IEEE Access
Multidisciplinary | Rapid Review | Open Access Journal

Received January 13, 2022, accepted January 31, 2022, date of publication February 7, 2022, date of current version February 24, 2022.

Digital Object Identifier 10.1109/ACCESS.2022.3149376

RUAM-IoD: A Robust User Authentication Mechanism for the Internet of Drones

MUHAMMAD TANVEER[1], AHMED ALKHAYYAT[2], ALAMGIR NAUSHAD[3], ABD ULLAH KHAN[3], (Member, IEEE), NEERAJ KUMAR[4,5,6], (Senior Member, IEEE), AND ABDULLAH G. ALHARBI[7], (Member, IEEE)

[1]School of Systems and Technology (SST), University of Management and Technology, Lahore 54770, Pakistan
[2]Department of Computer Technical Engineering, College of Technical Engineering, Islamic University, Najaf 54001, Iraq
[3]Department of Computer Science, National University of Sciences and Technology (NUST) Balochistan Campus (NBC), Quetta 87300, Pakistan
[4]School of Computer Science, University of Petroleum and Energy Studies, Dehradun, Uttarakhand 248007, India
[5]Department of Computer Science and Information Engineering, Asia University, Taichung City 41354, Taiwan
[6]Department of Electrical and Computer Engineering, King Abdulaziz University, Jeddah 21589, Saudi Arabia
[7]Department of Electrical Engineering, Faculty of Engineering, Jouf University, Sakaka 42421, Saudi Arabia

Corresponding author: Abd Ullah Khan (abdullah@nbc.nust.edu.pk)

ABSTRACT The revolutionary advancement in the capabilities of hardware tools, software packages, and communication techniques gave rise to the Internet of Things-supported drone networks (IoD), thereby enabling smooth communication among devices and applications, and impacting drastically the various aspects of human lives. However, with the increasing sophistication in the infrastructure of IoD, new security threats arise that require novel algorithms and schemes as solutions. To this end, several schemes have recently been proposed. However, some schemes cannot perfectly address the novel security aspects associated with IoD environments, while others cannot provide computational or communication efficiency. Motivated by these research gaps in the existing literature, we leverage elliptic curve cryptography along with symmetric encryption and hash function, and propose a novel and robust user authentication mechanism for the IoD, called RUAM-IoD. We validate the security of the established SK formally through the random oracle model. Similarly, we provide informal security analysis to demonstrate the security capabilities of RUAM-IoD against different pernicious security attacks. Likewise, we establish a comparison of the RUAM-IoD with several state-of-the-art authentication schemes to show that RUAM-IoD acquires less storage, communication, and computational cost.

INDEX TERMS Internet of Drone, privacy, unmanned aerial vehicles, key exchange, security.

I. INTRODUCTION

Unmanned aerial vehicle (UAV) or drone is a versatile platform for communication that provides flexibility in altitude, line-of-sighting, mobility, and so on. Consequently, drones can be considered as broadband wireless access solutions for terrestrial network devices [1], [2]. The applications that benefit from UAVs include military, traffic management, tracing and tracking, disaster management, surveillance and monitoring, wireless communication, and so on [3]. This implies that drones can increasingly become capable of providing ubiquitous computing, on board processing, and wireless communication. This way, drones can serve as airborne base stations, thereby expanding the reachability of terrestrial networks. Moreover, as a flying base station, a drone is immune to damage caused by geographical disasters and calamities, consequently proving economically efficient [4]. The most important and unique feature of the smart UAVs is the provisioning of effective, dependable, and quick connection establishment in urban and rural areas, large roads, expanding regions, etc.

To impart these services, UAVs are dependent on the Internet of Things (IoT) networks, thereby leading to the Internet of Drones (IoD) networks [5]. Fig. 1 general overview of the UAV/IoD based communication system. Typically, the IoD networks contain a ground station (GS), several flying drones, and a certain number of remote users. The drones play the role of collecting information from the environment of interest and transmitting the collected information to the corresponding server residing within GS. The GS controls through wireless channels (by sending control commands) the type of information required to be collected by the drones, and

The associate editor coordinating the review of this manuscript and approving it for publication was Xiaolong Li.

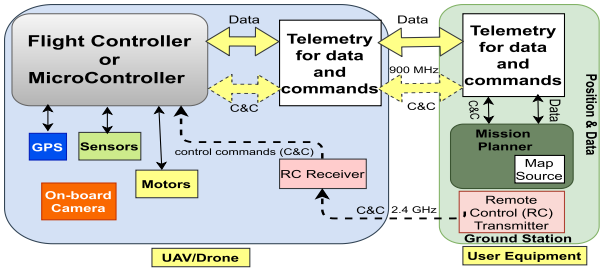

FIGURE 1. UAV/Drone based communication system [5], [8].

the frequency of the information collection [6], [7]. Remote users are the beneficiary of the information collected by the drones and processed by the GS. This implies that remote users can access in real-time the information collected by drones using the Internet. However, using the public wireless channels for such information retrieval from drones poses security threats and vulnerabilities, leading to unauthorized information exposure. Given that the information collected by drones is sensitive and private most of the time, its security and privacy cannot be ignored. This implies that the secure exchange of information between users and drones in the IoD environment is a critical requirement for realizing the benefits of drones and their applications [9].

However, there are several challenges to deal with for perfectly exploiting the IoD for the applications mentioned above. These challenges include power consumption of the drones, optimization of a drone's trajectorial motion, deployment of the remote user's association, the communication protocol used, the throughput and latency improvement, the effective link establishment, and security and privacy [10], [11].

II. RELATED WORK

Secure with privacy-preserving communication mechanisms are imperative for the IoD networks. Security and privacy requirements in the IoD networks are reviewed in [9], [21]. Moreover, various user AKA schemes are presented in [17], [22]–[30] to enable encrypted and reliable communications in the IoD environment. However, a vast majority of these AKA schemes are prone to a variety of security attacks. Table 1 tabulates user AKA schemes with their limitations and techniques employed in these schemes. The authors in [5] presented a terrestrial credential-based AKA scheme and employed a random oracle model (ROM) to validate the session key's (SK) security. The scheme checks the authenticity of the user before accessing the sensitive information from a specific drone in real-time and fixes a secret SK among drones and users for indecipherable communication. Nevertheless, the scheme is unable to prevent the attackes related to UI and PI. Similarly, Wazid *et al*. [12] designed a user AKA scheme and employed ROM to validate the SK's security. The scheme of Wazid *et al*. is unable to resist UI and PI attacks. Sajid *et al*. [27] proposed an AKA scheme by using ECC and SHA-160. The scheme renders user authentication, and SK establishment features to secure the communication in the IoD networks. Likewise, Tanveer *et al*. [31] devised an AKA scheme by employing SHA-256 and authenticated encryption with associative data (AEAD). Nevertheless, the scheme is incompetent in providing an anonymity feature, as pointed out in [17].

Proceeding in the same direction, Jangirala *et al*. [32] designed an AKA scheme for the IoT network to collect critical information from the IoT device deployed in the target field in real-time. The scheme utilizes ECC and SHA-160 cryptographic techniques to accomplish the AKA phase. But, the scheme is exposed to MITM, UI, DI, and SK disclosure attacks. In addition, the scheme suffers from a design flaw and lacks to provide untraceablity. The authors in *et al*. [33] proposed an AEAD and ECC-based AKA scheme to enable indecipherable communication in the IoD networks. The scheme enables a drone and user to establish an SK for indecipherable information exchange. Similarly, Tanveer *et al*. [34] proposed an AKA scheme by employing an AEAD encryption algorithm and SHA-256 to accomplish the AKA phase. The scheme assures the authenticity of the user before procuring the real-time information directly from the IoT device. Sutrala *et al*. [35] proposed an AKA scheme by employing ECC and SHA-256 cryptographic algorithms to guarantee secure communication between the user and the device deployed in the IoT environment. However, the scheme unable to resist the DSY attack. A user AKA scheme is suggested by Sajid *et al*. [27] for the IoD network that employs SHA-160 and ECC, which enables both the user and drone to communicate securely after establishing SK. Wazid *et al*. [12] designed AKA scheme employing ECC and SHA-160 to enable encrypted communication in the IoD environment. Nevertheless, the their scheme cannot provide resistance against certain attacks like UI and PI.

Aside from these works, Ali *et al*. [14] designed an AKA protocol to ensure the indecipherable communication between user and drone. The scheme is based on SHA-160 and XOR operation. However, it is found in [14] that the scheme is prone to PI, UI, SSD, forgery, and denial-of-service (DoS) attacks. Bera *et al*. [36] devised an AKA protocol for the IoD networks by employing cryptographic techniques, such as ECC and SHA-256. However, the authors couldn't stop DSY attack in their scheme. Iqbal *et al*. [37] formulated an AKA scheme for smart home environment by employing SHA-160 and XOR. However, it is proved in [38] that Iqbal *et al*.'s scheme is exposed to SK disclosure, UI, and MITM attacks. Moreover, it is also found that the presented approach does not provide user anonymity (UA) and mutual authentication (MA). A puncturable pseudorandom function-based user AKA scheme is proposed in [39]. The scheme permits users and end-devices to establish an SK after achieving MA. Vinoth *et al*. [40] designed a multi-factor AKA scheme for the IoT environment using SHA-160 and AES. However, the scheme is prone to DoS, DSY, replay, and device capture attacks.

TABLE 1. A cursory review of eminent user AKA schemes.

AKA schemes	Environment	Year	Limitations	Techniques
Wazid *et al.* [12]	IoT/IoD	2018	Prone to user impersonation (UI) and privilege insider (PI) attacks.	SHA and Exclusive-OR (XOR)
Gope and Sikdar [13]	IoT/SG	2019	Fragile to replay, man-in-the-middle (MITM), and CS impersonation (CSI) attacks. Suffers bio-metric de-synchronization and does not render SK verification mechanism.	XOR and secure hash algorithm (SHA)
Jangirala *et al.* [5]	IoT/IoD	2019	Prone to user impersonation (UI) and privilege insider (PI) attacks.	SHA and XOR
Ali *et al.* [14]	IoT/IoD	2020	Does not ensure the indistinguishability of encryption under chosen plaintext attack (IND-CPA) and anonymity and untraceability features.	XOR, advanced encryption standard (AES), SHA
Ever [15]	IoT/IoD	2020	Vulnerable to ephemeral secret leakage (ESL) and does not provide the new drone addition phase. In addition, it does not render untraceability and anonymity properties.	Elliptic curve cryptography (ECC), SHA, bi-linear paring, AES
Cho *et al.* [16]	IoD/IoT	2020	Does not resist ESL, anonymity and untraceability attacks. Does not render new drone addition phase.	ECC, SHA, ECC-based digital signature, and AES
Shehzad *et al.* [17]	IoT/IoD	2021	Does not provide anonymity and prone to drone/device impersonation (DI), CSI, and ESL attacks.	ECC, XOR, and SHA
Wazid *et al.* [18]	IIoT	2020	Unable to resist De-Synchronization (DSY) attack.	ECC, SHA, and XOR
Zhang *et al.* [19]	IoD	2020	Vulnerable to forgery and drone capture attacks.	SHA and XOR
Rena *et al.* [20]	IoT	2021	Unable to render untraceability and anonymity features. Does not protect stolen smart device (SSD), PI, ESL, UI, and bio-metric and password change (BPC) attacks.	SHA and XOR

A. MOTIVATION

Drones collect sensitive data from various environments of interest and dispatch the data to GS via a wireless channel. The channel is vulnerable and risky, and can be exploited. Moreover, oftentimes distant users need to collect the sensitive information, in real-time, directly from the deployed drone, rather than utilizing the data collected at the central server (*CS*) posted at GS. Therefore, it is imperative to allow only authorized users to obtain critical information directly from the drone. In addition, it is necessary to protect the communication between the drone and the remote user from being disclosed by an attacker or adversary. Above this, most of AKA schemes proposed so far for this purpose are insecure against the PI attack because, in these schemes, the secret information related to users are stored in plaintext form [31], [41]. An insider adversary can obtain the secret information associated with a specific user from *CS* and execute an attack on behalf of the user. Other AKA schemes proposed for providing secure communication with drones are vulnerable to BPC, SSD, UI, and MITM attacks. Thus, it is crucial to design a secure and reliable AKA scheme to ensure indecipherable communication for the IoD environment [9].

B. RESEARCH CONTRIBUTION

This paper has the following main contributions.

1) A novel and robust user authentication mechanism for the IoD, called RUAM-IoD, is presented, which is based on the AES-CBC-256 encryption, ECC, SHA-256 hash function, and XOR operation. The proposed RUAM-IoD authenticates the user prior to enabling him to access the network's resources. In addition, RUAM-IoD establishes a private SK to accomplish the encrypted communication in the IoD network while ensuring the user's anonymity during the execution of the AKA phase.
2) We perform informal security verification of RUAM-IoD that shows that RUAM-IoD is secure against a variety of security risks, including DSY, BPC, drone capture, and SSD attacks. Moreover, we employ ROM-based formal validation and prove the security strength of the established private SK. Furthermore, we perform Scyther-based security validation and demonstrate that RUAM-IoD is dependable against replay and MITM attacks. In addition, RUAM-IoD is able to prevent PI attack by storing, in encrypted form, the secret information associated with the users and drones in the memory of *CS*.
3) We compare RUAM-IoD with relevant AKA schemes and prove that RUAM-IoD is comparatively efficient in communication, storage, and computational costs. Moreover, we prove that RUAM-IoD renders enhanced security features than the related schemes.

