sensors

MDPI

Article

TriNymAuth: Triple Pseudonym Authentication Scheme for VANETs Based on Cuckoo Filter and Paillier Homomorphic Encryption

Luyuan Zhuang, Nan Guo * and Yufan Chen

Computer Science & Engineering College, Northeastern University, Shenyang 110167, China
* Correspondence: guonan@mail.neu.edu.cn

Abstract: In VANETs, owing to the openness of wireless communication, it is necessary to change pseudonyms frequently to realize the unlinkability of vehicle identity. Moreover, identity authentication is needed, which is usually completed by digital certificates or a trusted third party. The storage and the communication overhead are high. This paper proposes a triple pseudonym authentication scheme for VANETs based on the Cuckoo Filter and Paillier homomorphic encryption (called TriNymAuth). TriNymAuth applies Paillier homomorphic encryption, a Cuckoo Filter combining filter-level and bucket-level, and a triple pseudonym (homomorphic pseudonym, local pseudonym, and virtual pseudonym) authentication to the vehicle identity authentication scheme. It reduces the dependence on a trusted third party and ensures the privacy and security of vehicle identity while improving authentication efficiency. Experimental results show that the insert overhead of the Cuckoo Filter is about 10 μs, and the query overhead reaches the ns level. Furthermore, TriNymAuth has significant cost advantages, with an OBU enrollment cost of only 0.884 ms. When the data rate in VANETs $dr \leq 180$ kbps, TriNymAuth has the smallest total transmission delay cost and is suitable for shopping malls and other places with dense traffic.

Keywords: VANETs; Paillier homomorphic encryption; cuckoo filter; identity authentication; privacy protection

check for updates

Citation: Zhuang, L.; Guo, N.; Chen, Y. TriNymAuth: Triple Pseudonym Authentication Scheme for VANETs Based on Cuckoo Filter and Paillier Homomorphic Encryption. *Sensors* **2023**, *23*, 1164. https://doi.org/10.3390/s23031164

Academic Editor: Leonard Barolli

Received: 30 October 2022
Revised: 16 December 2022
Accepted: 19 December 2022
Published: 19 January 2023

Copyright: © 2023 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

In intelligent transportation systems, Vehicular Ad Hoc Networks (VANETs) can realize real-time communication between vehicles and infrastructure. VANETs play an active role in improving traffic safety, reducing environmental pollution, alleviating traffic congestion, and providing convenient transportation. Sensitive information such as vehicle position, trajectory, and speed, will be generated in the communication process of VANETs. Generally, it is not recommended to encrypt this sensitive information to protect vehicle privacy because vehicle communication in VANETs usually needs this information.

Vehicles usually use pseudonyms instead of their true identities to realize anonymous communication. It is necessary to achieve message integrity and authenticate the identity of the communication entity in the process of message transmission. Authentication ensures that only legal and valid entities have access to sensitive information transmitted in VANETs. Furthermore, the vehicle's location privacy is associated with the driver's personal information because the trajectory of the vehicle usually has something to do with the driver's personal information, which, if leaked, may endanger the driver's life and property. Therefore, user privacy protection is very important. To achieve the timely accountability of malicious vehicles, it is necessary to realize conditional privacy protection, which ensures that only the Trusted Authority (TA) can obtain the vehicle's real identity in a timely and effective manner and hold the malicious vehicle accountable.

At present, some anonymous authentication schemes based on the Bloom Filter (BF), which can reduce message transmission overhead and improve authentication efficiency,

are proposed. The BF is a data structure with high space utilization, but the BF does not support dynamic addition and deletion of entries. In 2014, Bin et al. [1] proposed a new data structure to improve the BF, called the Cuckoo Filter (CF). The CF is a data structure for approximate set membership queries. It improves the BF in the following three aspects: (1) it supports deleting items dynamically, (2) better search performance, and (3) Better storage performance for applications that require a low false positive rate.

1.1. Motivation and Contributions

This paper proposes a triple pseudonym authentication scheme for VANETs based on the Cuckoo Filter and Paillier homomorphic encryption (called TriNymAuth). The goal is to solve the problems of certificate storage and certificate management in traditional authentication schemes and solve the problem that other anonymous authentication schemes rely too much on RoadSide Unit (RSU) or TA for identity authentication. Based on the efficient query and storage advantages of the CF, TriNymAuth reduces storage and computation overhead. TriNymAuth uses homomorphic pseudonym (HomoNym), local pseudonym (LocNym), and virtual pseudonym (VirNym) to realize the triple authentication of vehicle identity: (1) HomoNym solves the pseudonym self-updating problem of the OnBoard Unit (OBU), and the Homomorphic Pseudonym Provider (HPP) does not need to preload HomoNyms for vehicles, which saves the storage space of the OBU. The HPP does not need to update the vehicle's HomoNyms online, which reduces the computational overhead of the HPP. (2) In V2V authentication, LocNym is used for vehicle identity authentication inside VANETs, which effectively prevents the HPP from associating vehicle identities inside VANETs. (3) VirNym prevents the RSU from associating LocNyms with OBU. As can be seen, triple pseudonym authentication ensures the security and privacy of vehicle identity while implementing identity authentication and reducing communication overhead between the vehicle and the RSU or the vehicle and the HPP during V2V authentication. The following presents the main contributions of TriNymAuth.

1. The idea of using triple pseudonyms is one of the main contributions of this paper.
   - HomoNym realizes the correspondence between the vehicle's real identity ID and multiple HomoNyms, such as OBU$_A$'s $(ID_A, HomoNym_{Ai})$. HomoNyms is updated synchronously by the HPP and OBU.
   - LocNym implements the correspondence between HomoNym and LocNym, such as OBU$_A$'s $(HomoNym_A, LocNym_A)$. LocNym is generated by the OBU and registered with the RSU.
   - VirNym implements the correspondence between LocNym and multiple VirNyms, such as OBU$_A$'s $(LocNym_A, VirNym_{Ai})$, and the OBU periodically updates a set of virtual pseudonyms to be used during V2V communication. Each virtual pseudonym becomes invalid when it is used up.

   The triple pseudonyms are updated synchronously. There are two update opportunities for triple pseudonyms: (1) update in accordance with the suggested updated cycle in 5GAA [2], and (2) update when OBU drives across RSU regions.
2. The update of the vehicle's HomoNyms does not depend on the preloading or online update of TA, which reduces the storage and communication overhead.
3. A two-stage HomoNym enrollment protocol is based on the CF. Fresh vehicles joining VANETs must apply to the RSU for HomoNym enrollment, the RSU forwards the enrollment message to the HPP, and the HPP verifies the validity of the vehicle's HomoNym using an efficient CF query service. The vehicle does not need to sign with the root certificate private key, and the HPP does not need to verify the signature of the vehicle, which reduces the computational overhead. Inside the VANETs, vehicles use LocNyms and local private keys that are regularly updated for identity authentication and do not rely on the HPP, which improves the authentication efficiency. This separation of internal and external identity authentication in VANETs achieves identity privacy protection, unlinkability of HomoNyms, non-repudiation, and message integrity.

4. The HomoNym revocation protocol is based on Paillier homomorphic encryption and the CF. In the aspect of identity tracing (revocation), because all the HomoNyms generated in the life cycle of the vehicle are stored in the HPP, the HPP can quickly obtain the malicious vehicle's true identity based on Paillier homomorphic decryption by using the additive property of homomorphic encryption in the abnormal situation, which realizes the traceability of vehicle identity and reduces the storage overhead. Different from other traditional authentication schemes that distribute Certificate Revocation Lists (CRL) for vehicles, TriNymAuth maintains the CRL in the CF to store revoked HomoNyms and verifies the validity of the HomoNyms by querying the CRL. This reduces the overhead of CRL distribution management.
5. The V2V authentication mechanism is based on VirNyms' exchange and usage. A series of VirNyms are generated and exchanged between vehicles for subsequent communication. The vehicle uses the Elliptic Curve Digital Signature Algorithm (ECDSA) [3] to sign VirNyms to realize identity authentication, which not only reduces the dependence on the RSU and the HPP but also solves the key escrow problem, which is the most common problem in identity-based authentication schemes. It improves the efficiency of authentication while ensuring the privacy protection of vehicle identity, non-repudiation, and message integrity. In addition, TriNymAuth also satisfies the security and privacy requirements of unlinkability, traceability, and avoiding impersonation attacks in VANETs.

1.2. Paper Organization

TriNymAuth is organized as follows: (1) Section 2 is the related works section, which introduces common authentication schemes in VANETs and explains the advantages of filter-based anonymous authentication schemes and the recent advancements in this field. (2) Section 3 is the preliminaries section, which introduces the system model and related technologies involved in the scheme. (3) Section 4 is the pseudonym management scheme based on Paillier homomorphic encryption and the CF, which introduces the specific protocols involved in each stage according to the pseudonym life cycle order. (4) Section 5 is the security and privacy analysis, which introduces the security and privacy satisfied by TriNymAuth and compares it with related works. (5) Section 6 is the performance analysis, which introduces the performance advantages of the CF itself and, on this basis, gives a comparative analysis of the enrollment cost, computational cost, communication cost, and total transmission delay of the scheme.

2. Related Works

This section introduces common authentication schemes in VANETs and divides them into identity-based authentication schemes, BF-based authentication schemes, and CF-based authentication schemes. The specific classification is as follows:

2.1. Identity-Based Authentication Scheme

A lot of identity-based authentication schemes in VANETs have been proposed in recent years. In 2008, an identity-based batch verification (IBV) scheme [4] was proposed, which realized conditional privacy protection. In the IBV scheme, the RSU could verify multiple signatures simultaneously, which reduced the verification time. Moreover, in the IBV scheme, certificates were not required, which significantly reduced the transmission overhead. In 2011, Chim et al. [5] proposed a Secure and Privacy Enhancing Communications Scheme (SPECS). SPECS found that there were some limitations in the IBV scheme [4]. Firstly, the IBV scheme relied heavily on the Tamper Proof Device (TPD), which preloaded all keys within the system. Once compromised, the entire system was compromised. Secondly, the IBV scheme did not meet the privacy requirements, and the real identity of the vehicle could be traced by anyone. Thirdly, the IBV scheme could not resist an impersonation attack and an anti-traceability attack. Finally, in IBV scheme's batch verification,

if there was something wrong in one of the signatures, the whole batch would be discarded, which reduced the efficiency of signature verification.

In 2020, AL-SHAREEDA et al. [6] proposed a VANET-Based Privacy-Preserving Communication Scheme (VPPCS). Based on signatures and verification, VPPCS realized identity authentication and ensured data privacy. However, when the pseudonym set expired, the vehicle needed to delete the old pseudonym set and then request to obtain a new pseudonym set, which increased the computational overhead of the system and the storage overhead of the vehicle.

2.2. BF-Based Authentication Scheme

SPECS [5] not only addressed the limitations of the IBV scheme [4], but also improved space utilization, reduced the storage overhead, and addressed the storage overhead problem in VPPCS [6]. SPECS [5] proposed a new identity-based authentication scheme using the BF and a binary search algorithm that used two shared secrets to meet the privacy requirements. In 2013, Horng et al. [7] proposed batch verification for secure pseudonymous authentication (b-SPECS+). The b-SPECS+ pointed out that SPECS was also not resistant to impersonation attacks. Therefore, b-SPECS+ improved the message signing phase in SPECS, and the improved scheme could meet the security and privacy requirements.

