applied sciences

MDPI

Article

DSVN: A Flexible and Secure Data-Sharing Model for VANET Based on Blockchain

Xiaoxuan Chen [1], Yineng Chen [2,*], Xiayu Wang [1], Xinghui Zhu [1] and Kui Fang [1,*]

[1] College of Information and Intelligence, Hunan Agricultural University, Changsha 410128, China
[2] School of Information Science and Engineering, Hunan Women's University, Changsha 410004, China
* Correspondence: yinengchen@hunau.edu.cn (Y.C.); fk@hunau.edu.cn (K.F.)

Abstract: Vehicular Ad Hoc Network (VANET) is an important part of the modern intelligent transportation system, which can provide vehicle communication at a certain distance. More importantly, VANET can provide route planning and autonomous driving for drivers by analyzing data. However, VANET's data privacy and security are a huge challenge when serving drivers. In this paper, we propose a VANET data-sharing model (DSVN) that combines ciphertext-based attribute encryption (CP-ABE), blockchain, and InterPlanetary File System (IPFS). DSVN uses an outsourced and revocable ciphertext policy attribute-based encryption (ORCP-ABE) scheme, which is improved based on CP-ABE. ORCP-ABE uses key encryption key (KEK) trees to manage user attribute groups and revoke user-level attributes. It eliminates redundant attributes in the access policy by attribute-weighted access trees. Moreover, DSVN has no single point of failure. We demonstrate the indistinguishability under the chosen-plaintext attack (IND-CPA) security of DSVN by a game based on the computational Diffie–Hellman (CDH) assumption. Experimental results show that DSVN can store and share data with low overhead. Additionally, it can revoke attributes of users safely.

Keywords: VANET; data sharing; CP-ABE; blockchain; attribute revocation

Citation: Chen, X.; Chen, Y.; Wang, X.; Zhu, X.; Fang, K. DSVN: A Flexible and Secure Data-Sharing Model for VANET Based on Blockchain. *Appl. Sci.* **2023**, *13*, 217. https://doi.org/10.3390/app13010217

Academic Editors: Nadejda Komendantova and Hossein Hassani

Received: 20 November 2022
Revised: 16 December 2022
Accepted: 19 December 2022
Published: 24 December 2022

1. Introduction

Vehicle-to-vehicle communication across shorter distances is made possible by the vehicular ad hoc network (VANET), a particular kind of mobile ad hoc networks (MANETs) [1]. VANET can help drivers get status information and real-time road condition information from other vehicles within a certain range [2–4]. However, VANET may expose users' private information, e.g., identities, location information, and trajectories. Once that shared private information is illegally used by malicious attackers, it will lead to terrible information security. Therefore, how to store and share VANET data safely is a research hotspot. Fortunately, the emergence of blockchain technology has addressed the problems above. Blockchain is decentralized, transparent, a consensus mechanism, and tamper-proof. [5–7]. It maintains the security of data and punishes malicious attackers for VANET. Currently, there are two types of blockchain-based VANET data-sharing schemes. The first type, the data is completely uploaded to the blockchain [8–10], and each node synchronizes the block data in real time. This strategy is not appropriate for large-scale data storage scenarios. The other, the data is kept on a cloud server, but transactions are recorded on the blockchain [11,12]. Such schemes cannot avoid single failure spots. To eliminate this failure, we store metadata on the InterPlanetary File System (IPFS), eliminating the high overhead and low efficiency of data storage resources on the blockchain. IPFS uses distributed storage and allows nodes to retrieve and store data as backups, avoiding single points of failure.

In addition, access control provided by a trusted third party cannot satisfy VANET data on the cloud server. This mechanism limits the flexibility of data sharing and has security concerns. It has become a new challenge to provide fine-grained access control