C. PAPER ORGANIZATION

The remaining parts of the paper is arranged as follow. The system model, i.e., the network and threat model, is described in Section III. The preliminaries are discussed in Section IV. The functional phases of the RUAM-IoD scheme are explained in detail in Section V. In Section VI, informal security analysis is carried out, ROM based validation is performed, and Scyther-based formal security is discussed. Lastly, the comparative analysis is presented in Section VII, and the concluding remarks in Section VIII.

III. SYSTEM MODEL

A. NETWORK MODEL

Fig. 2 shows the authentication model used in RUAM-IoD for the networking of drone service provider (DSP). Moreover, it is assumed that the airspace of a smart city is divided into the different fly zone (FZ). Likewise, it is assumed that the DSP network model consists of DSP registration center (*DRC*), *CS*, user ($U_e|e = 1, 2, 3, \cdots, n$), where the notation n denotes the users' number in the IoD environment, and drone ($D_x|x = 1, 2, 3, \cdots, N$), where the notation N represents the drones' number in the IoD environment. *DRC* is responsible for the registration and deployment of D_x in the specific FZ. *CS* is utilized to cache the sensitive data gathered by D_x. In addition, *CS* also stores the secret information

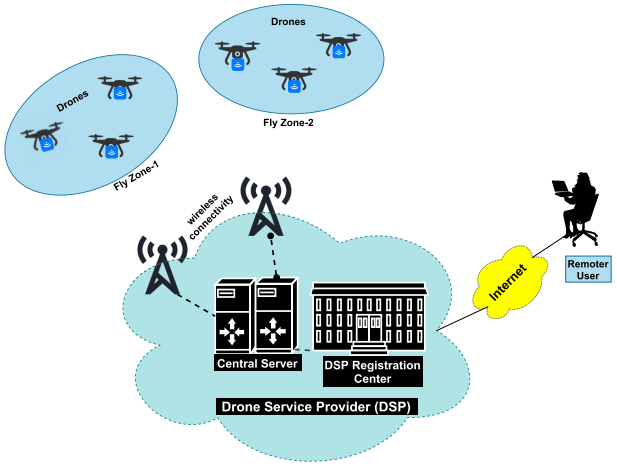

FIGURE 2. An application scenario: IoD network [8].

associated with U_e and D_x. Moreover, D_x is deployed to monitor and collect critical information from a specific FZ and to disseminate the collected information to *CS* via a wireless channel.

In a specific IoD application, for instance, a smart city traffic management system, U_e needs to collect traffic congestion information directly from D_x to avoid the delay. Thus, it is necessary to protect the information exchanged between U_e and D_x. Also, it is required to ensure that the attacker cannot modify the information communicated in the IoD environment. This implies that it is imperative to prevent the unauthorized U_e from accessing the IoD network resources. Therefore, an AKA scheme is necessary to ensure the encrypted communication between U_e and D_x after the authentication of U_e is carried out. RUAM-IoD ensures the encrypted communication after establishing SK between U_e and D_x.

B. THREAT MODEL

The network entities usually exchange information through a public communication channel exposed to various security risks. Thus, adversary $\mathcal{A}$ can compromise the information exchanged between the network entities by taking advantage of the public nature of the communication channel. In RUMP-IoD, we have contemplated the widely-accepted "Dolev-Yao (DY) threat model [42], [43]. Under the DY threat model, $\mathcal{A}$ can capture and eavesdrop on all the communicated information or message of the network's nodes. $\mathcal{A}$ can also alter, forge, delete, and implant bogus information while communicating with the network entities. In addition to DY model, "Canetti and Krawczyk's model (CK-adversary model) is also applied on the proposed RUAM-IoD. According to the CK-adversary model, $\mathcal{A}$ can compromise short-term secret (STS), session states, and SKs by hijacking the sessions. Therefore, the composition of SK established between network entities should be based on both STS and long-term secrets (LTS) to resist the ESL attacks. Furthermore, $\mathcal{A}$ can physically compromise or capture some D_x and smart devices SD_es. Consequently, $\mathcal{A}$ can extract the secret or confidential parameters, which are pre-loaded in the memory of these devices, by utilizing power analysis (PA) attacks. However, *DRC* is deemed as an entirely entrusted network entity in the IoD. environment.

IV. PRELIMINARIES

This section presents the preliminaries used in RUMP-IoD.

A. AES-CBC-256

AES-CBC-256 is stateless cipher block chaining mode of the AES algorithm, which satisfies the IND-CPA property. Logically, the encryption process of AES-CBC-256 defined as follows

$$CT_x = E_{key}\{(IV), PT\}, \tag{1}$$

where PT, IV, CT_x, and denotes the plaintext, the initialization vector, and the ciphertext, respectively. Moreover, *key* denotes the encryption key of size 256. Furthermore, the decryption process using AES-CBC-256 is defined by

$$PT = D_{key}\{(IV), CT_x\}, \tag{2}$$

where PT shows the plaintext retrieved from the decryption mechanism. In the proposed RUAM-IoD, AES-CBC-256 is employed as encryption/decryption algorithm, which satisfies IND-CPA property. Formally IND-CPA can be defined as follows [44], [45].

Definition 1: Let single/multiple eavesdropper are denoted by SE/ME, respectively. Let OR_{key1}, $OR_{key2}, \cdots, OR_{keyN}$ *denote N distinct independent encryption oracles corresponding to encryption keys* $key1, key2, \cdots, keyN$, *respectively. We denote the advantage function of SE/ME as*

$$\begin{aligned} Ad^{IND-CPA}_{SE,\Omega}(l) = {} & 2 \cdot Pb[SE \leftarrow OR_{key1}; (B0, B1 \leftarrow_R SE); \\ & \theta \leftarrow_R \{0,1\}; \gamma \leftarrow_R OR_{key1}(b_\theta) : \\ & SE(\gamma) = \theta] - 1, \end{aligned} \tag{3}$$

$$\begin{aligned} Ad^{IND-CPA}_{ME,\Omega}(l) = {} & 2 \cdot Pb[ME \leftarrow OR_{key1}, \cdots, OR_{keyN}; \\ & (B0, B1 \leftarrow_R SE); \theta \leftarrow_R \{0,1\}; \gamma_1 \\ & \leftarrow_R OR_{key1}(b_\theta), \cdots, \gamma_N \leftarrow_R OR_{key_N} \\ & (b_\theta) : ME(\gamma_1, \cdots, \gamma_N) = \theta] - 1. \end{aligned} \tag{4}$$

Here, Ω *denotes an encryption algorithm (AES-CBC-256), which is IND-CPA secure in ME/SE eavesdropper setting and* $Ad^{IND-CPA}_{SE,\Omega}(l)$ *or* $Ad^{IND-CPA}_{ME,\Omega}(l)$ *is trivial for* $\mathcal{A}$ *in polynomial time (TPoly).*

B. FUZZY EXTRACTOR

The fuzzy extractor (FE) mechanism is a broadly adopted tool to validate bio-metric authentication. FE is specified as a tuple $\{B_{in}, Lth, ERT\}$ and comprises the subsequent two algorithms.

1) $Gen(\cdot)$: It takes user's bio-metric information B_{in} as the input parameter and generates bio-metric key $BK \in [0,1]^{Lth}$, where Lth denotes the length of

TABLE 2. List of notations used in RUAM-IoD.

Notation	Description
ID_{U_e}, PAW_{U_e}	Identity and password of user
PID_{U_e}, SP_{U_e}	Pseudo-identity and secret parameter of user
PID_{D_x}, SP_{D_x}	Pseudo-identity and secret parameter of drone
$Athn^l \stackrel{?}{=} Athn$	If both $Athn^l$ and $Athn$ are equal
$EC(\alpha, \beta), P$	Elliptic curve with base point
S_{kg}, Pu_g	Private and public key of CS
S_{ku}, Pu_u	Private and public key of user
S_{kd}, Pu_d	Private and public key of drone
B_{in}	Bio-metric information of legitimate user
$Gen(.), Rep(.)$	FE bio-metric key generation and reproduction functions
ERT	Error tolerance parameter of FE algorithm
BK_{U_e}, rpp	FE generated bio-metric key and reproduction parameter
$BK^l_{U_e}$	Bio-metric key generated by FE and used in the login phase
$k, k1, k_u$	Secret keys used in the encryption process during the AKA process
$E_{key}(Str)$	Represents the encryption process to encrypt "Str"
$D_{key}(ctx)$	Represents the decryption process to decrypt "ctx"
$Pb[x]$	Represents the probability of event "x"
TP_{oly}	Polynomial-time
$\|, \mathcal{A}, H(.), \oplus$	Concatenation, adversary, hash function, XOR, and hash-function, respectively

BK and reproduction parameter rrp, i.e, $Gen(B_{in}) = (BK, rrp)$). In addition, $Gen(\cdot)$ is a deterministic algorithm.

2) $Rep(\cdot)$: It takes user's noisy B_{in} and rrp as the input parameter and generate BK, as $Rep(B'_{in}, rrp) = BK$ with the necessary condition $HD(B'_{in}, B_{in}) \leq ERT$, where ERT is the error tolerance and HD is the hamming distance between B'_{in} and B_{in}.

V. RUAM-IoD

This section presents our proposed AKA scheme, RUAM-IoD. The proposed RUAM-IoD comprises six phases: (i) System Initialization Phase, (ii) Drone Registration (DRG) Phase, (iii) User Registration (URG) Phase, (iv) AKA Phase, (v) BPC Phase, and (vi) Revocation Phase. These phases are elaborated in detail in the following sub-sections. Table 2 shows a list of notations employed in RUAM-IoD.

A. INITIALIZATION PHASE OF THE SYSTEM

DRC determines an elliptic curve, i.e., $EC(\alpha, \beta)$, of the form $Y^2 = X^3 + \alpha X + \beta \ (mod\ q)$ over $GF(q)$, q being the big prime number, with the condition $4\alpha^3 + 27\beta^2 \neq 0 \ (mod\ q)$, with $\mathcal{O}$ as the the point of infinity. DRC then selects picks a $EC(\alpha, \beta)$ base or generation point P, such that $P \in EC(\alpha, \beta)$, of order, say N, where $N.P = P+P+P+P+\cdots+(N\ times)$. DRC picks private key $S_{kg} \in Z_q^*$ for CS and generate a public as $Pu_g = S_{kg} \cdot P$. Moreover, DRC stores the parameters $\{S_{kg}, Pu_g, EC(\alpha, \beta), P\}$ in the tempered proof database of CS. Eventually, CS makes the parameters $\{Pu_g, EC(\alpha, \beta), P\}$ public in the IoD environment.

B. DRG PHASE

In DRG phase, DRC is responsible for registering D_x prior to its deployment in a particular FZ. In addition, DRC preloads some distinct secret parameters in the memory of D_x. These secret parameters are used during the AKA process. DRC executes the following measures to position a D_x in a particular FZ.

1) STEP DRG-1

DRC determines a distinct random-number R_{D_x} along with a pseudo-identity PID_{D_x} for a specific D_x. In addition, DRC computes the secret parameter SP_{D_x} for D_x as follows.

$$G = H(R_{D_x} \parallel PID_{D_x}), \tag{5}$$

$$SP_{D_x} = (G_1 \oplus G_2), \tag{6}$$

where G_1 and G_2 are procured after dividing G into two parts (*128 bits each because the output of the SHA-256 is 256 bits*).

2) STEP DRG-2

Eventually, DRC reserves the credential $\{PID_{D_x}, P, SP_{D_x}\}$ in D_x's memory. In addition, D_x has the access to all the public parameters of CS, such $\{Pu_g, EC(\alpha, \beta), P\}$.

C. URG PHASE

In URG phase, U_e register itself with DRC before accessing the services from a specific D_x, which is deployed the DSP. The DRC issues a smart device SD_e with pre-loaded secret parameters. The assigned secret parameters are validated by CS during the AKA process to allow U_e to procure the sensitive information from a particular D_x in real-time. DRC register a user by performing the following necessary steps.

1) STEP URG-1

U_e selects its password PAW_{U_e} and unique identity ID_{U_e}. In addition, U_e marks its bio-metric B_{in} at the sensor's bio-metric installed on SD_e and determines bio-metric key BK_{U_e} and rpp as $(BK_{U_e}, rpp) = Gen(B_{in})$. Moreover, SD_e determines $RM = H(ID_{U_e} \parallel PAW_{U_e})$ and contrives a registration request message $M_{rrm} : \{RM\}$ and dispatches M_{rrm} to DRC via a secure communication channel.

2) STEP URG-2

After getting M_{rrm} from U_e, DRC selects a random-number IV_{reg}, bearing 128 bits size, and determines the secret parameter SP_{U_e} and PID_{U_e} for U_e as follows.

$$A = H(S_{kg} \parallel IV_{reg} \parallel M_{rrm}), \tag{7}$$

$$SP_{U_e} = (A_1 \oplus A_2), \tag{8}$$

$$PID_{U_e} = (A_1 \oplus SP_{U_e}), \tag{9}$$

where A_1 and A_2 are derived by splitting A into two strings or parts of the same size (128 bits each). Moreover, DRC determines

$$Y = (PID_{U_e} \oplus SP_{U_e} \oplus PID_{D_x}), \tag{10}$$

$$K_g = (IV_{reg} \parallel Y), \tag{11}$$

$$CT_g = E_{K_g}\{(IV_{reg}), PT_g\}, \tag{12}$$

where K_g is the secret key, which is used to encrypt plaintext $PT_g = \{SP_{U_e}, SP_{D_x}, PID_{D_x}\}$ by employing AES-CBC-256

and stores the parameters $\{PID_{U_e}, CT_g, IV_{reg}\}$ in the memory of CS. *In the proposed RUAM-IoD, the sensitive information associated with U_e and D_x are stored in encrypted form.* Finally, DRC constructs a registration responses message $M_{rep} : \{PID_{U_e}, SP_{U_e}, PID_{D_x}\}$ and dispatches M_{rep} to U_e through a secure channel.