In order to ensure VANETs' security, it is essential to revoke the access rights of malicious nodes with a history of misconduct [8]. Certificates in the CRL refer to public key certificates. A pseudonym is actually a short-term public key. The CRL is not only a public key certificate revocation list but also a short-term public key (pseudonym) revocation list. The CRL is a common way to verify whether the certificate is revoked.

Based on the high space utilization of the BF, some schemes using the BF to store the CRL were proposed [9–12], which reduced the size of the CRL. In 2017, Zhong et al. [9] proposed a Conditional Privacy-Preserving Authentication (CPPA) scheme using the BF to reduce the size of the CRL. This scheme did not use bilinear pairing and reduced the computational overhead. However, if an RSU is compromised, the vehicles' real identities in the range of the RSU will be obtained by the adversary. Moreover, the BF needed to be updated whenever a notification message was generated because the BF did not support dynamic increase, so the time cost was large.

2.3. CF-Based Authentication Scheme

In 2017, Cui et al. [13] proposed a CF-based privacy-preserving authentication scheme (SPACF). Based on the dynamic update and deletion characteristics of CF, SPACF could alleviate the problem that updating the BF takes a long time. In SPACF's batch verification phase, the CF and binary search methods were adopted to achieve a high batch verification success rate. However, since SPACF [13] used identity-based signatures, there was a key escrow problem.

Zhang et al. [14] proposed to directly apply the CF to CRL generation and used the Certificate Expiration List (CEL) to assist the dynamic deletion of expired certificates. CEL provided more free space for storing other valid certificate fingerprints and reduced the probability of a hash collision.

In 2019, Alazzawi et al. [15] proposed a CPPA scheme for VANETs based on pseudonym roots to obtain pseudonyms. This scheme did not use bilinear pairing, had low computational overhead, did not use CRL, and reduced storage and communication overhead. However, the RSU was required to broadcast the notification messages in the CF periodically and act as the intermediate node between the vehicle and TA when the vehicles authenticated each other. The vehicle's identity authentication relied on TA, which was difficult to deploy in areas where RSUs were not deployed or were sparsely deployed.

In 2020, a CF-based privacy-preserving authentication scheme for V2V and V2I communication was proposed in the literature [16]. The scheme used a binary search algorithm to establish positive and negative filter pools for integer key verification and improved verification efficiency through batch verification.

In this paper, TriNymAuth uses Paillier homomorphic encryption to generate and self-update the vehicle's HomoNym, which solves the high computational overhead and reduces the overhead of storing HomoNym in OBU. The HPP uses the CF to store HomoNyms and CRL, which effectively uses the space of the HPP and avoids the overhead of distributing CRL and the overhead of managing CRL on OBU. In the two-stage HomoNym enrollment phase, the vehicle uses the HomoNym and LocNym to apply for HomoNym enrollment from the RSU. The HPP queries the HomoNym based on the CF to achieve efficient verification of HomoNym. In this phase, signatures and verification are not needed, which reduces the computational overhead. In V2V authentication of VirNyms' exchange and usage, VirNyms based on ECDSA algorithm signatures are used to achieve identity authentication between vehicles, which avoids impersonation attacks. Moreover, TriNymAuth satisfies VANETs' security and privacy requirements, such as unlinkability and traceability.

3. Preliminaries

This section describes the system model of TriNymAuth, the Cuckoo Filter for storing HomoNyms and the CRL, the Paillier encryption algorithm for homomorphic pseudonym generation and update, and the ECDSA algorithm for identity authentication during V2V communication.

3.1. System Model

Figure 1 shows the system model based on the CF, where the OBU registers with the HPP by in-person registration, so the link between the OBU and the HPP is not drawn in the figure. The system model consists of a management layer and a perception layer. The Dedicated Short Range Communication (DSRC) protocol is used to realize the wireless and wired communication between the two layers. The management layer contains the HPP. The perception layer includes OBUs and RSUs. The details of the entities are presented below:

- HPP

The HPP is a trusted third party and is not subject to any threats [17]. It is responsible for the generation of system parameters as well as the registration of entities in VANETs and generates the initial registration information for them. The HPP uses the CF as the internal storage data structure, maintains the CF service, and realizes the internal HomoNym update, query, and HomoNym state change services. A secure channel based on symmetric encryption is implemented between the HPP and RSUs.

- RSU

The RSU is semi-trusted. The communication distance between the RSU and vehicles is at least twice the communication distance between vehicles to ensure that when the RSU receives a message, all vehicles that receive the message are within the jurisdiction of the RSU [18,19]. The RSU has greater computing power than OBUs, which can check the messages' validity received from vehicles.

- OBU

OBU is a device which is installed in each vehicle. OBU can publish beacons periodically and is used to connect other vehicles. Each OBU has a TPD, which is used to save secure information. Each vehicle carries a root certificate and a pair of manufacturer public and private key pairs (such as PK_A and SK_A for OBU_A) when it leaves the manufacturer.

Based on the IEEE 802.11p communication protocol, VANETs implement real-time communication between vehicles and other vehicles and between vehicles and infrastructure, namely Vehicle-to-Vehicle (V2V) communication and Vehicle-to-Infrastructure (V2I) communication. In addition, as a special type of mobile ad hoc network, VANETs use vehicles as nodes and wireless and wired communication as their communication technology. Its particularity is mainly reflected in the following aspects: (1) high mobility of nodes,

(2) regularity of node movement, (3) privacy of node information, (4) frequent exchange of information, (5) differences in traffic scenes, (6) relatively sufficient resources in node, and (7) unstable wireless communication channel [20–23].

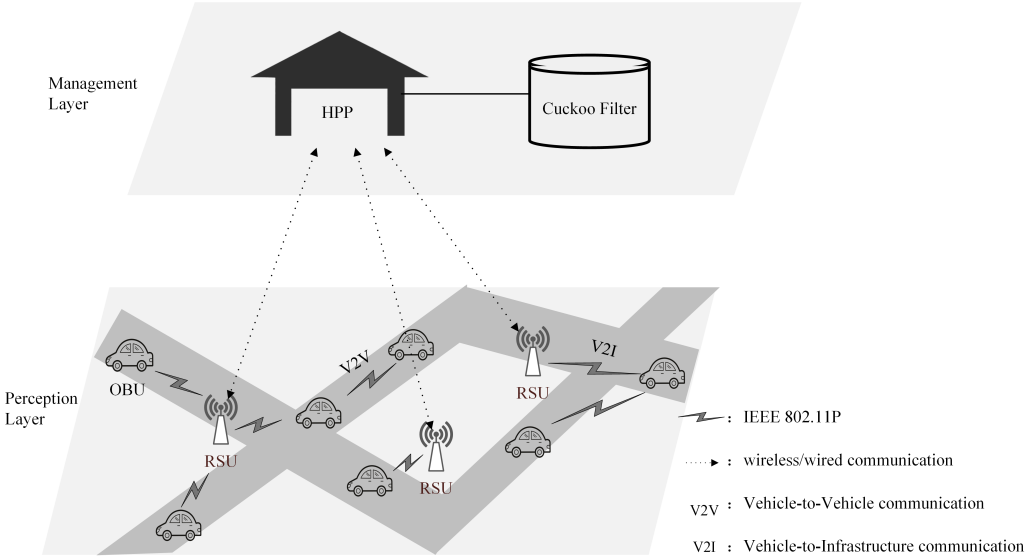

Figure 1. System model.

3.2. Cuckoo Filter

The number of HomoNyms in VANETs is constantly growing. Using filters to store HomoNyms in TriNymAuth can effectively save the storage space of HomoNyms in memory, avoid frequent disk reads and writes, and improve the management efficiency of HomoNyms. Using filters can store smaller fingerprints than using a traditional hash table to store HomoNyms. A popular filter data structure is the BF, created in 1970 by Burton Howard Bloom, where each raw data object is mapped to a number of k bits in a bit vector, where k is the number of independent hash functions. The CF is a compact variant of the Cuckoo hash table; it stores only fingerprints (using the hash function derived from inserting each item as a string) rather than a key-value pair [23]. The fingerprint is calculated by the hash function, usually within 4 bytes, and as long as the selected uniform hash function and the right length of the fingerprint are selected, the hash collision probability can be minimized.The BF does not support entry deletion. The CF not only supports adding and deleting items dynamically but also has greater search performance and better space efficiency. To query whether an item x is in a set, simply search the hash table for x's fingerprint and return true if the same fingerprint is found.

The CF can only use two hash functions to calculate candidate indexes, and because these two candidate indexes can be obtained by XOR operation, they are not completely independent, and the length of the filter is limited. Lailong Luo [24] and others believed that the reason for this phenomenon was that the CF had a strong dependence on the length of the filter when calculating the index of the cell or candidate bucket used to store elements. No matter how data sets changed, their capacity must be predefined and kept unchanged. Therefore, they proposed an Index-Independent Cuckoo Filter (I2CF), which decoupled the dependence between bucket index and filter length and realized bucket-level elastic capacity. Further, they organized I2CF into a dynamic list, thus obtaining filter-level elastic capacity. This is the Consistent Cuckoo Filter (CCF) used in this paper.

Algorithms 1–3 present the insert, query, and delete algorithms of the CCF, respectively, where x represents inserted data, f represents a fingerprint, *last* represents the last bucket that was replaced, and Max limits the maximum number of iterations. Pagh et al. [25] demonstrated that Max must be set to $\lceil 6log_{1+\delta/2}|T| \rceil$ in order for the expected time of all operations to be constant, where δ is a small number and $|T| \geq (2+\delta)n$. I2CF uses k

mutually independent hash functions, $h_i(f)(1 \leq i \leq k)$, to compute candidate buckets for fingerprint storage.

