electronics

MDPI

Article

A Drone-Assisted Anonymous Authentication and Key Agreement Protocol with Access Control for Accident Rescue in the Internet of Vehicles

Jihu Zheng [1], Haixin Duan [1,*], Chenyu Wang [2], Qiang Cao [2] and Guoai Xu [3] and Rui Fang [1]

[1] Institute for Network Sciences and Cyberspace, Tsinghua University, Beijing 100084, China; zhengjihu@catarc.ac.cn (J.Z.); fangrui@china-icv.cn (R.F.)
[2] School of Cyberspace Security, Beijing University of Posts and Telecommunications, Beijing 100876, China; wangchenyu@bupt.edu.cn (C.W.); scq@bupt.edu.cn (Q.C.)
[3] School of Computer Science and Technology, Harbin Institute of Technology, Shenzhen 518055, China; xga@hit.edu.cn
* Correspondence: duanhx@tsinghua.edu.cn

Abstract: The drone-assisted Internet of Vehicles (DIoV) displays great potential in the punctual provision of rescue services without geographical limitations. To ensure data security in accident response and rescue services, authentication schemes with access control are employed. These schemes ensure that only specific rescue vehicle operators acting within a valid period can achieve mutual authentication from a designated processor, while access for mismatched, revoked, or expired users is denied. However, the current alternatives fail to ensure session key forward secrecy, entities' mutual authentication, and user anonymity, thereby compromising users' privacy and the security of communications. Moreover, executing too many time-consuming operations on vehicles' resource-constrained devices inevitably degrades the performance of the authentication protocol. Balancing security and performance in the design of an authentication protocol with access control presents a significant challenge. To address this, a more efficient and robust authentication with access control has been designed. The proposed protocol ensures user anonymity through dynamic pseudonym allocation, achieves forward secrecy by excluding the long-term key from session key generation, and obtains mutual authentication by verifying the integrity of the messages exchanged. According to the security and performance analysis, it is demonstrated that the proposal is a robust, efficient, and cost-effective solution. In particular, the proposal can reduce the computational overhead by 66% compared to recent alternatives.

Keywords: authentication; access control; elliptic curve cryptography (ECC); drone-assisted Internet of Vehicles (DIoV)

check for updates

Citation: Zheng, J.; Duan, H.; Wang, C.; Cao, Q.; Xu, G.; Fang, R. A Drone-Assisted Anonymous Authentication and Key Agreement Protocol with Access Control for Accident Rescue in the Internet of Vehicles. *Electronics* **2024**, *13*, 1939. https://doi.org/10.3390/electronics13101939

Academic Editor: Hung-Yu Chien

Received: 19 March 2024
Revised: 5 May 2024
Accepted: 14 May 2024
Published: 15 May 2024

Copyright: © 2024 by the authors. Licensee MDPI, Basel, Switzerland. This article is an open access article distributed under the terms and conditions of the Creative Commons Attribution (CC BY) license (https://creativecommons.org/licenses/by/4.0/).

1. Introduction

Originally utilized for military purposes, drones—or unmanned aerial vehicles—are anticipated to become integral to the Internet of Vehicles (IoV), leveraging their capabilities for three-dimensional movement, high maneuverability, autonomous operation, and communication processing [1]. In remote mountainous regions, the traditional communication infrastructure fails to support timely data transmission for the IoV, particularly during critical periods such as vehicle accidents [2].

In the traffic accident rescue scenario of the DIoV, shown in Figure 1, the drone of the command center, serving as a drone gateway, provides prompt and sustainable services to facilitate accident investigations, and it allows the different departments in urban areas to quickly dispatch their own rescue vehicles to offer accident and emergency services (e.g., fire and medical services and road traffic accident clearing).

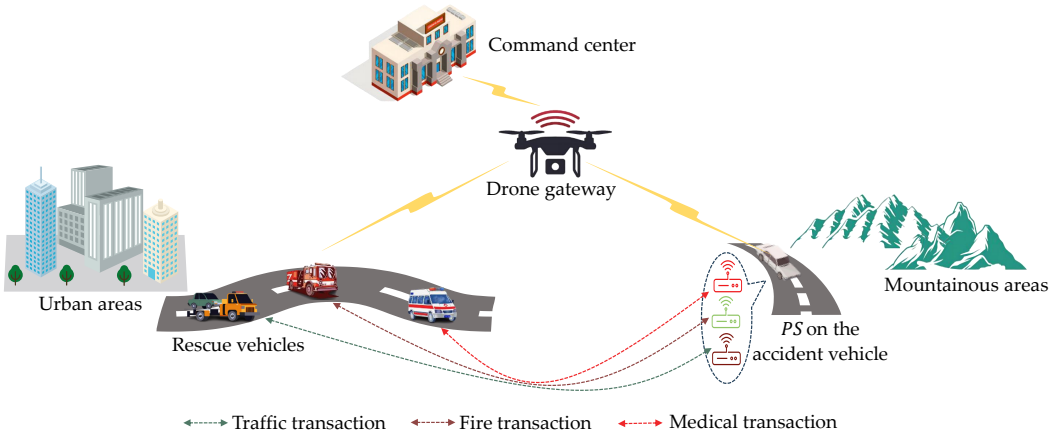


Figure 1. Accident rescue in DIoV.

However, malicious entities illegally accessing and potentially disrupting the transmitted data pose a significant safety hazard that cannot be overlooked in traffic accident rescue scenarios [3,4]. The technologies of multi-factor authentication (involving a password, a smart card and other factors) and key agreement [5,6] can be used to protect the transmitted data from unauthorized access or disruption. Upon mutual authentication, tunnels are built among rescue vehicles' users and the processor, and a session key will be established for secure communication [7,8].

Furthermore, an issue arises regarding how to provide specific rescue vehicles with legal access to particular data within a valid period, highlighting the need for fine-grained access control [9]. In this type of scheme, depending on the type and responsibilities of the rescue vehicle, only a specific rescue vehicle's user is allowed to obtain mutual authentication from a designated processor, while other users that are mismatched, revoked, or overdue are not entitled to certification. As shown in Figure 1, the processors (PS), which include the fire rescue processor, medical processor and road traffic accident clearing processor, can only interact with their respective rescue vehicles to ensure secure and accurate emergency services.

Existing authentication schemes with access control [10–16], however, fail to ensure the security of the session key and the privacy of the user [17]. Moreover, the large number of operations consumes vast amounts of storage, bandwidth and computational power, which drains the capacity of energy-limited vehicle networking devices such as vehicles' processors. Addressing the design of a more efficient and robust authentication scheme with access control, a new scheme for DIoV is studied in this paper, in which four key points are considered, as listed below.

Firstly, by defining the control policy and obeying the mechanism in which the message is delivered first, received later and verified last, the proposed scheme can meet the requirement that any rescue vehicle's user can only obtain authentication and negotiate the session key from the specified processor within the validity period. It is noted that the communication entities must verify one another's identities to satisfy mutual authentication. Alternative schemes [10,12,13,15] do not adhere to this important principle since there is a lack of authentication whereby the message receiver (i.e., the user) verifies the identity of the message sender (i.e., the DG).

Secondly, during the registration phase, the user in the rescue vehicle is free to select a password; however, there is no password verification table stored in the drone as a relay node, which can eliminate the risk of password exposure. Moreover, via the "mod" operation, the system can resist smart card loss attacks and password guessing attacks [18,19].

Thirdly, in the authentication phase, the user interacts with the DG by using a pseudonym. Note, that the dynamic of a pseudonym preserves the user's anonymity.

Additionally, regarding the session key of the user and processor, the long-term key of the DG is no longer used to compute the session key, so the session key adheres to forward secrecy.

Lastly, from the perspective of performance in terms of storage, communication and computational costs, the designed scheme can minimize these costs as much as possible in the energy-limited DIoV.

The remainder of this paper is organized as follows. Section 2 describes the related works, and then the designed scheme is shown in Section 3. In Section 4, the security analysis is provided, and the performance analysis of the proposed scheme is shown in Section 5. Finally, Section 6 gives a brief conclusion and highlights the ongoing research work.

2. Related Works

Initially, Das [20] introduced a two-factor authentication protocol incorporating both a password and a smart card to secure communications within wireless sensor networks (WSNs). Additionally, the European Union's General Data Protection Regulation (GDPR) has been implemented to enhance the security and privacy of vehicle accident data [9]. It is thus necessary to design authentication protocols with access control for the DIoV.

Our examination of existing research on the DIoV reveals a lack of protocols incorporating access control specifically designed for the DIoV. Specifically, there are mainly works that enable the vehicle's authentication with a trusted authority [21,22] and decentralized blockchain platform [23], which is similar to an ordinary authentication scenario in the IoT [18,24,25]. Other studies consider the vehicles' batch authentication by using signature fusion technology [26] and cross-domain authentication based on a two-way synchronization database mechanism [27]. Moreover, in the field of smart healthcare, some research works on authentication with access control can be found.

Utilizing the Chinese Remainder Theorem (CRT), Srinivas et al. [10] proposed a secure user authentication and access control scheme within a Cloud-of-Things-centric (CoTC) environment, specifically for wearable device monitoring systems. A notable observation is that, for forward secrecy, in their scheme, the long-term key does not need to be involved in constructing the session key. However, it is clear that significantly more storage resources would be consumed in their scheme and that no mutual authentication exists between the CoTC and sensor nodes.

Subsequently, Ref. [11] introduced an e-health-oriented scheme that integrates authentication, key agreement and access control guided by a control string specified by the medical server. Furthermore, they were the first to propose a method to transfer the ownership of patient information from the former physician to a new one, in order to allow more effective medical treatment. However, note that this scheme does not satisfy forward secrecy, or three-factor security and is not resistant to the inevitable type-l node capture attack [17].

In 2019, Banerjee et al. [12] introduced a method for time-limited user authentication and access control, whereby access privileges are automatically revoked upon the expiration of the allocated authentication period. In [12], ensuring the integrity and security of passwords is identified as a fundamental criterion, necessitating the development of a more robust authentication framework.

In the following year, after cryptanalyzing the developed scheme in [12] regarding its security flaws, such as its susceptibility to smart card loss attacks and stolen verifier attacks, Kumar et al. [13] provided an improved scheme that could effectively preserve users' privacy (anonymity and password security). Nonetheless, the issues of forward secrecy and vulnerability to type-I node capture attacks remain concerns regarding the session key's security.

Furthermore, Alzahrani et al. [14] found that the scheme in [12] cannot complete the authentication between the user and sensor node. To address the design flaw in [12], an improved scheme, ILAS-IoT, was presented in which, with the help of a gateway node, the user and a sensing device can complete the authentication phase. However, the security

flaws (i.e., the failure to provide forward secrecy and resist type-I node capture attacks) that were analyzed previously were alleviated with the application of the scheme in [14].