for data-sharing scenarios. The primary method of implementing data access control is encryption. However, conventional encryption schemes cannot satisfy the access control needs in the VANET scenario. For instance, it is challenging to provide decryption keys to the desired data access users when using the Advanced Encryption Standard (AES) [13]. Before encrypting the data using the RSA encryption algorithm, the owner must collect the public key from each user. New users cannot access data that has been encrypted [14]. Attribute-based encryption (ABE), suggested by Sahai et al., allows for precise access control over encrypted data [15]. Depending on the objects connected with the access control policy, ABE is further separated into ciphertext policy attribute-based encryption (CP-ABE) [16] and key policy attribute-based encryption (KP-ABE) [17]. Compared with KP-ABE, CP-ABE is more suitable for dynamic scenes. Therefore, most of the schemes supporting fine-grained access control in VANET are based on CP-ABE. In VANET, the user's attribute set is dynamic. Various circumstances may cause the user's attribute set to change. For instance, Mike works as a traffic cop in the supervisory division. He is transferred outside of his department due to a change in position. The "supervisory department" in user Mike's attribute set needs to be removed. In the traditional CP-ABE approach, the user's attribute set is immutable. When the user's attributes need to be modified, CP-ABE can only re-register after deleting the user. This method of processing adds overhead and is not safe during processing. We build a KEK tree based on the user set to manage the rights of different users. When revoking a specific user attribute, the revoked user updates its key while other users are unaffected.

In response to the security issues in VANET data sharing, we design a data-sharing model for VANET (DSVN). First, we build a collaborative network of consortium blockchain and IPFS with RSUs as nodes [18], a distributed storage network. The model ensures the tamper-proof and integrity of shared information through a blockchain consistency mechanism. Second, we implement an efficient encryption scheme ORCP-ABE. Our scheme supports computational outsourcing and user attribute revocation functions. In addition, we propose an attribute-weighted access tree. Data owners can eliminate redundant attributes and improve the efficiency of data encryption by constructing such access trees. Finally, we prove that the scheme is IND-CPA safe in a game model under the CDH assumption and test the scheme's performance through simulation experiments.

2. Related Work

2.1. VANET

VANET is a mobile network formed using wireless communication technology with moving vehicles and transportation facilities as nodes. VANET is an important part of intelligent transportation system. With the widespread application of VANET, the security of private data between vehicles has become a primary concern [19,20]. To solve this problem, many scholars have proposed solutions [21–25]. Deng et al. propose a secure VANET authentication scheme (PAS), where a software-defined network (SDN) is integrated as a suitable infrastructure to support anonymous authentication and pseudonym management [21]. Chen et al. propose a decentralized VANETs (DVANETs) architecture, where computing tasks are decomposed from centralized cloud services to edge computing (EC) nodes, thereby effectively reducing network communication overhead and congestion delay [22]. Li et al. implement blockchain instead of third-party service providers for user identity management and data storage, and lightweight VANET devices can outsource complex encryption and decryption operations to RSUs [23]. The scheme of Ma et al. records users' keys, uploads, and access transactions for auditing through blockchain [24]. The scheme of Zhang et al. implements outsourced encryption and revokes malicious users [25].

2.2. Blockchain

Blockchain is a chained data structure that combines data blocks in a sequentially connected manner in chronological order and is a cryptographically guaranteed, immutable,

and unforgeable distributed ledger. Blockchain originated from Bitcoin, which was proposed by Satoshi Nakamoto [26]. Cryptocurrencies have developed rapidly recently, and blockchain has received widespread attention. Blockchain technology has been widely used in finance, healthcare, industry, and other fields [27–29].

For data storage security issues, numerous academics have suggested blockchain-based data storage schemes [30–32]. A blockchain-based data access architecture for the Internet of Things was put forth by Shafagh et al. [30]. The model achieves secure access control management without needing a centralized, trusted third-party organization by employing blockchain as a distributed access control layer for the storage layer. A VANET security architecture based on blockchain and mobile edge computing was presented by Zhang et al. [31]. With blockchain technology, this architecture guarantees the security of VANET data during horizontal dissemination. A VANET untrustworthy system concept based on blockchain and certificate authority (CA) was presented by Javaid et al. [32]. This model can establish distributed trust management for secure data sharing while protecting privacy. According to the former study, a distributed storage system based on a blockchain can offer security and dependability that are superior to conventional methods. It can prevent third-party-caused data loss and privacy leaks.

