Computer Networks 195 (2021) 108219

Contents lists available at ScienceDirect

Computer Networks

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

A secure blockchain-oriented data delivery and collection scheme for 5G-enabled IoD environment

Azeem Irshad [a], Shehzad Ashraf Chaudhry [b], Anwar Ghani [a], Muhammad Bilal [c,*]

[a] *Department of computer science and software engineering, International Islamic University Islamabad, Pakistan*
[b] *Department of Computer Engineering, Faculty of Engineering and Architecture, Istanbul Gelisim University, Istanbul, Turkey*
[c] *Department of Computer Engineering, Hankuk University of Foreign Studies, Yongin-si, South Korea*

A R T I C L E I N F O

Index Terms:
Blockchain
Internet of drones
Security
Internet of Things
Data collection and delivery

A B S T R A C T

– There are innumerable ways the Internet of Drones (IoD) technology can impact our society. With the deployment of an airborne network, the IoD can support real-time low-cost delivery of services ranging from military surveillance to a myriad of civilian applications. Nevertheless, the drones employ insecure wireless communication channels to communicate with other entities in the system, inhibiting its induction in sensitive installations if insecure or inefficient Authenticated Key Agreement (AKA) schemes are employed. The blockchain, an open distributed ledger-based technology, is increasingly being adopted to address the security concern as discussed. Recently, Bera et al. presented an efficient blockchain-enabled AKA scheme for data management among various entities in IoD network. However, their scheme does not support anonymity and untraceability for the drones; also, it does not provide resistance to Ground station server impersonation attack, while the protocol has a few redundancies. Later, we proposed an enhanced blockchain-enabled AKA scheme BOD5-IOD to authenticate drones in the system. The BOD5-IOD, other than supporting a robust access control mechanism between drones and GSS, also ensures safe transactions among all entities in the IoD environment. The formal analysis and performance evaluation endorse that our scheme supports security requirements with computational and communication efficiency of 34.4% and 23.3%, respectively.

1. Introduction

The Internet of Drones (IoD) has nearly paved its way into every segment of society ranging from recreational to commercial and military applications. Alternatively, the UAVs have exhibited their promising capabilities in supporting numerous applications such as military surveillance, rescue, delivery, photography, agriculture, wildlife monitoring, traffic monitoring, etc. Following a recent Federal Aviation Administration (FAA) survey, the number of small-scale commercial or model-based drones or UAVs may grow as much as 7 million by the end of 2020 [1]. These drones may communicate data using wireless channels after monitoring it through sensors but also perform high-tech operations with the help of remote monitoring and intelligence. Moreover, it can also deliver lightweight packages to the target destination, depending on its application. Whatever be the application, i.e., data transfer, remote monitoring, sensing, operation or delivering the lightweight assignment, etc., the control data or communication between drone and control room/ground station server is always vulnerable to several security risks and threats [2-3].

The small-scale UAVs are equipped with several Internet of Things (IoT)-based smart devices such as sensors and actuators which are being used for sensing and collecting the captured data from a targeted spot towards any destination. In this connection, the drones need to quickly transfer live streaming video data that must be complemented with low latency and high bandwidth connection. The 5G connections may contribute to making such an IoD ecosystem viable [4-6]. The drones may be employed in many 5G-enabled use cases, including smart city, remote industrial control applications, smart agriculture, and many other scenarios.

The first generation (1G) of mobile communication was introduced in 1980; however, it was insecure, with poor battery support and voice quality. It was followed by second-generation (2G) in 1990 as called Global System for Mobile communication (GSM), having digital capabilities. However, due to the mobility problems and lower data transmission rates in 2G, the third generation (3G) technology was introduced in 2001, which supports multimedia messages, tracking, and

---

[*] Corresponding author.
E-mail address: m.bilal@ieee.org (M. Bilal).

https://doi.org/10.1016/j.comnet.2021.108219
Received 12 December 2020; Received in revised form 3 May 2021; Accepted 31 May 2021

augmented security [7]. Nonetheless, another fourth-generation (4G) was developed with the support of voice over LTE (VoLTE), higher data rates, and HD streaming due to the infrastructure issues and expensive gadgets. The 5G technology is introduced in 2020 for supporting ultra-fast Internet with higher bandwidth and reliability [8]. The 5G-oriented blockchain technology-based framework involves drones, ground station servers, control rooms, registration authority, and blockchain center. The 5G cellular technology may assist in three ways to connect the UAVs. 1) Administering the traffic of UAVs, 2) Beyond Visual Line of Sight (BVLOS)-based flights [9], 3) Transmission of data based on sensors. The Unmanned Aircraft System Traffic Management (UTM) regulates the traffic of drones and manned aviation and helps the drones integrate in routine air traffic. Similarly, the BVLOS technology can assist the drones in covering long distances comparatively.

For secure data delivery and collection, many authenticated key agreement schemes [4,6-9,13,23-25,30-31] have been designed to ensure the secure communication of data; however, those schemes were prone to many security drawbacks. Another efficient blockchain-enabled AKA scheme by Bera et al. [13] for data management among various entities in IoD network has been presented. However, it is witnessed that their scheme does not ensure anonymity as well as untraceability for the drones. Furthermore, it does not provide immunity from ground-station server impersonation threat, and at the same time, Bera et al.'s protocol [13] has a few redundancies. Consequently, we proposed an enhanced blockchain-enabled AKA scheme BOD5-IOD to authenticate drones in the system. The BOD5-IOD, other than supporting a robust access control mechanism between drones and GSS, also warrants safe transactions among all entities in the IoD environment. The formal analysis and performance evaluation approve that our scheme (BOD5-IOD) supports enhanced security requirements with optimal computational and communicational delays.

1.1. Threat model

Being on the insecure wireless communication channel, the IoD provides ample opportunities to the attacker to initiate forgery attacks against drones or GSS. A widely used threat model by Dolev-Yao (DY model) [10] is assumed to evaluate the security of the proposed scheme. In DY model, an adversary may intercept, edit, block, replay or delete the communication messages in transit, and initiate many launch forgery attacks. In this connection, a de facto CK-adversary model [11] is also assumed for analyzing the security, since the adversary is more potent under this model with the capability to compromise the long-term credentials, random secrets, and session keys. This affirms that the agreed session key between UAVs and GSS entities must be composed of short-term random secrets along with long-term credentials to avoid the ephemeral information and forward secrecy attacks. Such attacks may be defeated with the use of long-term as well as short-term secrets in the session key.

1.2. Research contributions

The salient points of the contribution are as follows:

1 We highlight the significance of secure transmission and receipt of data in a 5G-oriented IoD ecosystem.
2 We propose an enhanced and secure blockchain-oriented Data Delivery and Collection (DDC) scheme as titled BOD5-IOD that permits the authenticated key agreement (AKA) between UAVs and corresponding GSS in every flying zone FZ_j. On the basis of the suggested AKA procedure, the mutually agreed session keys among UAVs and GSSs can be established to communicate safely. The DDC process in BOD5-IOD permits recording all of the associated transactions among UAVs, GSS, and CR in order to generate private blocks with the help of GSS.

Table 1
Tabular depiction of most recent literature.

Scheme	Features	Drawbacks	Year
Jangirala et al. [7]	Blockchain-based RFID authentication scheme for IoD	Secret disclosure attack and traceability problems	2019
Srinivas et al. [8]	Temporal credential-based AKA scheme for IoD	Mutual authentication and privacy issues for drones	2019
SDPC [31]	Authentication scheme for secure content distribution for in-network caches	Lack of support of high mobility	2020
Cho et al. [30]	Authentication scheme for UAVs	Susceptible to ephemeral secret leakage attack	2020
Mandal et al. [6]	Certificateless-Signcryption based Three-Factor AKA for IoT Environment	Inefficient due to more communication overhead of sensors	2020
Yazdinejad et al. [9]	Decentralized blockchain-based AKA scheme for IoD	Complex management of distributed drone controllers and key distribution	2020
Bera et al. [13]	Blockchain-oriented secure data transmission and collection	Lacking mutual authentication and traceability problems	2020

3 Considering the limitations in previous research studies as shown in Table 1, we design a blockchain-based consensus algorithm to verify and append the blocks through a selected leader in multiple GSSs in the blockchain-oriented peer-to-peer network.
4 We employed a MIRACL library, a widely recognized collection of cryptographic primitives, for computing the execution time on the Raspberry PI 3 B+ and server platform.
5 Lastly, the performance analysis for BOD5-IOD has been evaluated to depict the efficacy of the contributed model on resource-deficient UAVs in the IoD environment.

1.3. Paper outline

The contents of the scheme are organized as stated below: Section II revisits the BSD2C-IoD scheme with respect to delivery and collection of data in IoD environment and addresses the concerns in BSD2C-IoD. Section III presents the proposed scheme countering the flaws in BSD2C-IoD. Section IV formally analyzes the proposed scheme using the ROR model and AVISPA and also depicts informal analysis in the end. Section V depicts the performance analysis. The last section concludes the scheme.

2. Revisiting BSD2C-IOD: Blockchain-oriented secure data transmission and collection scheme

The BSD2C-IOD presents a new blockchain-oriented secure data delivery and collection (DDC) scheme for the IoT-based 5G-enabled IoD ecosystem. The scheme assumes that all entities in the IoD system are well-synchronized with clock-timings so that the participants may employ timestamps to aid in thwarting replay attacks. Table 2 tabulates few significant notations as used in the scheme. The BSD2C-IOD comprises several procedures in its system model, such as system initialization procedure, registration procedure, access control procedure, secure DDC procedure, block generation, verification and addition in Blockchain center procedure, and the procedure for dynamic addition of drones. These procedures are elaborated in the following sub-sections.

2.1. System model

The system model for the 5G-oriented blockchain technology-based framework involves four entities, i.e., Registration Center (RC), Control Authorities (CAs), Ground station service providers (GSPs), and blockchain center (BC) as shown in Fig. 1. The RC and CA are responsible for the registration of CA_j, GSP_j, and drones DN_i inducted in various

Table 2
Notations description.

Notations	Significance
$Ep(u, v)$:	Elliptic Curve (Non-singular)
G :	Base point in $Ep(u, v)$ with n order
$a.G$:	Elliptic curve (EC)-based point multiplication
$A+B$	EC-based point Addition; A, B ϵ $Ep(u, v)$
RC:	Registration Center
CA_j:	jth control authority
GSP_j:	jth ground station service provider
DN_i	ith drone
ID_{RC}:	RC's identity
r_{RC}:	Master secret key of RC
Pub_{RC}:	Public key of RC ($Pub_{RC} = r_{RC}$.G)
ID_{CAj}:	Legal identity of CAj
r_{CAj}:	CAj's random private key
Pub_{CAj}:	CAj's public key ($Pub_{CAj} = r_{CAj}$.G)
mk_{CAj}:	Randomly generated master secret key of CAj
Pk_{CAj}:	Public key of CAj ($\mathrm{Pk}_{\mathrm{CAj}} = mk_{CAj}$.G)
$Cert_{CAj}$:	Certificate issued by RC to CAj
RTS_{CAj}:	Registration timestamp used by RC for CAj
ID_{GSPj}:	Legal identity of GSPj
RID_{GSPj}:	Pseudo-identity of GSPj
r_{GSPj}:	GSPj's random private key
Pub_{GSPj}:	GSPj's public key ($Pub_{GSPj}= r_{GSPj}.G$)
k_{GSPj}:	GSPj's private decryption key
Pk_{GSPj}:	GSPj's public encryption key
$Cert_{GSPj}$:	Certificate issued by CAj to GSPj
RTS_{GSPj}:	Registration timestamp issued by CAj for GSPj
ID_{DNi}:	Certificate issued by CAj to GSPj
RID_{DNi}:	Pseudo-identity of DNi
r_{DNi}:	Private certificate key of DNi
Pub_{DNi}:	Public signature key for DNi
k_{DNi}:	Private signature of DNi
Pk_{DNi}:	Public key for DNi ($Pk_{DNi} = k_{DNi}$.G)
$Cert_{DNi}$:	Certificate issued by CAj to DNi
E_{PkY}/ D_{kY}:	Public key encryption or decryption for entity Y

flying zones FZ_j [28-29]. The RC and CA_j are supposed to be fully trusted entities in the IoD-based environment. The GSPs collect data from drones and securely deliver them and form the transaction blocks for adding in the private blockchain in the Blockchain center [26-27].