In the context of wireless medical sensor networks, Yao et al. [15] highlighted the limitations of existing authentication mechanisms, specifically architectural inefficiencies and overlooked security flaws. They recommended a multi-faceted authentication architecture addressing user–server, patient–server and user–patient authentication. Nevertheless, the method's resistance to password-guessing attacks, precipitated by the password verification table, requires strengthening to prevent the disclosure of users' passwords.

Recently, focusing on securing communications in the edge-enabled Internet of Medical Things, Seyed Ahmad Soleymani et al. [16] used digital signatures and proposed an authentication and Authenticated Key Exchange (AKE) protocol. In their scheme, the user, edge node and medical center can be authenticated mutually to generate a final session key $SK = h(r_{sk} \cdot R_{mc} || R_u || r_{sk} \cdot R_{mc})$, in which SK is determined by the values of these three entities. However, session-specific temporary information attacks (also known as ephemeral secret leakage attacks) directly threaten the integrity of SK.

3. The Proposed Scheme

In this section, we propose a three-factor user authentication and key agreement protocol for the DIoV, featuring access control mechanisms. This protocol ensures that users from rescue vehicles obtain mutual authentication exclusively from designated processors on accident vehicles, based on their department and service type. However, users from rescue vehicles who have mismatched credentials, revoked access, or expired credentials will be denied authentication by the processor. We further describe the system model of our proposed scheme below.

3.1. System Model

The system model shown in Figure 2 consists of four entities: the command center (CC), the rescue vehicle (RV), the drone gateway (DG) and the accident vehicle (AV), with a series of transaction processors (PS_j). Further, the DG computes and then transmits messages between the RV's user and PS_j; PS_j collects certain real-time data from the accident vehicles, such as traffic transactions, medical transactions and fire transactions, enabling the users in the RVs to access real-time data to enable prompt rescue operations.

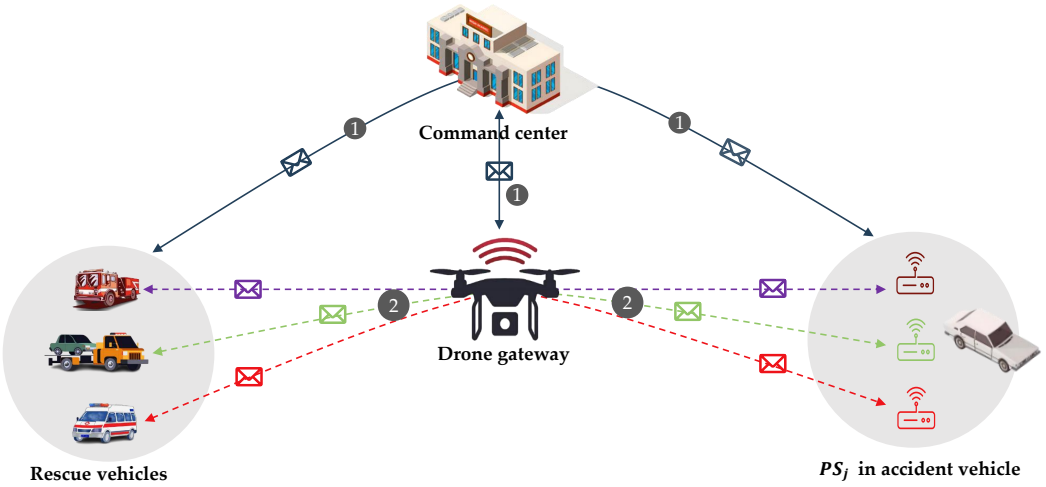


Figure 2. System model in the proposed scheme.

In Figure 2, we depict the secure channel transmissions (marked as '1') occurring during the registration phase and the public channel transmissions (marked as '2') taking place during the login and authentication phases. We then detail the authentication and key agreement process among the involved entities. Initially, the CC sets up the authentication system, generating long-term keys, secret values and public parameters. Users from rescue

vehicles (U_i) register with the CC through a secure channel, submitting registration requests and receiving a smart card from the CC. Similarly, DG and PS_j submit their identities to the CC via a secure channel to obtain identity-related secret values.

During the authentication phase, U_i sends a login request to the DG, which then verifies U_i's identity and determines their eligibility for authentication with the appropriate PS_j. Subsequently, the DG conveys a verification message to PS_j; upon PS_j authenticating the identity of DG, it calculates and sends back a message containing a session key and authentication parameters. The DG, after verifying PS_j's message, forwards it to U_i, who then authenticates DG, extracts the key parameters and recomputes the session key.

Additionally, the definitions of some terms used in the proposed scheme are listed in Table 1.

Table 1. Definitions of symbols used in the proposed scheme.

Symbol	Definition	Symbol	Definition
$\oplus$	XOR operation	PW_i	U_i's password
$h(\cdot)$	secure hash function	ID_i	U_i's real identity
$\|\|$	concatenate operation	PID_i	U_i's pseudonym
$BKG(\cdot)$	bio key generation	bio_i	U_i's biometric information
PS_j	identifier of processor in AV_j	$LTK_i(LTK_j)$	U_i's (PS_j's) secret value
U_i	legitimate user of rescue vehicle RV_i	$AVID_j$	AV_j's unique identity
$\rightarrow, \Rightarrow$	public channel, secure channel	x, y	long-term key pair of DG

Note. Similarly to [28], this paper uses $BKG(\cdot)$ to indicate the entire step dealing with the user's biometric information, i.e., $BKG(\cdot) \Leftrightarrow Gen(\cdot) + Rep(\cdot)$. Given the limited space, researchers can refer to the detailed $Gen(\cdot), Rep(\cdot)$ in [28]'s Sec. II-B.

3.2. System Setup Phase

In this phase, firstly, CC initiates an elliptic curve $E(\mathbb{F}_\mathbb{p})$ over a prime finite field $\mathbb{F}_\mathbb{p}$. Based on $E(\mathbb{F}_\mathbb{p})$, CC specifies an additive subgroup G with a $q-$order generator P, where p, q are two large primes with $|q| = n$, where n is a security parameter. Secondly, CC selects long-term key pair $x, y \in \mathbb{F}_\mathbb{p}$ and stores two secret values x, y temporarily, and parameters $\{E(\mathbb{F}_\mathbb{p}), P\}$ are public, respectively. It is noted that in order to achieve a high security level for a long-term key pair, a 320-bit ECC can be adopted, in which the length of the secret key is 160 bits and the ECC point is 320 bits [29,30].

By running the following operations in Sections 3.3–3.5, the proposed scheme of authentication with access control, mutual authentication, user anonymity and forward secrecy can be achieved.

- Authentication with access control: Since CC defines the control policy and DG obeys the mechanism by which the message is delivered first, received later and verified last, the proposed scheme can meet the requirement that any rescue vehicle's user can only achieve authenticity and negotiate the session key from the specified processor within the validity period.
- Mutual authentication: During authentication with access control, the communication entities must verify one another's identities to obtain mutual authentication. However, in alternative schemes [10,12,13,15], the user does not check the identity of the DG or gateway node and so [10,12,13,15] do not satisfy the need for mutual authentication.
- User anonymity: In the proposed scheme, the user uses a pseudonym provided by the CC in advance to communicate with the DG in a public channel. The pseudonym prevents the adversary from tracking the real user and thus preserves the user's anonymity well.
- Forward secrecy: During the generation of the session key, the scheme excludes the long-term key. Meanwhile, the alternative schemes [11–14] cannot achieve forward secrecy for the session key, given that the generation of their session keys relies on the long-term key.

3.3. Registration Phase

The registration phase enables the vehicle AV_j and the user of RV to complete the registration of related identity information in CC; meanwhile, the user of the rescue vehicle and AV_j receive feedback from CC to prepare for future authentication with access control. Specifically, three parts constitute the registration phase: one for AV_j with its processors PS_j, one for user U_i of RV and the last one for DG.

For the registration of some vehicle AV_j, three steps are required, as described below.

(1) AV_j securely sends its identity $AVID_j$ and a series of identifiers PS_j to the CC ($AV_j \Rightarrow CC : \{AVID_j, PS_j\}$).
(2) CC computes and returns $LTK_j = h(AVID_j||x)$ to PS_j in AV_j also via the secure channel ($CC \Rightarrow PS_j : \{LTK_j\}$).
(3) CC accumulates and lists all enrolled PS_j, i.e., $\Delta S = \{PS_j\}$.

For the user U_i of rescue vehicle RV, she/he also needs to complete the registration operation with CC.

(1) U_i chooses his/her identity, password pair (ID_i, PW_i) and a random number r and computes $HPW_i = h(ID_i||PW_i) \bmod n_0$, $A_0 = HPW_i \oplus r$, where $n_0 = 2^8$, as an integer [24].
(2) U_i sends A_0 to CC via the secure channel ($U_i \Rightarrow CC : \{A_0\}$).
(3) Upon CC obtaining the registration request A_0, CC records the current registration timestamp T_{reg}, generates a pseudonym PID_i for U_i and computes $LTK_i = h(PID_i||x)$, $A_1 = LTK_i \oplus A_0$ and $EID_i = h(PID_i||y)^{-1} \cdot T_{reg}$. It is noted that the output of computing $h(PID_i||y)$ is inverse in Z_q^*, i.e., $h(PID_i||y)^{-1} \in Z_q^*$.
(4) According to U_i's rescue department, CC generates a credential $token_{RV_i}$ and designates a finite time period ΔT_{auth} (e.g., 2024/04–2025/04) to enable authentication and specifies the corresponding PS_j set, i.e., $\Delta S_{RV_i} = \{PS_j\}$ with an authorized polynomial $f_i(t) = h(x||token_{RV_i}) + \prod_{PS_j \in \Delta S_{RV_i}} (t - h(PID_i||PS_j))$ over Z_p^*, where $\Delta S_{RV_i} \subseteq \Delta S$.
(5) CC encrypts $\{f_i(t), \Delta T_{auth}, EID_i, token_{RV_i}\}$ and obtains ciphertexts $F_i(t)$, i.e., $F_i(t) = E_{h(x||y)}[f_i(t), \Delta T_{auth}, EID_i, token_{RV_i}]$.
(6) CC inserts all parameters $\{PID_i, BKG(\cdot), A_1, \Delta S_{RV_i}, F_i(t), SUM\}$ into the smart card SC_i, where the parameter "SUM" denotes the maximum number of times that the smart card enables U_i to attempt the following login phase if U_i forgets the password.
(7) CC sends the smart card SC_i to U_i via the secure channel ($CC \Rightarrow U_i : \{SC_i\}$).
(8) Upon receiving SC_i, U_i further inputs his/her bio-information bio_i, and then the smart card computes $LTK_i = A_0 \oplus A_1$, $LTK_{ii} = BKG(bio_i)$ and $A_2 = h(ID_i||PW_i||LTK_i||LTK_{ii}||\Delta S_{RV_i}) \bmod n_0$ and updates $A_1 = LTK_i||\Delta S_u \oplus HPW_i$.
(9) Finally, the smart card stores $< PID_i, BKG(\cdot), A_1, A_2, F_i(t), SUM >$.