2.3. CP-ABE

ABE is the most promising cryptographic primitive supporting fine-grained access. ABE was first proposed by Sahai and Water in Fuzzy Identity Based Encryption (FIBE). [15]. Bethencourt presented the first CP-ABE method based on ABE [17]. This scheme allows the data owner to define an access policy. Only users who satisfy the policy can decrypt the data. Afterwards, many scholars put forward their own schemes based on this [33–35]. A scheme to implement access control over system attributes was put up by Water [32] utilizing a linear secret sharing scheme (LSSS) matrix. Green et al. [34] implement an effective and innovative scheme that can be outsourced. A multi-authority attributes-based encryption scheme was put forth by Lewko et al. [35].

The security of existing CP-ABE solutions is still inadequate. Most schemes cannot track down and deal with malicious users who leak their keys. Numerous academics have suggested original solutions to this issue. Praveen Kumar et al. used a dynamic traceable CP-ABE method with revocation [36]. This scheme dynamically tracks the decryptor during the outsourcing decryption process and helps to identify and revoke the malicious user who leaked the key. Kamalakanta et al. [37] suggested a practical encryption scheme with multi-authority and efficient revocation of ciphertext policy attributes. This scheme achieves user revocation by algorithmically updating the key for unrevoked users. However, this approach is only suitable for some scenarios with many users. Based on the traceable revocable ciphertext policy attribute, Yi et al. [38] suggested an equal-length ciphertext key encryption scheme. Based on achieving user revocation, the transmission efficiency is improved by constant-length ciphertext and key. These schemes effectively address the revocation of malicious users but cannot perform flexible revocation of user attributes when a user changes. A multi-permission ciphertext policy attribute and revocable permissions-based encryption scheme were presented by Yang et al. [39]. The scheme allows multiple permission authorities to participate in key distribution and enables at-tribute revocation when the user's access rights change. A blockchain-based revocable CP-ABE method was suggested by Xin et al. [40]. The scheme supports an expressive access control policy and allows attribute permissions to revoke some user attributes.

3. System Preview

3.1. Architecture

This scheme combines consortium blockchain, attribute-based encryption, and IPFS technology to propose a new distributed VANET data storage and sharing system, as shown in Figure 1.

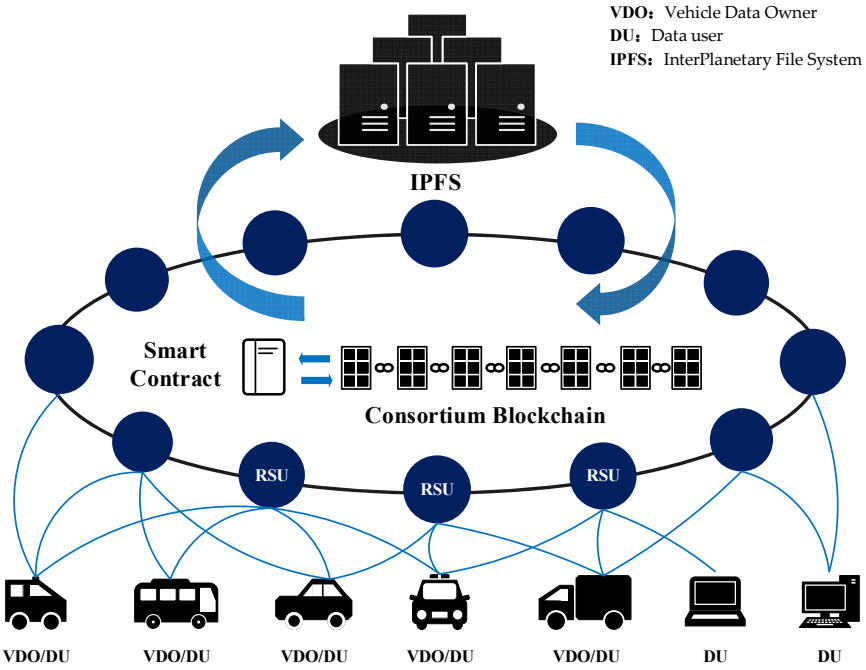

Figure 1. System model.

