electronics

MDPI

Article

An ECC-Based Authentication Protocol for Dynamic Charging System of Electric Vehicles

Jie Wang [1], Shengbao Wang [1,*], Kang Wen [1], Bosen Weng [1], Xin Zhou [2] and Kefei Chen [1]

[1] Key Laboratory of Cryptography of Zhejiang Province, Hangzhou Normal University, Hangzhou 310036, China; jwang@stu.hznu.edu.cn (J.W.); wenkang@stu.hznu.edu.cn (K.W.); wengbosen@stu.hznu.edu.cn (B.W.); kfchen@hznu.edu.cn (K.C.)
[2] School of Mathematics and Statistics, Fujian Normal University, Fuzhou 350117, China; aix@stu.hznu.edu.cn
* Correspondence: shengbaowang@hznu.edu.cn

Abstract: Dynamic wireless charging emerges as a promising technology, effectively alleviating range anxiety for electric vehicles in transit. However, the communication between the system's various components, conducted over public channels, raises concerns about vulnerability to network attacks and message manipulation. Addressing data security and privacy protection in dynamic charging systems thus becomes a critical challenge. In this article, we present an authentication protocol tailored for dynamic charging systems. This protocol ensures secure and efficient authentication between vehicles and roadside devices without the help of a trusted center. We utilize a physical unclonable function (PUF) to resist physical capture attacks and employ the elliptic curve discrete logarithm problem (ECDLP) to provide forward security protection for session keys. We validated the security of our proposed scheme through comprehensive informal analyses, and formal security analysis using the ROR model and formal analysis tool ProVerif. Furthermore, comparative assessments reveal that our scheme outperforms other relevant protocols in terms of efficiency and security.

Keywords: authentication; dynamic charging; handover; security; privacy

check for updates

Citation: Wang, J.; Wang, S.; Wen, K.; Weng, B.; Zhou, X.; Chen, K. An ECC-Based Authentication Protocol for Dynamic Charging System of Electric Vehicles. *Electronics* **2024**, *13*, 1109. https://doi.org/10.3390/electronics13061109

Academic Editors: Zbigniew Kotulski, Enrique Romero-Cadaval, Mehdi Sookhak and Zhuo Ma

Received: 16 December 2023
Revised: 6 February 2024
Accepted: 1 March 2024
Published: 18 March 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

With increasing urbanization, electric vehicles (EVs) play a crucial role in establishing green transportation by providing emission-free operation. They present a promising solution to address energy and environmental challenges [1]. However, the widespread adoption of electric vehicles has underscored the need for a well-developed charging infrastructure, particularly for EV users who are away from home and face time and distance constraints [2].

There are two main charging methods: static charging [3] and dynamic charging [4]. Static charging involves parking the electric vehicle at a charging station, turning off the engine, and connecting the charger to the charging port to obtain electric energy. Dynamic charging allows electric vehicles to charge while driving on the road, utilizing electromagnetic energy transfer between the vehicle and the charging pad. The dynamic charging system consists of the following entities: trusted service provider (TSP), roadside units (RSUs), electric vehicles (EVs), and charge pads (CPs). The system architecture is illustrated in Figure 1. The TSP acts as an energy provider and establishes the necessary infrastructure for electric vehicle charging, which includes RSUs and CPs to form a dynamic power station. The TSP is responsible for the registration of RSUs and EVs. RSUs serve as access points to the road charging area and manage a large number of CPs. They utilize TSP's public key to verify the legitimacy of users and allocate CPs to provide energy for EVs. EVs are equipped with on-board units, sensors, and Global Positioning System (GPS) and travel along the road network. EVs can establish communication links with both RSUs and CPs using dedicated short-range communications (DSRCs). CPs are components that facilitate the charging of electric vehicles in [5]. Each CP is capable of independently supplying energy to EVs. In

comparison to static charging, wireless dynamic charging represents a new paradigm and brings greater convenience. Firstly, the dynamic charging of electric vehicles overcomes the limitation of fixed charging stations and saves time. Secondly, mobile charging of electric vehicles can effectively extend the mileage of electric vehicles.

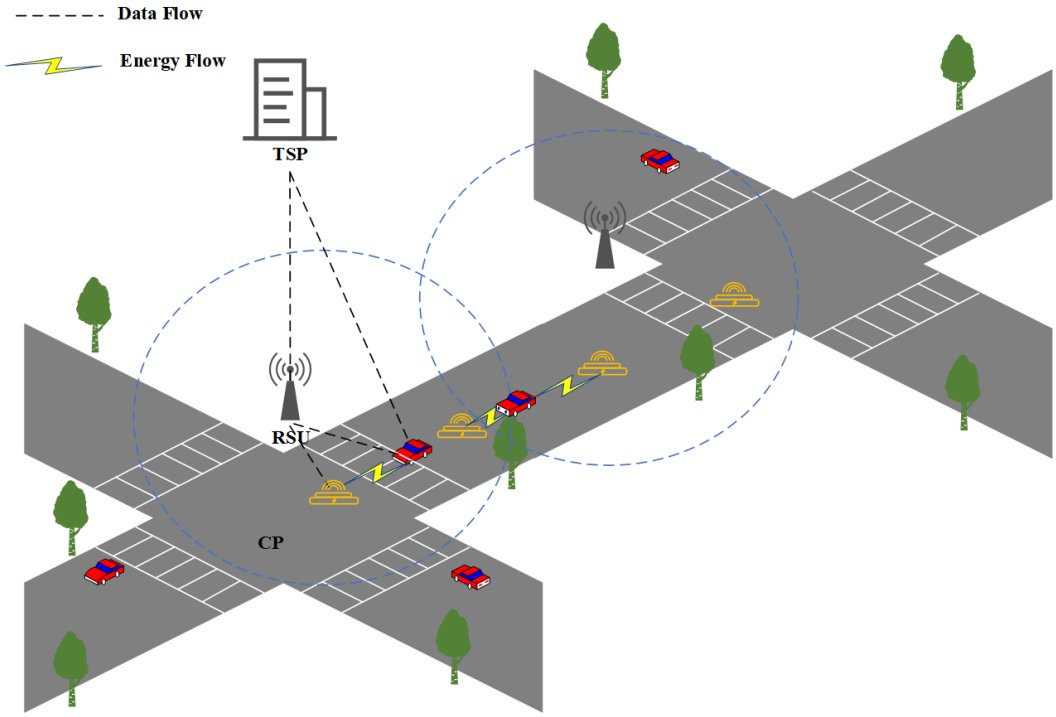

Figure 1. System model.

However, dynamic charging systems face security and communication challenges [6]. Communication between entities occurs over a public channel, which is susceptible to various security threats, including interception, eavesdropping, and message modification [7]. Adversaries can exploit these vulnerabilities to gain unauthorized benefits from energy transactions within the dynamic charging system through impersonation, replay attacks, and man-in-the-middle attacks. Therefore, secure authentication protocols are necessary to mitigate these threats.

1.1. Motivation

Our motivation arises from dual concerns within existing authentication protocols [8–14] for dynamic charging: computational efficiency and security vulnerabilities. The scheme [8,9] has a high computational overhead, while the schemes [10–14] reduce computational overhead but are vulnerable to common attacks. To address this, we propose a secure and efficient identity authentication scheme for electric vehicle dynamic charging scenarios. The main contributions of this paper are summarized as follows:

- We have designed an efficient authentication protocol based on elliptic curve cryptography (ECC) for the dynamic charging system of electric vehicles that can mitigate RSU capture attacks and provide perfect forward secrecy. Also, this protocol enables the authentication process between vehicles and roadside units (RSUs) without the need for a third-party service provider (TSP).
- During the inter-RSU handover authentication process for vehicles, we have adopted a novel approach that eliminates the use of shared secret keys, ensuring the independence of RSUs.
- The proposed protocol is demonstrated to be resilient against various attacks through informal proofs. Furthermore, the protocol's semantic security is formally established in the random oracle model.

- Performance analysis and security analysis demonstrate the practicality and efficiency of the proposed protocol.

1.2. Paper Organization

This paper is organized as follows: Section 2 summarizes related work in this research area. Section 3 describes the background, including the necessary preliminaries, system model, and threat model. Section 4 explains the proposed scheme and its main components. Section 5 provides an informal security analysis of the scheme. Section 6 gives a formal security proof in the Random or Real (ROR) model. Section 7 compares the performance of the proposed scheme with other relevant schemes. Finally, Section 8 concludes the paper.

2. Related Work

For dynamic wireless charging systems of electric vehicles, various key agreement protocols have been proposed to ensure secure and authenticated communication between the electric vehicle and the charging system. Roman et al. [8] designed an authentication scheme for a cloud-based wireless charging system for electric vehicles. In their scheme, vehicle users need to purchase tickets from an electric power service provider to enjoy charging services. However, their scheme requires heavy computation based on bilinear pairings and blind signatures, which can result in a large communication overhead. Rabieh et al. [9] proposed a privacy-preserving authentication scheme that achieves mutual authentication between electric vehicles (EVs) and charging plates without the involvement of a trusted third party. Their scheme also protects users' identities. However, the scheme faces challenges in resisting man-in-the-middle attacks and EV impersonation attacks, and the computational overhead is relatively high.