Algorithm 1 Insert(x)

Input. $x, last, Max, k$
Output. true/false
1: $f \leftarrow h_0(x)$
2: $last \leftarrow \emptyset$
3: **for** $i \leftarrow 0$ to Max **do**
4: calculate hash values $h_1(f), h_2(f), \ldots, h_k(f)$
5: acquiring buckets $B_1, B_2, \ldots, B_k$ corresponding to hash values from a hash ring
6: **if** $\exists B_i(1 \leq i \leq k)$ contains an empty entry e **then**
7: $e \leftarrow f$
8: **return** true
9: **end if**
10: select a fingerprint e from any bucket $B_i(1 \leq i \leq k, B_i \neq last)$
11: $f \leftrightarrow e$
12: $last \leftrightarrow B_i$
13: **end for**
14: **return** false

Algorithm 2 Query(x)

Input. x, k
Output. true/false
1: $f \leftarrow h_0(x)$
2: calculate hash values $h_1(f), h_2(f), \ldots, h_k(f)$
3: acquiring buckets $B_1, B_2, \ldots, B_k$ corresponding to hash values from a hash ring
4: **if** $\exists B_i(1 \leq i \leq k)$ and $f \in B_i$ **then**
5: **return** true
6: **end if**
7: **return** false

Algorithm 3 Delete (x)

Input. x, k
Output. true/false
1: $f \leftarrow h_0(x)$
2: calculate hash values $h_1(f), h_2(f), \ldots, h_k(f)$
3: obtain the bucket $B_1, B_2, \ldots, B_k$
4: **if** $\exists B_i(1 \leq i \leq k)$ and $f \in B_i$ **then**.
5: remove f form B_i
6: **return** true
7: **end if**
8: **return** false

3.3. Paillier Homomorphic Encryption

Based on the Paillier homomorphic encryption, this paper realizes the generation of the public and private key pair of the HPP, the initial HomoNym generation, the HomoNym update, and the recovery of the real identity of the revoked vehicle. The Paillier homomorphic encryption consists of three steps: (1) PKGA() (Paillier Key Generation Algorithm), (2) Paillier homomorphic encryption Paillier_Enc_PK$_P\{msg\}$, and (3) Paillier homomorphic decryption Paillier_Dec_SK$_P\{C\}$.

- PKGA()
  (1) First, choose two large prime numbers, p_1 and p_2, for which gcd($(p_1-1)(p_2-1)$) = 1, and calculate $N = p_1 p_2$.
  (2) Second, define L(x) = $\frac{(x-1)}{N}$, select $g \in Z^*_{N^2}$, where gcd(L(g^λ mod N^2), N) $= 1$.
  (3) Third, calculate μ = (L(g^λ mod N^2))$^{-1}$ mod N, and λ = LCM($p_1 - 1, p_2 - 1$), where LCM represents the least common multiple.
  (4) Finally, the public key is (N, g), and the private key is (λ, μ).
- Paillier_Enc_PK$_P\{msg\}$
  The Paillier homomorphic encryption Paillier_Enc_PK$_P\{msg\}$ is represented in Equation (1). For any plaintext message $msg \in Z_N$, a random number $r \in Z^*_N$ is chosen to calculate the ciphertext C.

$$C = E(msg, r) = g^{msg} r^N \ mod \ N^2 \tag{1}$$

- Paillier_Dec_SK$_P\{C\}$
  The Paillier homomorphic decryption Paillier_Dec_SK$_P\{C\}$ is represented in Equation (2). For the ciphertext $C \in Z^*_{N^2}$, the plaintext message msg is computed as follows:

$$msg = D(C, \lambda) = \frac{L(C^\lambda \ mod \ N^2)}{g^\lambda \ mod \ N^2} \ mod \ N = L(C^\lambda \ mod \ N^2) \cdot \mu \ mod \ N \tag{2}$$

The Paillier homomorphic encryption conforms to the property of additive homomorphism, for any plaintext $msg_1, msg_2, \in Z_N$, and any $r_1, r_2 \in Z^*_N$, corresponding to the ciphertext $C_1 = E[msg_1, r_1], C_2 = E[msg_2, C_2]$, satisfying Equation (3).

$$C_1 \cdot C_2 = E[msg_1, r_1] \cdot E[msg_2, r_2] = g^{msg_1 + msg_2} \cdot (r_1 \cdot r_2)^N \ mod \ N^2 \tag{3}$$

Equation (4) is used to decrypt the ciphertext.

$$D[C_1 \cdot C_2] = D[E[msg_1, r_1] \cdot E[msg_2, r_2] \ mod \ N^2] = msg_1 + msg_2 \ mod \ N \tag{4}$$

That is, we obtain $C_1 \cdot C_2 = msg_1 + msg_2$. The multiplication of ciphertexts is equal to the addition of plaintexts.

3.4. Elliptic Curve Digital Signature Algorithm

In this paper, the vehicle generates LocNyms and local private keys based on the ECDSA [3] Key Generation Algorithm (EKGA()); EKGA() is shown in Algorithm 4. The vehicle realizes identity authentication based on the ECDSA Algorithm ECDSA_Sign_SK$_X\{msg\}$ to avoid impersonation attacks. ECDSA_Sign_SK$_X\{msg\}$ is shown in Algorithm 5. The ECDSA Verification Algorithm ECDSA_Verify_PK$_X\{msg\}$ is shown in Algorithm 6.

Firstly, the global parameters used in Algorithms 4–6 are defined; q is a random prime number, an elliptic curve is defined by equation $y^2 = x^3 + ax + b$, $a, b \in Z_q$; P is the base point satisfying the elliptic curve equality, denoted by $P = (x_g, y_g)$; n is the order of point P; that is, n is the smallest positive integer satisfying $nP = 0$.

Algorithm 4 EKGA()

Input. $P, n, q, a, b,$
Output. Q, d
1: select a random integer $d \in [1, n-1]$
2: calculate $Q = dP$, obtain a solution point Q on the curve $E_q(a, b)$
3: return vehicle's local pseudonym is Q, local private key is d

Algorithm 5 ECDSA_Sign_SK$_X\{msg\}$

Input. $n, P, msg, d,$
Output. (r, s)

1: select a random integer or a pseudo-random integer $k \in [1, n-1]$
2: calculate the solution point of the curve $P = (x, y) = kP$, and $r = x \ mod \ n$. If $r = 0$, then skip to step 1
3: calculate $t = k^{-1} \ mod \ n$
4: calculate $e = H(msg)$, where H is the hash function SHA-256, which produces a 256-bit hash value
5: calculate $s = k^{-1}(e + dr) \ mod \ n$. If $s = 0$, then skip to step 1
6: return the signature of msg is (r, s)

Algorithm 6 ECDSA_Verify_PK$_X\{msg\}$

Input. $(r, s), n, msg, P, Q$
Output. accept/reject (r, s)

1: check whether r and s are integers between 1 and $n-1$
2: calculate a 256-bit Hash value $e = H(msg)$ using hash function SHA-256
3: calculate $w = s^{-1} \ mod \ n$
4: calculate $u_1 = ew$ and $u_2 = rw$
5: calculate solution point $X = (x_1, y_1) = u_1 P + u_2 Q$
6: if $X = 0$, reject (r, s), otherwise compute $v = x_1 \ mod \ n$
7: if and only if $v = r$, accept (r, s)

4. Pseudonym Management Scheme Based on Paillier Homomorphic Encryption and the CF

In this paper, it is agreed that TriNymAuth satisfies the following assumptions: (1) synchronize the clocks of all entities in VANETs, (2) the HPP is a fully trusted third party that will not be attacked, and (3) RSUs are honest but curious roadside units. TriNymAuth uses the symmetric encryption and public key encryption algorithm used in the WAVE standard protocol, IEEE Std 1609.2-2016 [26]. The symmetric encryption algorithm adopts the advanced encryption standard of a 128-bit key in CCM mode, that is, Advanced Encryption Standard-Counterwith Cipherlockchaining Message (AES-CCM) [27], and the asymmetric encryption algorithm is a P-256 Elliptic Curve Integrate Encrypt Scheme (ECIES) [28]. Moreover, in V2V authentication, in order to prevent impersonation attacks, the vehicle uses the ECDSA algorithm to sign and complete vehicle identity authentication.

The pseudonym life cycle of TriNymAuth is shown in Figure 2, including the following stages:

- HomoNym issuance. The HPP issues the HomoNym of the vehicle and sends it to the vehicle through a secure channel. After that, the vehicle and the HPP update the pseudonym synchronously, and the HPP calls the insert algorithm of the CF to save the HomoNym in the filter.
- Two-stage HomoNym enrollment. The two-stage HomoNym enrollment consists of: HomoNym verification and LocNym advertisement. Firstly, after the fresh vehicle enters the RSU area, it applies for HomoNym enrollment at the RSU. The RSU forwards the HomoNym to the HPP, and the HPP queries the CF to verify the validity of the HomoNym. After that, the RSU broadcasts the LList in the jurisdiction.
- HomoNym revocation. When there is a malicious vehicle, the vehicle sends the reporting message to the RSU, and the RSU verifies and forwards the HomoNym of the reported vehicle to the HPP, and the HPP performs a revocation operation on the malicious vehicle. In addition, the CF update service is used to update the HomoNym status.
- VirNym exchange. Before V2V communication, vehicles generated VList and exchanged it.

- VirNym usage. For V2V communication, vehicles use VirNym to achieve vehicle identity authentication inside VANETs.

Moreover, the main notations used in TriNymAuth and their descriptions are given in Table 1.

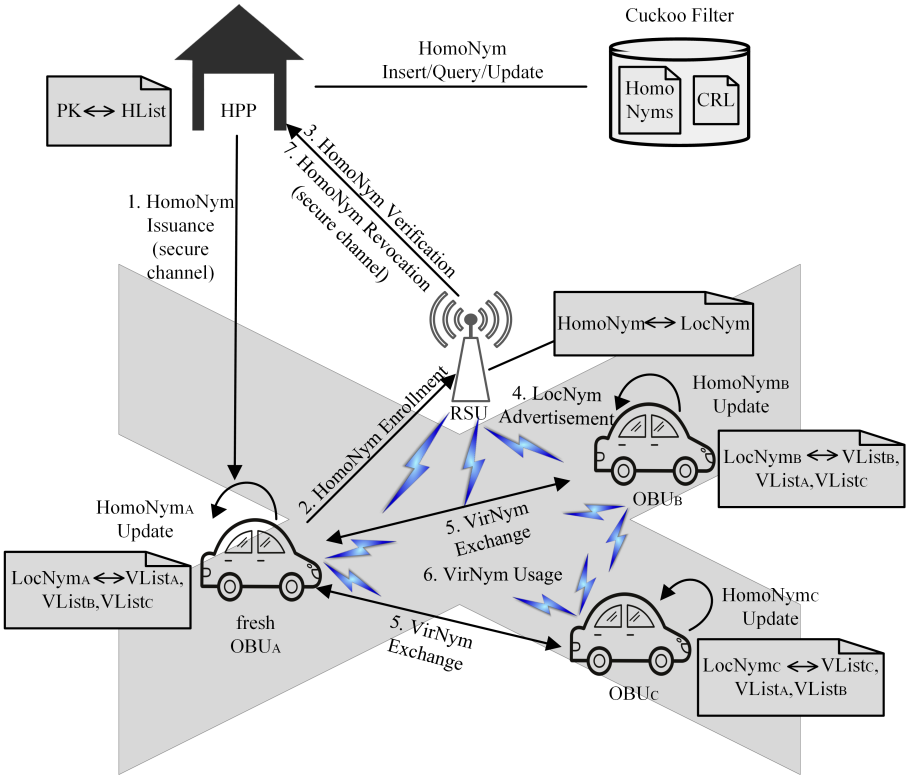

Figure 2. Pseudonym life cycle.

Table 1. Scheme notation and its description.

