Blockchain-Envisioned Secure Data Delivery and Collection Scheme for 5G-Based IoT-Enabled Internet of Drones Environment

Basudeb Bera, Sourav Saha, *Student Member, IEEE*, Ashok Kumar Das, *Senior Member, IEEE*, Neeraj Kumar, *Senior Member, IEEE*, Pascal Lorenz, *Senior Member, IEEE*, and Mamoun Alazab, *Senior Member, IEEE*

***Abstract*—The Internet of Drones (IoD) is widely used in a wide range of applications from military to civilian applications from the past years. However, during communication either with the control room/ground station server(s) or moving access points in the sky, security and privacy is one the crucial issues which needs to be tackled efficiently. In this direction, blokchain technology can be one of the viable solutions due to the immutability and traceability of various transactions and decentralized nature. In this paper, we provide in-depth challenges and issues of applicability of blokchain in 5G-based Internet of Things (IoT)-enabled IoD environment. We propose and analyze a new blokchain based secure framework for data management among IoD communication entities. The proposed scheme has ability to resist several potential attacks that are essential in IoT-enabled IoD environment. A detailed comparative analysis exhibits that the proposed scheme offers better security and functionality requirements, and also provides less communication and computation overheads as compared to other related schemes.**

***Index Terms*—Internet of Drones (IoD), Internet of Things (IoT), blockchain, data delivery, data collection, security.**

I. Introduction

THE Internet of Drones (IoD) is treated as a "layered network control architecture". It is primarily designed to co-relate access of Unmanned Aerial Vehicles (UAVs) (which are also called *drones*) for controlling airspace and providing support to various navigation activities [1]. Due to increase of commercial drones applications, recent forecasts indicate that there will be a 100 USD billion market opportunity over the coming years based on the drones [2]. There are several applications of drones where the drones can be widely used, ranging from military, newsgathering (for example, videography and photography), security, agricultural and logistics deployments, surveillance, medicine to traffic-monitoring applications [3], [4]. In this connection, 5G plays a very important role in which drones are supposed to take part of a contributory role. For instance, there are several 5G use cases in which the drones are used, such as smart meter, smart agriculture, remote manufacturing, remote training, industrial application and control, smart city, and so on [3].

The Internet of Things (IoT) consists of several smart objects (called IoT smart devices) that are connected to the Internet. This is an emerging cutting-edge environment in which the smart devices can be utilized and also operated [5]. It is expected that around 25 billion smart devices will be part of this global community by the year 2020 [4]. The drones are equipped with various IoT-enabled smart devices, such as inbuilt sensors that can sense physical event including concentration of dangerous gases and temperature, and inbuilt cameras that are capable of taking images or capturing videos of the target spot. The drones can then transmit the sensed data to the drone box with the help of wireless technology (for example, WiFi). In this way, the drones contribute towards the creation of the IoT environment.

Figs. 1 and 2 show an overall architecture of 5G-based IoT-enabled IoD along with the revolution towards 5G technology from 1G technology [6]. In 1980 s, the "First Generation (1G)" of mobile communications was started and with a short span of time it become very popular. However, there were various disadvantages of 1G, such as poor voice quality, battery life, and security. In 1990 s, the "Second Generation (2G)" was introduced with a "Global System for Mobile communication (GSM)". The key features of 2G include digital switching, SMS services, encryption in voice transmission. However, there were various limitations in 2G technology, including low limited mobility, low data transmission, and limited hardware capability. Later, in early 2001, the "Third Generation (3G)" was evolved that supports various features, such as multimedia message, email, location tracking and maps, and enhanced security. However, some limitations were also present in 3G technology, because of costly equipments and expensiveness to build infrastructure implementation. In 2010, the "Fourth Generation (4G)" was

Manuscript received February 24, 2020; revised May 17, 2020; accepted June 3, 2020. Date of publication June 8, 2020; date of current version August 13, 2020. This work was supported by the Ripple Centre of Excellence Scheme, CoE in Blockchain (Sanction No. IIIT/R&D Office/Internal Projects/001/2019), IIIT Hyderabad, India. The review of this article was coordinated by Dr. Y. Zhang. *(Corresponding author: Pascal Lorenz.)*

Basudeb Bera, Sourav Saha, and Ashok Kumar Das are with the Center for Security, Theory and Algorithmic Research, International Institute of Information Technology, Hyderabad 500 032, India (e-mail: basudeb.bera@research.iiit.ac.in; sourav.saha@research.iiit.ac.in; ashok.das@iiit.ac.in).

Neeraj Kumar is with the Department of Computer Science and Engineering, Thapar Institute of Engineering and Technology, Patiala 147004, India, with the Department of Computer Science and Information Engineering, Asia University, Taichung City 41354, Taiwan, and also with the King Abdul Aziz University, Jeddah 21589, Saudi Arabia (e-mail: neeraj.kumar@thapar.edu).

Pascal Lorenz is with the University of Haute Alsace, 68008 Colmar, France (e-mail: lorenz@ieee.org).

Mamoun Alazab is with the College of Engineering, IT and Environment, Charles Darwin University, Casuarina, NT 0810, Australia (e-mail: alazab.m@ieee.org).

Digital Object Identifier 10.1109/TVT.2020.3000576

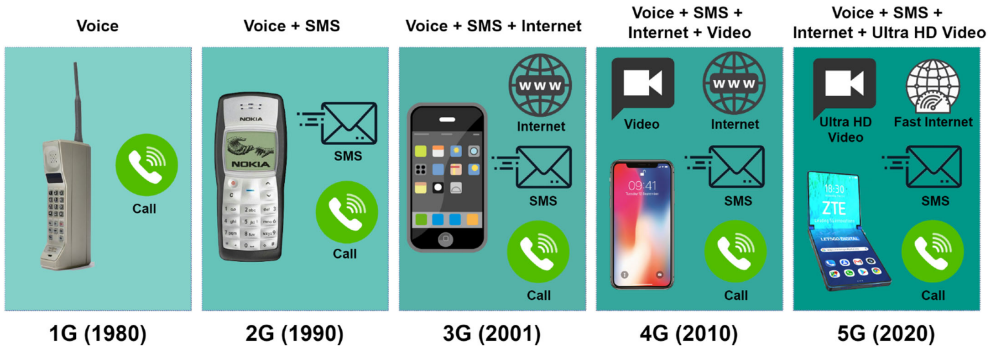

Fig. 1. Revolution of 5G technology.

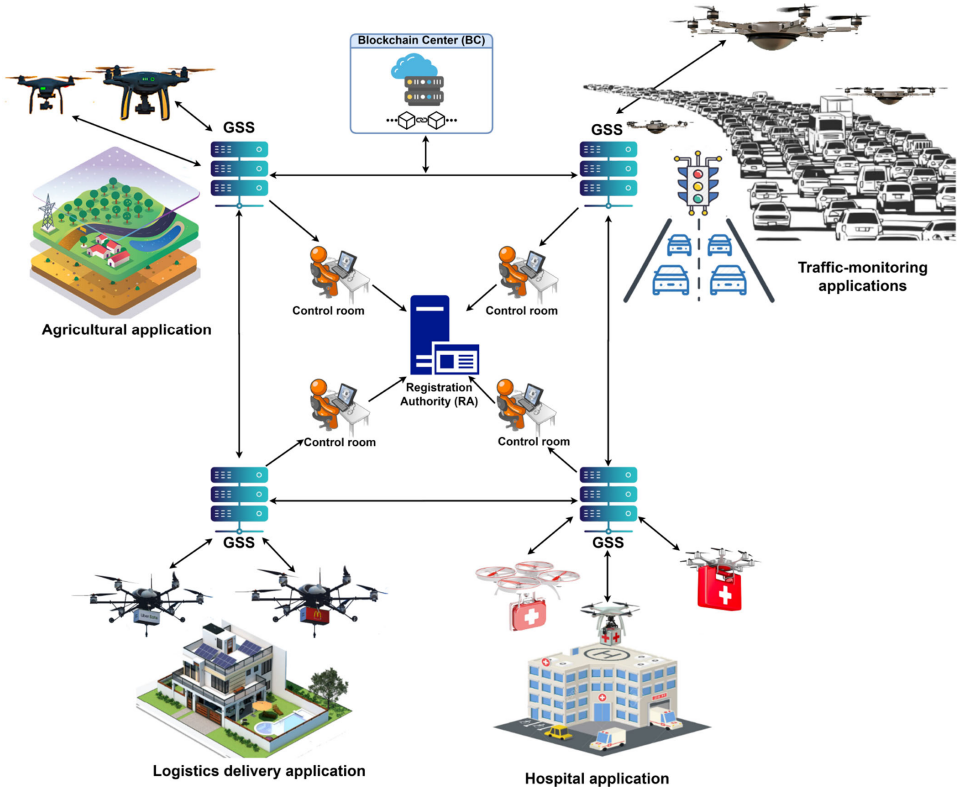

Fig. 2. Blockchain-envisioned 5G-enabled IoD environment.

introduced with the enhancement of 3G technology having some key features like higher data rate, reduced latency, HD video streaming and gaming, and Voice over LTE network (VoLTE). Finally, in 2020, the "Fifth Generation (5G)" comes to provide ultra fast internet and multimedia experience. The key features of 5G include higher security and reliable network, use beam forming and small cells, cloud infrastructure to improve efficiency, and easy maintenance. The 5G-enabled blockchain-envisioned IoD environment provided in Fig. 2 illustrates various communicating entities involved in the network, such as the drones, the ground station servers, control rooms, registration authority and the blockchain center. The detailed role of each individual entity is explained in Section III (see Fig. 3). It is worth noticing that the use of 5G celluar data helps in three major roles for connecting the drones: 1) "Managing Drone Traffic (UTM)," 2) "Beyond Visual Line of Sight (BVLOS)" flights, and 3) "sensor data transmission" [7]. UTM involves in the "regulation of the drone traffic itself and also its integration with the manned aviation" [7] because a large number of drones are already today deployed ranging from military to civilian applications. On the other side, various capabilities of BVLOS help a drone to make up far longer distances [8].

A. Threat Model

The drones in an IoD environment communicate over insecure wireless communication medium. In addition, the topology of an IoD environment is a kind of adhoc network. Therefore, there are opportunities to an adversary $\mathcal{A}$ to tamper with the data delivered to their respective GSS. The Dolev-Yao model (DY model) [9]

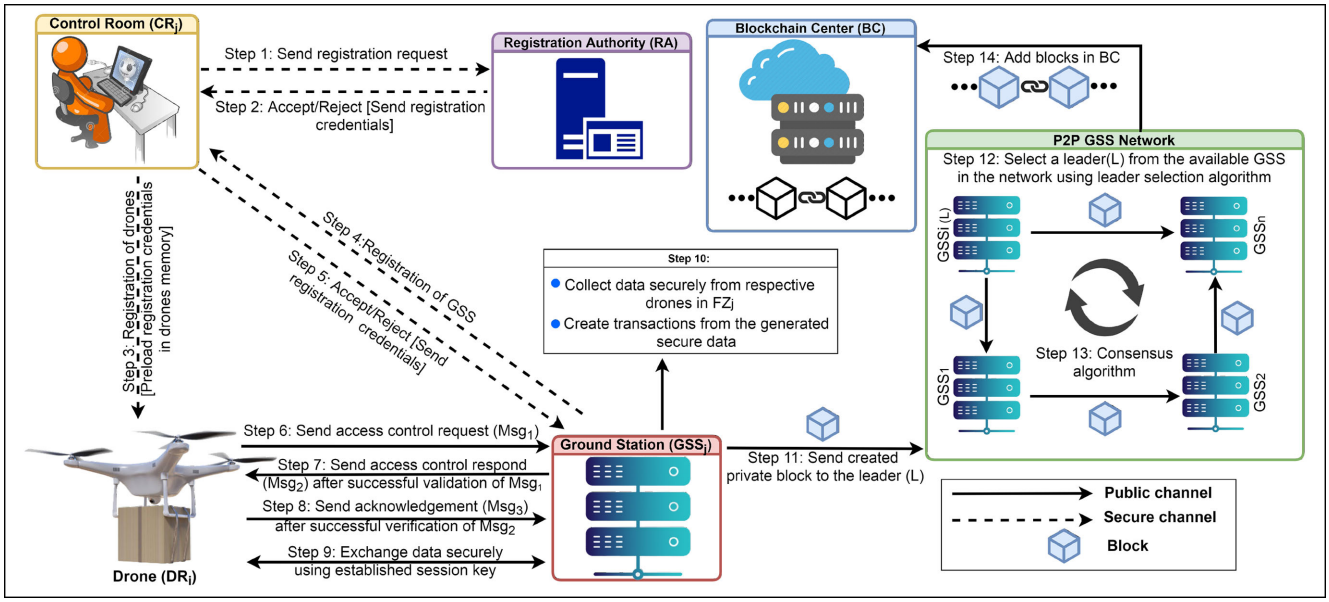

Fig. 3. Overall process in the proposed BSD2C-IoD.

is a widely used threat model in which $\mathcal{A}$ not only can eavesdrop communication among the drones and the GSSs, but also can modify, insert or delete the messages in between the communication. As a result, various types of attacks are possible in an IoD environment, such as replay, impersonation, man-in-the-middle, privileged-insider, Ephemeral Secret Leakage (ESL) and so on. The current *de facto* CK-adversary model [10] is considered more powerful threat model as compared to the standard DY threat model. Under the CK-adversary model, $\mathcal{A}$ has ability to tamper with the data as per the DY model, and in addition, $\mathcal{A}$ can also compromise the secret credentials, such as session keys, private keys and session state. Thus, the security of established session keys among the drones and the GSSs should depend on both the temporal and long-term secret credentials so that the session key will be compromised only when both secrets are compromised by the $\mathcal{A}$. Since a control room CR_j and the registration authority RA in an IoD environment (as shown in Fig. 3) are responsible for registering CR_j, and the GSS_j and its drones DR_i in each flying zone FZ_j, both RA and CR_j are treated as fully trusted nodes in the IoD environment. The GSSs are responsible for collecting the data securely from the drones and also to deliver the data to the drones, and creating the blocks of transactions to add them in the private blockchain in the Blockchain Center (BC). Therefore, the GSSs are also considered as trusted nodes. However, some drones may be physically captured and using the extracted credentials stored in those drones with the help of "power analysis attacks" [11], $\mathcal{A}$ can launch potentials attacks, such as impersonation attack on behalf of other non-compromised drones and ESL attack.