2.2. System initialization

The registration center RC selects few system parameters as RC, initially picks a non-singular elliptic curve (EC) as $E_p(u, v)$: $y^2=x^3+ux+v$ $(mod\, p)$ over the field of Galois [12], i.e. GF(p) with large prime p, where $u, v \,\epsilon\, Z_p$ be the constants with condition $4u^3 + 27\, v^2 \neq 0\ (mod\, p)$ and zero point, i.e. point at infinity. Then, the RC chooses a base point $G \,\epsilon\, E_p\ (u, v)$ having order n as much as p. The RC chooses the collision-resistant cryptographic one-way hash function SHA-256 $h(.)$. Moreover, the RC chooses its identity ID_{RC}, long-term secret key termed as master key $r_{RC} \,\epsilon\, Z_p$, with the calculation of corresponding public key $Pub_{RC} = r_{RC}.\ G$. The RC keeps the master key as secret, while other factors including $\{E_p(u, v), G, h(.), Pub_{RC}\}$ are openly published.

2.3. Registration procedure

In the registration phase, the control room CA_j is registered by the trusted RC on an offline basis. Thereafter, the CA_j registers the entities GSP_j and the associated drones DA_i in a flying zone FZ_j. The registration procedures for the CA_j, DN_i and GSP_j entities are elaborated as under:

2.3.1. Registration of CA_j

The RA adopts the following procedure to register the CA_j:

Step 1. RC chooses an identity ID_{CAj} for every CA_j, and selects a random private key $r_{CAj} \in Z^*_p$. Then it calculates a corresponding public key as $Pub_{CAj} = r_{CAj} \cdot G$, where $k{\cdot}G$ represents the elliptic

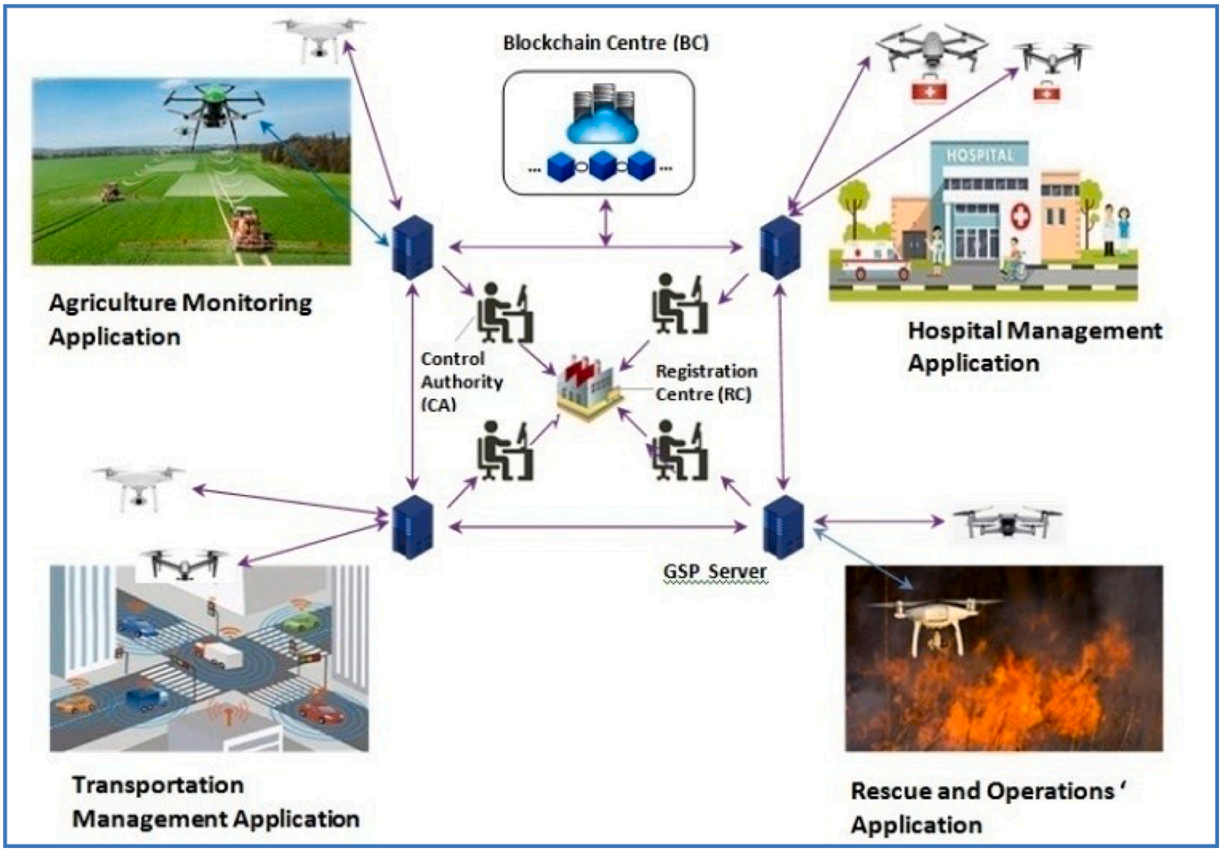


Fig. 1. Blockchain-enabled 5G oriented IoD ecosystem.

curve-based scalar point multiplication given that $k \in Z^*_p$. The RC generates a certificate for all CA_j entities as $Cert_{CAj} = r_{CAj} + h(ID_{CAj} || h(ID_{RC} || Pub_{RC} || Pub_{CAj} || RTS_{CAj}) * r_{RA} \pmod{p}$, where * represents modular multiplication, and RTS_{CAj} be the registration timestamp for CA_j. Thereafter, the RC deletes the factor r_{CAj} from its repository.
Step 2. Next before deployment, the RC stores the parameters in the memory of CA_j, i.e. $\{ID_{CAj}, ID_{RC}, Cert_{CAj}, Pub_{RC}, Pub_{CAj}, Ep(u, v), h(\cdot), G\}$.
Step 3. The CA_j selects a random master key as $mk_{CAj} \epsilon Z^*p$ and calculates the related public key $Pk_{CAj} = mk_{CAj} \cdot G$. Ultimately, RC publicly publishes the information as $\{ Pub_{RC}, Pub_{CAj}, E_p(u, v), h(\cdot); G\}$, while the CA_j holds ultimate parameters in its repository as $\{ID_{CAj}, ID_{RC}, Cert_{CAj}, Pk_{CAj}, Pub_{RC}, Pub_{CAj}, Ep(u, v), h(\cdot), G\}$.

2.3.2. Registration of GSP_j

The registration of GSP_j is performed by CA_j with the help of the following steps:

Step 1: Initially, the CA_j chooses a unique identity ID_{GSPj} and calculates the corresponding pseudo-identity $RID_{GSPj} = h(ID_{GSPj} || RTS_{GSPj} || mk_{CAj})$ where RTS_{GSPj} represent the registration timestamp for GSP_j. Then the CA_j chooses a random private key $r_{GSPj} \epsilon Zp^*$ and a corresponding public key $Pub_{GSPj} = r_{GSPj} \cdot G$. Besides, this CA_j computes a certificate for GSP_j as $Cert_{GSPj} = r_{GSPj} + h(RID_{GSPj} ||IDCAj|| PubCAj || Pub_{GSPj}) * mk_{CAj} \pmod{p}$.
Step 2: The CA_j stores the parameters RID_{GSPj} and $Cert_{GSPj}$ related to GSP_j in its repository while publishing the public key Pub_{GSPj} . Then for the sake of security, it deletes ID_{GSPj} and $rGSPj$ from its repository. Here, the GSP_j also chooses its decryption-based private key $k_{GSPj} \epsilon Zp^*$ and the related public key $Pk_{GSPj} = k_{GSPj} \cdot G$ for the sake of encryption.
Step 3: Lastly, the CA_j before deployment of the GSP_j, preloads it with the parameters as $\{RID_{GSPj}, ID_{CAj}, Cert_{GSPj}, Pub_{CAj}, Pub_{GSPj}, (k_{GSPj}, Pk_{GSPj}), Pk_{CAj}, Ep(u, v); h(\cdot), G\}$. Moreover, the CA_j for each GSPj, stores the public key Pk_{GSPj} in its repository and finally publishes the keys $\{Pub_{GSPj}, Pk_{GSPj}\}$ publicly.

2.3.3. Registration of Drone DN_i

The CA_j registers all drones DN_i before its deployment in the corresponding flying zone by adopting the following steps:

Step 1: Initially, the CA_j chooses an identity ID_{DNi} and also calculates the corresponding pseudo-identity $RID_{DNi} = h(ID_{DNi} || ID_{CAj} || mk_{CAj} || RTS_{DNi})$ in relation to each DN_i, where RTS_{DNi} denotes the registration timestamp.
Step 2: Next, the CA_j selects a certificate-based private key $r_{DNi} \epsilon Zp^*$, and calculate the related public key for each DN_i as $Pub_{DNi} = r_{DNi} \cdot G$, while the signature-based private key is $k_{DNi} \epsilon Zp^*$ and the corresponding signature-based public key for each DN_i as $Pk_{DNi} = k_{DNi} \cdot G$.
Step 3: Then, CA_j generates a certificate with respect to each drone DN_i as $Cert_{DNi} = r_{DNi} + h(RID_{DNi}|| Pub_{CAj} || Pub_{GSPj} || Pub_{DNi}) * mk_{CAj} \pmod{p}$. Next, it would delete ID_{DNi} and r_{DNi} from its repository. Finally, it stores the parameters $\{RID_{DNi}, Cert_{DNi}, (k_{DNi}, Pk_{DNi}), Pk_{CAj}, Ep(u, v), h(\cdot), G\}$ before deployment in a specific flying zone FZj.

2.4. Mutual authentication between DN_j and GSP_j

In this phase, the drone DN_i and the corresponding GSP_j in a flying zone FZ_j are mutually authenticated. Both of these entities are initialized with preliminary information in the registration phase. This procedure employs elliptic curve cryptography (ECC) for the generation of signatures, verification of certificates, and signatures. Upon successfully completing this procedure, the entities DN_i and GSP_j develop a mutually agreed session key as $SKV_{DNi,\ GSPj}= SKV_{GSPj,\ DNi}$. The following steps are included in this phase.

Step 1. Initially the DN_i chooses a random integer $r_1 \in Z^*p$ and engenders a fresh timestamp TS_1, and computes $r1' = h(RIDDNi ||r1 || Cert_{DNi} ||k_{DNi} ||TS1)$, $A_{DNi} = r1' \cdot G$. Then, DN_i computes a signature Sig_{DNi} on $r1'$ as $Sig_{DNi} = r_1' + h(Pk_{DNi} ||RID_{DNi} ||Pk_{CAj} ||Pub_{GSPj} ||A_{DNi}|| TS1) * k_{DNi} \pmod{p}$. After that DN_i constructs the authentication request message as $Msg_1 = \{RID_{DNi}, A_{DNi}, Cert_{DNi}, Sig_{DNi}, TS_1\}$ and submits towards GSP_j using a public channel.
Step 2. Upon receiving the request $Msg1$, the GSP_j validates timestamp TS_1. If it is fresh, the GSP_j verifies the certificate of DNi using the equality $Cert_{DNi} \cdot G =Pub_{DNi} + h(RID_{DNi} ||Pub_{CRj} ||Pub_{GSPj} || Pub_{DNi}) \cdot Pk_{CRj}$. If the verification fails, it declines the request; otherwise it further confirms the validity of signature using the condition $SigDNi \cdot G = A_{DNi} + h(Pk_{DNi} ||RID_{DNi}||Pk_{CAj} ||Pub_{GSPj} ||A_{DNi} ||TS_1) \cdot Pk_{DNi}$. It further proceeds to next step, if the signature verification holds true.
Step 3. Next, the GSP_j engenders a random integer $r_2 \in Z^*_p$ with a fresh timestamp TS_2. Then it calculates $r_2' = h(RID_{GSPj}||IDCAj ||r_2 || Cert_{GSPj} ||k_{GSPj}||TS2)$, $B_{GSPj}= r_2' \cdot G$. Thereafter, the GSP_j calculates Diffie-Hellman based key as $DHK_{GSPj,\ DNi} = r_2' \cdot A_{DNi} (= (r_2' * r_1') \cdot G)$. Next, it computes the session key $SK_{GSPj,\ DNi} = h(DHK_{GSPj,\ DNi} || RID_{DNi}||RID_{GSPj}||P\ k_{DNi} ||Pub_{GSPj})$ as well as session key verifier as $SKV_{GSPj,\ DNi}= h(SK_{GSPj,\ DNi} ||RID_{DNi} ||RID_{GSPj} ||B_{GSPj}||Cert_{GSPj} ||TS_1 || TS_2)$. In the last, GSP_j constructs the response message as $Msg_2 = \{RID_{GSPj}, Cert_{GSPj}, B_{GSPj}, SKV_{GSPj,DNi}, TS_2\}$ and delivers to DN_i on a public channel.
Step 4. Upon receiving the message Msg_2, the DN_i checks the genuineness of timestamp TS_2. If it is fresh, the DN_i further verifies the GSP_j's certificate as $Cert_{GSPj} \cdot G = Pub_{GSPj} + h(RID_{GSPj} ||ID_{CAj} ||Pub_{CAj} ||Pub_{GSPj}) \cdot Pk_{CNj}$. After the successful validation of certificate, the DNi builds the Diffie-Hellman based key as $DHK_{CNj,\ GSPj} = r_1' \cdot B_{GSPj}(= (r_1' * r_2') \cdot G = DHK_{GSPj,DNi})$, and recovers the session key as $SK_{DNi,GSPj} = h(DHK_{DNi,GSPj}||RID_{DNi} ||RID_{GSPj}||Pk_{DNi} ||Pub_{GSPj})$, and also derives $SKV_{DNi,GSPj}= h(SK_{DNi,GSPj} ||RID_{DNi} ||RID_{GSPj} ||B_{GSPj} || Cert_{GSPj}||TS_1 ||TS_2)$. Thereafter, the DN_i matches the equality for $SKV_{DNi,\ GSPj}= SKV_{GSPj,\ DNi}$. If it holds true, the DN_i builds a fresh timestamp TS_3 as well as an acknowledgement message as $ACK_{DNi,GSPj} = h(SK_{DNi,\ GSPj} ||TS2 ||TS3)$. Lastly, the DN_i forwards the message $Msg_3 = \{ ACK_{DNi,GSPj}, TS_3\}$ to GSP_j through public channel.
Step 5: After the receipt of message Msg_3, the GSP_j verifies the freshness of timestamp TS_3. If this is valid, the GSP_j calculates $ACK_{GSPj,\ DNi} = h(SK_{GSPj,\ DNi}||TS_2 ||TS_3)$ and compare the equality for $ACK_{GSPj,\ DNi}= ACK_{DNi,\ GSPj}$. If it holds true, an agreed session key $SK_{DNi,GSPj}$ $(=SK_{DNi,GSPj})$ is established as between the drone DN_i and GSP_j.

