The Journal of Supercomputing
https://doi.org/10.1007/s11227-024-06447-5

Blockchain-based cross-domain authentication in a multi-domain Internet of drones environment

Arivarasan Karmegam[1] · **Ashish Tomar**[2] · **Sachin Tripathi**[3]

Accepted: 11 August 2024

Abstract
As a new paradigm, the Internet of drones (IoD) is making the future easy with its flexibility and wide range of applications. However, these drones are prone to security attacks during communication because of this flexibility. The traditional authentication mechanism uses a centralized server which is a single point of failure to its network and a performance bottleneck. Also, privacy-preserving mechanisms involving a single authority are vulnerable to identity attacks if compromised. Moreover, cross-domain authentication schemes are getting more costly as the security requirements increase. So, this work proposes a blockchain-based cross-domain authentication scheme to make drone communication more secure and efficient. In this work, an elliptic curve digital signature algorithm (ECDSA) based message authentication scheme and a session key generation scheme are modeled. A two-phase pseudonym generation procedure is used to secure the identity of the drones. Hyperledger Fabric is used to implement the blockchain network, and the analysis is done using Hyperledger Caliper. Blockchain analysis through caliper shows the blockchain's performance for various loads of transactions. Security analysis of the proposed scheme shows that the scheme is secure from various security attacks. The performance analysis shows that the proposed scheme is more lightweight and efficient than most similar authentication schemes.

Keywords Blockchain · Internet of drones · Authentication · Cross-domain · Scyther

1 Introduction

Internet of things (IoT), with its endless applications, is changing the world in a rapid phase. By enabling a vast range of devices to communicate among themselves, IoT is making our lives easier and doing humanly impossible things. From

A. Tomar, S. Tripathi have contributed equally to this work.

Extended author information available on the last page of the article

Published online: 05 September 2024

smart home systems to smart health service systems, smart traffic management to agriculture automation, humankind has taken a great leap [1].

Unmanned aerial vehicles (UAVs), often called drones, are aircraft that operate without a human pilot. Primarily they are controlled from a ground control station by a human using a remote controller or by some automation algorithm, which calculates the air route using the sensors and GPS devices present onboard [2]. Research on the usage of drones has become a hot topic in the recent past, thanks to their autonomy, flexibility and wide range of applications [3, 4]. The Internet of drones (IoD) is a recent paradigm and a specific application of IoT. IoD, as architecture, is used to control and coordinate many drones to achieve various tasks. These tasks include but are not limited to wildlife surveillance, rescue missions, military operations and likewise. As sensors get smaller and more effective, the usage of IoD has become more necessary than ever [5, 6].

A centralized authentication server is used for data storage and processing in a traditional IoD environment, which is a single point of failure. Also, as the system scales, the centralized server becomes a performance bottleneck [7]. Here comes the need for a distributed IoD model, where the operations are shared among more than one server or managing node. This distributed model though effective brings with it its own data replication issues.

Consider a situation where drones from different domains must work on a single task. Because of their autonomy and flexibility, as mentioned above, these drones are vulnerable to different types of attacks. So each drone must authenticate other drones before communicating while preserving its identity. However, each domain may have a different authentication mechanism and registering each drone in all other domains is tedious and time-consuming. Though establishing a cross-domain connection through existing network infrastructures is possible, the security of the communication is still in question because the domains do not necessarily need to trust each other to the extent that they share confidential data among themselves [8, 9]. So, a distributed privacy-preserving authentication mechanism is needed where the drones from one domain can authenticate in other domains without going through a complex and separate registration procedure in every other domain.

Blockchain as a concept was first conceived and implemented by Satoshi Nakamoto to create the famous cryptocurrency, Bitcoin [10]. Blockchain, a secure distributed ledger, has attracted researchers' attention in the recent past for its characteristics like immutability, fault-tolerance and consensus mechanisms [11, 12]. For its distributed property and transparency, blockchain is used in various fields, like IoD, smart grids and VANETs. To address challenges discussed in previous passages and considering blockchain's features, many blockchain-based authentication schemes for distributed scenarios like UAVs and industrial Internet of things (IIOT) are modeled [13–17]. Both public and consortium blockchains are used in these works, each with its own advantages and disadvantages. For cryptography, digital signature and signcryption techniques are used with cryptographic frameworks like PKI-based, identity-based and certificateless cryptography. But these works have some scope for improvement which forms our motivation for this work.

1.1 Motivation

Most authentication schemes use pairing-based certificateless cryptography or heavy signature generation and verification procedures, which is computationally costly for an IoT environment. IoT devices are primarily stand-alone devices with limited power resources. So if the computational cost for authentication is huge, most of its power will be spent on authentication, and little energy will be left to do the work for which the device has been installed. Also, in the pseudonym-based techniques, the pseudonym is generated by a single trust authority, and if it is compromised, the drones under the specific authority will become vulnerable to identity attacks. Keeping the above vulnerabilities and problems in mind, a lightweight privacy-preserving authentication scheme for IoD is proposed, with a hierarchical batch-pseudonym generation and revocation procedure.

1.2 Contributions

1.2.1 This section summarizes our main contributions in this work

1. A blockchain-based cross-domain authentication scheme for IoD Scenario is designed, which uses ECC operations with a lightweight signature generation and verification procedure.
2. A hierarchical batch-pseudonym generation and revocation procedure is modeled, ensuring conditional privacy preservation and efficient identity management, along with a drone-to-drone session key generation method for secure communication.
3. The proposed scheme is implemented using Hyperledger Fabric, and through performance analysis using Hyperledger Caliper, and through formal and informal security analyses, we demonstrate the scheme's robustness against various attacks and its efficiency over existing methods.

The rest of the manuscript is organized as follows: Sect. 2 provides a review of the relevant literature for the proposed work. Section 3 illustrates the preliminary concepts required and the system architecture of the scheme. Section 4 explains the detailed steps of the proposed scheme. Section 5 presents the relevant security analysis of the proposed scheme, both practical and theoretical. Section 6 describes the performance of the proposed scheme, and Sect. 7 concludes the proposed scheme.

2 Related works

Though blockchain is still in its initial stages of development, considerable work has been done in authentication schemes using blockchain. Traditional PKI-based authentication scheme has a certificate authority (CA), which provides the IoT devices with their certificates. A device which wants to authenticate another device

requests the corresponding certificate from the CA. However, this CA has a massive overhead of certificate management and becomes a single failure point. ID-based cryptography was introduced to address the problems of the PKI framework. This method uses the device's ID as the public key. But this method had problems, such as key escrow issues. In 2003, Al-Riyami and Paterson [18] introduced certificateless public key cryptography (CL-PKC), which addresses the key escrow problem in ID-based cryptography. Following this, many authentication models have been designed by researchers using the CL-PKC framework.

Zheng et al. [19] proposed a new cryptographic framework called signcryption for the first time. The signcryption mechanism addresses confidentiality and authentication problems by combining the advantages of signature and encryption. Using this as a primitive, Xu et al. [15] proposed a certificateless signcryption mechanism based on blockchain. This work uses bilinear pairing-based cryptography. In this work, the blockchain network consists of multiple edge servers and is used as a public key directory. The message is signcrypted and sent to the receiver, where the receiver verifies the signcryption using the items sent and the sender details queried from the blockchain. A blockchain using the Go language is developed for this purpose. This work uses random oracle model for security analysis.

Zhang et al. [20] proposed a certificateless message signature authentication for traffic reporting in VANETs. The scheme uses bilinear pairing-based cryptography, and an adaptive t-threshold multi-signature mechanism for aggregate signature verification is modeled. In this work, blockchain is used as a tool to provide incentives. A cryptocurrency named TCoin is introduced in this paper. For every genuine violation report, the reporter gets rewarded with TCoins. This work shows one of the various ways in which blockchain can be used.

A pairing operation is much costlier than an ECC scalar multiplication operation [21, 22]. Based on this fact, many authors used ECC for their certificateless authentication scheme. Cheng et al. [14] proposed a blockchain-based mutual authentication scheme. This scheme enables mutual authentication between edge servers and IoT devices and incorporates certificateless, elliptic curve and pseudonym-based cryptography. Authors design authentication mechanisms for static conditions and dynamic intra-edge and inter-edge communication. Registration and key generation are done using smart contracts. Hyperledger Fabric is used to implement the blockchain network. The security analysis proves that this work is secure from more network attacks than the other existing authentication schemes.

Also, some theoretical work and its improvement on pairing-free certificateless signature have been done in Refs. [23, 24]. Based on this, Wang et al. [17] proposed a pairing-free certificateless scheme that uses blockchain technology and its smart contract to design a CLS-based scheme. The proposed scheme uses smart contract for key generation and distribution. Nevertheless, this work does not discuss the registration of devices, so what type of devices approach this network and how they register to the network before key generation is unclear [25].