B. Research Contributions

The following are the main contributions towards this work:

- We first discuss the importance of secure data delivery and collection in 5G-enabled IoD environment.
- Next, we propose a novel blockchain-based secure data delivery and collection scheme, called BSD2C-IoD, which permits access control between the drones and their respective GSS in each flying zone FZ_j. The access control helps in establishing session keys among the drones and the GSSs for secure communication. The data delivery and collection process in BSD2C-IoD allows recording of all the transactions among the CR, GSS and drones in order to form private blocks by the GSS.
- A consensus-based algorithm is designed to help in verifying and adding the blocks by a leader selected among the GSSs in the peer-to-peer(P2P) GSS network in the BC.
- Detailed security analysis along with formal security verification using the widely-applied "Automated Validation of Internet Security Protocols and Applications (AVISPA)" software tool [12] are carried out on BSD2C-IoD to show its robustness against various potential attacks needed in a blockchain-based 5G-enabled IoD deployment.
- Various cryptographic primitives using the broadly-accepted "Multiprecision Integer and Rational Arithmetic Cryptographic Library (MIRACL)" [13] have been executed for measuring the execution time under both a server and a Raspberry PI 3 B+ Rev 1.3 [14].
- Finally, performance analysis of BSD2C-IoD has been carried out to show its effectiveness, specifically for resource-limited drones.

C. Paper Outline

In the next section, we discuss the need of secure data delivery and collection in IoD environment. Section III discusses the various phases related to the proposed scheme. While Section IV provides both formal and informal security analysis, Section V shows the formal security verification of the proposed scheme through simulation under the AVISPA software automated tool.

In Section VI, we discuss the experimental results of various cryptographic primitives using the MIRACL. Section VII gives the performance analysis of the proposed scheme. A brief discussion on the proposed BSD2C-IoD is provided in Section VIII. The paper is finally concluded in Section IX.

II. Secure Data Delivery and Collection: Need of the Hour

The currently deployed drone delivery system uses to deliver food, packages, medicine, emergency help in flooding areas, and so on [15]. Data (images or videos) collection, such as collection of traffic monitoring, crowd monitoring, surveillance in border, and fireplaces, the drones play a significant role. Drone delivery system reduces the packages delivery time, fuel, and energy consumption by using battery power as compared to gasoline power system vehicles. The courier services and delivery service provider(s) also jointly stared online retailer business (for example, Amazon and DHL have started delivery of items to their customers). Amazon sets up "Amazon Prime Air" to provide deliveries using the drones known as "octocopters". In recent years, a drone delivery service has been established in London due to people demands, which can permit to exchange packages weighing up to 500G. In addition, Germany's express delivery company, Deutsche Post (DHL) also use the drones, called "parcelcopters" for the emergency delivery (for instance, high-priority goods like medicines to remote areas).

A delivery system in the IoD environment was built upon the trust among the delivery service provider(s) and their customers [15]. A service provider believes that their customers will not repudiate the delivery confirmation after successfully delivered the items in the proper destination. The customers need to also trust that the service provider will not deliver an item without presence of them or delivered an item in front of the door is stolen. In such kind of cases, it is very hard to maintain the responsibility. Due to such reasons, security plays a very important role in the IoD environment. This leads to design a secure data delivery and collection mechanism with the help of deployed drones. However, we need to maintain several security requirements, such as confidentiality, authentication, access control, non-repudiation, availability, freshness, etc. Apart from these requirements, we also face various security challenges in an IoD environment including "remote hijacking of drone," "privacy," "replay and man-in-the middle attacks," "impersonation attack," "privileged-insider attack," "physical drone capture attack," etc. as in other networks [16], [17]. Thus, it is crucial to keep in mind that the designed security protocol should be resistant to such security challenges.

Wu and Fan [18] discussed various challenges along with "modeling," "design," and "analysis of high mobility communication systems". They observed that due to high mobility of the devices during communication, the following variants occur: a) "fast channel variation in the physical layer," b) "fast link variations in the link layer," and c) "fast topology variation in the network layer". However, in 5G communication systems, high mobility communications have been incorporated as an integral part of the communication standards. We can also adapt the strategy suggested in [19] as follows. Applying the "mixed integer linear programming," the optimal paths of drones that minimize the fuel consumption can be calculated by considering "collision avoidance," "no-fly zones," and "altitude constraints".

For secure data delivery between source and destination, the path identification protocol is the most critical component. Yaar *et al.* [20] designed a path identifier algorithm, called Pi, which is a "packet marking approach". In Pi, since a path fingerprint is included in each packet, it enables a victim to recognize the packets that traverse the same paths through the Internet on a "per packet basis, regardless of source IP address spoofing". Hence, in order to defend "Distributed Denial of Service (DDoS)" attacks, we apply the Pi algorithm for secure data delivery between source and destination.

III. BSD2C-IoD: Blockchain-Based Secure Data Delivery and Collection Scheme

In this section, we propose a new "blockchain-based secure data delivery and collection scheme in the 5G-envisioned IoT-enabled IoD environment (BSD2C-IoD)". BSD2C-IoD makes utilization of timestamps and random numbers to protect replay attack against an adversary. For this reason, all the network entities are supposed to be synchronized with their clocks, which is also a typical assumption applied in designing security protocols [21]–[25]. The list of symbols with their significance tabulated in Table I is used to explain and analyze the proposed BSD2C-IoD.

BSD2C-IoD contains several phases, namely "system initialization phase," "registration phase," "access control phase," "secure data delivery and collection phase," "block creation, verification and addition in blockchain center phase" and "dynamic drones addition phase". The overall process in the proposed BSD2C-IoD involving various phases has been elaborated in Fig. 3.

The detailed descriptions of the phases related to BSD2C-IoD are now discussed in the following.

A. Phase 1: System Initialization

In this phase, the trusted registration authority (RA) picks the system parameters as follows.

The RA first selects a "non-singular elliptic curve of the form $E_p(u, v) : y^2 = x^3 + ux + v \pmod p$ over a Galois field $GF(p)$," where p is a sufficiently large prime, $u, v \in Z_p^* = \{1, 2, \ldots, p-1\}$ are two constants such that $4u^3 + 27v^2 \neq 0 \pmod p$ holds, and a "point at infinity or zero point $\mathcal{O}$". The RA then picks a base point $G \in E_p(u, v)$ of order n_0 as large as p. The RA also picks a "collision resistant one-way cryptographic hash function" $h(\cdot)$ (for example, SHA-256 hashing algorithm [26]). In addition, the RA picks its its own identity ID_{RA}, master (private) key $r_{RA} \in Z_p^*$ and calculates its corresponding public key $Pub_{RA} = r_{RA} \cdot G$. The RA then keeps master (private) key r_{RA} as secret, whereas the parameters $\{E_p(u, v), G, h(\cdot), Pub_{RA}\}$ as public.

If $P = (x_P, y_P)$ and $Q = (x_Q, y_Q) \in E_p(u, v)$ be two elliptic curve points, $R = (x_R, y_R) = P + Q$, known as the "elliptic

TABLE I
SYMBOL AND THEIR SIGNIFICANCE

Symbol	Significance
$E_p(u,v)$	A non-singular elliptic curve of the form: "$y^2 = x^3 + ux + v \pmod p$ with $4u^3 + 27v^2 \neq 0 \pmod p$"
G	A base point in $E_p(u,v)$ whose order is n_0 as large as p
$k \cdot G$	"Elliptic curve point multiplication": $k \cdot G = G + G + \cdots + G\,(k$ times)
$X + Y$	"Elliptic curve point addition"; $X, Y \in E_p(u,v)$
RA	Registration authority (a trusted authority)
CR_j	j^{th} control room (a trusted authority)
GSS_j	j^{th} ground station server
DR_i	i^{th} drone
ID_{RA}	Real identity of RA
r_{RA}	Master (private) key of RA
Pub_{RA}	Public key of RA, where $Pub_{RA} = r_{RA} \cdot G$
ID_{CR_j}	Real identity of CR_j
r_{CR_j}	Random private key of CR_j
Pub_{CR_j}	Public key of CR_j, where $Pub_{CR_j} = r_{CR_j} \cdot G$
mk_{CR_j}	Random master key of CR_j
Pk_{CR_j}	Public key of CR_j, where $Pk_{CR_j} = mk_{CR_j} \cdot G$
$Cert_{CR_j}$	Certificates issued by the RA to CR_j
RTS_{CR_j}	Registration timestamps issued by the RA to CR_j
ID_{GSS_j}	Real identity of GSS_j
RID_{GSS_j}	Pseudo-identity of GSS_j
r_{GSS_j}	Random private key of GSS_j
Pub_{GSS_j}	Public key of GSS_j, where $Pub_{GSS_j} = r_{GSS_j} \cdot G$
k_{GSS_j}	Private key (decryption key) of GSS_j
Pk_{GSS_j}	Public key (encryption key) of GSS_j, where $Pk_{GSS_j} = k_{GSS_j} \cdot G$
$Cert_{GSS_j}$	Certificates issued by the CR_j to GSS_j
RTS_{GSS_j}	Registration timestamps issued by the CR_j to GSS_j
ID_{DR_i}	Real identity of DR_i
RID_{DR_i}	Pseudo-identity of DR_i
r_{DR_i}	Private certificate key of DR_i
Pub_{DR_i}	Public key of DR_i, where $Pub_{DR_i} = r_{DR_i} \cdot G$
k_{DR_i}	Private signature key of DR_i
Pk_{DR_i}	Public signature key of DR_i, where $Pk_{DR_i} = k_{DR_i} \cdot G$
$Cert_{DR_i}$	Certificates issued by the CR_j to DR_i
RTS_{DR_i}	Registration timestamps for DR_i issued by the CR_j
$\|\|$	Concatenation operation
TS	"Current timestamp"
ΔT	"Maximum transmission delay associated with a message"
$h(\cdot)$	"Collision-resistant cryptographic one-way hash function"
E_{Pk_X}/D_{k_X}	Public key encryption/decryption by an entity X

curve point addition," is computed as follows [27]:

$$x_R = (\mu^2 - x_P - x_Q) \pmod p,$$
$$y_R = (\mu(x_P - x_R) - y_P) \pmod p,$$
$$\text{where } \mu = \begin{cases} \frac{y_Q - y_P}{x_Q - x_P} \pmod p, & \text{if } P \neq -Q \\ \frac{3{x_P}^2 + u}{2y_P} \pmod p, & \text{if } P = Q. \end{cases}$$

The "elliptic curve scalar (point) multiplication" of an elliptic curve point $G \in E_p(u,v)$ is denoted by $k \cdot G$, where $k \in Z_p^* = \{1, 2, \ldots, p-1\}$ is a scalar. This operation can be further done efficiently using the "repeated point doubling and addition operations". For instance if $G = (x_G, y_G) \in E_p(u,v)$, then $17 \cdot G = 2 \cdot (2 \cdot (2 \cdot (2 \cdot G))) + G$ is calculated with the help of using "one point addition" and "four points doubling operations".

B. Phase 2: Registration

In this phase, the registration of the control room CR_j is done securely in offline mode by the trusted registration authority (RA). In addition, the registration of the ground service station GSS_j and its associated drones DR_i in a flying zone FZ_j are also executed in offline mode securely by their CR_j. The overall registration process in summarized in Fig. 4.