This system consists of six parts:

- The vehicle data owner (VDO) is the producer and sharer of data in telematics. VDO represents vehicles and corresponding onboard devices, which usually do not have the performance of storing and sharing data on a large scale. In addition, it has dynamic characteristics, so it only connects to CBN through RSU as a user.
- The roadside unit (RSU) is a communication unit distributed along both sides of the road at a certain distance. RSU has strong processing performance, sufficient storage space, and a good network connection. In this system, all RSUs form a consortium chain network as nodes. RSUs perform user data upload and access operations within their coverage area while using their high performance to do most data encryption and decryption work.
- The consortium blockchain network (CBN) is an intermediate party ensuring data sharing security and trustworthiness. In this system, CBN is composed of all RSU nodes together. The information in the system that involves user privacy and data encryption is recorded securely on the blocks of CBN.
- The smart contract (SC) is a complete set of operation methods defined on CBN, automatically performing different operations in different phases. For example, in the system initialization phase, SC is responsible for generating system keys. In the user registration phase, RSU can write user registration information to BN by calling SC.
- The data user (DU) is the user of data. The DU requests data by calling SC through RSU. In the physical layer, DU and VDO may be the same entity, and the vehicle can share its data while requesting data.
- The InterPlanetary File System (IPFS) is the data service provider. All shared source data in the system are stored in IPFS. VDO uploads metadata to IPFS via RSU.

3.2. Definitions

3.2.1. Attribute-Weighted Access Tree

Data users have different identities, and their attribute sets are complex. The attributes of data users may have containment or hierarchical relationships. For example, in the traffic management department, Constable, Superintendent, Inspector, and Superintendent are one class of attributes representing different levels of police officers. There is a clear hierarchical relationship between these attributes. The access range of the high-level attributes

includes the access range of the low-level attributes, i.e., constable ⊂ superintendent ⊂ inspector ⊂ superintendent. This makes the access tree have a lot of redundant attributes. The attribute-weighted access tree can solve this problem.

As shown in Figure 2, the attribute-weighted access tree has three levels. The root node is a logical "or" the 2nd level non-leaf node is a logical "and", and the 3rd level leaf node is an attribute expression. The construction method of the access tree for attribute assignment: The system attribute set U will be divided into categories $L = \{L_1, L_2, L_3, \ldots, L_N\}$ based on entities and departments. Then, weights will be assigned to attributes in the same class where there is a continuous containment relationship for access rights. For example, if there is $L_{i,1} \subset L_{i,2} \subset L_{i,3} \cdots \subset L_{i,n}$, then the attribute $L_{i,1}$ will be assigned weight w of 1, $L_{i,2}$ weight w of 2, and so on for the rest. Finally, all attributes will be replaced with the corresponding category weight pairs.

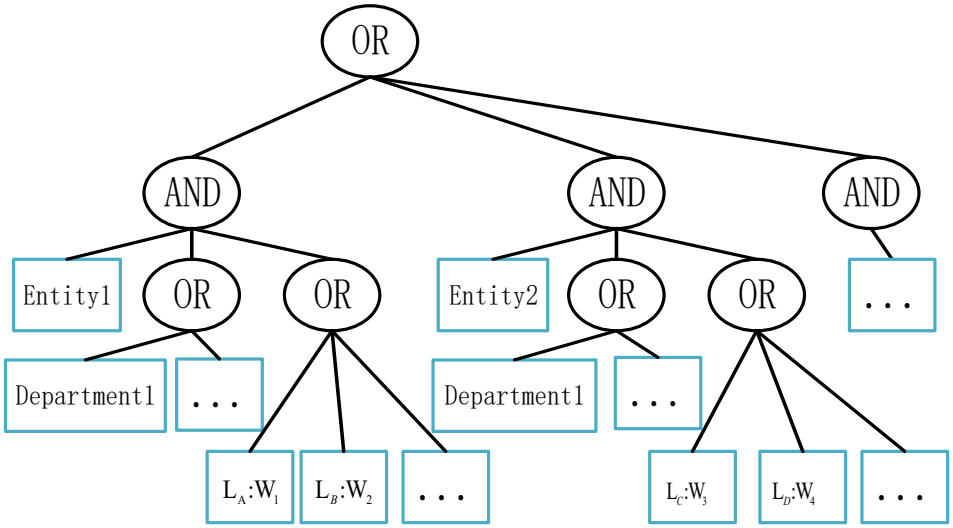

Figure 2. The attribute-weighted access tree.