3) STEP URG-3

After procuring M_{rep} from DRC, SD_e selects an initialization vector IV_a and computes

$$AR = (BK_{U_e} \oplus ID_{U_e}), \tag{13}$$
$$KR = (AR \parallel PAW_{U_e}), \tag{14}$$
$$Athn = H(PAW_{U_e} \parallel ID_{U_e} \parallel BK_{U_e} \parallel Y), \tag{15}$$
$$CT_{xt} = E_{KR}\{(IV_a), PT_{xt}\}, \tag{16}$$

where AR is obtained by XORing bio-metric key and identity of U_e, KR is the secret key used in the encryption process to encrypt the plaintext or the sensitive information associated with U_e. CT_{xt} is obtained by encrypting $PT_{xt} = \{PID_{U_e}, SP_{U_e}, PID_{D_x}, Y\}$ by using AES-CBC-256. $Athn$ is the authentication parameter, which is obtain by performing the hash operation on PAW_{U_e}, ID_{U_e}, BK_{U_e}, and Y. Finally, SD_e stores the credentials $\{CT_{xt}, Athn, rpp, Gen(.), ERT, Rep(.), IV_a\}$ in its inherent memory.

D. AKA PHASE

In this phase, U_e achieves the local authentication by providing its secret credentials as the input to SD_e. After performing local authentication, SD_e sends the AKA request to CS for the further validation of U_e. To ensure encrypted communication in the future, both U_e and D_x to set up an SK with the help of CS. Following steps are needed to execute this phase of AKA.

1) STEP AKA-1

U_e achieves the local authentication by making use of its own secret parameters, including password PAW_{U_e} and identity ID_{U_e}, and bio-metric B^l_{in}. After receiving these secret parameters, SD_e computes the bio-metric key $(BK^l_{U_e}) = Rep(B^l_{in}, rpp)$. To verify the authenticity of U_e's secret parameters, SD_e computes

$$AR^l = (BK^l_{U_e} \oplus ID_{U_e}), \tag{17}$$
$$KR^l = (AR^l \parallel PAW^l_{U_e}), \tag{18}$$
$$PT_{xt} = D_{KR^l}\{CT_{xt}\}, \tag{19}$$
$$PT_{xt} = \{PID_{U_e}, SP_{U_e}, PID_{D_x}, Y\}, \tag{20}$$
$$Athn^l = H(PAW^l_{U_e} \parallel ID_{U_e} \parallel BK^l \parallel Y), \tag{21}$$

and checks the condition $Athn^l \stackrel{?}{=} Athn$. If it holds, SD_e passes the local authentication of U_e and proceeds the AKA process. Otherwise, SD_e discontinues the AKA process.

2) STEP AKA-2

After achieving the local authentication, SD_e chooses a timestamp T_A, random-number R_{U_e}, distinct secret key S_{ku}, bearing 32, 128, 160 bits sizes, respectively, and calculates

$$Pu_u = P \cdot S_{ku}, \tag{22}$$
$$k = S_{ku} \cdot Pu_g, \tag{23}$$
$$U = H(k \parallel T_A), \tag{24}$$

where Pu_u denotes the public key of U_e, k represents the shared secret, which is obtained after performing ECC point multiplication of S_{ku} and Pu_g, and U is obtained after performing the hash operation on k and T_A. In addition, SD_e computes

$$Q_1 = (Y \parallel PID_{U_e}) \oplus U, \tag{25}$$
$$K_u = (Y \parallel SP_{U_e}), \tag{26}$$
$$IV_u = U_a \oplus U_b, \tag{27}$$
$$Q_2 = E_{K_u}\{(IV_u), R_{U_e}, PID_{D_x}\}, \tag{28}$$
$$Athn^{n1} = H(PID_{U_e} \parallel PID_{D_x} \parallel Y \parallel R_{U_e} \parallel T_A), \tag{29}$$

where Q_1 is obtained after XORing U and concatenation of Y and PID_{U_e}, K_u is the secret key of size 256 bits, used to encrypt R_{U_e}, and PID_{D_x}, IV_u denotes the initialization vector, Q_2 is obtained after preforming encryption using AES-CBC-256, and $Athn^{n1}$ represents the authentication parameter, which will be verified at the destination. Furthermore, SD_e constructs the message $MS_a : \{T_A, Q_1, Q_2, Pu_u, Athn^{n1}\}$ and sends MS_a to CS for further verification via open channel.

3) STEP AKA-3

Upon procuring MS_a, CS determines the freshness of MS_a after verifying the condition $T_{DL} \geq |T_r - T_A|$. If MS_a is fresh or within the specified time delay limit, CS computes

$$k1 = S_{kg} \cdot Pu_u, \tag{30}$$
$$U_2 = H(k1 \parallel T_A), \tag{31}$$
$$(Y \parallel PID_{U_e}) = Q_1 \oplus U_2. \tag{32}$$

where $k1$ denotes the shared secret between CS and U_e and U_2 is obtained after performing hash of $k1$ and T_A. Moreover, after retrieving Y and PID_{U_e} from Q_1, CS checks if PID_{U_e} exist in its database. If it is found, CS retrieves $\{CT_g\}$ related to PID_{U_e} from its own database. Furthermore, CS computes

$$K_g = (IV_{reg} \parallel Y), \tag{33}$$
$$PT_g = D_{K_g}\{(IV_{reg}), CT_g\}, \tag{34}$$

where K_g is the secret key, which is used to decrypt the encrypted (CT_g) information stored and after successful decryption process CS retrieves the plaintext $PT_g = \{SP_{U_e}, (PID_{D_x}, SP_{D_x})\}$. Additionally, CS computes

$$K_d = (Y \parallel SP_{U_e}), \tag{35}$$
$$IV_{u2} = U^a_2 \oplus U^b_2, \tag{36}$$
$$(R_{U_e}, PID_{D_x}) = D_{K_d}\{(IV_{u2}), Q_2\}, \tag{37}$$
$$Athn^{n2} = H(PID_{U_e} \parallel PID_{D_x} \parallel Y \parallel R_{U_e} \parallel T_A), \tag{38}$$

where K_d is the secret key used to decrypt Q_2 and IV_{u2} denotes the initialization vector. Finally, to validate the

authenticity of MS_a, CS validates the condition $Athn^{n1} \stackrel{?}{=} Athn^{n2}$. If it holds, CS accept the received MS_a. Otherwise, CS terminates the AKA procedure.

4) STEP AKA-4

After getting the validity of U_e verified, CS chooses a timestamps T_B, random-number R_G of size 32 and 128 bits, respectively. Moreover, CS computes

$$K_G = H(T_B \parallel SP_{D_x} \parallel PID_{D_x}), \tag{39}$$
$$R'_G = R_{U_e} \oplus PID_{U_e} \tag{40}$$
$$IV_G = K_G^a \oplus K_G^b, \tag{41}$$
$$Q_3 = E_{K_G}\{(IV_G), R_{U_e}, R'_G\}, \tag{42}$$
$$Athn^{n3} = H(T_B \parallel R_{U_e} \parallel R'_G \parallel SP_{D_x} \parallel PID_{D_x}), \tag{43}$$

where K_G is the secret key used in the encryption process, IV_G is the initialization vector, Q_3 is obtained after performing the encryption using AES-CBC-256, and $Athn^{n3}$ is the authentication parameter. Finally, CS constructs the message $MS_b : \{T_B, Q_3, Pu_u, Athn^{n3}\}$ and transmits MS_b to D_x via open channel.

5) STEP AKA-5

D_x after receiving MS_b verifies if the condition $T_{DL} \geq |T_r - T_B|$. If the condition holds, D_x considers MS_b as a licit message. Moreover, D_x computes

$$K_D = H(T_B \parallel SP_{D_x} \parallel PID_{D_x}), \tag{44}$$
$$IV_D = K_D^a \oplus K_D^b, \tag{45}$$
$$(R_{U_e}, R'_G) = D_{K_D}\{(IV_D, Q_3\}, \tag{46}$$
$$Athn^{n4} = H(T_B \parallel R_{U_e} \parallel R'_G \parallel SP_{D_x} \parallel PID_{D_x}), \tag{47}$$

where K_D represents the secret key used in the decryption process, IV_D is the initialization vector, and $Athn^{n4}$ is the authentication parameter. In addition, D_x validates the condition $Athn^{n3} \stackrel{?}{=} Athn^{n4}$ to verify the authenticity of MS_b. If it holds, D_x consider the received message MS_b as licit message and continue the AKA process. Moreover, D_x chooses a timestamps T_C and random-number R_D of size 32 ans 128 bits, respectively and calculates

$$Pu_d = P \cdot S_{kd}, \tag{48}$$
$$k_d = S_{kd} \cdot Pu_u, \tag{49}$$
$$U_D = H(k_d \parallel T_C \parallel R_{U_e}), \tag{50}$$
$$IV_{D1} = (U_D^a \oplus U_D^b), \tag{51}$$
$$Q_4 = E_{U_D}\{(IV_{D1}), R'_G \oplus R_D\}, \tag{52}$$

where Pu_d represents the public key of D_x, k_d denotes the shared secret between D_x and U_e, U_D signifies the secret key used in the encryption process, IV_{D1} is the initialization vector, and Q_4 is obtained by employing AES-CBC-256. Moreover, D_x calculates

$$SK_{D_x} = H(H(R_{U_e} \parallel R'_G \oplus R_D) \parallel H(k_d \parallel T_C \parallel R'_G \oplus R_D) \times \parallel PID_{D_x}), \tag{53}$$

$$Athn^{n5} = H(H(R_{U_e} \parallel R'_G \oplus R_D) \parallel T_C \parallel SK_{D_x} \parallel PID_{D_x}), \tag{54}$$

Finally, D_x fabricates a message $MS_c : \{T_C, Q_4, Pu_d, Athn^{n5}\}$ and dispatches it to U_e via open channel. Furthermore, for indecipherable the communication with U_e, D_x computes SK as

6) STEP AKA-6

After getting MS_c from D_x, U_e verifies the condition $T_{DL} \geq |T_r - T_C|$ to determine the freshness of the received message MS_c. If MS_c is received within the predefined time delay limit, U_e consider MS_c as the valid message. In addition, U_e computes

$$k_u = S_{ku} \cdot Pu_d, \tag{55}$$
$$U_u = H(k_u \parallel T_C \parallel R_{U_e}), \tag{56}$$
$$IV_{u1} = (U_u^a \parallel U_u^b), \tag{57}$$
$$R'_G \oplus R_D = D_{U_u}\{(IV_{u1}), Q_4\}, \tag{58}$$

where k_u is shared secret between U_e and D_x, U_u is the secret key to perform the encryption, and IV_{u1} is the initialization vector. In addition, SD_e computes SK to achieve the indecipherable communication with D_x and authentication parameter as follows

$$SK_{U_e} = H(H(R_{U_e} \parallel R'_G \oplus R_D) \parallel H(k_d \parallel T_C \parallel R'_G \oplus R_D) \times \parallel PID_{D_x}), \tag{59}$$
$$Athn^{n6} = H(H(R_{U_e} \parallel R'_G \oplus R_D) \parallel T_C \parallel SK_{U_e} \parallel PID_{D_x}), \tag{60}$$

where $Athn^{n6}$ is the authentication parameter. Finally, to validate the authenticity of MS_c, U_e checks $Athn^{n5} \stackrel{?}{=} Athn^{n6}$, if holds, authentication is success-full. Furthermore, picks a new initialization vector IV_a^n and computes $CT_{xt}^n = E_{KR^l}\{(IV_a^n), PT_{xt}\}$. Finally, SD_e replaces IV_a^n and CT_{xt}^n with IV_a and CT_{xt} in its own memory. The AKA process of RUAM-IoD is depicted in Fig. 3.

E. BPC PHASE

In the proposed RUAM-IoD, U_e is allowed to change/update its bio-metric and password. To change/update the bio-metric and password information, U_e needs to perform the necessary steps.