Notation	Description
$HList_A = \{HomoNym_A\}$	A hash list of HomoNyms of OBU_A
$VList_A = \{VirNym_A\}$	A hash list of VirNyms of OBU_A
$LList = \{LocNym\}$	A hash list of LocNyms of OBU under the jurisdiction of an RSU
r_{A0}	The first random seed of OBU_A
sym_{XY}	The symmetric key between X and Y
AES_Enc_sym$_{XY}\{msg\}$	Using the symmetric key sym_{XY} to encrypt message msg
AES_Dec_sym$_{XY}\{C\}$	Using the symmetric key sym_{XY} to decrypt ciphertext C
ECIES_Enc_PK$_X\{msg\}$	Using the public key PK_X of X to encrypt message msg
ECIES_Dec_SK$_X\{C\}$	Using the private key SK_X of X to decrypt ciphertext C
Paillier_Enc_PK$_X\{msg\}$	Using the public key to Paillier homomorphic encrypt message msg
Paillier_Dec_SK$_X\{C\}$	Using the private key SK_X of X to Paillier homomorphic decrypt ciphertext C
ECDSA_Sign_$\hat{SK}_X$/SK$_X\{msg\}$	Using the local private key $\hat{SK}_X$ of OBU_X or the manufacturer private key SK_X of OBU_X to sign msg with ECDSA
ECDSA_Verify_LocNym$_X$/PK$_X\{msg\}$	Using the local public key $LocNym_X$ of OBU_X or the manufacturer public key PK_X of OBU_X to do ECDSA signature verification on msg

4.1. System Initialization

In this phase, the HPP generates basic system parameters and issues public and private keys for the RSU through secure channel.

- HPP selects the cryptographic hash function h.

- HPP constructs a multiplicative cyclic group G of prime order l and generates a random public prime z and its public generator ϵ. The HPP constructs Paillier homomorphic encryption Paillier_Enc_PK$_P\{msg\}$.
- Finally, the HPP publishes the public system parameters, params = $\{G, h, l, z, \epsilon\}$.
- HPP Key generation. By using PKGA(), the HPP generates its own public key $PK_P = (N, g)$ and private key $SK_P = (\lambda, u)$.
- RSU Key generation. The HPP chooses a random number $SK_R \in {}_RZ_q$ as the private key of the RSU and computes $PK_R = h^{SK_R} \in G$ as the public key of the RSU. Based on the secure channel between the HPP and the RSU, the HPP encrypts PK_R and SK_R with the symmetric key sym_{HR} between them and transmits the ciphertext to the RSU. After receiving it, the RSU decrypts PK_R and SK_R using sym_{HR}.

4.2. Homomorphic Pseudonym Generation and SELF-Update Protocol Based on Paillier Homomorphic Encryption

In this phase, based on the secure channel between the HPP and OBU$_A$, the HPP uses symmetric key sym_{HA} to issue the first random seed, update cycle, and initial HomoNym to OBU$_A$ and updates the HomoNyms synchronously with OBU$_A$.

- Initial information issuance:
  1. The first random seed r_{A0} of the OBU$_A$.The HPP randomly selects the first random seed r_{A0} to the OBU$_A$, which is used to update the HomoNym simultaneously between the OBU$_A$ and the HPP.
  2. Update cycle. The HPP specifies the update cycle T_U [2] for HomoNyms. When the OBU$_A$ receives the initial HomoNym issued by the HPP, it will self-update HomoNym strictly according to the update cycle T_U specified by the HPP.
  3. Initial HomoNym. The HPP employs Paillier homomorphic encryption Paillier_Enc_PK$_P\{msg\}$ to generate the OBU$_A$'s initial HomoNym, along with its own public key PK_P and randomized seed r_{A0}. The calculation formula of the initial HomoNym is represented in Equation (5):

$$\begin{aligned} HomoNym_{A0} &= E(ID_A, r_{A0}) \\ &= [g \ mod \ N^2]^{ID_A} (r_{A0})^N \ mod \ N^2 \end{aligned} \tag{5}$$

  4. Finally, based on the AES-CCM symmetric encryption algorithm, the HPP encrypts $\{r_{A0}, T_U, HomoNym_{A0}\}$ by using the symmetric key sym_{HA}, then sends AES_Enc_ sym$_{HA}\{r_{A0}, T_U, HomoNym_{A0}\}$ to the OBU$_A$ through a secure channel. OBU$_A$ executes AES_Dec_sym$_{HA}\{r_{A0}, T_U, HomoNym_{A0}\}$ to decrypt and obtain the initial information issued by the HPP and saves in its own TPD.
- HomoNyms self-update
  Based on the update cycle, the vehicles self-update the shared random number and HomoNym synchronously with the HPP, and the calculation formula for the random number and HomoNym is introduced as follows:
  1. Based on the update cycle T_U, OBU$_A$ self-updates the shared random number synchronously with the HPP according to Equation (6):

$$\begin{aligned} r_{Ai} &= E(r_{A(i-1)}, sym_{HA}) \\ &= (HomoNym_{A(i-1)})^{r_A(i-1)} \cdot (sym_{HA})^N \ mod \ N^2 \end{aligned} \tag{6}$$

2. Based on the update cycle T_U, OBU_A self-updates the $HomoNym_{Ai}$ synchronously with the HPP according to Equation (7):

$$\begin{aligned} HomoNym_{Ai} &= HomoNym_{A(i-1)} \times E(0, r_{Ai}) \\ &= E(ID_A, r_{A(i-1)}) \times E(0, r_{Ai}) \\ &= E(ID_A + 0, r') \end{aligned} \tag{7}$$

After the update is completed, the HPP calls Insert($HomoNym_{Ai}$) (as shown in Algorithm 1) of the CF to insert the $HomoNym_{Ai}$ and the corresponding update cycle T_U into the CF.

With the continuous update of HomoNyms, a large number of HomoNyms expire, and deleting expired HomoNyms in real time is conducive to releasing the useful space of the filter reasonably. Because the CF can be deleted, when a HomoNym expires, the expired entry in the filter is backed up first, and then the HPP calls Delete($HomoNym$) (as shown in Algorithm 3) of the CF to delete the expired HomoNyms.

4.3. Two-Stage Homomorphic Pseudonym Enrollment Protocol Based on the CF

After the vehicle enters VANETs, the initial HomoNym is updated first. The vehicle generates its own LocNym and local private key based on ECDSA Key Generation Algorithm EKGA(), and then the two-stage homomorphic pseudonym enrollment is carried out. The following takes the OBU_A as an example to introduce the whole process.

First, OBU_A inputs sym_{HA} to its TPD, and then the TPD checks whether sym_{HA} matches the symmetric key it has stored. If it does, OBU starts successfully, OBU_A self-updates the initial homomorphic pseudonym $HomoNym_{A0}$, and at the same time OBU_A generates its own $LocNym_A$ and $\hat{SK}_A$ based on the ECDSA Key Generation Algorithm EKGA() (as shown in Algorithm 4). Then, the homomorphic pseudonym verification protocol and the local pseudonym advertisement protocol are executed. In this stage, since OBU_A cannot generate the HomoNyms of other vehicles, it does not need to sign its own HomoNym and LocNym during the homomorphic pseudonym enrollment stage, which reduces the cost of signing and signature verification. In addition, OBU_A updates $HomoNym_A$ and $LocNym_A$ periodically, and after each update, OBU_A needs to re-enroll.

4.3.1. Homomorphic Pseudonym Verification Protocol Based on the CF

When a fresh vehicle enters an RSU's jurisdiction, or when the vehicle's HomoNym and LocNym are updated, the vehicle applies for enrollment using its own HomoNym. The vehicle sends the encrypted HomoNym and LocNym to the RSU. The RSU decrypts it and forwards HomoNym to the HPP, which verifies the validity of the HomoNym through querying the CF and returns the verification result to the RSU. The results are not "expired" or "revoked" if they are valid. The RSU stores the vehicle's LocNym in *LList* as well as the table of correspondence between LocNym and HomoNym locally. The following takes the OBU_A as an example to introduce the whole process:

1. After entering the jurisdiction of the RSU or updating $HomoNym_A$ and $LocNym_A$, based on the ECIES public key encryption algorithm, OBU_A uses the PK_R of the RSU to encrypt $HomoNym_A$ and $LocNym_A$ and sends $\mathrm{ECIES_Enc_PK}_R\{HomoNym_A, LocNym_A\}$ to the RSU. After receiving it, the RSU decrypts $\mathrm{ECIES_Dec_SK}_R\{HomoNym_A, LocNym_A\}$ to obtain $HomoNym_A$ and $LocNym_A$.
2. Based on the AES-CCM symmetric encryption algorithm, the RSU encrypts its $HomoNym_A$ with the symmetric key sym_{HR} and sends the encrypted $\mathrm{AES_Enc_sym}_{HR}\{HomoNym_A\}$ to the HPP. The HPP uses sym_{HR} to decrypt $\mathrm{AES_Dec_sym}_{HR}\{HomoNym_A\}$ to obtain $HomoNym_A$. Then the HPP calls the CF's Query($HomoNym_A$) (as shown in Algorithm 2) to query $HomoNym_A$. If found, it proves that the $HomoNym_A$ is valid, not "expired" or "revoked". The HPP returns the verification result to the RSU. The RSU keeps the LocNyma of OBU_A in *LList* and the table of correspondence between $LocNym_A$ and

$HomoNym_A$ locally. If the $HomoNym_A$ cannot be found or is expired or revoked, it will be discarded.

Figure 3 shows the message transmitted by the VANETs entities during the homomorphic pseudonym enrollment phase, which is steps 2 and 3 in Figure 2. Due to the fact that vehicles cannot generate HomoNyms for other vehicles, OBU_A does not need to sign its $HomoNym_A$ and $LocNym_A$, but instead must only transmit ECIES_Enc_$PK_R\{HomoNym_A, LocNym_A\}$ and to the RSU. After the RSU decrypts and obtains $HomoNym_A$ and $LocNym_A$, $HomoNym_A$ will be encrypted to AES_Enc_$sym_{HR}\{HomoNym_A\}$ and transmitted to the HPP, which reduces the signature overhead of OBU_A and the verification cost of the HPP in this process.

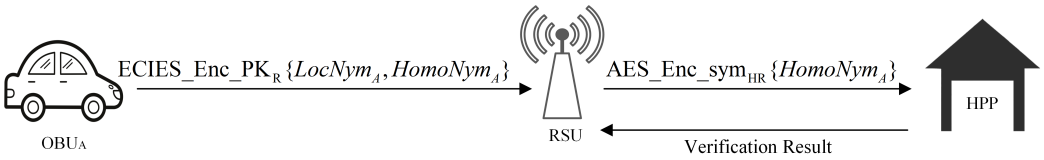

Figure 3. The messages transmitted between entities during the HomoNym enrollment and verification phase.

4.3.2. Local Pseudonym Advertisement Protocol

Based on the LocNyms sent by the vehicles during the HomoNym enrollment phase, the RSU integrates LocNyms and generates a LocNym hash list of vehicles, which is broadcast to the vehicles in the jurisdiction. The following takes the OBU_A as an example to introduce the whole process.

Based on the LocNyms sent by the vehicles during the HomoNym enrollment phase, the RSU integrates the LocNyms sent by the vehicles during the HomoNym enrollment phase and generates a LocNym hash list *LList* of all vehicles within the jurisdiction before broadcasting *LList*. In addition, when a fresh vehicle enters an RSU's jurisdiction, or when the vehicle's HomoNym and LocNym are updated, the RSU updates the *LList* and broadcasts it. After receiving it, vehicles update the *LList* stored locally in a time-priority queue manner.