2.5. Cryptanalysis of BSD2C-IOD

The BSD2C-IOD scheme is exposed to the following vulnerabilities.

1 No GSP_j's signature verification

One of the major drawbacks in BSD2C-IOD is that in this scheme, the drone DN_i is unable to duly authenticate the GSP_j entity, since DN_i does not verify the constructed signature of GSP_j in the protocol. After the receipt of the response message $Msg2$ from GSP_j, the DN_i only verifies the certificate of GSP_j as issued by the CA_j. Although the scheme provides unilateral authentication since the GSP_j properly verifies the authenticity of DNi through the validation of signature as created by the DN_i. The mutual authenticity bounds both of the entities to authenticate one another; however, this feature is missing in BSD2C-IOD.

1 **No drone DNi's anonymity**

Secondly, the scheme BSD2C-IOD does not provide anonymity or untraceability to the drone DN_i. This is because the pseudo-identity RID_{DNi} for DN_i remains same in each session. An adversary may comfortably link different sessions upon interception of the parameters for various sessions on public channel. This flaw can be remedied with the renewal of pseudo-identity parameters on both ends each time a session is terminated.

1 **Inefficient use of nonces**

The scheme BSD2C-IOD makes inefficient use of r_1 and r_2 nonces after engendering them. The judicious use of those nonces may ensure mutual authenticity to both participants such that the session key remains protected even if the public and private secret keys are revealed to the adversary.

3. BOD5-IOD: Blockchain-oriented secure data transmission and collection scheme

This section demonstrates an improved and secure blockchain-oriented DDC protocol in order to improve BSD2C-IOD [13], meant for authenticating drones in the system. We proposed an enhanced blockchain-enabled AKA scheme BOD5-IOD to support a secure and robust access control mechanism between drones and GSP, which might assist protected transactions among all entities in IoD environment.

3.1. System initialization procedure

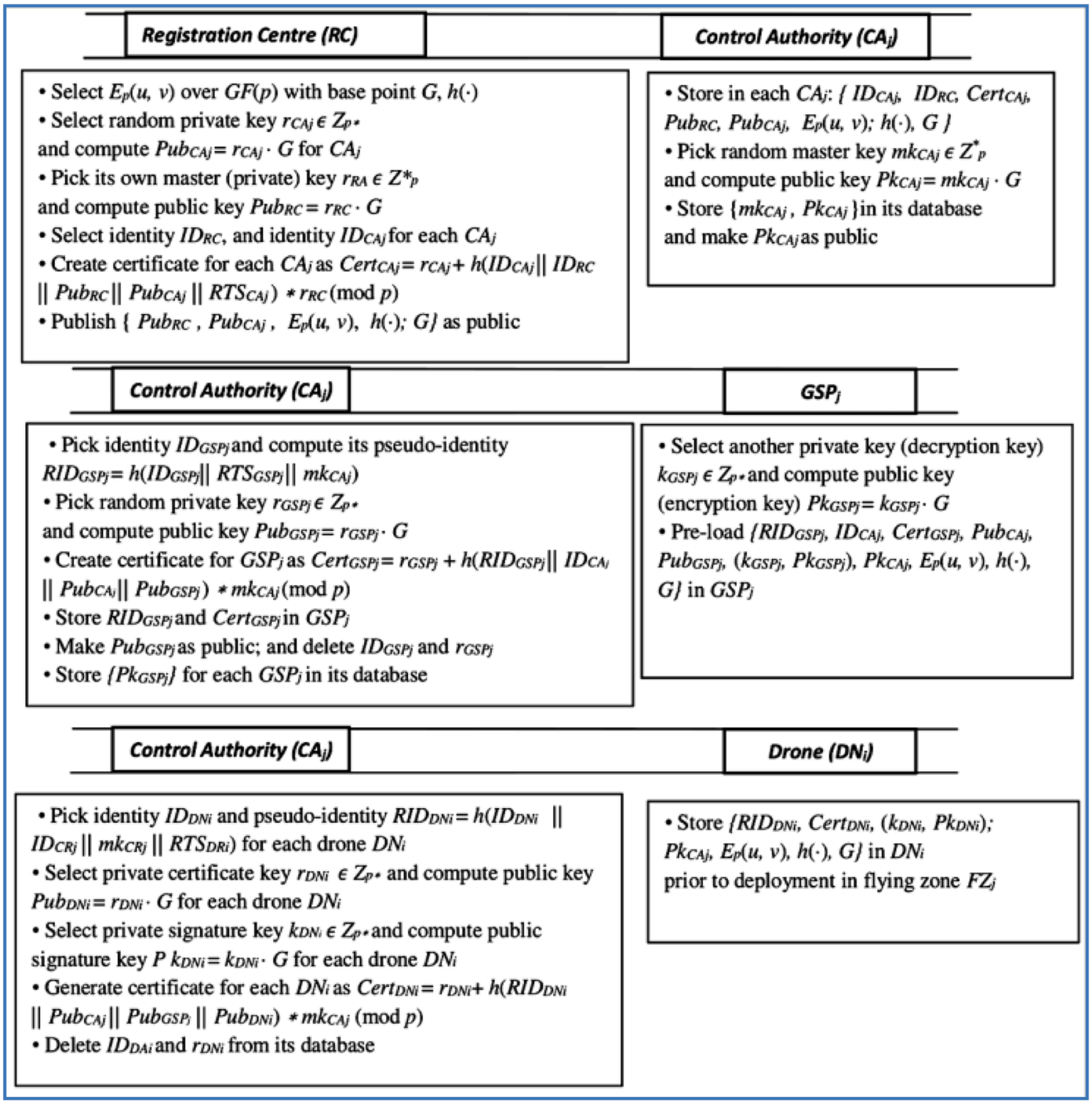
In BOD5-IOD, the registration center RC selects the system parameters such as identity ID_{RC}, master secret key $r_{RC} \in Zp$, public key $Pub_{RC} = r_{RC} \cdot G$ in the same manner as discussed in the initialization phase of BSD2C-IOD. The RC keeps the master key as secret, while other factors including $\{E_p(u, v), G, h(.), Pub_{RC}\}$ are openly published.

3.2. Registration procedure

In the registration phase, the control room CA_j is registered by the trusted RC on an offline basis. After that, the CA_j registers the entities GSP_j and the associated drones DA_i in a flying zone FZ_j. The steps involved in the registration procedure are depicted in Fig. 2. The registration procedures for the CA_j, DN_i and GSP_j entities are elaborated as under:

1 Registration of CA_j

The RA adopts the following procedure to register the CA_j:



Registration Centre (RC)

- Select $E_p(u, v)$ over $GF(p)$ with base point G, $h(\cdot)$
- Select random private key $r_{CAj} \in Z_{p^*}$ and compute $Pub_{CAj} = r_{CAj} \cdot G$ for CA_j
- Pick its own master (private) key $r_{RA} \in Z^*_p$ and compute public key $Pub_{RC} = r_{RC} \cdot G$
- Select identity ID_{RC}, and identity ID_{CAj} for each CA_j
- Create certificate for each CA_j as $Cert_{CAj} = r_{CAj} + h(ID_{CAj} \| ID_{RC} \| Pub_{RC} \| Pub_{CAj} \| RTS_{CAj}) * r_{RC} \pmod p$
- Publish $\{ Pub_{RC}, Pub_{CAj}, E_p(u, v), h(\cdot); G\}$ as public

Control Authority (CA_j)

- Store in each CA_j: $\{ ID_{CAj}, ID_{RC}, Cert_{CAj}, Pub_{RC}, Pub_{CAj}, E_p(u, v); h(\cdot), G \}$
- Pick random master key $mk_{CAj} \in Z^*_p$ and compute public key $Pk_{CAj} = mk_{CAj} \cdot G$
- Store $\{mk_{CAj}, Pk_{CAj}\}$ in its database and make Pk_{CAj} as public

Control Authority (CA_j)

- Pick identity ID_{GSPj} and compute its pseudo-identity $RID_{GSPj} = h(ID_{GSPj} \| RTS_{GSPj} \| mk_{CAj})$
- Pick random private key $r_{GSPj} \in Z_{p^*}$ and compute public key $Pub_{GSPj} = r_{GSPj} \cdot G$
- Create certificate for GSP_j as $Cert_{GSPj} = r_{GSPj} + h(RID_{GSPj} \| ID_{CAj} \| Pub_{CAj} \| Pub_{GSPj}) * mk_{CAj} \pmod p$
- Store RID_{GSPj} and $Cert_{GSPj}$ in GSP_j
- Make Pub_{GSPj} as public; and delete ID_{GSPj} and r_{GSPj}
- Store $\{Pk_{GSPj}\}$ for each GSP_j in its database

GSP_j

- Select another private key (decryption key) $k_{GSPj} \in Z_{p^*}$ and compute public key (encryption key) $Pk_{GSPj} = k_{GSPj} \cdot G$
- Pre-load $\{RID_{GSPj}, ID_{CAj}, Cert_{GSPj}, Pub_{CAj}, Pub_{GSPj}, (k_{GSPj}, Pk_{GSPj}), Pk_{CAj}, E_p(u, v), h(\cdot), G\}$ in GSP_j

Control Authority (CA_j)

- Pick identity ID_{DNi} and pseudo-identity $RID_{DNi} = h(ID_{DNi} \| ID_{CRj} \| mk_{CRj} \| RTS_{DRi})$ for each drone DN_i
- Select private certificate key $r_{DNi} \in Z_{p^*}$ and compute public key $Pub_{DNi} = r_{DNi} \cdot G$ for each drone DN_i
- Select private signature key $k_{DNi} \in Z_{p^*}$ and compute public signature key $P\,k_{DNi} = k_{DNi} \cdot G$ for each drone DN_i
- Generate certificate for each DN_i as $Cert_{DNi} = r_{DNi} + h(RID_{DNi} \| Pub_{CAj} \| Pub_{GSPj} \| Pub_{DNi}) * mk_{CAj} \pmod p$
- Delete ID_{DAi} and r_{DNi} from its database

Drone (DN_i)

- Store $\{RID_{DNi}, Cert_{DNi}, (k_{DNi}, Pk_{DNi}); Pk_{CAj}, E_p(u, v), h(\cdot), G\}$ in DN_i prior to deployment in flying zone FZ_j

Fig. 2. Registration phase.

