Research Article

An Enhanced RFID-Based Authentication Protocol using PUF for Vehicular Cloud Computing

Vikas Kumar, Rahul Kumar, Srinivas Jangirala, Saru Kumari, Sachin Kumar, and Chien-Ming Chen

1Department of Mathematics, SSV College Hapur, Hapur, Uttar Pradesh, India
2Jindal Global Business School, O. P. Jindal Global University, Sonipat, Haryana 131001, India
3Department of Mathematics, Ch. Charan Singh University, Meerut, U P, India
4Department of Computer Engineering, Ajay Kumar Garg Engineering College, Ghaziabad 201009, India
5College of Computer Science and Engineering, Shandong University of Science and Technology, Qingdao, Shandong, China

Correspondence should be addressed to Chien-Ming Chen; chienmingchen@ieee.org

Received 8 March 2022; Revised 2 May 2022; Accepted 19 May 2022; Published 30 July 2022

Academic Editor: Jie Cui

Copyright © 2022 Vikas Kumar et al. This is an open access article distributed under the Creative Commons Attribution License, which permits unrestricted use, distribution, and reproduction in any medium, provided the original work is properly cited.

RFID (radio frequency identification) is an Internet of Things (IoT) enabling technology. All physical devices can be connected to the Internet of Things thanks to RFID. When RFID is extensively utilized and fast increasing, security and privacy concerns are unavoidable. Interception, manipulation, and replay of the wireless broadcast channel between the tag and the reader are all possible security threats. Unverified tags or readers provide untrustworthy messages. IoT requires a safe and consistent RFID authentication system. PUFs are also physical one-way functions made up of the unique nanoscopic structure of physical things and their reactivity to random occurrences. PUF includes an unclonable feature that takes advantage of physical characteristicsto boost security and resistance to physical attacks. We analyze the security of the RSEAP2 authentication protocol that has been recently proposed by Safkhani et al., a hash-based protocol, and elliptic curve cryptosystem-based protocol. Our security analysis clearly shows important security pitfalls in RSEAP2 such as mutual authentication, session key agreement, and denial-of-service attack. In our proposed work, we improved their scheme and enhanced their version using physically unclonable function (PUF), which are used by the proposed protocol in tags. This research proposes a cloud-based RFID authentication technique that is both efficient and trustworthy. To decrease the RFID tag’s overhead, the suggested authentication approach not only resists the aforementioned typical attacks and preserves the tag’s privacy, but also incorporates the cloud server into the RFID system. According to simulation results, our approach is efficient. Moreover, according to our security study, our protocol can withstand a variety of attacks, including tracking, replay, and desynchronization assaults. Our scheme withstands all the 18 security features and further consumes the computation cost as 14.7088 ms which is comparable with the other schemes. Similarly, our scheme consumes the communication cost as 672 bits during the sending mode and 512 bits during the receiving mode. Overall, the performance of our proposed method is equivalent to that of related schemes and provides additional security features than existing protocols. Mutual authentication, session key generation, and ephemeral session security are all achieved. Using the real-or-random concept, we formalize the security of the proposed protocol.

1. Introduction

Recognition technologies are deserving of our attention as they are both essential parts of the Internet of Things. Recognition of barcodes, optical characters, biometric identity, and magnetic card identification and contact IC card identification are all examples of traditional automated identification technologies. However, when employed in the IoT, they have a number of drawbacks. Bar codes, for example, can only hold a limited amount of data; optical character recognition is too expensive; biological recognition is flawed; and magnetic card and contact IC card
identification need intimate touch, which is inflexible. Currently, some of these identification methods are unable to protect personal information [1]. In contrast, RFID is a noncontact automatic identification technology that does not need mechanical or visual contact between the system and the target, and security protections can help keep user information private. Because of these advantages, RFID has emerged as one of the most promising IoT technologies [2].

An RFID system consists of RFID tags, RFID readers, and a database server. Tag-affixed objects are uniquely identifiable, and their identifying information is saved. They communicate with the reader using radio waves. In a typical RFID system, the database server is a local back-end server.

When RFID devices generate a large number of data, back-end servers’ performance is limited. Cloud computing overcomes this problem in the IoT context. As a result, the integration of the cloud platform with the RFID system is required [2, 3]. RFID systems’ reliability and data processing capabilities have dramatically enhanced since the introduction of cloud computing. Almost all of the data acquired by RFID sensors are processed on the cloud, which can aid in the resolution of issues such as data loss and latency [4]. In the IoT, the most commonly used public cloud servers are only semi-trustworthy. Because of the properties described above, the RFID system is vulnerable to attack. As a result, IoT necessitates the use of a secure and reliable RFID authentication system.

Similarly, a number of protocols based on physically unclonable functions (PUFs) have been proposed [12–14]. PUFs are, in reality, physical one-way functions derived from the unique nanoscopic structure of physical things (e.g., integrated circuits, crystals, magnets, lenses, solar cells, or papers) and their reactivity to random occurrences. The quirks in the manufacturing process of the items are responsible for the innate uniqueness of the structure and reactivity. It enables for both the unique identification and authentication of an object. Furthermore, it is considered that copying an object’s PUF (and hence the object itself) is impossible, which might be seen as a security-by-design feature that prevents impersonation and cloning attacks. As a result, PUFs are regarded as a trustworthy and well-known physical security method for developing IoT authentication protocols. Physical devices are protected by PUF-based protocols, which are resistant to physical attacks and provide multilayer protection. Furthermore, even if the device is stolen, the attacker will not be able to use the PUF. However, the majority of proposed VANET solutions are still subject to different security concerns such as replay attacks, impersonation attacks, forgery attacks, and non-repudiation attacks. As a result, it is critical to build a viable VANET solution to address the existing issues.

2. Literature and Related Works

Several RFID authentication schemes have used elliptic curve cryptography in recent years (ECC). Due to the difficulties of resolving the discrete logarithm problem (DLP), ECCs have demonstrated their efficiency in ensuring security and privacy. The state-of-the-art of ECC-based RFID, mobile computing, and VCC authentication protocols are reviewed in this section and are shown in Table 1. Also, the details of PUF-based recent works are given in Table 2.

2.1. Problem Definition. Security protocols, such as authentication methods, are supposed to ensure the confidentiality, integrity, and availability (CIA triangle) of security. The parties to the protocols must be able to authenticate and synchronize with one another at any moment. Desynchronization attacks can break this condition by blocking protocol messages or forcing protocol parties to modify their shared secret values to different values, preventing the parties from authenticating each other and destroying service availability. Many protocols have been developed in the literature to satisfy CIA security standards; however, multiple instances of attacks [2, 10–14] against them show that they have failed to achieve the needed security. As a result, attempts to build a secure protocol are still continuing, and new attacks are emerging that provide designers fresh insight into how to (not) design a protocol. As a result of these assaults and security evaluations, the protocols have progressed.

2.2. Motivation and Contributions. In recent years, a number of key agreement and authentication techniques have been created. Most of these protocols have a greater calculation cost, making them unsuitable with devices with limited resources. We also noticed that the literature reviewed above did not take into account the physical factors of security for vehicle RFID communication systems in VCC situations.

A PUF-based protocol is capable of dealing with physical security risks. Even stealing the PUF from the on-board memory will not allow an attacker to obtain it. As a result, for VCC, we developed a PUF-enabled RFID-based authentication protocol. The following are some of the many contributions made by this research:

1. To build an authentication protocol for VCC communication, the system and threat models are defined first.
2. We created a PUF-enabled RFID-based authentication mechanism using the hypothesized attack model.
3. To keep the proposed protocol’s cost minimal, only fundamental cryptographic operations such as ECC, XOR, concatenation, and hash function are used. PUF is also used to protect against recognized physical security risks.
4. Our approach ensures that possible security threats are avoided, based on formal and informal security assessments.
5. The results of the performance study show that our protocol is superior to other similar protocols.
2.3. Roadmap of Article. The rest of the article is structured as follows: The preliminaries are presented in Section 3. The RSEAP2 system is described in detail in Section 4. We give a security study of the RSEAP2 protocol as well as various efficient and strong attacks against it in Section 5. The improved protocol is presented in Section 6. In Section 7, we provide a verifiable security analysis of our approach. The performance analysis is presented in Section 8. Section 9 concludes the article.

3. Definitions and Mathematical Preliminaries

The key size comparison between the public-key cryptosystems like ECC and RSA shows that the communication

| Scheme                  | Year | Cryptographic techniques                                                                 | Advantages                                                                 | Drawbacks/limitations                                                                 |
|-------------------------|------|----------------------------------------------------------------------------------------|--------------------------------------------------------------------------|--------------------------------------------------------------------------------------|
| Jiang et al. [3]        | 2018 | (i) Uses “one-way cryptographic hash function”                                         | (i) Fits for vehicular cloud networking environment                      | (ii) Prone to “replay attack”                                                        |
| Alamr et al. [5]        | 2018 | (i) Based on “RFID”                                                                    | (i) Applicable in IoT environment                                        | (ii) Vulnerable to “data integrity and key compromise”                                |
| Dinarvand and Barati [6]| 2019 | (i) Based on “RFID technology” uses “one-way cryptographic hash function”              | (i) Does not fit for generic IoT networking environment                  | (ii) No “impersonation and key compromise”                                            |
| Bagga et al. [1]        | 2018 | (i) Based on “three factors (user mobile device, user password, and personal biometrics)” | (i) Applicable in industrial IoT environment                              | (ii) Vulnerable to “known session key attack”                                         |
| Kumar et al. [7]        | 2020 | (i) Based on “three factors (smart card, user password, and biometrics)”              | (i) Fits for generic IoT networking environment                           | (ii) No “formal security” analysis                                                   |
| Jiang et al. [4]        | 2018 | (i) Based on “three factors (user mobile device, user password, and personal biometrics)” | (i) Applicable in cloud environment                                      | (ii) Vulnerable to “known session key attack”                                         |
| Hosseinzadeh et al. [8] | 2020 | (i) Based on “RFID systems” uses “one-way cryptographic hash function”                 | (i) Fits for IoT networking environment                                  | (ii) No “session key agreement”                                                      |
| Zhu [9]                 | 2020 | (i) Based on “RFID systems and quadratic residue” uses “Gong-Needham-Yahalom (GNY) logic” | (i) Fits for healthcare environment                                       | (ii) Desynchronization issues                                                       |
| Gabsi et al. [10]       | 2021 | (i) Based on “RFID systems” uses “arithmetic calculation of ECC”                      | (i) Fits for communicating reader to reader environment                   | (ii) Not suitable for cloud environment                                               |
| Mishra et al. [11]      | 2018 | (i) Based on “three factors (user mobile device, user password, and personal biometrics)” | (i) Applicable in industrial IoT environment                              | (ii) Vulnerable to “known session key attack”                                         |
| Safkhani et al. [2]     | 2021 | (i) Based on “RFID and ECC cryptosystem”                                                | (i) Fits for IoT networking environment                                  | (ii) Could not resist “denial-of-service”                                            |