Next, the detailed registration process of CR_j, GSS_j and DR_i is given below.

1) CR_j Registration: The following steps are required for CR_j registration by the RA:

Step CRR_1: RA picks a unique identity ID_{CR_j} for each CR_j, chooses a random private key $r_{CR_j} \in Z_p^*$ and computes its respective public key $Pub_{CR_j} = r_{CR_j} \cdot G$, where $k \cdot G = G + G + \cdots + G(k\ times)$ is known as the "elliptic curve scalar (point) multiplication" and $k \in Z_p^*$. RA creates a certificate for each CR_j as $Cert_{CR_j} = r_{CR_j} + h(ID_{CR_j} \,||ID_{RA}\, ||Pub_{RA}\, ||Pub_{CR_j}\, ||RTS_{CR_j}) * r_{RA} \pmod p$, where RTS_{CR_j} is registration timestamp of the CR_j, and $*$ denotes the ordinary modular multiplication in Z_p^*. After that, the RA deletes r_{CR_j} from its database for security issue.

Step CRR_2: Next, the RA pre-loads the following information prior to deployment of each CR_j: $\{ID_{CR_j}, ID_{RA}, Cert_{CR_j}, Pub_{RA}, Pub_{CR_j}, E_p(u,v), h(\cdot), G\}$.

Step CRR_3: CR_j then picks a random master key $mk_{CR_j} \in Z_p^*$ and computes its corresponding public key $Pk_{CR_j} = mk_{CR_j} \cdot G$. Finally, the RA publishes $\{Pk_{CR_j}, Pub_{RA}, Pub_{CR_j}, E_p(u,v), h(\cdot), G\}$ as public information. Note that CR_j has the credentials $\{ID_{CR_j}, ID_{RA}, Cert_{CR_j}, mk_{CR_j}, Pk_{CR_j}, Pub_{RA}, Pub_{CR_j}, E_p(u,v), h(\cdot), G\}$.

2) GSS_j Registration: The registration of each GSS_j is performed by the CR_j with the following steps:

Step GR_1: CR_j selects a unique identity ID_{GSS_j} and computes its pseudo-identity $RID_{GSS_j} = h(ID_{GSS_j}\, ||RTS_{GSS_j}\, ||mk_{CR_j})$, where RTS_{GSS_j} is the registration timestamp of GSS_j. CR_j then picks a random private key $r_{GSS_j} \in Z_p^*$ and calculates its respective public key $Pub_{GSS_j} = r_{GSS_j} \cdot G$. Also, CR_j generates a certificate for GSS_j as $Cert_{GSS_j} = r_{GSS_j} + h(RID_{GSS_j}\, ||ID_{CR_j}\, ||Pub_{CR_j}\, ||Pub_{GSS_j}) * mk_{CR_j} \pmod p$.

Step GR_2: CR_j then stores RID_{GSS_j} and $Cert_{GSS_j}$ in its database corresponding to GSS_j, and makes Pub_{GSS_j} as public; and also deletes ID_{GSS_j} and r_{GSS_j} for security reason. Next, GSS_j selects its another private key (decryption key) $k_{GSS_j} \in Z_p^*$ and computes the corresponding public key (encryption key) $Pk_{GSS_j} = k_{GSS_j} \cdot G$.

Step GR_3: After that CR_j pre-loads the information $\{RID_{GSS_j}, ID_{CR_j}, Cert_{GSS_j}, Pub_{CR_j}, Pub_{GSS_j}, (k_{GSS_j}, Pk_{GSS_j}), Pk_{CR_j}, E_p(u,v), h(\cdot), G\}$ in GSS_j prior to its deployment. In addition, CR_j also stores Pk_{GSS_j} for each GSS_j in its database and publishes the information $\{Pub_{GSS_j}, Pk_{GSS_j}\}$ as public.

3) Drone DR_i Registration: Prior to the deployment of the drones DR_i in a particular application field, CR_j registers them with the following steps:

Step DR_1: CR_j selects a unique identity ID_{DR_i} and computes respective pseudo-identity $RID_{DR_i} = h(ID_{DR_i}\, ||ID_{CR_j}\, ||mk_{CR_j}\, ||RTS_{DR_i})$ for each drone DR_i, where RTS_{DR_i} is the registration timestamp.

Step DR_2: CR_j picks a private certificate key $r_{DR_i} \in Z_p^*$ and compute its corresponding public key $Pub_{DR_i} = r_{DR_i} \cdot G$, and

(a) Registration of a controller room (CR_j) for an IoT application

Registration Authority (RA)	Control room (CR_j)
• Select $E_p(u,v)$ over $GF(p)$ with base point G, $h(\cdot)$ • Select random private key $r_{CR_j} \in Z_p^*$ and compute $Pub_{CR_j} = r_{CR_j} \cdot G$ for CR_j • Pick its own master (private) key $r_{RA} \in Z_p^*$ and compute public key $Pub_{RA} = r_{RA} \cdot G$ • Select identity ID_{RA}, and identity ID_{CR_j} for each CR_j • Create certificate for each CR_j as $Cert_{CR_j} = r_{CR_j} + h(ID_{CR_j} \|ID_{RA} \|Pub_{RA} \|Pub_{CR_j} \|RTS_{CR_j}) * r_{RA} \pmod p$ • Publish $\{Pub_{RA}, Pub_{CR_j}, E_p(u,v), h(\cdot), G\}$ as public	• Store in each CR_j: $\{ID_{CR_j}, ID_{RA}, Cert_{CR_j}, Pub_{RA}, Pub_{CR_j}, E_p(u,v), h(\cdot), G\}$ • Pick random master key $mk_{CR_j} \in Z_p^*$ and compute public key $Pk_{CR_j} = mk_{CR_j} \cdot G$ • Store $\{mk_{CR_j}, Pk_{CR_j}\}$ in its database and make Pk_{CR_j} as public

(b) Registration of Ground Station Server GSS_j

Control Room CR_j	Ground Station Server GSS_j
• Pick identity ID_{GSS_j} and compute its pseudo-identity $RID_{GSS_j} = h(ID_{GSS_j} \|RTS_{GSS_j} \|mk_{CR_j})$ • Pick random private key $r_{GSS_j} \in Z_p^*$ and compute public key $Pub_{GSS_j} = r_{GSS_j} \cdot G$ • Create certificate for GSS_j as $Cert_{GSS_j} = r_{GSS_j} + h(RID_{GSS_j} \|ID_{CR_j} \|Pub_{CR_j} \|Pub_{GSS_j}) * mk_{CR_j} \pmod p$ • Store RID_{GSS_j} and $Cert_{GSS_j}$ in GSS_j • Make Pub_{GSS_j} as public; and delete ID_{GSS_j} and r_{GSS_j} • Store $\{Pk_{GSS_j}\}$ for each GSS_j in its database	• Select another private key (decryption key) $k_{GSS_j} \in Z_p^*$ and compute public key (encryption key) $Pk_{GSS_j} = k_{GSS_j} \cdot G$ • Pre-load $\{RID_{GSS_j}, ID_{CR_j}, Cert_{GSS_j}, Pub_{CR_j}, Pub_{GSS_j}, (k_{GSS_j}, Pk_{GSS_j}), Pk_{CR_j}, E_p(u,v), h(\cdot), G\}$ in GSS_j

(c) Registration of drones DR_i in a flying zone FZ_j

Control Room CR_j	Drone DR_i
• Pick identity ID_{DR_i} and pseudo-identity $RID_{DR_i} = h(ID_{DR_i} \|ID_{CR_j} \|mk_{CR_j} \|RTS_{DR_i})$ for each drone DR_i • Select private certificate key $r_{DR_i} \in Z_p^*$ and compute public key $Pub_{DR_i} = r_{DR_i} \cdot G$ for each drone DR_i • Select private signature key $k_{DR_i} \in Z_p^*$ and compute public signature key $Pk_{DR_i} = k_{DR_i} \cdot G$ for each drone DR_i • Generate certificate for each DR_i as $Cert_{DR_i} = r_{DR_i} + h(RID_{DR_i} \|Pub_{CR_j} \|Pub_{GSS_j} \|Pub_{DR_i}) * mk_{CR_j} \pmod p$ • Delete ID_{DR_i} and r_{DR_i} from its database	• Store $\{RID_{DR_i}, Cert_{DR_i}, (k_{DR_i}, Pk_{DR_i}), Pk_{CR_j}, E_p(u,v), h(\cdot), G\}$ in DR_i prior to deployment in flying zone FZ_j

Fig. 4. Summary of registration phase for control room, ground station server and drones.

also private signature key $k_{DR_i} \in Z_p^*$ and its public signature key $Pk_{DR_i} = k_{DR_i} \cdot G$ for each drone DR_i.

Step DR_3**:** CR_j creates a certificate for each DR_i as $Cert_{DR_i} = r_{DR_i} + h(RID_{DR_i} \|Pub_{CR_j} \|Pub_{GSS_j} \|Pub_{DR_i}) * mk_{CR_j} \pmod p$. After that CR_j deletes ID_{DR_i} and r_{DR_i} from its database, and stores the credentials $\{RID_{DR_i}, Cert_{DR_i}, Pub_{DR_i}, (k_{DR_i}, Pk_{DR_i}), Pk_{CR_j}, E_p(u,v), h(\cdot), G\}$ prior to its deployment in a particular flying zone FZ_j. In addition, CR_j also publishes the information $\{Pub_{DR_i}, Pk_{DR_i}\}$ as public.

C. Phase 3: Access Control

This phase is helpful in mutual authentication between a drone DR_i and its associated GSS_j in a flying zone FZ_j. Both DR_i and GSS_j use the pre-loaded information during registration phase. This phase utilizes certificate verification, signature generation/verification based on "elliptic curve cryptography (ECC)" technique and "one-way cryptographic hash function $h(\cdot)$". At the end of mutual authentication, both DR_i and GSS_j will agree on a common session key SK_{DR_i,GSS_j} $(= SK_{GSS_j,DR_i})$ for secret communication. The detailed description of this phase is elaborated as follows.

Step $ACDG_1$**:** DR_i selects a random number $r_1 \in Z_p^*$ and generates a current timestamp TS_1, and computes $r_1' = h(RID_{DR_i} \|r_1 \|Cert_{DR_i} \|k_{DR_i} \|TS_1)$, $A_{DR_i} = r_1' \cdot G$. Next, DR_i generates signature on r_1' as $Sig_{DR_i} = r_1' + h(Pk_{DR_i} \|RID_{DR_i} \|Pk_{CR_j} \|Pub_{GSS_j} \|A_{DR_i} \|TS_1) * k_{DR_i} \pmod p$. After that, DR_i prepares access control request message $Msg_1 = \{RID_{DR_i}, A_{DR_i}, Cert_{DR_i}, Sig_{DR_i}, TS_1\}$ and dispatches it to GSS_j via public channel.

Step $ACDG_2$: After receiving Msg_1 at time TS_1^*, GSS_j checks validity of TS_1 by $|TS_1^* - TS_1| < \Delta T$. If it is valid, GSS_j proceeds to validate the certificate by the condition: $Cert_{DR_i} \cdot G = Pub_{DR_i} + h(RID_{DR_i} \|Pub_{CR_j} \|Pub_{GSS_j} \|Pub_{DR_i}) \cdot Pk_{CR_j}$. If it is not valid, GSS_j rejects DR_i's request message; otherwise it further verifies the signature by $Sig_{DR_i} \cdot G = A_{DR_i} + h(Pk_{DR_i} \|RID_{DR_i} \|Pk_{CR_j} \|Pub_{GSS_j} \|A_{DR_i} \|TS_1) \cdot Pk_{DR_i}$. If the signature validation passes, GSS_j goes to next step.

Step $ACDG_3$**:** GSS_j generates a random number $r_2 \in Z_p^*$ and a current timestamp TS_2, and computes $r_2' = h(RID_{GSS_j} \|ID_{CR_j} \|r_2 \|Cert_{GSS_j} \|k_{GSS_j} \|TS_2)$, $B_{GSS_j} = r_2' \cdot G$. Next, GSS_j computes the Diffie-Hellman type key as $DHK_{GSS_j,DR_i} = r_2' \cdot A_{DR_i} (= (r_2' * r_1') \cdot G)$, and also the session key shared with DR_i as $SK_{GSS_j,DR_i} = h(DHK_{GSS_j,DR_i} \|RID_{DR_i} \|RID_{GSS_j} \|Pk_{DR_i} \|Pub_{GSS_j})$ and the session key verifier as $SKV_{GSS_j,DR_i} = h(SK_{GSS_j,DR_i} \|RID_{DR_i} \|RID_{GSS_j} \|B_{GSS_j} \|Cert_{GSS_j} \|TS_1 \|TS_2)$. GSS_j constructs the access control response message as $Msg_2 = \{RID_{GSS_j}, Cert_{GSS_j}, B_{GSS_j}, SKV_{GSS_j,DR_i}, TS_2\}$ and sends it to DR_i via open channel.