During DG's registration, it receives a key pair (x, y) and a set $\Delta S = \{PS_j\}$ from CC through a secure channel. This provision is critical in facilitating subsequent authentication with access control processes.

To authenticate with a given PS_j on an accident vehicle, a user from a rescue vehicle (U_i) must undergo the following login and authentication phases. Figure 3 provides a comprehensive overview of all the steps involved in these phases to facilitate the readers' understanding.

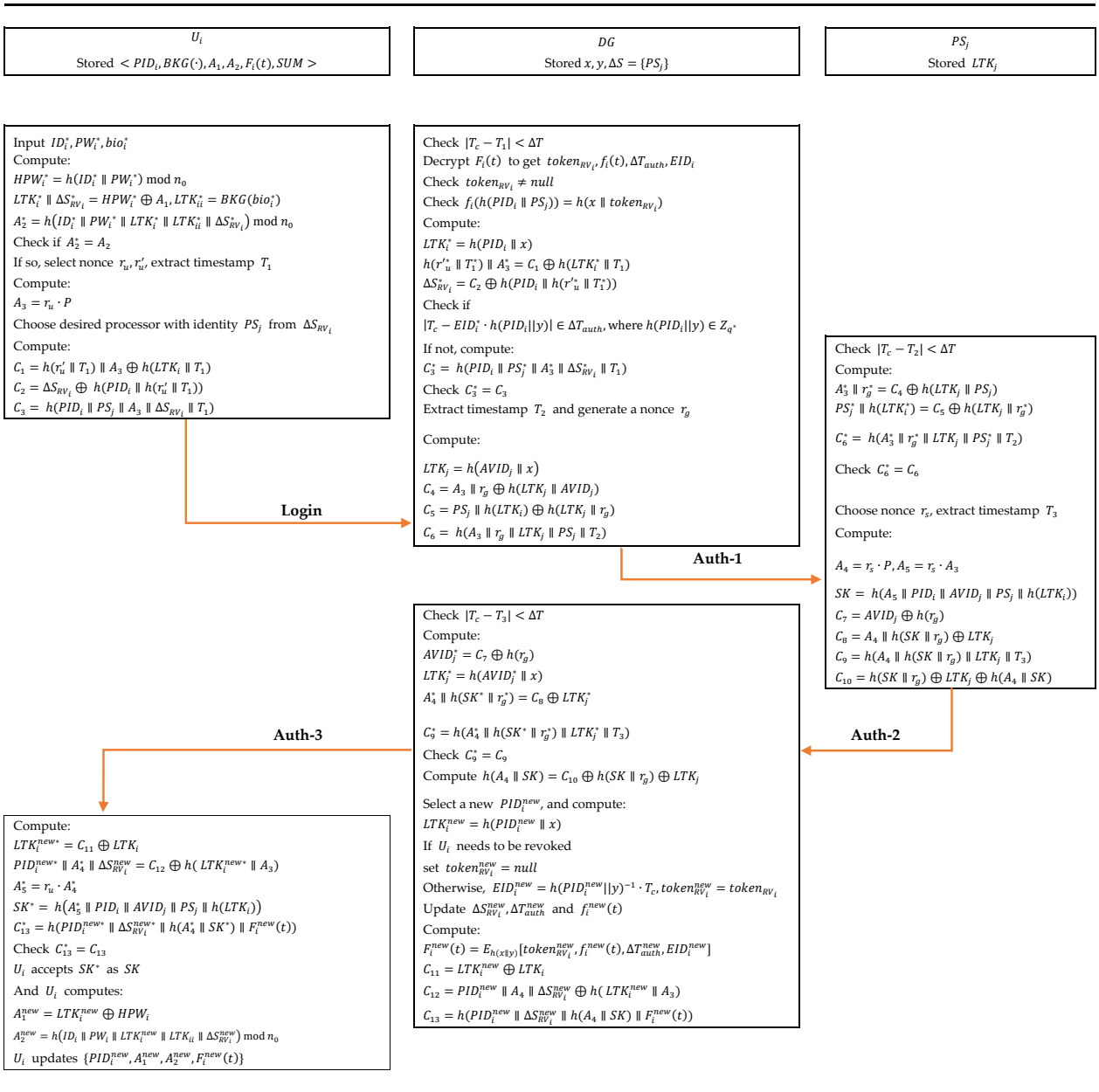


Figure 3. Login and authentication phase.

3.4. Login Phase

In the login phase, U_i enters his/her related identity, password and bio-information into the smart card; then, this smart card verifies the real identity of U_i. If this is completed, the smart card transmits U_i's further authentication request to DG. The detailed steps are shown below.

(1) U_i inputs (ID_i^*, PW_i^*) and bio-information bio_i^* to the smart card.

(2) The smart card computes the following values: $HPW_i^* = h(ID_i^*||PW_i^*) \bmod n_0$, $LTK_i^*||\Delta S_{RV_i}^* = HPW_i^* \oplus A_1$, $LTK_{ii}^* = BKG(bio_i^*)$, $A_2^* = h(ID_i^*||PW_i^*||LTK_i^*||LTK_{ii}^*||\Delta S_{RV_i}^*) \bmod n_0$.

(3) Then, the smart card checks whether $A_2^* = A_2$ holds or not, where A_2 has been stored in the smart card during the registration phase. If not, the smart card stops this session and meanwhile updates the value of SUM by adding the number 1. If

SUM exceeds the maximal value, such as 3, this smart card will be suspended until U_i re-registers.

(4) Otherwise, the smart card extracts timestamp T_1, selects random numbers $r_u, r'_u \in Z_p^*$ and the processor with identity PS_j from ΔS_{RV_i}, which U_i wishes to obtain authentication, and computes the following values: $A_3 = r_u \cdot P$, $C_1 = h(r'_u||T_1)||A_3 \oplus h(LTK_i||T_1)$, $C_2 = \Delta S_{RV_i} \oplus h(PID_i||h(r'_u||T_1))$, $C_3 = h(PID_i||PS_j||A_3||\Delta S_{RV_i}||T_1)$.

(5) Lastly, the smart card sends the authentication request containing $\{PID_i, PS_j, F_i(t), C_1, C_2, C_3, T_1\}$ to DG via the open channel (**Login**: $U_i \rightarrow DG$: $\{PID_i, PS_j, F_i(t), C_1, C_2, C_3, T_1\}$).

3.5. Authentication Phase

In the **auth**entication phase, we mainly consider the mutual authentication in $U_i \rightleftharpoons DG \rightleftharpoons PS_j$, and then a session key SK between U_i and PS_j will be negotiated and used to protect the secret information in future communications.

The first step is **Auth-1**, where DG verifies the identity of the user and transmits a related message to PS_j. The detailed operations are shown in the following.

(1) Given an authentication request $\{PID_i, PS_j, F_i(t), C_1, C_2, C_3, T_1\}$ from U_i, DG first determines whether the time gap of the current timestamp T_c and T_1 is less than a threshold value ΔT or not (i.e., $|T_c - T_1| < \Delta T$). If $|T_c - T_1| > \Delta T$, DG stops this session.

(2) Otherwise, DG decrypts $F_i(t)$ to recover $\{f_i(t), \Delta T_{auth}, EID_i, token_{RV_i}\}$ by using symmetric key $h(x||y)$ and checks if $token_{RV_i} \neq$ is null. If not, it means that this user's access has been revoked.

(3) Otherwise, DG verifies whether $f_i(h(PID_i||PS_j)) = h(x||token_{RV_i})$. If not, DG directly discards this request, since U_i at this time is not authorized (or does not match) to run authentication with PS_j (i.e., $PS_j \notin \Delta S_{RV_i}$).

(4) Otherwise, DG computes $LTK_i^* = h(PID_i||x)$, $h(r'^*_u||T_1^*)||A_3^* = C_1 \oplus h(LTK_i^*||T_1)$ and $\Delta S_{RV_i}^* = C_2 \oplus h(PID_i||h(r'^*_u||T_1^*))$. At this moment, DG checks if $EID_i^* \cdot h(PID_i||y) = h(y||PID_i)$. If this holds, it means that U_i's authentication service has been revoked, and DG discards this session.

(5) Otherwise, DG verifies if $|T_c - EID_i^* \cdot h(PID_i||y)| \in \Delta T_{auth}$. If not, this denotes that U_i's time allocated to run authentication with PS_j has been exceeded, and DG stops this session.

(6) Otherwise, DG computes values $C_3^* = h(PID_i||PS_j^*||A_3^*||\Delta S_{RV_i}^*||T_1^*)$ and checks if $C_3^* = C_3$; if so, DG randomly selects a nonce r_g, extracts the timestamp T_2 and then computes $LTK_j = h(AVID_j||x)$, $C_4 = A_3||r_g \oplus h(LTK_j||PS_j)$, $C_5 = PS_j||h(LTK_i) \oplus h(LTK_j||r_g)$ and $C_6 = h(A_3||r_g||LTK_j||PS_j||T_2)$.

(7) DG transmits the message $\{PID_i, C_4, C_5, C_6, T_2\}$ to PS_j in the open channel, denoted by **Auth-1**: $DG \rightarrow PS_j$: $\{PID_i, C_4, C_5, C_6, T_2\}$).

To facilitate the readers' understanding, we give the following remark on how the user in a rescue vehicle can only obtain authentication with a specified PS_j and be authenticated within a specified time period.

Remark 1. *A user U_i associated with a set $\Delta S_{RV_i} = \{PS_j\}$ can only be authorized to query authentication with a designated $PS_j \in \Delta S_{RV_i}$. In the event that U_i wishes to query authentication with some unauthorized node $PS_k \notin \Delta S_{RV_i}$, this request will be directly declined by DG finding $f_i(h(PID_i||PS_k)) \neq h(x||token_{RV_i})$, since only the PS's identifier in ΔS_{RV_i} meets polynomial $f_i(t)$, which is preset by CC, and this $f_i(t)$ is unknown to U_i. Thus, each U_i will only run authentication with a corresponding authorized PS_j.*

Remark 2. *Regarding authentication within a specific timeframe, one occasion is that without retirement or dismissal, some U_i's time allocated for authentication may be exceeded. At this time, by DG checking $|T_c - EID_i \cdot h(PID_i||y)| \notin \Delta T_{auth}$, U_i cannot run authentication with his/her*

*authorized PS_j any longer. In the event of retirement and dismissal for U_i, regardless of whether $|T_c - EID_i \cdot h(PID_i||y)| \in \Delta T_{auth}$ or not, U_i cannot run authentication with his/her authorized PS_j any longer by only DG checking if $token_{RV_i}$= null. This is because, before this authentication session, DG has preset the future token $token^{new}_{RV_i}$= null to revoke this U_i (here, this preset operation can be seen in the update step of **Auth-3**).*

In the following **Auth-2**, via the received message $\{PID_i, C_4, C_5, C_6, T_2\}$, PS_j verifies the identity of DG and then uses the user's and its own secret to negotiate a session key. The detailed operations can be seen below.