Step 1. RC chooses an identity ID_{CAj} for every CA_j, and selects a random private key $r_{CAj} \in Z^*_p$. Then it calculates a corresponding public key as $Pub_{CAj} = r_{CAj} \cdot G$, where $k \cdot G$ represents the elliptic curve-based scalar point multiplication given that $k \in Z^*_p$. The RC generates a certificate for all CA_j entities as $Cert_{CAj} = r_{CAj} + h(ID_{CAj} \,||\, h(ID_{RC} \,||\, Pub_{RC} \,||\, Pub_{CAj} \,||\, RTS_{CAj}) * r_{RA} \pmod p$, where * represents modular multiplication, and RTS_{CAj} be the registration timestamp for CA_j. Thereafter, the RC deletes the factor r_{CAj} from its repository.
Step 2. Next before deployment, the RC stores the parameters in the memory of CA_j, i.e. $\{ID_{CAj}, ID_{RC}, Cert_{CAj}, Pub_{RC}, Pub_{CAj}, Ep(u, v), h(\cdot), G\}$.
Step 3. The CA_j selects a random master key as $mk_{CAj} \epsilon Z^*p$ and calculates the related public key $Pk_{CAj} = mk_{CAj} \cdot G$. Ultimately, the RC publicly publishes the information as $\{ Pub_{RC}, Pub_{CAj}, E_p(u, v), h(\cdot); G\}$, while the CA_j holds ultimate parameters in its repository as $\{ID_{CAj}, ID_{RC}, Cert_{CAj}, Pk_{CAj}, Pub_{RC}, Pub_{CAj}, Ep(u, v), h(\cdot), G\}$.

1 **Registration of GSP_j:**

CAj performs the registration of GSP_j with the help of the following steps:

Step 1: Initially, the CA_j chooses a unique identity ID_{GSPj} and calculates the corresponding pseudo-identity $RID_{GSPj} = h(ID_{GSPj} \,||\, RTS_{GSPj} \,||\, mk_{CAj})$ where RTS_{GSPj} represent the registration timestamp for GSP_j. Then the CA_j chooses a random private key $r_{GSPj} \epsilon Zp^*$ and a corresponding public key $Pub_{GSPj} = r_{GSPj} \cdot G$. Besides, this CA_j computes a certificate for GSP_j as $Cert_{GSPj} = r_{GSPj} + h(RID_{GSPj} \,||IDCAj||\, PubCAj \,||\, Pub_{GSPj}) * mk_{CAj} \pmod p$.
Step 2: The CA_j stores the parameters RID_{GSPj} and $Cert_{GSPj}$ related to GSP_j in its repository while publishing the public key Pub_{GSPj} . Then for the sake of security, it deletes ID_{GSPj} and r_{GSPj} from its repository. Here, the GSP_j also chooses its decryption-based private key $k_{GSPj} \epsilon Zp^*$ and the related public key $Pk_{GSPj} = k_{GSPj} \cdot G$ for the sake of encryption.
Step 3: Lastly, the CA_j before deployment of the GSP_j, preloads it with the parameters as $\{RID_{GSPj}, ID_{CAj}, Cert_{GSPj}, Pub_{CAj}, Pub_{GSPj}, (k_{GSPj}, Pk_{GSPj}), Pk_{CAj}, Ep(u, v); h(\cdot), G\}$. Moreover, the CAj for each GSPj, stores the public key Pk_{GSPj} in its repository and finally publishes the keys $\{Pub_{GSPj}, Pk_{GSPj}\}$ publicly.

1 **Registration of Drone DN_i:**

The CA_j registers all drones DN_i before its deployment in the corresponding flying zone by adopting the following steps:

Step 1: Initially, the CA_j chooses an identity ID_{DNi} and also calculates the corresponding pseudo-identity $RID_{DNi} = h(ID_{DNi} \,||\, ID_{CAj} \,||\, mk_{CAj} \,||\, RTS_{DNi})$ in relation to each DN_i, where RTS_{DNi} denotes the registration timestamp.
Step 2: Next, the CA_j selects a certificate-based private key $r_{DNi} \epsilon Zp^*$, and calculate the related public key for each DN_i as $Pub_{DNi} = r_{DNi} \cdot G$, while the signature-based private key is $k_{DNi} \epsilon Zp^*$ and the corresponding signature-based public key for each DN_i as $Pk_{DNi} = k_{DNi} \cdot G$.
Step 3: Then, CA_j generates a certificate with respect to each drone DN_i as $Cert_{DNi} = r_{DNi} + h(RID_{DNi}||\, Pub_{CAj} \,||\, Pub_{GSPj} \,||\, Pub_{DNi}) * mk_{CAj} \pmod p$. Next, it would delete ID_{DNi} and r_{DNi} from its repository. Finally, it stores the parameters $\{RID_{DNi}, Cert_{DNi}, (k_{DNi}, Pk_{DNi}), Pk_{CAj}, Ep(u, v), h(\cdot), G\}$ before deployment in a specific flying zone FZ_j.

3.3. Mutual authentication between DN_j and GSP_j

In this phase, the drone DN_i and the corresponding GSP_j in a flying zone FZ_j are mutually authenticated. Both of these entities are initialized with preliminary information in the registration phase. This procedure employs elliptic curve cryptography (ECC) to generate signatures, verification of certificates, and signatures. Upon completing this procedure, the entities DN_i and GSP_j develop a mutually agreed session key as $SKV_{DNi,\ GSPj} = SKV_{GSPj,\ DNi}$. The following steps are included in this phase.

Step 1. Initially the DN_i chooses a random integer $r_1 \in Z^*p$ and engenders a fresh timestamp TS_1, and computes $A_{DNi} = r1 \cdot G$, $X_{DNi} = r_1 . PK_{GSPj}$, $ACert_{DNi} = Cert_{DNi} + r_1 . k_{DNi}$, $AID_{DNi} = RID_{DNi} \oplus X_{DNi}$. Then, DN_i computes a signature Sig_{DNi} on $r1$ as $Sig_{DNi} = r_1 + h(Pk_{DNi}||RID_{DNi}||Pk_{CNj}||Pub_{GSPj}||A_{DNi}||TS_1) * k_{DNi} \pmod p$. After that DN_i constructs the authentication request message as $Msg1 = \{AID_{DNi}, ADNi, ACert_{DNi}, Sig_{DNi}, TS1\}$ and submits towards GSP_j using a public channel.
Step 2. Upon receiving the request $Msg1$, the GSP_j validates timestamp TS_1. If it is fresh, the GSP_j computes $X_{DNi} = k_{GSPj} . A_{DNi}$, $RID_{DNi} = AID_{DNi} \oplus X_{DNi}$, and verifies the dynamic certificate of DN_i using the equality $ACert_{DNi} \cdot G = Pub_{DNi} + h(RID_{DNi}||Pub_{CAj}||\, Pub_{GSPj}||Pub_{DNi}) \cdot Pk_{CAj} + X_{DNi}$. If the verification fails, it declines the request; otherwise it further confirms the validity of signature using the condition $Sig_{DNi} \cdot G = A_{DNi} + h(Pk_{DNi} \,||RID_{DNi}||Pk_{CAj}||\, Pub_{GSPj}||A_{DNi} \,||TS_1) \cdot Pk_{DNi}$. It further proceeds to next step, if the signature verification holds true.
Step 3. Next, the GSP_j engenders a random integer $r_2 \in Z^*_p$ with fresh timestamp TS_2. Then it calculates $B_{GSPj} = r_2 \cdot G$, $X_{GSPj} = r_2 . PK_{DNi}$, $AID_{GSPj} = RID_{GSPj} \oplus X_{GSPj}$, and $ACert_{GSPj} = Cert_{GSPj} + r_2 . k_{GSPj}$. Next, it computes the session key $SK_{GSPj,\ DNi} = h(X_{DNi}||X_{GSPj}||RID_{DNi}||RID_{GSPj}||TS_1||TS_2)$ as well as session key verifier as $SKV_{GSPj,\ DNi} = h(SK_{GSPj,\ DNi}||B_{GSPj}||Cert_{GSPj}||TS1 \,||TS2)$. In the last, GSP_j constructs the response message as $Msg_2 = \{AID_{GSPj}, ACert_{GSPj}, B_{GSPj}, SKV_{GSPj,DNi}, TS_2\}$ and delivers to DNi on a public channel.
Step 4. Upon receiving the message Msg_2, the DN_i checks the genuineness of timestamp TS_2. If it is fresh, the DN_i computes $X_{GSPj} = k_{DNi} . B_{GSPj}$, $RID_{GSPj} = AID_{GSPj} \oplus X_{GSPj}$ and verifies the dynamic certificate as $ACert_{GSPj} . G = Pub_{GSPj} + h(RID_{GSPj}||\, ID_{CNj}||Pub_{CAj}||Pub_{GSPj}) \cdot Pk_{CAj} + X_{GSPj}$. In case the timestamp and the dynamic certificate are legal, it computes the session key as $SK_{DNi,GSPj} = h(X_{DNi}||X_{GSPj}||RID_{DNi}||RID_{GSPj}||\, TS_1||TS_2)$. Next, it validates the session key verifier as $SKV_{DNi,GSPj} = h(SK_{DNi,GSPj} \,||B_{GSPj}||Cert_{GSPj}||TS_1 \,||TS_2)$ as well. Thereafter, the DN_i matches the equality for $SKV_{DNi,GSPj} = SKV_{GSPj,\ DNi}$. If it holds true, the DN_i builds a fresh timestamp TS_3 as well as an acknowledgement message as $ACK_{DNi,GSPj} = h(SK_{DNi,GSPj}||TS_2 \,||TS_3)$. Lastly, the DN_i forwards the message $Msg_3 = \{ACK_{DNi,}GSP_j, TS_3\}$ to GSP_j through public channel.
Step 5: After the receipt of message Msg_3, the GSP_j verifies the freshness of timestamp TS_3. If this is valid, the GSP_j calculates $ACK_{GSPj,\ DNi} = h(SK_{GSPj,\ DNi}||TS_2 \,||TS_3)$ and compare the equality for $ACK_{GSPj,\ DNi} = ACK_{DNi,\ GSPj}$. If it holds true, and agreed session key $SK_{DNi,GSPj}$ $(=SK_{DNi,GSPj})$ is established as between the drone DN_i and GSP_j.

3.4. Secure data delivery and collection

This section elaborates on different Data Delivery And Collection (DDC)-based transactions among CA_j, GSP_j, and DN_i in any flying zone FZ_j. We employ few transactions as given below:

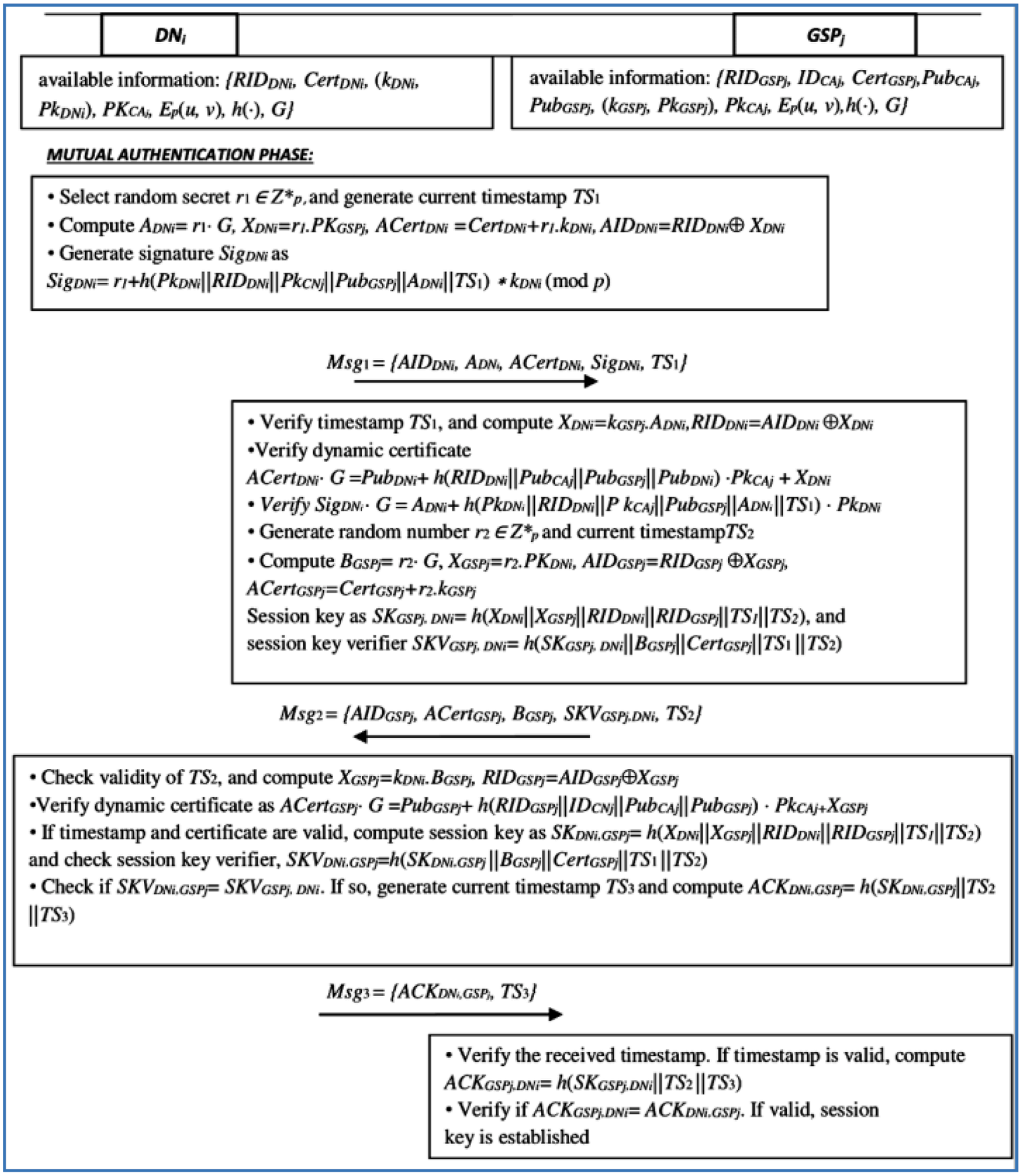
- We term the transaction as $Tr_{CA\text{-}GSP\text{-}rq}$ between CA_j to GSP_j regarding data delivery (DD) request from CA_j to GSP_j. This transaction is performed with secure encryption using the public key Pk_{GSPj} of GSP_j. This encrypted transaction, i.e $Tr_{CA\text{-}GSP\text{-}rq}$, will be decrypted by the GSP_j with the help of its own private key k_{GSPj}.
- The transaction $Tr_{CA\text{-}GSP\text{-}rq}$ represents the DD request from GSP_j to DN_i that gets encrypted using the created session key $SK_{DNi,\ GSPj}$ between DN_i and GSP_j. After the decryption of $Tr_{CA\text{-}GSP\text{-}rq}$ using $SK_{DNi,\ GSPj}$, the DN_i may handover the package delivery (say medicine, food deliveries etc) to the appropriate destination.