To improve performance, some lightweight solutions have been proposed, but some security issues still exist. For example, Pazos-Revilla et al. [10] proposed a blind signature-based physical layer assistance scheme for dynamic charging systems to ensure their safety. The key idea is that when an EV authenticates itself to a TSP, the TSP sends a secret seed to the EV to efficiently calculate the shared group key with an RSU. However, the RSU, being exposed in public places, is easy for adversaries to capture, which could lead to the leakage of the shared group key. Li et al. [11] proposed a fast authentication scheme (FADEC) based on elliptic curve cryptography (ECC) to meet the communication requirements during dynamic inductive charging. However, their scheme was found to be vulnerable to replay attacks and privacy issues.

Babu et al. [12] proposed a lightweight authentication scheme based on ECC, where vehicles authenticate with roadside units (RSUs) with the help of edge nodes. However, their scheme is vulnerable to replay attacks and does not satisfy non-linkability. Babu et al. [13] presented a lightweight authentication scheme that incorporates vehicle handover authentication. In this process, the vehicle initiates a handover request with a previously certified roadside unit to communicate with other RSUs under its assistance. Nevertheless, their scheme lacks perfect forward secrecy and is susceptible to replay attacks. Furthermore, Babu et al. [14] proposed another lightweight authentication scheme based on physical unclonable functions. In this scheme, RSUs acquire the physical unclonable function (PUF) response values uploaded by vehicles from a trusted center to enable mutual authentication between vehicles and RSUs. However, their scheme exhibits vulnerabilities to replay attacks and lacks non-linkability and perfect forward secrecy.

In recent years, blockchain technology has been widely applied in dynamic wireless charging systems. Alshaeri et al. [15] proposed a dynamic electric vehicle charging energy trading scheme based on blockchain technology. In their scheme, vehicles purchase tickets from energy providers through smart contracts, and these tickets are encrypted using a shared secret value of the energy provider and RSUs. Abouyoussef et al. [16] proposed a blockchain-based network strategy to support privacy protection for executing dynamic charging. Tajmohammadi et al. [17] proposed a secure and lightweight dynamic wire-

less charging payment protocol. The protocol employs symmetric encryption and XOR operations to safeguard the privacy of the communication.

3. Preliminaries

3.1. Elliptic Curve Cryptography

Elliptic curve cryptography [18] is a public key encryption technology based on elliptic curves over a finite field. Let F_p denote a finite field with a large prime order p. E denote an elliptic curve: $y^2 = x^3 + ax + b \bmod p$, where $x, y, a, b \in F_p$. G is a cyclic subgroup over F_p and P is the generator point.

Definition 1. *(Elliptic Curve Discrete Logarithm Problem): Given a base point P and a point $Q = x \cdot P$, it is computationally difficult to determine the integer x from Q.*

Definition 2. *(Elliptic Curve Computational Diffie–Hellman): Give a base point P and two pointa $Q_1 = a \cdot P$ and $Q_2 = b \cdot P$, it is computationally difficult to compute $Q = a \cdot b \cdot P$ in probabilistic polynomial time.*

3.2. Fuzzy Extractor

A fuzzy extractor [19] can extract the same outputs from the inputs with a certain amount of noise, and it is used to extract and recover the user's biological key. This process can be described by a pair of functions, denoted as $EXT = (Gen, Rep)$.

- $(\sigma, \tau) = Gen(bio)$. The generating function, denoted as $Gen(.)$, takes the input biological information *bio* and produces the biological key σ along with auxiliary information τ.
- $\sigma = Rep(bio, \tau)$. The recovery function, denoted as $Rep(.)$, takes the input biological information *bio* and auxiliary information τ to recover the biological key σ.

3.3. Physical Unclonable Function

A physical unclonable function (PUF) [20] is a hardware security primitive that leverages the unique physical characteristics of a chip to generate an unpredictable response. It possesses reproducibility, uniqueness, and unpredictability properties. By exploiting manufacturing variances, a unique mapping function between the challenge signal and response is established, which can be formalized as $Res = PUF(Cha)$, where Cha represents the challenge and Res represents the response.

In this paper, a PUF is employed within the RSU to safeguard stored confidential information, preventing the adversary from obtaining any information from the RSU.

3.4. Threat Model

In the model we proposed, the TSP is assumed to be a completely trusted and honest entity. The RSU is assumed to be an honest but curious entity. Specifically, the RSU will honestly execute protocol processes and steps; however, it cannot be excluded that the RSU may attempt to obtain more private information from the running process. The EV is assumed to be an entity that could potentially engage in malicious behaviors. They are at risk of illegal operations caused by hacking attacks. $\mathcal{A}$ is defined as the adversary in our scheme. $\mathcal{A}$ has the following capabilities:

- $\mathcal{A}$ can overhear, intercept, and synthesize any publicly transmitted messages. This is in line with the Dolev–Yao threat model [21].
- $\mathcal{A}$ can either be a registered user or an insider attacker with privileged access, capable of obtaining additional information beyond publicly available messages.
- $\mathcal{A}$ can launch side-channel attacks to obtain information stored in the smart card and RSU.

3.5. Security Goals

In this section, we will focus on introducing the security design goals of the proposed protocol. Specifically, the security objectives of this protocol include the following:

Confidentiality: Sensitive information involved in the charging process cannot be intercepted by unauthorized entities during communication, ensuring that only authorized entities can access the data generated during the charging process.

Message integrity: Due to the TSP not being involved in direct communication between EVs and RSUs, EVs and RSUs need to have the capability to mutually verify if messages have been tampered with to ensure the integrity of message transmission.

Anonymity and unlinkability: The personal information of EVs is effectively protected with anonymity during the charging process. Charging behavior and related data cannot be traced or linked back to specific user identities.

Mutual authentication: Mutual authentication is performed between EVs and RSUs to ensure that the EVs are legitimate and trustworthy and that they can verify the identity of the RSUs, establishing a two-way trust relationship.

4. The Proposed Scheme

The proposed scheme consists of six phases, namely system initialization, vehicle registration, RSU registration, login and authentication, charging authentication, and handover authentication. The notations of our protocol are shown in Table 1.

Table 1. Notions.

Notions	Description
TSP	Trusted service provider
RSU_j	j^{th} roadside units
EVU_i	i^{th} electric vehicle user
CP	Charge pad
ID_i	Identity of EVU_i
PW_i	Password of EVU_i
PK_{RSU_j}, c_j	Public and private key pair of RSU_j
PK_{TSP}, s	Public and private key pair of TSP
PID_i	Pseudo-identity of vehicle
T_*	Timestamp
$h(.)$	One-way hash function
$Gen(.)$	The generating function of fuzzy extractor
$Rep(.)$	The reproduction function of fuzzy extractor
$PUF()$	Physical unclonable function
Bio_i	The biological information of EV_i
σ_i, τ_i	Biological key and auxiliary parameter
Cha_j, Res_j	The challenge and response of the PUF in RSU_j
SC_i	The smart scard of EVU_i
$\oplus$	Exclusive OR operation
$\|$	Concatenation operator

4.1. Initialization Phase

In this phase, TSP initializes the system environment to generate system parameters. TSP selects a large prime number q, a non-singular elliptic curve $E(a, b)$ on a finite field F_q and a point $P \in E(a, b)$ as the base point. Then, TSP selects a long-term private key $s \in Z_q^*$, and computes $PK_{TSP} = s \cdot P$. Here, based on the elliptic curve discrete logarithm problem (ECDL), s is secure. Next, TSP chooses SHA-256 as the hash function $h : \{0,1\}^* \rightarrow \{0,1\}^{256}$, and the generation function $Gen(.)$ and recovery function $Rep(.)$ for the fuzzy extractor. Finally, TSP distributes $\{Gen(.), Rep(.), h(.), P, PK_{TSP}\}$ to each entity.

4.2. Vehicle Registration Phase

In this phase, the electric vehicle user EVU_i registers with TSP to obtain its private key. The registration process is conducted over a secure channel between EVU_i and TSP, as shown in Table 2.

Table 2. Vehicle registration phase.

EVU_i		TSP
Input ID_i, PW_i, Bio_i		
	$\xrightarrow[\text{Secure channel}]{\{ID_i\}}$	
		Verify the uniqueness of ID_i
		Generate random numbers r_i, x_i
		$PID_i = h(ID_i \Vert r_i)$
		$X_i = x_i \cdot P$
		$d_i = x_i + h(PID_i \Vert X_i) \cdot s$
		Store $\{ID_i, x_i, r_i\}$ in secure memory
	$\xleftarrow[\text{Secure channel}]{\{PID_i, X_i, d_i\}}$	
$(\sigma_i, \tau_i) = Gen(Bio_i)$		
$A_i = PID_i \oplus h(PW_i \Vert \sigma_i)$		
$B_i = X_i \oplus h(PID_i \Vert \sigma_i)$		
$C_i = d_i \oplus h(X_i \Vert \sigma_i)$		
$D_i = h(ID_i \Vert PW_i \Vert \sigma_i)$		
$\{A_i, B_i, C_i, D_i, \tau_i, Rep(.)\}$ stored in SC_i		

Step VR1: EVU_i selects an identity ID_i and sends the identity ID_i to the trusted service provider (TSP).