(1) Upon receiving the message $\{PID_i, C_4, C_5, C_6, T_2\}$, PS_j first checks whether $|T_c - T_2| < \Delta T$. If not, PS_j stops this session.
(2) Otherwise, PS_j obtains $A_3^*||r_g^* = C_4 \oplus h(LTK_j||PS_j)$, and computes $PS_j^*||h(LTK_i^*) = C_5 \oplus h(LTK_j||r_g^*)$, $C_6^* = h(A_3^*||r_g^*||LTK_j||PS_j^*||T_2)$ and checks if $C_6^* = C_6$. If not, PS_j ceases the subsequent operations.
(3) Otherwise, PS_j selects a nonce r_s, extracts a corresponding timestamp T_3 and computes $A_4 = r_s \cdot P$, $A_5 = r_s \cdot A_3$.
(4) PS_j computes a session key $SK = h(A_5||PID_i||AVID_j||PS_j||h(LTK_i))$, and then it computes $C_7 = AVID_j \oplus h(r_g)$, $C_8 = A_4||h(SK||r_g) \oplus LTK_j$, $C_9 = h(A_4||h(SK||r_g)||LTK_j||T_3)$, $C_{10} = h(SK||r_g) \oplus LTK_j \oplus h(A_4||SK)$.
(5) Eventually, PS_j sends the message $\{C_7, C_8, C_9, C_{10}, T_3\}$ to DG via the open channel, denoted by **Auth-2**: $PS_j \rightarrow DG : \{C_7, C_8, C_9, C_{10}, T_3\}$.

Following this, in **Auth-3** DG receives PS_j's message and verifies the identity of PS_j. Then, for the user, DG updates the authentication parameters, which include the access control. Further, DG sends the updated message to the user. The detailed operations can be seen in **Auth-3**.

(1) With the message sent from PS_j, DG first checks whether $|T_c - T_3| < \Delta T$. If not, DG stops this session.
(2) Otherwise, DG computes the values $AVID_j^* = C_7 \oplus h(r_g)$, $LTK_j^* = h(AVID_j^*||x)$, $A_4^*||h(SK^*||r_g^*) = C_8 \oplus LTK_j^*$, $C_9^* = h(A_4^*||h(SK^*||r_g^*)||LTK_j^*||T_3)$ and checks if $C_9^* = C_9$. If not, DG terminates this authentication.
(3) Otherwise, DG obtains $h(A_4||SK) = C_{10} \oplus h(SK||r_g) \oplus LTK_j$ and runs the following update operations.
(4) DG updates a new pseudonym PID_i^{new} for U_i.
(5) DG updates $LTK_i^{new} = h(PID_i^{new}||x)$.
(6) DG updates $token_{RV_i}^{new} = token_{RV_i}$. Of course, if the user needs to be revoked, $token_{RV_i}^{new}$= null.
(7) DG updates $EID_i^{new} = h(PID_i^{new}||y)^{-1} \cdot T_c$.
(8) DG updates ΔT_{auth}^{new} and sets $f_i^{new}(t) = h(x||token_{RV_i}^{new}) + \prod_{AVID_j \in \Delta S_{RV_i}^{new}}(t - h(PID_i^{new}||PS_j))$, where $\Delta S_{RV_i}^{new} = \Delta S_{RV_i}$ with no change for the processor(s); or $\Delta S_{RV_i}^{new} = \Delta S_{RV_i} + \Delta S_{RV_i}^{add}$ with an added new processor(s); or $\Delta S_{RV_i}^{new} = \Delta S_{RV_i} - \Delta S_{RV_i}^{del}$ with a deleted processor(s); or $\Delta S_{RV_i}^{new} = \Delta S_{RV_i} - \Delta S_{RV_i}^{del} + \Delta S_{RV_i}^{add}$ with a deleted and then a newly added processor(s).
(9) DG further computes $F_i(t)^{new} = E_{h(x||y)}[f_i^{new}(t), \Delta T_{auth}^{new}, EID_i^{new}, token_{RV_i}^{new}]$, $C_{11} = LTK_i^{new} \oplus LTK_i$, $C_{12} = PID_i^{new}||A_4||\Delta S_{RV_i}^{new} \oplus h(LTK_i^{new}||A_3)$, and then $C_{13} = h(PID_i^{new}||\Delta S_u^{new}||h(A_4||SK)||F_i^{new}(t))$.
(10) DG transmits message $\{F_i^{new}(t), C_{11}, C_{12}, C_{13}\}$ to U_i in the open channel, denoted by **Auth-3**: $DG \rightarrow U_i : \{F_i^{new}(t), C_{11}, C_{12}, C_{13}\}$.

After receiving the response from DG, U_i authenticates the DG's identity and recomputes the session key. Finally, U_i stores the related updated authentication parameters.

(1) U_i computes: $LTK_i^{new*} = C_{11} \oplus LTK_i$, $PID_i^{new*}||A_4^*||\Delta S_{RV_i}^{new*} = C_{12} \oplus h(LTK_i^{new*}||A_3)$, $A_5^* = r_u \cdot A_4^*$ and $SK^* = h(A_5^*||PID_i||AVID_j||PS_j||h(LTK_i))$.

(2) U_i computes $C_{13}^* = h(PID_i^{new*}||\Delta S_{RV_i}^{new*}||h(A_4^*||SK^*)||F_i^{new}(t))$ and then checks if $C_{13}^* = C_{13}$. If not, U_i discards this session.
(3) If "=" holds, U_i regards this SK^* as the negotiated session key SK.
(4) U_i updates $A_1^{new} = LTK_i^{new} \oplus HPW_i$, $A_2^{new} = h(ID_i||PW_i||LTK_i^{new}||LTK_{ii}||\Delta S_{RV_i}^{new})$ mod n_0.
(5) U_i replaces parameters $\{PID_i, A_1, A_2, F_i(t)\}$ with $\{PID_i^{new}, A_1^{new}, A_2^{new}, F_i^{new}(t)\}$ in smart card SC_i.

3.6. Password Change Phase

For enhanced security, users (U_i) are able to modify or update their password independently, without necessitating an interaction with the command center (CC). This process is bifurcated into two primary stages: the verification of the user's identity, executed by the smart card, followed by U_i's update of the parameters, including PW_i, A_1, A_2. The procedural steps for these operations are delineated as follows.

(1) As described in the login phase, U_i first enters the old password PW_i^{old} and identity ID_i in the smart card.
(2) When the smart card verifies that $A_2^* = A_2$ holds, it enables U_i to choose a new password PW_i^{new}, and it updates $HPW_i^{new} = h(ID_i||PW_i^{new})$ mod n_0, $A_1^{new} = LTK_i \oplus HPW_i^{new}$, $A_2^{new} = h(ID_i||PW_i^{new}||LTK_i||LTK_{ii}||\Delta S_{RV_i})$ mod n_0
(3) The smart card finally replaces parameters $\{A_1, A_2\}$ with $\{A_1^{new}, A_2^{new}\}$.

4. Security Analysis of the Proposed Scheme

To properly validate the authentication with access control operation in the application layer, the proposed scheme should be robust in the underlying layer. Next, the security and reliability are further analyzed in the formal analysis (Section 4.1) and the heuristic analysis (Section 4.2).

4.1. Formal Analysis of the Proposed Scheme

In this part, we introduce some basics for the formal proof in Section 4.1.1; then, in Section 4.1.2, we give the detailed security proof in the form of Theorem 1.

4.1.1. Basics for Formal Proof

Before the simulation, the simulator initiates an elliptic curve $E(\mathbb{F}_\mathbb{p})$ over a prime finite field $\mathbb{F}_\mathbb{p}$. Based on $E(\mathbb{F}_\mathbb{p})$ and a security parameter n, the simulator specifies a $q-$order additive subgroup G with a generator P, where p, q are two large primes with $|q| = n$, and the size of P is 320 bits. Next, U_i obtains the information $\{ID_i, PW_i, Bio_i\}$ and the smart card that contains $\{PID_i, BKG(\cdot), A_1, A_2, EID_i, F_i(t), SUM\}$; CC generates a long-term key pair x, y; PS_j owns the identity–secret key pair $\{AVID_j, LTK_j\}$.

Then, three entities U_i, DG, PS_j involved in the proposed scheme $\mathbb{S}$ instantiate instances $\prod_{U_i}^u, \prod_{DG}^g, \prod_{PS_j}^s$, respectively. If there is no need to differentiate the three instances, the instance can be simply marked as $\prod^t$. Further, each instance will be regarded as an oracle; that is, upon receiving an input message that is valid/incorrect or null, the oracle will correspondingly accept/reject it or return "$\perp$", meaning that there is no response.

Additionally, we introduce some terms used in the security proof below.

Accepted state. An instance $\prod^t$ will be an accepted state if $\prod^t$ receives the last expected protocol message. Meanwhile, the ordered concatenation of all sent and received messages will shape the session identifier of $\prod^t$ in each session.

Partnering. Two instances $\prod^{t_1}, \prod^{t_2}$ are partnered if these two accepted states' instances $\prod^{t_1}, \prod^{t_2}$ are authenticated mutually with the identical session identification being normally shared; meanwhile, $\prod^{t_1}, \prod^{t_2}$ are partners.

Adversary. An eCK (extended Canetti–Krawczyk) adversary $\mathcal{A}$ is capable of interacting with users (U_i), the drone gateway (DG) or any processor (PS_j) by initiating information queries to their respective oracles and a simulator. Utilizing the responses obtained, $\mathcal{A}$

endeavors to compromise the integrity of the authentication messages and the established session key. The potential queries that $\mathcal{A}$ can execute, leveraging the capacities outlined for an eCK adversary in Table 2, include the following:

Table 2. Description of eCK (extended Canetti–Krawczyk) adversary capacities [31].