Table 1: Summary of cryptographic techniques applied and limitations of previous existing user authentication mechanisms.
| Scheme            | Year | Cryptographic techniques and environment                                                                 | Advantages                                                                 | Drawbacks/limitations                                                                 |
|-------------------|------|-----------------------------------------------------------------------------------------------------------|----------------------------------------------------------------------------|---------------------------------------------------------------------------------------|
| Xu et al. [15]    | 2021 | Based on PUF is applicable for RFID healthcare systems                                                    | Fits for healthcare systems                                               | Does not support "revocability." Vulnerable to "know session key attack"            |
| Gope and Sikdar   | 2019 | Based on smart grid communication systems. The lightweight cryptographic primitives such as physically unclonable functions and one-way hash function is utilized | A novel privacy-aware authenticated key agreement scheme which can not only ensure secure communication between smart meters and the service providers, but also the physical security of smart meters | (i) Does not support "revocability and password/biometric update"                     |
| Cao et al. [16]   | 2021 | (i) Based on "three factors (user mobile device, user password, and personal biometrics" (ii) Applies "ECC cryptographic technique" (iii) Uses "fuzzy extractor for biometric verification" | (i) Applicable in smart grid environment and data collection scheme | (ii) Vulnerable to "known session key attack"                                        |
| Zhang et al. [17] | 2020 | Key distribution in wireless sensor networks                                                                | It did not only save the storage overhead, but also provided perfect resilience against sensor capture attacks | This cannot resist anonymity, traceability, and forward secrecy attacks               |
| Mall et al. [18]  | 2022 | This approach is a survey on PUF-based authentication and key agreement protocols for IoT, WSN, and smart grids | (i) This survey paper can be utilized to understand the technologies such as IoT, WSN, and smart grids and the way to address the AKA in these technologies (ii) Systematically and taxonomically examine and discuss with pros and cons of AKA applications to the fast-growing areas of IoT, WSNs, and smart grids based on a meticulous survey of existing literature | This study fails to address the security pitfalls which can integrate all these technologies |
| Liu et al. [19]   | 2021 | Key distribution for dynamic sensor networks                                                                | Compared with traditional key predistribution schemes, the proposal reduces the storage overhead and the key exposure risks and thereby improves the resilience against node capture attacks | This study cannot be applied to the current technologies such as IoT and cloud computing |
| Mukhopadhyay [20]| 2016 | PUFs as promising tools for security in Internet of Things. This article discusses about security violation in the authentication of a commercial IoT | (i) Studied the lightweight construction of PUFs (ii) Proof context test-bed simulations were presented for commercially available tools to show how PUFs can interact with other IoT nodes to provide overall security | This study fails to address the security features and how they can be applied for the AKA protocols |
| Wang et al. [21]  | 2021 | Blockchain and lightweight authentication protocol for wireless medical sensor networks. Applies "fuzzy extractor for biometric verification" | Incorporated for blockchain and wireless medical sensor networks           | (i) Desynchronization attacks                                                         |
|                   |      |                                                                                                           |                                                                            | (ii) Excess communication cost                                                       |
messages can utilize the elliptic curve cryptosystem to reduce the communication bandwidth. The key size comparison between ECC and RSA is given in Table 3.

3.1. Background of ECC. “Let $E$ denotes an elliptic curve over the prime finite field $F_q$, where $q$ be the large prime number. An equation of elliptic curve over $F_q$ is given by $y^2 = x^3 + ax + b$ (mod $q$), where $a, b \in F_q$. The elliptic curve is said to be nonsingular if $4a^3 + 27b^2 \equiv 0$ (mod $q$). The additive elliptic curve group $G$ is defined as $G = \{(u, v); u, v \in F_q, (u, v) \in E\} \cup \{\Phi\}$, where the point $\Phi$ is known as asymptotic point which work as the identity element or zero element in $G$.”

Some operations on the group $G$ are as follows [2, 7]:

1. Let $\mathcal{V} = (u, v) \in G$, then define $-\mathcal{V} = (u, -v)$ and $\mathcal{V} + (-\mathcal{V}) = \Phi$.

2. If $\mathcal{V}_1 = (u_1, v_1)$ and $\mathcal{V}_2 = (u_2, v_2) \in G$, then $\mathcal{V}_1 + \mathcal{V}_2 = (u_3, v_3)$, where $u_3 = u_1^2 - u_2 - u_1 - u_2$ mod $q$ and $v_3 = v_1 u_2 - v_2 u_1 - v_1 u_2$ mod $q$.

3. Let $\mathcal{V} = (u, v) \in G$, then scalar multiplication in $G$ is defined as: $\eta \cdot \mathcal{V} = \mathcal{V} + \mathcal{V} + \ldots + \mathcal{V}$ ($\eta$ times).

4. If $g$ is the generator of $G$ with order $\eta$, then $\eta \cdot g = \Phi$.

(a) “Elliptic curve discrete logarithm problem (ECDLP)” Finding $\mu \in \mathbb{Z}_q^*$ such that $\mathcal{V} = \mu \cdot \mathcal{V}$, for a given $\mathcal{V}, \mathcal{V} \in G$ is difficult.

(b) “Elliptic curve computational Diffie–Hellman problem (ECCDH): If $g$ is the generator of $G$ and $a, g, b, g$ are supplied ($g, a, g, b),$ then computing $a \cdot b \cdot g$ in $G$ is difficult.

3.2. Physically Unclonable Function. The PUF hardware primitive accepts a challenge $\mathcal{C}$ and generates the matching response $R$ from the physical properties of its integrated chip (IC) and $C$. A PUF may easily be thought of as a one-way function $R = PUF(C)$ since both the accepted challenge $C$ and the produced answer $R$ are bit strings [14].

In essence, PUF security is based on the fact that, even if various ICs use the same production processes, each IC will be somewhat different owing to manufacturing variances. The following are the characteristics of PUF [15]:

(i) Uniqueness: A PUF cannot be duplicated;

(ii) Unidirectionality: In the real manufacturing circuit, the variances between input and output function mapping are both fixed and unpredictable. It is the hardware counterpart of the one-way function in this regard;

(iii) Invulnerability: Any effort to tamper with the device containing the PUF will cause the PUF to modify its behaviour and, as a result, it will be destroyed [14].

3.3. Network Model. Figure 1 represents the architecture which we applied for the design of communication among the participants. The RFID tag communicates with the

| Scheme                  | Year | Cryptographic techniques and environment                                           | Advantages                                                                                                                                  | Drawbacks/limitations                                      |
|-------------------------|------|-----------------------------------------------------------------------------------|-------------------------------------------------------------------------------------------------------------------------------------------|------------------------------------------------------------|
| Lee and Chen [23]       | 2021 | Lightweight fog computing-based authentication protocols using physically unclonable functions for Internet of medical Things | (i) The proposed protocols use lightweight cryptographic operations, including a one-way cryptographic hash function, the barrel shifter physically unclonable function (BS-PUF) | This study is restricted to fog environment                |
| Hassija et al. [24]     | 2021 | A survey on supply chain security: application areas, security threats, and solution architectures | (i) This article discusses the supply chain’s security critical application areas and presents a detailed survey of the supply security issues in the existing supply chain architecture | This study is a survey work and fails to address the security features and how they can be applied for the AKA protocols |
roadside RFID reader and thereby the communication passes through the vehicular cloud server. In order to communicate efficiently, the communication parties have to undergo the authentication and key agreement phase to establish a session key. More details regarding how the participants actually take part in the authentication and key agreement and communication process is discussed in the next section.

3.4. Threat Model. The "CK-adversary model" is widely regarded as the “current de facto standard model in modeling key-exchange protocols.” Using the “CK-adversary model,” the adversary A can “deliver messages (as in the DY model),” and in addition, A can also “compromise other information, such as session state, private keys, and session keys.” “Since the sessions as procedures run inside a party, the internal state of a session is well-defined. An important point here is that what information is included in the local state of a session. For instance, the information revealed in this way may be the exponent used by a party. Typically, the revealed information will include all the local state of the session and its subroutines, except for the local state of the subroutines that directly access the long-term secret information.” Therefore, it is important that “the leakage of some forms of secret information, such as session ephemeral (short-term) secrets or session key, should have the least possible effect on the security of other secret credentials of the communicating entities in an authenticated key-exchange protocol.” We demonstrate that the proposed technique is secure against well-known attacks and offers session key security and strong credentials’ privacy under the CK-adversary model through a comprehensive formal security analysis.

| S. No | ECC key size (bits) | RSA key size (bits) | Key size ratio |
|-------|---------------------|--------------------|---------------|
| 1     | 163                 | 1024               | 1:6           |
| 2     | 256                 | 3072               | 1:12          |
| 3     | 384                 | 7680               | 1:20          |
| 4     | 512                 | 15360              | 1:30          |

3.5. Security Requirements for an IoT-Based RFID Communication System. To the best of our knowledge and based on the available literature, many authentication algorithms for RFID communication systems have been proposed in recent years. The best ways for making RFID systems appropriate for a wide variety of applications are authentication and key agreement. Several forms of security threats might arise during the transfer of messages between RFID tags and readers.

Any authentication mechanism attempting to secure a viable RFID-based system should meet the following security requirements: Impersonation attack: By repeating a message recorded from the channels, an attacker might try to imitate genuine protocol participants (such as the cloud database server, RFID reader, or RFID tag). At all costs, any impersonation should be avoided.