- Likewise, another transaction $Tr_{DN\text{-}GSP\text{-}rq}$ depicts the DDC response from DN_i to GSP_j, which may be encrypted using $SK_{DNi,\ GSPj}$.
- There might be other application scenarios, say smart transportation or smart agriculture etc, where the drones DNi after deployment require submitting the collected data in the form of secure transactions, i.e. $Tr_{DN\text{-}GSP\text{-}data}$ towards GSP_j with the help of session key $SK_{DNi,\ GSPj}$.

3.5. Block creation, verification and addition in BC center

A block is created in this phase by the GSP_j, and we assume a block $Block_i$ utilize the transactions as available to GSP_j which is also shown in Fig. 3. A lots of transactions encrypted with the GSP_j's public key can be contained in a $Block_i$ constituted by GSP_j. The GSP_j generates signatures on the block using elliptic-curve digital signature algorithm (ECDSA) [14]. The immutability as well as transparency features of the block are ensured with the use of created signature, Merkle tree, and the existing block hash root in the blockchain [13]. In P2P GSP-based network with n_{GSP} number of GSPs, a leader say L is selected with the help of any leader selection procedure or algorithm. Then, the block $Block_i$ is forwarded to the leader L to promote consensus for verification as well as addition in blockchain, which is depicted in algorithm 1. The Practical Byzantine Fault Tolerance (PBFT)-based consensus algorithm is employed [15].

The smart contract is deemed to be a digital agreement among the entities which could be executed and verified digitally by the entities themselves, and it could be implemented irrespective of any human involvement [16-17]. It enables the legal implementation of the transactions and contracts through online verification and validation procedures. Moreover, the agreement implementations among the participants are immutable, irreversible, and traceable. Following this, the blockchain system may act in a reliable, cost-effective, efficient and



available information: $\{RID_{DNi},\ Cert_{DNi},\ (k_{DNi},\ Pk_{DNi}),\ PK_{CAi},\ E_p(u,v),\ h(\cdot),\ G\}$

available information: $\{RID_{GSPj},\ ID_{CAj},\ Cert_{GSPj}, Pub_{CAj},\ Pub_{GSPj},\ (k_{GSPj},\ Pk_{GSPj}),\ Pk_{CAj},\ E_p(u,v), h(\cdot),\ G\}$

- Select random secret $r_1 \in Z^*_p$, and generate current timestamp TS_1
- Compute $A_{DNi} = r_1 \cdot G$, $X_{DNi} = r_1.PK_{GSPj}$, $ACert_{DNi} = Cert_{DNi} + r_1.k_{DNi}$, $AID_{DNi} = RID_{DNi} \oplus X_{DNi}$
- Generate signature Sig_{DNi} as
$Sig_{DNi} = r_1 + h(Pk_{DNi} \| RID_{DNi} \| Pk_{CNj} \| Pub_{GSPj} \| A_{DNi} \| TS_1) * k_{DNi} \pmod p$

$Msg_1 = \{AID_{DNi},\ A_{DNi},\ ACert_{DNi},\ Sig_{DNi},\ TS_1\}$ →

- Verify timestamp TS_1, and compute $X_{DNi} = k_{GSPj}.A_{DNi}, RID_{DNi} = AID_{DNi} \oplus X_{DNi}$
- Verify dynamic certificate
$ACert_{DNi} \cdot G = Pub_{DNi} + h(RID_{DNi} \| Pub_{CAj} \| Pub_{GSPj} \| Pub_{DNi}) \cdot Pk_{CAj} + X_{DNi}$
- Verify $Sig_{DNi} \cdot G = A_{DNi} + h(Pk_{DNi} \| RID_{DNi} \| P\,k_{CAj} \| Pub_{GSPj} \| A_{DNi} \| TS_1) \cdot Pk_{DNi}$
- Generate random number $r_2 \in Z^*_p$ and current timestamp TS_2
- Compute $B_{GSPj} = r_2 \cdot G$, $X_{GSPj} = r_2.PK_{DNi}$, $AID_{GSPj} = RID_{GSPj} \oplus X_{GSPj}$,
$ACert_{GSPj} = Cert_{GSPj} + r_2.k_{GSPj}$
Session key as $SK_{GSPj.\ DNi} = h(X_{DNi} \| X_{GSPj} \| RID_{DNi} \| RID_{GSPj} \| TS_1 \| TS_2)$, and
session key verifier $SKV_{GSPj.\ DNi} = h(SK_{GSPj.\ DNi} \| B_{GSPj} \| Cert_{GSPj} \| TS_1 \| TS_2)$

$Msg_2 = \{AID_{GSPj},\ ACert_{GSPj},\ B_{GSPj},\ SKV_{GSPj.DNi},\ TS_2\}$ ←

- Check validity of TS_2, and compute $X_{GSPj} = k_{DNi}.B_{GSPj}$, $RID_{GSPj} = AID_{GSPj} \oplus X_{GSPj}$
- Verify dynamic certificate as $ACert_{GSPj} \cdot G = Pub_{GSPj} + h(RID_{GSPj} \| ID_{CNj} \| Pub_{CAj} \| Pub_{GSPj}) \cdot Pk_{CAj} + X_{GSPj}$
- If timestamp and certificate are valid, compute session key as $SK_{DNi.GSPj} = h(X_{DNi} \| X_{GSPj} \| RID_{DNi} \| RID_{GSPj} \| TS_1 \| TS_2)$ and check session key verifier, $SKV_{DNi.GSPj} = h(SK_{DNi.GSPj} \| B_{GSPj} \| Cert_{GSPj} \| TS_1 \| TS_2)$
- Check if $SKV_{DNi.GSPj} = SKV_{GSPj.\ DNi}$. If so, generate current timestamp TS_3 and compute $ACK_{DNi.GSPj} = h(SK_{DNi.GSPj} \| TS_2 \| TS_3)$

$Msg_3 = \{ACK_{DNi,GSPj},\ TS_3\}$ →

- Verify the received timestamp. If timestamp is valid, compute $ACK_{GSPj.DNi} = h(SK_{GSPj.DNi} \| TS_2 \| TS_3)$
- Verify if $ACK_{GSPj.DNi} = ACK_{DNi.GSPj}$. If valid, session key is established

Fig. 3. Proposed mutual authentication.

secure manner. In the proposed scheme (BOD5-IOD), the smart contract may be employed in each GSP to verify the transactions as collected from different participating entities and the created blocks by the GSP in the framework. Consequently, a man-in the middle-attack may be successfully avoided in smart contracts due to robust integrity in the BC system. Hence, the BC technology in support of smart contracts may be used potentially for secure communication among the autonomous agents in the contributed scheme (BOD5-IOD).

3.6. Adding drones dynamically into the system

The drones may also be captured physically or malfunctioned by an attacker. Consequently, a few new drones can be added in the IOD-based environment. For instance, a new drone entity $\mathrm{DN}_{\mathrm{i}}^{\mathrm{new}}$ may be dynamically added in any flying zone FZ_{j}. For the implementation of this task, the control authority CAj chooses a unique identity $\mathrm{ID}_{\mathrm{DNi}}^{\mathrm{new}}$ and computes associated pseudo-identity $\mathrm{RID}_{\mathrm{DNi}}^{\mathrm{new}} = \mathrm{h}(\mathrm{ID}_{\mathrm{DNi}}^{\mathrm{new}} \,||\, \mathrm{ID}_{\mathrm{CAj}} \,||\, \mathrm{mk}_{\mathrm{CAj}} \,||\, \mathrm{RTS}_{\mathrm{DNi}}^{\mathrm{new}})$, while $\mathrm{RTS}_{\mathrm{DNi}}^{\mathrm{new}}$ being the registration timestamp. Thereafter, CAj selects a private certificate key $\mathrm{r}_{\mathrm{DNi}}^{\mathrm{new}}$ and an associated public key as $\mathrm{Pub}_{\mathrm{DNi}}^{\mathrm{new}}= \mathrm{r}_{\mathrm{DNi}}^{\mathrm{new}}.$ G, and then it picks private signature key $\mathrm{k}_{\mathrm{DNi}}^{\mathrm{new}}$ as well as public signature key $\mathrm{Pk}_{\mathrm{DNi}}^{\mathrm{new}} = \mathrm{k}_{\mathrm{DNi}}^{\mathrm{new}}$. G for $\mathrm{DNi}^{\mathrm{new}}$. Next, the CAj constructs a certificate in relation to $\mathrm{DNi}^{\mathrm{new}}$ as $= Cert_{\mathrm{DNi}}^{\mathrm{new}} = r_{DNi}^{\mathrm{new}} + h(RID_{DNi}^{\mathrm{new}} \,||\, Pub_{CAj} \,||\, Pub_{GSPj} \,||\, Pub_{\mathrm{DNi}}^{\mathrm{new}}) * mk_{CAj} \pmod p$. Eventually, the CAj stores the contents $\{ RID_{\mathrm{DNi}}{}^{\mathrm{new}}, Cert_{\mathrm{DNi}}^{\mathrm{new}}, (k_{\mathrm{DNi}}^{\mathrm{new}}, Pk_{\mathrm{DNi}}^{\mathrm{new}}), Pub_{CAj}, Ep(u; v); h(\cdot); G\}$ before deploying $\mathrm{DNi}^{\mathrm{new}}$ in the flying zone FZj. Then, the CA_j deletes the parameters $\mathrm{ID}_{\mathrm{DNi}}^{\mathrm{new}}$ and $\mathrm{r}_{\mathrm{DNi}}^{\mathrm{new}}$ from its repository to boost the security.

4. Security analysis

This section demonstrates formally and informally that BOD5-IOD may resist several potential threats posed to other contemporary authentication protocols tailored for IoD system environment.

4.1. Formal security analysis employing ROR Model

In this analysis, we employ a widely adopted Real-Or-Random (ROM) oracle model [18] as regards to BOD5-IOD for proving the mutual authenticity of agreed session key between DN_i and GSP_i against the malicious attacker $\mathscr{A}$. A semantic security-based narrative on ROR model is depicted is Definition 1 and Theorem 1. To achieve this objective, $\mathscr{A}$ implements the queries as defined in Table 3. Moreover, the approach to "collision defiant, cryptographic one-way hash digest function h(.)" is provided for all participating entities, including the attacker $\mathscr{A}$. In BOD5-IOD, the function h(.) is modeled as a random oracle.

Participants: In BOD5-IOD, the four entities participate in the mutual authentication phase, i.e. RC, CA_{j}, DN_{i}, and $\mathrm{GSP}_{\mathrm{j}}$. The DNi and GSPj mutually interact with each other to create session key without the involvement of RC. We assume that the notations $\mathscr{L}^{\ell 1}{}_{\mathrm{DNi}}$ and $\mathscr{L}^{\ell 2}{}_{\mathrm{GSPj}}$ characterize ℓ_1^{th} and ℓ_2^{th} instances for the entities DNi and GSPj, respectively. We term those instances as the random oracles.