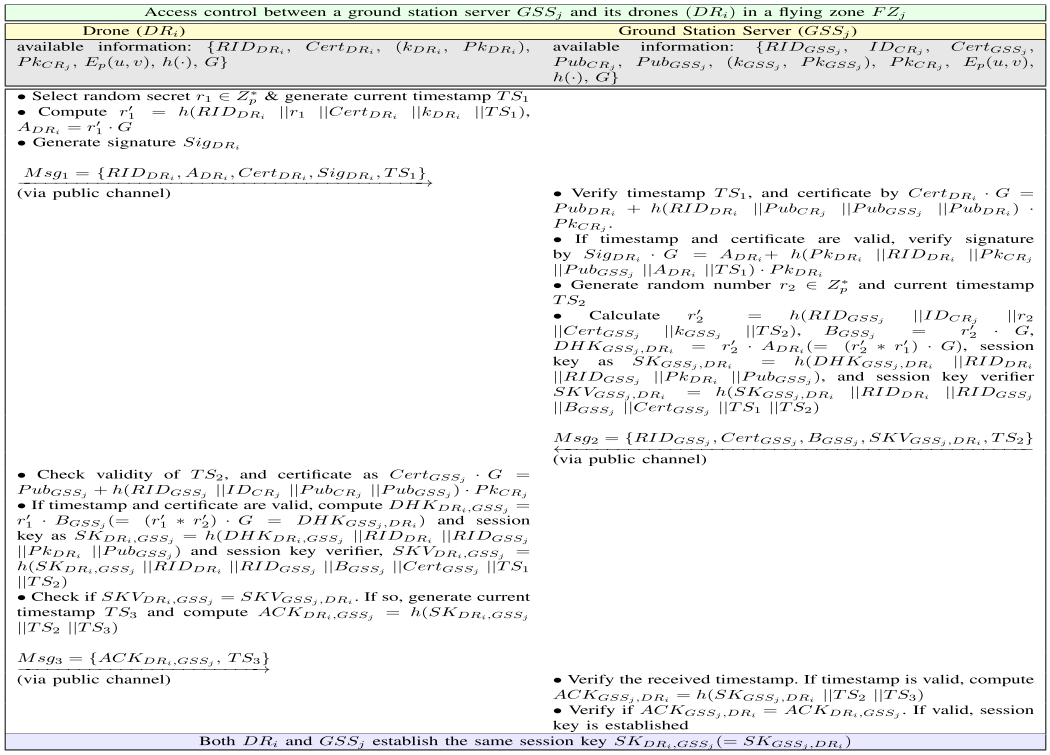

Fig. 5. Summary of access control phase.

Step $ACDG_4$**:** After receiving Msg_2 at time TS_2^*, DR_i checks validity of TS_2 by $|TS_2^* - TS_2| < \Delta T$. If it is valid, DR_i validates the certificate by the condition: $Cert_{GSS_j} \cdot G = Pub_{GSS_j} + h(RID_{GSS_j} \| ID_{CR_j} \| Pub_{CR_j} \| Pub_{GSS_j}) \cdot Pk_{CR_j}$. If the certificate is validated successfully, DR_i computes the Diffie-Hellman type key as $DHK_{DR_i,GSS_j} = r_1' \cdot B_{GSS_j} (= (r_1' * r_2') \cdot G = DHK_{GSS_j,DR_i})$, and derives the session key shared with GSS_j as $SK_{DR_i,GSS_j} = h(DHK_{DR_i,GSS_j} \| RID_{DR_i} \| RID_{GSS_j} \| Pk_{DR_i} \| Pub_{GSS_j})$ and the session key verifier as $SKV_{DR_i,GSS_j} = h(SK_{DR_i,GSS_j} \| RID_{DR_i} \| RID_{GSS_j} \| B_{GSS_j} \| Cert_{GSS_j} \| TS_1 \| TS_2)$. After that DR_i verifies if $SKV_{DR_i,GSS_j} = SKV_{GSS_j,DR_i}$, and if it is valid then DR_i generates current timestamp TS_3 and computes an acknowledgement $ACK_{DR_i,GSS_j} = h(SK_{DR_i,GSS_j} \| TS_2 \| TS_3)$. Finally, DR_i dispatches the acknowledge message $Msg_3 = \{ACK_{DR_i,GSS_j}, TS_3\}$ to GSS_j via public channel.

Step $ACDG_5$**:** After receiving Msg_3 at time TS_3^*, GSS_j checks validity of TS_3 by the condition: $|TS_3^* - TS_3| < \Delta T$. If the timestamp is valid, GSS_j computes $ACK_{GSS_j,DR_i} = h(SK_{GSS_j,DR_i} \| TS_2 \| TS_3)$ and verifies if $ACK_{GSS_j,DR_i} = ACK_{DR_i,GSS_j}$ is satisfied. If it is verified successfully, the session key $SK_{DR_i,GSS_j} (= SK_{GSS_j,DR_i})$ is established between DR_i and GSS_j for their secret communications.

This phase is also briefed in Fig. 5.

D. Phase 4: Secure Data Delivery and Collection

In this section, we discuss various data delivery and collection related transactions among CR_j, GSS_j and the drones DR_i in a flying zone FZ_j as shown in Fig. 6. We have the following types of transactions:

Type of Transaction	Description
$Tx_{CR-GSS-req}$	It represents a transaction between a control room (CR_j) and its GSS_j. It is the data delivery request from CR_j to GSS_j.
$Tx_{GSS-DR-req}$	It means a transaction between GSS_j and its drones DR_i. It denotes the data delivery request from GSS_j to DR_i in a particular flying zone FZ_j.
$Tx_{DR-GSS-res}$	It denotes a transaction between DR_i and its GSS_j. It is the data delivery/collection response from DR_i to GSS_j.
$Tx_{DR-GSS-data}$	It means a transaction between GSS_j and its drones DR_i. It represents the collection message from GSS_j to DR_i in a particular flying zone FZ_j.

Fig. 6. Details of various transactions Tx_i.

- The transaction $Tx_{CR-GSS-req}$ is between CR_j and its GSS_j, and it is related to data delivery request from CR_j to GSS_j which is sent securely by encrypting $Tx_{CR-GSS-req}$ with the public key Pk_{GSS_j} of the GSS_j. The encrypted $Tx_{CR-GSS-req}$ is decrypted by GSS_j using its own private key k_{GSS_j}.
- The transaction $Tx_{GSS-DR-req}$ denotes the data delivery request from GSS_j to DR_i, which is encrypted with the established session key SK_{DR_i,GSS_j} between GSS_j and DR_i. After decrypting this encrypted transaction with SK_{DR_i,GSS_j}, DR_i will deliver the items (for examples, goods, medicines, etc.) to the proper destination.
- The transaction $Tx_{DR-GSS-res}$ represents the data delivery/collection response from DR_i to GSS_j (for instance, e-receipt) that is encrypted with SK_{DR_i,GSS_j}.

Algorithm 1: Consensus for Block Verification and Addition in Blockchain.

Input: A block $Block_i$ as shown in Fig. 7 and $n_{f_{GSS}}$: the number of faulty GSS nodes in the P2P GSS network

Output: Commitment and addition of block $Block_i$ into blockchain after its successful validation

1: Assume L, say GSS_l, is selected as the leader and it wants to add $Block_i$ into blockchain
2: L generates current timestamp TS_{GSS_j} for each follower ground station server node GSS_j and performs voting process
3: L encrypts voting request $VotReq$ as $E_{Pk_{GSS_j}}(VotReq, TS_{GSS_j})$ and sends a message containing the same block, and $E_{Pk_{GSS_j}}(VotReq, TS_{GSS_j})$ to each follower node GSS_j, $(j = 1, 2, \cdots, n_{GSS}, \forall j \neq l)$, where $E(\cdot)$ and $D(\cdot)$ are the encryption and decryption functions, respectively
4: Assume the message from L is received by each follower GSS_j in the P2P GSS network at time $TS^*_{GSS_j}$
5: **for** each follower node GSS_j **do**
6: Decrypt the message as $(VotReq,' TS_{GSS_j}) = D_{k_{GSS_j}}[E_{Pk_{GSS_j}}(VotReq, TS_{GSS_j})]$
7: Verify timestamp, Merkle tree root, current block hash, and signature on received block $Block_i$
8: If all checks are verified successfully, send the voting reply $VotRep$ and block verification status $BVStatus$ as $\{E_{Pk_L}(VotReq,' VotRep, BVStatus)\}$ to L
9: **end for**
10: If $VCnt$ denotes the vote counter, initialize $VCnt \leftarrow 0$
11: **for** each received response message $\{E_{Pk_L}(VotReq,' VotRep, BVStatus)\}$ from the responded follower GSS_j **do**
12: Compute $(VotReq,' VotRep, BVStatus) = D_{k_L}[E_{Pk_L}(VotReq,' VotRep, BVStatus)]$
13: **if** $((VotReq' = VotReq)$ and $((VotRep = valid)$ and $(BVStatus = valid)))$ **then**
14: Set $VCnt = VCnt + 1$
15: **end if**
16: **end for**
17: **if** $(VCnt > 2.n_{f_{GSS}} + 1)$ **then**
18: Send the commit response to all follower nodes
19: Add block $Block_i$ to the blockchain
20: **end if**

- There may be other instances (for example, smart agriculture) where the deployed drones DR_i need to send the collected information in form of the transactions $Tx_{DR-GSS-data}$ to GSS_j securely using the key SK_{DR_i,GSS_j} (see steps 9–10 in Fig. 3).

E. Phase 5: Block Creation, Verification and Addition in Blockchain Center

In this phase, the GSS_j will form a block, say $Block_i$ using the transactions (shown in Fig. 6) that are available to GSS_j (as provided in Fig. 7).

Block Header	
Block Version	$BVer$
Previous Block Hash	PBH
Merkle Tree Root	MTR
Timestamp	TS
Creator of Block	CB_{ID} (Identity of one of the GSSs, say GSS_j in P2P GSS network)
Public key of Signer GSS_j	Pk_{GSS_j}
Block Payload (Encrypted Transactions)	
List of t_n Encrypted Transactions $\#i$ (Tx_i)	$\{E_{Pk_{GSS_j}}(Tx_i)\|i = 1, 2, \cdots, t_n\}$
Current Block Hash	$CBHash$
Signature on $CBHash$	$BSign = ECDSA.Sig_{k_{GSS}}(CBHash)$, where $ECDSA.Sig(\cdot)$ and $ECDSA.Ver(\cdot)$ represent ECDSA signature generation and verification algorithms, respectively

Fig. 7. Structure of a block $Block_i$ based on various transactions.

The block $Block_i$ created by GSS_j contains several encrypted transactions using the public key of the GSS_j. The signature on the block is generated by GSS_j with the help of the "elliptic curve digital signature algorithm (ECDSA)" [28]. The generated signature, Merkle tree, and current block hash root assist to achieve "immutability and transparency properties" of the block maintained in the blockchain.

Next, a leader, say L is selected by the leader selection algorithm in P2P GSS network containing n_{GSS} GSSs as in [29]. The block $Block_i$ is then passed to the leader L for the purpose of consensus in order to perform block verification along with addition in blockchain as shown in Algorithm 1. Note that we have applied the "Practical Byzantine Fault Tolerance (PBFT)" consensus algorithm [30]. The steps 11–14 in Fig. 3 explain the overall procedure of block creation, verification and addition of a block in the blockchain center.

Smart contract is considered as a "computer program or an agreement between two parties" that can be self-executed and self-verified digitally, and it can be also deployed without any human interventions [31], [32]. It verifies and validates the "correct execution of the transactions" in the system, and also meets the "legal contracts". It is worth noticing that the agreements between the parties are traceable, immutable as well as irreversible. It then helps a blockchain system to be more robust, secure, efficient and cost-effective. In our proposed BSD2C-IoD, a smart contract can be implemented in each GSS in order to verify the collected transactions from various entities and also the blocks formed by the GSS in the system. As a result, data modification attack is prevented and also data integrity is preserved by using smart contracts. Thus, the blockchain technology along with smart contracts can be used as "potential candidates for organizing secure interaction between autonomous agents" [33] in the proposed BSD2C-IoD.

F. Phase 6: Dynamic Drones Addition Phase

Sometimes, some drones may be malfunctioned or physically captured by an adversary. Therefore, some new drones may be added in the IoD environment.