Replay attack: In this attack, an outsider tries to deceive other certified participants by restating intercepted data. This attack is aimed at a user whose data have been intercepted by an untrustworthy third party. Mutual authentication: The authentication procedure takes place between the RFID tag and the back-end database server. Messages are exchanged across an unprotected communication route between the tag, reader, and server. This is the most crucial feature of any authentication system. Mutual authentication must also be accomplished with all three RFID system players present.

Tag anonymity: This is the most critical and required security criterion to reduce forgeries and assure security. Furthermore, the RFID authentication method retains its anonymity if an opponent is unable to trace an RFID tag during message transmission over a public channel. There are two types of anonymity, namely strong anonymity and weak anonymity. Furthermore, in order to protect their security and privacy, participants in IoT communication do not reveal their true identities.

Man-in-the-middle attack: In this attack, an adversary listens to the transmitted data before attempting to remove or change the data supplied to recipients.

Insider attack: Any insider can play the role of adversary in the RFID communication system.
Desynchronization attack: If a protocol’s authentication is reliant on shared values, an adversary may cause desynchronization difficulties. If the shared data are updated by the server but the tag is not, the server might be unable to validate the tag in the future. Attempts to desynchronize should be avoided at all costs.

Untraceability: Untraceability in the RFID communication system means that no one can track the participants’ activity patterns or their relayed messages.

Session key agreement: A session key agreement will be made between users and their mobile devices, as well as the network control centre, following the successful deployment of the proposed protocol.

Confidentiality: The security of RFID communications between the tag and the reader is ensured by encrypting shared secrets on the public channel.

Perfect forward secrecy: This is utilized in the authentication protocol architecture to keep previously transmitted messages private, so that an adversary who obtains the entities private and public keys will be unable to deduce a past session key.

Availability: The authentication and key agreement mechanism between the RFID tag and the RFID back-end database server operates continuously in an RFID system. To accomplish the characteristic of accessibility, the shared secret information between the RFID tag and the RFID back-end database server must be updated in most authentication procedures. However, security issues such as denial-of-service (DoS) or desynchronization attacks may cause this process to be disrupted. As a result of these problems, the RFID system’s efficiency may be jeopardized. Hence, this issue should be considered while creating an authentication mechanism.

4. RSEAP2 Protocol

We offer a brief explanation of RSEAP2 [2] in this section. The tag $T_j$ and the cloud database server $S$ interact through the reader $R_j$ to establish a session key $SK_{ST}$ in this protocol. It is divided into two parts. The tag enrollment or startup phase is the first step, in which the tag talks with $S$ via a secure connection to provide the needed data. The login and authentication phase is the second phase of the protocol, and it is used to perform mutual authentication and share the session key $SK_{ST} = SK_{TS}$. This part of the communication takes place via a public network. We have made use of the notations as shown in Table 4.

In the initialization phase of RSEAP2, the server $S$ chooses an elliptic curve $E(F_q)$ over $F_q$ and a generator $g$ over $G$. It also selects $x_s, g$ as its secret key and its public key will be $x_s, g$. Any tag $T_j$ which aims to register with $S$ inputs its $ID_{T_j}$ and $pw_{T_j}$, generates a random value $R_{T_j} = g^{j_{T_j}}$, computes $PWT = h(ID_{T_j}[PW]),$ and sends the tuple $M_{R_1} = \{PW, ID_{T_j}, TS_{R_1}\}$ to $S$. Once $S$ receives $M_1$, verifies the timestamp, that is $|TS_{R_2} - TS_{R_1}| \leq t_{TS} \times \Delta T$ at the first. Next, it generates $sn_{T_j}$ and sets it as the $T_j$’s serial number, computes $X_{T_j} = h(s_{T_j}[ID_{T_j} \omega x_{T_j}]), A_{T_j} = h(PWT)$, and stores $s_{T_j}$ corresponding to $ID_{T_j}$. It then sends tuple $M_{R_2} = \{A_{T_j}, B_{T_j}, sn_{T_j}, g, x_s, g, G, h(.)\}$ to $T_j$. The tag $T_j$ stores $\{A_{T_j}, B_{T_j}, sn_{T_j}, g, x_s, g, G, h(.)\}$.

The description of the protocol is as follows:

L1. $T_j$ uses its credentials $(ID_{T_j}, pw_{T_j}, R_{T_j})$, computes $PWT = h(ID_{T_j}[pw_{T_j} \omega R_{T_j}]), X_{T_j} = g^{R_{T_j}},$ and verifies $A_{T_j} = h(PWT)$. If verification is successful, $T_j$ generates $\alpha \in F^*$, calculates $\alpha g$ and $W_1 = h(\alpha \omega X_{T_j}^{\alpha} ID_{T_j})$, and sends $M_1 = \{ID_{T_j} \omega \alpha \omega g, \alpha \omega x_{T_j} \omega \alpha \omega g, TS_{L_{A1}}\}$ to the reader $R_j$.

L2. The reader checks the timestamp, that is, $|TS_{L_{A2}} - TS_{L_{A1}}| \leq t_{TR} \times \Delta T$, generates $\beta, g$, and then sends $M_2 = \{\alpha \omega X_{T_{L_{A1}}} \omega \beta \omega g, (h((TS_{L_{A1}} \omega TS_{L_{A2}}) \omega (x_{R_i} \omega g))) | (ID_{R_i}) \omega \beta \omega x_{R_i} \omega g\}$ to $S$.

L3. Once $S$ receives $M_2$, it verifies the timestamps, that is, $|TS_{L_{A4}} - TS_{L_{A3}}| \leq t_{TS} \times \Delta T$ and $|TS_{L_{A4}} - TS_{L_{A3}}| \leq t_{TS} \times \Delta T$. Next $S$ extracts $h(TS_{L_{A1}} \omega TS_{L_{A2}} \omega (x_{R_i} \omega g)) | (RID_{L_{A1}} \omega (x_{R_i} \omega g)) | (ID_{R_i}) \omega (x_{R_i} \omega g)| (TS_{L_{A1}} \omega TS_{L_{A2}} \omega (x_{R_i} \omega g)) \omega (x_{R_i} \omega g)$ from the database, and evaluates $h(TS_{L_{A1}} | TS_{L_{A2}} | (x_{R_i} \omega g))$ to authenticate $R_i$. After the successful authentication on $R_i$, parameters, $S$ extracts $ID_{R_i} \omega W_{i}^*$, verifies $W_1^* = h((\alpha \omega X_{T_j}^{\alpha} ID_{T_j}) | (TS_{L_{A1}}),)$ retrieves the related $s_{T_j}$ using $ID_{T_j}$, computes $X_{T_j} = h(s_{T_j} \omega ID_{T_j} \omega x_{T_j})$, and verifies $W_1^* = h((\alpha \omega g) \omega (x_{T_j}^{\alpha} ID_{T_j}) | (TS_{L_{A1}})).$ Further generating $\gamma \in \mathbb{F}_q$, computing the session key $SK_{ST} = h((ID_{T_j} \omega x_{T_j} \omega g) | (x_{T_j} \omega g) | (x_{T_j} \omega g) | (x_{T_j} \omega g) | (x_{T_j} \omega g) | (x_{T_j} \omega g) \omega (x_{T_j} \omega g) \omega (x_{T_j} \omega g) \omega (x_{T_j} \omega g) \omega (x_{T_j} \omega g) \omega (x_{T_j} \omega g) \omega (x_{T_j} \omega g) \omega (x_{T_j} \omega g))$ to $R_j$.

L4. $R_j$ verifies the timestamp, that is, $|TS_{L_{A5}} - TS_{L_{A3}}| \leq t_{TS} \times \Delta T$ and $|TS_{L_{A5}} - TS_{L_{A3}}| \leq t_{TS} \times \Delta T$, and then computes $SK_{TS} = h((ID_{T_j} \omega x_{T_j} \omega g) | (x_{T_j} \omega g) \omega x_{T_j} \omega g) | (x_{T_j} \omega g) | (x_{T_j} \omega g) | (x_{T_j} \omega g) \omega (x_{T_j} \omega g) \omega (x_{T_j} \omega g) \omega (x_{T_j} \omega g) \omega (x_{T_j} \omega g) \omega (x_{T_j} \omega g) \omega (x_{T_j} \omega g) \omega (x_{T_j} \omega g))$ and checks $W_2^* = h(ID_{T_j} \omega SK_{ST}).$ It also sets $SK_{TS}$ as the session key.

5. Security Analysis of RSEAP2

5.1. Inefficient Mutual Authentication Attack. On receiving the message $M_2$ from the reader $R_j$, the cloud database server $S$ extracts and computes to validate the user and reader. The details are as follows:

(1) The cloud server performs the computations and validates the timestamps such as
Table 4: Notations along with their descriptions.

| Symbol         | Description                                                                 |
|----------------|-----------------------------------------------------------------------------|
| $T_i, R_j, S$  | $i^{th}$ tag, $j^{th}$ reader, and cloud server                             |
| $ID_{Ti}, ID_{Rj}$ | Unique identities of $T_i$ and $R_j$                                      |
| $pwTi$          | Password of $T_i$                                                           |
| $x_s$           | Long-term private secret key of the S                                       |
| $\|$            | Operations of bitwise concatenation and bitwise XOR                         |
| $SK_{ST}/SK_{ST}$ | Session key established between $T_i$, and $S$                           |
| PUF             | Physically unclonable function                                              |
| $h(\cdot)$     | Cryptographic collision-resistant one-way hash function                     |
| $\alpha, \beta$ | Random numbers                                                             |
| $\Delta$        | Maximum threshold transmission delay allowed                                |
| $\Delta T$      | Timestamps used at $i^{th}$ transmission                                     |
| $i = j$          | Validation check, if expression $i$ matches $j$ or not                     |
| $A$             | An adversary                                                                |

$$|TS_{LA4} - TS_{LA1}| \leq t_{TS}x\Delta T \quad \text{and} \quad |TS_{LA4} - TS_{LA3}| \leq t_{RS} \times \Delta T.$$  

(2) Next $S$ extracts $h^*(((TS_{LA1}||TS_{LA2}||x_{Rj}||g))||RID_j = h(((TS_{LA1}||TS_{LA2}||x_{Rj}||g))||RID_j$, retrieves $x_{Rj}||g$ from the database, and evaluates $h(((TS_{LA1}||TS_{LA2}||x_{Rj}||g))$ to authenticate $R_j$.