Step VR2: Upon receiving ID_i, the TSP first queries the database to verify the uniqueness of ID_i. If it is not unique, TSP rejects the vehicle registration. Else, it selects the random numbers r_i and x_i. Then, the TSP calculates $PID_i = h(ID_i \Vert r_i)$, $X_i = x_i \cdot P$, $d_i = x_i + h(PID_i \Vert X_i) * s$. Finally, the TSP sends $\{PID_i, X_i, d_i\}$ to EVU_i and stores $\{ID_i, x_i, r_i\}$ in its secure memory.

Step VR3: Upon receiving the message, EVU_i first inserts the smart card, inputs PW_i and Bio_i, and calculates $(\sigma_i, \tau_i) = Gen(Bio_i)$. Next, the values $A_i = PID_i \oplus h(PW_i \Vert \sigma_i)$, $B_i = X_i \oplus h(PID_i \Vert \sigma_i)$, $C_i = d_i \oplus h(X_i \Vert \sigma_i)$, and $D_i = h(ID_i \Vert PW_i \Vert \sigma_i)$ are computed. Finally, $\{A_i, B_i, C_i, D_i, \tau_i, Rep(.)\}$ are stored in the smart card SC_i.

4.3. RSU Register Phase

In this phase, the RSU generates its own public and private keys and initiates a registration request to TSP. Each RSU has an independent public/private key pair, instead of using a shared key as in traditional schemes. The registration process is conducted over a secure channel between RSU_j and TSP, as shown in Table 3.

Step RR1: RSU_j selects a random number $c_j \in Z_q^*$ and computes $Pk_{RSU_j} = c_j \cdot P$.

Step RR2: RSU_j sends Pk_{RSU_j} to the trusted service provider (TSP) via a secure channel. Upon receiving Pk_{RSU_j}, the TSP verifies the uniqueness of the identity, stores Pk_{RSU_j} in its memory and sends $\{ACK\}$ to RSU_j.

Step RR3: Upon receiving the message, RSU_j generates a challenge value Cha_j and uses the physical unclonable function (PUF) to calculate the response value Res_j. RSU_j selects the group key G_{pad}, delivers G_{pad} to the charging pads (CPs), and then computes $W_j = c_j \oplus h(Pk_{RSU_j} \Vert Res_j)$ and $Y_j = G_{pad} \oplus h(Pk_{RSU_j} \Vert c_j)$. Finally, RSU_j stores $\{Cha_j, W_j, Y_j\}$ in its memory.

4.4. Login and Authentication Phase

In the login and authentication phase, EVU_i needs to mutually authenticate its identity with the accessed RSU before requesting charging. The authentication process is described in Table 4.

Table 3. RSU registration phase.

RSU_j		TSP
Select a random number c_j $Pk_{RSU_j} = c_j \cdot P$		
	$\xrightarrow[\text{Secure channel}]{\{Pk_{RSU_j}\}}$	
		Verify the uniqueness of Pk_{RSU_j} Store Pk_{RSU_j} in its memory
	$\xleftarrow[\text{Secure channel}]{\{ACK\}}$	
Generate a challenge cha_j $Res_j = PUF(Cha_j)$ $W_j = c_j \oplus h(Pk_{RSU_j} \| Res_j)$ $Y_j = G_{pad} \oplus h(Pk_{RSU_j} \| c_j)$ Store $\{Cha_j, W_j, Y_j\}$		

Table 4. Login and authentication.

EVU_i		RSU_j
Input ID_i,PW_i,Bio_i $\sigma_i = Rep(Bio_i, \tau_i)$ $D_i^* = h(ID_i \| PW_i \| \sigma_i)$ Chech $D_i^* \stackrel{?}{=} D_i$ Generate a random number m_i and the timestamp T_1 $PID_i = A_i \oplus h(PW_i \| \sigma_i)$ $X_i = B_i \oplus h(PID_i \| \sigma_i)$ $d_i = C_i \oplus h(X_i \| \sigma_i)$ $P_i = m_i \cdot P$ $P_{ij} = m_i \cdot PK_{RSU_j}$ $M_1 = PID_i \oplus h(P_{ij} \| T_1)$ $M_2 = X_i \oplus h(PID_i \| P_{ij} \| T_1)$ $e_i = d_i + h(PID_i \| P_i \| T_1) * m_i$		
	$\xrightarrow[\text{Insecure channel}]{Mes_1=\{M_1,M_2,e_i,P_i,T_1\}}$	
		Check the freshness of T_1 $Res_j = PUF(Cha_j)$ $c_j = W_j \oplus h(Pk_{RSU_j} \| Res_j)$ $P_{ij} = c_j \cdot P_i$ $PID_i = M_1 \oplus h(P_{ij} \| T_1)$ $X_i = M_2 \oplus h(PID_i \| P_{ij} \| T_1)$ Verify if $e_i \cdot P \stackrel{?}{=} X_i +$ $h(PID_i \| X_i) \cdot PK_{TSP} + h(PID_i \| P_i \| T_1) \cdot P_i$ If not, abort the request. Else, Generate a random number n_j and the timestamp T_2 $Q_j = n_j \cdot P$ $Q_{ij} = n_j \cdot P_i$ $SK = h(Q_{ij} \| Pk_{RSU_j} \| PID_i)$ $M_3 = h(Q_j \| SK \| PID_i \| Pk_{RSU_j} \| T_2)$
	$\xleftarrow[\text{Insecure channel}]{Mes_2=\{M_3,Q_j,T_2\}}$	
Check the freshness of T_2 $SK = h(m_i \cdot Q_j \| Pk_{RSU_j} \| PID_i)$ $M_3^* = h(Q_j \| SK \| PID_i \| Pk_{RSU_j} \| T_2)$ Check $M_3 \stackrel{?}{=} M_3^*$		

Step LA1: EVU_i initiates the login process by inserting the smart card (SC_i) and entering the identity ID_i, password PW_i, and biological information Bio_i. SC_i computes $\sigma_i = Rep(Bio_i, \tau_i)$ and $D_i^* = h(ID_i \| PW_i \| \sigma_i)$. If $D_i^* \neq D_i$, SC_i rejects EVU_i's login request. Otherwise, go on.

EVU_i selects a random number $m_i \in Z_q^*$ and timestamp T_1. Next, EVU_i computes $PID_i = A_i \oplus h(PW_i \| \sigma_i)$, $X_i = B_i \oplus h(PID_i \| \sigma_i)$, $d_i = C_i \oplus h(X_i \| \sigma_i)$, $P_i = m_i \cdot P$, $P_{ij} = m_i \cdot P_{RSU_j}$, $M_1 = PID_i \oplus h(RID_j \| P_{ij} \| T_1)$, $M_2 = X_i \oplus h(PID_i \| P_{ij} \| T_1)$, and $e_i = d_i + h(PID_i \| P_i \| T_1) * m_i$. Finally, EVU_i sends the message $Mes_1 = \{M_1, M_2, e_i, P_i, T_1\}$ to RSU_j via a public channel.

Step LA2: When RSU_j receives EVU_i's request, it first checks the freshness of T_1. If it is valid, RSU_j computes $Res_j = PUF(Cha_j)$, $c_j = W_j \oplus h(RID_j \| Res_j)$, $P_{ij} = c_j \cdot P_i$, $PID_i = M_1 \oplus h(RID_j \| P_{ij} \| T_1)$, and $X_i = M_2 \oplus h(PID_i \| P_{ij} \| T_1)$. After that, RSU_j checks $e_i \cdot P = X_i + h(PID_i \| X_i) \cdot PK_{TSP} + h(PID_i \| P_i \| T_1) \cdot P_i$. If the condition is not satisfied, RSU_j aborts the request. Otherwise, RSU_j continues the request.

RSU_j selects a random number $n_j \in Z_q^*$ and generates the timestamp T_2. Then, RSU_j computes $Q_j = n_j \cdot P$, $Q_{ij} = n_j \cdot P_i$, $SK = h(Q_{ij} \| Pk_{RSU_j} \| PID_j)$, and $M_3 = h(Q_j \| SK \| PID_i \| Pk_{RSU_j} \| T_2)$, where SK is session key. Finally, RSU_j sends the message $Mes_2 = \{M_3, Q_j, T_2\}$ to EVU_i via a public channel.

Step LA5: After receiving Mes_2, EVU_i checks the validity of T_2. If it is correct, EVU_i further computes $SK = h(m_i \cdot Q_j \| Pk_{RSU_j} \| PID_i)$, $M_3^* = h(Q_j \| SK \| PID_i \| Pk_{RSU_j} \| T_2)$, and verifies if $M_3 \stackrel{?}{=} M_3^*$. If the verification is successful, RSU_j is authenticated by EVU_i. Otherwise, the session is terminated.

EVU_i and RSU_j generate a session key SK to encrypt subsequent communications. With the secure channel established via SK, EVU_i is then able to send a charging request to RSU_j securely.

4.5. Charging Authentication Phase

After EVU_i completes mutual authentication with the RSU, it needs the help of the RSU_j to realize the charging functionality with the CPs. As shown in Table 5, RSU_j issues a charging credential tag_i to EVU_i and CPs. Table 6 shows the process of EVU_i initiating a charging request to CP.

Step CA1: EVU_i selects a random number v_i, calculates $CH_{req} = E_{SK}(v_i, PID_i)$ and initiates a charging request $MES_3 = \{CH_{req}\}$ to RSU_j.