4.4. Homomorphic Pseudonym Revocation Protocol Based on Paillier Homomorphic Encryption and the CF

This section illustrates the process of revoking any vehicle that broadcasts false information. Since vehicles communicate with each other using VirNyms, when there is a vehicle broadcasting false information, other vehicles will report the LocNym information corresponding to the VirNym to the RSU, and the RSU will look up the corresponding HomoNym of LocNym in the relation table of HomoNym and LocNym stored locally. Then, the RSU will report the HomoNym to the HPP. The HPP calls the HomoNym state management service of the CF to update the pseudonym state of the malicious vehicle in the revocation period to "revoked". Finally, the HPP uses its own private key SK_P to calculate the vehicle's real identity based on Paillier homomorphic decryption and carries out the corresponding punishment. The following takes the OBU_A and OBU_B as examples to introduce the whole process:

1. When OBU_B discovers illegal behavior by OBU_A, OBU_B uses the PK_R of the RSU to encrypt the evidence of violation and the $LocNym_A$, $LocNym_B$ of OBU_A and OBU_B and sends the ECIES_Enc_$PK_R\{evidence, LocNym_A, LocNym_B\}$ to the RSU.
2. The RSU decrypts ECIES_Dec_$SK_R\{evidence, LocNym_A, LocNym_B\}$ and obtains the report message, then looks up $LocNym_A$ and $LocNym_B$ in the local *LList*. If the LocNyms are discovered, the identities of OBU_A and OBU_B are established.
3. The RSU finds the $HomoNym_A$ of the OBU_A corresponding to $LocNym_A$ in the relation table of HomoNym and LocNym stored locally. The RSU encrypts $HomoNym_A$ with the symmetric key sym_{HR} and sends AES_Enc_$sym_{HR}\{HomoNym_A\}$ to the HPP,

which decrypts AES_Dec_sym$_{HR}\{HomoNym_A\}$ and obtains $HomoNym_A$ using the AES-CCM symmetric encryption algorithm.

4. According to the $HomoNym_A$ of the reported OBU$_A$, the HPP calculates its true identity ID_A and all HomoNyms in the revocation period t and calls the CF's pseudonym state change service to set the status of all HomoNyms in the revocation period to "revoked".
5. According to the additive property of homomorphic encryption, the HPP can obtain the real identity ID_A of OBU$_A$ by using Paillier homomorphic decryption Paillier_Dec _SK$_P\{C\}$, which is represented in Equation (8).

$$D(HomoNym_{Ai}) = D(E(ID_A + 0, r_{Ai})) \tag{8}$$

4.5. V2V Authentication Protocol Based on Virtual Pseudonyms Exchange and Usage

4.5.1. V2V Authentication Protocol Based on Virtual Pseudonyms Exchange

Before communication, the vehicle randomly selects a set of 20-byte random numbers as its own VList based on the random number generator and then exchanges the VList for subsequent mutual identity authentication. Because VirNym is a random number, it effectively prevents impersonation attacks. In V2V communication, the vehicle uses its own local private key to sign. After receiving the signature, other vehicles verify the signature using the vehicle's LocNym to achieve identity authentication. The following takes OBU$_A$ and OBU$_B$ as examples to introduce the whole process of VirNyms exchange:

1. Before the first communication between OBU$_A$ and OBU$_B$, based on the ECDSA digital signature algorithm, by using OBU$_A$'s local private key $\hat{SK}_A$, OBU$_A$ executes ECDSA_Sign_$\hat{SK}_A\{VList_A\}$ (as shown in Algorithm 5), the VirNym hash list $VList_A$ of OBU$_A$ is signed as $[VList_A]_{\hat{SK}_A}$, and then the signature $[VList_A]_{\hat{SK}_A}$ is sent to OBU$_B$.
2. After receiving it, OBU$_B$ uses the $LocNym_A$ of OBU$_A$ to verify the signature $[VList_A]_{\hat{SK}_A}$, executes ECDSA_Verify_LocNym$_A\{VList_A\}$ (as shown in Algorithm 6) to obtain the VirNym hash list $VList_A$ of OBU$_A$, and saves it locally.
3. Similarly, by using OBU$_B$'s local private key $\hat{SK}_A$, OBU$_B$ executes ECDSA_Sign_$\hat{SK}_B$ $\{VList_B\}$ (as shown in Algorithm 5), the VirNym hash list $VList_B$ of OBU$_B$ is signed as $[VList_B]_{\hat{SK}_B}$, and then the signature is sent to OBU$_A$.
4. After receiving it, OBU$_A$ uses the $LocNym_B$ of OBU$_B$ to verify the signature $[VList_B]_{\hat{SK}_B}$, executes ECDSA_Verify_LocNym$_B\{VList_B\}$ (as shown in Algorithm 6) to obtain the VirNym hash list $VList_B$ of OBU$_B$ and saves it locally.

Figure 4 illustrates the VirNym hash list $VList$ transmitted between vehicles during the VirNym exchange phase, which is step 5 in Figure 2.

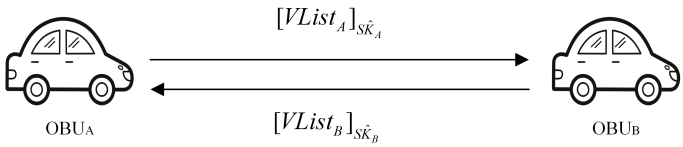

Figure 4. Virtual pseudonyms exchange phase.

4.5.2. V2V Authentication Protocol Based on Virtual Pseudonyms Usage

When communicating, in order to prevent impersonation attacks, vehicles need to use the ECDSA algorithm to sign VirNyms. The following takes the OBU$_A$ and OBU$_B$ as examples to introduce the whole process:

1. Based on the ECDSA algorithm, OBU$_A$ uses its own local private key $\hat{SK}_A$ to sign $VirNym_A$, and after ECDSA_Sign_$\hat{SK}_A\{VirNym_A\}$ (as shown in Algorithm 5), sends the signature $[VirNym_A]_{\hat{SK}_A}$ and message msg_A to OBU$_B$.
2. After receiving them, OBU$_B$ locally queries $VList$ to find LocNym that can verify the signature, and then OBU$_B$ uses the queried $LocNym_A$ that corresponds to $VList_A$ to

verify the signature $[VirNym_A]_{\hat{SK}_A}$, executes ECDSA_Verify_LocNym$_A\{VirNym_A\}$ (as shown in Algorithm 6), and obtains $VirNym_A$. Since both $LocNym_A$ and $VirNym_A$ are from OBU$_A$, it shows that OBU$_A$ did not carry out an impersonation attack.

3. Similarly, based on the ECDSA algorithm, OBU$_B$ uses its own local private key $\hat{SK}_B$ to sign $VirNym_B$, and after ECDSA_Sign_$\hat{SK}_B\{VirNym_B\}$ (as shown in Algorithm 5), sends the signature $[VirNym_B]_{\hat{SK}_B}$ and message msg_B to OBU$_A$.
4. After receiving them, OBU$_A$ queries $VList$ locally to find LocNym that can verify the signature, and then OBU$_A$ uses the queried $LocNym_B$ that corresponds to $VList_B$ to verify the signature $[VirNym_B]_{\hat{SK}_B}$, executes ECDSA_Verify_LocNym$_B\{VirNym_B\}$ (as shown in Algorithm 6), and obtains $VirNym_B$. Since both $LocNym_B$ and $VirNym_B$ are from OBU$_B$, it shows that OBU$_B$ did not carry out an impersonation attack. OBU$_A$ and OBU$_B$ complete the communication.

5. Security and Privacy Analysis

The security and privacy analysis of TriNymAuth is performed in this section, and it is demonstrated that TriNymAuth meets almost all of the security and privacy requirements in VANETs. Table 2 compares the TriNymAuth to the related work safety. Comparison results show that TriNymAuth has great superiority.

Table 2. Security and privacy comparison.

Security	SPECS [5]	b-SPECS+ [7]	SPACF [13]	VPPCS [6]	TriNymAuth
Resistance to impersonation attack	-	✓	-	✓	✓
Identity privacy protection	✓	✓	✓	✓	✓
Traceability	✓	✓	✓	✓	✓
Unlinkability	-	-	✓	✓	✓

5.1. Security Analysis

In order to achieve secure and efficient vehicle identity authentication, triple pseudonym authentication is used to realize vehicle identity authentication hierarchically and regionally, which decouples internal and external vehicle identities in VANETs. The integrity and non-repudiation of TriNymAuth messages are thus guaranteed, and message tampering attacks are avoided. Meanwhile, the timestamp is included in the message sent by the vehicle, which avoids the replay attack.

In the following, the prevention measures when different attackers carry out impersonation attacks are analyzed according to the attack model shown in Figure 5.

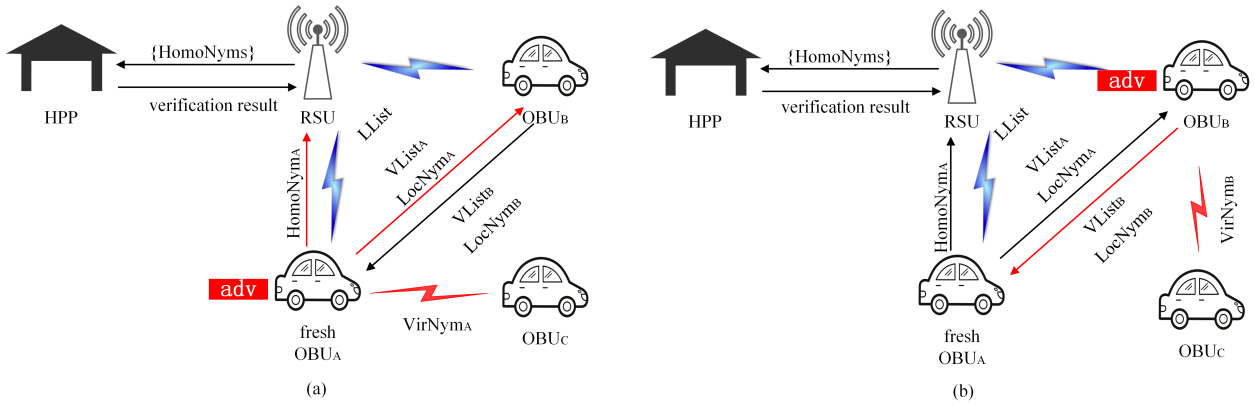

Figure 5. Attack model. The attacker in (**a**) is a fresh OBU$_A$ that joins VANETs. The attacker in (**b**) is the existing OBU$_B$ in VANETs.