TABLE II
QUERIES AND THEIR MOTIVES

Query	Motive
$Execute(\Upsilon_{DR_i}^{l_1}, \Upsilon_{GSS_j}^{l_2})$	$\mathcal{A}$ executes this query to eavesdrop the messages communicated between DR_i and GSS_j
$CorruptDrone(\Upsilon_{DR_i}^{l_1})$	$\mathcal{A}$ executes this query to pull out "the secret credentials stored in a compromised DR_i "
$Reveal(\Upsilon^l)$	$\mathcal{A}$ executes this query to disclose the session key $SK_{DR_i,GSS_j}(= SK_{GSS_j,DR_i})$ shared between Υ^l and its respective partner
$Test(\Upsilon^l)$	$\mathcal{A}$ executes this query to validate the revealed session key $SK_{DR_i,GSS_j}(= SK_{GSS_j,DR_i})$ by utilizing a "random outcome of a flipped unbiased coin, b"

Suppose a new drone DR_i^{new} needs to be added in a flying zone FZ_j. To perform this task, its control room CR_j first selects a unique identity $ID_{DR_i}^{new}$ and calculates respective pseudo-identity $RID_{DR_i}^{new} = h(ID_{DR_i}^{new} ||ID_{CR_j} ||mk_{CR_j} ||RTS_{DR_i}^{new})$ for DR_i^{new}, where $RTS_{DR_i}^{new}$ is the registration timestamp. Next, CR_j picks a private certificate key $r_{DR_i}^{new} \in Z_p^*$ and computes its corresponding public key $Pub_{DR_i}^{new} = r_{DR_i}^{new} \cdot G$, and also selects a private signature key $k_{DR_i}^{new} \in Z_p^*$ and its public signature key $Pk_{DR_i}^{new} = k_{DR_i}^{new} \cdot G$ for DR_i^{new}. After that CR_j creates a certificate for DR_i^{new} as $Cert_{DR_i}^{new} = r_{DR_i}^{new} + h(RID_{DR_i}^{new} ||Pub_{CR_j} ||Pub_{GSS_j} ||Pub_{DR_i}^{new}) * mk_{CR_j} \pmod p$. Finally, CR_j stores the information $\{RID_{DR_i}^{new}, Cert_{DR_i}^{new}, (k_{DR_i}^{new}, Pk_{DR_i}^{new}), Pk_{CR_j}, E_p(u,v), h(\cdot), G\}$ prior to DR_i^{new}'s deployment in the flying zone FZ_j. CR_j also erases $ID_{DR_i}^{new}$ and $r_{DR_i}^{new}$ from its database for security reason.

IV. SECURITY ANALYSIS

In this section, we exhibit that BSD2C-IoD can resist various potential attacks needed in an IoD environment through the use of both formal and informal security analysis.

A. Formal Security Analysis Under ROR Model

In our formal security analysis, we adopt the broad accepted "Real-Or-Random (ROR)" oracle model [34] to prove the session key between DR_i and GSS_j in the access control phase provided in Fig. 5 for the proposed BSD2C-IoD is secure against an adversary $\mathcal{A}$. We provide below the detailed description of the ROR model by semantic security defined in Definition 1 and session key security in Theorem 1. For achieving this task, $\mathcal{A}$ will execute queries that are tabulated in Table II. Furthermore, as reported in [35], access to a "collision-resistant one-way cryptographic hash function $h(\cdot)$" is supplied to all the engaged members including the adversary $\mathcal{A}$. In BSD2C-IoD, we model $h(\cdot)$ as a random oracle, say $Hash$.

Participants: In BSD2C-IoD, there are four participants, namely RA, CR_j, DR_i, and GSS_j. DR_i and GSS_j are involved in the access control phase to establish a session key and apart from that RA only involves during the registration and dynamic node (drone) addition phases. The symbols $\Upsilon_{DR_i}^{l_1}$ and $\Upsilon_{GSS_j}^{l_2}$ represent the l_1^{th} and l_2^{th} instances of DR_i and GSS_j, respectively. These instances are called the "random oracles".

Accepted state: An instance Υ^l goes to an "accepted state" when the last valid communicated message is received. After receiving all the communicated messages in a particular session, they are arranged in sequence and it will then establish the "session identification sid of Υ^l for the running session".

Partnering: Two instances (Υ^{l_1} and Υ^{l_2}) are partners to each other if they satisfy the following conditions:

- Υ^{l_1} and Υ^{l_2} are required to be in "accepted states".
- Υ^{l_1} and Υ^{l_2} are required to share the same sid and they need to "mutually authenticate each other".
- Υ^{l_1} and Υ^{l_2} are required to be "mutual partners of each other".

Freshness: An instance $\Upsilon_{DR_i}^{l_1}$ or $\Upsilon_{GSS_j}^{l_2}$ is known to be fresh when the established session key $SK_{DR_i,GSS_j}(= SK_{GSS_j,DR_i})$ between DR_i and GSS_j is not disclosed to $\mathcal{A}$ by executing the Reveal(Υ^l) query described in Table II.

We now define "semantic security" of our proposed BSD2C-IoD in Definition 1, which is helpful to prove Theorem 1.

Definition 1 (Semantic security): If $Adv_{\mathcal{A}}^{BSD2C-IoD}(t_{poly})$ denotes the "advantage of an adversary $\mathcal{A}$ running in polynomial time t_{poly}" in breaking the semantic security of the proposed BSD2C-IoD for computing the established session key SK_{DR_i,GSS_j} $(= SK_{GSS_j,DR_i})$ among a drone DR_i and an GSS_j in a particular session. Then,

$$Adv_{\mathcal{A}}^{BSD2C-IoD}(t_{poly}) = |2Pr[b' = b] - 1|,$$

where b and b' are respectively the "correct" and "guessed" bits.

Theorem 1: Let an adversary $\mathcal{A}$ running in polynomial time t_{poly} attempt to compute the session key SK_{DR_i,GSS_j} $(= SK_{GSS_j,DR_i})$ shared between a drone DR_i and an GSS_j for a particular session in the proposed protocol, BSD2C-IoD. If q_h, $|Hash|$, and $Adv_{\mathcal{A}}^{ECDDHP}(t_{poly})$ denote the number of "$Hash$ queries," the range space of "a one-way collision-resistant hash function $h(\cdot)$," and the advantage of breaking the "Elliptic Curve Decisional Diffie-Hellman Problem (ECDDHP)" respectively, then

$$Adv_{\mathcal{A}}^{BSD2C-IoD}(t_{poly}) \leq \frac{q_h^2}{|Hash|} + 2Adv_{\mathcal{A}}^{ECDDHP}(t_{poly}).$$

Proof: The proof of this theorem is similar to the proof that was done in [21]–[25], [35], [36]. An adversary $\mathcal{A}$ will play three games, say $Game_j^{\mathcal{A}}$ $(j = 0, 1, 2)$, against the proposed BSD2C-IoD. Let $Succ_{Game_j}^{\mathcal{A}}$ denote "an event that $\mathcal{A}$ can guess the random bit b in the game $Game_j^{\mathcal{A}}$ correctly". $\mathcal{A}$'s advantage to win $Game_j^{\mathcal{A}}$ in BSD2C-IoD is then defined by $Adv_{\mathcal{A},Game_j}^{BSD2C-IoD} = Pr[Succ_{Game_j}^{\mathcal{A}}]$. Each of the game $Game_j^{\mathcal{A}}$ is elaborated below.

$\mathbf{Game_0^{\mathcal{A}}}$: In this game, $\mathcal{A}$ plays against the proposed BSD2C-IoD by utilizing an "actual attack" with the help of the ROR model. $\mathcal{A}$ picks a random bit b prior to beginning of the initial game $\mathbf{Game_0^{\mathcal{A}}}$. The "semantic security" explained in Definition 1 gives the following:

$$Adv_{\mathcal{A}}^{BSD2C-IoD}(t_{poly}) = |2Adv_{\mathcal{A},Game_0}^{BSD2C-IoD} - 1|. \quad (1)$$

$\mathbf{Game_1^{\mathcal{A}}}$: This game corresponds to an *eavesdropping game*, where $\mathcal{A}$ performs $Execute$ query that is defined in Table II. By utilizing this query, $\mathcal{A}$ tries to derive the session key SK_{DR_i,GSS_j} $(= SK_{GSS_j,DR_i})$ from all intercepted communicated messages $Msg_1 = \{RID_{DR_i},$

A_{DR_i}, $Cert_{DR_i}$, Sig_{DR_i}, $TS_1\}$, $Msg_2 = \{RID_{GSS_j}$, $Cert_{GSS_j}$, B_{GSS_j}, SKV_{GSS_j,DR_i}, $TS_2\}$, and $Msg_3 = \{ACK_{DR_i,GSS_j}$, $TS_3\}$. Next, $\mathcal{A}$ executes $Reveal$ and $Test$ queries to validate the derived session key (whether it is a correct one or just a random key). The session key is $SK_{DR_i,GSS_j} = h(DHK_{DR_i,GSS_j} ||RID_{DR_i} ||RID_{GSS_j} ||Pk_{DR_i} ||Pub_{GSS_j})$, where $DHK_{DR_i,GSS_j} = r'_1 \cdot B_{GSS_j}$ $(= (r'_1 * r'_2) \cdot G = DHK_{GSS_j,DR_i})$. That implies that $SK_{DR_i,GSS_j} (= SK_{GSS_j,DR_i})$. As all the temporal credentials and long term secrets are protected by $h(\cdot)$, only hijacking the messages Msg_i $(i = 1, 2, 3)$ will not lead to increase the success probability to derive the session key. Therefore, both the games $Game_0^{\mathcal{A}}$ and $Game_1^{\mathcal{A}}$ become "indistinguishable under the eavesdropping attack". As a result, the following condition holds:

$$Adv_{\mathcal{A},Game_1}^{BSD2C-IoD} = Adv_{\mathcal{A},Game_0}^{BSD2C-IoD}. \quad (2)$$

$\mathbf{Game_2^{\mathcal{A}}}$: In this game, $\mathcal{A}$ plays an "active attack," where $\mathcal{A}$ simulates $Hash$ and $CorruptDrone$ queries. To derive the session key SK_{DR_i,GSS_j} $(= SK_{GSS_j,DR_i})$, the adversary $\mathcal{A}$ needs to derive $DHK_{DR_i,GSS_j} = r'_1 \cdot B_{GSS_j} (= (r'_1 * r'_2) \cdot G = DHK_{GSS_j,DR_i})$. Assume that $\mathcal{A}$ has the hijacked messages Msg_i $(i = 1, 2, 3)$. Therefore, he/she has the knowledge of $A_{DR_i} = r'_1 \cdot G = h(RID_{DR_i} ||r_1 ||Cert_{DR_i} ||k_{DR_i} ||TS_1) \cdot G$ and $B_{GSS_j} = r'_2 \cdot G = h(RID_{GSS_j} ||ID_{CR_j} ||r_2 ||Cert_{GSS_j} ||k_{GSS_j} ||TS_2) \cdot G$. Since $DHK_{DR_i,GSS_j} = (h(RID_{DR_i} ||r_1 ||Cert_{DR_i} ||k_{DR_i} ||TS_1) * h(RID_{GSS_j} ||ID_{CR_j} ||r_2 ||Cert_{GSS_j} ||k_{GSS_j} ||TS_2)) \cdot G = DHK_{GSS_j,DR_i})$, and all the secret credentials $\{k_{GSS_j}, k_{DR_i}, r_{GSS_j}, r_{DR_i}, mk_{CR_j}\}$ are protected by the hash function $h(\cdot)$, to derive DHK_{DR_i,GSS_j} the adversary $\mathcal{A}$ has to solve the ECDDHP. Furthermore, by utilizing the $CorruptDrone$ query, $\mathcal{A}$ will get the secret credential k_{DR_i}. Then, without having the knowledge of other secrets, such as $\{r_1, r_2, r_{GSS_j}, r_{DR_i}, mk_{CR_j}\}$, $\mathcal{A}$ will not be able to derive session key $SK_{DR_i,GSS_j} (= SK_{GSS_j,DR_i})$. Therefore, both the games $Game_1^{\mathcal{A}}$ and $Game_2^{\mathcal{A}}$ are indistinguishable if we exclude the simulation of $Hash$ and $CorruptDrone$ queries, and solving of ECDDHP is not a hard problem. The "hash collision resistant" property and the advantage of "solving ECDDHP" will lead to the following relation with the help of the birthday paradox:

$$|Adv_{\mathcal{A},Game_1}^{BSD2C-IoD} - Adv_{\mathcal{A},Game_2}^{BSD2C-IoD}| \leq \frac{q_h^2}{2|Hash|} + Adv_{\mathcal{A}}^{ECDDHP}(t_{poly}). \quad (3)$$

After executing all the games, $\mathcal{A}$ requires to correctly guess a bit b to win the game $Game_2^{\mathcal{A}}$. Therefore, we have,

$$Adv_{\mathcal{A},Game_2}^{BSD2C-IoD} = \frac{1}{2}. \quad (4)$$

Eq. (1) gives

$$\frac{1}{2}.Adv_{\mathcal{A}}^{BSD2C-IoD}(t_{poly}) = \left| Adv_{\mathcal{A},Game_0}^{BSD2C-IoD} - \frac{1}{2} \right|. \quad (5)$$

Solving Eqs. (2), (3) and (4), and applying the triangular inequality, the following derivation is produced:

$$\begin{aligned} &\frac{1}{2}.Adv_{\mathcal{A}}^{BSD2C-IoD}(t_{poly}) \\ &= |Adv_{\mathcal{A},Game_0}^{BSD2C-IoD} - Adv_{\mathcal{A},Game_2}^{BSD2C-IoD}| \\ &= |Adv_{\mathcal{A},Game_1}^{BSD2C-IoD} - Adv_{\mathcal{A},Game_2}^{BSD2C-IoD}| \\ &\leq \frac{q_h^2}{2|Hash|} + Adv_{\mathcal{A}}^{ECDDHP}(t_{poly}). \end{aligned} \quad (6)$$

Finally, if "a factor of 2" is multiplied on both sides of Eq. (6), we arrive to the final derivation:

$$Adv_{\mathcal{A}}^{BSD2C-IoD}(t_{poly}) \leq \frac{q_h^2}{|Hash|} + 2Adv_{\mathcal{A}}^{ECDDHP}(t_{poly}).$$

■

B. Information Security Analysis

We show in the following that BSD2C-IoD has potential to protect various attacks.