$\mathbf{I}_*$	**Attack Capacities**
C_1	$\mathcal{A}$ can acquire previous session keys between communication entities
C_2	$\mathcal{A}$ can learn DG's secret key pair when considering the system's eventual failure
C_3	$\mathcal{A}$ can obtain ephemeral secrets when testing the security of the session key
C_4	$\mathcal{A}$ can fully control the open channel and then intercept, modify, insert and delete any transmitted messages from the open channel
C_5	$\mathcal{A}$ can enumerate all items offline in the Cartesian product of identity space and password space $D_{id} \times D_{pw}$ within polynomial time
C_6	$\mathcal{A}$ can break some processor and then extract the stored sensitive data and even control the broken processor to participate in the next communication interaction
C_7	In a 3-factor user authentication scheme, $\mathcal{A}$ can compromise two of the three following factors: (a) password; (b) data in the smart card; (c) bio-information

- *Execute*$(\prod_{U_i}^{u}, \prod_{DG}^{g}, \prod_{PS_j}^{s})$. $\mathcal{A}$ can run a query to simulate the entire authentication process and obtain a desirable message exchange among U_i, DG and PS_j.
- *Send*$(\prod^{t}, m)$. In a 'send' query, $\mathcal{A}$ can send a message m and then launch an active attack for a participating instance $\prod^{t}$. According to the $\mathbb{S}$, if m is valid and $\prod^{t}$ has also received the message m, this simulator returns a response.
- *SessionKeyReveal* $(\prod^{t})$. In this query, besides the session key to be tested, $\mathcal{A}$ can obtain the other session keys via *SessionKeyReveal* $(\prod^{t})$.
- *EphemeralKeyReveal* $(\prod^{t})$. This query means that adversary $\mathcal{A}$ can obtain the entities' ephemeral secrets, such as nonces or random numbers.
- *Corrupt*$(\prod_{U_i}^{u}, \alpha \in \{-1, 0, 1\})$. In this query, according to the value α, $\mathcal{A}$ can acquire related authentication factors stored in U_i. Specifically, $\mathcal{A}$ retrieves passwords (to $\alpha = -1$), data stored in the smart card (to $\alpha = 0$) and bio-information bio_i (to $\alpha = 1$).
- *Corrupt*$(\prod_{DG}^{g})$. In this query, $\mathcal{A}$ can grasp the long-term key pair (x, y).
- *Corrupt*$(\prod_{PS_j}^{s})$. This query states that the secret of PS_j can be obtained by $\mathcal{A}$.

Freshness. Three instances $\prod_{U_i}^{u}$, $\prod_{DG}^{g}$ and $\prod_{PS_j}^{s}$ are fresh if $\mathcal{A}$ does not grasp the session key between U_i and PS_j by using the ***reveal*** queries shown above.

Test$(\prod^{t})$. The 'test' query evaluates the semantic security of the session key SK, and, in this query, $\mathcal{A}$ is capable of querying only once. According to $\mathbb{S}$, the instance $\prod^{t}$ can be $\prod_{U_i}^{u}$ or $\prod_{PS_j}^{s}$. Generally, "$\perp$" (null) will be returned if instance $\prod^{t}$ has not computed the SK or $\prod^{t}$ is not fresh, or ***Test***$(\prod^{t})$ has been queried before the 'test' query. Otherwise, the oracle in this query will choose one unbiased coin $b \in \{0, 1\}$. With the value of b, ***Test***$(\prod^{t})$ returns the final result, i.e., "result = the real session key" (if $b = 1$) or "result = a random string that has the same length as the session key" (if $b = 0$).

Semantic Security. For a given scheme $\mathbb{S}$, a probabilistic polynomial time (PPT) adversary $\mathcal{A}$ has made a sequence of queries including *Execute*$(\cdot)$, *Send*$(\cdot)$, *Corrupt*$(\cdot)$, and *Reveal*$(\cdot)$. Now, $\mathcal{A}$ wishes to guess the value of b in the *Test* query and return a guessed value b^*. Let *Succ*$(\mathcal{A})$ denote the advantage of $\mathcal{A}$ correctly guessing b^* of b, i.e., $b^* = b$. Then, we define the advantage of $\mathcal{A}$ whereby $\mathcal{A}$ successfully breaks the session key's semantic security in the following:

$$Adv_{\mathbb{S}}^{\mathcal{A}} = 2\Pr[Succ(\mathcal{A})] - 1$$

4.1.2. Semantic Security Proof

In the following, Theorem 1 derives the advantage whereby $\mathcal{A}$ can break the semantic security of the session key in the proposed scheme.

Theorem 1. *Let $\mathbb{S}$ be the designed scheme and $|\mathcal{D}|$ be the space of the password. Then, a PPT adversary $\mathcal{A}$, by querying **Execute**$(\cdot)$ q_e times, **Send**$(\cdot)$ q_s times, **Hash**$(\cdot)$ q_h times and **Biohashing**$(\cdot)$ $q_{BKG(\cdot)}$ times, breaks the semantic security of the session key in $\mathbb{S}$ with the following advantage $Adv^{\mathcal{A}}_{\mathbb{S},\mathcal{D}}$, which is less than*

$$\frac{q_h^2+6q_s}{2^{l_1}} + \frac{(q_s+q_e)^2}{p} + \frac{q^2_{BKG(\cdot)}+2q_{BKG(\cdot)}}{2^{l_2}} + 2(C'q^{s'}_{send} + Adv_p^{ECDL}(n) + Adv_p^{ECCDH}(n))$$

Proof. Here, we give the theorem's proof by setting a sequence of games, namely **Game$_1$** to **Game$_9$**. Moreover, let $Succ_l$ denote that $\mathcal{A}$ correctly guesses the b in the *Test* query of **Game$_l$**, $\{l = 1, 2, \cdots, 9\}$.

Game$_1$: This game simulates a real attack under the random-or-real oracle. Then, the oracle directly chooses a bit b. Thus,

$$Adv^{\mathcal{A}}_{\mathbb{S},\mathcal{D}} = 2\Pr[Succ_1] - 1 \tag{1}$$

Game$_2$: This game maintains a hash list Ψ_h and a BKG$(\cdot)$ list $\Phi_{BKG(\cdot)}$. The adversary $\mathcal{A}$ queries a hash value $h(\gamma)$; then, the hash oracle Θ_h takes γ to retrieve Ψ_h. If there is a retrieved hash value $h(\gamma)$ in Ψ_h, Θ_h returns $h(\gamma)$. Otherwise, a random string ψ will be returned to $\mathcal{A}$, while (γ, ψ) is stored in Ψ_h. As for BKG$(\cdot)$'s oracle $\Theta_{BKG(\cdot)}$, it is simulated in the same way with hash oracle Θ_h.

Based on the known two lists, $\mathcal{A}$ executes a *Test* query to guess the value of b. Factually, given $SK = h(A_5||PID_i||AVID_j||PS_j||h(LTK_i))$, it states that the secret values, including U_i's r_u, LTK_i and PS_j's r_s, are embedded in the SK. Thus, without the secret values, $\mathcal{A}$ cannot compute SK and has no way to decide whether $b = 0$ or $b = 1$.

Hence, the advantage of winning this game is equal to that of **Game$_1$**, i.e.,

$$\Pr[Succ_1] = \Pr[Succ_2] \tag{2}$$

Game$_3$: Based on **Game$_2$**, $\mathcal{A}$ in **Game$_3$** initiates an active attack to convince a communication entity to accept a forged message by executing queries $Send(\cdot)$, $Hash(\cdot)$, $Biohashing(\cdot)$. When compared with **Game$_2$**, only when a collision is found can a forged message be made, and $\mathcal{A}$'s advantage can be achieved in **Game$_3$**. Equally, if the following collisions occur, the game is aborted.

(i) $\mathcal{A}$ can find a collision in the hash values or BKG$(\cdot)$'s outputs, and the probability is $\frac{q_h^2}{2^{l_1+1}}$ or $\frac{q^2_{BKG(\cdot)}}{2^{l_2+1}}$, where l_1 and l_2 denote the length of the output by function $h(\cdot)$ and BKG$(\cdot)$, respectively.

(ii) Another collision that $\mathcal{A}$ is capable of finding is the choice of random numbers $(r_u, r'_u, r_g, r_s \in Z_p^*)$, with a probability of $\frac{(q_s+q_e)^2}{2p}$.

Thus, we have

$$|\Pr[Succ_3] - \Pr[Succ_2]| \leqslant \frac{q_h^2}{2^{l_1+1}} + \frac{q^2_{\mathrm{BKG}(\cdot)}}{2^{l_2+1}} + \frac{(q_s+q_e)^2}{2p} \tag{3}$$

Game$_4$: $\mathcal{A}$ in this game wishes to guess C_3, C_6, C_9, C_{13} without initiating a hash query or BKG$(\cdot)$ query.

Obviously, we can obtain

$$|\Pr[Succ_4] - \Pr[Succ_3]| \leqslant \frac{q_s}{2^{l_1}} \tag{4}$$

Game$_5$: In this game, $\mathcal{A}$ also tries to guess A_1, but without initiating a hash query or BKG$(\cdot)$ query.

Similarly, we can obtain

$$|\Pr[Succ_5] - \Pr[Succ_4]| \leqslant \frac{q_s}{2^{l_1}} \tag{5}$$

Game$_6$: In this game, via the $Corrupt(\prod_{U_i}^{u}, \alpha)$ query, $\mathcal{A}$ plans to compute A_2. There are three cases considered.

- Case$_1$, $Corrupt(\prod_{U_i}^{u}, \alpha = -1, 0)$: The probability that $\mathcal{A}$ guesses the user's bio-information is less than $\frac{q_{BKG(\cdot)}}{2^{l_2}}$;
- Case$_2$, $Corrupt(\prod_{U_i}^{u}, \alpha = 0, 1)$: Based on the technology of "fuzzy keywords + honeywords" , the probability that $\mathcal{A}$ guesses U_i's password is no more than $C' q_{send}^{s'}$, in which $\mathcal{A}$ has made at most q_{send} active attacks in password space $\mathcal{D}$, and C', s' are parameters that can be depicted by a linear regression [32].
- Case$_3$, $Corrupt(\prod_{U_i}^{u}, \alpha = -1, 1)$: The probability that $\mathcal{A}$ guesses the key value of A_1 is less than $\frac{q_s}{2^{l_1}}$;

Therefore, we can obtain

$$|\Pr[Succ_6] - \Pr[Succ_5]| \leqslant C' q_{send}^{s'} + \frac{q_s}{2^{l_1}} + \frac{q_{BKG(\cdot)}}{2^{l_2}} \tag{6}$$

Game$_7$: In this game, $\mathcal{A}$ interacts with the *EphemeralKeyReveal* ($\prod^{t}$) oracle and *SessionKeyReveal* ($\prod^{t}$) oracle. Then, $\mathcal{A}$ will obtain some outdated session keys $SK_{outdated}$, the nonces r_u, r_s. Following this, $\mathcal{A}$ wishes to corrupt the LTK_i. Similarly, given the technology of "fuzzy keywords + honeywords", $\mathcal{A}$ cannot grasp LTK_i from smart card SC_i. Another possible variation is that $\mathcal{A}$ tries to find a collision in the hash values.