Firstly, when a fresh OBU$_A$ joining the network is the attacker, the impersonation attack of the attacker in different life cycle stages of the pseudonym is analyzed as follows:

1. In the two-stage HomoNym enrollment phase. The adversary OBU_A generates an invalid HomoNym and sends it to the RSU for HomoNym enrollment. The RSU receives it and forwards it to the HPP. The HPP verifies the HomoNym of OBU_A. By comparing the HomoNym with the HomoNym stored locally in the HPP, it can be known that the HomoNym used by the OBU_A is invalid, so as to effectively avoid the possible impersonation attack launched by a fresh vehicle in the two-stage HomoNym enrollment phase.
2. In the VirNym exchange phase. The VirNym impersonation attack cannot be carried out in this phase because the OBU_A is a fresh vehicle and cannot obtain the VirNym of other vehicles. In addition, TriNymAuth can effectively avoid the impersonation attack launched by a fresh vehicle in the VirNym exchange phase because the vehicle generates its own local private key based on the EKGA () algorithm, which cannot be obtained by other vehicles through monitoring.
3. In the VirNym usage phase. OBU_A obtains a series of VirNyms of OBU_B in the VirNym exchange phase. If OBU_A wants to pretend to be OBU_B, in the VirNym usage phase, OBU_A signs $VirNym_B$ using its own local private key $\hat{SK}_A$ and sends the signature and message to OBU_C. Following receipt of the signature, OBU_C searches $VList$ locally for LocNym that can verify the signature, and then OBU_C uses the queried $LocNymA$ to verify the signature and obtain $VirNym_B$. OBU_A's impersonation attack failed because OBU_C discovered that $VirNym_B$ was not from OBU_A. So, TriNymAuth can effectively avoid the possible impersonation attack launched by a fresh vehicle joining VANETs during the VirNym usage phase.

Furthermore, when the OBU_B is the attacker, the impersonation attack is examined at various stages of the pseudonym's life cycle.

1. In the VirNym exchange phase. The adversary OBU_B sends the $VList_C$ of OBU_C to OBU_A as its own VirNym hash list. However, the adversary OBU_B cannot obtain the local private key $\hat{SK}_C$ of OBU_C and can only sign the $VList_C$ using its own local private key $\hat{SK}_B$, so the impersonation of OBU_C by the adversary OBU_B cannot be successfully implemented.
2. In the VirNym usage phase. The adversary OBU_B obtains the $VList_C$ of OBU_C in the VirNym exchange phase. In the VirNym usage phase, OBU_B signs $VirNym_C$ using its own local private key $\hat{SK}_B$ and sends the signature and message to OBU_A. After receiving them, OBU_A queries $VList$ locally to find LocNym that can verify the signature, and then OBU_A uses the queried $LocNymB$ to verify the signature and obtain $VirNym_C$. OBUB's impersonation attack failed because OBU_A discovered that $VirNym_C$ is not from OBU_B. Therefore, TriNymAuth can effectively avoid the possible impersonation attacks launched by vehicles in the VirNym usage phase.

5.2. Privacy Analysis

TriNymAuth's performance in achieving vehicle identity privacy protection, unlinkability, and traceability is analyzed below.

- Identity privacy protection

To begin with, only the HPP is aware of the connection between HomoNym and real identity in terms of identity privacy protection. Based on Paillier homomorphic encryption, the HPP can use its private key to decrypt the HomoNym to obtain the vehicle's real identity. Since vehicles use VirNyms for communication, HomoNyms are not involved in the communication process. When the adversary is a vehicle, it cannot obtain the vehicle's HomoNyms through monitoring, so the vehicle's real identity cannot be further obtained. Therefore, TriNymAuth ensures the vehicle's identity privacy protection.

- Unlinkability

When the adversary is a vehicle, the link between HomoNyms cannot be implemented because the adversary cannot obtain the HomoNyms of other vehicles. The adversary can

link a finite number of VirNyms and LocNyms of other vehicles, and when it comes to the update cycle T_U or the vehicle drives across the RSU regions, the $VList$ and LocNym is updated. So, LocNym and VirNym achieve conditional unlinkability. Moreover, the adversary cannot associate the LocNym or VirNym of other vehicles with their HomNym and real identity ID.

- Traceability

Vehicle OBU_A uses $VirNym_A$ for V2V communication, and when the adversary is OBU_A, the vehicle can obtain the $LocNym_A$ of the OBU_A by querying the locally stored $VList$. Because the RSU saves the correspondence between LocNyms and HomoNyms of the OBU_A, when the OBU_A broadcasts false information, the RSU can obtain the $HomoNym_A$ of OBU_A by its $LocNym_A$ and will forward it to the HPP. The HPP decrypts $HomoNym_A$ using its own private key SK_P, and reveals the real identity ID_A of the OBU_A. Therefore, TriNymAuth guarantees the traceability of vehicle identity.

6. Performance Analysis

Based on the advantages of the CF, we analyze the enrollment cost, computational cost to verify the signature, communication cost in the VirNym exchange phase and usage phase, and the total transmission delay of our scheme. Then, we compare TriNymAuth's performance with SPECS [5], b-SPECS+ [7], SPACF [13], and VPPCS [6].

6.1. Advantage of the CF

The CF adopted in this paper combines a filter-level filter (CCF_F) and a bucket-level filter (CCF_B). Through experimental analysis, the time cost of TriNymAuth in query and insert is as follows:

6.1.1. Query Overhead

The time complexity of CCF_F's query is about $O(k \cdot s \cdot log(m/s))$, while the time complexity of CCF_B's query is about $O(k \cdot log(m)$, the former having one more constant factor s than the latter. Among them, m represents the buckets' total number in the filter, s is the number of I2CF, and k is the number of hash functions. Table 3 compares the query overhead (single message) of CCF_F and CCF_B.

Table 3. The query overhead of CCF_F and CCF_B (ns).

CF	Query Time (ns)
CCF_F	2934
CCF_B	1689

6.1.2. Insert Overhead

Table 4 compares the average insertion time overhead of CCF_F and CCF_B. Since a large number of relocations occur during the insertion of CCF_B, the insertion time cost of CCF_B is large This is because a lot of relocations occur during insertion and may be accompanied by the phenomenon of "data migration" [29]. Data migration occurs when the distance between two buckets on the hash ring is so close that most of the data can only be allocated to one bucket. There are enough empty buckets on the hash ring but the load factor is still too high, which leads to the sharp increase in relocation times. Set max to the upper limit of relocation times, m is the buckets' number, and the insertion time complexity of CCF_B is $O(max \cdot logm)$. The insertion time complexity of CCF_F is $O(max \cdot log(m/s))$, so CCF_F's insertion time is smaller than CCF_B, and the smaller s is, the less time it takes to insert, but the cost is increased query and delete time because they need to traverse all I2CF Consider extreme $s = m$, it is not hard to think the filter will retreat into a linked list or query, and the delete time complexity will increase to a linear level.

Table 4. The average insert overhead of CCF_F and CCF_B(μs).

CF	Insert Time (μs)
CCF_F	2.5
CCF_B	11.7

TriNymAuth uses the CF combining CCF_F and CCF_B, and the actual time cost is between them about 10 μs.

6.1.3. Space Overhead

Figure 6 analyzes the cumulative distribution of space utilization for CCF_F and CCF_B. It can be seen that the space utilization of CCF_F and CCF_B exceeds 80% in about 50% and 63% of the cases, respectively. The CF combined with CCF_F and CCF_B has better elasticity and higher space utilization, which makes it more suitable for dynamic data set representation. In VANETs, improving the space utilization of the HomoNym management system can make the system have more free memory and execute more complex tasks at the same time.

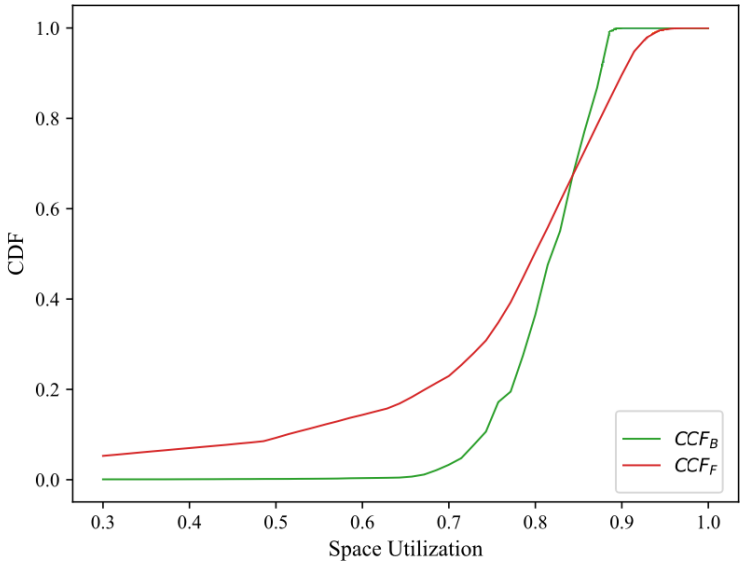

Figure 6. CDF of space utilization of CCF_F and CCF_B under the same workflow.

6.1.4. False Positive Rate

TriNymAuth randomly selects a byte from the HomoNym of the original data set and performs an XOR operation with another randomly generated byte. TriNymAuth replaces the result with the selected byte in the HomoNym, obtains the test data set, and removes the intersection with the original data set. After inserting the original data set's HomoNyms into the CF, the query operation is performed on the HomoNyms in the test data set in the filter. When the fingerprint length is 16 bits or 32 bits, the false positive rates of CCF_F and CCF_B are 0.6204 and 0.0014, respectively.

Compared with the fingerprint length of 16 bits, when the fingerprint length is 32 bits, the false positive rate of CCF_F and CCF_B is reduced to about 0.1%, which is significantly improved. Therefore, the fingerprint length should be greater than or equal to 32 bits in practice.

6.2. Enrollment Cost Analysis and Comparison

TriNymAuth protocols are performed on a simulator, which is written in C++. We suppose that the jurisdiction area of the RSU is a circle with a radius of 1 km, and the

travel range of vehicles through the RSU is 0 to 2 km. The vehicle travels at random speeds ranging from 5 m/s to 40 m/s (20–144 km/h).

