang, D. He, X. Huang, D. Wang, K.-K.R. Choo, J. Wang, White-box implementation of the identity-based signature scheme in the IEEE P1363 standard for public key cryptography, IEICE Trans. Inf. Syst. 103 (2) (2020) 188–195.
[10] Y.K. Ever, A secure authentication scheme framework for mobile-sinks used in the internet of drones applications, Comput. Commun. 155 (2020) 143–149.
[11] M.A. Ferrag, M. Derdour, M. Mukherjee, A. Derhab, L. Maglaras, H. Janicke, Blockchain technologies for the internet of things: Research issues and challenges, IEEE Internet Things J. 6 (2) (2018) 2188–2204.
[12] M.A. Ferrag, L. Shu, The performance evaluation of blockchain-based security and privacy systems for the internet of things: A tutorial, IEEE Internet Things J. 8 (24) (2021) 17236–17260.
[13] B. Prasanalakshmi, K. Murugan, K. Srinivasan, S. Shridevi, S. Shamsudheen, Y.-C. Hu, Improved authentication and computation of medical data transmission in the secure IoT using hyperelliptic curve cryptography, J. Supercomput. 78 (1) (2022) 361–378.
[14] M.A. Khan, I. Ullah, N. Kumar, O.S. Oubbati, I.M. Qureshi, F. Noor, F.U. Khanzada, An efficient and secure certificate-based access control and key agreement scheme for flying ad-hoc networks, IEEE Trans. Veh. Technol. 70 (5) (2021) 4839–4851.
[15] L. Vigano, Automated security protocol analysis with the AVISPA tool, Electron. Notes Theor. Comput. Sci. 155 (2006) 61–86.

[16] M. Turkanović, B. Brumen, M. Hölbl, A novel user authentication and key agreement scheme for heterogeneous ad hoc wireless sensor networks, based on the internet of things notion, Ad Hoc Netw. 20 (2014) 96–112.
[17] Y. Zhang, D. He, L. Li, B. Chen, A lightweight authentication and key agreement scheme for internet of drones, Comput. Commun. 154 (2020) 455–464.
[18] M.S. Farash, M. Turkanović, S. Kumari, M. Hölbl, An efficient user authentication and key agreement scheme for heterogeneous wireless sensor network tailored for the internet of things environment, Ad Hoc Netw. 36 (2016) 152–176.
[19] R. Amin, S.H. Islam, G. Biswas, M.K. Khan, L. Leng, N. Kumar, Design of an anonymity-preserving three-factor authenticated key exchange protocol for wireless sensor networks, Comput. Netw. 101 (2016) 42–62.
[20] S. Challa, M. Wazid, A.K. Das, N. Kumar, A.G. Reddy, E.-J. Yoon, K.-Y. Yoo, Secure signature-based authenticated key establishment scheme for future IoT applications, IEEE Access 5 (2017) 3028–3043.
[21] X. Jia, D. He, L. Li, K.-K.R. Choo, Signature-based three-factor authenticated key exchange for internet of things applications, Multimedia Tools Appl. 77 (2018) 18355–18382.
[22] M. Wazid, A.K. Das, N. Kumar, A.V. Vasilakos, J.J. Rodrigues, Design and analysis of secure lightweight remote user authentication and key agreement scheme in internet of drones deployment, IEEE Internet Things J. 6 (2) (2018) 3572–3584.
[23] L. Chen, S. Qian, M. Lim, S. Wang, An enhanced direct anonymous attestation scheme with mutual authentication for network-connected UAV communication systems, China Commun. 15 (5) (2018) 61–76.
[24] M. Tanveer, A.H. Zahid, M. Ahmad, A. Baz, H. Alhakami, LAKE-IoD: Lightweight authenticated key exchange protocol for the internet of drone environment, IEEE Access 8 (2020) 155645–155659.
[25] T. Alladi, V. Chamola, N. Kumar, et al., PARTH: A two-stage lightweight mutual authentication protocol for UAV surveillance networks, Comput. Commun. 160 (2020) 81–90.
[26] J. Srinivas, A.K. Das, N. Kumar, J.J. Rodrigues, TCALAS: Temporal credential-based anonymous lightweight authentication scheme for internet of drones environment, IEEE Trans. Veh. Technol. 68 (7) (2019) 6903–6916.
[27] G. Cho, J. Cho, S. Hyun, H. Kim, SENTINEL: A secure and efficient authentication framework for unmanned aerial vehicles, Appl. Sci. 10 (9) (2020) 3149.
[28] B. Bera, D. Chattaraj, A.K. Das, Designing secure blockchain-based access control scheme in IoT-enabled internet of drones deployment, Comput. Commun. 153 (2020) 229–249.
[29] B. Bera, S. Saha, A.K. Das, N. Kumar, P. Lorenz, M. Alazab, Blockchain-envisioned secure data delivery and collection scheme for 5g-based iot-enabled internet of drones environment, IEEE Trans. Veh. Technol. 69 (8) (2020) 9097–9111.
[30] M. Nikooghadam, H. Amintoosi, S.H. Islam, M.F. Moghadam, A provably secure and lightweight authentication scheme for internet of drones for smart city surveillance, J. Syst. Archit. 115 (2021) 101955.
[31] S. Hussain, K. Mahmood, M.K. Khan, C.-M. Chen, B.A. Alzahrani, S.A. Chaudhry, Designing secure and lightweight user access to drone for smart city surveillance, Comput. Stand. Interfaces 80 (2022) 103566.