CLS schemes address the key escrow problem and the centralized key generation center problem to some extent, but the KGC problem is not completely avoidable by CLS-based schemes and is likely to become a single point of failure. So, researchers started using signature-based authentication schemes where

the cryptography is mostly based on elliptic curve discrete logarithmic problem (ECDLP) [26] and computational Diffie–Hellman problem (CDHP) [27, 28]. Tan et al. [16] propose a blockchain-assisted authentication scheme for UAVs. The proposed signature-based authentication is a simple but effective authentication model where the blockchain is used to store the authentication information of UAVs securely. But, in this work, a drone's frequent update of the one-time public key costs its power consumption due to frequent blockchain access. Also, as only *GCS* participates in the privacy-preserving mechanism, the identities of drones will be leaked if *GCS* is compromised.

Feng et al. [7] proposed a blockchain-based cross-domain authentication scheme for IoD. They design a threshold-based multi-signature approach for identity management in collaborative domains. Different approaches for intra-domain and inter-domain authentication are designed. The author mentions that the consensus mechanism takes more time as the number of domains increases, and in addition to the latency caused by the consensus mechanism, the threshold-based multi-signature approach also adds to the latency of the scheme. Recently, Luo et al. [29] introduced an identity authentication for cross-domain IoT based on split-chain mechanism. They designed the split-chain structure of blockchain in such a way that it increases the authentication efficiency. Chen et al. [30] considered VANET scenario and proposed a certificate-based cross-domain authentication mechanism with efficient batch verification. To avoid single point of failure, they presented a two-way synchronized database. So, many existing works do not address the cross-domain scenario, and those who address this problem are not efficient enough for an IoT environment.

3 Preliminaries

This section discusses some essential concepts whose understanding is required for the proposed scheme.

3.1 Elliptic curve cryptography

Elliptic curve cryptography is an approach to public key cryptography based on elliptic curves defined over a finite field and the operations defined on that curves [31]. For ECC, the curves are defined as

$$E = (x, y)|y^2 = x^3 + ax + b \tag{1}$$

where $4a^3 + 27b^2 \neq 0$.

The proposed authentication scheme is built based on elliptic curve discrete logarithm problem (ECDLP). Given an elliptic curve group G, let P be a generator of G, and Q be an element of G, then finding a scalar point $x \in Z_q^*$ such that $Q = xP$ holds, in polynomial time is almost impossible.

3.2 Blockchain

Blockchain is a distributed ledger technology (DLT) where the blocks containing the transaction details of the network are securely linked using a cryptographic hash. Each block includes the cryptographic hash of the previous block, and the link grows. Blockchain, a peer-to-peer network, has nodes that participate in the smart-contract execution, by which the irreversible transaction is created, verified and stored in blocks using different consensus mechanisms. Based on the accessibility requirements, blockchain is categorized into public blockchain, private blockchain and consortium blockchain. In the proposed scheme, Hyperledger Fabric, a consortium blockchain, is used to create blockchain network. According to its documentation, Hyperledger Fabric is an open-source enterprise-grade permissioned DLT platform [32]. A consortium blockchain has organizations as its participant. Each organization is a private entity and the employees of the organization form the nodes of the blockchain. Identity of each employee is known to the organization. The key components that makeup Hyperledger Fabric Network are described below.

1. *Peer:* Peer forms the basic node of Fabric Network, and they are fundamental because they manage ledgers, transaction proposals and also endorsements if assigned.
2. *Orderer:* Orderer or ordering node does the transaction ordering, creates the block and distributes the block among the peers for endorsement. They enforce basic access control over channels, limiting who can read and write data to them and who can configure them.
3. *Channel:* Channel is a medium used by components within a fabric blockchain network to communicate and transact privately. A channel can have multiple peers as a member, and a peer can be a part of multiple channels.
4. *Ledger:* In Hyperledger Fabric, a world state and a blockchain combine to form a ledger. World state is a database which holds the current values of the ledger states. The ledger state is normally expressed as a key pair value, and the current ledger state can be easily accessed from world state without going through the transaction history. Blockchain is a transaction log which records the transaction details from the beginning that led to this ledger state. This blockchain is immutable.
5. *Smart Contract and Chaincode:* A smart contract is the agreed rules between organizations and the business logic written in the form of executable code. The smart contract is invoked to generate transactions and interact with the ledger. Hyperledger Fabric uses the terms smart contract and chaincode interchangeably. A smart contract written in executable code is wrapped in a chaincode deployed on a blockchain network.

3.3 System architecture and assumptions

The system architecture of the proposed scheme is given in Fig. 1. A consortium blockchain is chosen for this work, as we assume that the participating domains are a high level organization who have their participant's identity confirmed. This gives confidentiality which is lacking in public blockchain. Though it is normally not advised to upload sensitive information on blockchain without encryption, using a consortium blockchain provides us an advantage over works on public blockchain in terms of confidentiality with respect to nodes of the blockchain network. The components of this architecture and assumptions are explained below.

1. *Control Authority:* Control authority (CoA) is the trusted authority of this model and is assumed to be made up of members from every organization, which is a part of this network, for neutral decision-making. *CoA* is responsible for completing the hierarchical pseudonym generation and distribution to drones. It is a member of the blockchain network and uploads pseudonym details to the blockchain. It also takes care of the revocation of pseudonyms of rogue drones.
2. *Domain:* A domain in this blockchain network is an organization or a group which wants to collaborate with other domains in the network to perform a specific task. This organization is similar to the organization's definition in a consortium blockchain. There can be any number of domains in this network.
3. *Ground Control Station:* Ground control station (GCS) is a semi-trusted authority which acts as the administrator of a domain. It is responsible for registering drones and generating cryptographic materials such as drone keys and partial pseudonyms for authentication purposes. It also uploads public cryptographic materials and the drone's license to the blockchain. In this work, only a single GCS is assumed for each domain. This can be a single point of failure for normal cases, but for a blockchain-based network, the drones can contact CoA for

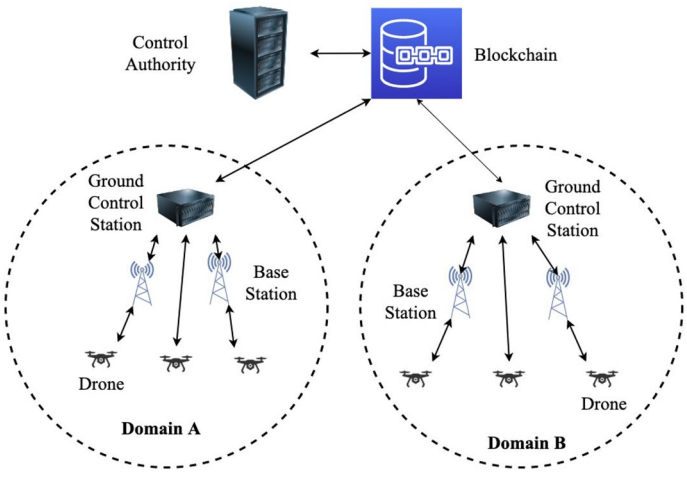


Fig. 1 System Architecture

authentication purposes, which has a replica of the blockchain ledger. Only the registration is disturbed during a down time of GCS.

4. *Base Station:* Base station (BS) of a particular domain is distributed area-wise in the real world. Base station bridges the communication between *GCS* and drones when the distance is huge. It is assumed that the communication between *GCS* and base station happens through a wired channel.
5. *Drone:* Drones are the IoT devices in this scheme and are capable of moving around the mission area and collecting data. Each drone is equipped with the necessary hardware and software to perform cryptographic operations using elliptic curve cryptography (ECC).
6. *Blockchain:* Blockchain in this work is implemented through Hyperledger Fabric. The interaction with the blockchain happens by invoking a chaincode, which reads, writes and updates the ledger.

3.4 Security goals

The proposed authentication scheme will achieve the following security goals.

1. *Confidentiality:* The data being communicated among the drones should remain secret.
2. *Decentralization:* The authentication should not be dependent on a single trust authority but should be managed by multiple parties. There should not be any node in the network, which is a single point of failure.
3. *Mutual Authentication:* The drones should be able to authenticate each other.
4. *Identity Protection:* Pseudonyms are generated for drones, which are used during communication instead of real identities. There is no way the other members of the network or attackers can get to know the real identity of a drone.
5. *Traceability:* When some malicious drones are reported or detected, the authorities concerned should be able to trace the real identity of the drone.
6. *Batch-Pseudonym Generation and Revocation:* Pseudonyms for drones should be generated in batches. The corresponding drone's pseudonym batch should be revoked in case of malicious activity.
7. *Resistance against Common Security Attacks:* The proposed scheme should be resistant to various security attacks such as identity forgery, impersonation, replay attack, DDoS attack, modification attack and likewise.