Accepted state: Upon the receipt of the legitimate last communication message, the instance $\mathscr{L}^{\ell}$ comes to an accepted state. After getting all of the related communication messages for any session, those messages are brought into a sequence, and then term an identity *sid* of $\mathscr{L}^{\ell}$ for identifying the session of the current session.

Table 3
Queries and their objectives.

Queries	Objective
Execute($\mathscr{L}^{\ell 1}_{DN_i}$, $\mathscr{L}^{\ell 2}_{GSP_j}$)	$\mathscr{A}$ employs this query to forge messages exchanged between DNi and GSPj
Compromise_Drone ($\mathscr{L}^{\ell 1}_{DN_i}$)	$\mathscr{A}$ employs this query to get secret credentials from the memory of compromised DNi
Reveal ($\mathscr{L}^{\ell}$)	$\mathscr{A}$ employs this query to reveal session key as shared between $\mathscr{L}^{\ell}$ and its associated partner
Test ($\mathscr{L}^{\ell}$)	$\mathscr{A}$ employs this query to verify the revealed session key by using the randomly flipped unbiased coin b

Partnering: The interacting instances such as $\mathscr{L}^{\ell 1}$ and $\mathscr{L}^{\ell 2}$ serve as partners to one another in case those instances satisfy the conditions as given below:

- The instances $\mathscr{L}^{\ell 1}$ and $\mathscr{L}^{\ell 2}$ must be in accepted states.
- The instances $\mathscr{L}^{\ell 1}$ and $\mathscr{L}^{\ell 2}$ must share the same session identity sid and authenticate each on a mutual basis.
- The instances $\mathscr{L}^{\ell 1}$ and $\mathscr{L}^{\ell 2}$ must be partners serving on mutual basis.

Freshness: The instances $\mathscr{L}^{\ell 1}{}_{\mathrm{DNi}}$ and $\mathscr{L}^{\ell 2}{}_{\mathrm{GSPj}}$ are regarded as fresh if the constructed session key $SK_{DNi,\ GSPj}$ $(=SK_{GSPj,\ DNi})$ between the entities DNi and GSPj is not revealed to the adversary with the use of Reveal ($\mathscr{L}^{\ell}$) query as shown in Table 3. The semantic security of the contributed model BOD5-IOD is defined in Definition 1, forming the basis of Theorem 1.

Definition 1. We assume an advantage for the attacker be $Adv_{\mathscr{A}}^{BOD5-IOD}\ (\mathscr{I}_p)$ in the polynomial amount of time $\mathscr{I}_p$ in compromising the semantic security of BOD5-IOD in regards to calculating the agreed session key $SK_{DNi,\ GSPj}$ $(=SK_{GSPj,\ DNi})$ between $\mathrm{GSP}_{\mathrm{j}}$ and DN_{i} for a specific session. Then

$$Adv_{\mathscr{A}}^{BOD5-IOD}\ (\mathscr{I}_p) = |2.Pr[b' = b] - 1| \tag{1}$$

Where *b*' and *b* represent guessed and correct bits, respectively.

Theorem 1. We assume an attacker $\mathscr{A}$ running in polynomial amount of time $\mathscr{I}_p$ attempting to calculate the session key $SK_{DNi,\ GSPj}$ $(=SK_{GSPj,\ DNi})$, which is shared between DNi and GSPj as regards to any specific session in the suggested model, BOD5-IOD. If q_{sh}, |hash|, and $Adv_{\mathscr{A}}^{ECD-DHP}\ (\mathscr{I}_p)$ represent the number of hash function-based queries, the range capacity for cryptographic collision-resistant one-way hash function h(.), the advantage for compromising the Elliptic-Curve Decisional Diffie-Hellman Problem (ECDDHP), respectively. Consequently,

$$Adv_{\mathscr{A}}^{BOD5-IOD}\ (\mathscr{I}_p) \le \frac{q_{sh}^2}{|hash|} + Adv_{\mathscr{A}}^{ECD-DHP}\ (\mathscr{I}_p) \tag{2}$$

Proof. An attacker $\mathscr{A}$ plays three games, i.e. $Gm_j^{\mathscr{A}}\,(j = 0,\ 1,\ 2)$ to prove the security properties in BOD5-IOD. The $Sucs_{Gm_j}^{\mathscr{A}}$ represents an event that the attacker may correctly guess the bit *b* on a random basis in game $Gm_j^{\mathscr{A}}$. We can define the advantage of $\mathscr{A}$ in winning $Gm_j^{\mathscr{A}}$ for BOD5-IOD is defined as $Adv_{\mathscr{A},\ Gm_j}^{BOD5-IOD} = \Pr[\mathrm{Sucs}_{\mathrm{Gm_j}}^{\mathscr{A}}]$. Each of the games $Gm_j^{\mathscr{A}}$ may be illustrated as under:

$Gm_0^{\mathscr{A}}$: In this game, the adversary $\mathscr{A}$ launches an actual attack against BOD5-IOD with the use of Real-Or-Random (ROR) model. For this, $\mathscr{A}$ chooses a random bit *b* before the initiation of game $Gm_0^{\mathscr{A}}$. The semantic security as described in the Definition 1 can be represented as:

$$Adv_{\mathscr{A}}^{BOD5-IOD}\ (\mathscr{I}_p) = \left|2\ Adv_{\mathscr{A},\ Gm_0}^{ECD-DHP}\ (\mathscr{I}_p) - 1\right| \tag{3}$$

$Gm_1^{\mathscr{A}}$*:* The game $Gm_1^{\mathscr{A}}$ may correspond to an eavesdropping game in which the adversary performs an Execute query as shown in Table 3. With the use of this query, the adversary may attempt to recover the session key $SK_{DNi,\ GSPj}$ $(= SK_{GSPj,\ DNi})$ out of all seized communication messages on public channel, i.e. $Msg1 = \{AID_{DNi}, A_{DNi}, ACert_{DNi}, Sig_{DNi}, TS_1\}$, $Msg_2 = \{AID_{GSPj}, ACert_{GSPj}, B_{GSPj}, SKV_{GSPj,DNi}, TS_2\}$, and $Msg_3 = \{ACK_{DNi,}GSP_j, TS_3\}$. Then, the adversary performs the execution of Test and Reveal queries for verifying the recovered session key. In this manner, it may discern whether the session key is legitimate or any random key. The legal session key is computed as $SK_{DNi,GSPj}= h(X_{DNi}||$

$X_{GSPj}|| RID_{DNi}||RID_{GSPj}|| TS_1||TS_2$), where $X_{GSPj}=k_{DNi}.B_{GSPj}$ and $RID_{GSPj}=AID_{GSPj}\oplus X_{GSPj}$. This computation implies $SK_{DNi,\ GSPj}$ ($= SK_{GSPj,\ DNi}$). This also suggests that merely the eavesdropping of messages Msg_1, Msg_2 and Msg_3 may not increase the success probability for the adversary to extract the long term secrets or the temporal credentials, this is because of the fact both of these parameters are protected under the collision-resistant one-way hash function *h(.)*. Hence both of the above games $Gm_0^{\mathscr{A}}$ and $Gm_1^{\mathscr{A}}$ remain indistinguishable in relation to eavesdropping threat. Consequently, it results into the following equation:

$$Adv_{\mathscr{A},\ Gm_1}^{BOD5-IOD} = Adv_{\mathscr{A},\ Gm_0}^{BOD5-IOD} \tag{4}$$

$Gm_2^{\mathscr{A}}$: In this game, the adversary models *Hash* as well as *Compromise_Drone* queries for launching an active attack. For recovering the session key $SK_{DNi,\ GSPj}$ ($= SK_{GSPj,\ DNi}$), the attacker requires X_{DNi} and X_{GSPj} parameters. However, even if the adversary is able to successfully eavesdrop the messages Msg_i $(1\leq i \leq 3)$, he/she would still require k_{DNi} to compute X_{GSPj} or r_1 parameter to compute X_{DNi}. The critical credentials are protected under the cryptographic one-way hash function. To recover these parameters, the attacker $\mathscr{A}$ must solve the ECD-DHP problem; nevertheless, it is a hard problem and unlikely to be solvable in a polynomial amount of time. Moreover, with the use of *Compromise_Drone* query, $\mathscr{A}$ might even recover k_{DNi}, yet without knowing r_1, r_2, and other related factors, it might not be able to compute session key $SK_{DNi,\ GSPj}$ ($= SK_{GSPj,\ DNi}$). Hence, both of these games remain indistinguishable upon the exclusion of modeling for *Compromise_Drone* and *Hash* queries. This advantage of hash-based collision resistance and the hardness for ECD-DHP leads to the under-mentioned birthday paradox:

$$\left| Adv_{\mathscr{A},\ Gm_1}^{BOD5-IOD} = Adv_{\mathscr{A},\ Gm_2}^{BOD5-IOD} \right|$$

$$\leq \frac{q_{sh}^2}{2|Hash|} + Adv_{\mathscr{A}}^{ECD-DHP}\left(\mathscr{I}_p\right) \tag{5}$$

With the use of illustrated games, the adversary requires to guess a bit b for winning game $Gm_2^{\mathscr{A}}$. Thus, we have,

$$Adv_{\mathscr{A},\ Gm_2}^{BOD5-IOD} = \frac{1}{2}$$

According to Eq. (1)

$$\frac{1}{2}.Adv_{\mathscr{A}}^{BOD5-IOD}\left(\mathscr{I}_p\right) = \left| Adv_{\mathscr{A},\ Gm_0}^{BOD5-IOD} - \frac{1}{2} \right|$$

After solving the Eqs. (2), (3) and (4) and considering the triangular inequality, we can derive the following equation:

$$\frac{1}{2}.Adv_{\mathscr{A}}^{BOD5-IOD}\left(\mathscr{I}_p\right)$$

$$= |\ Adv_{\mathscr{A},\ Gm_0}^{BOD5-IOD}\left(\mathscr{I}_p\right) - \left| Adv_{\mathscr{A},\ Gm_2}^{BOD5-IOD} \right|$$

$$= |\ Adv_{\mathscr{A},\ Gm_1}^{BOD5-IOD}\left(\mathscr{I}_p\right) - \left| Adv_{\mathscr{A},\ Gm_2}^{BOD5-IOD} \right|$$

$$\leq \frac{q_{sh}^2}{2|Hash|} + Adv_{\mathscr{A}}^{ECD-DHP}\left(\mathscr{I}_{\mathrm{p}}\right) \tag{6}$$

Ultimately, by using Eq (6) we get to the following derivation:

$$Adv_{\mathscr{A}}^{ECD-DHP}\left(\mathscr{I}_p\right) \leq \frac{q_{sh}^2}{|Hash|} + 2\ Adv_{\mathscr{A}}^{ECD-DHP}\left(\mathscr{I}_p\right) \tag{7}$$

4.2. AVISPA-based formal security verification

AVISPA [19,20] is an automated push-button tool to validate the features of authentication protocols and internet applications. The tool not only provides a modular approach to specify the security goals but also helps to demonstrate the protocol model in a specified formal language. It is implemented with various back-ends providing multiple heterogeneous state-of-the-art mechanisms for automatic protocol analysis. The AVISPA can implement four back-ends: a) On the fly mode-checker (OFMC), (b) Constraint logic-oriented Attack Searcher (CL-AtSe), (c) SAT-oriented Model Checker (SATMC), and (d) Tree Automata related to Automatic Approximations for Analyzing Security Protocols (TA4SP). For security verification on a formal basis, we performed the simulation of BOD5-IOD using "Security Protocol Animator for AVISPA (SPAN)". The corresponding results are reported in Fig. 3 using CL-AtSe and OFMC back-ends, while other back-ends such as TA4SP and SATMC lack the support for bitwise XOR operation were ignored due to uncertain results. The Dolev-Yao (DY) based threat model is adopted by AVISPA [20]. That is, a malicious adversary may edit, block, delete, or append the fake contents in the message during the interaction, besides intercepting the communication message. In the simulation, under the back-end related to OFMC, the aggregate execution time was recorded as 398 milliseconds, whereas the number of depth and visited nodes were 6 plies and 85 nodes, respectively. Using the back-end for CL-AtSe, one state was reported with the translation time as 0.17 sec. With respect to CL-AtSe and OFMC back-ends, it is clearly manifested in the simulation modeling report that our scheme BOD5-IOD is protected from both man-in-the-middle and replay attacks.