[32] M. Tanveer, N. Kumar, M.M. Hassan, et al., RAMP-IoD: A robust authenticated key management protocol for the internet of drones, IEEE Internet Things J. 9 (2) (2021) 1339–1353.
[33] M. Tanveer, A. Alkhayyat, A. Naushad, N. Kumar, A.G. Alharbi, et al., RUAM-IoD: A robust user authentication mechanism for the internet of drones, IEEE Access (2022).
[34] S. Javed, M.A. Khan, A.M. Abdullah, A. Alsirhani, A. Alomari, F. Noor, I. Ullah, An efficient authentication scheme using blockchain as a certificate authority for the internet of drones, Drones 6 (10) (2022) 264.
[35] T. Wollinger, J. Pelzl, C. Paar, Cantor versus Harley: optimization and analysis of explicit formulae for hyperelliptic curve cryptosystems, IEEE Trans. Comput. 54 (7) (2005) 861–872.
[36] T. Wollinger, J. Pelzl, V. Wittelsberger, C. Paar, G. Saldamli, Ç.K. Koç, Elliptic and hyperelliptic curves on embedded $\mu$p, ACM Trans. Embed. Comput. Syst. (TECS) 3 (3) (2004) 509–533.
[37] I. Ullah, N. Ul Amin, M. Zareei, A. Zeb, H. Khattak, A. Khan, S. Goudarzi, A lightweight and provable secured certificateless signcryption approach for crowdsourced IIoT applications, Symmetry 11 (11) (2019) 1386.
[38] J. Pelzl, T. Wollinger, J. Guajardo, C. Paar, Hyperelliptic curve cryptosystems: Closing the performance gap to elliptic curves, in: International Workshop on Cryptographic Hardware and Embedded Systems, Springer, 2003, pp. 351–365.
[39] Wiki, Hyperelliptic curve, 2022, URL https://en.wikipedia.org/wiki/Hyperelliptic_curve.
[40] V.S. Naresh, R. Sivaranjani, N. VES Murthy, Provable secure lightweight hyper elliptic curve-based communication system for wireless sensor networks, Int. J. Commun. Syst. 31 (15) (2018) e3763.
[41] Y. Soni, L. Maglaras, M.A. Ferrag, Blockchain based voting systems, in: European Conference on Cyber Warfare and Security, Academic Conferences International Limited, 2020, pp. 241–248.
[42] D. Hamouda, M.A. Ferrag, N. Benhamida, H. Seridi, PPSS: A privacy-preserving secure framework using blockchain-enabled federated deep learning for industrial IoTs, Pervasive Mob. Comput. (2022) 101738.
[43] M.S. Eddine, M.A. Ferrag, O. Friha, L. Maglaras, EASBF: An efficient authentication scheme over blockchain for fog computing-enabled internet of vehicles, J. Inf. Secur. Appl. 59 (2021) 102802.
[44] A. Derhab, M. Guerroumi, A. Gumaei, L. Maglaras, M.A. Ferrag, M. Mukherjee, F.A. Khan, Blockchain and random subspace learning-based IDS for SDN-enabled industrial IoT security, Sensors 19 (14) (2019) 3119.
[45] D. Dolev, A. Yao, On the security of public key protocols, IEEE Trans. Inform. Theory 29 (2) (1983) 198–208.
[46] R. Canetti, H. Krawczyk, Universally composable notions of key exchange and secure channels, in: International Conference on the Theory and Applications of Cryptographic Techniques, Springer, 2002, pp. 337–351.
[47] B. Blanchet, Modeling and verifying security protocols with the applied pi calculus and ProVerif, Found. Trends® Priv. Secur. 1 (1–2) (2016) 1–135.
[48] G. Lowe, P. Broadfoot, M.L. Hui, Casper: a compiler for the analysis of security, in: Protocols Proceedings of the 1997, IEEE. Computer Society Symposium on Research in Security and Privasy, 1997, pp. 18–30.
[49] C.J. Cremers, The scyther tool: Verification, falsification, and analysis of security protocols, in: International Conference on Computer Aided Verification, Springer, 2008, pp. 414–418.
[50] V. Odelu, A.K. Das, A. Goswami, An efficient biometric-based privacy-preserving three-party authentication with key agreement protocol using smart cards, Secur. Commun. Netw. 8 (18) (2015) 4136–4156.
[51] H.H. Kilinc, T. Yanik, A survey of SIP authentication and key agreement schemes, IEEE Commun. Surv. Tutor. 16 (2) (2013) 1005–1023.
[52] M.A. Khan, I. Ullah, S. Nisar, F. Noor, I.M. Qureshi, F.U. Khanzada, N.U. Amin, An efficient and provably secure certificateless key-encapsulated signcryption scheme for flying ad-hoc network, IEEE Access 8 (2020) 36807–36828.
[53] A.D.E. Berini, B. Farou, M.A. Ferrag, H. Seridi, H. Akdag, A new static path planning strategy for drones, Internet Technol. Lett. 5 (6) (2022) e386.
[54] C. Pu, A. Wall, I. Ahmed, K.-K.R. Choo, SecureIoD: A secure data collection and storage mechanism for internet of drones, in: 2022 23rd IEEE International Conference on Mobile Data Management, MDM, IEEE, 2022, pp. 83–92.