1) Replay Attack: In BSD2C-IoD, the current system timestamps and the random numbers are applied in the construction of messages Msg_1, Msg_2 and Msg_3. The old replayed messages by any adversary $\mathcal{A}$ simply can noticed by the particular recipients by checking the timestamps attached in the received messages. Thus, BSD2C-IoD is resilient against "replay attack".

2) Man-in-the-Middle (MiTM) Attack: In this attack, an adversary $\mathcal{A}$ may eavesdrop the access control request message Msg_1 between a drone DR_i and ground station server GSS_j from public channel, and try to tamper the request message to generate another valid message, say $Msg'_1 = \{RID_{DR_i}, A_{DR_i}{}', Cert_{DR_i}, Sig_{DR_i}{}', TS'_1\}$. For this issue, $\mathcal{A}$ can generate a random number $r_1^* \in Z_p^*$ and current timestamp TS'_1, and then calculates $r'_1 = h(RID_{DR_i} ||r_1^* ||Cert_{DR_i} ||k_{DR_i} ||TS'_1)$, $A'_{DR_i} = r'_1 \cdot G$, signature on r'_1 as $Sig'_{DR_i} = r'_1 + h(Pk_{DR_i} ||RID_{DR_i} ||Pk_{CR_j} ||Pub_{GSS_j} ||A_{DR_i} ||TS'_1) * k_{DR_i} \pmod p$. However, without knowledge of the credential k_{DR_i}, $\mathcal{A}$ can not create the valid signature Sig'_{DR_i} and r'_1. Similarly, for access control response message Msg_2, without knowing the credential k_{GSS_j}, $\mathcal{A}$ cannot also compute valid r'_2, $B_{GSS_j}, DHK_{GSS_j,DR_i}$ and SKV_{GSS_j,DR_i}. A similar situation arises for the message Msg_3, because $\mathcal{A}$ needs to generate session key for valid message construction, Hence, BSD2C-IoD is secure against MiTM attack.

3) Drone Impersonation Attack: In this attack, $\mathcal{A}$ tries to behave himself/herself as a legitimate entity on behalf of a registered drone DR_i. Next, $\mathcal{A}$ attempts to construct an authorized access control request message Msg_1^*. To execute this goal, $\mathcal{A}$ requires to pick random number $r_1^* \in Z_p^*$ and current timestamp TS_1^*. After that $\mathcal{A}$ requires computation of $r'_1 = h(RID_{DR_i} ||r_1^* ||Cert_{DR_i} ||k_{DR_i} ||TS_1^*)$ and $A_{DR_i} = r'_1 \cdot G$. Without knowledge of the master secret key k_{DR_i}, it is quite computationally hard to calculate valid r'_1 and A_{DR_i}. Therefore, "drone impersonation attack" is resisted in BSD2C-IoD.

4) GSS Impersonation Attack: Assume an adversary $\mathcal{A}$ behaves like a legalized (GSS_j), and try to build a legitimate access

control response message Msg_2^*. To accomplish this task, $\mathcal{A}$ can initially picks random number $r_2^* \in Z_p^*$ and a timestamp TS_2^*. However, computation of $r_2' = h(RID_{GSS_j} || ID_{CR_j} || r_2^* || Cert_{GSS_j} || k_{GSS_j} || TS_2^*)$, B_{GSS_j}, and session key verifier SKV_{GSS_j,DR_i} is computationally hard for $\mathcal{A}$, because without the master secret key k_{GSS_j} $\mathcal{A}$ can not calculate valid r_2', B_{GSS_j} and SKV_{GSS_j,DR_i}. GSS impersonation attack is then protected in BSD2C-IoD.

5) Privileged-Insider Attack: In the registration phase, neither a drone nor an GSS sends registration details to the trusted control room (CR). Instead, the CR creates all the credentials for every GSS along with drones prior to their deployment. This restriction provide to the CR being a "privileged-insider attacker" to get all the credential which are stored in GSS's and DR's memory. Thus, the privileged-insider attack is also resisted in BSD2C-IoD.

6) Physical Drone Capture Attack: Due to an antagonistic environment, an adversary $\mathcal{A}$ may physically capture some of the drones. Then, $\mathcal{A}$ can extract all the stored credentials $\{RID_{DR_i}, Cert_{DR_i}, (k_{DR_i}, Pk_{DR_i}), Pk_{CR_j}, E_p(u, v), h(\cdot), G\}$ from a compromised drone DR_i utilizing the "power analysis attacks" [11]. Since the stored secret information $\{RID_{DR_i}, Cert_{DR_i}, (k_{DR_i}, Pk_{DR_i})\}$ in each DR_i are distinct and unique from the stored information in other GSSs and the drones. Hence compromise of these credentials do not effect in establishing of the session keys between the other non-compromised GSSs and the drones. This circumstance is called "unconditionally secure against drone capture attack". BSD2C-IoD is resisted by "physical drone capture attack".

7) Ephemeral Secret Leakage (ESL) Attack: During access control phase between a drone DR_i and the GSS_j, DR_i computes the session key SK_{DR_i,GSS_j} shared with its associated GSS_j, where $SK_{DR_i,GSS_j} = h(DHK_{DR_i,GSS_j} || RID_{DR_i} || RID_{GSS_j} || Pk_{DR_i} || Pub_{GSS_j})$, where $DHK_{DR_i,GSS_j} = DHK_{GSS_j,DR_i}$. Since to compute DHK_{DR_i,GSS_j}, random nonces and timestamps are essential, each session key is also distinct in every session. It is worth noticing that the constructed session key is the composition of the pair "session-specific (ephemeral) credentials" (known as "short term secrets") and the "long-term secrets" (master secret keys). The session key SK_{GSS_j,DR_i} can only be disclosed only when $\mathcal{A}$ can compromise both the ephemeral and long-term secrets. Due to distinct session keys, even if a session key is revealed for a specific session, it will not lead to compromise other session keys over previous and future sessions. As a result, BSD2C-IoD is secure against "session-temporary information attack" and it also support the "perfect forward and backward secrecy" feature. Therefore, ESL attack is also then protected in BSD2C-IoD.

8) Block Verification in Blockchain: In BSD2C-IoD, the block addition in the blockchain by an GSS in the P2P GSS network is performed by the PBFT consensus algorithm [30]. Suppose a verifier **Ver** wish to verify a block $Block_i$ as shown in Fig. 7. To reach this goal, **Ver** requires to calculate the "Merkle tree root" on the all encrypted transactions in $Block_i$ and current block hash on $Block_i$. If any one of them mismatch with the previously saved values, **Ver** then discards the $Block_i$. Otherwise, **Ver** further moves to verify the block signature $BSign$ utilizing the "ECDSA signature verification algorithm" [28]. Thus, the block verification passes a three-level verification procedure in order to verify a block which is added in the blockchain.

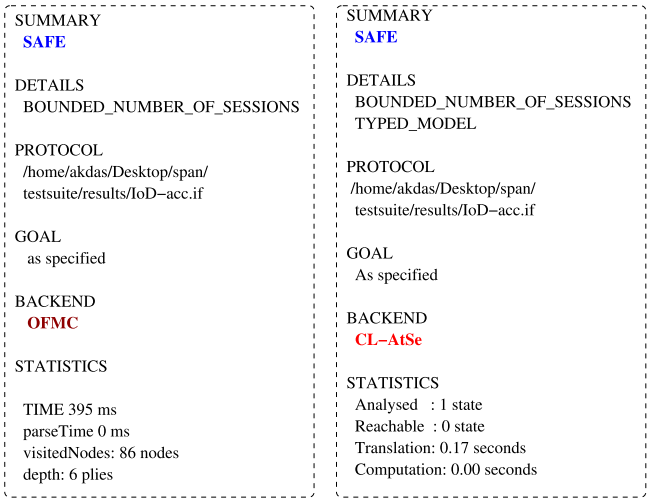

Fig. 8. Simulation results of BSD2C-IoD under OFMC and CL-AtSe backends.

V. Formal Security Verification Using AVISPA: Simulation Study

AVISPA [12] is a "push-button tool for the automated validation of Internet security protocols and applications". It offers a "modular and expressive formal language for specifying protocols along with their security goals". It also unites different back-ends that work a heterogeneity of state-of-the-art automatic analysis mechanisms. The four back-ends are implemented in AVISPA: a) "On-the-fly mode-checker (OFMC)," b) "Constraint-logic-based Attack Searcher (CL-AtSe)," c) "SAT-based Model Checker (SATMC)" and d) "Tree Automata based on Automatic Approximations for the Analysis of Security Protocols (TA4SP)". A security protocol requires to be implemented using the "High-Level Protocol Specification Language (HLPSL)" of AVISPA. The detailed descriptions of AVISPA and its HLPSL implementation can be further found in [12].

The simulation of the proposed BSD2C-IoD scheme under the "Security Protocol ANimator for AVISPA (SPAN)" tool [37] has been performed for formal security verification. The detailed simulation results are provided in Fig. 8 under OFMC and CL-AtSe backends only. Since other two backs, namely SATMC and TA4SP, do not currently support "bitwise XOR operations," we have ignored the simulation results reported under these backends as the results were "inconclusive". It is worth noticing that AVISPA implements the "Dolev-Yao threat model (DY model)" [9]. Thus, an intruder (i) not only can intercept the communicated messages, but can also modify, delete, or even insert the malicious message contents in between the communication.

For the "replay attack checking," the back-ends (OFMC and CL-AtSe) verify if the legal agents can execute a specified protocol by means of executing a search of an intruder having

TABLE III
EXECUTION TIME (IN MILLISECONDS) ON A SERVER FOR CRYPTOGRAPHIC PRIMITIVES USING MIRACL

Primitive	Max. time (ms)	Min. time (ms)	Average time (ms)
T_h	0.149	0.024	0.055
T_e	0.248	0.046	0.072
T_{ecm}	2.998	0.284	0.674
T_{eca}	0.002	0.001	0.002
T_{bp}	7.951	4.495	4.716

passive nature. Next, the back-ends provides the intruder about the knowledge of the normal sessions between the authorized agents. On the other side, for the "Dolev-Yao (DY) model check," the back-ends verify if any "man-in-the-middle attack" may be performed by the intruder or not. Under OFMC back-end, the total execution time was 395 milliseconds, while the number of visited nodes and depth were 86 nodes and 6 plies, respectively. Under CL-AtSe backend, one state was analyzed with the translation time of 0.17 seconds. Under both the OFMC and CL-AtSe backends, the simulation results demonstrated in Fig. 8 clearly show that the proposed BSD2C-IoD is safe against both "replay" and "man-in-the-middle" attacks.

VI. EXPERIMENTAL RESULTS USING MIRACL

In this section, various cryptographic primitives using the broadly-accepted "Multiprecision Integer and Rational Arithmetic Cryptographic Library (MIRACL)" [13] have been executed for measuring the execution time. MIRACL is a "C/C++ based programming software library" that has been already accepted by the cryptographers as the "gold standard open source SDK for elliptic curve cryptography (ECC)".

Let T_e, T_{ecm}, T_{eca}, T_h, and T_{bp} denote the time needed for "modular exponentiation," "elliptic curve point multiplication," "elliptic curve point addition," "one-way hash function using SHA-256 hashing algorithm," and "bilinear pairing operation," respectively. A non-singular elliptic curve of the form: "$y^2 = x^3 + ux + v \pmod p$" (see Table I) is considered for the elliptic curve point addition and multiplication.