1) STEP BPC-1

After receiving old secret parameters, such as $PAW_{U_e}^o$ and B_{in}^o (*both new and old bio-metric information are same*), SD_e performs the following computation to update the secret parameters, such as $PAW_{U_e}^o$ and B_{in}^o.

$$(BK_{U_e}^o) = Rep(B_{in}^o, rpp), \tag{61}$$
$$AR^o = (BK_{U_e}^o \oplus ID_{U_e}), \tag{62}$$
$$KR^o = (AR^o \parallel PAW_{U_e}^o), \tag{63}$$
$$PT_{xt} = D_{KR^o}\{(IV_a), CT_{xt}\}, \tag{64}$$

Remote User U_e/SD_e	**DSP Registration Center/Central Server** DRC/CS	**Drone** D_x
$\{CT_{xt}, Athn, rpp, Gen(.), ERT, Rep(.), IV_a\}$	$\{PID_{U_e}, CT_g, IV_{reg}\}$	$\{PID_{D_x}, P, SP_{D_x}\}$
inputs PAW_{U_e} and ID_{U_e}, imprints bio-metric B^l_{in}, computes $(BK^l_{U_e}) = Rep(B^l_{in}, rpp)$, $AR^l = (BK^l_{U_e} \oplus ID_{U_e})$, $KR^l = (AR^l \parallel PAW^l_{U_e})$, $PT_{xt} = D_{KR^l}\{(IV_a), CT_{xt}\}$, $PT_{xt} = \{PID_{U_e}, SP_{U_e}, PID_{D_x}, Y\}$, $Athn^l = H(PAW^l_{U_e} \parallel ID_{U_e} \parallel BK^l_{U_e} \parallel Y)$, checks $Athn^l = Athn$, if holds, picks T_A, R_{U_e}, and S_{ku}, computes $k = S_{ku} \cdot Pu_g$, $Pu_u = P \cdot S_{ku}$, $U = H(k \parallel T_A)$, $Q_1 = (Y \parallel PID_{U_e}) \oplus U$, $K_u = (Y \parallel SP_{U_e})$, $IV_u = U_a \oplus U_b$, $Q_2 = E_{K_u}\{(IV_u), R_{U_e}, PID_{D_x}\}$, $Athn^{n1} = H(PID_{U_e} \parallel PID_{D_x} \parallel Y \parallel R_{U_e} \parallel T_A)$, $\xrightarrow[(U_e \rightarrow CS)]{MS_a:\{T_A, Q_1, Q_2, Pu_u, Athn^{n1}\}}$.	verifies if $T_{DL} \geq \lvert T_r - T_A \rvert$, If so, computes $k1 = S_{kg} \cdot Pu_u$, $U_2 = H(k1 \parallel T_A)$, $(Y \parallel PID_{U_e}) = Q_1 \oplus U_2$, checks if PID_{U_e} exist in its database, if found, retrieves $\{CT_g, IV_{reg}\}$ related to PID_{U_e}, computes $K_g = (IV_{reg} \parallel Y)$, $PT_g = D_{K_g}\{(IV_{reg}), CT_g\}$, retrieves $PT_g = \{SP_{U_e}, (PID_{D_x}, SP_{D_x})\}$, computes $K_d = (Y \parallel SP_{U_e})$, $IV_{u2} = U^a_2 \oplus U^b_2$, $(R_{U_e}, PID_{D_x}) = D_{K_d}\{(IV_{u2}), Q_2\}$, $Athn^{n2} = H(PID_{U_e} \parallel PID_{D_x} \parallel Y \parallel R_{U_e} \parallel T_A)$, validates if $Athn^{n1} \stackrel{?}{=} Athn^{n2}$, if holds, picks T_B, R_G and computes $R^{'}_G = R_G \oplus PID_{U_e}$, $K_G = H(T_B \parallel SP_{D_x} \parallel PID_{D_x})$, $IV_G = K^a_G \oplus K^b_G$, $Q_3 = E_{K_G}\{(IV_G), R_{U_e}, R^{'}_G\}$, $Athn_{a3} = H(T_B \parallel R_{U_e} \parallel R^{'}_G \parallel SP_{D_x} \parallel PID_{D_x})$, $\xrightarrow[(CS \rightarrow D_x)]{MS_b:\{T_B, Q_3, Pu_u, Athn^{n3}\}}$.	verifies if $T_{DL} \geq \lvert T_r - T_B \rvert$, if holds, computes $K_D = H(T_B \parallel SP_{D_x} \parallel PID_{D_x})$, $IV_D = K^a_D \oplus K^b_D$, $(R_{U_e}, R^{'}_G) = D_{K_D}\{(IV_D), Q_3\}$, $Athn^{n4} = H(T_B \parallel R_{U_e} \parallel R^{'}_G \parallel SP_{D_x} \parallel PID_{D_x})$, validates $Athn^{n3} \stackrel{?}{=} Athn^{n4}$, if holds, picks T_C and R_D, computes $Pu_d = P \cdot S_{kd}$, $k_d = S_{kd} \cdot Pu_d$, $U_D = H(k_d \parallel T_C \parallel R_{U_e})$, $IV_{D1} = (U^a_D \oplus U^b_D)$, $Q_4 = E_{U_D}\{(IV_{D1}), R^{'}_G \oplus R_D\}$, $SK_{D_x} = H(H(R_{U_e} \parallel R^{'}_G \oplus R_D) \parallel H(k_d \parallel T_C \parallel R^{'}_G \oplus R_D) \parallel PID_{D_x})$, $Athn_{a5} = H(H(R_{U_e} \parallel R^{'}_G \oplus R_D) \parallel T_C \parallel SK_{D_x} \parallel PID_{D_x})$, $\xleftarrow[(U_e \leftarrow D_x)]{MS_c:\{T_C, Q_4, Pu_d, Athn^{n5}\}}$.	
verifies if $T_{DL} \geq \lvert T_r - T_C \rvert$, if holds, computes $k_u = S_{ku} \cdot Pu_d$, $U_u = H(k_u \parallel T_C \parallel R_{U_e})$, $IV_{u1} = (U^a_u \parallel U^b_u)$, $R^{'}_G \oplus R_D = D_{U_u}\{(IV_{u1}), Q_4\}$, where $R^{'}_G = R_G \oplus PID_{U_e}$, $SK_{U_e} = H(H(R_{U_e} \parallel R^{'}_G \oplus R_D) \parallel H(k_d \parallel T_C \parallel R^{'}_G \oplus R_D) \parallel PID_{D_x})$, $Athn^{n6} = H(H(R_{U_e} \parallel R^{'}_G \oplus R_D) \parallel T_C \parallel SK_{D_x} \parallel PID_{D_x})$, and checks $Athn^{n5} \stackrel{?}{=} Athn^{n6}$, if holds, authentication is success-full, picks new IV^n_a and computes $CT^n_{xt} = E_{KR^l}\{(IV^n_a), PT_{xt}\}$, replaces IV^n_a and CT^n_{xt} with IV_a and CT_{xt} in its own memory.		
$SK_{U_e}(= SK_{D_x}) = H(H(R_{U_e} \parallel R^{'}_G \oplus R_D) \parallel H(k_d \parallel T_C \parallel R^{'}_G \oplus R_D) \parallel PID_{D_x})$		

FIGURE 3. RUAM-IoD user AKA phase.

where $PT_{xt} = \{PID_{U_e}, SP_{U_e}, PID_{D_x}, Y\}$. In addition, SD_e computes $Athn^o = H(PAW^o_{U_e} \parallel ID_{U_e} \parallel BK^o U_e \parallel Y)$ and checks the condition $Athn^o \stackrel{?}{=} Athn$. If it is valid, SD_e notifies to U_e to select its new/fresh secret parameters, such as new password $PAW^n_{U_e}$ and fresh $Bio^n_{U_e}$.

2) STEP BPC-2

After receiving $PAW^n_{U_e}$ and $Bio^n_{U_e}$ from U_e. SD_e performs the following computations

$$(BK^n_{U_e}, rpp^n) = Gen(B^n_{in}), \tag{65}$$
$$AR^n = (BK^n_{U_e} \oplus ID_{U_e}), \tag{66}$$
$$KR^n = (AR^n \parallel PAW^n_{U_e}), \tag{67}$$
$$Athn^n = H(PAW^n_{U_e} \parallel ID_{U_e} \parallel BK^n_{U_e} \parallel Y), \tag{68}$$
$$CT^n_{xt} = E_{KR^n}\{(IV^n_a), PT_{xt}\}. \tag{69}$$

Finally, SD_e replaces the old stored credentials $\{CT_{xt}, Athn, rpp, Gen(.), ERT, Rep(.), IV_a\}$ with new credentials $\{CT^n_{xt}, Athn^n, rpp^n, Gen(.), ERT^n, Rep(.), IV^n_a\}$ in its own memory. The BPC phase is summarized in Fig. 4.

F. REVOCATION PHASE

It is assumed that a valid U_e of the IoD environment lost its SD_e. However, U_e can obtained new SD^n_e with new/fresh credentials from DRC and executes the following steps to perform the revocation (RvP) phase.

User U_e	**Smart-Device** SD_e
inputs $PAW^o_{U_e}$ and ID_{U_e}, imprints bio-metric B^o_{in}, $\xrightarrow{\{PAW_{U_e}, ID_{U_e}, B^o_{in}\}}$.	computes $(BK^o_{U_e}) = Rep(B^o_{in}, rpp)$, $AR^o = (BK^o_{U_e} \oplus ID_{U_e})$, $KR^o = (AR^o \parallel PAW^o_{U_e})$, $PT_{xt} = D_{KR^o}\{(IV_a), CT_{xt}\}$, $PT_{xt} = \{PID_{U_e}, SP_{U_e}, PID_{D_x}, Y\}$, $Athn^o = H(PAW^o_{U_e} \parallel ID_{U_e} \parallel BK^o_{U_e} \parallel Y)$, checks $Athn^o = Athn$, if holds, notifies to U_e to select new password $PAW^n_{U_e}$ and imprint new $Bio^n_{U_e}$.
inputs $PAW^n_{U_e}$ and $Bio^n_{U_e}$, $\xrightarrow{\{ID_{U_e}, PAW^n_{U_e}, B^n_{in}\}}$. replaces $\{CT_{xt}, Athn, rpp, Gen(.), ERT, Rep(.), IV_a\}$ with $\{CT^n_{xt}, Athn^n, rpp^n, Gen(.), ERT^n, Rep(.), IV^n_a\}$ in SD_e's memory.	computes $(BK^n_{U_e}, rpp^n) = Gen(B^n_{in})$, $AR^n = (BK^n_{U_e} \oplus ID_{U_e})$, $KR^n = (AR^n \parallel PAW^n_{U_e})$, $Athn^n = H(PAW^n_{U_e} \parallel ID_{U_e} \parallel BK^n_{U_e} \parallel Y)$, picks new IV^n_a, $CT^n_{xt} = E_{KR}\{(IV^n_a), PT_{xt}\}$.

FIGURE 4. PBC phase.

1) STEP RvP-1

After getting new SD^n_e from DRC, U_e inputs its secret parameters, such as PAW_{U_e} and ID_{U_e} and computes $RM_{rov} = H(ID_{U_e} \parallel PAW_{U_e})$. SD^n_e constructs a revocation message $M_{ro1} : \{RM_{ro1}\}$ to CS via a secure communication channel. After receiving M_{ro1}, CS computes the following computation

$$AA = H(S_{kg} \parallel IV_{reg} \parallel M_{ro1}), \tag{70}$$
$$SP_{U_e} = (AA_1 \oplus AA_2), \tag{71}$$
$$PID_{U_e} = (AA_1 \oplus SP_{U_e}). \tag{72}$$

CS checks the existence of PID_{U_e} in its own database. If PID_{U_e} is detected, CS removes the information associated with PID_{U_e} and dispatches a message to U_e for new registration.

2) STEP RvP-2

After getting the new registration request from U_e, DRC conducts the same procedure as accomplished in Step URG-2 under Section V-C. Subsequently, the new secret parameters are dispatched to U_e, $M_{ro3} : \{PID_{U_e}^{new}, SP_{U_e}^{new}, PID_{D_x}\}$. Upon receiving M_{ro3}, SD_e executes Step URG-3 of Section V-C and updates $\{CT_{xt}^{new}, Athn^{new}, rpp^{new}, Gen(.), ERT^{new}, Rep(.), IV_a^{new}\}$ in its own memory. In addition, CS stores the credentials $\{PID_{U_e}^{new}, CT_g^{new}, IV_{reg}^{new}\}$ in its own memory.

G. DYNAMIC DRONE DEPLOYMENT PHASE

The proposed RUAM-IoD renders the functionality of dynamic drone deployment (DDD) phase. DRC following step to deploy a new D_x^{new} drone in the target FZ.

1) STEP DDD-1

DRC chooses a random-number $R_{D_x}^{new}$ and a unique pseudo-identity $PID_{D_x}^{new}$ for D_x^{new}. Moreover, DRC calculates the secret parameter $SP_{D_x}^{new}$ for D_x^{new} as follows

$$G = H(R_{D_x}^{new} \parallel PID_{D_x}^{new}), \tag{73}$$

$$SP_{D_x}^{new} = (G_1 \oplus G_2), \tag{74}$$

where G_1 and G_2 are obtained after dividing G into two equal parts.

2) STEP DDD-2

Finally, DRC pre-loads the credentials $\{PID_{D_x}^{new}, P, SP_{D_x}^{new}\}$ in the memory of D_x^{new}. In addition, D_x^{new} has the access to all the public credentials of CS, such $\{Pu_g, EC(\alpha, \beta), P\}$.

VI. SECURITY ANALYSIS

This section presents the informal analysis of RUAM-IoD to demonstrates its immunity/resistance against different pernicious security vulnerabilities, such as CSI, UI, SSD, DoS, and BPC attacks. Furthermore, ROM-based analysis is conducted to prove SK's security, established between U_e and D_x. Moreover, Scyther is employed to illustrate that RUAM-IoD can resist or protect replay and MITM attacks.

A. INFORMAL SECURITY ANALYSIS

In this subsection, informal analysis of RUAM-IoD is conducted to show its effectiveness against the succeeding attacks.

1) BPC ATTACK

After procuring the information $\{CT_{xt}, Athn, rpp, Gen(.), ERT, Rep(.), IV_a\}$, which are pre-loaded in the memory of SD_e, $\mathcal{A}$ need to update the password of U_e. However, to update the password of U_e, $\mathcal{A}$ picks $PAW_{U_e}^{\mathcal{A}}$ and $ID_{U_e}^{\mathcal{A}}$, and $B_{in}^{\mathcal{A}}$ on behalf of U_e and perform the following computations $(BK_{U_e}^{\mathcal{A}}) = Rep(B_{in}^{\mathcal{A}}, rpp)$, $AR^{\mathcal{A}} = (BK_{U_e}^{\mathcal{A}} \oplus ID_{U_e})$, $KR^{\mathcal{A}} = (AR^{\mathcal{A}} \parallel PAW_{U_e}^{\mathcal{A}})$, $PT_{xt}^{\mathcal{A}} = D_{KR^{\mathcal{A}}}\{(IV_a), CT_{xt}\}$, and $Athn^{\mathcal{A}} = H(PAW_{U_e}^{\mathcal{A}} \parallel ID_{U_e}^{\mathcal{A}} \parallel BK_{U_e}^{\mathcal{A}} \parallel Y^{\mathcal{A}})$. To check if the decryption process is successful, $\mathcal{A}$ verifies the condition $Athn^{\mathcal{A}} = Athn$. Hovered, It is computationally infeasible for $\mathcal{A}$ to determine the secret credentials, such as PAW_{U_e} and ID_{U_e}, and B_{in} associated with U_e simultaneously. Therefore, $\mathcal{A}$ cannot perform theses computation successfully without the knowledge of PAW_{U_e} and ID_{U_e}, and B_{in} and canot update the password of U_e. Thus, the proposed RUAM-IoD is secured against BPC attack.