4.3. Experimental results using MIRACL

We measured the execution time of the employed cryptographic primitives in designing the proposed scheme by using the widely recognized Multi-precision Integer and Rational Arithmetic Cryptographic Library (MIRACL) [21]. The MIRACL is based upon C/C++ software library and is widely adopted by the researchers as "gold standard open-source SDK for ECC" to research cryptography. The two cases were considered for computing the execution time regarding cryptographic operations concerning the exchange of messages between DNi and GSPj:

Case I. The server-based resources to implement MIRACL are assumed with the following setting: Ubuntu 20.04.1 LTS 64-bit OS with 8GB RAM, Intel Core i7 with a CPU of 2.3 GHz. The readings for each cryptographic primitive were captured with 100 runs and recorded the maximum, minimum, and average timings in milliseconds.

Case II. The client-oriented platform regarding MIRACL was considered Raspberry PI 3B+ Rev 1.4 [22], having 64 bit CPU, 1GB RAM, and Ubuntu OS 20.04.1 LTS. The readings for each cryptographic operation were recorded with 100 runs and noted the minimum, maximum, and average timings in milliseconds.

4.4. Informal security analysis

In this section the informal security analysis for BOD5-IOD is presented.

4.4.1. Supports Mutual authentication

In the proposed scheme, unlike BSD2C-IOD, where only unilateral authentication was supported, the GSP_j and DNi mutually authenticate each other with the help of respective certificates and signatures [32-34, [40] 43-45]. In our scheme, the GSP_j authenticates DN_i on the basis of the comparison of $ACert_{DNi}\cdot G$ against the computation employing Pub_{CAj}, Pk_{CAj} and X_{DNi}. Similarly, the DN_i duly authenticates GSP_j by calculating X_{GSPj} and verifying the dynamic certificate as $ACert_{GSPj}.G$ against the computation using Pub_{GSPj}, Pub_{CAj}, Pk_{CAj} and X_{GSPj}. Hence, the BOD5-IOD ensures mutual authenticity for the involved participants.

4.4.2. Assured untraceability for DN_i

In the proposed scheme, unlike BSD2C-IOD, the DN_i remains untraceable [35-36][41]. This is because the DN_i, in the proposed scheme, submits pseudo-identity RID_{DNi} after encryption within the signature without being exposed in the public message. In this manner, the BOD5-IOD can achieve mutual authentication between DN_i and GSP_j, since the DN_i remains untraceable by an adversary having access to public messages.

4.4.3. Drone or GSPj impersonation attack

Our scheme supports mutual authentication to both participants since both participants verify the authenticities of one another by certificates and signatures. This property certifies that the adversary may not initiate DN_i and GSP_j impersonation attack following the BOD5-IOD protocol.

4.4.4. Drone physical capture attack

If the drone DN_i is physically captured by the adversary, it may recover the parameters RID_{DNi}, $Cert_{DNi}$, (k_{DNi}, Pk_{DNi}), Pk_{CAj} from the memory of DN_i [37-39, 42]. However, the adversary may not be able to launch a physical capture attack on drones, since the recovered parameters may not be able to compute the previous session keys, i.e. $SK_{DNi,GSPj}= SK_{GSPj,DNi} = h(X_{DNi}||X_{GSPj}|| RID_{DNi}||RID_{GSPj}||TS_1||TS_2)$ as established among the genuine participants.

5. Performance Evaluation Analysis

In this section, a comparative analysis is performed based on security functionalities, computational and communicational overheads among different schemes, including Tian et al. [23], Luo et al. [24], Li et al. [24], and BOD5-IOD [13]. The communication and computational costs for the mutual authentication phase of BOD5-IOD between DNi and GSPj is depicted in Table 4 and Table 6. We assume that the communication delay analysis for timestamp, a hash function (SHA-256), elliptic curve point multiplication, random integer, and identity take 32, 256, 320 (160+160), 160 and 160 bits, respectively. We also assume that a cryptosystem of ECC-based 160-bit key provides an equivalent level of security as that of an RSA-based 1024-bit key. In BOD5-IOD, the communication messages such as $Msg_1=\{AID_{DNi}, ADNi, ACert_{DNi}, Sig_{DNi}, TS1\}$, $Msg_2= \{ AID_{GSPj}, ACert_{GSPj}, B_{GSPj}, SKV_{GSPj,DNi}, TS2\}$ and $Msg_3= \{ ACKDNi,GSPj, TS3\}$ take 928-bits, 1024-bits and 288-bits, respectively. The analysis on communication delay for various schemes and BOD5-IOD is shown in Table 6. The communication cost for the proposed scheme is comparatively lower than [23-25]. However, it is equivalent to the communication cost of BSD2C-IOD as 2240 bits.

For the comparison of computational delay, we assume T_{me}, T_{bp}, T_{pa}, T_{pm} and T_h represent the execution time of modular exponentiation, bilinear pairing operation, elliptic curve-based point addition, elliptic curve-based point multiplication, and collision-resistant one-way hash function, respectively. In the contributed BOD5-IOD, the DN_i calculates the computational delay as $5T_h + 5T_{PM} + 2T_{PA}$, while the GSP_j computes the same as $5T_h + 7T_{PM} + 2T_{PA}$. The experimental findings are applied as shown in section VI for computing the execution times of various crypto-primitives by using MIRACL. We assume the execution delay for different crypto-primitives on Raspberry PI 3 as assumed in [13] for the drone embedded with multiple IoT sensors and smart devices. Likewise, we assume the execution time of employed crypto-primitives on the end of GSP server. Thereafter, on account of assumed computed delays for executing those primitives, a comparison between BOD5-IOD and the rest of the contemporary schemes has been drawn, as summarized in Table 4. According to this Table, our scheme takes a computational delay of *13.017ms,* which is quite low as compared to Luo et al. [24] and Li et al. [25] taking 32.393ms and 32.393, respectively. However, our scheme takes more computational cost than Tian et al. [23] and Bera et al. [13]. The Tian et al. scheme employs lightweight operations, is nonetheless vulnerable to session-specific temporary information attack, and does not support mutual authentication and perfect forward secrecy. Bera et al. [13] also take a comparatively low computational cost of *11.022ms* than our scheme, yet it is susceptible to GSP's impersonation attack, as well as lacking mutual authentication.

Moreover, the scheme [13] does not support anonymity for the drones. Table 5 exhibited the security-based functionality features for compared schemes and proposed models. Besides, Fig. 4 shows the graph for computational and security comparisons. Referring to this Table, the schemes [23] and [24] do not support mutual authentication, dynamic drone addition, and blockchain-oriented verification. Also, [23] is not immune to drone physical capture attack as well as session-specific Temporary Information Attack (SSTIA). The Tian et al. does not support perfect forward secrecy neither provides resistance against SSTIA. Table 5 demonstrates that BOD5-IOD has a conspicuous advantage over existing schemes in terms of functional features for security. Moreover, unlike BSD2C-IOD, the DNi remains untraceable in the proposed scheme, since drone DNi, submits pseudo-identity RID_{DNi} in encrypted form, which assures anonymity and untraceability for the drones. In addition, the computational and communication efficiencies in the proposed model are compared to previous studies, which are quantified as 34.4% and 23.3%, respectively. As per the results, the involvement of the blockchain center in the proposed scheme promotes immutability and traceability of transactions and assists in eliminating any trusted third party for secure data delivery and collection using decentralized management.

6. Conclusion

The contributed model serves as an improvement over Bera et al. scheme that intended to provide a blockchain-based authenticated key

Table 5
Functionality comparison.

	[24]	[25]	[23]	[13]	[Ours]
Resistance against RA	✓	✓	✓	✓	✓
Supports drone's anonymity	✓	✓	✓	×	✓
Immune to MIDMA	✓	✓	✓	✓	✓
Supports mutual authentication	×	×	×	×	✓
Immune to DIA	✓	✓	✓	✓	✓
Resists GIA	✓	✓	-	×	✓
Resists SSTIA	×	×	×	✓	✓
Immune to DPCA	×	×	✓	✓	✓
Supports FSV	✓	✓	×	✓	✓
Supports BOV	×	×	×	✓	✓
Supports DDA	×	×	✓	✓	✓
Achieves PFS	✓	✓	×	✓	✓

RA: Replay Attack, MIDMA: Man-in-the-Middle attack, DIA: Drone Impersonation Attack, GIA: GSPj impersonation attack, SSTIA: session-specific temporary information attack, DPCA: Drone Physical capture attack, PFS: Perfect Forward Secrecy, DDA: Dynamic drone Addition, BOV: Blockchain oriented verification, FSV: Formal Security Verification.

Table 4
Computational cost.

	[24]	[25]	[23]	[13]	[Ours]
DN_i	$1T_{BP}+1T_H$ $\approx 32.393ms$	$1T_{BP}+1T_H$ $\approx 32.393ms$	$8T_{ME}+9T_H$ $\approx 4.605ms$	$6T_H+4T_{PM}+1T_{PA}$ $\approx 11.022ms$	$5T_H+5T_{PM}+2T_{PA}$ $\approx 13.017ms$
GSP_j	$3T_{PM}+3T_{BP}+3T_H+1T_{PA} +1T_{ME} \approx 16.409ms$	$3T_{PM}+4T_{BP}+1T_H+2T_{PA} +1T_{ME} \approx 20.945ms$	-	$6T_H+6T_{PM}+2T_{PA}$ $\approx 4.378ms$	$5T_H+7T_{PM}+2T_{PA}$ $\approx 4.997ms$
Total delay	$\approx 48.802ms$	$\approx 53.338ms$	$\approx 4.605ms$	$\approx 15.4ms$	$\approx 18.014ms$

Table 6
Comparison of Communication cost (bits).

	Number of messages	Communication Cost (bits)
[24]	2	3040
[25]	2	3488
[23]	2	11712
[13]	3	2240
[Ours]	3	2240

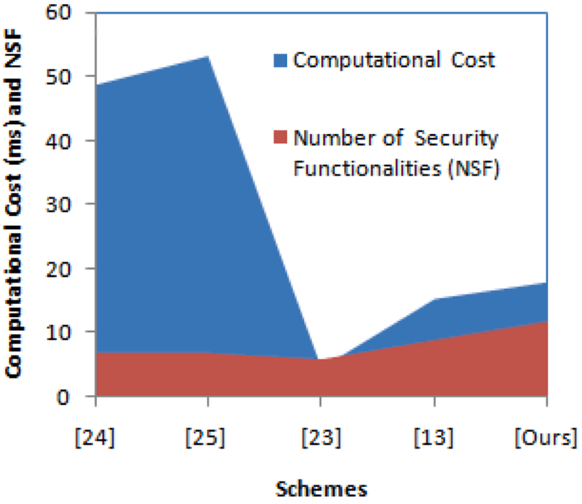


Fig. 4. Graph exhibiting computational delay and security.

agreement scheme for drones. The Bera et al., bearing serious problems in its model, was unable to support anonymity or untraceability for the drones. Furthermore, an adversary may initiate a Ground Station Server impersonation attack against the drones, which serves as a serious implication for the practicability of Bera et al. scheme. This paper proposed an enhanced blockchain-enabled authentication protocol BOD5-IOD for authenticating the registered drones in the system. The BOD5-IOD, other than supporting a robust access control mechanism between drones and GSS, also ensures safe transactions among all members in the IoD environment. The formal analysis and performance evaluation exhibit that our scheme supports all security requirements with computational and communication efficiencies. We shall work on bringing the computational cost further down by either eliminating the public key certificates or minimizing the elliptic curve point multiplication operations from the authentication process.

Authors' contributions

All authors contributed equally to this work.

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.

References

[1] Federal Aviation Administration. FAA Aerospace Forecast, 2016-2036. https://www.faa.gov/data_research/aviation/aerospace_forecasts/media/FY2016-36_FAA_Aerospace_Forecast.pdf [Accessed December, 2020].

[2] B. Li, Z. Fei, Y. Zhang, UAV Communications for 5G and Beyond: Recent Advances and Future Trends, IEEE Internet of Things J. 6 (2) (2019) 2241–2263.

[3] A.K. Das, M. Wazid, N. Kumar, A.V. Vasilakos, J.J.P.C. Rodrigues, Biometrics-Based Privacy-Preserving User Authentication Scheme for Cloud-Based Industrial Internet of Things Deployment, IEEE Internet of Things J. 5 (6) (2018) 4900–4913.

[4] S. Malani, J. Srinivas, A.K. Das, K. Srinathan, M. Jo, Certificate Based Anonymous Device Access Control Scheme for IoT Environment, IEEE Internet of Things J. 6 (6) (2019) 9762–9773.

[5] M. Wazid, A.K. Das, V. Odelu, N. Kumar, W. Susilo, Secure Remote User Authenticated Key Establishment Protocol for Smart Home Environment, IEEE Trans. Dependable Secure Comput. 17 (2) (2020) 391–406.

[6] S. Mandal, B. Bera, A.K. Sutrala, A.K. Das, K.R. Choo, Y. Park, Certificateless Signcryption-Based Three-Factor User Access Control Scheme for IoT Environment, IEEE Internet of Things J. 7 (4) (2020) 3184–3197.