Step CA2: RSU_j selects a random number v_j and generates the timestamp T_3, and expiration time $Time_{end}$, where $Time_{end}$ is the valid time period of the credential. Next, RSU_j calculates $(PID_i, v_i) = D_{SK}(CH_{req})$, $Tag_i = h(v_i \| v_j \| PID_i \| Pk_{RSU_j} \| T_3 \| Time_{end})$, $G_{jk} = h(c_j \cdot PK_{RSU_k} \| T_3)$, $M_4 = E_{SK}(Tag_i, G_{jk})$, and $M_5 = h(Tag_i \| G_{jk} \| M_4 \| T_3)$. Also, RSU_j calculates $G_{pad} = Y_j \oplus h(RID_j \| s_j)$ and $M_6 = E_{G_{pad}}(Tag_i)$. Finally, RSU_j sends $MES_4 = \{M_4, M_5, T_3\}$ and $MES_5 = \{M_6\}$, respectively, to EVU_i and CPs through a public channel.

Step CA3: EVU_i calculates $(Tag_i, G_{jk}) = D_{SK}(M_4)$ and $M_5^* = h(Tag_i \| G_{jk} \| M_4 \| T_3)$ and verifies $M_5^* \stackrel{?}{=} M_5$.

Step CA4: EVU_i then calculates $M_7 = h(Tag_i \| T_4)$, and sends $Mes_6 = \{M_7, T_4\}$ to the CP.

Step CA5: The CP receives the message sent by RSU_j and EVU_i, decrypts Tag_i, calculates $M_7^* = h(Tag_i \| T_4)$, and after verifying the consistency between M_7^* and M_7, allows the user to charge.

From the response of RSU_j, EVU_i obtains a token Tag_i, which represents its charging authorization. With Tag_i, all CPs deployed within the coverage area of RSU_j can recognize EVU_i as an authorized electric vehicle user, and EVU_i can seamlessly obtain charging services.

Table 5. Charging authentication 1.

EVU_i	RSU_j	CP
Generate a random number v_i $CH_{req} = E_{SK}(v_i, PID_i)$		
	$\xrightarrow[\text{Insecure channel}]{MES_3=\{CH_{req}\}}$	
	Select a random number v_j, Generate timestamp T_3 and $Time_{end}$ $(PID_i, v_i) = D_{SK}(CH_{req})$ $Tag_i = h(v_i \Vert v_j \Vert PID_i \Vert Pk_{RSU_j} \Vert T_3 \Vert Time_{end})$ $G_{jk} = h(c_j \cdot PK_{RSU_k} \Vert T_3)$ $M_4 = E_{SK}(Tag_i, G_{jk})$ $M_5 = h(Tag_i \Vert G_{jk} \Vert M_4 \Vert T_3)$ $G_{pad} = Y_j \oplus h(RID_j \Vert c_j)$ $M_6 = E_{G_{pad}}(Tag_i)$	
	$\xleftarrow[\text{Insecure channel}]{Mes_4=\{M_4,M_5,T_3\}}$	$\xrightarrow[\text{Insecure channel}]{MES_5=\{M_6\}}$
$(Tag_i, G_{jk}) = D_{SK}(M_4)$ $M_5^* = h(Tag_i \Vert G_{jk} \Vert M_4 \Vert T_3)$ Check $M_5^* \stackrel{?}{=} M_5$		

Table 6. Charging authentication 2.

EVU_i	CP
$M_7 = h(Tag_i \Vert T_3)$	
$\xrightarrow[\text{Insecure channel}]{MES_6=\{M_7,T_4\}}$	
	$Tag_i = D_{G_{pad}}(M_6)$ $M_7^* = h(Tag_i \Vert T_4)$ Check $M_7^* \stackrel{?}{=} M_7$

4.6. Handover Authentication

EVU_i is transferred from one RSU_j to another RSU_k during dynamic charging and running for handover authentication. The process of handover authentication is presented in Table 7.

Step HA1: EVU_i first generates a random number k_i. Then, EVU_i calculates $N_i = k_i \cdot P$, $HA_{req} = E_{SK}(PID_i)$, and $M_8 = h(N_i \Vert PK_{RSU_j} \Vert PID_i \Vert T_3 \Vert T_5)$, and sends a message $MES_7 = \{HA_{req}, N_i, M_8, PK_{RSU_j}, T_3, T_5\}$ to RSU_k.

Step HA2: After RSU_k receives the message, it first verifies the timestamp and calculates $Res_k = PUF(Cha_k)$, $c_k = W_k \oplus h(PK_{RSU_k} \Vert Res_k)$, $G_{jk}^* = h(c_k \cdot PK_{RSU_j} \Vert T_3)$, $PID_i = D_{G_{jk}^*}(HA_{req})$, and $M_8^* = h(N_i \Vert PK_{RSU_j} \Vert PID_i \Vert T_3 \Vert T_5)$, and checks $M_8^* \stackrel{?}{=} M_8$. Then, RSU_k computes $SK^* = h(c_k \cdot N_i \Vert PID_i \Vert PK_{RSU_k})$ and $M_9 = h(SK^* \Vert PK_{RSU_k} \Vert PID_i \Vert T_6)$. Finally, RSU_k sends a message $MES_8 = \{M_9, T_6\}$ to EVU_i.

Step HA3: After EVU_i receives the message, it verifies the timestamp and computes $SK^* = h(k_i \cdot PK_{RSU_k} \Vert PID_i \Vert PK_{RSU_k})$ and $M_9^* = h(SK^* \Vert PK_{RSU_k} \Vert PID_i \Vert T_6)$, and checks $M_9^* \stackrel{?}{=} M_9$.

When EVU_i moves from the area of RSU_j to the area of RSU_k, it needs to complete a handover authentication. After successful handover authentication, a new session key SK^* is generated between EVU_i and RSU_k. With SK^*, EVU_i sends a charging request to RSU_k and obtains a new charging credential.

Table 7. Handover authentication.

EVU_i		RSU_k
Generate a random number k_i $N_i = k_i \cdot P$ $HA_{req} = E_{G_{jk}}(PID_i)$ $M_8 = h(N_i \| PK_{RSU_j} \| PID_i \| T_3 \| T_5)$	$\xrightarrow[\text{Insecure channel}]{MES_7=\{HA_{req}, N_i, M_8, PK_{RSU_j}, T_3, T_5\}}$	
		Check the freshness of T_3 and T_5 $Res_k = PUF(Cha_k)$ $c_k = W_k \oplus h(PK_{RSU_k} \| Res_k)$ $G_{jk}^* = h(c_k \cdot PK_{RSU_j} \| T_3)$ $PID_i = D_{G_{jk}^*}(HA_{req})$ $M_8^* = h(N_i \| PK_{RSU_j} \| PID_i \| T_3 \| T_5)$ Check $M_8^* \stackrel{?}{=} M_8$ $SK^* = h(c_k \cdot N_i \| PID_i \| PK_{RSU_k})$ $M_9 = h(SK^* \| PK_{RSU_k} \| PID_i \| T_6)$
	$\xleftarrow[\text{Insecure channel}]{Mes_8=\{M_9, T_6\}}$	
Check the freshness of T_6 $SK^* = h(k_i \cdot PK_{RSU_k} \| PID_i \| PK_{RSU_k})$ $M_9^* = h(SK^* \| PK_{RSU_k} \| PID_i \| T_6)$ Check $M_9^* \stackrel{?}{=} M_9$		

5. Informal Security Analysis

5.1. Replay Attack

In our protocol, timestamps are used to ensure the freshness of communication messages. In each session, the freshness of the timestamps is verified when receiving publicly transmitted messages. Any replayed messages cannot pass this freshness verification. Therefore, the proposed scheme is resistant to replay attacks.

5.2. Smart Card Lost Attack

Assuming the smart card is obtained by the adversary $\mathcal{A}$ after being lost, $\mathcal{A}$ attempts to retrieve data $SC_i = \{A_i, B_i, C_i, D_i, \tau_i, Rep(.)\}$ from the smart card using a power analysis attack, where $A_i = PID_i \oplus h(PW_i \| \sigma_i)$, $B_i = X_i \oplus h(PID_i \| \sigma_i)$, $C_i = d_i \oplus h(X_i \| \sigma_i)$, and $D_i = h(ID_i \| PW_i \| \sigma_i)$, and σ_i is the biometric key. However, due to the absence of σ_i, $\mathcal{A}$ cannot obtain any valid parameters. Therefore, the proposed protocol is not vulnerable to smart card lost attacks.

5.3. RSU Captured Attack

The adversary $\mathcal{A}$ attempts power analysis attacks to extract the stored parameters $\{cha_j, W_j, Y_j\}$ from RSU_j. Here, $W_j = c_j \oplus h(RID_j \| res_j)$ and $Y_j = G_{pad} \oplus h(RID_j \| c_j)$, while cha_j represents the challenge of the PUF. As $PUF(cha_j)$ produces variable outputs, the secret parameters c_j and G_{pad} remain inaccessible to $\mathcal{A}$. In this manner, our scheme effectively withstands RSU physical capture attacks.

5.4. User Impersonation Attack

Assuming an adversary $\mathcal{A}$ attempts to impersonate a vehicle and sends an authentication request to the RSU, $\mathcal{A}$ would need to know the vehicle's private key d_i and pseudo-identity PID_i to forge the message $Mes_1 = \{M_1, M_2, e_i, P_i, T_1\}$. However, as demonstrated in Section 5.2, $\mathcal{A}$ cannot obtain this sensitive information from the smart card. Hence, our protocol is resilient against user impersonation attacks.