2) SSD ATTACK

Assume that $\mathcal{A}$ can obtain the lost/stolen smart device SD_e of U_e. $\mathcal{A}$ by employing PA attack can extricate the information $\{CT_{xt}, Athn, rpp, Gen(.), ERT, Rep(.), IV_a\}$, which are pre-loaded in the memory of SD_e. $\mathcal{A}$ cannot gain any confidential or secret information related to U_e because all the sensitive information are stored in the encrypted form. It is imperative for $\mathcal{A}$ to determine $KR = (AR \parallel PAW_{U_e})$, where $AR = (BK_{U_e} \oplus ID_{U_e})$ to make the encryption process successful. $\mathcal{A}$ requires to know ID_{U_e}, PAW_{U_e}, and BK_{U_e}, to compute KR. Computationally, it is impracticable for $\mathcal{A}$ to determine the bio-metric key BK_{U_e}, which is used in deriving KR. The secret key KR used to decrypt the encrypted information retrieved from SD_e's memory. Therefore, without knowing KR, Computationally, it is infeasible for $\mathcal{A}$ to extricate any sensitive information related to U_e after retrieving information form SD_e. Thus, RUAM-IoD is secured with respect to SSD attack.

3) MITM ATTACK

According to DY model, the adversary, $\mathcal{A}$, can capture, modify or compromise all the exchanged message, which are communicated over the wireless channel. During the AKA process, the communicated message are $MS_a : \{T_A, Q_1, Q_2, Pu_u, Athn^{n1}\}$, $MS_b : \{T_B, Q_3, Pu_u, Athn^{n3}\}$, and $MS_c : \{T_C, Q_4, Pu_d, Athn^{n5}\}$ Now, $\mathcal{A}$ may attempt to alter the content of the transmitted messages to make the message receiving entity believe that the received messages are from the legitimate entity. If $\mathcal{A}$ tries to reconstruct MS_a, $\mathcal{A}$ requires to alter the contents of $Q_1, Q_2, Pu_u, Athn^{n1}$, which requires the knowledge of PID_{U_e}, SP_{U_e}, and S_{ku}. Moreover, to reconstruct MS_b, $\mathcal{A}$ requires the knowledge SP_{D_x} and S_{ku}. S_{kd} and R_{U_e} are the necessitated parameters to regenerate MS_c. Therefore, it is impractical for $\mathcal{A}$ to regenerate a valid message without the knowing the secret credentials associated with a specific entity. Thus, RUAM-IoD can withstand MITM attack.

4) DoS ATTACK

Local authentication is necessary to prevent U_e from sending too many AKA requests to CS. To accomplish the local authentication, U_e requires to inputs its secret credentials, such as ID_{U_e}, PAW_{U_e}, and BK_{U_e} at the interface of SD_e and execute the following computations $(BK_{U_e}) = Rep(B_{in}, rpp)$,

$AR = (BK_{U_e} \oplus ID_{U_e})$, $KR = (AR \parallel PAW_{U_e})$, $PT_{xt} = D_{KR}\{(IV_a), CT_{xt}\}$, and $Athn = H(PAW_{U_e} \parallel ID_{U_e} \parallel BK_{U_e} \parallel Y)$. To check if the decryption process is successful, $\mathcal{A}$ checks the condition $Athn^l = Athn$. The condition will hold if $\mathcal{A}$ enters the valid secret login credentials. Otherwise, SD_e terminates the login process and does not send AKA requests to CS. Under this situation, local authentication prevents U_e from sending a large number of AKA requests to CS. Therefore, the proposed RUAM-IoD can resist DoS attack.

5) IMPERSONATION ATTACK

According to the DY model, $\mathcal{A}$ has the capability to expropriate all the exchanged messages, such as $MS_a : \{T_A, Q_1, Q_2, Pu_u, Athn^{n1}\}$, $MS_b : \{T_B, Q_3, Pu_u, Athn^{n3}\}$, and $MS_c : \{T_C, Q_4, Pu_d, Athn^{n5}\}$. $\mathcal{A}$ after capturing the MS_a can impersonates as U_e. However, to impersonate as a legitimate U_e, $\mathcal{A}$ requires to reconstruct a message to make believe CS that this reconstructed message is from a legitimate U_e of the system. However, to construct a licit MS_a, $\mathcal{A}$ requires to know the secret credentials, such as SP_{U_e} and S_{ku}. Moreover, It is computationally impracticable for $\mathcal{A}$ to procure the secret credentials of U_e. Thus, $\mathcal{A}$ cannot effectuate the UI attack. To reconstruct MS_b and MS_c $\mathcal{A}$ need to know the secret credentials of CS and D_x. $\mathcal{A}$ cannot impersonate as a legitimate CS and D_x in the communication system without the knowledge of secret credentials of CS and D_x. Therefore, RUAM-IoD is secured against UI, DI, and CSI attacks.

6) UA AND UNTRACEABILITY

$\mathcal{A}$ has the capability to expropriate all the exchanged messages, such as $MS_a : \{T_A, Q_1, Q_2, Pu_u, Athn^{n1}\}$, $MS_b : \{T_B, Q_3, Pu_u, Athn^{n3}\}$, and $MS_c : \{T_C, Q_4, Pu_d, Athn^{n5}\}$, which transmitted over the public communication channel during the AKA process. It is difficult for $\mathcal{A}$ to determine the real-identities of network entities from captured MS_a, MS_b, and MS_c. Therefore, the proposed RUAM-IoD can resist the IG guessing attack. In addition, MS_a, MS_b, and MS_c are randomly generated because they incorporate the latest timestamp and fresh random-number. After capturing two d messages from different AKA sessions it is hard for $\mathcal{A}$ to determine any significant information by correlating these two messages. Thus, RUAM-IoD ensures UA and untraceablity features.

7) REPLAY ATTACK

In RUAM-IoD, there are three messages are exchanged, i.e., $MS_a : \{T_A, Q_1, Q_2, Pu_u, Athn^{n1}\}$, $MS_b : \{T_B, Q_3, Pu_u, Athn^{n3}\}$, and $MS_c : \{T_C, Q_4, Pu_d, Athn^{n5}\}$ to accomplish the AKA process. MS_a, MS_b, and MS_c are exchanged over the public communication channel. According to the DY model, $\mathcal{A}$ can potentially capture, modify or compromise all the disseminated messages in the IoD environment. Now, $\mathcal{A}$ may attempt to replay the messages to excerpt some estimable information from network entities involved in the AKA process. Each message communicated during the AKA process is produced using the participant's latest timestamp and a fresh random-number. Therefore, the message receiving entity checks the validity of the timestamp. In case of an invalid timestamp, the message is contemplated as replayed message, and the message receiving network entity declines the validation of the replayed messages, restricting $\mathcal{A}$ from effectuating the replay attack.

8) PI ATTACK

In RUAM-IoD, DRC is contemplated as a fully trusted network entity and CS is considered as semi-trusted network entity. $\mathcal{A}$ can obtain the secret credentials associated with the legitimate U_e and D_x of the communication system and can effectuate any malicious attacks on the behalf of U_e and D_x. In RUAM-IoD, secret credentials related to U_e and D_x are stored in encrypted form and insider attacker cannot procure secret information related to U_e and D_x without knowing the secret key S_{kg} of CS, stored in temper proof database of CS. Therefore, RUAM-IoD is secured against PI attack.

9) DRONE CAPTURE ATTACK

In the IoD environment, it is tough to monitor the drone for all the time (24×7). Thus, $\mathcal{A}$ can capture some drones, which are deployed in the IoD environment. After capturing a drone $\mathcal{A}$ can extract the sensitive information, such as $\{PID_{D_x}, P, SP_{D_x}\}$. Since all the drones are assigned with distinct and unique secret parameters. Therefore, the secret parameters of compromised drones are not useful to derive SK, which is established between the non-compromised drone and U_e. Thus, RUAM-IoD is secured against drone capture attack.

10) ESL ATTACK

Adversary under the CK-adversary model can compromise the secret credentials (LTS and STS) and session state, in addition to actions permitted under the DY model. In RUAM-IoD, the session key $SK_{U_e}(= SK_{D_x}) = H(H(R_{U_e} \parallel R'_G \oplus R_D) \parallel H(k_d \parallel T_C \parallel R'_G \oplus R_D) \parallel PID_{D_x})$ is constructed using both STS and LTS secret credentials. By compromising STS, $\mathcal{A}$ will not be able to construct the session key $SK_{U_e}(= SK_{D_x})$, which is established between U_e and D_x. Similarly, by compromising LTS, $\mathcal{A}$ will be able to derive the session key $SK_{U_e}(= SK_{D_x})$. Therefore, to derive the session key $SK_{U_e}(= SK_{D_x})$, $\mathcal{A}$ requires to know both LTS and STS secret credentials, which is a computationally expensive task for $\mathcal{A}$. Thus, RUAM-IoD is resistant to ESL attack.

B. FORMAL SECURITY VERIFICATION THROUGH ROM

In our proposed scheme, ROM-based formal method is employed to prove the security strength of the SK that is established during the AKA phase. It is worth noting that a total of three participants, i.e., U_e, CS, and D_x, play roles during the AKA process. Moreover, from theorem 1, we verify that $\mathcal{A}$ is unable to determine the SK, which is determined and set up between the network entities, U_e and D_x, by means of CS. ROM has the following components, which are associated with the different queries, accessed by $\mathcal{A}$.

TABLE 3. Description/explanation of various queries used by adversary under ROM.

Query	Description
$Send(\pi^{p1}, Msg)$	This query enables $\mathcal{A}$ to effectuate an active attack by transmitting a message Msg to π^{p1} and also π^{p1} sends response accordingly.
$Test(\pi^{p1})$	$\mathcal{A}$ by using this query can check whether an SK, which is established between network entities, is the actual key or random output like the output of a flipped coin, say B.
$Reveal(\pi^{p1})$	$\mathcal{A}$ by effectuating this query can procure an SK, which is established during the AKA process between π^{p1} and its partner network entity.
$CorruptSD(\pi^{p1}_{U_e})$	$\mathcal{A}$ by effectuating this query can extricate secret credential, which are stored in SD_{U_e}'s memory by employing PA attack.
$CorruptD(\pi^{p3}_{D_x})$	$\mathcal{A}$ by effectuating this query can extricate secret credential, which are stored in D_x's memory by employing PA attack.
$Execute(\pi^{p1}_{U_e}, \pi^{p2}_{CS}, \pi^{p3}_{D_x})$	This query enables $\mathcal{A}$ to eavesdrop on all the messages, which are communicated messages among network entities, such as U_e, CS, and D_x.

1) PARTICIPANTS

The instances $p1$, $p2$, and $p3$ of U_e, CS, and D_x are shown by $\pi^{p1}_{U_e}$, π^{p2}_{CS}, and $\pi^{p3}_{D_x}$, respectively, which are also deemed as random oracles.

2) FRESHNESS

$\pi^{p1}_{U_e}$ and $\pi^{p3}_{D_x}$ are deemed to be fresh if the SK established between U_e and D_x is not known to $\mathcal{A}$ when $\mathcal{A}$ performs *Reveal* query, as explained in Table 3.

3) ACCEPTED STATE

The instance π^p is deemed to be in accepted state when it receives the last expected message while carrying out the AKA process. In addition to this, *Sid* symbolizes the session identifier of π^p for the present AKA session. It is worth noting that *Sid* is created by concatenating the exchanged messages generated in sequence by π^p.

4) PARTNERING

Two instances π^{p1} and π^{p2} are considered to be partners in case the three subsequent conditions are simultaneously fulfilled: 1) π^{p1} and π^{p2} need to exchange the common *Sid* after authenticating each other conjointly, 2) π^{p1} and π^{p2} need to be in accepted states, and 3) π^{p1} and π^{p2} need to be interdependent partners.

5) ADVERSARY

DY model stipulates that $\mathcal{A}$ has the capabilities to seize all the messages disseminated among the entities in the IoD environment. This implies that $\mathcal{A}$, by means of the queries defined in Table 3, can modify, inject, and delete the communicated messages.

Moreover, this also implies that $\mathcal{A}$ has the capability to access the hash function $H(.)$. It is worth noting that $H(.)$ is modeled as a random-oracle, say RSH. Above this, the queries, which are defined in Table 3, are exploited by $\mathcal{A}$ to simulate an attack.

Definition 2: Elliptic Curve Discrete Logarithm Problem (ECDLP): For any $Pu_g = S_{kg} \cdot P$, $Ad^{ECDLP}(TP_{oly})$ is the for $\mathcal{A}$'s advantage or the probability to derive Skg from Pu_g within polynomial-time TP_{oly}. It is hard for $\mathcal{A}$ to determine Skg from Pu_g within polynomial-time, which makes $Ad^{ECDLP}(TP_{oly})$ trivial and defined as the elliptic curve discrete logarithm problem (ECDLP).