In SPECS [5], b-SPECS+ [7], and SPACF [13], the enrollment cost is divided into two parts: (1) First is the initial handshake cost. The initial handshake is performed when a vehicle enters the jurisdiction of a new RSU. Although the initial handshake only needs to be performed once in the whole system, it also incurs a large computational overhead. In this process, the initial handshake cost of a vehicle is made up of one ECC encryption and decryption, one signature, and one signature verification [5,7,13]. Thus, the whole phase's total computation time is as follows:

$$4T_{e\cdot m} + T_h + (2T_{bp} + 2T_{b\cdot pm} + T_{mtp} + T_h) \approx 18.0213 \text{ ms}$$

where $T_{e\cdot m}$ is the Elliptic Curve Cryptography (ECC)-based scale multiplication operation's execution time, T_h is the one-way hash function operation's execution time, T_{bp} is the bilinear pairing operation's execution time, $T_{b\cdot pm}$ is the bilinear pairing-based scale multiplication operation's execution time, T_{mtp} is the bilinear pairing-related MapToPoint hash operation's execution time. The initial handshake cost of the RSU is made up of one ECC decryption and one signature verification [5,7,13]. Thus, the whole phase's total computation time is as follows:

$$2T_{e\cdot m} + (2T_{bp} + 2T_{b\cdot pm} + T_{mtp} + T_h) \approx 17.1372 \text{ ms}$$

The initial handshake cost of the TA is made up of two ECC encryption, one ECC decryption, one signature, and one signature verification [5,7,13]. Thus, the whole phase's total computation time is as follows:

$$6T_{e\cdot m} + T_h + (2T_{bp} + 2T_{b\cdot pm} + T_{mtp} + T_h) \approx 18.9053 \text{ ms.}$$

(2) Second is the periodic handshake cost. For the shared secret with the RSU, a new secret is generated every time the vehicle moves into the region of another RSU. At this time, the vehicle needs to perform encryption and decryption operations with the TA. In this process, the periodic handshake cost of a vehicle is made up of one ECC encryption and decryption, one signature, and one signature verification [5,7,13]. Thus, the whole phase's total computation time is as follows:

$$4T_{e\cdot m} + (2T_{bp} + 2T_{b\cdot pm} + T_{mtp} + T_h) \approx 18.0212 \text{ ms}$$

The periodic handshake cost of the RSU is made up of one ECC decryption and one signature verification [5,7,13]. Thus, the whole phase's total computation time is as follows:

$$2T_{e\cdot m} + (2T_{bp} + 2T_{b\cdot pm} + T_{mtp} + T_h) \approx 17.1372 \text{ ms}$$

The periodic handshake cost of the HPP is made up of two ECC encryption, one ECC decryption, and one signature [5,7,13]. Thus, the whole phase's total computation time is as follows:

$$6T_{e\cdot m} + T_h \approx 2.6521 \text{ ms}$$

In VPPCS [6], the enrollment cost of a vehicle is made up of one ECC encryption and decryption, one signature, and one signature verification [6]. Thus, the whole phase's total computation time is as follows:

$$4T_{e\cdot m} + T_h + (2T_{bp} + 2T_{b\cdot pm} + T_{mtp} + T_h) \approx 18.0213 \text{ ms}$$

The enrollment cost of the RSU is made up of one ECC encryption and decryption, one signature, and one signature verification [6]. Thus, the whole phase's total computation time is as follows:

$$4T_{e\cdot m} + T_h + (2T_{bp} + 2T_{b\cdot pm} + T_{mtp} + T_h) \approx 18.0213 \text{ ms}$$

In the HomoNym enrollment phase of TriNymAuth, the enrollment cost can be divided into the following parts. First, based on the ECIES public key encryption algorithm, the vehicle encrypts the HomoNym and LocNym and forwards them to the RSU. The RSU decrypts and obtains HomoNym and LocNym. Secondly, based on the AES-CCM symmetric encryption algorithm, the RSU encrypts the homonym and forwards it to the HPP, which decrypts it and obtains HomoNym. The HPP verifies the validity of HomoNym by querying the CF. Finally, based on the AES-CCM symmetric encryption, the HPP encrypts the verification result and sends it to the RSU. The experiments show that the AES-CCM symmetric encryption algorithm with a 128-bit key length can encrypt and decrypt a 128-byte message in time $T_{AES}^{enc} = 0.5855$ ms and $T_{AES}^{dec} = 0.5375$ ms, respectively. The enrollment cost of the vehicle is made up of one ECIES encryption in this process. Thus, the whole phase's total computation time is as follows:

$$2T_{e \cdot m} \approx 0.8840 \text{ ms}$$

The enrollment cost of the RSU is made up of one ECIES decryption, one AES symmetric encryption and decryption. As a result, the total computation time for the entire phase is as follows:

$$2T_{e \cdot m} + T_{AES}^{enc} + T_{AES}^{dec} \approx 2.0070 \text{ ms}$$

The enrollment cost of the HPP is made up of one AES symmetric encryption and decryption and one CF query. Thus, the whole phase's total computation time is as follows:

$$T_{AES}^{enc} + T_{AES}^{dec} + 0.5 \times (2.934 + 1.689) \times 10^{-3} \approx 1.1253 \text{ ms}$$

The cost of the HomoNym enrollment phase only includes ECIES and AES-CCM symmetric encryption and decryption costs and the CF query costs, which do not involve the overhead of signature and signature verification. It greatly improves the enrollment efficiency of vehicles and reduces the computational cost.

Table 5 analyzes and compares the specific overhead of our scheme, TriNymAuth, and SPECS [5], b-SPECS+ [7], SPACF [13], and VPPCS [6] in the enrollment phase.

Table 5. Enrollment cost analysis and comparison (ms).

Schemes	Phase	Entity	Enrollment Cost (ms)
SPECS [5] b-SPECS+ [7] SPACF [13]	Initial handshake	OBU	18.0213
		RSU	17.1372
		TA	18.9053
	periodic handshake	OBU	18.0212
		RSU	17.1372
		TA	2.6521
VPPCS [6]	periodic enrollment	OBU	18.0213
		RSU	18.0213
TriNymAuth	periodic enrollment	OBU	0.8840
		RSU	2.0070
		HPP	1.1253

6.3. Computational Cost Analysis and Comparison

The traditional anonymous identity authentication scheme uses complex mathematical calculations (such as bilinear pairing and so on) to achieve identity authentication, which has a large computational overhead. In this paper, vehicles use ECDSA-based signatures to achieve identity authentication, which greatly reduces the overhead.

SPECS [5] and b-SPECS+ [7] are established on cryptographic operations based on bilinear pairing; however, SPACF [13], VPPCS [6], and TriNymAuth are established on ECC-based cryptographic operations.

In SPECS [5] and b-SPECS+ [7], the computational overhead of single message verification is made up of one one-way hash function operation, one bilinear pairing-related

MapToPoint hash function operation, one bilinear pairing-related point addition operation, two bilinear pairing operations, and two bilinear pairing-related scalar multiplication operations. Thus, the whole phase's total computation time is as follows:

$$T_h + T_{mtp} + T_{bp \cdot a} + 2T_{bp} + 2T_{b \cdot pm} \approx 16.2532 \text{ ms}$$

where $T_{bp \cdot a}$ is the bilinear pairing-based point addition operation's execution time.

In SPACF [13], the computational overhead of single message verification is made up of one one-way hash function operation, one ECC-related point addition operation, and two ECC-related scalar multiplication operations. Thus, the whole phase's total computation time is as follows:

$$T_h + T_{e.a} + 2T_{e.m} \approx 0.8859 \text{ ms}$$

In VPPCS [6], the computational overhead of single message verification is made up of one one-way hash function operations, one-point addition operations, and two ECC-related scalar multiplication operations. Thus, the whole phase's total computation time is as follows:

$$T_h + T_{e.sm} + 2T_{e.m} \approx 0.8979 \text{ ms}$$

In our scheme, TriNymAuth, the computational overhead of single message verification is made up of two double one-point multiplications over an elliptic curve. Thus, the whole phase's total computation time is as follows:

$$2 \times (2T_{mul}) \approx 1.5600 \text{ ms}$$

where T_{mul} is the one-point multiplication's execution time.

6.4. Communication Cost Analysis and Comparison

This section analyzes the communication cost of TriNymAuth. Since vehicles need to carry out the VirNym exchange phase before communication, the communication cost of TriNymAuth is divided into two parts: (1) the communication cost of the VirNym exchange, and (2) the communication cost of a single beacon. The comparison between TriNymAuth and SPECS [5], b-SPECS+ [7], SPACF [13], and VPPCS [6] is also given when analyzing the communication overhead of a single beacon.

6.4.1. VirNym Exchange Communication Cost

Different from other anonymous authentication schemes, in TriNymAuth, OBU_A needs to exchange $VList$ through the VirNym exchange phase before communication, which is used to realize the identity authentication during the VirNym use phase. TriNymAuth generates random numbers for anti-collision based on SHA-1, and the number of bits of VirNyms needs to be greater than or equal to 20 bytes. In the VirNym exchange stage, the content of OBU_A broadcast to the verifier is $VList_A$, and the communication overhead is 20 c bytes, where c is the number of VirNyms. Based on the security certificate management system (SCMS) in the United States, the pseudonym replaces the simple specification and is stored by encryption. When the vehicle moving distance is more than 2 km and it is mobile for more than five minutes [2], the RSU jurisdiction is replaced. Otherwise, an hour later LocNyms will be replaced. So, the $c = 12$, VirNym exchange stage of the communication overhead is 240 bytes, and the communication overhead is reduced while ensuring the number of VirNyms needed for vehicle communication.

6.4.2. Single Beacon Transfer Communication Cost

In this section, taking OBU_A as an example, TriNymAuth is compared with SPECS [5], b-SPECS+ [7], SPACF [13], and VPPCS [6] regarding communication cost. For convenience, we assume that the scheme environment is the same for all schemes discussed below. Since the vehicles in the scheme all involve messages when communicating, the communication overhead of messages is temporarily not considered in the comparative discussion.

In SPECS [5] and b-SPECS+ [7], the content that OBU_A broadcasts to the verifier is $\{ID_i, M_i, \sigma_i\}$, where ID_i is the anonymous identity, M_i is the message, and σ_i is the signature. $ID_i = (ID_{i1}, ID_{i2})$ and $ID_{i1}, ID_{i2} \in G_1$. $\sigma_i = SK_{i1} + h(M_i)SK_{i2}$ and $\sigma_i \in G_1$. Therefore, the communication cost is $128 \times 3 = 384$ bytes.

In SPACF [13], the content that OBU_A broadcasts to the verifier is $\{M_i, T_i, ID_i, \sigma_i\}$, where M_i is the message, T_i is the timestamp, ID_i is the anonymous identity, $ID_i = (ID_{i1}, ID_{i2}), ID_{i1} = a \cdot PK_P \in G, ID_{i2} = PK \oplus h(m_i \cdot ID_{i1}) \in Z_q^*$, σ_i is the signature, $\sigma_i = r_i + m_i H_2(ID_i||M_i||T_i)\ mod\ q$, and m_i is the new secret value. Therefore, the communication overhead is $40 \times 1 + 20 \times 2 + 4 = 84$ bytes.

In VPPCS [6], the content that OBU_A broadcasts to the verifier is $\{PID_{il}^1, PID_{il}^2, \sigma_m, T, T_{SK_{il}}\}$, where PID_{il}^1, PID_{il}^2 are the anonymous identities, σ_m is the signature, and $T, T_{SK_{il}}$ are two timestamps. $PID_{il}^1 \in G$ and $PID_{il}^2, \sigma_m \in Z_q^*$. Therefore, the communication cost is $40 + 20 \times 2 + 2 \times 4 = 88$ bytes.

In the VirNym usage phase of TriNymAuth, the content that OBU_A broadcasts to the verifier is $\{m_i, [VirNym_A]_{SK_A}, T_A\}$, where m is the message, $[VirNym_A]_{SK_A}$ is the signature, and T_A is the timestamp, therefore, the communication overhead is $64 + 4 = 68$ bytes.

VC represents the verification cost (ms), CC represents the communication cost (bytes), and TTD represents the total transmission delay (ms). The comparison of our scheme TriNymAuth and SPECS [5], b-SPECS+ [7], SPACF [13], and VPPCS [6] in terms of VC, CC and TTD is given in Table 6, where *dr* stands for data rate in VANETs.