We consider the following two cases for computing the execution time needed for various cryptographic primitives under a server and a drone/smart device:

- **Case 1:** The first platform considered for MIRACL here for a server with the setting as follows: Ubuntu 18.04.4 LTS, with memory: 7.7 GiB, processor: Intel Core i7-8565 U CPU @ 1.80 GHz × 8, OS type: 64-bit and disk: 966.1 GB. The experiments on each cryptographic primitive are performed for 100 runs. After that we calculate the maximum, minimum and average time (in milliseconds) for each cryptographic primitive from these 100 runs. The experimental results are reported in Table III.
- **Case 2:** The second platform that we have considered for MIRACL is as follows: Raspberry PI 3 B+ Rev 1.3, with CPU: 64-bit, Processor: 1.4 GHz Quad-core, 4 cores, Memory (RAM): 1 GB, and OS: Ubuntu 20.04 LTS, 64-bit [14]. The experiments on each cryptographic primitive are also executed for 100 runs. We then calculate the maximum, minimum and average run-time (in milliseconds) for each cryptographic primitive from these 100 runs, and the experimental results are tabulated in Table IV.

TABLE IV
EXECUTION TIME (IN MILLISECONDS) ON A RASPBERRY PI 3 FOR CRYPTOGRAPHIC PRIMITIVES USING MIRACL

Primitive	Max. time (ms)	Min. time (ms)	Average time (ms)
T_h	0.643	0.274	0.309
T_e	0.493	0.178	0.228
T_{ecm}	4.532	2.206	2.288
T_{eca}	0.021	0.015	0.016
T_{bp}	32.79	27.606	32.084

TABLE V
COMPARATIVE STUDY ON COMMUNICATION COSTS

Scheme	No. of messages	Total cost (in bits)	Gain
Luo et al. [38]	2	3040	36%
Li et al. [39]	2	3488	56%
Tian et al. [40]	2	$384s$ + 11712	509%
BSD2C-IoD	3	2240	–

Note: s: "no. of pseudonyms of an UAV (drone) in Tian *et al.*'s scheme" [40] ($s = 5$).

VII. COMPARATIVE ANALYSIS

In this section, we provide a comparative study on security and functionality features, communication and computation overheads among the proposed BSD2C-IoD and other related schemes, such as the schemes of Li *et al.* [39], Luo *et al.* [38], and Tian *et al.* [40].

A. Communication Costs Comparison

We contemplate the computation cost and communication cost for the access control phase of BSD2C-IoD between DR_i and GSS_j as shown in Fig. 5. For communication cost analysis, it is assumed that "identity," "random number," "elliptic curve point" $P = (P_x, P_y) \in E_p(u, v)$ where x and y coordinates of P are P_x and P_y respectively, hash output (here SHA-256 hashing algorithm), and timestamp are 160, 160, (160 + 160) = 320, 256 and 32 bits, respectively. It is assumed that 160-bit ECC provides the same security level as that for 1024-bit RSA cryptosystem. In BSD2C-IoD, the three messages $Msg_1 = \{RID_{DR_i}, A_{DR_i}, Cert_{DR_i}, Sig_{DR_i}, TS_1\}$, $Msg_2 = \{RID_{GSS_j}, Cert_{GSS_j}, B_{GSS_j}, SKV_{GSS_j,DR_i}, TS_2\}$, and $Msg_3 = \{ACK_{DR_i,GSS_j}, TS_3\}$ demand (256 + 320 + 160 + 160 + 32) = 928 bits and (256 + 160 + 320 + 256 + 32) = 1024 bits, and (256 + 32) = 288 bits, which altogether need 2240 bits. The comparative study on communication costs among BSD2C-IoD and other schemes provided in Table V. Our proposed BSD2C-IoD achieves 36%, 56% and 509% gains over the schemes of Luo *et al.* [38], Li *et al.* [39] and Tian *et al.* [40], respectively. Thus, BSD2C-IoD requires significantly less communication cost as compared to that for the existing schemes [38]–[40].

B. Computation Costs Comparison

We assume that $T_h, T_{ecm}, T_{eca}, T_{bp}$ and T_e denote for the time required to execute a "one-way cryptographic hash function," an "elliptic curve point multiplication" and an "elliptic curve point

TABLE VI
COMPARATIVE STUDY ON COMPUTATION COSTS

Scheme	Smart device/drone cost	Gain for a smart device/drone	GSS/server cost	Gain for GSS/server
Luo *et al.* [38]	$T_{bp} + T_h$ ≈ 32.393 ms	194%	$3T_{ecm} + 3T_{bp} + 3T_h + T_{eca} + T_e$ ≈ 16.409 ms	275%
Li *et al.* [39]	$T_{bp} + T_h$ ≈ 32.393 ms	194%	$3T_{ecm} + 4T_{bp} + T_h + 2T_{eca}$ ≈ 20.945 ms	378%
Tian et al. [40]	$8T_e + 9T_h$ ≈ 4.605 ms	–	–	–
BSD2C-IoD	$6T_h + 4T_{ecm} + T_{eca}$ ≈ 11.022 ms	–	$6T_h + 6T_{ecm} + 2T_{eca}$ ≈ 4.378 ms	–

TABLE VII
COMPARATIVE STUDY ON FUNCTIONALITY AND SECURITY ATTRIBUTES

Attribute (A)	Luo *et al.* [38]	Li *et al.* [39]	Tian et al. [40]	BACS-IoD
FSA_1	✓	✓	✓	✓
FSA_2	✓	✓	✓	✓
FSA_3	×	×	×	✓
FSA_4	✓	✓	✓	✓
FSA_5	✓	✓	✓	✓
FSA_6	✓	✓	N/A	✓
FSA_7	✓	✓	✓	✓
FSA_8	×	×	✓	✓
FSA_9	✓	✓	×	✓
FSA_{10}	×	×	×	✓
FSA_{11}	×	×	✓	✓
FSA_{12}	×	×	×	✓

FSA_1: "replay attack"; FSA_2: "man-in-the-middle attack"; FSA_3: "mutual authentication"; FSA_4: "key agreement"; FSA_5: "device/drone impersonation attack"; FSA_6: "GSS/server impersonation attack"; FSA_7: "malicious device deployment attack"; FSA_8: "resilience against drone/device physical capture attack"; FSA_9: "formal security verification using AVISPA tool"; FSA_{10}: "ESL attack under the CK-adversary model"; FSA_{11}: "support dynamic drone/device addition phase"; FSA_{12}: "support blockchain-based solution"
✓: "a scheme is secure or it supports an attribute"; ×: "a scheme is insecure or it does not support an attribute"; N/A: means "not applicable" in a scheme.

addition," a "bilinear pairing" and a "modular exponentiation," respectively. In the proposed BSD2C-IoD, a drone DR_i needs the computation cost of $6T_h + 4T_{ecm} + T_{eca}$, where an GSS_j needs computation cost of $6T_h + 6T_{ecm} + 2T_{eca}$. We apply our experimental results reported in Section VI for various cryptographic primitives using MIRACL. For a drone (IoT smart device), we consider the execution time of various cryptographic primitives on a Raspberry PI 3 provided in Table IV. On the other side, for the GSS (server), we consider the execution time of various cryptographic primitives provided in Table III. Based on these results, a comparative analysis on computation costs among BSD2C-IoD and other schemes are provided in Table VI using the average execution time of cryptographic primitives. It is observed that BSD2C-IoD has 194% and 194% gains over the schemes of Luo *et al.* [38] and Li *et al.* [39] for drones (smart devices) computation costs point of view, where as BSD2C-IoD has also 275% and 378% gains over the schemes of Luo *et al.* [38] and Li *et al.* [39] for GSS (server) computation cost. Although Tian *et al.*'s scheme [40] requires less computation costs as compared to BSD2C-IoD, better security and more functionality features are provided in BSD2C-IoD as compared to the scheme of Tian *et al.* [40].

C. Security and Functionality Features Comparison

A comparative study on the "functionality and security attributes" (FSA_1–FSA_{12}) among BSD2C-IoD and other schemes is provided in Table VI. It exhibits that BSD2C-IoD has superiority over security and also offers more functionality features as compared to the existing schemes [38]–[40].

VIII. DISCUSSION

A new blockchain-based secure data delivery and collection scheme (BSD2C-IoD) has been put forward in this paper to secure the IoD environment. In BSD2C-IoD, the RA is responsible for registering/enrolling the control rooms (CR) and selecting the system parameters during the system initialization phase. A control room (CR_j) only involves during the registration of GSS_j and drones DR_i. Since the registration process in one-time procedure, it is worth noting that BSD2C-IoD is not a centralized scheme here. The access control proposed in the paper helps in establishing session keys among the drones DR_i and the GSSs for their secure communication. Furthermore, the data delivery and collection mechanism in BSD2C-IoD permits recording of all the transactions communicated among CR, GSS and drones for creating private blocks by the GSS, and then verifying and adding the blocks by a GSS_j being the leader in the P2P GSS network in the BC. As a result, BSD2C-IoD is a decentralized scheme in this sense. However, we require a detailed blockchain simulation (for example, Hyperledger, which is a "multi-project open source collaborative effort hosted by The Linux Foundation for the purpose of creating advance cross-industry blockchain technologies" [41]) for practical deployment in the IoD environment. In the future, we plan to do such kind of blockchain-based simulation study.

IX. CONCLUSION

This paper proposed a novel blockchain-envisioned secure data delivery and collection scheme for 5G-enabled IoD environment (BSD2C-IoD). BSD2C-IoD not only provides access control mechanism among the drones and the GSS in the flying zones, it also provides secure transactions among the drones, the GSSs and the CRs in the IoD environment, which are then used to form various blocks by the respective GSS. The formed blocks are then forwarded to a leader chosen among a set of the GSSs available in the P2P GSS network and the leader takes care of the block for verification and addition of the blocks in the blockchain maintained by the BC using the PBFT consensus algorithm. BSD2C-IoD is shown to be robust against a number of potential attacks needed in the blockchain. The performance analysis of BSD2C-IoD is also performed.

ACKNOWLEDGMENT

The authors would like to thank the anonymous reviewers and the Associate Editor for their valuable comments and suggestions which helped us to improve the presentation and quality of the paper.

REFERENCES

[1] B. Li, Z. Fei, and Y. Zhang, "UAV communications for 5G and beyond: Recent advances and future trends," *IEEE Internet Things J.*, vol. 6, no. 2, pp. 2241–2263, Apr. 2019.

[2] "Drones reporting for work," Goldman Sachs Research, 2020. [Online]. Available: https://www.goldmansachs.com/our-thinking/technology-driving-innovation/drones/, Accessed: Jan. 2020.

[3] A. Takacs, X. Lin, S. Hayes, and E. Tejedor, "Drones and networks: Ensuring safe and secure operations," 2018, Ericsson white paper, GFMC-18:000526, [Online]. Available: https://www.ericsson.com/4adfc0/assets/local/reports-papers/white-papers/10201110_wp_dronesandmobilenetworks_nov18.pdf, Accessed: Jan. 2020.

[4] T. Lagkas, V. Argyriou, S. Bibi, and P. Sarigiannidis, "UAV IoT framework views and challenges: Towards protecting drones as 'Things'," *Sensors*, vol. 18, no. 11, 2018.

[5] S. Kumari, M. Karuppiah, A. K. Das, X. Li, F. Wu, and N. Kumar, "A secure authentication scheme based on elliptic curve cryptography for IoT and cloud servers," *J. Supercomput.*, vol. 74, no. 12, pp. 6428–6453, 2018.

[6] "Evolution of wireless technologies 1G to 5G in mobile communication," 2018. [Online]. Available: https://www.rfpage.com/evolution-of-wireless-technologies-1g-to-5g-in-mobile-communication/, Accessed: Jan. 2020.

[7] L. Schroth, "5G and drones: Improving drone connectivity for UTM, BVLOS flights and sensor data transmission," 2020. [Online]. Available: https://www.droneii.com/drones-and-5g-improving-drone-connectivity, Accessed: May 2020.

[8] M. Choudhary, "What is BVLOS and why is it important for drone industry?" 2019. [Online]. Available: https://www.geospatialworld.net/blogs/what-is-bvlos-and-why-is-it-important-for-drone-industry/, Accessed on May 2020.

[9] D. Dolev and A. Yao, "On the security of public key protocols," *IEEE Trans. Inf. Theory*, vol. 29, no. 2, pp. 198–208, Mar. 1983.

[10] R. Canetti and H. Krawczyk, "Universally composable notions of key exchange and secure channels," in *Proc. Int. Conf. Theory Appl. Cryptographic Techn.*, Amsterdam, The Netherlands, 2002, pp. 337–351.

[11] T. S. Messerges, E. A. Dabbish, and R. H. Sloan, "Examining smart-card security under the threat of power analysis attacks," *IEEE Trans. Comput.*, vol. 51, no. 5, pp. 541–552, May 2002.