3.2.2. The Key Encryption Key Tree

The key encryption key (KEK) tree is a complete binary tree constructed based on the user set [41], as shown in Figure 3. KEK tree can provide a non-revoking user update function to achieve attribute revocation. For example, suppose the system user set is User $= \{u_1, u_2, \ldots, u_n\}$, and the system attribute set is $U = \{att_1, att_2, \ldots, att_m\}$. Then, the steps for the system to construct a KEK tree are as follows:

1. Each user in the user set User is designated in the leaf node of the binary tree, and each node stores a random value θ_i;
2. Path node generation algorithm $Path(u_i)$: For any user u_i, all nodes passing through the path from its corresponding leaf node to the root node are defined as the path nodes of user u_i;
3. The minimum coverage set algorithm $Mincs(G_j)$: For the attribute group G_j with the attribute att_j, the minimum set of nodes in the KEK tree covering all users of G_j is the minimum coverage set;
4. Calculate the intersection of $Path(u_i)$ and $Mincs(G_j)$: If the user has the attribute att_j, then the intersection has only one node V_k. θ_k is a random value stored in node V_k. If the user has no attributes att_j, then the intersection set is empty.

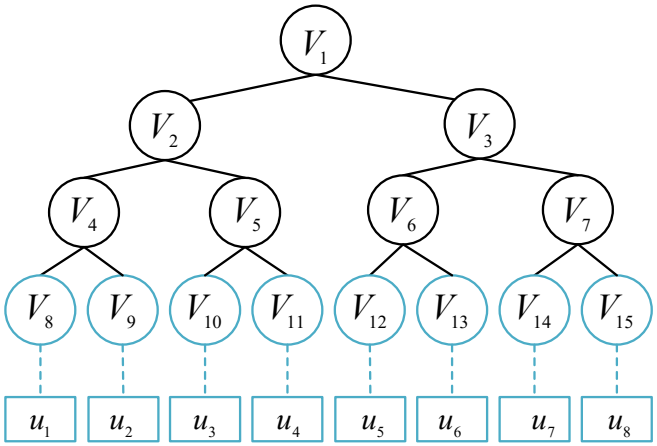

Figure 3. The key encryption key tree.

3.2.3. Security Definition

1. The computational Diffie–Hellman (CDH) assumption: Suppose there are cyclic groups $\mathbb{G}_0$ and $\mathbb{G}_T$ of the same prime order p. g is a generator of $\mathbb{G}_0$, and $e : \mathbb{G}_0 \times \mathbb{G}_0 \to \mathbb{G}_T$ is a bilinear pair. Choose a,b randomly from $\mathbb{Z}_p$. The computational Diffie–Hellman problem constructs a polynomial adversary $\mathcal{A}$ that takes the tuple $(\mathbb{G}_0, p, g, g^a, g^b)$ as input and outputting $e(g,g)^{ab} \in \mathbb{G}_T$, which has the advantage:

$$Adv_{\mathcal{A}} = \Pr[\mathcal{A}(\mathbb{G}_0, p, g, g^a, g^b) = e(g,g)^{ab}]. \quad (1)$$

Definition 1. *The CDH assumption holds if no probabilistic polynomial-time adversary $\mathcal{A}$ has a nonnegligible advantage in solving the CDH problem.*

2. Security model: Below, we define the ciphertext indistinguishability under chosen-plaintext attacks.

Setup. The challenger C runs the initialization algorithm, generates the public parameters, and sends them to the adversary $\mathcal{A}$.

Phase 1. The adversary $\mathcal{A}$ is allowed to select a set S of attributes for a key query. The challenger C randomly runs the registration algorithm and returns the result to $\mathcal{A}$.

Challenge. The adversary $\mathcal{A}$ submits challenge access structure T^* and equal-length messages M_0 and M_1, which are sent to the challenger. The challenger C chooses $\theta \in \{0,1\}$ randomly, and runs pre-encryption and re-encryption algorithm to encrypt m_θ and generates ciphertext CT'. Then C return CT' to $\mathcal{A}$.

Phase 2. The adversary $\mathcal{A}$ can make a key query in the same way as in Phase 1, except that the set of attributes S satisfying access structure T^* related to the challenge ciphertext CT' cannot be queried in the key query phase.