Definition 3 (Semantic Security): Let B is the correct bit and B' is the guessed by $\mathcal{A}$. If condition $B = B'$ holds, $\mathcal{A}$ wins the game. If $Pb[Succ]$ signify the probability of success, $\mathcal{A}$'s advantage in breaching SK's security, established while executing the AKA phase of RUAM-IoD is represented by $Ad^{RUAM-IoD}_{\mathcal{A}} = 2 \cdot |Pb[Succ] - 1|$. RUAM-IoD is protected if $Ad^{RUAM-IoD}_{\mathcal{A}}$ is trivial under the ROM.

The proof of SK's security of the proposed RUAM-IoD is presented in Theorem 1.

Theorem 1: Let $Ad^{RUAM-IoD}_{\mathcal{A}}(TP_{oly})$ is advantage of $\mathcal{A}$ running against the proposed RUAM-IoD in TP_{oly} to compromise the SK's security, which is established between U_e and D_x. If Q_h designates SHA-256 quires, $|RSH|$ indicates output size of SHA-256, Q_{CS} denotes the send queries, Lth represents the size of $BK^l_{U_e}$, $|PSWD|$ symbolizes the dictionary of passwords, $Ad^{IND-CPA}_{\Omega,\mathcal{A}}(l)$ signifies $\mathcal{A}$'s advantaged to breach the security of AES-CBC-256 in TP_{oly} (Definition 1), and $Ad^{ECDLP}_{\mathcal{A}}(TP_{oly})$ designates the advantage in compromising ECDLP (Definition 2). $\mathcal{A}$'s advantage to compromise the SK's security, which is set up between U_e and D_x while executing the proposed RUAM-IoD can be defined as:

$$Ad^{RUAM-IoD}_{\mathcal{A}}(TP_{oly}) \leq \frac{H_Q^2}{|RSH|} + \frac{Q_{CS}}{2^{Lth-1}|PWD|} + 2 \cdot Ad^{IND-CPA}_{\Omega,\mathcal{A}}(l) + 2 \cdot Ad^{ECDLP}_{\mathcal{A}}(TP_{oly}). \quad (75)$$

Proof: Following five games ($GM_x | x = 0, 1, 2, 3, 4$) are utilized to prove Theorem 1. We follow the same method to prove Theorem 1 as in [12].

GM_0 : This game is associated with real attack, which is executed by $\mathcal{A}$ against RUAM-IoD in the ROM. It is imperative for $\mathcal{A}$ to select the bit B at the beginning of GM_0. The semantic security of RUAM-IoD renders the following:

$$Ad^{RUAM-IoD}_{\mathcal{A}}(TP_{oly}) = |2 \cdot Pb[Succ0] - 1|. \quad (76)$$

GM_1 : An eavesdropping attack is effectuated in this game, in which $\mathcal{A}$ can expropriate all the messages, such as MS_a : $\{T_A, Q_1, Q_2, Pu_u, Athn^{n1}\}$, MS_b : $\{T_B, Q_3, Pu_u, Athn^{n3}\}$, and MS_c : $\{T_C, Q_4, Pu_d, Athn^{n5}\}$ by using the *Execute* query defined in Table 3, which are exchanged during execution of the AKA process. Upon the completion of this game, $\mathcal{A}$ required to make the *Reveal* query along with *Test* query to determine whether the derived SK is the correct key or a random key. In the proposed RUAM-IoD is computed as $SK_{U_e}(= SK_{D_x}) = H(H(R_{U_e} \parallel R'_G \oplus R_D) \parallel H(k_d \parallel T_C \parallel R'_G \oplus R_D) \parallel PID_{D_x})$, which is the amalgamation of both the LTS and STS parameters. Therefore, $\mathcal{A}$ requires knowing both STS parameters, such as R_{U_e}, R_G, R_D, S_{ku}, and S_{kd} and LTS parameters, such as PID_{D_x} and PID_{U_e} to construct a valid $SK_{U_e}(= SK_{D_x})$. Therefore, only by capturing the communicated message, such as MS_a, MS_b, and MS_c, $\mathcal{A}$'s winning

possibility/probability of GM_1 is not enhance at all. Thus, both the games GM_1 and GM_2 remains indistinguishable. So, we get

$$Pb[Succ0] = Pb[Succ1]. \tag{77}$$

GM_2 : In this game, $\mathcal{A}$ launches an active attack, which incorporates the *Send* and *RSH* oracles and attempts to convince a specific network entity to receive the modified message. In addition, $\mathcal{A}$ can implement any number of queries to find a collision in the hash digest. However, all the exchanged messages are protected by the irreversible and collision-resistant SHA-256. Therefore, it is infeasible for $\mathcal{A}$ to attain the collision in the output (hash digest) produced by SHA-256. Then, by birthday paradox, the succeeding result is achieved:

$$|Pb[Succ1] - Pb[Succ2]| \leq \frac{H_Q^2}{2 \cdot |RSH|}. \tag{78}$$

GM_3 : *CorruptSD* query is implemented in this game. Therefore, $\mathcal{A}$ can extract all the sensitive information, such as $\{CT_{xt}, Athn, rpp, Gen(.), ERT, Rep(.), IV_a\}$, which are preloaded in the memory of SD_e employing PA attack. $\mathcal{A}$, from the extracted information cannot procure any useful information because the secret information assigned to U_e are stored in the encrypted form. Therefore, $\mathcal{A}$ need to decrypt CT_x to procure the secret parameters. However, to make the decryption process successful, $\mathcal{A}$ requires to compute the secret key $KR = (AR \parallel PAW_{U_e})$, which is used for the encryption process. The secret key KR is the amalgamation of ID_{U_e}, PAW_{U_e}, and BK_{U_e}. The guessing probability of the bio-metric key BK_{U_e} is $\frac{1}{2^{Lth}}$, which is negligible. Thus, it is impractical for $\mathcal{A}$ to get any secret parameter by extracting the information from the memory of SD_e. In addition, U_e is permitted to make a restricted number of wrong password attempts. Under these conditions, GM_2 and GM_3 are indistinguishable in the exclusion of guessing attack; the subsequent result is procured:

$$|Pb[Succ2] - Pb[Succ3]| \leq \frac{Q_{CS}}{2^{Lth}|PWD|}. \tag{79}$$

GM_4 : This is the final game, $\mathcal{A}$ will try to derive the session key $SK_{U_e}(= SK_{D_x})$, which is establish between U_e and D_x by eavesdropping all the exchanged message, such as MS_a, MS_b, and MS_c. In the proposed RUAM-IoD, the session key is constructed as $SK_{U_e}(= SK_{D_x}) = H(H(R_{U_e} \parallel R'_G \oplus R_D) \parallel H(k_d \parallel T_C \parallel R'_G \oplus R_D) \parallel PID_{D_x})$, where $k_d = S_{kd} \cdot Pu_u$. It is impractical for $\mathcal{A}$ to derive S_{ku} from the public key of user Pu_u and S_{kd} from public key of drone Pu_d in polynomial time and is referred to ECDLP problem in ECC (Definition 2). In addition, the secret parameters, such as R_{U_e}, R_G, and R_D are exchanged among the network entities in encrypted form. In RUAM-IoD, AES-CBC-256 is used as the encryption algorithm, which is secure (IND-CPA secure) to use and $\mathcal{A}$ cannot breach the security of AES-CBC-256 in polynomial time (Definition 1. Therefore, it is hard for $\mathcal{A}$ to derive $SK_{U_e}(= SK_{D_x})$. So, both the games GM_3 and GM_4 remain indistinguishable in the absence of breaching the security of AES-CBC-256 and solving the ECDLP. The following result can be achieved:

$$|Pb[Succ3] - Pb[Succ4]| \leq Ad_{\Omega,\mathcal{A}}^{IND-CPA}(l) + Ad_{\mathcal{A}}^{ECDLP}(TP_{oly}). \tag{80}$$

$\mathcal{A}$ has accomplished all the queries. Therefore, $\mathcal{A}$ requires to determine the bit B' in order to win the game after executing the *Test* query. It is then obvious that

$$Pb[Succ4] = 1/2. \tag{81}$$

From (76) and (77), we get

$$Ad_{\mathcal{A}}^{RUAM-IoD}(TP_{oly}) = |2 \cdot Pb[Succ0] - \frac{1}{2}|. \tag{82}$$

From (82), we get

$$\frac{1}{2} \cdot Ad_{\mathcal{A}}^{RUAM-IoD}(TP_{oly}) = |Pb[Succ0] - \frac{1}{2}|. \tag{83}$$

By using (81) and (83), we obtain

$$\frac{1}{2} \cdot Ad_{\mathcal{A}}^{RUAM-IoD}(TP_{oly}) = |Pb[Succ1] - Pb[Succ4]| \tag{84}$$

By utilizing the triangular inequality, following

$$\begin{aligned} &|Pb[Succ1] - Pb[Succ4]| \\ &\leq |Pb[Succ1] - Pb[Succ2]| \\ &\quad + |Pb[Succ2] - Pb[Succ4]| \\ &\leq |Pb[Succ1] - Pb[Succ2]| + |Pb[Succ2] - Pb[Succ3]| \\ &\quad + |Pb[Succ3] - Pb[Succ4]|. \end{aligned} \tag{85}$$

By utilizing (78), (79), (81), and (85), we get

$$\begin{aligned} &\frac{1}{2}.Ad_{\mathcal{A}}^{RUAM-IoD}(TP_{oly}) \\ &= \frac{H_Q^2}{2 \cdot |RSH|} + \frac{Q_{CS}}{2^{Lth}|PWD|} \\ &\quad + Ad_{\Omega,\mathcal{A}}^{IND-CPA}(l) + Ad_{\mathcal{A}}^{ECDLP}(TP_{oly}). \end{aligned} \tag{86}$$

Hence, from equation (86), we get

$$\begin{aligned} &Ad_{\mathcal{A}}^{RUAM-IoD}(TP_{oly}) \\ &\leq \frac{H_Q^2}{|RSH|} + \frac{Q_{CS}}{2^{Lth-1}|PWD|} \\ &\quad + 2 \cdot Ad_{\Omega,\mathcal{A}}^{IND-CPA}(l) + 2 \cdot Ad_{\mathcal{A}}^{ECDLP}(TP_{oly}). \end{aligned} \tag{87}$$

■

C. SECURITY EVALUATION USING SCYTHER TOOL

Fig. 5 exhibits the result generated through Scyther tool-based formal security validation. Scyther is utilized extensively to prove the security perspectives of any security protocol in an automated way. Compared to other security protocol validation tools, such as Pro-Verify and AVISPA, Scyther is more commonly employed by the researcher to validate the security of the proposed AKA schemes. One of the advantages of Scyther is that it is based on the DY

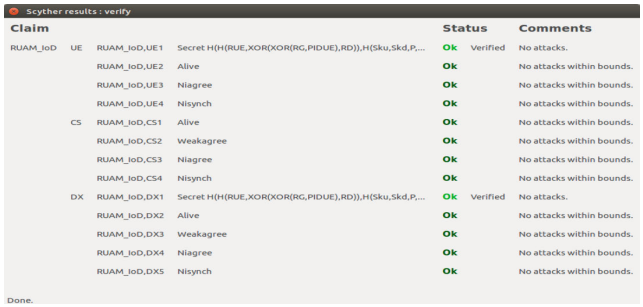

FIGURE 5. Scyther-based analysis shows that RUAM-IoD is secure.

adversarial model and the simulation results it generates to ensure that the secret parameters are not disclosed while executing the AKA scheme.

Since Scyther uses the security protocol description language (SPDL), a python-like language, for the description of security protocols hence, RUAM-IoD is coded in SPDL. To this end, three roles are defined in the SPDL script, which are U_e, CS, and D_x. In addition, there are different claims in SPDL generated either manually or automatically. Scyther facilitates describing and verifying these claims. For instance, the ''Alive claim'' guarantees that a network entity has accomplished some events. ''Nisynch claims'' guarantees that all the communicated messages between two network entities are delivered successfully. ''Weak-agree'' ensures the AKA scheme is protected against the impersonation attack. All these automatically generated claims are verified according to the procedure shown in Fig. 5. In addition, the manually generated claim, such as $claim(UE, Secret, SKU)$ and $claim(DX, Secret, SKD)$ are also verified, which indicates that an attacker cannot determine the secret SK. Fig. 5 indicates that the proposed RUAM-IoD is safe and an attacker cannot find any vulnerability.

VII. PERFORMANCE EVALUATION

RUAM-IoD is compared with the existing AKA scheme, such as Wazid *et al.* [18], Sutrala *et al.* [35], and Jangirala *et al.* [32]. The performance of RUAM-IoD is measured in terms of computational, memory/storage, and communication costs. We utilize the widely-accepted ''Multi-precision Integer and Rational Arithmetic Cryptographic Library (MIRACL)'' to conduct the experimental evaluation for different cryptographic primitives. This will enable us to estimate the computational time of the cryptographic primitives on the succeeding two environments (platforms):

1) We consider the settings (platform ($PF-1$)): Intel(R) Core(TM) i7-6700 with CPU: 3.40 GHz, RAM: 8 GB, OS: Ubuntu 16.04 LTS, 64-bit to simulate the server (CS) type environment.
2) The settings (platform ($PF-2$)) are considered for simulating the drone (D_x) and user (U_e): Raspberry Pi (RP-3) with CPU: Quad-core@1.2 GHz (64 bits), RAM: 1 GB, and OS: Ubuntu 16.04 LTS (64-bit). In the existing literature, the same environment is used to conduct experiments on resource-constricted devices [46], [47].

TABLE 4. Cryptographic primitives with computational time.