Thus, we have

$$|\Pr[Succ_7] - \Pr[Succ_6]| \leqslant \frac{q_h^2}{2^{l_1+1}} \tag{7}$$

Game$_8$: In this game, by executing the $Corrupt(\prod_{PS_j}^{s})$ query and $Corrupt(\prod_{DG}^{g})$ query, $\mathcal{A}$ can obtain the secret value LTK_j of PS_j and further A_3, A_4. However, given A_3 (resp. A_4) in the 320-bit elliptic curve, $\mathcal{A}$ cannot resolve r_u (resp. r_s) from A_3 (resp. A_4), based on the fact that no available PPT solution can be used to break the elliptic curve discrete logarithm problem (ECDLP) [33].

Thus, the following deduction holds:

$$|\Pr[Succ_8] - \Pr[Succ_7]| \leqslant Adv_p^{ECDL}(n) \tag{8}$$

where $Adv_p^{ECDL}(n)$ denotes the advantage whereby $\mathcal{A}$ breaks the (ECDLP) problem.

Game$_9$: This game simulates the case in which $\mathcal{A}$ tries to compute the session key; at this time, $\mathcal{A}$ no longer asks queries $Execute(\cdot)$, $Send(\cdot)$ and $Corrupt(\cdot)$. However, given A_3, A_4 in the 320-bit elliptic curve, $\mathcal{A}$ cannot resolve r_u, r_s to obtain A_5, based on the fact that no available PPT solution can be used to break the elliptic curve computational Diffie–Hellman (ECCDH) problem [34].

In particular,

$$|\Pr[Succ_9] - \Pr[Succ_8]| \leqslant Adv_p^{ECCDH}(n) \tag{9}$$

where $Adv_p^{ECCDH}(n)$ denotes the advantage whereby $\mathcal{A}$ solves the (ECCDH) problem.

At present, $\mathcal{A}$ has no non-negligible advantage to guess b than $\frac{1}{2}$ and so $\Pr[Succ_9] = \frac{1}{2}$.

Hence, from Equations (1)–(9) and the triangular inequality, we obtain

$$\begin{aligned} Adv^{\mathcal{A}}_{\mathbb{S},\mathcal{D}} &= 2\Pr[Succ_1] - 1 \\ &= 2\Pr[Succ_9] - 1 + 2(\Pr[Succ_1] - \Pr[Succ_9]) \\ &\leqslant \frac{2q_h^2 + 6q_s}{2^{l_1}} + \frac{(q_s + q_e)^2}{p} + \frac{q^2_{BKG(\cdot)} + 2q_{BKG(\cdot)}}{2^{l_2}} + \Delta \end{aligned} \tag{10}$$

Thus, one can see that the PPT adversary $\mathcal{A}$ cannot break the semantic security of the session key with a non-negligible advantage $Adv^{\mathcal{A}}_{\mathbb{S},\mathcal{D}}$ that is less than $(\frac{2q_h^2+6q_s}{2^{l_1}} + \frac{(q_s+q_e)^2}{p} + \frac{q^2_{BKG(\cdot)}+2q_{BKG(\cdot)}}{2^{l_2}} + \Delta)$, where $\Delta = 2(C'q^{s'}_{send} + Adv^{ECDL}_p(n) + Adv^{ECCDH}_p(n))$. □

4.2. Heuristic Analysis of the Protocol

Utilizing the heuristic analysis method [19], which is a widely recognized approach to assessing security without the need for complex mathematical formulas, provides a straightforward yet comprehensive examination of a scheme or protocol's security aspects. This analysis demonstrates that the proposed protocol offers essential security features and is resilient against known cyber threats.

(1) *Mutual Authentication*: Based on the protocol's login and verification phase, U_i and DG authenticate each other as DG checks if $C_3^* = C_3$ and U_i checks if $C_{13}^* = C_{13}$. DG and PS_j can authenticate each other bidirectionally by verifying whether $C_6^* = C_6$ and $C_9^* = C_9$, respectively. Therefore, the proposed protocol can recognize mutual authentication.

(2) *Session Key Agreement*: Session key agreement means that no one can solely pre-negotiate the session key. Specifically, given $SK = h(A_5||PID_i||AVID_j||PS_j||h(LTK_i))$, it states that SK must consist of U_i's newly secret r_u and PS_j's timely secret r_s, so that no one can manipulate the session key.

(3) *Forward Secrecy*: This property guarantees that the security of the established session keys remains intact even if the long-term keys of DG are compromised (for instance, through side-channel attacks, SCAs [30,35]). Despite an adversary's potential knowledge of the long-term keys and subsequent secrets, the complexity of the ECCDH problem prevents the adversary from calculating the session key SK, thereby ensuring the protocol's resilience in maintaining confidentiality over time.

(4) *User Anonymity*: User anonymity consists of the user's identity protection, which cannot be discerned by the adversary, and the user's un-traceability, which ensures that the adversary cannot distinguish whether two full sessions originate from the same user.

For identity protection, on one hand, in the registration phase, U_i only sends the value A_0 to DG and so there is no exposed identity information ID_i that the adversary can extract, even if the adversary corrupts the DG. On the other hand, in the verification phase, U_i's identity ID_i has been perfectly embedded in A_2^{new} and still cannot be obtained by the adversary.

For the user's un-traceability, the randomness inherent in pseudonym PID_i eliminates any statistical properties that an adversary might exploit to determine whether two sessions originate from the same user, thus enhancing privacy.

(5) *Processor Impersonation Attack*: This attack [19] considers an inside adversary (e.g., the legitimate physician U_i). In this attack scenario, U_i could grasp PS_j's secret key LTK_j via his own values (LTK_i, A_4) and then impersonate this PS_j to create a forged session key for the next new U_i^{new}. Factually, in this proposed protocol, the adversary cannot grasp the secret value LTK_j from $\{C_8, C_{10}\}$, since he/she has no secret value r_g of DG. As a result, this attack is futile.

(6) *Password Guessing Attack*: As researched in [36], password guessing attacks can be divided into attack-I, where the adversary leverages the verification value in the smart card to guess the password, and attack-II, where the adversary uses the verification value in the public channel to guess the password.

For attack-I, even if the adversary knows verification values A_1, A_2 in the smart card, he/she cannot check the correctness of the guessed PW_i^* and ID_i^*, since the congruence of the "modulus" operation in HPW_i and A_2 and the limited "SUM" seriously affects the correctness of the guessed password of the adversary.

As for the adversary in attack-II, the password-related verification value is only attributed to LTK_i. Although the adversary obtains LTK_i and even owns A_1, given the similar congruence of the "modulus" operation in HPW_i, he/she cannot verify the correctness of the guessed PW_i^* and ID_i^*.

(7) *De-Synchronization Attack*: A de-synchronization attack may occur if the communication entities have to update any parameters upon the session key that has been established. However, this attack is impossible. Indeed, U_i needs to change LTK_i to LTK_i^{new} and the corresponding PID_i to PID_i^{new}, ΔS_{RV_i} to $\Delta S_{RV_i}^{new}$. Specifically, U_i firstly recovers and names LTK_i^{new*}, PID_i^{new*}, $\Delta S_{RV_i}^{new*}$ from C_{11}, C_{12}, and then checks if $h(PID_i^{new*}||\Delta S_{RV_i}^{new*}||h(A_4^*||SK^*)) = C_{13}$. If so, '$new*$' is set to '$new$'. It is the verification of the correctness of C_{13} that guarantees the synchronization update of LTK_i, PID_i and ΔS_{RV_i}, since U_i can instantly detect this attack once $h(PID_i^{new*}||\Delta S_{RV_i}^{new*}||h(A_4^*||SK^*)) \neq C_{13}$.

(8) *Replay Attack*: In a replay attack, the adversary often sends old un-changed messages to try to pass the verification of entities. Indeed, random numbers r, r_u, r_u', r_g and r_s are chosen by U_i, DG and PS_j, respectively. These random numbers ensure the freshness and independence of the exchanged messages in each session; therefore, there are no un-changed messages that can be used to initiate a replay attack.

(9) *DoS Attack*: In the proposed protocol, in order to render DG unavailable (i.e., a DoS attack), the adversary can replay the old message $\{(PID_i, PS_j, F_i(t), C_1, C_2, C_3, T_1)\}$ repeatedly. However, this attack can be effectively eliminated by DG checking if the time gap between the current time T_c and T_1 exceeds the set value ΔT (for example, 3 min). If so, DG ignores this session. Further, even the adversary may change T_1 to make the time gap less than ΔT; DG also discards this session by finding the verification failure regarding value C_3, where C_3 can only be computed via the original T_1.

(10) *Privileged Insider Attack*: In this attack, the adversary (or even a corrupted DG) can extract the legitimate user's identity information ID_i in the registration phase. Factually, each U_i in the proposed protocol sends an A_0 to DG, and ID_i cannot be obtained, since the ID_i has been encapsulated with $r \in Z_p$ and the "modulus" operation.

(11) *Processor's Node Capture Attack*: An adversary capable of compromising a processor to obtain secret LTK_j and associated values A_3 and A_4 still cannot compute the session key SK without solving the computationally difficult elliptic curve discrete logarithm problem [33].

(12) *Session-Specific Temporary Information Attack*: In this attack (also known as the ephemeral secret attack, ESL) [31], the adversary can learn the session key by obtaining nonces such as random numbers r_u and r_s. However, in our scheme, apart from random numbers, the long-term information LTK_i also constitutes the SK and cannot be captured by the adversary.

5. Performance Analysis of the Proposed Scheme

In this section, we present a detailed performance analysis comparing our authentication protocol against seven recent alternatives. This includes a functionality comparison based on the criteria in Table 3, outlined in Table 4, as well as an evaluation of the storage, communication and computational costs, detailed in Table 5.

Table 3. Ten criteria for evaluation of authentication schemes.

$\dagger_*$	**Ideal Attributes**	$\ddagger_*$	**Security Attributes**
$\dagger_1$	Password friendly	$\ddagger_1$	User anonymity
$\dagger_2$	Sound repairability	$\ddagger_2$	No password exposure
$\dagger_3$	Provision of key agreement	$\ddagger_3$	Forward secrecy
$\dagger_4$	Mutual authentication	$\ddagger_4$	Resistance to known attacks
$\dagger_5$	No password verification table	$\ddagger_5$	No smart card loss attack

(1) *Functionality Analyses*

To evaluate the scheme's advantages and disadvantages in functionality, we adopt the widely accepted 10 criteria [17], containing five ideal ($\dagger_*$) attributes and five security ($\ddagger_*$) attributes, as described in Table 3. The $\ddagger_4$ states that certain attacks, namely password guessing attacks, privileged insider attacks, de-synchronization attacks, replay attacks, stolen verifier attacks, node impersonation attacks, processor's node capture attacks, DoS attacks and session-specific temporary information attacks, with the exception of breaking the user's smart card, cannot be effectively initiated by the adversary with all capabilities.