(3) After the successful authentication on $R_j$, parameters, $S$ extracts $ID_T^x||W_1$, verifies $W_1^x = h((a||g)\oplus (X_1^j||ID_T^x)||TS_{LA1})$, retrieves the related $sn_i$ using $ID_T^x$, computes $X_{Ti}^x = h(sn_i||ID_T^x||x_i)$, and verifies $W_1^x = h(((a||g)\oplus (X_1^j||ID_T^x)||TS_{LA1})$ the authenticity of the user.

(4) It further generates $\gamma \in \mathcal{F}_\alpha^x$ and computes the session key $SK_{ST} = h(((ID_T^x||X_{Ti}^x)(a||g)\oplus x_{Ti}, a||g))||sn_i \oplus (TS_{LA1}||TS_{LA3})).$

But the conflict here is that the cloud server fails to compute the proper session key to pass it on to the tag for the validation. The reason is that the cloud server could not retrieve the random values generated by the tag and reader such as $\alpha, \beta \in \mathcal{F}_\alpha^x$, and in the session key the cloud server uses $(a||g)\oplus x_{Ti}, a||g$ value without the knowledge of the random numbers. Though the cloud server performs this computation, it would be certainly a garbage value which the tag cannot validate at any given point of time. Thus, this scheme holds the inefficiency to perform mutual authentication.

5.2. Inefficient Session Key Establishment Attack. On receiving the message $m_3$ from the cloud server, the tag performs the mutual authentication verification. But, the verification gets fails. The details are as follows:

(1) As discussed in the above Section 5.1, we understood that the cloud server fails to compute the authentic session key. However, on receiving the message from the cloud server, $T_i$ verifies the timestamp, that is, $|TS_{LA7} - TS_{LA1}| \leq t_{RS}x\Delta T$ and $|TS_{LA5} - TS_{LA1}| \leq t_{RS}x\Delta T$, and then computes $SK_{TS} = h((ID_T^x \oplus X_{Ti}^x)||\alpha, \beta, g)\oplus x_{Ti}, a||g))||sn_i \oplus (TS_{LA1}||TS_{LA3})$ and checks $W_1^x = h((ID_T^x||SK_{TS}$). If so, it sets $SK_{TS}$ as the session key.

(2) Now you can see that the tag $T_i$ did not retrieve or has the potential to draw out the value $(a||g)\oplus x_{Ti}, a||g$ but still perform the computation to validate the session key.

This validation never gets successful as it is a known fact that without the proper parameters and values the verification fails and the tag and the cloud server cannot establish the session key for the future communications. Thus, this scheme holds the inefficiency to perform session key establishment.

5.3. Denial-of-Service Attack. According to RSEAP2’s scheme, the legitimate participants tries to communicate to each other and get the services as and when required, but from the security flaw as shown above in Sections 5.1 and 5.2, we understood that the scheme fails to establish the session key and mutual authentication. This shows the enough conclusive evidence that the scheme fails to provide services to the participants thought the tag and readers are the legitimate participants in the system. Hence, this scheme is prone to the denial-of-service attack.

6. Our Proposed Scheme

This section presents the proposed secure authentication protocol and the program architecture which is divided into a tag, a reader, and a cloud server for parallel processing, with each component working independently. In this architecture as shown in Figure 2, the tag initiates the communication by computing the validating message and transmits the validating message with a virtual ID to the reader. Upon receiving the message, it challenges the reader to validate the message. Thus, the reader computes the validating message and transmits the validating message with the virtual ID to the cloud server for further process. Once the message is received by the cloud server, it validates the reader message thereby the cloud server authenticates the reader and tag. After the successful authentication, it computes the session key to establish the key. Further, at the next stage, the reader receives the Ack1 and Ack2 from the cloud server as an acknowledgment. Then the check happens in the next stage, where the tag receives Ack1 from the reader.
and simultaneously the reader checks the received Ack2. Finally, once the check is successful, the tag establishes the session key and end the process (see process flow diagram in Figure 2).

In this section, we present our proposed scheme. In the initialization phase, the server S chooses an elliptic curve $E(\mathbb{F}_q)$ over $\mathbb{F}_q$ and a generator $g$ over $G$. It also selects $x_i \in \mathbb{F}_q^*$ as its secret key and its public key will be $x_i \cdot g$. Any tag $T_i$ which aims to register with $S$, inputs its $ID_{T_i}$, $pw_{T_i}$, generates challenge $C_{T_i}$, computes $\alpha_{T_i} = \text{PUF}(C_{T_i})$, $pa_{T_i} = \alpha_{T_i} \cdot g$, and sends the tuple $M_{R1} = \{ID_{T_i}, \alpha_{T_i}, C_{T_i}\}$ to $S$. Once $S$ receives $M_{R1}$, verifies in the records whether $ID_{T_i}$ exists or not. If the $ID_{T_i}$ is new, it generates $s_{T_i} \in \mathbb{F}_q^*$, computes $\beta_{T_i} = h(s_{T_i} \cdot x_i, g)$, and stores the tuple $\{\beta_{T_i}, C_{T_i}, s_{T_i}, \alpha_{T_i}\}$ corresponding to $ID_{T_i}$. It then sends tuple $M_{R2} = \{\beta_{T_i}, \alpha_{T_i}\}$ to $T_i$. The tag $T_i$ computes $\beta_{T_i} = h(s_{T_i} \cdot x_i, g)$, and stores $\{\beta_{T_i}, \alpha_{T_i}\}$. Similarly, reader $R_j$ aims to register with $S$, generates challenge $C_{R_j}$, computes $\alpha_{R_j} = \text{PUF}(C_{R_j})$, and sends $M_{R2} = \{ID_{R_j}, \alpha_{R_j}\}$ to the cloud database server $S$. $S$ computes $\beta_{R_j}$ by its private key and $C_{R_j} = pa_{R_j} \cdot ID_{R_j}$, $\alpha_{R_j} = \text{PUF}(C_{R_j})$, $\beta_{R_j} = h((\alpha_{T_i} \cdot ID_{T_i})||\beta_{T_i})$; sends $M_{R2} = \{\beta_{R_j}, \alpha_{R_j}\}$ and stores $\{\beta_{R_j}, \alpha_{R_j}\}$. The illustration of the tag registration and reader registration is shown in Table 5 and Table 6, respectively.

6.1. Login and Authentication Phase. To access the services from $S$, $T_i$ needs to establish a session key with $S$. The following steps are accomplished by $T_i$, $R_j$, and $S$ during this phase. The illustration is shown in Table 7.

LA1: The tag logs on by ($ID_{T_i}$, $pw_{T_i}$), computes $\alpha_{T_i} = pa_{T_i} \cdot g$, $\beta_{T_i} = h(ID_{T_i}||pw_{T_i})$, verifies $\beta_{T_i} = h(ID_{T_i}||pw_{T_i})$, and generates challenge $C_{T_i}$ to compute $W_1 = h((a \cdot g) \cdot \alpha_{T_i})\beta_{T_i}||ID_{T_i}||TS_{L_A1})$, and sends $M_1 = \{ID_{T_i}, \alpha_{T_i}, \beta_{T_i}, W_1\}$ to $R_j$.

LA2: On receiving the request, $R_j$ verifies $TS_{L_A1}$ and $TS_{L_A3}$ extracts $M_1, TS_{L_A1}, W_2||\beta_{T_i}$; and sends $W_2 = h(a \cdot g)\beta_{T_i}||ID_{T_i}||TS_{L_A3}$ to $S$.

LA3: On receiving the request, $S$ verifies $TS_{L_A1}$ and $TS_{L_A3}$; extracts $M_1, TS_{L_A1}, W_2||\beta_{T_i}$; validates $W_2 = h(a \cdot g)\beta_{T_i}||ID_{T_i}||TS_{L_A3}$; and extracts $\{\beta_{T_i}, \alpha_{T_i}\}$ to verify $W_2$$= h((a \cdot g)\alpha_{T_i})\beta_{T_i}||ID_{T_i}||TS_{L_A3}$ and on success, generates $\beta_{T_i} \in \mathbb{F}_q^*$ computes $S_{K_{ST}} = h((ID_{T_i}||\beta_{T_i}||TS_{L_A1}||\alpha_{T_i}||ID_{T_i}||TS_{L_A3}))$, $W_3 = h(a \cdot g)S_{K_{ST}}||\alpha_{T_i}||W_2||\beta_{T_i}$, and sends $M_3 = \{W_3\}$ to $T_i$.

LA4: After receiving the response $S, R_j$ checks $TS_{L_A3}$, verifies $W_3 = h((ID_{R_j}||\alpha_{R_j}||TS_{L_A3}||\beta_{R_j}))$, and sends $M_4 = \{W_3, \beta_{R_j}, TS_{L_A3}\}$ to $T_i$.

LA5: On receiving the response from $R_j$, $T_i$ verifies $TS_{L_A3}$, computes $S_{K_{ST}} = h((ID_{T_i}||\beta_{T_i}||TS_{L_A1}||\alpha_{T_i}||ID_{T_i}||TS_{L_A3}))$, and checks $W_3 = h(a \cdot g)S_{K_{ST}}||\alpha_{T_i}||\beta_{T_i}$. On successful verification, $T_i$ sets $S_{K_{TS}}$ as the session key.

6.2. Revocation and Reissue Phase. To revoke the access of $T_i$, $S$ checks for the availability of $ID_{T_i}$ during the subsequent login attempts. The tag will be given or refused access on the basis of the check. Since all dynamic identities have a finite lifetime, it is also impossible to continuously use the same dynamic identity.

In addition, the next steps to get new credentials are crucial when a tag $T_i$ from an approved registered user is stolen/lost.

RR1: The tag keeps the same $ID_{T_i}$, but chooses a password $pw_{T_i}$ and generates challenge $C_{R_j}$ to compute $\alpha_{R_j} = \text{PUF}(C_{R_j})$, $\beta_{R_j} = \text{PUF}(ID_{T_i}||pw_{T_i})$. Further submitting the revocation request $M_{R1} = \{ID_{T_i}, \beta_{R_j}, C_{R_j}\}$ to the cloud database server $S$ through secure channel.