Algorithm 1 Chaincode functions for different operations

```
   Input: Parameters for security and interface.
1  function (s *SmartContract) CreateAsset(ctx
     contractapi.TransactionContextInterface, drpk, did, gpk, v, treg):
2  |  exists := s.AssetExists(ctx, drpk)
   |   if exists = True then
3  |  |  return fmt.Errorf("Exists", drpk)
4  |  end
5  |  asset := Asset{Dr_PK: drpk, D_ID: did, G_PK: gpk, V: v, T_reg: treg,}
   |   assetJSON, err := json.Marshal(asset)
   |
6  |  return ctx.GetStub().PutState(drpk, assetJSON);
7  End

   Input: Identity
   Output: Authentication parameters
8  funcion (s *SmartContract) ReadAsset(ctx
     contractapi.TransactionContextInterface, id):
9  |  assetJSON := ctx.GetStub().GetState(id)
   |   if assetJSON == nil then
10 |  |  return nil, fmt.Errorf("Not existing", id)
11 |  end
12 |  var asset Asset
   |   err = json.Unmarshal(assetJSON, &asset)
   |
13 |  return &asset, nil
14 End

   Input: Security parameters
15 funcion (s *SmartContract) UpdateAsset(ctx
     contractapi.TransactionContextInterface, drpk, did, gpk, v, treg):
16 |  exists := s.AssetExists(ctx, drpk) if !exists then
17 |  |  return fmt.Errorf("Not existing", drpk)
18 |  end
19 |  asset := Asset{Dr_PK: drpk, D_ID: did, G_PK: gpk, V: v, T_reg: treg,}
   |   assetJSON, err := json.Marshal(asset)
   |
20 |  return ctx.GetStub().PutState(drpk, assetJSON);
21 End

   Input: Identity
22 funcion (s *SmartContract) DeleteAsset(ctx
     contractapi.TransactionContextInterface, id):
23 |  exists := s.AssetExists(ctx, id)
   |   if !exists then
24 |  |  return fmt.Errorf("Not existing", id)
25 |  end
26 |  return ctx.GetStub().DelState(id)
27 End
```

4 Proposed scheme

In this section, the details of each phase of the proposed cross-domain authentication scheme are discussed in detail. It begins with system initialization in Sect. 4.1, where the control authority (CoA) sets up the elliptic curve parameters and generates cryptographic materials. In Sect. 4.2, drone registration follows, describing the offline process where drones receive their keys and partial pseudonyms from the ground control station (GCS). Next, pseudonym generation and distribution in Sect. 4.3 explains how the CoA completes the pseudonym generation using one-way hash chains and distributes them to the drones. Message Authentication in Sect. 4.4 details the steps for drones to authenticate each other using the generated pseudonyms and licenses stored on the blockchain. The session key generation Sect. 4.5 outlines the process for establishing secure communication between drones after mutual authentication. Drone license revocation in Sect. 4.6 describes the procedure for revoking the pseudonyms and licenses of malicious drones to maintain network security. Algorithm 1 shows the chaincode functions for different operations. Figure 2 shows the complete flow of the proposed scheme.

4.1 System initialization

CoA, the neutral trust authority in this scheme, defines the endorsement policies. *CoA* sets a non-singular elliptic curve $\mathbb{E}(p)$, defined by a large prime number p. *CoA* defines additive group $\mathbb{G}$, consisted by points from the curve $\mathbb{E}(p)$, with order q and generator P. O is the point at infinity. Four hash functions $h_1 : \{0,1\}^* \rightarrow \mathbb{Z}_q^*$, $h_2 : \{0,1\}^* \rightarrow \mathbb{Z}_q^*$, $h_3 : \{0,1\}^* \rightarrow \mathbb{Z}_q^*$, $h_4 : \{0,1\}^* \rightarrow \{0,1\}^k$. *CoA* generates its key pairs $\{c_{\text{sk}}, C_{\text{pk}}\}$ where $C_{\text{pk}} = c_{\text{sk}}.P$. Each *GCS* generates its own key pair $\{g_{\text{sk}}, G_{\text{pk}}\}$. *GCS* also generates key pairs $\{b_{\text{sk}_n}, B_{\text{pk}_n}\}$ for base stations belonging to its domain and the secret key is preloaded into its local memory. Public parameters and public keys are broadcasted by *CoA* to all entities in the network and are recorded on the blockchain. *GCSs* uploads base stations' and their public key to the blockchain. Some of the important notations used in this work are summarized in Table 1.

4.2 Drone registration

Drone registration is done offline (Fig. 3). Drone dr_i gets its real ID *RID* during production, which depends on the manufacturer, its model, manufactured date and other details. The drone submits its real ID RID_i and other legal details to *GCS* of its domain.

1. *GCS* verifies the documents of the drone. If the documents are valid, then the *GCS* proceeds with registration.

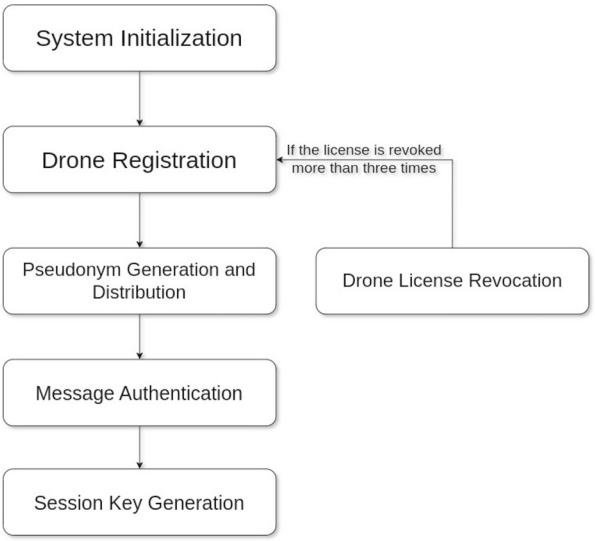


Fig. 2 Flow of the proposed scheme

Table 1 Notation Table

Notation	Description
p	Large prime number
q	Prime order
P	Generator of additive group $\mathbb{G}$
h_1, h_2, h_3, h_4	Hash Functions
$c_{\text{sk}}, C_{\text{pk}}$	Secret Key and Public Key of CoA
$g_{\text{sk}}, G_{\text{pk}}$	Secret Key and Public Key of GCS
$d_{\text{sk}_i}, D_{\text{pk}_i}$	Secret Key and Public Key of Drone dr_i
x_i, X_i	Cryptographic Material for Drone dr_i
RID_i	Real Identity of Drone dr_i
TID_i	Partial Pseudonym of Drone dr_i
D_{id}	Domain ID
V_i	Validity of the Drone dr_i's License
$T_{\text{reg}}, T_i^{\text{req}}, T_i^{\text{res}}, T_i^{\text{auth}}$	Timestamps

2. *GCS* generates the drone's key pair as

$$d_{\text{sk}_i} = h_1\left(a||\text{RID}||T_{\text{reg}}\right) \tag{2}$$

$$D_{\text{pk}_i} = d_{\text{sk}_i} \cdot P \tag{3}$$

where a is a random *nonce* T_{reg} is the timestamp of the registration.

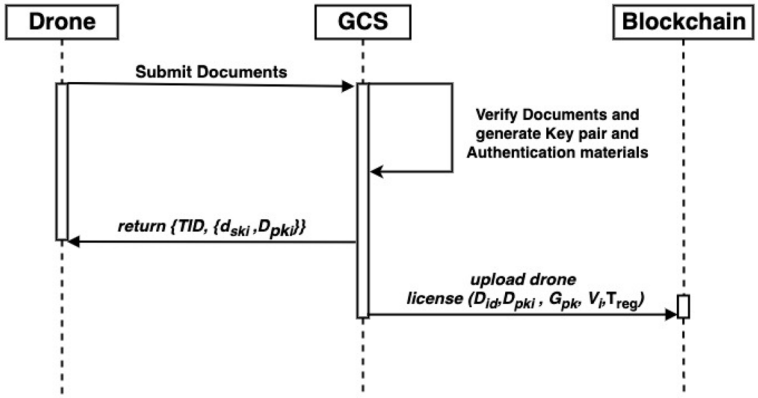


Fig. 3 Drone Registration

3. *GCS* generates the partial pseudonym as

$$\mathrm{TID}_i = \mathrm{RID}_i \oplus h_2\left(g_{\mathrm{sk}} \cdot T_{\mathrm{reg}}\right) \tag{4}$$

4. *GCS* generates a cryptographic material as $x_i \in Z_q^*$ and $X_i = x_i.P$.
5. *GCS* sends $\{\mathrm{TID}_i, x_i, \{d_{\mathrm{sk}_i}, D_{\mathrm{pk}_i}\}\}$ to the drone dr_i.
6. *GCS* generates license for drone dr_i as $\{D_{\mathrm{id}}, D_{\mathrm{pk}_i}, G_{\mathrm{pk}}, X_i, V_i, T_{\mathrm{reg}}\}$ and records it to the blockchain, where D_{id} is the domain id and V_i is the validity of the license. The validity will be either *true* or *false*.