5.5. RSU Impersonation Attack

Assuming $\mathcal{A}$ tries to impersonate the RSU to authenticate a vehicle, they would need to know the RSU's private key c_j to forge the message $Mes_2 = \{M_3, Q_i, T_2\}$. Nevertheless, as explained in Section 5.3, $\mathcal{A}$ cannot access any useful information from the RSU. Consequently, our protocol can withstand RSU impersonation attacks.

5.6. Perfect Forward Secrecy

In our protocol, EVU_i and RSU_j share a common session key $SK = h(m_i \cdot n_j \cdot P \| RID_j \| PID_i)$. Even if the adversary $\mathcal{A}$ can obtain the private keys d_i and c_j, $\mathcal{A}$ still cannot calculate the session key because $\mathcal{A}$ needs to solve the elliptic curve computational Diffie–Hellman problem to obtain $m_i \cdot n_j \cdot P$ from $m_i \cdot P$ and $n_j \cdot P$. Thus, the security of previous and future session keys remains safe.

5.7. No Online Trust Authority

In the proposed scheme, the trusted service provider (TSP) is responsible for system initialization and generating secrets for entities during the registration phase. However, once this setup is completed, the TSP does not actively engage in the authentication process between electric vehicle users (EVUs) or roadside units (RSUs) and charging points (CPs). As a result, the TSP does not need to maintain an online presence during the authentication procedures.

5.8. Anonymity and Non-Linkability

In the proposed scheme, the vehicle's pseudo-identity is represented as $PID_i = h(ID_i \| r_i)$. The non-reversibility of hash functions makes it challenging to link the pseudo-identity PID_i to the actual identity of the EVU_i. Moreover, PID_i remains concealed throughout the authentication process, and adversaries cannot extract it from either the public channel or the smart card. Consequently, the scheme ensures non-linkability, preventing adversaries from associating specific users with different sessions.

6. Formal Security Analysis

6.1. Formal Proof

In this section, we establish the semantic security of the proposed protocol under the ROR model [22]. The random oracle model is very suitable for analyzing the security of key exchange protocols. In this model, we design a simulator that interacts with the assumed adversary in a series of game-based interactions. The simulator fairly generates and sends information such as parameters and data to the adversary according to the protocol specification. The adversary chooses whether to attack based on the received information, such as decryption or forgery. If the adversary cannot win over the simulator with a significant probability in a sufficient number of rounds of games, then under this game framework, we can consider the protocol to be secure.

The participants consist of EVs and RSUs. For example, let I_{Vi} and I_{RSUj} represent instances of EVU_i and RSU_j, respectively. Adversary $\mathcal{A}$ can launch various queries in an attempt to compromise the security of authentication and session keys. The details of these queries are listed in Table 8.

In semantic security, $\mathcal{A}$ is allowed to make a single query to the function $Test(I_{Vi}, I_{RSUj}, r)$ and multiple other queries to verify the correctness of the return value from $Test(I_{Vi}, I_{RSUj}, r)$. The advantage of $\mathcal{A}$ in guessing the value of r is defined as $Adv_{\mathcal{A}} = |2Pr[suc(\mathcal{A})] - 1| < \eta$, where $Adv_{\mathcal{A}}$ represents the advantage and η is a sufficiently small value.

Table 8. Queries performed in ROR model.

Queries	Description
$Execute(I_{Vi}, I_{RSUj})$	Adversary $\mathcal{A}$ can intercept all publicly transmitted information.
$CorruptU(I_{Vi})$	$\mathcal{A}$ performed a side-channel attack on the smart card and obtained the stored information $\{A_i, B_i, C_i, D_i, \tau_i, Rep(.)\}$.
$CorruptRSU(I_{RSUj})$	$\mathcal{A}$ performed a side-channel attack on the RSU and obtained the stored information $\{RID_j, cha_j, W_j, Y_j\}$.
$Send(I_{Vi}, I_{RSUj}, m)$	$\mathcal{A}$ forges message m and sends it to I_{Vi} and I_{RSUj}. Upon receiving m, if m is valid, I_{Vi} and I_{RSUj} reply to $\mathcal{A}$.
$Reveal(I_{Vi}, I_{RSUj})$	The session keys between I_{Vi} and I_{RSUj} can be queried by $\mathcal{A}$.
$Test(I_{Vi}, I_{RSUj}, r)$	$\mathcal{A}$ selects a session for a challenge. If $u = 1$, $\mathcal{A}$ can obtain the real session key. On the other hand, if $u = 0$, $\mathcal{A}$ will receive a randomly generated string of the same length as the real session key.

Theorem 1 aims to prove that the proposed scheme attains semantic security in the random oracle model, meaning that $Adv_{\mathcal{A}}$ cannot obtain any useful information from the interactive process.

Theorem 1. *Let Adv represent the advantage of adversaries obtaining session keys in polynomial time:* $Adv_{\mathcal{A}} \leq \frac{q_{Ha}^2}{2^{l_{Ha}}} + \frac{q_{Se}}{|l_1||l_2|2^{l_{bio}-1}} + 2Adv_{\mathcal{A}}^{PUF} + 2Adv_{\mathcal{A}}^{ECDLP}$. *$q_{Ha}$, q_{Se}, and q_{Ex} represent the number of hash, send, and execute queries performed by $\mathcal{A}$. l_{Ha} and l_{bio} are the lengths of the hash and biological keys, respectively. l_1 and l_2 are the sizes of the uniformly distributed identity and password dictionaries, and $|l_1|$ and $|l_2|$ represent the size of the range space of each dictionary. The advantages of breaking the PUF and ECDLP by $\mathcal{A}$ are denoted as $Adv_{\mathcal{A}}^{PUF}$ and $Adv_{\mathcal{A}}^{ECDLP}$, respectively.*

Proof. To verify the semantic security of the proposed protocol, the five games $Game_i (0 \leq i \leq 4)$ can be performed by $\mathcal{A}$. $Suc_i (0 \leq i \leq 4)$ means $\mathcal{A}$ can distinguish the session key and a random number u in the $Game_i$. □

$Game_0$: In this game, $\mathcal{A}$ simulates the real attack to the proposed protocol. If $\mathcal{A}$ directly guess the bit u, we obtain

$$Adv_A = |2Pr[Suc_0] - 1]| \tag{1}$$

$Game_1$: In this game, $\mathcal{A}$ simulates an eavesdropping attack using the *Execute* query, allowing $\mathcal{A}$ to intercept all publicly transmitted messages. Then, $\mathcal{A}$ verifies the output of the session key or the random number u using the *Reveal* and *Test* queries. The session key $SK = h(m_i \cdot n_j \cdot P \| RID_j \| PID_j)$ is protected using a hash function. Thus, we obtain

$$Pr[Suc_1] = Pr[Suc_0] \tag{2}$$

$Game_2$: In this game, $\mathcal{A}$ simulates a collision attack on the hash results. To achieve this, $\mathcal{A}$ needs to find a hash collision within polynomial time. As defined by the birthday paradox [23], we obtain

$$|Pr[Suc_2] - Pr[Suc_1]| \leq \frac{q_{Ha}^2}{2^{l_{Ha}}} \tag{3}$$

$Game_3$: In this game, $\mathcal{A}$ executes Corrupt and CorruptRSU queries to obtain the stored information $\{A_i, B_i, C_i, D_i, \tau_i, Rep(.)\}$ in the smart card and $\{RID_j, cha_j, W_j, Y_j\}$ in the RSU. However, it is important to note that $\mathcal{A}$ cannot directly obtain valuable parameters as all the values are masked with secret values ID_i, PW_i, Bio_i, and res_j. To succeed in this game, $\mathcal{A}$ must either accurately guess ID_i, PW_i, and Bio_i, or break the physical unclonable function. The password dictionary is denoted as l_1, the identity dictionary as l_2, and the length of

biological keys as l_{bio}. We will assume the probability of $\mathcal{A}$ breaking the PUF as $Adv^{\mathcal{A}}_{PUF}$. Therefore, we obtain

$$|Pr[Suc_3] - Pr[Suc_2]| \leq \frac{q_{Se}}{|l_1||l_2|2^{l_{bio}}} + Adv^{PUF}_{\mathcal{A}} \tag{4}$$

Game$_4$: $\mathcal{A}$ is capable of obtaining $P_i = m_i \cdot P$ and $Q_j = n_j \cdot P$, which are utilized for session key agreement. By obtaining P_i and Q_j, $\mathcal{A}$ has access to pairs of points on the elliptic curve. To successfully win this game, $\mathcal{A}$ must be able to solve the elliptic curve discrete logarithm problem (ECDLP) [24]. However, without knowledge of the respective scalars m_i and n_j, solving the ECDLP and determining the values of m_i and n_j becomes a challenging task. Therefore, the successful completion of the game requires $\mathcal{A}$ to possess the ability to solve the ECDLP, which is considered a computationally infeasible problem. We obtain

$$|Pr[Suc_4] - Pr[Suc_3]| \leq Adv^{ECDLP}_{\mathcal{A}} \tag{5}$$

All the games have been executed by the adversary. To win the game, $\mathcal{A}$ needs to guess the correct bit u. Therefore, we have

$$Pr[Suc_4] = \frac{1}{2} \tag{6}$$

Combining the above formulas, we have

$$\begin{aligned} \frac{1}{2}Adv_{\mathcal{A}} &= |Pr[Suc_0] - \frac{1}{2}| && (7)\\ &= |Pr[Suc_1] - Pr[Suc_4]| && (8)\\ &\leq |Pr[Suc_1] - Pr[Suc_2]| + |Pr[Suc_2] - Pr[Suc_3]| + |Pr[Suc_3] - Pr[Suc_4]| && (9) \end{aligned}$$