[7] Jangirala, A.K. Das, A.V. Vasilakos, 'Designing secure lightweightblockchain-enabled RFID-based authentication protocol for supply chainsin 5G mobile edge computing environment, IEEE Trans. Ind. Informat. 16 (11) (Nov. 2020) 7081–7093.

[8] 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.

[9] A. Yazdinejad, R.M. Parizi, A. Dehghantanha, H. Karimipour, G. Srivastava, M. Aledhari, Enabling Drones in the Internet of Things with Decentralized Blockchain-based Security, IEEE Internet of Things J. (2020).

[10] D. Dolev, A. Yao, On the security of public key protocols, IEEE Trans. Inf. Theory 29 (2) (1983) 198–208.

[11] R. Canetti, H. Krawczyk, Universally Composable Notions of Key Exchange and Secure Channels, in: International Conference on the Theory and Applications of Cryptographic Techniques (EUROCRYPT'02), Amsterdam, The Netherlands, 2002, pp. 337–351.

[12] B.D. Deebak, F. Al-Turjman, A smart lightweight privacy preservation scheme for IoT-based UAV communication systems, Comput. Commun. 162 (2020) 102–117.

[13] 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.

[14] D. Johnson, A. Menezes, S. Vanstone, The Elliptic Curve Digital Signature Algorithm (ECDSA, Int. J. Inf. Secur. 1 (1) (2001) 36–63.

[15] M. Castro, B. Liskov, Practical Byzantine fault tolerance and proactive recovery, ACM Trans. Comput. Syst. 20 (4) (2002) 398–461.

[16] D. Magazzeni, P. McBurney, W. Nash, Validation and Verification of Smart Contracts: A Research Agenda, IEEE Computer 50 (9) (2017) 50–57.

[17] Y. Zhang, S. Kasahara, Y. Shen, X. Jiang, J. Wan, Smart contract based access control for the internet of things, IEEE Internet of Things J. 6 (2) (2019) 1594–1605.

[18] M. Abdalla, P.A. Fouque, D. Pointcheval, Password-based authenticated key exchange in the three-party setting, in: 8th International Workshop on Theory and Practice in Public Key Cryptography (PKC'05), Lecture Notes in Computer Science, Les Diablerets, Switzerland 3386, 2005, pp. 65–84.

[19] S.D. Kumar, R. Amin, V. Satyanarayana, R. Chaudhry, Blockchain-based secured event-information sharing protocol in internet of vehicles for smart cities, Computers & Electrical Engineering 86 (106719) (2020).

[20] AVISPA, "Automated Validation of Internet Security Protocols and Applications," 2019, http://www.avispa-project.org/. Accessed on October 2019.

[21] "MIRACL Cryptographic SDK: Multiprecision Integer and Rational Arithmetic Cryptographic Library," 2020, Accessed on April 2020. [Online]. Available: https://github.com/miracl/MIRACL.

[22] "Raspberry Pi 3 Model B+," 2020, Accessed on April 2020. [Online]. Available: https://www.raspberrypi.org/products/raspberry-pi-3-model-b-plus/.

[23] Y. Tian, J. Yuan, H. Song, Efficient privacy-preserving authentication framework for edge-assisted Internet of Drones, J. Inf. Secur. Appl. 48 (2019), 102354.

[24] M. Luo, Y. Luo, Y. Wan, Z. Wang, Secure and efficient access control scheme for wireless sensor networks in the cross-domain context of the IoT, Secur. Commun. Netw. (2018) 1–10, https://doi.org/10.1155/2018/6140978 [Online]. Available:.

[25] F. Li, Y. Han, C. Jin, Practical access control for sensor networks in the context of the Internet of Things, Comput. Commun. 89-90 (2016) 154–164.

[26] M. Wazid, B. Bera, A. Mitra, A.K. Das, R. Ali, Private blockchain-envisioned security framework for AI-enabled IoT-based drone-aided healthcare services, in: Proceedings of the 2nd ACM MobiCom Workshop on Drone Assisted Wireless Communications for 5G and Beyond, 2020, pp. 37–42.

[27] M. Wazid, A.K. Das, S. Shetty, J.J. Rodrigues, On the design of secure communication framework for blockchain-based internet of intelligent battlefield things environment, in: IEEE INFOCOM 2020-IEEE Conference on Computer Communications Workshops, 2020, pp. 888–893.

[28] ... T. Li, J. Ma, X. Ma, C. Gao, H. Wang, C. Ma, J. Zhang, Lightweight secure communication mechanism towards UAV networks, in: 2019 IEEE Globecom Workshops, 2019, pp. 1–6.

[29] V. Hassija, V. Saxena, V. Chamola, A blockchain-based framework for drone-mounted base stations in tactile internet environment, in: IEEE INFOCOM 2020-IEEE Conference on Computer Communications Workshops (INFOCOM WKSHPS), 2020, pp. 261–266.

[30] G. Cho, J. Cho, S. Hyun, H. Kim, Sentinel: A secure and efficient authentication framework for unmanned aerial vehicles, Appl. Sci. 10 (9) (2020).

[31] M. Bilal, S. Pack, Secure Distribution of Protected Content in Information-Centric Networking, IEEE Syst. J. 14 (2) (2020) 1921–1932, https://doi.org/10.1109/JSYST.2019.2931813.

[32] S.K. Dwivedi, R. Amin, S. Vollala, R. Chaudhry, Blockchain-based secured event-information sharing protocol in Internet of vehicles for smart cities, Comput. Electr. Eng. 86 (2020), 106719.

[33] S.K. Dwivedi, R. Amin, S. Vollala, Blockchain based secured information sharing protocol in supply chain management system with key distribution mechanism, J. Inf. Secur. Appl. 54 (2020), 102554.

[34] A. Irshad, M. Usman, S.A. Chaudhry, H. Naqvi, M. Shafiq, A provably secure and efficient authenticated key agreement scheme for energy internet-based vehicle-to-grid technology framework, IEEE Trans. Ind. Appl. 56 (4) (2020) 4425–4435, https://doi.org/10.1109/TIA.2020.2966160.

[35] S.A. Chaudhry, Correcting "PALK: Password-based anonymous lightweight key agreement framework for smart grid, Int. J. Electr. Power Energy Syst. 125 (2021), 106529.

[36] S.A. Chaudhry, M.S. Farash, N. Kumar, M.H. Alsharif, PFLUA-DIoT: a pairing free lightweight and unlinkable user access control scheme for distributed IoT environments, IEEE Syst. J. (2020), https://doi.org/10.1109/JSYST.2020.3036425.

[37] M. Bilal, SG. Kang, A secure key agreement protocol for dynamic group, Cluster Comput. 20 (2017) 2779–2792.

[38] A. Irshad, S.A. Chaudhry, O.A. Alomari, K. Yahya, N. Kumar, A novel pairing-free lightweight authentication protocol for mobile cloud computing framework, IEEE Syst. J. (2020), https://doi.org/10.1109/JSYST.2020.2998721.

[39] S.A. Chaudhry, K. Yahya, F. Al-Turjman, M.-H. Yang, A secure and reliable device access control scheme for IoT based sensor cloud systems, IEEE Access 8 (2020) 139244–139254, https://doi.org/10.1109/ACCESS.2020.3012121.

[40] S.A. Chaudhry, H. Alhakami, A. Baz, F. Al-Turjman, Securing demand response management: a certificate-based access control in smart grid edge computing infrastructure, IEEE Access 8 (2020) 101235–101243, https://doi.org/10.1109/ACCESS.2020.2996093.

[41] M. Rana, A. Shafiq, I. Altaf, M. Alazab, S.A.Chaudhry K.Mahmood, Y.B. Zikria, A secure and lightweight authentication scheme for next generation IoT infrastructure, Comput. Commun. 165 (2021) 85–96.

[42] Y. Wu, H.N. Dai, H. Wang, K.K.R Choo, Blockchain-based privacy preservation for 5g-enabled drone communications, IEEE Network 35 (1) (2021) 50–56.

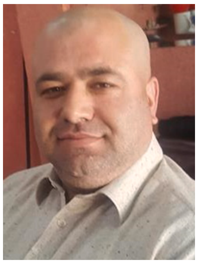
[43] L. Tan, H. Xiao, K. Yu, M. Aloqaily, Y. Jararweh, A blockchain-empowered crowdsourcing system for 5G-enabled smart cities, Comput. Standards Interfaces 76 (2021), 103517.

[44] M Bilal, S-G. Kang, An Authentication Protocol for Future Sensor Networks, Sensors 17 (5) (2017) 979.

[45] A.K. Sutrala, M.S. Obaidat, S. Saha, A.K. Das, M. Alazab, Y. Park, Authenticated key agreement scheme with user anonymity and untraceability for 5G-enabled softwarized industrial cyber-physical systems, IEEE Trans. Intell. Transp. Syst. (2021).

Azeem Irshad received master's degree from Arid Agriculture University, Rawalpindi, Pakistan. Then he completed his PhD from International Islamic University, Islamabad, Pakistan. He has authored more than 64 international journal and conference publications, including 33 SCI-E journal publications. His research work has been cited over 646 times with 12h-index and 14 i-10-index. He received Top Peer-Reviewer Award from Publons in 2018 with 126 verified reviews. He has served as a reviewer for more than 40 reputed journals including IEEE Systems Journal, IEEE Communications Magazine, IEEE TII, IEEE Consumer Electronics Magazine, IEEE Sensors Journal, IEEE TVT, IEEE IAS, Computer Networks, Information Sciences, CAEE, Cluster Computing, AIHC, JNCA and FGCS, notably. His research interests include strengthening of authenticated key agreements in Cloud-IoT, smart grid, pervasive edge computing, CPS, 5G networks, WSN, Ad hoc Networks, e-health clouds, SIP, and multi-server architectures.

Shehzad Ashraf Chaudhry received the master's and Ph.D. degrees (with Distinction) from International Islamic University Islamabad, Pakistan, in 2009 and 2016, respectively. He is currently working as an Associate Professor with the Department of Computer Engineering, Faculty of Engineering and Architecture, Istanbul Gelisim University, Istanbul, Turkey. He has authored over 120 scientific publications appeared in different international journals and proceedings, including more than 86 in SCI/E journals. With an H-index of 29 and an I-10 index 57, his work has been cited over 2420 times. He has also supervised over 40 graduate students in their research. His current research interests include lightweight cryptography, elliptic/hyper elliptic curve cryptography, multimedia security, E-payment systems, MANETs, SIP authentication, smart grid security, IP multimedia subsystem, and next generation networks. He occasionally writes on issues of higher education in Pakistan. Dr. Chaudhry was a recipient of the Gold Medal for achieving 4.0/4.0 CGPA in his Masters. Considering his research, Pakistan Council for Science and Technology granted him the Prestigious Research Productivity Award, while affirming him among Top Productive Computer Scientist in Pakistan. Recently, he is listed among Top 2% Computer Scientists across the world in Stanford University's report. He is also serving as guest editor for many WoS indexed journals and have served/serving as a TPC member of various international conferences. He is also an active reviewer of many WoS indexed journals.



Dr. Anwar Ghani is a faculty member at the Department of Computer Science & Software Engineering, International Islamic University Islamabad. He received his Doctorate in Computer Science and MS Computer Science from the Department of Computer Science & Software Engineering, International Islamic University Islamabad in 2016 and 2011. He received his BS in Computer Science from the University of Malakand K.P.K, Pakistan in 2007. Dr. Ghani worked as a Software Engineer in Bioman Technologies from 2007 to 20011. He was selected as an exchange student under – EURECA program in 2009 for VU University Amsterdam Netherland, and EXPERT program in 2011 for Masaryk University Czech Republic, funded EUROPEAN commission. His broad research interests include wireless sensor networks, Next Generation Networks, Information Security, Energy Efficient Collaborative Communication.

Muhammad Bilal received the B.Sc. degree in computer systems engineering from the University of Engineering and Technology, Peshawar, Pakistan, in 2008, the M.S. degree in computer engineering from the Chosun University, Gwangju, South Korea, in 2012, and the Ph.D. degree in information and communication network engineering from the School of Electronics and Telecommunications Research Institute (ETRI), Korea University of Science and Technology, in 2017. He was a Postdoctoral Research Fellow at Smart Quantum Communication Center, Korea University, Seoul, South Korea, in 2017/2018. Currently, he is an Assistant Professor with the Division of Computer and Electronic Systems Engineering, Hankuk University of Foreign Studies, Yongin, South Korea. His research interests include design and analysis of network protocols, network architecture, network security, IoT, named data networking, Blockchain, cryptology, and future Internet. . He is an editor of IEEE Future Directions Ethics and Policy in Technology Newsletter and IEEE Internet Policy Newsletter.