Notations	Cryptographic Primitive	PF-2	PF-1
T_{EPM}	ECC point multiplication	2.92 ms	0.605 ms
T_{EPA}	ECC point addition	0.154 ms	0.004 ms
T_{HF}	Hash Function	0.311 ms	0.029 ms
T_{ENC}	AES-CBC-256 (Encryption scheme)	0.425 ms	0.036 ms
$T_{FE} \approx T_{EPM}$	Fuzzy Extractors	2.92 ms	0.605 ms

TABLE 5. Comparison of security features.

Features	Sutrala *et al.* [35]	Wazid *et al.* [18]	Jangirala *et al.* [32]	RUAM-IoD
SSD	✓	✓	✓	✓
PG	✓	✓	✓	✓
CSI	✓	✓	✓	✓
DSY	×	×	✓	✓
Replay attack	✓	✓	✓	✓
PI	✓	✓	✓	✓
RvP	✓	×	✓	✓
DI	✓	✓	×	✓
UI	✓	✓	×	✓
UA	✓	✓	×	✓
MITM	✓	✓	✓	✓
Design flaw	✓	✓	×	✓
ROM	✓	×	✓	✓

Note: ✓: represents the supported features; × : signify the not defended features

Each cryptographic (algorithm) primitive is executed for 100 time for $PF-1$ and $PF-2$ to procure the average computational time different cryptographic primitives. Table 4 provides the average computational time of various cryptographic primitives.

A. SECURITY FEATURES COMPARISON

In this subsection, we compare the security feature of RUAM-IoD with Wazid *et al.* [18], Sutrala *et al.* [35], and Jangirala *et al.* [32]. To this end, a comparative analysis of the security features of RUAM-IoD and the related scheme is presented in Table 5. It is shown in the table that the scheme of Sutrala *et al.* [35] cannot resist DSY attack, and the scheme of Wazid *et al.* [18] is susceptible to DSY attack and does not render ROM-based analysis and RvP phase. The scheme of Jangirala *et al.* [32] is susceptible to MITM, UI, parallel session, DI, and SK compromise attacks and does not render the anonymity and untraceablity features. In contrast, RUAM-IoD is secured against the DSY, UI, DI, and SK compromised attacks.

B. COMPUTATIONAL COST COMPARISON

Computational cost denotes the CPU time required by a security scheme to complete its AKA process. Thus, without losing the security features, minimizing the computational cost is a critical design goal of AKA or security schemes. Table 4 presents the computational cost of various cryptographic primitives, which are used to compute the computational cost of RUAM-IoD and the related AKA schemes. The computational cost at user side in the proposed RUAM-IoD is $8T_{HF} + 4T_{ENC} + 3T_{EPM} + T_{FE} \approx [15.825]$ ms, while

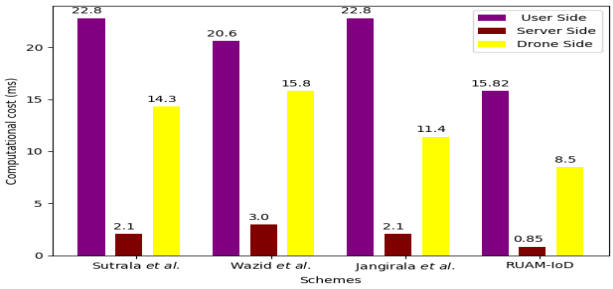

FIGURE 6. Computational cost at U_e, CS, and D_x side.

TABLE 6. Comparison of computational cost.

Protocol/Scheme	Computational Time Required to Perform AKA Phase
Sutrala *et al.* [35]	$33T_{HF} + 13T_{EPM} + 5T_{EPA} + T_{FE} \approx [39.21]$ **ms**
Wazid *et al.* [18]	$32T_{HF} + 13T_{EPM} + 3T_{EPA} + T_{FE} \approx [39.59]$ **ms**
Jangirala *et al.* [32]	$35T_{HF} + 11T_{EPM} + 4T_{EPA} + T_{FE} \approx [36.34]$ **ms**
RUAM-IoD	$20T_{HF} + 9T_{ENC} + 6T_{EPM} + T_{FE} \approx [25.282]$ **ms**

Sutrala *et al.* [35], Wazid *et al.* [18], and Jangirala *et al.* [32] require $16T_{HF} + 5T_{EPM} + 2T_{EPA} + T_{FE} \approx [22.804]$ ms, $19T_{HF} + 4T_{EPM} + T_{EPA} + T_{FE} \approx [20.66]$ ms, and $16T_{HF} + 5T_{EPM} + 2T_{EPA} + T_{FE} \approx [22.804]$ ms. RUAM-IoD requires less computational cost at the user side than the related AKA schemes, as shown in Fig. 6. The CS, stationed at the DRC of DSP, is a critical component in the IoD environment. So, it is desirable to reduce the computational cost at CS. The computational cost at CS side in the proposed RUAM-IoD is $5T_{HF} + 3T_{ENC} + T_{EPM} \approx [0.858]$ ms, while Sutrala *et al.* [35], Wazid *et al.* [18], and Jangirala *et al.* [32] require $9T_{HF} + 3T_{EPM} + 2T_{EPA} \approx [2.084]$ ms, $T_{HF} + 5T_{EPM} + T_{EPA} \approx [3.058]$ ms, and $11T_{HF} + 3T_{EPM} + T_{EPA} \approx [2.138]$ ms. So, RUAM-IoD incurs lesser computational cost than the related AKA schemes as shown in Fig. 6. Aside from this, Fig. 7 shows that the computational cost increases at CS, in all the schemes, as the number of authentication (user) requests increases at CS. However, RUAM-IoD reduces the computational cost in comparison to the other schemes. In the proposed RUAM-IoD, the computational cost at drone (D_x) or sensor node $6T_{HF} + 2T_{ENC} + 2T_{EPM} \approx [8.55]$ ms, while Sutrala *et al.* [35], Wazid *et al.* [18], and Jangirala *et al.* [32] require $8T_{HF} + 4T_{EPM} + T_{EPA} \approx [14.32]$ ms, $12T_{HF} + 4T_{EPM} + T_{EPA} \approx [15.87]$ ms, and $8T_{HF} + 3T_{EPM} + T_{EPA} \approx [11.40]$ ms. Fig. 6 also shows that RUAM-IoD needs less computational resources at the drone side than required by the related AKA schemes. This implies that RUAM-IoD is suitable for drone environment because drone being a resource-constrained device requires a reduced level of computational cost. In addition, Table 3 and Fig. 8 illustrate the total computational cost required to accomplish the AKA process of RUAM-IoD.

C. COMMUNICATION COST COMPARISON

Communication cost signifies the number of communicated messages (bits) transmitted to perform the AKA process.

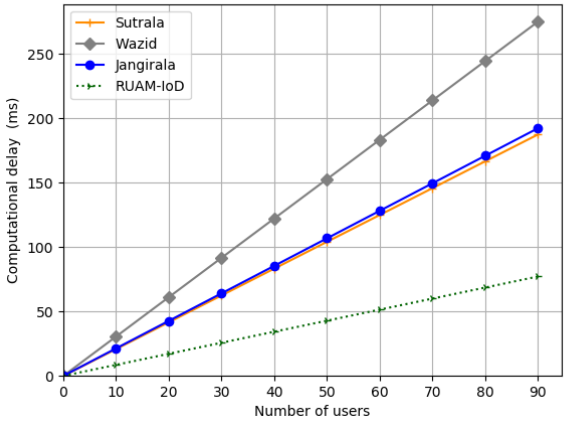

FIGURE 7. Computational delay at CS with increasing the AKA requests.

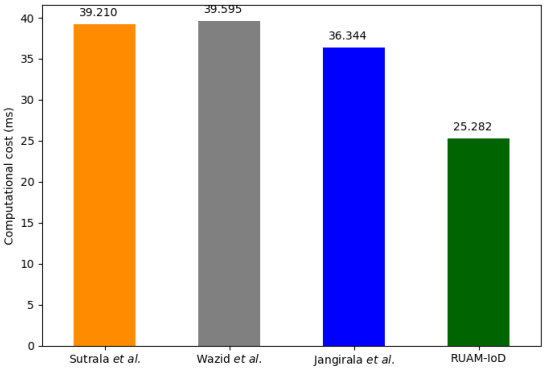

FIGURE 8. Computational cost to accomplish the AKA process (single user).

Therefore, it is essential to reduce the communication cost required to accomplish AKA process without risking the security traits of a security scheme. In the proposed RUAM-IoD, during the AKA process, the communicated messages are $MS_a : \{T_A, Q_1, Q_2, Pu_u, Athn^{n1}\}$, $MS_b : \{T_B, Q_3, Pu_u, Athn^{n3}\}$, and $MS_c : \{T_C, Q_4, Pu_d, Athn^{n5}\}$. The length MS_a, MS_b, and MS_c is $\{32 + 256 + 256 + 160 + 256\} = 960$ bits, $\{32 + 256 + 160 + 256\} = 704$ bits, and $\{32 + 128 + 160 + 256\} = 576$ bits, respectively. Thus, the total communication cost required by RUAM-IoD to accomplish the AKA phase is $\{960 + 704 + 576\} = [2240]$ bits. Contrarily, Wazid *et al.* [18], Sutrala *et al.* [35], and Jangirala *et al.* [32] require [3360] bits, [3200] bits, and [2656] bits, respectively. So, it is evident from Table 7 and Fig. 9 that RUAM-IoD demands less communication cost than demanded by the related AKA protocols. Fig. 10 illustrates the communication cost incurred when multiple users need to obtain the real-time information from a particular D_x concurrently.

D. STORAGE COST COMPARISON

As drones are resource-constricted devices with limited storage/memory resources, diminishing its memory utilization is the pressing need when designing an AKA protocol.

TABLE 7. Communication cost comparison.

Protocol/Scheme	Communicated Messages During AKA Phase	Total
Sutrala *et al.* [35]	$U_e \xrightarrow{1152} CS/GS \xrightarrow{1024} D_x/SN_x \xrightarrow{1024} U_e$	[3200] bits
Wazid *et al.* [18]	$U_e \xrightarrow{1152} CS/GS \xrightarrow{1184} D_x/SN_x \xrightarrow{1024} U_e$	[3360] bits
Jangirala *et al.* [32]	$U_e \xrightarrow{1152} CS/GS \xrightarrow{672} D_x/SN_x \xrightarrow{832} U_e$	[2656] bits
RUAM-IoD	$U_e \xrightarrow{704} CS/GS \xrightarrow{672} D_x/SN_x \xrightarrow{480} U_e$	[2240] bits

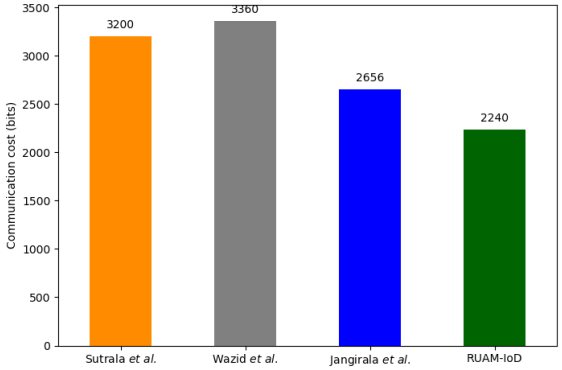

FIGURE 9. Communication cost required to execute the AKA phase (single user).

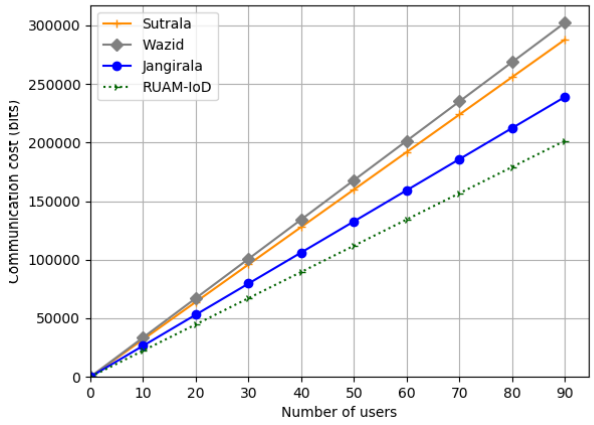

FIGURE 10. Communication cost at D_x side with increasing user authentication requests.

In RUAM-IoD, three entities are involved in the accomplishment of AKA process. These entities include U_e, CS, and D_x. Moreover, in RUAM-IoD, U_e, CS, and D_x are required to store $\{CT_{xt}, Athn, rpp, Gen(.), ERT, Rep(.), IV_a\}$, $\{PID_{U_e}, CT_g, IV_{reg}\}$, and $\{PID_{D_x}, SP_{D_x}, P\}$, respectively. U_e, CS, and D_x have to store $\{512+256+160+8+128\} = [1064]$ bits, $\{128 + 384 + 128\} = 640$ bits, and [256] bits, respectively. This way, RUAM-IoD demands a storage/memory capacity of $\{1064 + 640 + 256\} = [1948]$ bits. Contrarily, the AKA scheme of Wazid *et al.* [18], Sutrala *et al.* [35], and Jangirala *et al.* [32] require to store [4696] bits, [4320] bits, and [1768] bits, respectively. This comparison is more visibly illustrated in Fig. 11 wherein RUAM-IoD needs fewer memory/storage cost than Wazid *et al.* [18] and

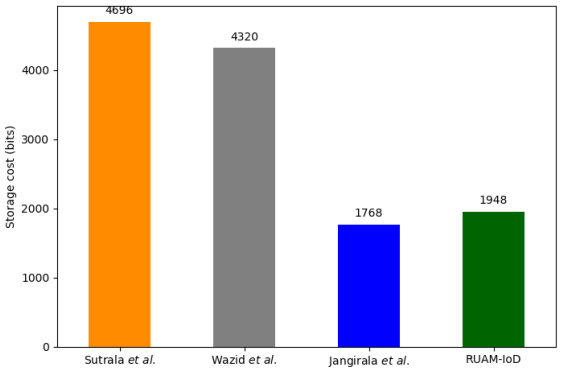

FIGURE 11. Storage cost comparison.