4.3 Pseudonym generation and distribution

During the drone registration, the drone's partial pseudonym TID_i is generated by *GCS*. *CoA* completes this pseudonym generation process (Fig. 4). *CoA* generates the cryptographic materials required for the pseudonym generation phase well before a drone approaches it. One-way hash chains are cryptographic hashing techniques used to generate a set of keys from a single seed [33]. Lamport introduced it in 1981 for securing passwords from intruders [34]. This technique uses a seed and a cryptographic hash function. The hash function is applied successfully to the seed to produce a set of hash values which is computationally impossible to invert. *CoA* generates a batch of seeds as $\{s_1, s_2, \ldots s_m\}$. For every seed, it generates a hash chain as

$$H^N(s) \leftarrow H^{N-1}(s) \ldots H^1(s) \leftarrow s. \tag{5}$$

From this hash chain, *CoA* calculates the *k*th key as

$$e_k = \sum_{i=1}^{k} H^i(s) (\mathrm{mod}\ q) \tag{6}$$

Drone dr_i sends $P_{\mathrm{req}} = \{D_{\mathrm{id}}, D_{\mathrm{pk}_i}, \mathrm{TID}_i, T_i^{\mathrm{req}}\}$ to *CoA* with a hash

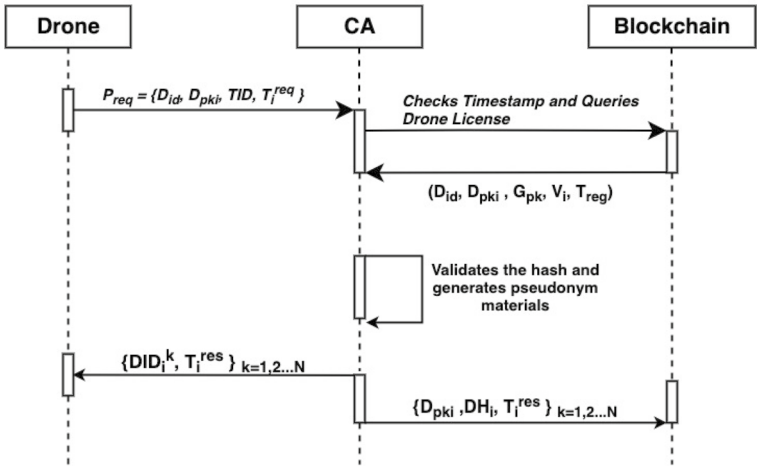


Fig. 4 Pseudonym Generation and Distribution

$$\alpha_i = d_{\mathrm{sk}_i} + x_i \cdot h_3\left(G_{\mathrm{pk}} || P_{\mathrm{req}}\right) \tag{7}$$

where T_{req} is the timestamp of the pseudonym request.

1. *CoA* first checks the timestamp T_i^{req}. If it is valid, then it queries drone's license $\{D_{\mathrm{id}}, D_{\mathrm{pk}_i}, G_{\mathrm{pk}}, X_i, V_i, T_{\mathrm{reg}}\}$ from the blockchain.
2. *CoA* checks the validity status V_i. If it is true, it checks the hash α_i using the public keys queried from the blockchain as

$$\alpha_i.P = D_{\mathrm{pk}_i} + X_i \cdot h_3\left(G_{\mathrm{pk}} || P_{\mathrm{req}}\right) \tag{8}$$

3. If it holds, then *CoA* calculates a batch of pseudonym material as

$$\mathrm{DID}_i^k = \mathrm{TID}_i \oplus h_2\left(e_i^k \cdot T_i^{\mathrm{res}}\right) \tag{9}$$

and sends $\{\mathrm{DID}_i^k, T_i^{\mathrm{res}}\}_{k=1,2\ldots N}$ to drone where T_i^{res} is the timestamp of the query response.

4. *CoA* stores $\{D_{\mathrm{pk}_i}, \mathrm{DID}_i^k, T_i^{\mathrm{res}}\}_{k=1,2\ldots N}$ and the corresponding master seed s_m in the local storage. If any request for revocation is received, these materials can be used to revoke the pseudo-identity.
5. *CoA* uploads $\{D_{\mathrm{pk}_i}, \mathrm{DH}_i, T_i^{\mathrm{res}}\}_{k=1,2\ldots N}$ to the blockchain, where DH_i is the hash value of DID_i^k, which enables a transparent traceability in the blockchain network.

After receiving the pseudonym materials, drone dr_i generates

$$l_k^i \in Z_q^* \tag{10}$$

$$L_k^i = l_k^i \cdot P \tag{11}$$

and obtains pseudonym as $\text{PID}_k^i = \{L_k^i, \text{DID}_k^i, T_i^{\text{res}}\}_{k=1,2\ldots N}$.

4.4 Message authentication

If drone dr_i wants to communicate to drone dr_j, it happens as follows (Fig. 5). Let M be the message to be communicated. Each pseudonym is used only once for security purposes.

1. dr_i calculates

$$y = l_k^i + d_{\text{sk}_i}.h_3\left(\text{PID}_k^i||M||T_i^{\text{auth}}\right) \tag{12}$$

and sends $\text{M\_Auth} = \{D_{\text{id}}, \text{PID}_k^i, M, T_i^{\text{auth}}, y\}$ to dr_j, where T_i^{auth} is the timestamp of authentication request.
2. After receiving M_Auth, dr_j verifies the timestamp T_i^{auth} and the timestamp of the pseudonym T_i^{res}, which ensures the freshness of both the request and the pseudonym. A batch of pseudonym is valid for a week.
3. If the timestamps are fresh, then dr_j checks whether DID_k^i is in the revocation list. If not, the receiver queries dr_i's license from its *GCS* or nearest *BS*.
4. The *GCS* in turn queries the drone license from the ledger and sends the license $\{D_{\text{id}}, D_{\text{pk}_i}, G_{\text{pk}}, X_i, V_i, T_{\text{reg}}\}$ to dr_j.
5. dr_j verifies the validity status V_i. If it is *true*, then the hash-based signature y is verified as

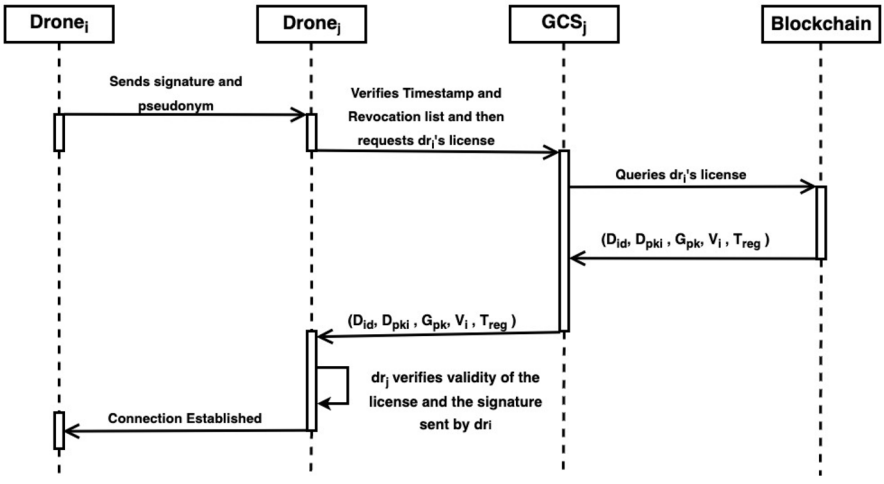


Fig. 5 Message Authentication

$$y \cdot P = L_k^i + D_{\mathrm{pk}_i} \cdot h_3(\mathrm{PID}_k^i || M || T_i^{\mathrm{auth}}). \tag{13}$$

If it does not hold, the connection is refused. If it holds, dr_j also follows the same process discussed above and gets verified with dr_i.

4.5 Session key generation

After mutual authentication between the two drones, they have each other's licenses and pseudonym details. So if a necessity arises for further communication, they generate a session key as follows.

1. Drone dr_i, using the elements of the license and pseudonym of drone dr_j generates the session key $\mathrm{ssk}_{ij} = h_4((l_i^k + d_{\mathrm{sk}_i}) \cdot (L_j^k + D_{\mathrm{pk}_j}))$.
2. Drone dr_i selects a random $nonce \in Z_q^*$ sends a session request to drone dr_j as $\mathrm{ses\_req} = \mathrm{ssk}_{ij}(\mathrm{DID}_i^k || \mathrm{nonce})$.
3. Drone dr_j upon receiving the ses_req generates the session key as $\mathrm{ssk}_{ij} = h_4((l_j^k + d_{\mathrm{sk}_j}) \cdot (L_i^k + D_{\mathrm{pk}_i}))$.
4. Then drone dr_j decrypts the request sent by drone dr_i and checks whether DID_k^i is present in the revocation list. If not, the drone generates a session response as $\mathrm{ses\_res} = \mathrm{ssk}_{ij}(\mathrm{DID}_j^k || \mathrm{nonce} + 1)$.
5. Drone dr_j saves the session key ssk_{ij} with an id $h_4(\mathrm{DID}_i^k || \mathrm{DID}_j^k)$ for future reference and initiates a timeout counter to monitor inactivity.
6. Drone dr_j upon receiving the ses_res, decrypts the information and checks whether DID_j is in the revocation list. If not, then it confirms the correctness of nonce + 1.
7. If it is valid, then drone dr_i too saves the session key ssk_{ij} with an id $h_4(\mathrm{DID}_i^k || \mathrm{DID}_j^k)$ for future reference and initiates a timeout counter to monitor inactivity.
8. If the session key ssk_{ij} remains inactive for a predefined period, it is invalidated, and a new session key must be generated for future communications.