Guess. The adversary $\mathcal{A}$ outputs a guess θ^* for θ. If $\theta^* = \theta$, then $\mathcal{A}$ wins the game. The $\mathcal{A}$'s advantage is defined:

$$Adv_{\mathcal{A}} = |\Pr[\theta^* = \theta] - \frac{1}{2}|. \quad (2)$$

Definition 2. *If the adversary $\mathcal{A}$ cannot selectively win by a non-negligible advantage in polynomial time, the scheme is secure against Chosen-Plaintext Attacks.*

4. System Design

4.1. System Flow

The description of the symbols and abbreviations appearing in this paper are shown in Table 1.

Table 1. Description of Symbols.

Symbols	Description
λ	Safety parameters
U,S	System attribute set and user attribute set
PK, MSK	System public key and private key
DPK, DSK	Data public key and private key
id, SK_{id}	User id and key
USK	Global parameters for user access
kek	Attribute Encryption Information
KEK	User attribute group encryption information
M	Data address information
T,T^*	Access tree
CT	Pre-encrypted ciphertext
G_x	The attribute groups of attribute x
CT', $CT*$	Encrypted ciphertext
Hdr	Encryption header
$PDCT$	Pre-decrypted ciphertext
$\overline{\mathrm{M}}$	The decrypted data address information

As shown in Figure 4, the system has the following five main phases:

1. System initialization: RSU inputs λ and U as parameters and invokes the initialization contract. The contract will execute algorithm 1 to generate PK, MSK, DPK, and DSK, and record them in the genesis block of CBN;
2. User registration: VDO (or DU) sends the registration request containing the user id and user attribute set S to RSU. Then, RSU verifies the authenticity and validity of the registration information and then invokes the user registration contract. The contract reads the PK, MSK and DPK from CBN block and executes algorithm 2 to generate SK_{id}, kek, USK and KEK. id and S are written to the block of CBN. Finally, RSU returns SK_{id} and kek to VDO as the result of successful registration;
3. Data upload: VDO uploads Data to IPFS via RSU. IPFS returns the retrieval code to the RSU. Then RSU generates M corresponding to the retrieval code and returns it to VDO as the result of data upload. VDO receives M and invokes algorithm 3 to generate the attribute-weighted access tree T^*, and then invokes algorithm 4 to generate CT. VDO sends CT, G to RSU to invoke the data upload contract. The contract executes algorithm 5 to generate CT', Hdr and writes the relevant information into the block of CBN. Finally, RSU broadcasts CT' to DUs;
4. Data access: data user DU sends SK_{id} and CT' to RSU. RSU invokes the data access contract. If DU satisfies the data access condition, the contract will execute algorithm 6 to calculate and get $PDCT$ and send it to DU. Otherwise, the execution of the contract will be terminated by the execution failure of algorithm 6. Then, DU executes algorithm 7 to calculate $\overline{\mathrm{M}}$ and return it to RSU. After receiving $\overline{\mathrm{M}}$, RSU will read the data retrieval code and connect IPFS to download data. Finally, DU successfully accesses data;
5. User attribute revocation: RSU invokes the user attribute revocation contract with revoked user id and att as input. The contract calls algorithm 8 to update DPK, DSK and outputs the updated KEK. Then, contract reads the encrypted ciphertext CT' associated with the user id in the CBN block and calls algorithm 9. The algorithm 9 updates Hdr and CT', and writes the updated $CT*$ on the new block in CBN. Finally, RSU broadcasts the updated $CT*$ to DUs.