[12] AVISPA, "Automated validation of internet security protocols and applications," 2019. [Online]. Available: http://www.avispa-project.org/, Accessed: Oct. 2019.

[13] "MIRACL Cryptographic SDK: Multiprecision integer and rational arithmetic cryptographic library," 2020. [Online]. Available: https://github.com/miracl/MIRACL, Accessed: Apr. 2020.

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

[15] S. Seo, J. Won, E. Bertino, Y. S. Kang, and D. Choi, "A security framework for a drone delivery service," in *Proc. 2nd Workshop Micro Aerial Vehicle Netw., Syst., Appl. Civilian Use*, Singapore, 2016, pp. 29–34.

[16] N. Kumar, R. Iqbal, S. Misra, and J. J. Rodrigues, "An intelligent approach for building a secure decentralized public key infrastructure in vanet," *J. Comput. Syst. Sci.*, vol. 81, no. 6, pp. 1042–1058, 2015.

[17] D. He, N. Kumar, and J. Lee, "Privacy-preserving data aggregation scheme against internal attackers in smart grids," *Wireless Netw.*, vol. 22, no. 2, pp. 491–502, 2016.

[18] J. Wu and P. Fan, "A survey on high mobility wireless communications: Challenges, opportunities and solutions," *IEEE Access*, vol. 4, pp. 450–476, 2016.

[19] M. Mozaffari, W. Saad, M. Bennis, Y. Nam, and M. Debbah, "A tutorial on UAVs for wireless networks: Applications, challenges, and open problems," *IEEE Commun. Surv. Tut.*, vol. 21, no. 3, pp. 2334–2360, Jul.–Sep. 2019.

[20] A. Yaar, A. Perrig, and D. Song, "Pi: A path identification mechanism to defend against DDoS attacks," in *Proc. Symp. Secur. Privacy*, Berkeley, CA, USA, 2003, pp. 93–107.

[21] B. Bera, D. Chattaraj, and A. K. Das, "Designing secure blockchain-based access control scheme in IoT-enabled Internet of Drones deployment," *Comput. Commun.*, vol. 153, pp. 229–249, 2020.

[22] A. K. Das, M. Wazid, N. Kumar, A. V. Vasilakos, and J. J. P. C. Rodrigues, "Biometrics-based privacy-preserving user authentication scheme for cloud-based industrial internet of things deployment," *IEEE Internet Things J.*, vol. 5, no. 6, pp. 4900–4913, Dec. 2018.

[23] S. Malani, J. Srinivas, A. K. Das, K. Srinathan, and M. Jo, "Certificate-based anonymous device access control scheme for IoT environment," *IEEE Internet Things J.*, vol. 6, no. 6, pp. 9762–9773, Dec. 2019.

[24] M. Wazid, A. K. Das, V. Odelu, N. Kumar, and W. Susilo, "Secure remote user authenticated key establishment protocol for smart home environment," *IEEE Trans. Dependable Secure Comput.*, vol. 17, no. 2, pp. 391–406, Apr. 2020.

[25] S. Mandal, B. Bera, A. K. Sutrala, A. K. Das, K. R. Choo, and Y. Park, "Certificateless signcryption-based three-factor user access control scheme for IoT environment," *IEEE Internet Things J.*, vol. 7, no. 4, pp. 3184–3197, Apr. 2020.

[26] W. E. May, "Secure hash standard," 2015, FIPS PUB 180-1, National Institute of Standards and Technology (NIST), U.S. Department of Commerce, Apr. 1995. [Online]. Available: http://nvlpubs.nist.gov/nistpubs/FIPS/NIST.FIPS.180-4.pdf, Accessed: Jan. 2020.

[27] N. Koblitz, A. Menezes, and S. A. Vanstone, "The state of elliptic curve cryptography," *Des., Codes Cryptography*, vol. 19, no. 2-3, pp. 173–193, 2000.

[28] D. Johnson, A. Menezes, and S. Vanstone, "The elliptic curve digital signature algorithm (ECDSA)," *Int. J. Inf. Secur.*, vol. 1, no. 1, pp. 36–63, 2001.

[29] H. Zhang, J. Wang, and Y. Ding, "Blockchain-based decentralized and secure keyless signature scheme for smart grid," *Energy*, vol. 180, pp. 955–967, 2019.

[30] M. Castro and B. Liskov, "Practical Byzantine fault tolerance and proactive recovery," *ACM Trans. Comput. Syst.*, vol. 20, no. 4, pp. 398–461, 2002.

[31] D. Magazzeni, P. McBurney, and W. Nash, "Validation and verification of smart contracts: A research agenda," *IEEE Comput.*, vol. 50, no. 9, pp. 50–57, Sep. 2017.

[32] Y. Zhang, S. Kasahara, Y. Shen, X. Jiang, and J. Wan, "Smart contract-based access control for the internet of things," *IEEE Internet Things J.*, vol. 6, no. 2, pp. 1594–1605, Apr. 2019.

[33] A. Kapitonov, I. Berman, S. Lonshakov, and A. Krupenkin, "Blockchain based protocol for economical communication in industry 4.0," in *Proc. Crypto Valley Conf. Blockchain Technol.*, Zug, Switzerland, 2018, pp. 41–44.

[34] M. Abdalla, P. A. Fouque, and D. Pointcheval, "Password-based authenticated key exchange in the three-party setting," in *Proc. 8th Int. Workshop Theory Pract. Public Key Cryptography, Lecture Notes Comput. Sci.*, Les Diablerets, Switzerland, 2005, vol. 3386, pp. 65–84.

[35] C. C. Chang and H. D. Le, "A provably secure, efficient, and flexible authentication scheme for ad hoc wireless sensor networks," *IEEE Trans. Wireless Commun.*, vol. 15, no. 1, pp. 357–366, Jan. 2016.

[36] J. Srinivas, A. K. Das, N. Kumar, and J. J. P. C. Rodrigues, "TCALAS: Temporal credential-based anonymous lightweight authentication scheme for internet of drones environment," *IEEE Trans. Veh. Technol.*, vol. 68, no. 7, pp. 6903–6916, Jul. 2019.

[37] AVISPA, "SPAN, the security protocol ANimator for AVISPA," 2019. [Online]. Available: http://www.avispa-project.org/, Accessed: Oct. 2019.

[38] M. Luo, Y. Luo, Y. Wan, and Z. Wang, "Secure and efficient access control scheme for wireless sensor networks in the cross-domain context of the IoT," *Secur. Commun. Netw.*, vol. 2018, pp. 1–10, 2018.

[39] F. Li, Y. Han, and C. Jin, "Practical access control for sensor networks in the context of the Internet of Things," *Comput. Commun.*, vol. 89–90, pp. 154–164, 2016.

[40] Y. Tian, J. Yuan, and H. Song, "Efficient privacy-preserving authentication framework for edge-assisted Internet of Drones," *J. Inf. Secur. Appl.*, vol. 48, 2019, Art. no. 102354.

[41] "Hyperledger: Advancing business blockchain adoption through global open source collaboration," 2020. [Online]. Available: https://www.hyperledger.org/, Accessed: May 2020.

Basudeb Bera received the M.Sc. degree in mathematics and computing from IIT (ISM) Dhanbad, India, in 2014, and the M.Tech. degree in computer science and data processing from IIT Kharagpur, India, in 2017. He is currently working toward the Ph.D. degree in computer science and engineering from the Center for Security, Theory and Algorithmic Research, IIIT Hyderabad, India. His research interests include cryptography, network security and blockchain technology. He has published five papers in international journals and conferences in his research areas.

Sourav Saha (Student Member, IEEE) received the B.Tech degree in computer science and engineering from Central Institute of Technology, Kokrajhar, India, and the M.S. degree in computer science and engineering from the Indian Institute of Information Technology, Sri City, Chittoor, India. He is currently a Ph.D. student in computer science and engineering at the Center for Security, Theory and Algorithmic Research, International Institute of Information Technology, Hyderabad, India. His research interests include network security and blockchain technology. He has published seven papers in international journals and conferences in his research areas.

Ashok Kumar Das (Senior Member, IEEE) received the M.Sc. degree in mathematics, the M.Tech. degree in computer science and data processing, and the Ph.D. degree in computer science and engineering, from IIT Kharagpur, India. He is currently an Associate Professor with the Center for Security, Theory and Algorithmic Research, IIIT, Hyderabad, India. His current research interests include cryptography and network security including security in smart grid, Internet of Things (IoT), Internet of Drones (IoD), Internet of Vehicles (IoV), Cyber-Physical Systems (CPS) and cloud computing, and blockchain. He has authored over 220 papers in international journals and conferences in the above areas, including over 190 reputed journal papers. He was a recipient of the Institute Silver Medal from IIT Kharagpur. He is on the editorial board of *KSII Transactions on Internet and Information Systems, International Journal of Internet Technology and Secured Transactions (Inderscience)*, and *IET Communications*, is a Guest Editor for Computers & Electrical Engineering (Elsevier) for the Special Issue on Big data and IoT in e-healthcare and for *ICT Express* (Elsevier) for the Special Issue on Blockchain Technologies and Applications for 5G Enabled IoT, and has served as a Program Committee Member in many international conferences. He also severed as one of the Technical Program Committee Chairs of the International Congress on Blockchain and Applications (BLOCKCHAIN'19), Avila, Spain, June 2019.

Neeraj Kumar (Senior Member, IEEE) received the Ph.D. degree in CSE from Shri Mata Vaishno Devi University, Katra (J&K), India, and was a Postdoctoral Research Fellow at Coventry University, Coventry, U.K. He is working as an Associate Professor at the Department of Computer Science and Engineering, Thapar Institute of Engineering and Technology (Deemed to be University), Patiala, Punjab, India. He has published more than 350 technical research papers in leading journals and conferences from IEEE, Elsevier, Springer, John Wiley etc. Some of his research findings are published in top cited journals such as IEEE Transactions on Knowledge and Data Engineering, IEEE Transactions on Industrial Electronics, IEEE Transactions on Dependable and Secure Computing, IEEE Transactions on Intelligent Transportation Systems, IEEE Transactions on Consumer Electronics, IEEE Transactions on Industrial Informatics, IEEE Transactions on Vehicular Technology, IEEE Transactions on Intelligent Transportation, IEEE Network, IEEE Communications Magazine, IEEE Wireless Communications, IEEE Internet of Things Journal, IEEE Systems Journal, Future Generation Computer Systems, Journal of Network and Computer Applications, and Computer Communications. He is on the editorial board of ACM Computing Survey, IEEE TRANSACTIONS ON SUSTAINABLE COMPUTING, *IEEE Network Magazine, IEEE Communication Magazine, Journal of Networks and Computer Applications* (Elsevier), *Computer Communications* (Elsevier), *International Journal of Communication Systems* (Wiley), and *Security and Privacy* (Wiley).

Pascal Lorenz (Senior Member, IEEE) received the M.Sc. and Ph.D. degrees from the University of Nancy, France, in 1990 and 1994, respectively. Between 1990 and 1995 he was a Research Engineer at WorldFIP Europe and at Alcatel-Alsthom. He is a Professor with the University of Haute-Alsace, France, since 1995. His research interests include QoS, wireless networks and high-speed networks. He is the author/co-author of 3 books, 3 patents and 200 international publications in refereed journals and conferences. He was Technical Editor of the *IEEE Communications Magazine* Editorial Board (2000–2006), *IEEE Networks Magazine* since 2015, IEEE TRANSACTIONS ON VEHICULAR TECHNOLOGY since 2017, Chair of IEEE ComSoc France (2014–2018), Financial chair of IEEE France (2017–2019), Chair of Vertical Issues in Communication Systems Technical Committee Cluster (2008–2009), Chair of the Communications Systems Integration and Modeling Technical Committee (2003–2009), Chair of the Communications Software Technical Committee (2008–2010) and Chair of the Technical Committee on Information Infrastructure and Networking (2016–2017). He is Associate Editor for *International Journal of Communication Systems* (IJCS-Wiley), *Journal on Security and Communication Networks* (SCN-Wiley) and *International Journal of Business Data Communications and Networking*, *Journal of Network and Computer Applications* (JNCA-Elsevier). He is an IARIA fellow and member of many international program committees.

Mamoun Alazab (Senior Member, IEEE) received the Ph.D. degree in computer science from the School of Science, Information Technology and Engineering, Federation University of Australia. He is currently an Associate Professor with the College of Engineering, IT and Environment, Charles Darwin University, Australia. He is also a Cyber Security Researcher and a Practitioner with industry and academic experience. His research is multidisciplinary that focuses on cyber security and digital forensics of computer systems with a focus on cybercrime detection and prevention, including cyber terrorism and cyber warfare. He works closely with government and industry on many projects, including the Northern Territory (NT) Department of Information and Corporate Services, IBM, Trend Micro, the Australian Federal Police (AFP), etc. He is the Founder and the Chair of the IEEE Northern Territory (NT) Subsection Detection and Prevention.