Hence, $Adv_{\mathcal{A}} \leq \frac{q^2_{Ha}}{2^{l_{Ha}}} + \frac{q_{Se}}{|l_1||l_2|2^{l_{bio}-1}} + 2Adv^{PUF}_{\mathcal{A}} + 2Adv^{ECDLP}_{\mathcal{A}}$

6.2. Automatic Formal Verification by ProVerif

Before deploying security protocols in real networks, it is crucial to thoroughly assess the depth and comprehensiveness of their ability to provide robust security. To achieve this goal, we conducted extensive simulation tests on the proposed protocol using the ProVerif simulator. ProVerif is a commonly used formal analysis tool for validating security protocols. It evaluates the robustness of a protocol under different attack scenarios by establishing a model of the protocol and automatically analyzing its security properties. Our simulation tests included simulating various types of attacks, such as man-in-the-middle attacks and replay attacks.

We define channel, basic types, and functions in Figure 2. The proposed scheme involves five events, namely, VLoginPhase(), VAuthentication(), VSessionKey(), RSession(), and RAuthentication(). VLoginPhase() indicates the login phase of the vehicle user, VAuthentication() indicates that the vehicle user sends an authentication request, RAuthentication() indicates that the RSU passes the authentication of the vehicle user, RSession() indicates that the RSU agrees on the session key, and VSessionKey() indicates that the vehicle user argees the session key. The above events and queries are shown in Figure 3.

The operations of the vehicle user and RSU are shown in Figures 4 and 5, respectively. The main process is presented in Figure 6. As shown in Figure 7, the results of the ProVerif simulation provide strong evidence of the security of our scheme. Specifically, the simulation shows that the session key, the secret parameter of the RSU, and the password of the user are all secure against attacks. At the same time, the process of mutual authentication is performed in sequence.

```
(*----------The public channel.----------*)
free chn:channel.
(*----------The basic type.----------*)
type Vehicle.  （*--type participant.--*)
type RSU.
type TSP.
type key.
type CP.
type nonce.
type bioinformation.
type timestamp.
(*---------- The basic variables.----------*)
free IDi:bitstring [private]. (*--The identify of Vehicle.--*)
free PWi:bitstring [private]. (*--The password of Vehicle.--*)
free bioi:bioinformation [private]. (*--The bioinformation of Vehicle.--*)
free s:bitstring [private]. (*--The secret parameter of TSP.--*)
free SKv:bitstring [private].(*--The session key of Vehicle.--*)
free SKr:bitstring [private].(*--The session key of RSU.--*)
free PKrsuj:bitstring [private]. (*--The identify of RSU.--*)
free cj:bitstring [private]. (*-- The secret parameter of RSU.--*)
free di:bitstring [private]. (*-- The secret parameter of Vehicle--*)
free Gpad:bitstring [private]. (*-- The shared secret of CPs and RSU--*)
free P:bitstring. (*--The basic point--*)
free chaj:bitstring. (*--The Challenge value--*)
free vehicle:Vehicle.
free tsp:TSP.
free rsu:RSU.
(*-- Hash operation --*)
fun Hash(bitstring):bitstring.
(*-- Fuzzy Extractor algorithm operation.--*)
fun Gen(bioinformation):bitstring.
fun Rep(bioinformation,bitstring):bitstring.
(*-- pufuction algorithm operation.--*)
fun pufuction(bitstring):bitstring.
(*-- Bit operation--*)
fun bit_timestamp(timestamp):bitstring.
fun key_bit(bitstring):key.
fun bit_nonce(nonce):bitstring.
(*-- Symmetric encryption and decryption algorithm operation.--*)
fun Enc(bitstring,bitstring):bitstring.
fun Dec(bitstring,bitstring):bitstring.
(*--ECC operation--*)
fun EccMul(bitstring, bitstring): bitstring.
fun EccAdd(bitstring, bitstring): bitstring.
(*--bitstring operation--*)
fun add(bitstring, bitstring) : bitstring.
fun mul(bitstring, bitstring) : bitstring.
(*-- XOR operation.--*)
fun XOR(bitstring,bitstring):bitstring.
equation forall x:bitstring,y:bitstring;
XOR(XOR(x,y),y)=x.
(*-- puf operation.--*)
fun pufuctionuction(bitstring) : bitstring.
(*--Concat operation--*)
fun Con(bitstring,bitstring):bitstring.
reduc forall x:bitstring,y:bitstring;
Split(Con(x,y))=(x,y).
(*--Check timestamp Fresh operation--*)
fun checktimestampfresh(bitstring,bool):bool
reduc forall T:bitstring;
checktimestampfresh(T,true)=true
otherwise forall T:bitstring;
checktimestampfresh(T,false)=false.
```

Figure 2. Definitions. * – * –: Comments.

```
(*-------------------------event-------------------------------*)
event VLoginPhase(Vehicle).
event VAuthentication(Vehicle).
event VSessionKey(Vehicle).
event RSessionKey(RSU).
event RAuthentication(RSU).

(*-------------------------queries-----------------------------*)
query attacker(SKv).
query attacker(SKr).
query attacker(PWi).
query attacker(cj).
query attacker(di).
query inj-event(VAuthentication(vehicle)) ==> inj-event(VLoginPhase(vehicle)).
query inj-event(RAuthentication(rsu)) ==> inj-event(VAuthentication(vehicle)).
query inj-event(RSessionKey(rsu)) ==> inj-event(RAuthentication(rsu)).
query inj-event(VSessionKey(vehicle)) ==> inj-event(RSessionKey(rsu)).
```

Figure 3. Events and queries. * – * –: Comments.

```
(*---------------------EVU's process--------------------*)
let
VehicleProcess(IDi:bitstring,PWi:bitstring,bioi:bioinformation,Ai:bitstring,Bi:bitstring,Ci:bitstring,
Di:bitstring,tao:bitstring,PKrsuj:bitstring)=
    let sigma=Rep(bioi,tao) in
    let nDi=Hash(Con(IDi,Con(PWi,sigma))) in
    if nDi = Di then
        event VLoginPhase(vehicle);
    new rmi:nonce;
    new Time1:timestamp;
    let mi=bit_nonce(rmi) in
    let T1=bit_timestamp(Time1) in
    let PIDi=XOR(Ai,Hash(Con(PWi,sigma))) in
    let Xi=XOR(Bi,Hash(Con(PIDi,sigma))) in
    let di=XOR(Ci,Hash(Con(Xi,sigma))) in
    let Pi=EccMul(mi,P) in
    let Pij=EccMul(mi,PKrsuj) in
    let M1=XOR(PIDi,Hash(Con(Pij,T1))) in
    let M2=XOR(Xi,Hash(Con(PIDi,Con(Pij,T1)))) in
    let ei=add(di,mul(Hash(Con(PIDi,Con(Pi,T1))),mi)) in
    out(chn,(M1,M2,ei,Pi,T1));
    event VAuthentication(vehicle);
    in(chn,(M3:bitstring,Qj:bitstring,T2:bitstring));
    let SKv=Hash(Con(EccMul(mi,Qj),Con(PKrsuj,PIDi))) in
    if checktimestampfresh(T2,true) then
        let nM3=Hash(Con(Qj,Con(SKv,Con(PKrsuj,T2)))) in
        if nM3=M3 then
        event VSessionKey(vehicle).
```

Figure 4. Process of the user. * – * –: Comments.

```
(*------------------------RSU's process----------------------*)
let RSUProcess(PKrsuj:bitstring,PKtsp:bitstring,chaj:bitstring,Wj:bitstring,Yj:bitstring)=
    in(chn,(M1:bitstring,M2:bitstring,ei:bitstring,Pi:bitstring,T1:bitstring));
    if checktimestampfresh(T1,true) then
        let resj= pufuction(chaj) in
        let cj=XOR(Wj,Hash(Con(PKrsuj,resj))) in
        let Pij=EccMul(cj,Pi) in
        let PIDi=XOR(M1,Hash(Con(Pi,T1))) in
        let Xi=XOR(M2,Hash(Con(PIDi,Con(Pij,T1)))) in
        if
EccMul(ei,P)=EccAdd(Xi,EccAdd(EccMul(Hash(Con(PIDi,Xi)),PKtsp),EccMul(Hash(Con(PIDi,
Con(Pi,T1))),Pi))) then
            event RAuthentication(rsu);
            new rnj:nonce;
            let nj=bit_nonce(rnj) in
            new Time2:timestamp;
            let T2=bit_timestamp(Time2) in
            let Qj=EccMul(nj,P) in
            let Qij=EccMul(nj,Pi) in
            let SKr=Hash(Con(Qij,Con(PKrsuj,PIDi))) in
            let M3=Hash(Con(Qj,Con(SKr,Con(PIDi,Con(PKrsuj,T2))))) in
            event RSessionKey(rsu);
            out(chn,(M3,Qj,T2)).
```