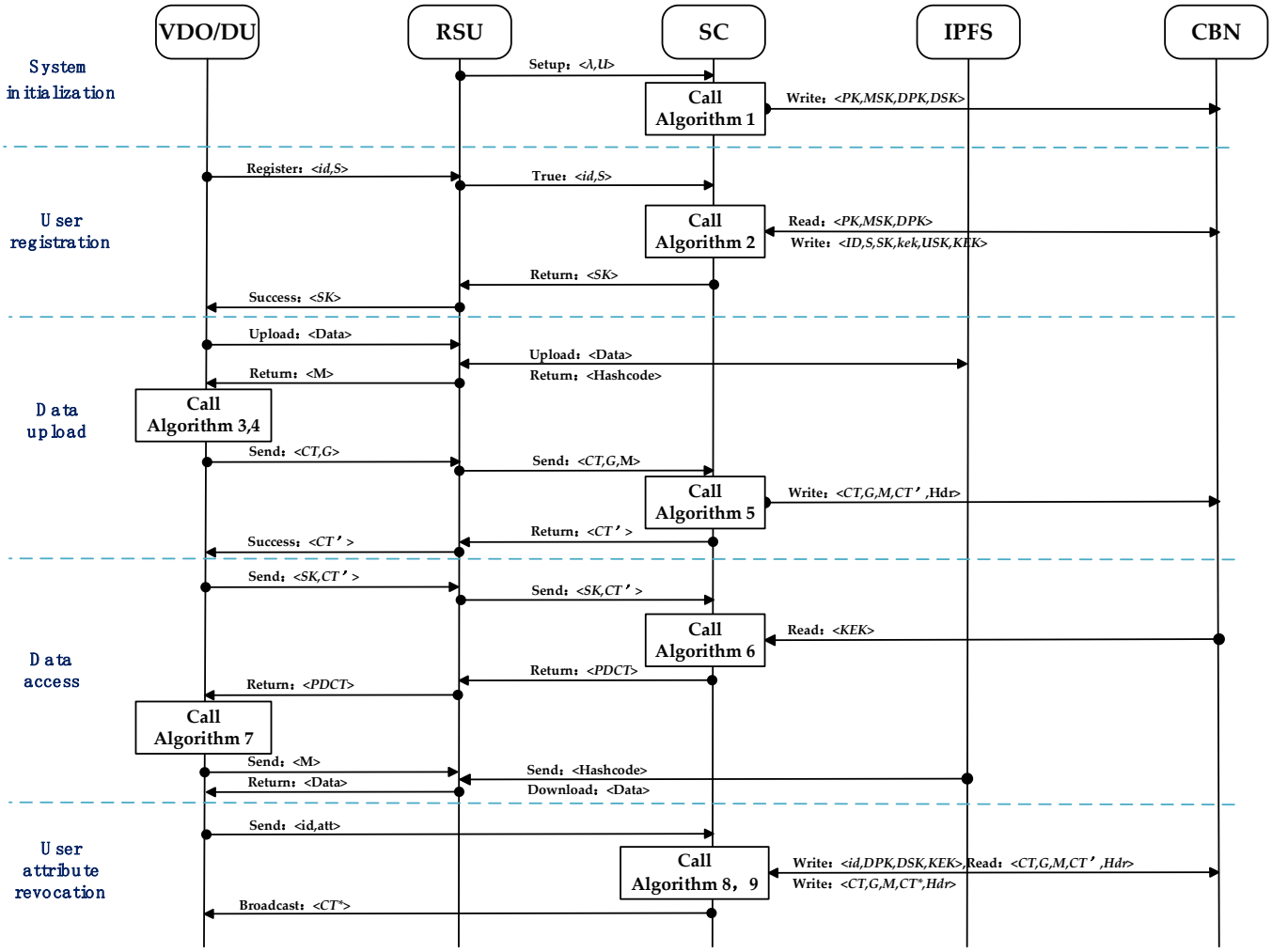

Figure 4. System flow.

4.2. ORCP-ABE Algorithm

The ORCP-ABE algorithm consists of the following algorithms.

- Initialization $(\lambda, U) \rightarrow PK, MSK, DPK, DSK$: The algorithm takes security parameters λ and system attribute set U as input. Algorithm constructs a bilinear mapping $e : \mathbb{G}_0 \times \mathbb{G}_0 \rightarrow \mathbb{G}_T$. Where $\mathbb{G}_0$ and $\mathbb{G}_T$ are two bilinear groups of prime of order p. g is the generator of the cyclic group $\mathbb{G}_0$. Then algorithm randomly selects the index value t_i for each attribute $att_i \in U$. Finally, algorithm randomly selects two numbers a, β, then calculates system key pair MSK, PK, and data key pair DSK, DPK:

$$\begin{gathered} MSK = (\beta, g^{\alpha}), PK = \left(g, e(g,g)^{\alpha}, h\right) \\ DSK = \left\{t_i \middle| 1 \leq i \leq n\right\}, DPK = \left\{T_i = g^{t_i} \middle| 1 \leq i \leq n\right\} \end{gathered} \tag{3}$$