Thus a connection is established.

4.6 Drone license revocation

In an IoD scenario, it is necessary to constantly check the drones and take action on any malicious or misbehaving drones before they cause considerable damage to the network. If a drone has been classified as misbehaving or malicious, let the drone's pseudonym be $\mathrm{PID}_k^i = \{L_k^i, \mathrm{DID}_k^i, T_i^{\mathrm{res}}\}$. Revocation is done as follows.

1. A network entity complains about a malicious drone to *CoA* using its pseudonym PID_k^i.
2. *CoA* searches for the pseudonym batch from its local memory and adds it to the revocation list.

3. If a drone's pseudo-identity is revoked for the third time, the corresponding *GCS* is notified about the revocation. The *GCS* changes the validity of the corresponding drone's license to *false*. If the drone wants to communicate again, it must undergo the registration process once again from the beginning.

The *GCS* broadcasts the revocation list whenever there is an update in the list. Base stations also help the *GCS* in this broadcast by sharing the load. Thus the devices are mostly aware of the malicious drones, thus maintaining safe communication with authorized devices. Additionally, session keys that remain inactive for a predefined timeout period are invalidated automatically. This ensures that inactive or compromised session keys do not pose a security risk.

5 Security analysis

This section analyzes the security of the proposed scheme. First a formal analysis will be done using BAN logic [36], to show that our proposed scheme is logically correct. The proposed scheme is also analyzed using Scyther tool to check for any attacks present. Also, an informal security analysis concerning the security goals mentioned in Sect. 3.4 is also done.

5.1 Formal analysis

Following are the different notations used in BAN Logic:

- $P \mid\equiv X$: Principal P believes statement X.
- $P \lhd X$: Principal P sees statement X.
- $P \mid\sim X$: Principal P once said X.
- $P \mid \Longrightarrow X$: Principal P has jurisdiction over statement X.
- $\#(X)$: Formula X is fresh.
- $\{X\}_K$: Formula X is encrypted under key K.
- $P \overset{K}{\leftrightarrow} Q$: Principal P and principal Q may use shared key K to communicate. The key K will be known only to principals P and Q.
- $\overset{K}{\mapsto} P$: Principal P has a public key K. Its corresponding private key (K^{-1}) is only known to the principal P or a principal trusted by P.

The main logic postulates used in the proofs of BAN logic are as follows:

- Message meaning rule : $\frac{P\mid\equiv Q\overset{K}{\leftrightarrow}P,\ P\lhd\{X\}_{K^{-1}}}{P\mid\equiv Q\mid\sim X}$
- Nonce verification rule : $\frac{P\mid\equiv\#(X),\ P\mid\equiv Q\mid\sim X}{P\mid\equiv Q\mid\equiv X}$
- Jurisdiction rule : $\frac{P\mid\equiv Q\mid \Longrightarrow X,\ P\mid\equiv Q\mid\equiv X}{P\mid\equiv X}$
- Freshness rule: $\frac{P\mid\equiv\#(X)}{P\mid\equiv\#(X,Y)}$

As a first step to prove the security of the proposed scheme, the messages of the scheme are transformed into idealized form as follows:

- M1: $\mathrm{dr}_j \lhd \{D_{\mathrm{id}}, \mathrm{PID}_k^i, M, T_i^{\mathrm{auth}}, y\}_{\mathrm{dr}_{\mathrm{sk}_i}}$
- M2: $\mathrm{dr}_j \lhd \{\mathrm{nonce}, \mathrm{dr}_i \overset{\mathrm{ssk}_{ij}}{\leftrightarrow} \mathrm{dr}_j\}_{\mathrm{ssk}_{ij}}$
- M3: $\mathrm{dr}_i \lhd \{\mathrm{nonce}+1, \mathrm{dr}_i \overset{\mathrm{ssk}_{ij}}{\leftrightarrow} \mathrm{dr}_j\}_{\mathrm{ssk}_{ij}}$

The following are the initial assumptions about the proposed scheme:

- A1: $\mathrm{dr}_j \mid\equiv \mathrm{dr}_i \mid \Longrightarrow M$
- A2: $\mathrm{dr}_j \mid\equiv \#\left(T_i^{\mathrm{auth}}\right)$
- A3: $\mathrm{dr}_j \mid\equiv \overset{D_{\mathrm{pk}_i}}{\mapsto} \mathrm{dr}_i$
- A4: $\mathrm{dr}_i/\mathrm{dr}_j \mid\equiv \mathrm{dr}_i \overset{\mathrm{ssk}_{ij}}{\leftrightarrow} \mathrm{dr}_j$
- A5: $\mathrm{dr}_i/\mathrm{dr}_j \mid\equiv \#\left(\mathrm{dr}_i \overset{\mathrm{ssk}_{ij}}{\leftrightarrow} \mathrm{dr}_j\right)$

To prove that the proposed scheme is secure with respect to the message authentication and session key generation methods described here, the following goals are to be achieved:

- G1: $\mathrm{dr}_j \mid\equiv M$
- G2: $\mathrm{dr}_i \mid\equiv \mathrm{dr}_j \mid\equiv \mathrm{dr}_i \overset{\mathrm{ssk}_{ij}}{\leftrightarrow} \mathrm{dr}_j$
- G3: $\mathrm{dr}_j \mid\equiv \mathrm{dr}_i \mid\equiv \mathrm{dr}_i \overset{\mathrm{ssk}_{ij}}{\leftrightarrow} \mathrm{dr}_j$

Now the idealized messages of the scheme and assumptions are used with BAN logic rules to give the proofs in order to achieve the goals mentioned above.

Using M1, A3 and message meaning rule, we get
S1: $\mathrm{dr}_j \mid\equiv \mathrm{dr}_i \mid\sim \{D_{\mathrm{id}}, \mathrm{PID}_k^i, M, T_i^{\mathrm{auth}}, y\}$
Using S1, A2 and nonce verification rule, we get
S2: $\mathrm{dr}_j \mid\equiv \mathrm{dr}_i \mid\equiv \{D_{\mathrm{id}}, \mathrm{PID}_k^i, M, T_i^{\mathrm{auth}}, y\}$
Using S2, A1 and jurisdiction rule, we get
S3: $\mathrm{dr}_j \mid\equiv M$ (**G1 satisfied**)
Using M2, A4 and message meaning rule, we get
S4: $\mathrm{dr}_j \mid\equiv \mathrm{dr}_i \mid\sim \mathrm{dr}_i \overset{\mathrm{ssk}_{ij}}{\leftrightarrow} \mathrm{dr}_j$
Using S4, A5 and nonce verification rule, we get
S5: $\mathrm{dr}_i \mid\equiv \mathrm{dr}_j \mid\equiv \mathrm{dr}_i \overset{\mathrm{ssk}_{ij}}{\leftrightarrow} \mathrm{dr}_j$ (**G2 satisfied**)
G3 can also be proved in a similar way as G2.
Thus, logical correctness of the proposed scheme is proved using BAN logic.

5.2 Scyther tool analysis

Scyther is an automatic security protocol verification tool developed by CISPA Helmholtz Center for Information Security in Saarbruecken, Germany [37]. The formal analysis in this tool is done under the assumption that all the cryptographic functions used in the corresponding protocol are perfect. This logical tool can be used to prove whether the protocol is logically correct and safe or if there are attacks. The input for this tool is given in the form of Security Protocol Description Language (.spdl) using which the steps in the protocol are described and the security claims for the protocol are specified for the tool to verify the same. Figure 6 shows the spdl code for the proposed scheme, and Fig. 7 shows the output for the same.