RR2: On receiving the request, $S$ checks the database for the availability of $\alpha_{T_i} = h((a \cdot g)\cdot ID_{T_i})||\beta_{T_i}$, where $\beta_{T_i}$ is computed by private key of $S$. If $\alpha_{T_i}$ is not available, the cloud database server computes and sends the tuple $M_{R2} = \{\alpha_{T_i}, C_{T_i}\}$ to $T_i$ over the secure channel.

RR3: Finally, for each tag, the cloud server $S$ issues the new credentials.

RR4: After receiving the new credentials, $T_i$ completes the registration process as processed in the registration phase.

6.3. Tag’s Password/Update Phase. A registered tag $T_i$ can update his/her current password and follow the steps without contacting $S$.

PU1: The tag logs on by ($ID_{T_i}$, $pw_{T_i}$), computes $\alpha_{T_i} = pa_{T_i}\cdot h(ID_{T_i}||pw_{T_i})$, and verifies $W_1 = h((a \cdot g)\cdot ID_{T_i})||\beta_{T_i})$. On unsuccessful verification, this process gets terminated by $T_i$. Otherwise, $T_i$ uses new password.

PU2: $T_i$ picks $pw_{new}$, computes $\alpha_{T_i} = \text{PUF}(C_{R_j})$, $\beta_{T_i} = h((a \cdot g)\cdot ID_{T_i}||pw_{new})$, and $W_1 = h((a \cdot g)\cdot ID_{T_i}||pw_{new})$; and stores $\{W_2, \beta_{T_i}, TS_{L_A3}\}$ to complete the process.

7. Formal Security Analysis

Formal security examination strategies are usually used to inspect and evaluate diverse check plans. According to literature [25], various security assessment systems can be used...
**Table 5: Tag registration phases of our scheme.**

| Tag $T_i$ | Cloud database server $S$ |
|-----------|-----------------------------|
| **Inputs** $(ID_{T_i}, pw_{T_i})$ | **Verifies** $ID_{T_i}$ | **Computes** $p_id_{T_i}$ by private key of $S$ |
| Generates challenge $C_{T_i}$ |  | $A_{T_i} = h((\alpha_{T_i} \oplus ID_{T_i})||p_id_{T_i})$ |
| Computes $\alpha_{T_i} = PUF(C_{T_i})$ |  | $\Rightarrow M_{\alpha_{T_i}} = [p_id_{T_i}, A_{T_i}]$ SecureChannel |
| $p_{\alpha_{T_i}} = \alpha_{T_i} \oplus h(ID_{T_i}||pw_{T_i})$ |  | Stores $\{p_id_{T_i}, ID_{T_i}, C_{T_i}, \alpha_{T_i}\}$ |
| $\Rightarrow M_{\alpha_{T_i}} = [ID_{T_i}, \alpha_{T_i}, C_{T_i}]$ SecureChannel |  | **Transmit Ack1 and Ack2 to Reader** |
| PWT = $h(ID_{T_i}||pw_{T_i}||\alpha_{T_i})||p_{\alpha_{T_i}})$, deletes $(ID_{T_i}, C_{T_i}, \alpha_{T_i})$ and stores $\{\text{PWT}, p_id_{T_i}, p_{\alpha_{T_i}}, A_{T_i}\}$ |  | **Update database** |

**Table 6: Reader registration phases of our scheme.**

| Reader $R_j$ | Cloud database server $S$ |
|-------------|-----------------------------|
| Generates challenge $C_{R_j}$ | **Computes** $p_id_{R_j}$ by private key of $S$ |
| Computes $\alpha_{R_j} = PUF(C_{R_j})$ | $C_{R_j} = p_{\alpha_{R_j}} \oplus ID_{R_j}$ |
| $p_{\alpha_{R_j}} = C_{R_j} \oplus ID_{R_j}$ | $\alpha_{R_j} = PUF(C_{R_j})$ |
| $\Rightarrow M_{\alpha_{R_j}} = [p_id_{R_j}, A_{R_j}]$ SecureChannel | $A_{R_j} = h((\alpha_{R_j} \oplus ID_{R_j})||p_id_{R_j})$ |
| **Deletes** $(ID_{R_j}, C_{R_j}, \alpha_{R_j})$ | $\Rightarrow M_{\alpha_{R_j}} = [p_id_{R_j}, A_{R_j}]$ SecureChannel |
| Stores $\{p_id_{R_j}, p_{\alpha_{R_j}}, A_{R_j}\}$ | Stores $\{p_id_{R_j}, ID_{R_j}, C_{R_j}, \alpha_{R_j}\}$ |
Table 7: Login and authentication phases of our scheme.

| TAGTj       | Reader Rj                              | Cloud database server S |
|-------------|----------------------------------------|-------------------------|
| Logs on by (IDTj, pwTj) | Computes a<j>′<sub>T</sub> = paRj/bh(IDTj,pwTj) | Verifies TS<sub>LA1</sub>, compute C<sub>Rj</sub> = paRj/bIDRj | Verifies TS<sub>LA1</sub> and TS<sub>LA3</sub> Exacts M<sub>1</sub>, TS<sub>LA1</sub>, (W<sub>2</sub>|pid<sub>Rj</sub>) | Verifies W<sub>2</sub> = h(aRj|ARj||TS<sub>LA1</sub>)               |
| Computes a<j>′<sub>T</sub> = paRj/bh(IDTj,pwTj) | Generates α ∈ F′<sup>+</sup>                | Computes a<sub>Rj</sub> = PUF(C<sub>Rj</sub>)          | Verifies W<sub>3</sub> = h((α.g.a<sub>Tj</sub>)|A<sub>Tj</sub>|IDTj)||TS<sub>LA1</sub>) | Verifies W<sub>4</sub> = h(IDRj|ARj||TS<sub>LA3</sub>||pid<sub>Rj</sub>) |
| Generates α ∈ F′<sup>+</sup>                | Computes W<sub>1</sub> = h((α.g.a<sub>Tj</sub>)|A<sub>Tj</sub>|IDTj)||TS<sub>LA1</sub>) | Verifies C<sub>Rj</sub> = paRj/bIDRj | Verifies W<sub>1</sub> = h((α.g.a<sub>Tj</sub>)|A<sub>Tj</sub>|IDTj)||TS<sub>LA1</sub>) | M<sub>1</sub> = (M<sub>1</sub>, TS<sub>LA1</sub>, W<sub>1</sub>|pid<sub>Rj</sub>) |
| Verifies W<sub>1</sub> = h((α.g.a<sub>Tj</sub>)|A<sub>Tj</sub>|IDTj)||TS<sub>LA1</sub>) | Checks TS<sub>LA3</sub> and computes SK<sub>TS</sub> = h(IDRj|ARj||TS<sub>LA3</sub>||pid<sub>Rj</sub>) | Checks W<sub>3</sub> = h((α.g.a<sub>Tj</sub>)|A<sub>Tj</sub>|IDTj)||TS<sub>LA1</sub>) | Checks W<sub>3</sub> = h((α.g.a<sub>Tj</sub>)|A<sub>Tj</sub>|IDTj)||TS<sub>LA1</sub>) | Checks W<sub>3</sub> = h((α.g.a<sub>Tj</sub>)|A<sub>Tj</sub>|IDTj)||TS<sub>LA1</sub>) |
| Checks W<sub>3</sub> = h((α.g.a<sub>Tj</sub>)|A<sub>Tj</sub>|IDTj)||TS<sub>LA1</sub>) | to set SK<sub>TS</sub> as the session key | Checks W<sub>4</sub> = h(IDRj|ARj||TS<sub>LA3</sub>||pid<sub>Rj</sub>) | Checks W<sub>4</sub> = h(IDRj|ARj||TS<sub>LA3</sub>||pid<sub>Rj</sub>) | Checks W<sub>4</sub> = h(IDRj|ARj||TS<sub>LA3</sub>||pid<sub>Rj</sub>) |

7.1. ROR Model-Based Proof. Under this model, adversaries say that an attacker has access to a set of executing entity queries including CorruptTi (T<sub>i</sub>), Test (P<sub>i</sub>), Execute (T<sub>i</sub>, S<sub>i</sub>), and Reveal (P<sub>i</sub>), which perform simulation to check the real attack. The query descriptions of such queries are given in Table 8. The ROR model components are as follows:

(i) Participants: The associated participants with the proposed scheme are the tag Ti, reader Rj, or a cloud server S<sub>i</sub>. The instances t<sub>i</sub> and t<sub>j</sub> of T<sub>i</sub> and S<sub>i</sub> are marked as P<sub>i</sub>T<sub>i</sub> and P<sub>j</sub>S<sub>i</sub>, which are known as oracles.

(ii) Accepted state: If the peer points achieve an accepted status when the final communication has been authenticated, the instance “accepted status” comes under “accepted state.” For the ongoing session, sid is a P<sub>i</sub> session ID created in a sequence by PPt after the sent and received messages were rearranged.

(iii) Partnering: The following things must be accomplished to be partnered between P<sub>i</sub>T<sub>i</sub> and P<sub>j</sub>S<sub>i</sub>:

1. They are in “accepted states.”
2. They possess the same sid. Further also “authenticate mutually with each other.”
3. They are also “mutual partners of each other.”

(iv) Freshness: P<sub>i</sub>T<sub>i</sub> or P<sub>j</sub>S<sub>i</sub> is fresh when the constructed session key between T<sub>i</sub> and S<sub>i</sub> is not leaked to <i>A</i> using the Reveal (P<sub>i</sub>) query listed in Table 8.

The proposed scheme undergoes “semantic security” as defined in Definition 1.

Definition 1. If Adv<sub>A</sub> <sub>fid</sub>−PUF (t<sub>p</sub>) is the “advantage of an adversary A running in polynomial time t<sub>p</sub> in breaching the semantic security of <i>R</i>fid − PUF to extract the session key (SK<sub>TS</sub>) among a tag T<sub>i</sub> and a cloud server S<sub>i</sub>)

Adv<sub>A</sub> <sub>fid</sub>−PUF (t<sub>p</sub>) = |Pr[c = c′] − 1|, where c and c′ indicate the guessed bits.