Sutrala *et al.* [35] with a marginal increment in memory/storage cost compared to Jangirala *et al.* [32].

VIII. CONCLUSION

This paper has presented an AKA scheme, called RUAM-IoD, for securing the communication between a remote user and a drone. To this end, RUAM-IoD checks the authenticity of a remote user before allowing him to access, in real-time, the sensitive information from a drone deployed in a particular FZ. After validating the authenticity of the remote user, RUAM-IoD establishes an SK between the user and the drone to make their communication indecipherable. The effectiveness of RUAM-IoD is verified against various security attacks through informal analysis. Furthermore, the security of the established SK is validated using ROM-based formal analysis. In addition, Scyther-based validation is performed on RUAM-IoD that demonstrated that the RUAM-IoD is secure against various security attacks. Furthermore, the performance analysis demonstrated that RUAM-IoD requires less computational, storage, and communication cost without compromising the security features.

REFERENCES

[1] M. M. Azari, G. Geraci, A. Garcia-Rodriguez, and S. Pollin, "UAV-to-UAV communications in cellular networks," *IEEE Trans. Wireless Commun.*, vol. 19, no. 9, pp. 6130–6144, Sep. 2020.
[2] A. Noorwali, M. A. Javed, and M. Z. Khan, "Efficient UAV communications: Recent trends and challenges," *Comput., Mater. Continua*, vol. 67, no. 1, pp. 463–476, 2021.
[3] H. Shakhatreh, A. H. Sawalmeh, A. Al-Fuqaha, Z. Dou, E. Almaita, I. Khalil, N. S. Othman, A. Khreishah, and M. Guizani, "Unmanned aerial vehicles (UAVs): A survey on civil applications and key research challenges," *IEEE Access*, vol. 7, pp. 48572–48634, 2019.
[4] J. Wang, Z. Na, and X. Liu, "Collaborative design of multi-UAV trajectory and resource scheduling for 6G-enabled Internet of Things," *IEEE Internet Things J.*, vol. 8, no. 20, pp. 15096–15106, Oct. 2021.
[5] J. Srinivas, A. K. Das, N. Kumar, and J. J. 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, Oct. 2019.
[6] M. Gharibi, R. Boutaba, and S. L. Waslander, "Internet of Drones," *IEEE Access*, vol. 4, pp. 1148–1162, 2016.
[7] R. J. Hall, "An Internet of Drones," *IEEE Internet Comput.*, vol. 20, no. 3, pp. 68–73, May/Jun. 2016.
[8] S. Atoev, O.-H. Kwon, S.-H. Lee, and K.-R. Kwon, "An efficient SC-FDM modulation technique for a UAV communication link," *Electronics*, vol. 7, no. 12, p. 352, Nov. 2018.

[9] M. Yahuza, M. Y. I. Idris, I. B. Ahmedy, A. W. A. Wahab, T. Nandy, N. M. Noor, and A. Bala, "Internet of Drones security and privacy issues: Taxonomy and open challenges," *IEEE Access*, vol. 9, pp. 57243–57270, 2021.

[10] A. Fotouhi, H. Qiang, M. Ding, M. Hassan, L. G. Giordano, A. Garcia-Rodriguez, and J. Yuan, "Survey on UAV cellular communications: Practical aspects, standardization advancements, regulation, and security challenges," *IEEE Commun. Surveys Tuts.*, vol. 21, no. 4, pp. 3417–3442, 4th Quart., 2019.

[11] A. Birk, B. Wiggerich, H. Bülow, M. Pfingsthorn, and S. Schwertfeger, "Safety, security, and rescue missions with an unmanned aerial vehicle (UAV)," *J. Intell. Robot. Syst.*, vol. 64, no. 1, pp. 57–76, Oct. 2011.

[12] M. Wazid, A. K. Das, N. Kumar, A. V. Vasilakos, and J. J. P. C. Rodrigues, "Design and analysis of secure lightweight remote user authentication and key agreement scheme in Internet of Drones deployment," *IEEE Internet Things J.*, vol. 6, no. 2, pp. 3572–3584, Apr. 2019.

[13] P. Gope and B. Sikdar, "An efficient privacy-preserving authentication scheme for energy internet-based vehicle-to-grid communication," *IEEE Trans. Smart Grid*, vol. 10, no. 6, pp. 6607–6618, Nov. 2019.

[14] Z. Ali, S. A. Chaudhry, M. S. Ramzan, and F. Al-Turjman, "Securing smart city surveillance: A lightweight authentication mechanism for unmanned vehicles," *IEEE Access*, vol. 8, pp. 43711–43724, 2020.

[15] Y. K. Ever, "A secure authentication scheme framework for mobile-sinks used in the Internet of Drones applications," *Comput. Commun.*, vol. 155, pp. 143–149, Apr. 2020.

[16] G. Cho, J. Cho, S. Hyun, and H. Kim, "SENTINEL: A secure and efficient authentication framework for unmanned aerial vehicles," *Appl. Sci.*, vol. 10, no. 9, p. 3149, Apr. 2020.

[17] S. A. Chaudhry, K. Yahya, M. Karuppiah, R. Kharel, A. K. Bashir, and Y. B. Zikria, "GCACS-IoD: A certificate based generic access control scheme for Internet of Drones," *Comput. Netw.*, vol. 191, May 2021, Art. no. 107999.

[18] M. Wazid, A. K. Das, N. Kumar, and M. Alazab, "Designing authenticated key management scheme in 6G-enabled network in a box deployed for industrial applications," *IEEE Trans. Ind. Informat.*, vol. 17, no. 10, pp. 7174–7184, Oct. 2021.

[19] Y. Zhang, D. He, L. Li, and B. Chen, "A lightweight authentication and key agreement scheme for Internet of Drones," *Comput. Commun.*, vol. 154, pp. 455–464, Oct. 2020.

[20] M. Rana, A. Shafiq, I. Altaf, M. Alazab, K. Mahmood, S. A. Chaudhry, and Y. B. Zikria, "A secure and lightweight authentication scheme for next generation IoT infrastructure," *Comput. Commun.*, vol. 165, pp. 85–96, Jan. 2021.

[21] C. Lin, D. He, N. Kumar, K.-K. R. Choo, A. Vinel, and X. Huang, "Security and privacy for the Internet of Drones: Challenges and solutions," *IEEE Commun. Mag.*, vol. 56, no. 1, pp. 64–69, Jan. 2018.

[22] S. U. Jan, F. Qayum, and H. U. Khan, "Design and analysis of lightweight authentication protocol for securing IoD," *IEEE Access*, vol. 9, pp. 69287–69306, 2021.

[23] U. C. Cabuk, G. Dalkilic, and O. Dagdeviren, "CoMAD: Context-aware mutual authentication protocol for drone networks," *IEEE Access*, vol. 9, pp. 78400–78414, 2021.

[24] M. A. Khan, I. Ullah, N. Kumar, O. S. Oubbati, I. M. Qureshi, F. Noor, and F. U. Khanzada, "An efficient and secure certificate-based access control and key agreement scheme for flying ad-hoc networks," *IEEE Trans. Veh. Technol.*, vol. 70, no. 5, pp. 4839–4851, May 2021.

[25] M. Nikooghadam, H. Amintoosi, S. H. Islam, and M. F. Moghadam, "A provably secure and lightweight authentication scheme for Internet of Drones for smart city surveillance," *J. Syst. Archit.*, vol. 115, May 2021, Art. no. 101955.

[26] M. W. Akram, A. K. Bashir, S. Shamshad, M. A. Saleem, A. A. AlZubi, S. A. Chaudhry, B. A. Alzahrani, and Y. B. Zikria, "A secure and lightweight drones-access protocol for smart city surveillance," *IEEE Trans. Intell. Transp. Syst.*, early access, Dec. 10, 2021, doi: 10.1109/TITS.2021.3129913.

[27] S. Hussain, S. A. Chaudhry, O. A. Alomari, M. H. Alsharif, M. K. Khan, and N. Kumar, "Amassing the security: An ECC-based authentication scheme for Internet of Drones," *IEEE Syst. J.*, vol. 15, no. 3, pp. 4431–4438, Sep. 2021.

[28] M. Tanveer, A. U. Khan, H. Shah, S. A. Chaudhry, and A. Naushad, "PASKE-IoD: Privacy-protecting authenticated key establishment for Internet of Drones," *IEEE Access*, vol. 9, pp. 145683–145698, 2021.

[29] M. Tanveer, A. U. Khan, N. Kumar, A. Naushad, and S. A. Chaudhry, "A robust access control protocol for the smart grid systems," *IEEE Internet Things J.*, early access, Sep. 17, 2021, doi: 10.1109/JIOT.2021.3113469.

[30] M. Tanveer, A. U. Khan, H. Shah, A. Alkhayyat, S. A. Chaudhry, and M. Ahmad, "ARAP-SG: Anonymous and reliable authentication protocol for smart grids," *IEEE Access*, vol. 9, pp. 143366–143377, 2021.

[31] M. Tanveer, A. H. Zahid, M. Ahmad, A. Baz, and H. Alhakami, "LAKE-IoD: Lightweight authenticated key exchange protocol for the Internet of Drone environment," *IEEE Access*, vol. 8, pp. 155645–155659, 2020.

[32] J. Srinivas, A. K. Das, M. Wazid, and A. V. Vasilakos, "Designing secure user authentication protocol for big data collection in IoT-based intelligent transportation system," *IEEE Internet Things J.*, vol. 8, no. 9, pp. 7727–7744, May 2021.

[33] M. Tanveer, A. U. Khan, N. Kumar, and M. M. Hassan, "RAMP-IoD: A robust authenticated key management protocol for the Internet of Drones," *IEEE Internet Things J.*, vol. 9, no. 2, pp. 1339–1353, Jan. 2022.

[34] M. Tanveer, G. Abbas, Z. H. Abbas, M. Bilal, A. Mukherjee, and K. S. Kwak, "LAKE-6SH: Lightweight user authenticated key exchange for 6LoWPAN-based smart Homes," *IEEE Internet Things J.*, vol. 9, no. 4, pp. 2578–2591, Feb. 2022.

[35] A. K. Sutrala, M. S. Obaidat, S. Saha, A. K. Das, M. Alazab, and Y. Park, "Authenticated key agreement scheme with user anonymity and untraceability for 5G-enabled softwarized industrial cyber-physical systems," *IEEE Trans. Intell. Transp. Syst.*, early access, Feb. 17, 2021, doi: 10.1109/TITS.2021.3056704.

[36] B. Bera, A. K. Das, S. Garg, M. J. Piran, and M. S. Hossain, "Access control protocol for battlefield surveillance in drone-assisted IoT environment," *IEEE Internet Things J.*, vol. 9, no. 4, pp. 2708–2721, Feb. 2022.

[37] W. Iqbal, H. Abbas, P. Deng, J. Wan, B. Rauf, Y. Abbas, and I. Rashid, "ALAM: Anonymous lightweight authentication mechanism for SDN enabled smart homes," *IEEE Internet Things J.*, 2020.

[38] S. Yu, A. K. Das, and Y. Park, "Comments on 'ALAM: Anonymous lightweight authentication mechanism for SDN enabled smart homes,'" *IEEE Access*, vol. 9, pp. 49154–49159, 2021.

[39] M. A. Saleem, Z. Ghaffar, K. Mahmood, A. K. Das, J. J. P. C. Rodrigues, and M. K. Khan, "Provably secure authentication protocol for mobile clients in IoT environment using puncturable pseudorandom function," *IEEE Internet Things J.*, vol. 8, no. 22, pp. 16613–16622, Nov. 2021.

[40] R. Vinoth, L. J. Deborah, P. Vijayakumar, and N. Kumar, "Secure multifactor authenticated key agreement scheme for industrial IoT," *IEEE Internet Things J.*, vol. 8, no. 5, pp. 3801–3811, Mar. 2021.

[41] M. Tanveer, G. Abbas, and Z. H. Abbas, "LAS-6LE: A lightweight authentication scheme for 6LoWPAN environments," in *Proc. 14th Int. Conf. Open Source Syst. Technol. (ICOSST)*, Dec. 2020, pp. 1–6.

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

[43] M. Tanveer, G. Abbas, Z. H. Abbas, M. Waqas, F. Muhammad, and S. Kim, "S6AE: Securing 6LoWPAN using authenticated encryption scheme," *Sensors*, vol. 20, no. 9, p. 2707, May 2020.

[44] S. Wu and K. Chen, "An efficient key-management scheme for hierarchical access control in E-medicine system," *J. Med. Syst.*, vol. 36, no. 4, pp. 2325–2337, Aug. 2012.

[45] M. Wazid, A. K. Das, V. Odelu, N. Kumar, M. Conti, and M. Jo, "Design of secure user authenticated key management protocol for generic IoT networks," *IEEE Internet Things J.*, vol. 5, no. 1, pp. 269–282, Feb. 2018.

[46] T. Alladi, N. Naren, G. Bansal, V. Chamola, and M. Guizani, "SecAuthUAV: A novel authentication scheme for UAV-ground station and UAV-UAV communication," *IEEE Trans. Veh. Technol.*, vol. 69, no. 12, pp. 15068–15077, Dec. 2020.

[47] T. Alladi, S. Chakravarty, V. Chamola, and M. Guizani, "A lightweight authentication and attestation scheme for in-transit vehicles in IoV scenario," *IEEE Trans. Veh. Technol.*, vol. 69, no. 12, pp. 14188–14197, Dec. 2020.