Fig. 6 Scyther spdl code

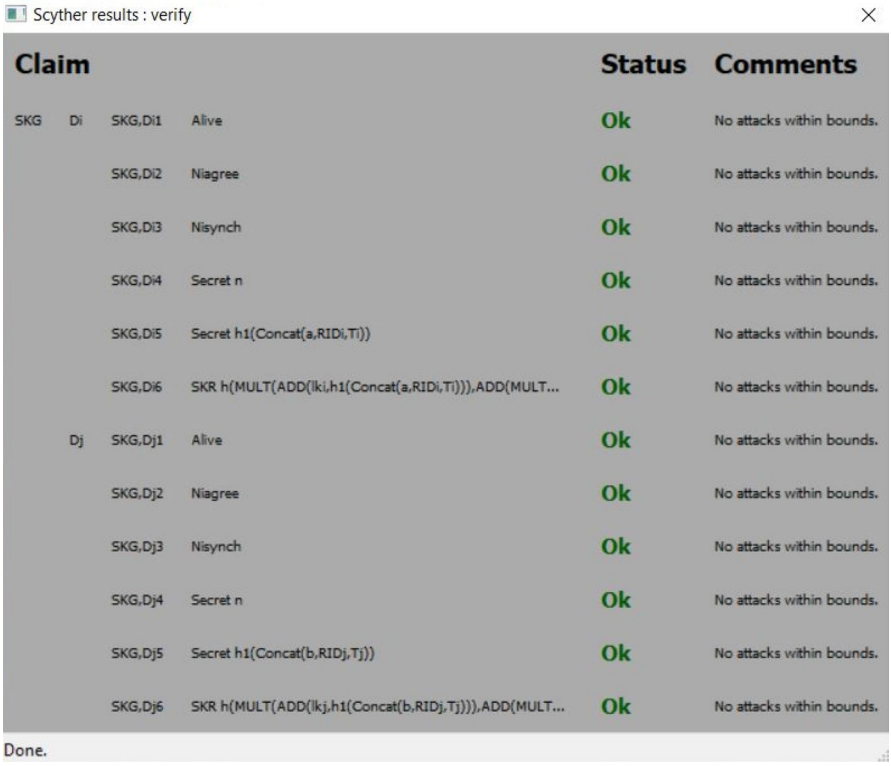


Fig. 7 Scyther Analysis Result

5.3 Informal analysis

Confidentiality: The communication between the drones is encrypted using the session keys generated as mentioned in Sect. 4.5, thus ensuring confidentiality.

Decentralization: The drone's license is uploaded to the blockchain, and the drones need not be dependent on any single authority to query the blockchain. Thus the proposed scheme will form a decentralized network.

Mutual Authentication: During the session key generation, as mentioned in Sect. 4.5, the public key of the drone is obtained from the drone's license queried from the blockchain. Using this, the drones verify each other's identity, ensuring mutual authentication.

Identity Protection: A hierarchical pseudonym generation procedure is discussed in this scheme, and this pseudonym is used for authentication instead of real identity. To retrieve the real identity of the drone from the pseudonym, one must know the secret key generated by the control authority and the secret key of the corresponding

GCS of the domain to which the drone belongs. This procedure ensures identity protection.

Traceability: If any necessity arises to retrieve the original identity of any drone, the secret key *GCS* and secret key generated by *CoA* are required, needing both the authority's cooperation. Once both authorities cooperate, the real identity of the drone can be obtained by the equation mentioned below.

$$\mathrm{RID} = \mathrm{DID}_k^i \oplus h_2\left(g_{\mathrm{sk}} \cdot T_{\mathrm{reg}}\right) \oplus h_2\left(e_k^i \cdot T_i^{\mathrm{res}}\right) \tag{14}$$

Batch-Pseudonym Generation and Revocation: As discussed in Sect. 4.3, *CoA* generates the batch pseudonym and distributes it to the drone securely. Also, as discussed in Sect. 4.6, if any malicious activity is reported, *CoA* will take action by verifying and revoking the batch of pseudonyms generated for that particular drone.

Resistance against Common Security Attacks: The proposed scheme is secure against common security attacks as explained below:

1. *Identity Forgery:* The identity of the drone goes through hierarchical pseudonym generation, which includes two random timestamps. Also, the hash of the timestamp and the secret key of the concerned authorities are used in pseudonym generation. Even if anybody gets their hand on the public keys, finding the secret key comes under the ECDLP hard problem discussed in Sect. 3.1.
2. *Impersonation:* During the session key generation, a random *nonce* is generated and sent to the other drone. Only a legitimate drone can complete this session key generation by decrypting the *nonce*. Thus impersonation attack is avoided.
3. *Replay Attack:* Every session key involves the usage of a different pseudonym material (l_i^k) from a batch of pseudonyms. Thus even if anyone gets their hand on old keys, it is of no use.
4. *DDoS Attack:* Blockchain, by default, is resistant to DDoS attacks as it is a peer-to-peer distributed network. As the network is decentralized, as proven above, devices can access the blockchain through other nodes present, even if one node is under attack.

The proposed scheme is compared with Feng et al. [7], Yang et al. [13], Tan et al. [16], Shen et al. [35], which are some recent blockchain-based authentication schemes. Table 2 compares the security goals with these authentication schemes to prove the security of the proposed scheme.

6 Performance analysis

This section discusses the performance of the proposed scheme concerning the computation and communication costs. It also discusses the simulation environment in which the blockchain network is executed and evaluated.

Table 2 Comparison of Security Goals

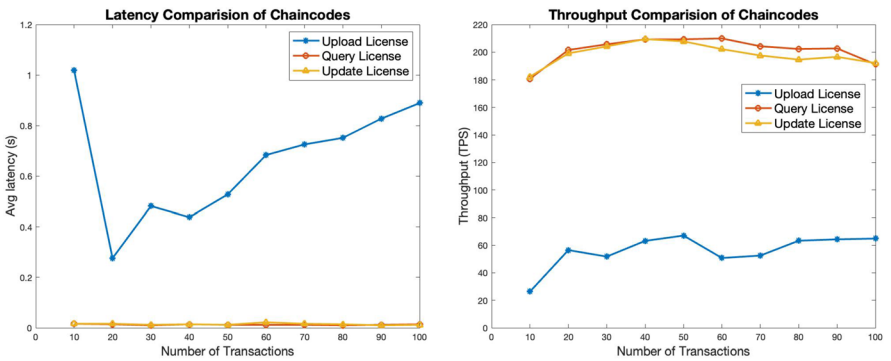
Security Goals	[7]	[13]	[16]	[35]	Ours
Confidentiality	Yes	Yes	Yes	No	Yes
Mutual Authentication	Yes	No	Yes	Yes	Yes
Decentralization	Yes	Yes	Yes	Yes	Yes
Privacy-Preserving	Yes	Yes	Yes	Yes	Yes
Traceability	Yes	Yes	Yes	Yes	Yes
Revocation	No	Yes	Yes	No	Yes
Batch-Pseudonym Generation	No	Yes	No	No	Yes
Key Agreement	Yes	No	Yes	No	Yes
Resistance to Common Attacks	Yes	Yes	Yes	Yes	Yes

6.1 Simulation setup

The experiments and simulations are conducted using an Apple MacBook Air with Apple M1 chip CPU@3.2 GHz and 8 GB RAM running on macOS 12.6 (Monterey). For the blockchain simulations, Hyperledger Fabric Version 2.4.3 is used with a Docker Desktop Version 4.12.0. to analyze the blockchain, Hyperledger Caliper Version 0.5.0 is used (Fig. 8).

6.2 Blockchain performance analysis

This work considers a Hyperledger Fabric model with three organizations, each with a single peer. One organization represents *CoA*, and the other two represent two domains with a *GCS* each. Chaincodes are installed in all three peers according to their roles. The simulation setup is shown in Fig. 9. Hyperledger Caliper is used to analyze the installed chaincode. The benchmark specification file showing one round of analysis is shown in Listing 1.



(a) Blockchain Latency Analysis (b) Blockchain Throughput Analysis

Fig. 8 Hyperledger Caliper Analysis

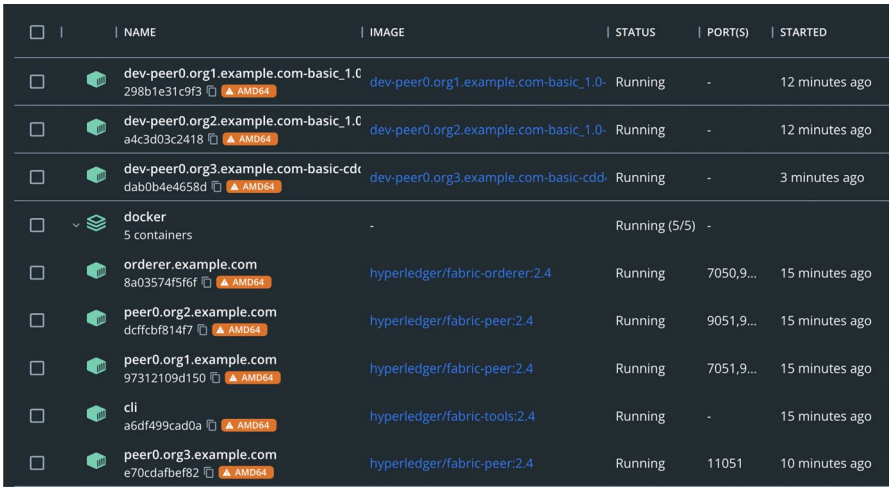


Fig. 9 Blockchain Simulation Docker Desktop

```
test:
    name: Drone Authentication Blockchain Analysis
    description: txnumber [10 to 100] in fixed rate [tps 300]
    workers:
      type: local
      number: 1
    rounds:
    - label: uploadLicense
      description: Upload License benchmark
      txNumber: 10
      rateControl:
        type: fixed-rate
        opts:
          tps: 300
      workload:
```

Listing 1 Caliper Benchmark Specification The network is tested for uploading licenses, querying license and updating license operations. The execution time of each operation is calculated by taking the average of 10 consecutive runs. For each of these operations, the number of transactions (t × Number) is varied from 10 to 100 in a step of 10. Fixed rate is used as the rate controller with a transactions per second (tps) value of 300. The results are recorded and plotted in a graph as shown in Fig. 8a, b. The graph shows that uploading the license is costly in the blockchain compared to other operations. Though the latency for other smart contacts is negligible and the throughput is as desired, upload license smart contract has an average latency of 0.7 s and an average throughput of 08 TPS. This operation is a little heavy but a very important step in securing this cross-domain authentication scheme. However, the graph for uploading licenses only increases linearly and not exponentially.