Table 6. Comparison of verification cost, communication cost, and total transmission delay.

Schemes	VC	CC	TTD (dr = 500 kbps)	TTD (dr = 180 kbps)
SPECS [5], b-SPECS+ [7]	16.2532	384	22.3972	33.3199
SPACF [13]	0.8859	84	2.2299	9.2859
VPPCS [6]	0.8979	88	2.3059	4.8090
TriNymAuth	1.5600	68	2.6480	4.5820

Based on the formula "TTD = $\frac{CC \times 8}{dr}$ + VC", the TTD of SPECS [5] and b-SPECS+ [7] is expressed as $\frac{384 \times 8}{dr} + 16.2532$ (ms), the TTD of SPACF [10] is expressed as $\frac{84 \times 8}{dr} + 0.8859$ (ms), the TTD of VPPCS [6] is expressed as $\frac{88 \times 8}{dr} + 0.8979$ (ms), and the TTD of TriNymAuth is denoted by $\frac{68 \times 8}{dr} + 1.5600$ (ms). When the transmission distance is less than 300 m, the transmission rate of 802.11p in the 915 MHz band is lower than 500 kbps [30]. The calculation shows that when the data rate in VANETs $dr \leq 500$ kbps, TriNymAuth's total transmission delay is better than SPECS [5] and b-SPECS+ [7], and when $dr \leq 180$ kbps, TriNymAuth's total transmission delay cost is the lowest of the five. Therefore, TriNymAuth is suitable for shopping malls and other places with dense traffic.

7. Conclusions

This paper proposes a triple pseudonym authentication scheme for VANETs based on the CF and Paillier homomorphic encryption (TriNymAuth). Paillier homomorphic encryption is used to achieve an efficient HomoNym self-update. The HPP can efficiently verify the validity of HomoNyms based on the CF queries. TriNymAuth uses HomoNym, LocNym, and VirNym to realize the triple authentication of vehicle identity, which improves the authentication efficiency while ensuring the privacy and security of vehicle identity. The experimental results show that the CF performs well in terms of insertion and query time, space utilization, and false positive rate. In particular, the CF achieves 10 µs insertion overhead, and the query overhead reaches the level of ns. Moreover, in the case of more than 50%, the space utilization exceeds 80%, and the space utilization of the scheme is high. The false positive rate is as low as 1% when the fingerprint length is 32 bits. Comparing the enrollment, verification, and communication costs of TriNymAuth with those of SPECS [5], b-SPECS+ [7], SPACF [13], and VPPCS [6], it can be seen that TriNymAuth has great

advantages in the enrollment cost, and when the data rate in VANETs $dr \leq 180$ kbps, TriNymAuth has the smallest total delay cost and is suitable for shopping malls and other places with dense traffic.

Author Contributions: Methodology, L.Z. and N.G.; software, Y.C.; validation, L.Z., N.G. and Y.C.; writing—original draft preparation, L.Z.; writing—review and editing, L.Z. All authors have read and agreed to the published version of the manuscript.

Funding: This work was supported by National Natural Science Foundation of China grant number 52130403 and the Fundamental Research Funds for the Central Universities grant number N2017003.

Institutional Review Board Statement: Not applicable.

Informed Consent Statement: Not applicable.

Data Availability Statement: Not applicable.

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

References

1. Fan, B.; Andersen, D.G.; Kaminsky, M.; Mitzenmacher, M.D. Cuckoo filter: Practically better than bloom. In Proceedings of the 10th ACM International on Conference on emerging Networking Experiments and Technologies, Sydney, Australia, 2–5 December 2014; pp. 75–88. [CrossRef]
2. *5GAA: Analysis of C-V2X Security and Privacy Requirements and Impact on SCMS Design*; 5GAA: Munich, Germany, 2018.
3. Hall, T.A.; Keller, S.S. *The FIPS 186-4 Elliptic Curve Digital Signature Algorithm Validation System (ECDSA2VS)*; National Institute of Standards and Technology: Gaithersburg, MD, USA, 2010.
4. Zhang, C.; Lu, R.; Lin, X.; Ho, P.H.; Shen, X. An efficient identity-based batch verification scheme for vehicular sensor networks. In Proceedings of the IEEE INFOCOM 2008-The 27th Conference on Computer Communications, Phoenix, AZ, USA, 13–18 April 2008; pp. 246–250. [CrossRef]
5. Chim, T.W.; Yiu, S.M.; Hui, L.C.; Li, V.O. SPECS: Secure and privacy enhancing communications schemes for VANETs. *Ad Hoc Netw.* **2011**, *9*, 189–203. [CrossRef]
6. Al-Shareeda, M.A.; Anbar, M.; Manickam, S.; Yassin, A.A. Vppcs: Vanet-based privacy-preserving communication scheme. *IEEE Access* **2020**, *8*, 150914–150928. [CrossRef]
7. Horng, S.J.; Tzeng, S.F.; Pan, Y.; Fan, P.; Wang, X.; Li, T.; Khan, M.K. b-SPECS+: Batch verification for secure pseudonymous authentication in VANET. *IEEE Trans. Inf. Forensics Secur.* **2013**, *8*, 1860–1875. [CrossRef]
8. Khan, T.; Ahmad, N.; Cao, Y.; Jalal, S.A.; Asif, M.; Cruichshank, H. Certificate revocation in vehicular ad hoc networks techniques and protocols: A survey. *Sci. China Inf. Sci.* **2017**, *60*, 1–18. [CrossRef]
9. Zhong, H.; Huang, B.; Cui, J.; Xu, Y.; Liu, L. Conditional privacy-preserving authentication using registration list in vehicular ad hoc networks. *IEEE Access* **2017**, *6*, 2241–2250. [CrossRef]
10. Raya, M.; Papadimitratos, P.; Aad, I.; Jungels, D.; Hubaux, J.P. Eviction of misbehaving and faulty nodes in vehicular networks. *IEEE J. Sel. Areas Commun.* **2007**, *25*, 1557–1568. [CrossRef]
11. Rabieh, K.; Mahmoud, M.M.; Akkaya, K.; Tonyali, S. Scalable certificate revocation schemes for smart grid ami networks using bloom filters. *IEEE Trans. Dependable Secur. Comput.* **2015**, *14*, 420–432. [CrossRef]
12. Qi, J.; Gao, T. A privacy-preserving authentication and pseudonym revocation scheme for VANETs. *IEEE Access* **2020**, *8*, 177693–177707. [CrossRef]
13. Cui, J.; Zhang, J.; Zhong, H.; Xu, Y. SPACF: A secure privacy-preserving authentication scheme for VANET with cuckoo filter. *IEEE Trans. Veh. Technol.* **2017**, *66*, 10283–10295. [CrossRef]
14. Zhang, H.; Zhang, D.; Chen, H.; Xu, J. Improving efficiency of pseudonym revocation in VANET using cuckoo filter. In Proceedings of the 2020 IEEE 20th International Conference on Communication Technology (ICCT), Nanning, China, 28–31 October 2020; pp. 763–769. [CrossRef]
15. Alazzawi, M.A.; Lu, H.; Yassin, A.A.; Chen, K. Robust conditional privacy-preserving authentication based on pseudonym root with cuckoo filter in vehicular ad hoc networks. *KSII Trans. Internet Inf. Syst. (TIIS)* **2019**, *13*, 6121–6144. [CrossRef]
16. Rengarajan, A.; Thaha, M.M. SPCACF: Secured privacy-conserving authentication scheme using cuckoo filter in VANET. *Scalable Comput. Pract. Exp.* **2020**, *21*, 101–105. [CrossRef]
17. Sun, Y.; Lu, R.; Lin, X.; Shen, X.; Su, J. An efficient pseudonymous authentication scheme with strong privacy preservation for vehicular communications. *IEEE Trans. Veh. Technol.* **2010**, *59*, 3589–3603. [CrossRef]
18. Zhang, C.; Lin, X.; Lu, R.; Ho, P.H. RAISE: An efficient RSU-aided message authentication scheme in vehicular communication networks. In Proceedings of the 2008 IEEE International Conference on Communications, Beijing, China, 19–23 May 2008; pp. 1451–1457. [CrossRef]
19. Gao, H.; Liu, C.; Li, Y.; Yang, X. V2VR: Reliable hybrid-network-oriented V2V data transmission and routing considering RSUs and connectivity probability. *IEEE Trans. Intell. Transp. Syst.* **2020**, *22*, 3533–3546. [CrossRef]

20. Sarakis, L.; Orphanoudakis, T.; Leligou, H.C.; Voliotis, S.; Voulkidis, A. Providing entertainment app lications in VANET environments. *IEEE Wirel. Commun.* **2016**, *23*, 30–37. [CrossRef]
21. Abboud, K.; Omar, H.A.; Zhuang, W. Interworking of DSRC and cellular network technologies for V2X communications: A survey. *IEEE Trans. Veh. Technol.* **2016**, *65*, 9457–9470. [CrossRef]
22. Wu, Q.; Domingo-Ferrer, J.; González-Nicolás, U. Balanced trustworthiness, safety, and privacy in vehicle-to-vehicle communications. *IEEE Trans. Veh. Technol.* **2009**, *59*, 559–573. [CrossRef]
23. Yao, L.; Wang, J.; Wang, X.; Chen, A.; Wang, Y. V2X routing in a VANET based on the hidden Markov model. *IEEE Trans. Intell. Transp. Syst.* **2017**, *19*, 889–899. [CrossRef]
24. Luo, L.; Guo, D.; Rottenstreich, O.; Ma, R.T.; Luo, X.; Ren, B. The consistent cuckoo filter. In Proceedings of the IEEE INFOCOM 2019-IEEE Conference on Computer Communications, Paris, France, 29 April–2 May 2019; pp. 712–720. [CrossRef]
25. Pagh, R.; Rodler, F.F. Cuckoo hashing. *J. Algorithms* **2004**, *51*, 122–144. ._57616. [CrossRef]
26. Group, I.W. IEEE standard for wireless access in vehicular environments-security services for applications and management messages. *IEEE Std* **2016**, *1609*, 7426684. [CrossRef]
27. Wright, M.A. The advanced encryption standard. *Netw. Secur.* **2001**, *2001*, 11–13. [CrossRef]
28. Shoup, V. A proposal for an ISO standard for public key encryption. *Cryptol. Eprint Arch.* **2001**. Available online: https://eprint.iacr.org/2001/112 (accessed on 29 October 2022).
29. Karger, D.; Lehman, E.; Leighton, T.; Panigrahy, R.; Levine, M.; Lewin, D. Consistent hashing and random trees: Distributed caching protocols for relieving hot spots on the world wide web. In Proceedings of the Twenty-Ninth Annual ACM Symposium on Theory of Computing, El Paso, TX, USA, 4–6 May 1997; pp. 654–663. [CrossRef]
30. Kenney, J.B. Dedicated short-range communications (DSRC) standards in the United States. *Proc. IEEE* **2011**, *99*, 1162–1182. [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.