In Table 4, for the evaluation of $\dagger_1$ to $\dagger_5$, we can see that all schemes satisfy $\dagger_1$, $\dagger_2$ and $\dagger_3$, i.e., they are password friendly, have sound repairability and provide key agreement. However, $\dagger_4$, $\dagger_5$ differ from the three initially discussed. Specifically, for $\dagger_4$, the scheme of [10,12,13,15] cannot meet this important mutual authentication requirement, because the number of messages (3 or 5) in their scheme does not guarantee that the entities can verify one another's identities. As for $\dagger_5$, only scheme [15] retains more password-related parameters in the server (or drone gateway) and inevitably results in threats to the password security.

Table 4. Summary of functionality comparison among all authentication schemes.

Scheme	**Ref.**	**No. Messages**	**Criteria**									
			$\dagger_1$	$\dagger_2$	$\dagger_3$	$\dagger_4$	$\dagger_5$	$\ddagger_1$	$\ddagger_2$	$\ddagger_3$	$\ddagger_4$	$\ddagger_5$
Srinivas et al.	[10]	**3**	𝕐	𝕐	𝕐	**N**	𝕐	𝕐	**N**	𝕐	**N**	**N**
Aghili et al.	[11]	**4**	𝕐	𝕐	𝕐	𝕐	𝕐	𝕐	**N**	**N**	**N**	**N**
Banerjee et al.	[12]	**3**	𝕐	𝕐	𝕐	**N**	𝕐	𝕐	**N**	**N**	**N**	**N**
Kumar et al.	[13]	**3**	𝕐	𝕐	𝕐	**N**	𝕐	𝕐	𝕐	**N**	**N**	𝕐
Alzahranl et al.	[14]	**4**	𝕐	𝕐	𝕐	𝕐	𝕐	𝕐	**N**	**N**	**N**	**N**
Yao et al.	[15]	**5**	𝕐	𝕐	𝕐	**N**	**N**	**N**	**N**	𝕐	**N**	**N**
Soleymani et al.	[16]	**6**	—	𝕐	𝕐	𝕐	—	**N**	—	𝕐	**N**	—
Our scheme	—	**4**	𝕐	𝕐	𝕐	𝕐	𝕐	𝕐	𝕐	𝕐	𝕐	𝕐

Concerning the five security attributes labeled $\ddagger_1$ through $\ddagger_5$, no existing scheme successfully implements them all. Specifically, regarding $\ddagger_1$, which pertains to user anonymity, the scheme in [15] is lacking. In the scheme in [15], users directly provide their unmasked identities during the registration phase at the registration center. If the gateway is compromised, the users' anonymity is subsequently at risk. In contrast, our scheme enhances the security by utilizing public key cryptography for attributes $\ddagger_3$ and $\ddagger_4$; modular arithmetic for attributes $\ddagger_2$, $\ddagger_4$ and $\ddagger_5$ and a timestamp mechanism for $\ddagger_4$, thus ensuring robust security measures.

(2) *Overhead Comparisons*

To facilitate detailed comparisons of the overhead, Table 6 establishes a reasonable reference length for all necessary terms. It is important to note that, according to NIST's

recommendations [23,37], SHA-256 is a collision-resistant hash function suitable for DIoV. As for the ECC, in this paper, due to the limited resources of DIoV devices, we target the ECC with 80-bit security (i.e., the length of the ECC key is 160 bits). Moreover, our scheme can be easily extended to 128-bit security (i.e., the length of the ECC key will be 256 bits); in this case, one might use future versions of NVIDIA DRIVE AGX Orin [38], which is a high-performance on-board processor.

Subsequently, for the eight authentication schemes, Table 5 presents comparisons of the storage costs, communication costs and time consumption. Additionally, although there are no existing protocols that incorporate access control for the DIoV, several research works on authentication with access control for medical scenarios are available. These can serve as valuable references for the development of authentication with an access control scheme for the DIoV. We have chosen to compare these medical-oriented schemes with our newly designed scheme. Therefore, the costs incurred at the gateway are considered equivalent to those on the DG, and the costs at the sensor node are analogous to those on the processor (PS).

Table 5. Comparisons of storage, communication and computational costs among eight authentication schemes.

Scheme	Ref.	Storage Cost: bits			Communication Cost: bits			Computational Cost: ms		
		User	DG	PS	User	DG	PS	User	DG	PS
Srinivas et al.	[10]	2208 ★	$384n_s + 3072$ ★	$512n_u + 384$ ★	800	1056	1056	9.952	9.042	2.184
Aghili et al.	[11]	1312 ★	$512(n_u + n_s) + 160$	288	1472	1344	448	2.184	2.548	0.728
Banerjee et al.	[12]	1408	$128n_s + 160$	416	416	160	544	1.822	0.913	0.365
Kumar et al.	[13]	1440	$256n_s + 160$	256	416	160	544	1.275	1.095	0.366
Alzahranl et al.	[14]	1536	$128n_s + 160$	416	704	704	416	2.369	1.094	0.367
Yao et al.	[15]	128	$128n_sn_u + 896(n_u + n_s) + 160$	$256 + 384n_u$ ★	1216	1568	3104	3.018	3.746	4.97
Soleymani et al.	[16]	320	$864n_s + 160$	864	800	2208	544	2.98	3.708	2.98
Our scheme	—	$128n_{ps_u} + 800$	$128n_{ps} + 320$	256	1344	2080	1216	3.352	3.823	2.836

★ Here, we do not additionally evaluate the storage costs for the following functions stored: hash $h(\cdot)$ [10], biohash $h_{Bio}(\cdot)$ [11,15] and $PUF(\cdot)$ [15].

Table 6. The reference length of all terms.

Symbols	bits
hash value (h)	256 [37]
ECC point (p)	320 [33,35]
counter (SUM)	32
timestamp (t)	
modulus (n_0)	
secret key (x)	160
random/nonce (r)	
biometric key generation $(BKG(\cdot))$	
user's/processor's identity (ID)	128
tolerance error value (tev)	
symmetric ciphertext size (enc)	
public reproduction parameter (prp)	

For the storage costs consumed, e.g., in computing the storage costs in our scheme, it is the sum of the sizes of parameters $\{PID_i, BKG(\cdot), A_1, A_2, EID_i, F_i(t), SUM\}$ that the user stores. With the reference length in Table 6, the storage costs of the user can be calculated

as $|PID_i| + |A_1| + |A_2| + |EID_i| + |F_i(t)| + |SUM| = 128n_{ps_u} + 800$ bits, where n_{ps_u} is the number of processors that the user can be allocated to query authentication, and the size of biometric key generation function $BKG(\cdot)$ does not need to be quantized.

Similarly, regarding the user storage costs in other schemes, the schemes of Yao et al. (128 bits) [15] and Soleyma et al. (320 bits) [16] have lower storage costs than the other six schemes. As for the DG's storage resources that need to be consumed to realize authentication with access control, this value is inevitably influenced by the two parameters n_u (the number of users) and n_s or n_{ps_u} (the number of medical sensor nodes or processor nodes that the user can be allocated to run authentication), and the cost in our scheme is $128n_{ps_u} + 320$ bits. Thus, the more processors that are involved in authentication, the more storage resources will be consumed. In evaluating the processor's storage costs, our scheme and others, with the exception of the schemes proposed by Srinivas et al. [10] and Yao et al. [15], demonstrate efficient storage utilization. Notably, our scheme holds a significant advantage in comparison to the seven state-of-the-art alternatives.

Regarding the communication costs, our scheme incurs higher overheads for the user and the drone gateway (DG), requiring 1344 bits and 2080 bits, respectively, to ensure secure authentication. The processor's communication costs in our scheme, amounting to 1216 bits, are competitive with those of other schemes.

Regarding the comparison of the time consumed in all eight schemes, recent research [39] shows that the running time for ECC point multiplication is 0.508 ms; symmetric encryption and decryption (AES-128) take about 0.00054 ms each and a hash function (SHA-256) takes about 0.182 ms [23], and this value can be regarded as the runtime of $BKG(\cdot)$. In [13], a PUF operation takes 0.43 ms. Note, that although the running times of these operations were tested in different works, this does not affect the results of the comparison, and the chosen approach has been widely utilized in previous works.

Based on Table 5, to secure the agreed session key, the time consumed by the user is 3.352 ms, which is reduced by **66%** compared with Srinivas et al.'s scheme [10]. The time taken by the DG to complete user and processor authentication is 3.823 ms. To establish a robust session key with forward secrecy, leveraging Diffie–Hellman key exchange technology [15,40], the processor in our scheme requires 2.836 ms, which is a significant 43% reduction compared to Yao et al.'s scheme [15].

Overall, our scheme excels or is at least competitive in terms of the storage, communication and computational time costs, setting a benchmark for efficiency that other schemes struggle to meet, particularly in addressing security vulnerabilities.

6. Conclusions

Our authentication mechanism, tailored to the service type of rescue vehicles, ensures that only authorized users can achieve mutual authentication with a designated processor, thereby enhancing user privacy and data protection. By balancing security with performance, we have developed a more efficient and robust authentication protocol with access control. The command center (CC) specifies the access control, processed by the DG, allowing rescue vehicle users to authenticate and negotiate session keys with the designated processor (PS). The security analysis confirms the protocol's capability for mutual authentication, ensuring the session key's forward secrecy and maintaining users' anonymity. The performance analysis further reveals the protocol's efficiency and cost-effectiveness, particularly highlighting a significant reduction in operational time by at least 66% compared to recent proposals [10]. At present, fog computing [41,42] represents a novel paradigm; it can effectively enhance latency-sensitive applications such as catastrophe management and content transference applications. Thus, our future research will focus on developing an authentication encryption protocol to secure data communication within fog computing environments.

Author Contributions: Validation, methodology, writing—original draft, J.Z. and C.W.; writing—review and editing, Q.C. and R.F.; validation, H.D. and G.X. All authors have read and agreed to the published version of the manuscript.

Funding: This research was mainly supported by the National Key Research and Development Program of China under grant no. 2023YFB2504801 (Data Security Risk Evolution Mechanism and Security Framework for Vehicle-Road-Cloud Multi-Network Convergence System).

Data Availability Statement: Data are contained within the article.

Conflicts of Interest: The authors declare no conflicts of interest.