Furthermore, Definition 2 is about “collision-resistant one-way hash function” and Definition 3 is about “elliptic curve decisional Diffie–Hellman problem (ECDDDHP),” for brevity <i>R</i>fid − PUF.

Definition 2. A “deterministic function,” say h: {0, 1}∗ → {0, 1}∗, is a “one-way collision-resistant hash function” if it produces fixed length of <i>L</i> output string h(m) ∈ {0, 1}∗ as “hash value or message digest” upon an arbitrary length input string m ∈ {0, 1}∗. Let an adversary <i>A</i> want to find a hash collision. Then, the “advantage” of <i>A</i> in attacking “hash collision” is provided by Adv<sub>A</sub> <sub>Hash</sub> (t<sub>p</sub>) = Pr[(m<sub>1</sub>, m<sub>2</sub>) ←<i>A</i>: m<sub>1</sub> ≠ m<sub>2</sub>, h(m<sub>1</sub>) = h(m<sub>2</sub>)]. Pr(X) here shows the chance that the pair will be randomly picked by <i>A</i> in the case of “random event X” and
where \( \alpha \) can extract the stored credentials by compromised tag \( T_j \)'s memory.

**Definition 3.** Consider an elliptic curve \( \langle P, G \rangle \), the ECDDHP is "for a quadruple \( \langle P, u, v, P \rangle \), decide whether \( u \cdot v = \text{Hash}(P) \) or it is a uniform value," where \( u = \text{Hash}(u) \cdot \text{Hash}(v) = ((u_{\frac{1}{2}}, 2), \ldots, (u_{-1}, 1)) \).

To make ECDDHP intracable, the chosen prime number must be at least 160-bit number.

**Theorem 1.** Suppose our scheme \( \text{Rfid} \) works in "polynomial time \( t_p \)" and the adversary is working to gain advantage on \( \text{Rfid} \) - PUF. If query \( \text{Rfid} \) \( \text{Hash} \) and \( \text{Adv}_P \text{ECDDHP} \) \( t_p \) indicate the "cardinality of hash queries," "size of one-way hash function \( h(x) \)," and "\( \alpha \)'s advantage in breaching ECDDHP in time \( t_p \)" (see Definition III-A), respectively, and chosen passwords follow the Zipf's law [26], then the bit-lengths of the PUF key \( \text{PUF}(C) \) where \( \ast \) refers to \( T_j \) / \( T_j \) and the tag identity \( ID_j \), are \( l_1 \) and \( l_2 \), respectively, \( y \) and \( s_i \) are the Zipf's parameters [26] respectively, \( s_i \)'s advantage in compromising the semantic security of the proposed scheme \( \text{Rfid} - \text{PUF} \) is \( \text{Adv}_P \text{ECDDHP} \) \( t_p \) \( \leq 2^{\eta} \text{Adv}_P \text{ECDDHP} \) \( t_p \) \( + \text{max}\{\text{query}_1/2^{l_1} \}, \text{query}_2/2^{l_2} \}, \text{query}_\frac{1}{2^1} \cdot \text{query}_\frac{1}{2^1} \}. \)

**Proof.** This proof is presented in the similar way as presented by authentication protocols. Here four games are played, such as \( G_k \), \( k = 0, 1, 2, 3 \) related to the evidence where \( G_0 \) is the starting and \( G_3 \) is the finishing game. We define Succ\( G_k \) as "an event wherein \( \alpha \) can guess the random bit \( c \) in the game \( G_k \) correctly" and also the "advantage of \( \alpha \) in winning the game \( G_k \) as \( \text{Adv}_P \text{ECDDHP} \) \( t_p \). The detailed study of these games is as follows:

- \( G_0 \): \( G_0 \) is the same as the real ROR model protocol. Therefore, the semantic security of \( \text{Rfid} - \text{PUF} \) is defined in Definition 1.

\[
\text{Adv}_P \text{ECDDHP} \ (t_p) = 2 \cdot \text{Adv}_P \text{ECDDHP} - 1
\]

- \( G_1 \): In this game, we model the "eavesdropping attack" in which \( \alpha \) can intercept all the communication messages \( M_1 = \{ \text{pid} \ast A_T, \text{PID} \ast W_1 \}, \text{MA}, \text{g}, \text{g}, \text{g}, \text{TS}_{L_1} \}, M_2 = \{ \text{M}, \text{TS}_{L_1} \}, M_3 = \{ \text{W}_1, \text{W}_2, \text{TS}_{L_1}, \text{g}, \text{g}, \text{TS}_{L_1} \}, M_4 = \{ \text{W}_2, \text{g}, \text{g}, \text{TS}_{L_1} \} \) while executing "authentication and key agreement phase" in Section A using Execute query as discussed in Table 8. To confirm whether the "calculated session key \( \text{SK}_{TS} \) between \( T_i \) and \( S \) is real or a random number," \( \alpha \) can execute both Reveal and Test queries. The established session key is \( \text{SK}_{TS} = h((ID_j \ast A_T), (a, g), (x, g)) \) if \( \text{TS}_{L_1} \ast \text{TS}_{L_1} \ast \text{TS}_{L_1} \), \( \text{SK}_{TS} \) shared between \( T_i \) and \( S \), \( \alpha \) needs to calculate \( h(a, b, g, ID_j), (A_T, ID_j) \) which in a polynomially restricted time \( t_p \) is computationally costly owing to the intractability of ECDDHP. Since \( G_2 \) and \( G_3 \) games are "indistinguishable," the following is excepted to include the question and ECDDHP of \( \text{CorruptTi} \):
\[
\begin{align*}
|\text{Adv}_{\text{Adv}}^{\text{Rfid-PUF}} - \text{Adv}_{\text{Adv}}^{\text{Rfid-PUF}}| & \leq \text{Adv}_{\text{Adv}}^{\text{ECDDHP}}(t_p) \\
+ \max \left\{ \frac{\text{query}_1}{2^h}, \frac{\text{query}_2}{2^h}, \beta \cdot \text{query}_3^{\beta} \right\}. 
\end{align*}
\]

Now, all the relevant queries related to the above games are executed, and then the Reveal query is executed along with Test query to guess the random bit \( c \). Thus, we get
\[
\text{Adv}_{\text{Adv}}^{\text{Rfid-PUF}} = \frac{1}{2}
\]

Combining equations (1), (2), and (5) derives:
\[
\begin{align*}
\frac{1}{2} \cdot \text{Adv}_{\text{Adv}}^{\text{Rfid-PUF}}(t_p) &= \left| \text{Adv}_{\text{Adv}}^{\text{Rfid-PUF}} - \frac{1}{2} \right| \\
&= \left| \text{Adv}_{\text{Adv}}^{\text{Rfid-PUF}} - \text{Adv}_{\text{Adv}}^{\text{Rfid-PUF}} \right| \\
&\leq \left| \text{Adv}_{\text{Adv}}^{\text{Rfid-PUF}} - \text{Adv}_{\text{Adv}}^{\text{Rfid-PUF}} \right| \\
+ \left| \text{Adv}_{\text{Adv}}^{\text{Rfid-PUF}} - \text{Adv}_{\text{Adv}}^{\text{Rfid-PUF}} \right|.
\end{align*}
\]

Next, combining equations (3), (4), and (6) provide the following result:
\[
\begin{align*}
\frac{1}{2} \cdot \text{Adv}_{\text{Adv}}^{\text{Rfid-PUF}}(t_p) &\leq \frac{\text{query}_1^2}{2^{h}|\text{Hash}|} + \text{Adv}_{\text{Adv}}^{\text{ECDDHP}}(t_p) \\
+ \max \left\{ \frac{\text{query}_1}{2^h}, \frac{\text{query}_2}{2^h}, \beta \cdot \text{query}_3^{\beta} \right\}.
\end{align*}
\]

Finally, the equation (7) is multiplied by 2 on both sides to get
\[
\begin{align*}
\text{Adv}_{\text{Adv}}^{\text{Rfid-PUF}}(t_p) &\leq 2 \text{Adv}_{\text{Adv}}^{\text{ECDDHP}}(t_p) + \frac{\text{query}_1^2}{|\text{Hash}|} \\
+ 2 \max \left\{ \frac{\text{query}_1}{2^h}, \frac{\text{query}_2}{2^h}, \beta \cdot \text{query}_3^{\beta} \right\}.
\end{align*}
\]

7.2. Informal Security Analysis

**Proposition 1.** Location privacy (non-traceability)

**Proof.** The tag \( T_i \) simply transmits the message
\[
M_1 = \{ (\text{pid}_{T_i} || A_{T_i} || W_1) \oplus \alpha \cdot x \cdot g, \alpha \cdot g, TS_{LA1} \},
\]
with
\[
W_1 = h(\alpha \cdot g) \oplus (A_{T_i} || ID_{T_i} || TS_{LA1}).
\]
Only \( (\text{pid}_{T_i} || A_{T_i} || W_1) \) of this message can be utilized to identify the tag. On each session, the variables alpha and \( TS_{LA1} \) masked and randomized the token described above. The attacker has no control over any of these values. If a collision happens on the specified value by \( T_i \) in the worst-case scenario, the adversary could detect it by monitoring the alpha \( g \) fraction of \( M_1 \), and then \( T_i \) could be monitored. However, the adversary’s advantage in finding a collision after \( N \) protocol sessions is \( O(N^2/|F^2|) \), which is modest enough in practice. Furthermore, \( M_1 \) makes no mention of \( R_j \) or \( S \).

The reader \( R_j \) delivers \( M_2 = \{ M_1, TS_{LA2}, (W_2 || \text{pid}_{R_j}) \} \) to \( S \), where \( (W_2 || \text{pid}_{R_j}) \) may be used to monitor the reader and determine whether the \( W_2 \) fraction has a collision. Similarly, after \( N \) protocol executions, the adversary has an advantage of \( O(N^2/|F^2|) \) in detecting a collision. As a result, the opponent’s chances of success are slim.