6.3 Computation cost analysis

For cryptographic analysis, two libraries are used: pairing-based cryptography (pbc) version 0.5.14 for pairing-based operations and Crypto++ library version 8.7 for other cryptographic operations, including ECC. A non-singular elliptic curve $E : y^2 = x^3 + ax + b \pmod{p}$ is considered for ECC operations where p is a prime number and $a, b \in Z_p^*$. G is an additive group on the curve E with a generator P and an order q. For bilinear pairing operations, the pbc library is used. The benchmark programs are run using the parameters of a Type A curve provided in the library files, and the values are noted. The running times of cryptographic operations are listed in Table 3. Normal operations like addition, multiplication and other operations whose values are too small to be considered are ignored. The computational analysis is performed using the time cost of various operations as mentioned in Table 3. Tables 4, 5 and Fig. 10 present the overall computational analysis. For

Table 3 Time Cost for various Cryptographic Operations

Symbol	Description	Time
T_{bp}	Bilinear Pairing	9.9 ms
T_{htp}	Hash to Point Mapping	0.0008 ms
T_{exp}	Modular Exponentiation on $\mathbb{G}_T$	0.871 ms
$T_{\text{pa}}^{\text{ecc}}$	Point Addition in ECC	0.012 ms
$T_{\text{sm}}^{\text{ecc}}$	Scalar Multiplication in ECC	0.566 ms
T_h	General hash Function	0.0003 ms
T_{enc}	AES Encryption	0.035 ms
T_{dec}	AES Decryption	0.017 ms

Table 4 Comparison of Computation Overhead (SG & SV)

Schemes	SG	SV
Feng et al. [7]	$2T_{\text{enc}} + 1T_{\text{dec}} + 1T_h \simeq 0.087$ ms	$1T_{\text{bp}} + 2T_{\text{dec}} + 1T_{\text{enc}} + 1T_h \simeq 9.969$ ms
Yang et al. [13]	$1T_h \simeq 0.0003$ ms	$3T_{\text{sm}}^{\text{ecc}} + 4T_{\text{pa}}^{\text{ecc}} + 2T_h \simeq 1.747$ ms
Tan et al. [16]	$1T_h \simeq 0.0003$ ms	$2T_{\text{sm}}^{\text{ecc}} + 1T_{\text{pa}}^{\text{ecc}} + 1T_h \simeq 1.114$ ms
Shen et al. [35]	$1T_{\text{bp}} + 1T_{\text{exp}} + 1T_h \simeq 10.771$ ms	$2T_{\text{bp}} + 1T_{\text{exp}} + 2T_h \simeq 20.672$ ms
Ours	$1T_h \simeq 0.0003$ ms	$2T_{\text{sm}}^{\text{ecc}} + 1T_{\text{pa}}^{\text{ecc}} + 1T_h \simeq 1.114$ ms

Table 5 Comparison of Computation Overhead (SKG)

Schemes	SKG
Feng et al. [7]	$2(1T_{\text{enc}} + 1T_{\text{dec}} + 1T_h) \simeq 0.105$ ms
Tan et al. [16]	$2(1T_{\text{pa}}^{\text{ecc}} + 1T_{\text{sm}}^{\text{ecc}} + 1T_h) + 2T_{\text{enc}} + 1T_{\text{dec}} \simeq 0.112$ ms
Shen et al. [35]	$2(1T_{\text{bp}} + 2T_{\text{exp}}) \simeq 23.284$ ms
Ours	$2(1T_{\text{pa}}^{\text{ecc}} + 1T_h + 1T_{\text{enc}} + 1T_{\text{dec}}) \simeq 0.129$ ms

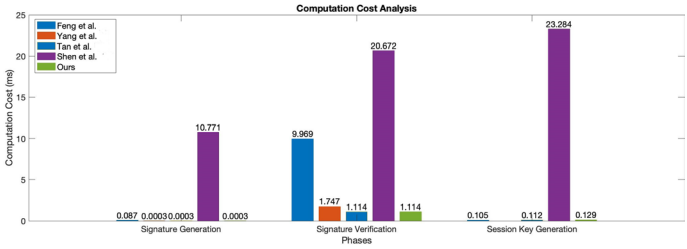


Fig. 10 Computation Cost Analysis

clarity and easy comparison, the whole scheme is divided into three parts: (1) signature generation (SG), (2) signature verification (SV) and (3) session key generation (SKG). It can be seen that the proposed scheme outperforms most of the compared schemes. Though Tan et al.'s [16] work seems equal to or more lightweight than the proposed scheme, they have some drawbacks. In the proposed scheme, batch pseudonyms are generated and distributed one batch at a time to drones. In this way, the drone need not depend on any other authority for a one-time key for some time and need not upload a one-time key constantly in the blockchain for record purposes. But in Tan et al. the drone generates a one-time key each time it wants to communicate with another drone and constantly records the one-time key in the blockchain. For a stand-alone IoT device, frequently accessing blockchain is a huge drawback, which is not included in the computational analysis.

6.4 Communication cost analysis

In this section, we compare the communication cost of the proposed scheme with other similar schemes. Here, we consider the length of IDs and timestamps to be 4 bytes. The length of a general hash output will be 32 bytes. The length of an element in $\mathbb{G}$ is 64 bytes and an element in $\mathbb{Z}_q^*$ is 20 bytes. AES encryption output is of length 256 bits. In the proposed scheme, for signature generation, M_Auth is sent to the other drone, which has the contents with length $(4+100+S_m+4+32) = 140 + S_m$ bytes. During the signature verification, the license of 136 bytes is queried from the blockchain. The session key generation has an AES encryption communicated to and fro between the drones with a pseudonym material and a *nonce*. So that gives 136 bytes. This comparison shows that the proposed scheme is more efficient than most related schemes. Though lesser communication cost is essential,

Table 6 Comparison of Communication Overhead

Schemes	SG	SV	SKG
Feng et al. [7]	496 bytes	400 bytes	320 bytes
Yang et al. [13]	$(236+S_m)$ bytes	204 bytes	–
Tan et al. [16]	$(56+S_m)$ bytes	$256+2l$ bytes	160 bytes
Shen et al. [35]	$(612+2*S_m)$ bytes	$(300+S_m)$ bytes	332 bytes
Ours	$(140+S_m)$ bytes	136 bytes	136 bytes

from the security analysis, we prove that our scheme is also secure while performing efficiently. The communication cost comparison with other similar authentication schemes is given in Table 6.

6.5 Discussion

The proposed blockchain-based cross-domain authentication scheme significantly enhances the security and efficiency of drone communication in the Internet of drones (IoD) environments. By leveraging elliptic curve cryptography (ECC) for lightweight and secure signature generation and verification, and implementing a hierarchical batch-pseudonym generation and revocation procedure, our scheme ensures conditional privacy and efficient identity management. The integration of Hyperledger Fabric as the blockchain framework provides a tamper-proof ledger that bolsters security against various attacks, including impersonation and replay attacks. Our performance analysis using Hyperledger Caliper demonstrates that the scheme achieves improved transaction throughput and reduced latency compared to existing methods. Additionally, the implementation of a session key timeout mechanism further strengthens security by invalidating inactive keys, mitigating risks associated with prolonged key usage. Overall, the proposed scheme offers a robust solution for secure and efficient drone communication, with potential applications in various fields such as public safety, emergency response and commercial drone operations.

7 Conclusion

In this work, we designed a Blockchain-based Cross-domain Authentication Scheme for the IoD scenario. The IoD environment's distributed feature and the blockchain's peer-to-peer nature are matched, and the scheme is modeled to take full advantage of the blockchain's features. An elliptic curve cryptography scheme is designed with a hierarchical pseudonym generation procedure. The confidentiality and integrity of the drone's details are maintained by recording it on blockchain and allowing only authorized parties to access it. A revocation procedure in case of malicious drone reporting is also discussed. The implementation is done on Hyperledger Fabric, and the blockchain analysis uses Hyperledger Caliper. The security analysis confirms the effective security of the proposed scheme. The computation and communication cost analysis show that the proposed scheme is more efficient than most recent similar authentication schemes. Despite these advantages, there are some limitations to our proposed scheme. The performance of our scheme might degrade in scenarios with extremely high drone mobility and dense network environments, where the frequent updating of pseudonyms and session keys could introduce some overhead. Furthermore, the scalability of the blockchain network could be constrained by the inherent limitations of Hyperledger Fabric itself. But, as recently Hyperledger Fabric has released v3.0.0-beta which replaces the raft consensus with Byzantine fault tolerant ordering service based on [38], it would be a major advantage in terms of security. In the future, we will work on a hierarchical blockchain model, to reduce the authentication latency for drones from the same domain. Also, the

pseudonym generation process will be further decentralized, exploring more scalable blockchain frameworks to enhance the system's robustness and efficiency.