Figure 5. Process of the RSU. * – * –: Comments.

```
(*-----------------Main process------------------*)
process
    new rmi:nonce;
    new xmi:nonce;
    let ri=bit_nonce(rmi) in
    let xi=bit_nonce(xmi) in
    let PIDi=Hash(Con(IDi,ri)) in
    let Xi=EccMul(xi,P) in
    let di=add(xi,mul(Hash(Con(PIDi,Xi)),s)) in
    let PKtsp=EccMul(s,P) in
    let (sigma:bitstring,tao:bitstring) =Gen(bioi) in
    let Ai=XOR(PIDi,Hash(Con(PWi,sigma))) in
    let Bi=XOR(Xi,Hash(Con(PIDi,sigma))) in
    let Ci=XOR(di,Hash(Con(Xi,sigma))) in
    let Di=Hash(Con(IDi,Con(PWi,sigma))) in
    let PKrsuj=EccMul(cj,P) in
    let resj=pufuction(chaj) in
    let Wj=XOR(cj,Hash(Con(PKrsuj,resj))) in
    let Yj=XOR(Gpad,Hash(Con(PKrsuj,cj))) in
    (
    (!VehicleProcess(IDi,PWi,bioi,Ai,Bi,Ci,Di,tao,PKrsuj))|
    (!RSUProcess(PKrsuj,PKtsp,chaj,Wj,Yj))
)
```

Figure 6. Main process. * – * –: Comments.

```
(*----------------Verification summary------------------*)
Query not attacker(SKv[]) is true.
Query not attacker(SKr[]) is true.
Query not attacker(PWi[]) is true.
Query not attacker(cj[]) is true.
Query not attacker(di[]) is true.
Query inj-event(VAuthentication(vehicle[])) ==> inj-event(VLoginPhase(vehicle[])) is true.
Query inj-event(RAuthentication(rsu[])) ==> inj-event(VAuthentication(vehicle[])) is true.
Query inj-event(RSessionKey(rsu[])) ==> inj-event(RAuthentication(rsu[])) is true.
Query inj-event(VSessionKey(vehicle[])) ==> inj-event(RSessionKey(rsu[])) is true.
```

Figure 7. Results . * – * –: Comments.

7. Performance Comparison

We compare our proposed protocol against existing protocols [8,10,14] based on computational efficiency, communication overhead, and security level.

First, we analyze the security of related schemes. In Roman et al.'s scheme [8], the EV purchases tickets from the TSP and then sends a charging request to Fog Server after being certified. The EV and Fog Server establish a session key with random numbers and use the session key to deliver a valid ticket. Fog Server verifies the validity of ticket and helps the EV connect to an RSU. However, in this way, the EV cannot seamlessly charge from the RSU. Additionally, their scheme fails to achieve mutual authentication and provide perfect forward security. In Pazos-Revilla et al.'s scheme [10], the EV sends a charging request to a TSP, and the session key k is composed of public parameters g_x and g_y. After the TSP's verification, it encrypts a secret parameter token with k and sends it to the EV. However, their scheme uses a Diffie–Hellman key exchange to generate the session key, which is vulnerable to man-in-the-middle attacks. In Babu et al.'s scheme [14], the EV stores PUF's challenge–response pairs in a TSP during the registration phase. Authentication of the EV's identity by an RSU requires help from the TSP. The session key between the EV and RSU is randomly generated without ensuring perfect forward security.

Additionally, the above schemes do not consider a situation where the RSU is captured. Since RSUs are deployed in public areas without any protection mechanisms, they are easy to capture by adversaries. Table 9 compares the security features of the proposed protocol with existing protocols [8,10,14]. Our scheme provides more functional and security properties than all other related protocols.

Table 9. Comparison of security and properties.

Scheme		[8]	[10]	[14]	Ours
Attacks/Properties	A_1	✓	×	✓	✓
	A_2	✓	✓	✓	✓
	A_3	✓	✓	✓	✓
	A_4	✓	×	✓	✓
	A_5	✓	✓	✓	✓
	A_6	×	×	×	✓
	P_1	✓	✓	✓	✓
	P_2	×	✓	×	✓
	P_3	×	✓	✓	✓
	P_4	×	✓	✓	✓
	P_5	✓	✓	×	✓
	P_6	×	✓	×	✓

×: suffer (attacks)/no (properties). ✓: resist (attacks)/possess (properties). Attacks/properties: A_1: off-line password guess attack; A_2: impersonation attack; A_3: replay attack; A_4: man-in-middle attack; A_5: smart card loss attack; A_6: RSU captured attack; P_1: identity anonymity; P_2: no online trust authority; P_3: seamless handover; P_4: mutual authentication; P_5: unlinkability; P_6: perfect forward secrecy.

When an EV roams within or between multiple charging stations, the seamless switching of authentication facilities can be enabled through the exchange of short handover messages. Therefore, achieving continuous authentication requires not only efficient computation but also full consideration of the performance impacts brought by communication overhead. In order to calculate the computational costs of the proposed protocol and compare them with existing related proposals, we adopted the time costs of the scheme proposed by Babu et al. [14] as a measure of the execution time required for different cryptographic operations. The experiments were conducted in a Raspberry Pi environment equipped with a quad-core ARM Cortex-A53 processor and 1GB RAM. The computational costs for the various operations are presented in Table 10. As shown in Table 11, the proposed scheme reduces the computational costs compared to related schemes [8] and [10]. However, compared to [14], our scheme incurs higher computational costs. This is because our scheme satisfies more security attributes. Therefore, our scheme keeps the computational overhead relatively low among related schemes.

Table 10. Simulation result.

Operation	Description	Time (ms)
T_p	Bilinear pairing	≈32.084 ms
T_{exp}	Modular exponentiation	≈5.326 ms
T_{ecc}	ECC point multiplication	≈2.288 ms
$T_{(e/d)}$	Encryption/decryption	≈0.511 ms
T_s	Signature generation	≈2.597 ms
T_{hash}	One-way hash function	≈0.016 ms
T_v	Signature verification	≈4.901 ms
T_s	Signature generation	≈2.597 ms
T_{puf}	Signature generation	≈3.333 ms
T_{fhd}	Signature generation	≈6.370 ms

Table 11. Computational time comparison.

Scheme	Required Operations	Total Time (ms)
[8]	$4T_{ecc} + T_s + T_v + 4T_p + 14T_{hash}$	≈145.21 ms
[10]	$6T_{exp} + 3T_s + T_v + 11T_{hash}$	≈44.824 ms
[14]	$28T_{hash} + 3T_{puf} + 2T_{fhd}$	≈23.187 ms
Ours	$14T_{ecc} + 8T_{e/d} + 2T_{hash} + 2T_{puf}$	≈43.234 ms

To perform an efficiency analysis of the communication overhead of our proposed protocol, we define the specific size of memory overhead for different operands as follows:

- 256 bits for hash functions;
- 320 bits for elliptic curve points;
- 128 bits for AES encryption;
- 128 bits for identities;
- 128 bits for random numbers;
- 32 bits for timestamps.

With the given parameters and message size assumptions, we conducted a comparative analysis of the communication costs of our proposed protocol in comparison to existing protocols in Figure 8. The existing related protocols of Roman et al. [8], Pazos-Revilla et al. [10], and Babu et al. [14] require 4320 bits, 3968 bits, and 3392 bits, respectively. In our scheme, the communication overhead required in the initial authentication is 1728 bits. The communication overhead required in the charging authentication is 1216 bits. The communication overhead required in the handover authentication process is 1376 bits. Hence, the total communication of our scheme is 4320 bits. This is similar to [8]. Our scheme incurs higher communication overhead than schemes [10,14]. However, as shown

in Table 9, Refs. [10,14] cannot satisfy more security attributes. Therefore, the communication cost of our scheme is feasible.

Therefore, our scheme is not only more secure with lower computational overhead compared to related schemes but it is also more suitable for the needs of wireless charging systems. However, in terms of communication overhead, our scheme does not provide significant improvement. In future work, we intend to adopt batch authentication strategies to further reduce time overhead.