Finally, \( R_i \) sends \( M_4 = \{ W_3, \beta \cdot g, TS_{LA3} \} \) to \( T_i \). The adversary’s only target in this communication could be \( W_3 \). This token is a function of \( SK_{TS} = h(\alpha \cdot g || x \cdot \alpha \cdot g || (\text{pid}_{T_j} || A_{T_j} || TS_{LA3} || TS_{LA5})) \), which is randomized by \( T_j, R_j, S \) on each session. Overall, the location privacy of all of our entities (i.e., \( T_i, R_j, S \)) is guaranteed by our protocol. □

**Proposition 2.** Mutual authentication and session key agreement

**Proof.** It is obvious that the pairs \((S, T_i)\) and \((S, R_j)\) are mutually authenticated if a legitimate tag \( T_i \) connects with an honest server \( S \) through a valid reader \( R_j \), and within acceptable time thresholds. However, we do not require mutual authentication between the reader \( R_j \) and the tag \( T_i \) in this protocol. In more detail, \( S \) is the source of trust for \( T_i \), while \( R_j \) is only a gateway to \( S \). The following is a list of the session key’s correctness and mutual agreement:

**Correction Proof:**

\[
\begin{align*}
W_3 &= h(\beta \cdot (\alpha \cdot g)) \oplus SK_{ST} \oplus A_{T_i} \oplus \text{pid}_{T_i} \\
&= h(\beta \cdot (\alpha \cdot g)) \oplus (h(1D_{T_i} \oplus A_{T_i})) \oplus (\alpha \cdot g) \oplus (\text{pid}_{T_i}) \\
&= h(\alpha \cdot g) \oplus (h(1D_{T_i} \oplus A_{T_i})) \oplus (\alpha \cdot g) \oplus (\text{pid}_{T_i}) \\
&= h(\alpha \cdot g) \oplus (h(1D_{T_i} \oplus A_{T_i})) \oplus (\alpha \cdot g) \oplus (\text{pid}_{T_i}) \\
&= h(\alpha \cdot g) \oplus SK_{TS} \oplus A_{T_i} \oplus \text{pid}_{T_i} = W_3'.
\end{align*}
\]
Because the tag and the server have mutual authentication, $S$ has already authenticated $R_j$ and $T_i$ may trust the reader $R_j$. As a result, our technique ensures mutual authentication and establishes suitable session key agreement.

**Proposition 3. Physical security**

*Proof.* Any alteration or damage to the device with built-in PUF will cause PUF to respond differently or the device to become unavailable, according to PUF’s characteristics. It is impossible to collect any relevant information in an accessible environment since car sensors do not preserve any information. Physical attacks, aside from rendering the system’s physical security, are unable to extract any relevant information. As a result, the suggested protocol can ensure the system’s physical security.

**Proposition 4. Achieving forward secrecy**

*Proof.* In our proposed scheme, the session key is computed as $SK_{TS} = h((ID_{TS} \oplus A_{TS}))(a, b, g)(x, a, g)$ if $(SN, TA_{LS})$. This session key is established between the tag $T_i$ and the server $S$. If $\mathcal{A}$ wishes to compromise the session key, $\mathcal{A}$ requires the knowledge of the session-specific random values $\{a, b\}$, fixed value $\omega_{TS}$, and the identities of the participants involved in the session key establishment. Now, even if $pw_{TS}, pa_{TS}$ are compromised by $\mathcal{A}$, due to the lack of knowledge of $CT_{TS}$ or random values $\{a, b\}$ and fixed value $\omega_{TS}$, attacker fails to compute $W_i$. Thus, $\mathcal{A}$ does not gain any advantage even if he compromises $pw_{TS}, pa_{TS}$. Therefore, $\mathcal{A}$ cannot compute the previous/current/future session keys.

**Proposition 5. Message authentication**

*Proof.* In this protocol, the server authenticates $M_1$ and $M_2$. The reader $R_j$ authenticates $S$, $M_1$, partially and the tag $T_i$ totally. The use of random integers and the one-way hash function ensure the integrity of all messages. Any alteration to the conveyed message causes the receiver to reject the message.

For instance, consider $M_1 = \{(pid_{TS} \oplus A_{TS} \oplus W_1 \oplus \alpha, x, g, a, g)\}$. $x$ is a random string value, $M_2 = h((\alpha, g) \oplus \omega_{TS})$. Then, $T_i$ retrieves the related $sn_i$ value using $\omega_{TS}$ and $\omega_{TS}$ and computes and verifies $\alpha = h((\alpha, g) \oplus \omega_{TS})$. $\alpha$ is a random string value, $W_i = h((\alpha, g) \oplus \omega_{TS})$. $\alpha$ is a random string value, $W_i = h((\alpha, g) \oplus \omega_{TS})$. Therefore, $\mathcal{A}$ cannot compute the previous/current/future session keys.

**Proposition 6. Replay attack**

*Proof.* In a replay attack, the adversary attempts to use a previously traded message at a later time $t'$. Any message received outside of the threshold time ($t' + \Delta T$) is likely to be rejected in our protocol. Aside from that, the one-way hash function ensures the integrity of timestamps. As a result, replay attacks against our protocol are impossible. Finally, the adversary may break the tag’s anonymity if he extracted $xs.g$ from the $\alpha.xs.g$ and $a.g$ pair. It is most likely the same as solving ECCDHP, which is known to be a difficult task (see Section 3.1).

**Proposition 7. Impersonation attack**

*Proof.* Due to the integrity of $TS_{LSA}$, the only way to spoof the tag is to construct a valid $M_1$. It is not possible, however, without guessing or computing a valid $V_i = h((\alpha, g) \oplus \omega_{TS})$. $V_i$ is the attack time’s timestamp. The adversary also lacks $A_{TS}$ and $ID_{TS}$. As a result, the adversary’s chance of successfully impersonating the tag is $2^{-n}$, where $n$ is the hash function’s bit-length. To put it another way, the repeat attack is a waste of time.

**Proposition 8. Impersonation attack**

*Proof.* Because the integrity of $TS_{LSA}$ is guaranteed in our protocol, the adversary cannot replay messages to impersonate a reader. As a result, generating a legitimate message $\alpha, \beta, g$ is the only way to impersonate the $R_j$ in front of $S$. The opponent, on the other hand, lacks $V_i = h((\alpha, g) \oplus \omega_{TS})$. $V_i$ is the attack time’s timestamp. The adversary also lacks $A_{TS}$ and $ID_{TS}$. Even if she/he obtains the values $V_i$, $\alpha, \beta, g$, and $A_{RS}$ in some other way, she/he must extract $\alpha_{RS}$ from $M_2$ in order to determine $ID_{R_j}$. It necessitates reverse engineering of the one-way hash function, which is a difficult challenge that makes the assault impracticable. As a result, impersonating $R_j$ to $S$ is not feasible under this protocol.
advantage in committing this impersonation attack is negligible (i.e., $2^{-w}$).

**Proposition 8. Offline password guessing attack**

**Proof.** The rationale for security against this attack is nearly comparable to that of RSEAP2. In a nutshell, PWT $= h(ID_{Ti}, \| (pw_{Ti} \| a_{Ti}) \| pa_{Ti})$ calculates the tag’s temporary password. Even if the adversary could estimate PWT, the value $\alpha_{Ti}$, which is a random integer created by the tag $T_i$, is still required. As a result, the opponent who could not foresee $\alpha_{Ti}$ will be defeated by this assault.

**Proposition 9. Desynchronization attack**

**Proof.** Because there is no updating phase of shared parameters after the protocol execution concludes, our proposed technique is immune to desynchronization assaults. The attacker may only block the $M_i$ message if the tag $T_i$ is used to set the session key $SK_{TS}/SK_{ST}$. Because $T_i$ has not received $M_i$ in a timely manner, this entity may need to restart the login and authentication step in order to reestablish the session key. We wish to underline that the aforementioned situation is distinct from an impersonation assault—as previously stated, an adversary cannot impersonate a valid tag. In addition, the tag $T_i$ must start the protocol; otherwise, the server $S$ would reject the request.

**Proposition 10. Insider attack**

**Proof.** In the initialization phase of our scheme, $T_i$ sends $M_{R1} = \{ID_{Ti}, a_{Ti}, C_{Ti}\}$ to $S$ and receives $M_{R2} = \{pid_{Ti}, A_{Ti}\}$ in return. Further computes, where PWT $= h(ID_{Ti}, \| (pw_{Ti} \| a_{Ti}) \| pa_{Ti})$. Likely, the chances for an insider attacker to disclose $pw_{Ti}$ are almost null (i.e., $2^{-w}$).

**Proposition 11. Man-in-the-middle attack**

**Proof.** To carry out a successful man-in-the-middle attack, an adversary must be able to impersonate a protocol entity or modify a message without being discovered. Nonetheless, the aforementioned attack will fail in our suggested protocol for the following reasons. For starters, as we explained in Section 7, the adversary’s advantage in impersonating the tag, the reader, or the server is insignificant. Second, we have shown (5) that any change to the transmitted message causes the receiver to reject the received message. Finally, we demonstrated how an opponent cannot properly relay a message to deceive about his distance or replay an earlier message in Sections 6. As a result, the suggested protocol is impenetrable to a man-in-the-middle assault.

**Proposition 12. Ephemeral secret leakage (ESL) attack**

**Proof.** As described in the Proposition 2, both $T_i$ and $S$ establish a common session key during the execution of the proposed scheme. The session key is computed as $SK_{TS} = h((ID_{Ti} \oplus A_{Ti} || (a, g, x, a, g)) \| (sn_{Ti} \oplus TS_{ST} || TS_{LAS}))$. The SK-security of the proposed scheme relies on the secret credentials as discussed in the following two cases:

Case 1. Let us consider if knows the ephemeral (short-term) secret credentials $\alpha$ and $\beta$. It is computationally infeasible for $A$ to create the valid session key $SK_{TS}$ without the knowledge of the long-term secrets $AR_j$, $A_{Ti}$, $a_{Rj}$, and $x_i$. Therefore, both forward as well as backward secrecy along with the SK-security are preserved in the proposed scheme. Moreover, in the proposed scheme, with the help of the session hijacking attack, a session key is leaked in a particular session; it has no affect to compromise the security of other previous as well as future sessions. By summing up all these cases, the proposed scheme is secure against the ESL attack.