References

1. Menouar, H.; Guvenc, I.; Akkaya, K.; Uluagac, A.; Kadri, A.; Tuncer, A. UAV-Enabled Intelligent Transportation Systems for the Smart City: Applications and Challenges. *IEEE Commun. Mag.* **2017**, *55*, 22–28. [CrossRef]
2. Miao, J.; Wang, Z.; Ning, X.; Shankar, A.; Maple, C.; Rodrigues, J. A UAV-Assisted Authentication Protocol for Internet of Vehicles. *IEEE Trans. Intell. Transp. Syst.* 2024, *early access*. [CrossRef]
3. Tian, Y.; Yuan, J.; Song, H. Efficient privacy-preserving authentication framework for edge-assisted Internet of Drones. *J. Inf. Secur. Appl.* **2019**, *48*, 102354. [CrossRef]
4. Ever, Y.K. A secure authentication scheme framework for mobile-sinks used in the Internet of Drones applications. *Comput. Commun.* **2020**, *155*, 143–149. [CrossRef]
5. Wang, D.; Wang, P.; Wang, C. Efficient multi-factor user authentication protocol with forward secrecy for real-time data access in wsns. *ACM Trans. Cyber-Phys. Syst.* **2020**, *4*, 1–26. [CrossRef]
6. Manivannan, D.; Moni, S.; Zeadally, S. Secure authentication and privacy-preserving techniques in vehicular ad hoc networks (VANETs). *Veh. Commun.* **2020**, *25*, 100247. [CrossRef]
7. Zhang, J.; Cui, J.; Zhong, H.; Bolodurina, I.; Liu, L. Intelligent Drone-assisted Anonymous Authentication and Key Agreement for 5G/B5G Vehicular Ad-Hoc Networks. *IEEE Trans. Netw. Sci. Eng.* **2021**, *8*, 2982–2994. [CrossRef]
8. Khan, M.; Ullah, I.; Alkhalifah, A.; Rehman, S.; Shah, J.; Uddin, M.; Alsharif, M.; Algarni, F. A Provable and Privacy-Preserving Authentication Scheme for UAV-Enabled Intelligent Transportation Systems. *IEEE Trans. Ind. Inform.* **2022**, *18*, 3416–3425. [CrossRef]
9. Voigt, P.; Von dem Bussche, A. *The EU General Data Protection Regulation (GDPR): A Practical Guide*, 1st ed.; Springer: Berlin, Germany, 2017; p. 10-5555.
10. Srinivas, J.; Das, A.; Kumar, N.; Rodrigues, J. Cloud centric authentication for wearable healthcare monitoring system. *IEEE Trans. Dependable Secur. Comput.* **2018**, *17*, 942–956. [CrossRef]
11. Aghili, S.; Mala, H.; Shojafar, M.; Peris-Lopez, P. LACO: Lightweight three-factor authentication, access control and ownership transfer scheme for e-health systems in IoT. *Futur. Gener. Comp. Syst.* **2019**, *96*, 410–424. [CrossRef]
12. Banerjee, S.; Odelu, V.; Das, A.; Srinivas, J.; Kumar, N.; Chattopadhyay, S.; Choo, K. A provably secure and lightweight anonymous user authenticated session key exchange scheme for internet of things deployment. *IEEE Internet Things J.* **2019**, *6*, 8739–8752. [CrossRef]
13. Kumar, D.; Jain, S.; Khan, A.; Pathak, P. An improved lightweight anonymous user authenticated session key exchange scheme for Internet of Things. *J. Ambient Intell. Humaniz. Comput.* **2020**, *14*, 5067–5083.. [CrossRef]
14. Alzahrani, B.; Chaudhry, S.; Barnawi, A.; Xiao, W.; Chen, M.; Al-Barakati, A. ILAS-IoT: An improved and lightweight authentication scheme for IoT deployment. *J. Ambient Intell. Humaniz. Comput.* **2020**, *13*, 5123–5135.. [CrossRef]
15. Yao, H.; Yan, Q.; Fu, X.; Zhang, Z.; Lan, C. ECC-based lightweight authentication and access control scheme for IoT E-healthcare. *Soft Comput.* **2022**, *26*, 4441–4461. [CrossRef]
16. Soleymani, S.; Goudarzi, S.; Anisi, M.; Jindal, A.; Kama, N.; Ismail, S. A privacy-preserving authentication scheme for real-time medical monitoring systems. *IEEE J. Biomed. Health Inform.* **2023**, *27*, 2314–2322. [CrossRef]
17. Wang, C.; Wang, D.; Tu, Y.; Xu, G.; Wang, H. Understanding node capture attacks in user authentication schemes for wireless sensor networks. *IEEE Trans. Dependable Secur. Comput.* **2020**, *19*, 507–523. [CrossRef]
18. Zou, S.; Cao, Q.; Lu, R.; Wang, C.; Xu, G.; Ma, H.; Cheng, Y.; Xi, J. A robust and effective 3-factor authentication protocol for smart factory in IIoT. *Comput. Commun.* **2024**, *220*, 81–93. [CrossRef]
19. Zou, S.; Cao, Q.; Wang, C.; Huang, Z.; Xu, G. A robust two-factor user authentication scheme-based ECC for smart home in IoT. *IEEE Syst. J.* **2022**, *16*, 4938–4949. [CrossRef]
20. Das, M. Two-factor user authentication in wireless sensor networks. *IEEE Trans. Wirel. Commun.* **2009**, *8*, 1086–1090. [CrossRef]
21. Awais, S.; Wu, Y.; Mahmood, K.; Muhammad, H.; Badar, S.; Kharel, R.; Das, A. Provably secure fog-based authentication protocol for VANETs. *Comput. Netw.* **2024**, *246*, 110391. [CrossRef]
22. Su, H.; Dong, S.; Wang, N.; Zhang, T. An efficient privacy-preserving authentication scheme that mitigates TA dependency in VANETs. *Veh. Commun.* **2024**, *45*, 100727. [CrossRef]
23. El-Zawawy, M.; Brighente, A.; Conti, M. Authenticating Drone-Assisted Internet of Vehicles Using Elliptic Curve Cryptography and Blockchain. *IEEE Trans. Netw. Serv. Manag.* **2023**, *20*, 1775–1789. [CrossRef]
24. Wang, C.; Wang, D.; Duan, Y.; Tao, X. Secure and Lightweight User Authentication Scheme for Cloud-Assisted Internet of Things. *IEEE Trans. Inf. Forensic Secur.* **2023**, 18, 2961–2976. [CrossRef]
25. Han, Y.; Guo, H.; Liu, J.; Ehui, B.; Wu, Y.; Li, S. An enhanced multi-factor authentication and key agreement protocol in Industrial Internet of Things. *IEEE Internet Things J.* **2024**, *11*, 16243–16254. [CrossRef]

26. Shen, H.; Wang, T.; Chen, J.; Tao, Y.; Chen, F. Blockchain-based Batch Authentication Scheme for Internet of Vehicles. *IEEE Trans. Veh. Technol.* 2024, *early access*. [CrossRef]
27. Chen, Y.; Zhang, J.; Wei, X.; Wang, Y.; Cui, J. Cross-Domain Authentication Scheme for Vehicles Based on Given Virtual Identities. *IEEE Internet Things J.* **2024**, *11*, 15869–15879.. [CrossRef]
28. Li, X.; Niu, J.; Bhuiyan, M.; Wu, F.; Karuppiah, M.; Kumari, S. A robust ecc-based provable secure authentication protocol with privacy preserving for industrial internet of things. *IEEE Trans. Ind. Inform.* **2018**, *14*, 3599–3609. [CrossRef]
29. NIST. Recommendation for Key Management-Part 1: General. Standard SP 800-57 (Part 1, Rev. 5). 2020. Available online: https://csrc.nist.gov/pubs/sp/800/57/pt1/r5/final (accessed on 13 May 2020).
30. Nannipieri, P.; Crocetti, L.; Matteo, S.; Fanucci, L.; Saponara, S. Hardware Design of an Advanced-Feature Cryptographic Tile within the European Processor Initiative. *IEEE Trans. Comput.* 2023, *early access*. [CrossRef]
31. LaMacchia, B.; Lauter, K.; Mityagin, A. Stronger security of authenticated key exchange. In Proceedings of the International Conference on Provable Security, ProvSec 2007, Berlin, Germany, 1–2 November 2007; pp. 1–16.
32. Wang, D.; Wang, P. Two birds with one stone: Two-factor authentication with security beyond conventional bound. *IEEE Trans. Dependable Secur. Comput.* **2018**, *15*, 708–722. [CrossRef]
33. Koblitz, N. Elliptic curve cryptosystems. *Math. Comput.* **1987**, *48*, 203–209. [CrossRef]
34. Li, X.; Peng, J.; Obaidat, M.; Wu, F.; Khan, M.; Chen, C. A secure three-factor user authentication protocol with forward secrecy for wireless medical sensor network systems. *IEEE Syst. J.* **2020**, *14*, 39–50. [CrossRef]
35. Matteo, S.; Baldanzi, L.; Crocetti, L.; Nannipieri, P.; Fanucci, L.; Saponara, S. Secure Elliptic Curve Crypto-Processor for Real-Time IoT Applications. *Energies* **2021**, *14*, 4676. [CrossRef]
36. Wang, C.; Xu, G. Cryptanalysis of three password-based remote user authentication schemes with non-tamper-resistant smart card. *Secur. Commun. Netw.* **2017**, *2017*, 1619741. [CrossRef]
37. Cryptographic Key Length Recommendation. Available online: https://www.keylength.com/en/4/ (accessed on 24 May 2020).
38. Available online: https://www.nvidia.com/content/dam/en-zz/Solutions/self-driving-cars/drive-platform/auto-print-drive-product-brief-final.pdf (accessed on 13 May 2020).
39. Wang, C.; Wang, D.; Xu, G.; He, D. Efficient Privacy-Preserving User Authentication Scheme with Forward Secrecy for Industry 4.0. *Sci. China-Inf. Sci.* **2022**, *65*, 112301. [CrossRef]
40. Ma, C.-G.; Wang, D.; Zhao, S.-D. Security flaws in two improved remote user authentication schemes using smart cards. *Int. J. Commun. Syst.* **2014**, *27*, 2215–2227. [CrossRef]
41. Hashemi, S.; Sahafi, A.; Rahmani, A.; Bohlouli, M. Service and Energy Management in Fog Computing: A Taxonomy Approaches, and Future Directions. *J. Electr. Comput. Eng. Innov.* **2024**, *12*, 15–38.
42. Sadri, A.; Rahmani, A.; Saberikamarposhti, M.; Hosseinzadeh, M. Fog data management: A vision, challenges, and future directions. *J. Netw. Comput. Appl.* **2021**, *174*, 102882. [CrossRef]

Disclaimer/Publisher's Note: The statements, opinions and data contained in all publications are solely those of the individual author(s) and contributor(s) and not of MDPI and/or the editor(s). MDPI and/or the editor(s) disclaim responsibility for any injury to people or property resulting from any ideas, methods, instructions or products referred to in the content.