Author Contributions Conceptualization, methodology and writing—original draft preparation were performed by AK; formal analysis and investigation were performed by AK and AT; writing—review and editing was performed by ST and AT; supervision was performed by ST.

Funding This declaration is "not applicable."

Data availability This declaration is "not applicable."

Declarations

Conflict of interest On behalf of all authors, the corresponding authors declare that they have no known conflict of interest that are directly or indirectly related to the work submitted for publication.

Ethical Approval This declaration is "not applicable."

References

1. Kumar S, Tiwari P, Zymbler M (2019) Internet of things is a revolutionary approach for future technology enhancement: a review. J Big data 6(1):1–21
2. Bera B, Das AK, Sutrala AK (2021) Private blockchain-based access control mechanism for unauthorized UAV detection and mitigation in internet of drones environment. Comput Commun 166:91–109
3. Mozaffari M, Saad W, Bennis M, Nam Y-H, Debbah M (2019) A tutorial on UAVS for wireless networks: applications, challenges, and open problems. IEEE Commun Surv Tutor 21(3):2334–2360
4. Chen Y, Chen J (2021) A secure three-factor-based authentication with key agreement protocol for e-health clouds. J Supercomput 77:3359–3380
5. Gharibi M, Boutaba R, Waslander SL (2016) Internet of drones. IEEE Access 4:1148–1162
6. Wazid M, Das AK, Kumar N, Vasilakos AV (2019) Design of secure key management and user authentication scheme for fog computing services. Future Gen Comput Syst 91:475–492
7. Feng C, Liu B, Guo Z, Yu K, Qin Z, Choo K-KR (2021) Blockchain-based cross-domain authentication for intelligent 5g-enabled internet of drones. IEEE Internet Things J 9(8):6224–6238
8. Shen M, Liu H, Zhu L, Xu K, Yu H, Du X, Guizani M (2020) Blockchain-assisted secure device authentication for cross-domain industrial IoT. IEEE J Sel Areas Commun 38(5):942–954
9. Jain S, Doriya R (2022) Security framework to healthcare robots for secure sharing of healthcare data from cloud. Int J Inf Technol 14(5):2429–2439
10. Nakamoto S (2008) Bitcoin: a peer-to-peer electronic cash system. Decentralized business review
11. Abbas S, Talib MA, Ahmed A, Khan F, Ahmad S, Kim D-H (2021) Blockchain-based authentication in internet of vehicles: a survey. Sensors 21(23):7927
12. Jain S, Nandhini C, Doriya R (2021) ECC-based authentication scheme for cloud-based robots. Wirel Pers Commun 117(2):1557–1576
13. Yang Y, Wei L, Wu J, Long C, Li B (2021) A blockchain-based multidomain authentication scheme for conditional privacy preserving in vehicular ad-hoc network. IEEE Internet Things J 9(11):8078–8090
14. Cheng G, Chen Y, Deng S, Gao H, Yin J (2021) A blockchain-based mutual authentication scheme for collaborative edge computing. IEEE Trans Comput Soc Syst 9(1):146–158
15. Xu G, Dong J, Ma C, Liu J, Cliff UGO (2022) A certificateless signcryption mechanism based on blockchain for edge computing. IEEE Internet Things J 10(14):11960–11974
16. Tan Y, Wang J, Liu J, Kato N (2022) Blockchain-assisted distributed and lightweight authentication service for industrial unmanned aerial vehicles. IEEE Internet Things J 9(18):16928–16940
17. Wang W, Xu H, Alazab M, Gadekallu TR, Han Z, Su C (2021) Blockchain-based reliable and efficient certificateless signature for IIOT devices. IEEE Trans Ind Inf 18(10):7059–7067

18. Al-Riyami SS, Paterson KG (2003) Certificateless public key cryptography. In: International Conference on the Theory and Application of Cryptology and Information Security. Springer: New York, pp 452–473
19. Zheng Y (1997) Digital signcryption or how to achieve cost (signature & encryption)<< cost (signature)+ cost (encryption). In: Advances in Cryptology-CRYPTO'97: 17th Annual International Cryptology Conference Santa Barbara, California, USA August 17–21, 1997 Proceedings, Springer: New York, pp 165–179
20. Zhang L, Xu J (2022) Blockchain-based anonymous authentication for traffic reporting in vanets. Connect Sci 34(1):1038–1065
21. Koblitz N (1987) Elliptic curve cryptosystems. Math Comput 48(177):203–209
22. Miller VS (1985) Use of elliptic curves in cryptography. In: Conference on the Theory and Application of Cryptographic Techniques. Springer: New York, pp 417–426
23. Yeh K-H, Tsai K-Y, Fan C-Y (2015) An efficient certificateless signature scheme without bilinear pairings. Multimed Tools Appl 74:6519–6530
24. Wang L, Chen K, Long Y, Mao X, Wang H (2015) A modified efficient certificateless signature scheme without bilinear pairings. In: 2015 International Conference on Intelligent Networking and Collaborative Systems. IEEE, pp 82–85
25. Zhang S, Yan Z, Liang W, Li K-C, Di Martino B (2024) BCAE: a blockchain-based cross domain authentication scheme for edge computing. IEEE Internet Things J
26. Yasuda M, Shimoyama T, Kogure J, Izu T (2016) Computational hardness of IFP and ECDLP. Appl Algebra Eng Commun Comput 27:493–521
27. Maurer U, Wolf S (1998) Diffie-Hellman, decision Diffie-Hellman, and discrete logarithms. In: Proceedings. 1998 IEEE International Symposium on Information Theory (Cat. No. 98CH36252). IEEE, p 327
28. Cui J, Zhu Y, Zhong H, Zhang Q, Gu C, He D (2024) Efficient blockchain-based mutual authentication and session key agreement for cross-domain IIOT. IEEE Internet Things J
29. Luo D, Cai Q, Sun G, Yu H, Niyato D (2024) Split-chain based efficient blockchain-assisted cross-domain authentication for IoT. IEEE Trans Netw Serv Manag
30. Chen Y, Zhang J, Wei X, Wang, Y, et al (2024) Cross-domain authentication scheme for vehicles based on given virtual identities. IEEE Internet Things J
31. Kapoor V, Abraham VS, Singh R (2008) Elliptic curve cryptography. Ubiquity 2008(May):1–8
32. Androulaki E, Barger A, Bortnikov V, Cachin C, Christidis K, De Caro A, Enyeart D, Ferris C, Laventman G, Manevich Y et al (2018) Hyperledger fabric: a distributed operating system for permissioned blockchains. In: Proceedings of the Thirteenth EuroSys Conference, pp 1–15
33. Panda SS, Jena D, Mohanta BK, Ramasubbareddy S, Daneshmand M, Gandomi AH (2021) Authentication and key management in distributed IoT using blockchain technology. IEEE Internet Things J 8(16):12947–12954
34. Lamport L (1981) Password authentication with insecure communication. Commun ACM 24(11):770–772
35. Shen M, Liu H, Zhu L, Xu K, Yu H, Du X, Guizani M (2020) Blockchain-assisted secure device authentication for cross-domain industrial IoT. IEEE J Sel Areas Commun 38(5):942–954
36. Burrows M, Abadi M, Needham R (1990) A logic of authentication. ACM Trans Comput Syst (TOCS) 8(1):18–36
37. Mauw S, Cremers C (2012) Operational Semantics and Verification of Security Protocols. Information Security and Cryptography. Springer, New York
38. Barger A, Manevich Y, Meir H, Tock Y (2021) A Byzantine Fault-Tolerant Consensus Library for Hyperledger Fabric. https://arxiv.org/abs/2107.06922

Authors and Affiliations

Arivarasan Karmegam[1] · Ashish Tomar[2] · Sachin Tripathi[3]

✉ Ashish Tomar
ashishtomar244@gmail.com

Arivarasan Karmegam
arivu.osr@gmail.com

Sachin Tripathi
sachin2781@iitism.ac.in

[1] IMDEA Networks Institute, 28918 Madrid, Spain

[2] School of Computer Science Engineering and Technology, Bennett University, Greater Noida, Uttar Pradesh 201310, India

[3] Department of Computer Science and Engineering, Indian Institute of Technology (Indian School of Mines), Dhanbad, Jharkhand 826004, India