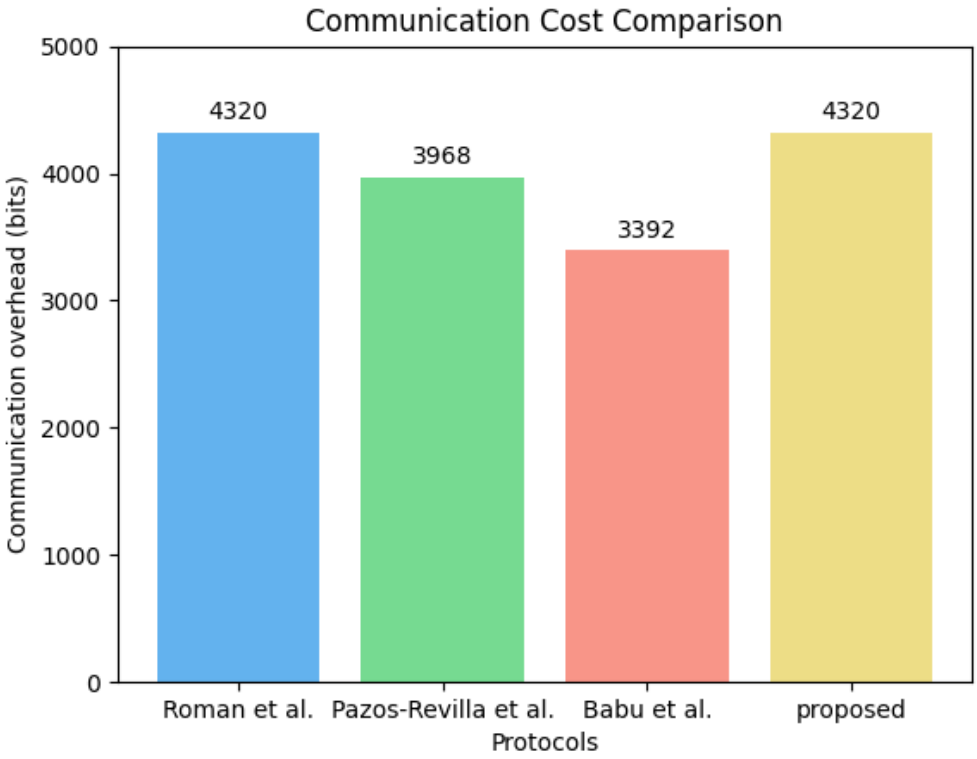

Figure 8. Communication costs comparison [8,10,12].

8. Conclusions

In this paper, we present an elliptic curve cryptography (ECC)-based authentication scheme tailored for dynamic charging systems, enabling mutual authentication between vehicles and roadside units (RSUs) without the need for a trusted third party. As a vehicle transitions to the next RSU, Diffie–Hellman exchanges are leveraged to facilitate seamless handover authentication. To evaluate the security of our protocol, we used the formal tool ProVerif, and employed the ROR model to ensure its semantic security.

Our proposed protocol exhibits robustness against a range of attacks. The incorporation of pseudo identities safeguards user real identities, while the inclusion of physical unclonable functions and biometrics fortifies the defense against RSU physical capture attacks and smart card loss attacks, respectively. Furthermore, comprehensive performance and security analyses show that the proposed protocol is practical.

Author Contributions: Conceptualization, J.W.; Methodology, J.W. and S.W.; Formal analysis, J.W.; Investigation, K.W. and B.W.; Resources, S.W.; Writing—original draft, J.W. and S.W.; Writing—review & editing, X.Z.; Supervision, K.C. All authors have read and agreed to the published version of the manuscript.

Funding: This work was supported by the National Natural Science Foundation of China under Grant U21A20466 and the Hangzhou Joint Fund of the Zhejiang Provincial Natural Science Foundation of China under Grant No. LHZSZ24F020002.

Data Availability Statement: The data presented in this study are available in this article.

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

References

1. Cai, L.; Pan, J.; Zhao, L.; Shen, X. Networked Electric Vehicles for Green Intelligent Transportation. *IEEE Commun. Stand. Mag.* **2017**, *1*, 77–83. [CrossRef]
2. Faraj, M.; Basir, O. Range anxiety reduction in battery-powered vehicles. In Proceedings of the 2016 IEEE Transportation Electrification Conference and Expo (ITEC), Dearborn, MI, USA, 27–29 June 2016; pp. 1–6. [CrossRef]
3. Márquez-Fernández, F.J.; Bischoff, J.; Domingues-Olavarría, G.; Alaküla, M. Assessment of Future EV Charging Infrastructure Scenarios for Long-Distance Transport in Sweden. *IEEE Trans. Transp. Electrif.* **2022**, *8*, 615–626. [CrossRef]
4. Kaswan, A.; Jana, P.K.; Das, S.K. A Survey on Mobile Charging Techniques in Wireless Rechargeable Sensor Networks. *IEEE Commun. Surv. Tutorials* **2022**, *24*, 1750–1779. [CrossRef]
5. Machura, P.; Li, Q. A critical review on wireless charging for electric vehicles. *Renew. Sustain. Energy Rev.* **2019**, *104*, 209–234. [CrossRef]
6. Babu, P.R.; Palaniswamy, B.; Reddy, A.G.; Odelu, V.; Kim, H.S. A survey on security challenges and protocols of electric vehicle dynamic charging system. *Secur. Priv.* **2022**, *5*, e210. [CrossRef]
7. Ashrif, F.F.; Sundararajan, E.A.; Ahmad, R.; Hasan, M.K.; Yadegaridehkordi, E. Survey on the authentication and key agreement of 6LoWPAN: Open issues and future direction. *J. Netw. Comput. Appl.* **2023**, 103759. [CrossRef]
8. Roman, L.F.; Gondim, P.R. Authentication protocol in CTNs for a CWD-WPT charging system in a cloud environment. *Hoc Netw.* **2020**, *97*, 102004. [CrossRef]
9. Rabieh, K.; Wei, M. Efficient and privacy-aware authentication scheme for EVs pre-paid wireless charging services. In Proceedings of the 2017 IEEE International Conference on Communications (ICC), Paris, France, 21–25 May 2017; pp. 1–6. [CrossRef]
10. Pazos-Revilla, M.; Alsharif, A.; Gunukula, S.; Guo, T.N.; Mahmoud, M.; Shen, X. Secure and Privacy-Preserving Physical-Layer-Assisted Scheme for EV Dynamic Charging System. *IEEE Trans. Veh. Technol.* **2018**, *67*, 3304–3318. [CrossRef]
11. Li, H.; Dán, G.; Nahrstedt, K. Proactive key dissemination-based fast authentication for in-motion inductive EV charging. In Proceedings of the 2015 IEEE International Conference on Communications (ICC), London, UK, 8–12 June 2015; pp. 795–801. [CrossRef]
12. Babu, P.R.; Amin, R.; Reddy, A.G.; Das, A.K.; Susilo, W.; Park, Y. Robust Authentication Protocol for Dynamic Charging System of Electric Vehicles. *IEEE Trans. Veh. Technol.* **2021**, *70*, 11338–11351. [CrossRef]
13. Babu, P.R.; Reddy, A.G.; Palaniswamy, B.; Kommuri, S.K. EV-Auth: Lightweight Authentication Protocol Suite for Dynamic Charging System of Electric Vehicles With Seamless Handover. *IEEE Trans. Intell. Veh.* **2022**, *7*, 734–747. [CrossRef]
14. Babu, P.R.; Reddy, A.G.; Palaniswamy, B.; Das, A.K. EV-PUF: Lightweight Security Protocol for Dynamic Charging System of Electric Vehicles Using Physical Unclonable Functions. *IEEE Trans. Netw. Sci. Eng.* **2022**, *9*, 3791–3807. [CrossRef]
15. Alshaeri, A.; Younis, M. A Blockchain-based Energy Trading Scheme for Dynamic Charging of Electric Vehicles. In Proceedings of the 2021 IEEE Global Communications Conference (GLOBECOM), Madrid, Spain, 7–11 December 2021; pp. 1–6. [CrossRef]
16. Abouyoussef, M.; Ismail, M. Blockchain-Based Privacy-Preserving Networking Strategy for Dynamic Wireless Charging of EVs. *IEEE Trans. Netw. Serv. Manag.* **2022**, *19*, 1203–1215. [CrossRef]
17. Tajmohammadi, M.; Mazinani, S.M.; Nikooghadam, M.; Al-Hamdawee, Z. LSPP: Lightweight and Secure Payment Protocol for Dynamic Wireless Charging of Electric Vehicles in Vehicular Cloud. *IEEE Access* **2019**, *7*, 148424–148438. [CrossRef]
18. Boneh, D. The decision diffie-hellman problem. In Proceedings of the International Algorithmic Number Theory Symposium, Portland, OR, USA, 21–25 June 1998; Springer: Berlin/Heidelberg, Germany, 1998; pp. 48–63.
19. Dodis, Y.; Reyzin, L.; Smith, A. Fuzzy extractors: How to generate strong keys from biometrics and other noisy data. In Proceedings of the Advances in Cryptology-EUROCRYPT 2004: International Conference on the Theory and Applications of Cryptographic Techniques, Interlaken, Switzerland, 2–6 May 2004; Proceedings 23; Springer: Berlin/Heidelberg, Germany, 2004; pp. 523–540.
20. Xie, Q.; Ding, Z.; Zheng, P. Provably Secure and Anonymous V2I and V2V Authentication Protocol for VANETs. *IEEE Trans. Intell. Transp. Syst.* **2023**, *24*, 7318–7327. [CrossRef]
21. Dolev, D.; Yao, A. On the security of public key protocols. *IEEE Trans. Inf. Theory* **1983**, *29*, 198–208. [CrossRef]
22. Bellare, M.; Rogaway, P. Random oracles are practical: A paradigm for designing efficient protocols. In Proceedings of the Proceedings of the 1st ACM Conference on Computer and Communications Security, Fairfax, VA, USA, 3–5 November 1993; pp. 62–73.

23. Flajolet, P.; Gardy, D.; Thimonier, L. Birthday paradox, coupon collectors, caching algorithms and self-organizing search. *Discret. Appl. Math.* **1992**, *39*, 207–229. [CrossRef]
24. Koblitz, N. Elliptic curve cryptosystems. *Math. Comput.* **1987**, *48*, 203–209. [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.