**8. Observations and Performance Analysis**

We use the implementation results in [2] “(CPU: Intel(R) Core(TM)2T6570 2.1 GHz, Memory: 4G, OS: Win7 32-bit, Software: Visual C++ 2008, MIRA CL C/C++ Library)” to estimate the computation time. Because SHA-2 occupies 15.8 cycles per bytes [27], it takes $T_{fun}^{SHA} = 0.0004 \times (15.8/11.4) = 0.0005$ milliseconds to compute. To be clear, the number $T_{fun}^{SHA}$ corresponds to a single call to the SHA-2 compression function (fun). The SHA-2 compression function has a message-block length of 512 bits. We built the new protocol in detail to reduce the amount of calls to this compression function, particularly on the tag side, which is the most limited device. Finally, the time required to calculate scalar multiplication on ECC-160, represented by $T_{EMP}$, is 7.3529 milliseconds, whereas the time required to calculate a chaotic map is $T^{CH} = T_{EMP}$ [28].

The needed time for encryption/decryption of a symmetric scheme $T^{Sym}$ varies depending on the employed symmetric encryption method; however, the stated time for AES is $T^{Sym} = 0.1303$ milliseconds. The details are shown in Table 9.

The hash function output, nonces, timestamps, tag/reader identities, a symmetric encryption output block, and elliptic curve points all have bit widths of 160, 160, 32, 160, 128, and 320 bits, respectively, for the performance analysis. We compare the computational and communication expenses of RSEAP2 with our method in Table 10. Because tags are the most limited
In the system, we focus on the devices.

There are no major changes in consuming time when compared to RSEAP2, as shown in Figure 3, simply a minor improvement in our approach. Our scheme is much more efficient than RSEAP2 in terms of bits sent (and received), as shown in Figure 4. It entails a significant reduction in power consumption, which is a critical metric in such devices. Finally, in Table 11, we compare and contrast the security qualities afforded by comparable systems with our scheme (see Figure 5 for an instance). To summarize, the new protocol is more efficient and secure than the old one.

### Table 9: Approximate time required for various operations.

| Notation | Description | Approximate computation time (in milliseconds) |
|----------|-------------|-----------------------------------------------|
| $T_h^{fun}$ | Hash function | 0.0005 |
| $T_{EMP}$ | ECC point multiplication | 7.3529 |
| $T_{SymEnc/Dec}$ | Symmetric encryption/decryption | 0.1303 |
| $T_{QR} \approx T_{EMP}$ | QR code | 7.3529 |
| $T_{MAC} \approx T_h^{fun}$ | Chaotic map | 7.3529 |
| $T_{fun}$ | Message authentication code | 0.0005 |

### Table 10: Comparison of communication costs.

| Scheme | Communication cost (sending mode) | Communication cost (receiving mode) | Computation cost (in milliseconds) | Time (ms) |
|--------|------------------------------------|------------------------------------|----------------------------------|-----------|
| Jiang et al. [3] | 768 | 768 | $9T_h^{fun} + 5T_{Sym} + 3T_{EMP}$ | 37.4205 |
| Kumar et al. [7] | 832 | 544 | $9T_h^{fun} + 3T_{EMP}$ | 22.0632 |
| Mishra et al. [11] | 672 | 224 | $4T_h^{fun} + 2T_{CH}$ | 14.7078 |
| Jiang et al. [4] | 1280 | 800 | $9T_h^{fun} + T_{Sym} + 5T_{EMP}$ | 22.1935 |
| Safkhani et al. [2] | 672 | 512 | $6T_h^{fun} + 3T_{EMP}$ | 22.0617 |
| Our scheme | 672 | 512 | $6T_h^{fun} + 2T_{EMP}$ | 14.7088 |

![Figure 3: Computation cost comparison.](image3.png)

![Figure 4: Communication cost comparison.](image4.png)
9. Concluding Remarks

In this article, we designed a PUF and RFID-based authentication protocol for vehicular cloud computing environment which ensures secure communication among the participating entities such as tag, reader, and the cloud server. The uniqueness property of PUF and ECC allows significant functional advantages in ensuring and designing the secure key establishment and communication. Our proposed protocol efficiently supports for the revocation and reissue features and tag’s friendly password update/change mechanism. Using the provable random oracle model, we presented the advantages of an adversary in violating the security features. Moreover, through the informal security analysis, we have shown that the proposed scheme successfully prevents all the well-known security attacks for authentication protocols. Our scheme withstands all the 18 security features and further consumes the computation cost of $67\text{fun} + 27\text{EMP} = 14.7088$ ms which is comparable with the other schemes. Similarly, our scheme consumes the communication cost as 672 bits during the sending mode and 512 bits during the receiving mode. Overall, the performance of our proposed scheme is comparable with the related schemes and provides more security features compared to the other related existing protocols.

Data Availability

No data collection method is applied.

Conflicts of Interest

The authors declare that they have no conflicts of interest.

Acknowledgments

This study did not receive any funding in any form.

References

[1] P. Bagga, A. K. Das, M. Wazid, J. J. P. C. Rodrigues, K.-K. R. Choo, and Y. Park, “On the design of mutual authentication and key agreement protocol in internet of vehicles-enabled intelligent transportation system,” *IEEE Transactions on Vehicular Technology*, vol. 70, no. 2, pp. 1736–1751, 2021.

[2] M. Safkhani, C. Camara, P. Peris-Lopez, and N. Bagheri, “Rseap2: an enhanced version of RSEAP, an RFID based authentication protocol for vehicular cloud computing,” *Vehicular Communications*, vol. 28, Article ID 100311, 2021.

[3] Q. Jiang, J. Ni, J. Ma, L. Yang, and X. Shen, “Integrated authentication and key agreement framework for vehicular cloud computing,” *IEEE Network*, vol. 32, no. 3, pp. 28–35, 2018.

[4] Q. Jiang, N. Zhang, J. Ni, J. Ma, X. Ma, and K.-K. R. Choo, “Unified biometric privacy preserving three-factor authentication and key agreement for cloud-assisted autonomous vehicles,” *IEEE Transactions on Vehicular Technology*, vol. 69, no. 9, pp. 9390–9401, 2020.

[5] A. A. Alamr, F. Kausar, J. Kim, and C. Seo, “A secure ECC-based RFID mutual authentication protocol for internet of things,” *The Journal of Supercomputing*, vol. 74, no. 9, pp. 4281–4294, 2018.
[6] N. Dinarvand and H. Barati, "An efficient and secure RFID authentication protocol using elliptic curve cryptography," *Wireless Networks*, vol. 25, no. 1, pp. 415–428, 2019.

[7] V. Kumar, M. Ahmad, D. Mishra, S. Kumari, and M. K. Khan, "RSEAP: RFID based secure and efficient authentication protocol for vehicular cloud computing," *Vehicular Communications*, vol. 22, Article ID 100213, 2020.

[8] M. Hosseinzadeh, O. H. Ahmed, S. H. Ahmed et al., "An enhanced authentication protocol for RFID systems," *IEEE Access*, vol. 8, 2020.

[9] F. Zhu, "Secmap: a secure RFID mutual authentication protocol for healthcare systems," *IEEE Access*, vol. 8, p. 192, 2020.

[10] S. Gabsi, Y. Kortli, V. Beroulle, Y. Kieffer, A. Alasiry, and B. Hamdi, "Novel ECC-based RFID mutual authentication protocol for emerging IoT applications," *IEEE Access*, vol. 9, 2021.

[11] D. Mishra, V. Kumar, D. Dharminder, and S. Rana, "SFVCC: chaotic map-based security framework for vehicular cloud computing," *IET Intelligent Transport Systems*, vol. 14, no. 4, pp. 241–249, 2020.

[12] U. Chatterjee, R. S. Chakraborty, and D. Mukhopadhyay, "A PUF-based secure communication protocol for IoT," *ACM Transactions on Embedded Computing Systems*, vol. 16, no. 3, pp. 1–25, 2017.

[13] M. N. Aman, K. C. Chua, and B. Sikdar, "Mutual authentication in IoT systems using physical unclonable functions," *IEEE Internet of Things Journal*, vol. 4, no. 5, pp. 1327–1340, 2017.

[14] Q. Jiang, X. Zhang, N. Zhang, Y. Tian, X. Ma, and J. Ma, "Three-factor authentication protocol using physical unclonable function for IoV," *Computer Communications*, vol. 173, pp. 45–55, 2021.

[15] H. Xu, X. Chen, F. Zhu, and P. Li, "A novel security authentication protocol based on physical unclonable function for RFID healthcare systems," *Wireless Communications and Mobile Computing*, vol. 2021, Article ID 8844178, 14 pages, 2021.

[16] P. Gope and B. Sikdar, "Privacy-aware authenticated key agreement scheme for secure smart grid communication," *IEEE Transactions on Smart Grid*, vol. 10, no. 4, pp. 3953–3962, 2018.

[17] Y.-N. Cao, Y. Wang, Y. Ding, H. Zheng, Z. Guan, and H. Wang, "A PUF-based lightweight authenticated metering data collection scheme with privacy protection in smart grid," in *Proceedings of the 2021 IEEE Intl Conf on Parallel & Distributed Processing with Applications, Big Data & Cloud Computing, Sustainable Computing & Communications, Social Computing & Networking (ISPA/BDCloud/SocialCom/SustainCom)*, pp. 876–883, IEEE, New York City, NY, USA, October 2021.

[18] Z. Zhang, Y. Liu, Q. Zuo, L. Harn, S. Qiu, and Y. Cheng, "PUF-based key distribution in wireless sensor networks," *Computers, Materials & Continua*, vol. 64, no. 2, pp. 1261–1280, 2020.

[19] P. Mall, R. Amin, A. K. Das, M. T. Leung, and K.-K. R. Choo, "PUF-based authentication and key agreement protocols for IoT, WSNs and smart grids: a comprehensive survey," *IEEE Internet of Things Journal*, vol. 9, 2022.

[20] Y. Liu, Y. Cui, L. Harn et al., "PUF-based mutual-authenticated key distribution for dynamic sensor networks," *Security and Communication Networks*, vol. 2021, Article ID 5532683, 13 pages, 2021.