  where $a, \beta, t_i \in Z_p^*$, $h = g^{\beta}$, and n is the number of attributes in U (note: The system public key PK and the data public key DPK are publicly accessible);
- Key generation $(id, S, MSK) \rightarrow SK_{id}$: The algorithm takes user number id, attribute set S and system private key MSK as input. Algorithm randomly selects uid, USK, calculates user private key SK_{id} and user attribute encryption information kek. For each attribute $att_i \in S$, algorithm calculates node intersection V_j and judges whether V_j is empty. If $V_j = \varnothing$, algorithm stop the calculation, else calculates user attribute group encryption information KEK:

$$\begin{aligned} SK_{id} &= \left(D = g^{\frac{\alpha+uid}{\beta}}, D' = g^{uid \cdot USK}, USK\right), \\ kek &= \left\{att_i, kek_i = g^{t_i \cdot uid \cdot USK}\right\}_{att_i \in S'} \\ KEK &= \left\{KEK_i = (kek_i)^{\frac{1}{\theta_j}} = g^{\frac{t_i \cdot uid \cdot USK}{\theta_j}}\right\}_{att_i \in S} \end{aligned} \tag{4}$$

where $uid, USK \in Z_p^*, V_j = Path(u_i) \cap Mincs(G_j)$ and θ_j is the value in node V_j (note: The system private key MSK is not visible and is only accessed through calls during the user registration phase);

- Access tree generation $(T, U) \rightarrow T^*$: The algorithm takes access structure T and the system attribute set U as input. First, algorithm classifies $att_i \rightarrow L_j$ and assigns weights w for each attribute $att_i \in U$. Then, algorithm replaces each attribute in T with the corresponding category weight pairs $L_j : w$. Finally, algorithm constructs attribute-weighted tree T^*;
- Pre-encryption $(M, T^*) \rightarrow CT$: The algorithm takes the message M and attribute-weighted tree $T*$ as input. First, algorithm generates a randomly univariate polynomial $Q_{node}(x)[\,]$ or each tree node. The secret value of node $node$ is $Q_{node}(0)$. Then, algorithm calculates pre-encrypted ciphertext CT:

$$CT = \left(T^*, \widetilde{C} = Me(g,g)^{\alpha s}, C = h^s, \forall y \in Y : C'_y = g^{Q_y(0)}\right) \tag{5}$$

where s is the secret value of root node, y is a leaf node of T^*, and Y is leaf node set;

- Encryption $(CT, G) \rightarrow CT', Hdr$: The algorithm takes attribute-weighted tree T, pre-encrypted ciphertext CT and access structure attribute set G as input. Algorithm randomly selects k_y, and calculates CT':

$$\begin{aligned} CT' &= \left(T^*, \widetilde{C} = Me(g,g)^{\alpha s}, C = h^s, \forall y \in Y : C'_y = g^{Q_y(0)} \cdot g^{k_y}\right), \\ Hdr &= \left\{v_j, E\left(k_y\right) = g^{k_y \cdot \theta_j / t_i}\right\}_{v_j \in Mincs(G_i)} \end{aligned} \tag{6}$$

where $k_y \in Z_p^*$, G_i is the set of user attributes containing attribute att_i, and θ_j is the value in the leaf node V_j (note: V_j is different from y; V_j is a leaf node in the key encryption key tree, where y is a leaf node in the visited tree);

- Pre-decryption $(CT', SK_{id}, KEK) \rightarrow PDCT$: The algorithm takes ciphertext CT', access user key SK_{id} and user attribute group encryption information KEK. Algorithm preorder traversal attribute-weighted tree T^*. For node $x \in T^*$, algorithm calculates the decryption value of leaf node DN or the decryption value of non-leaf node F_x. If x is leaf node calculate DN:

$$\begin{aligned} DN(CT', SK_{id}, KEK, x) &= \frac{e\left(D', C'_y\right)}{e\left(KEK_i, E\left(k_y\right)\right)}, \\ &= e(g,g)^{uid \cdot USK \cdot Q_x(0)} \end{aligned} \tag{7}$$

else calculate F_x:

$$\begin{aligned} F_x &= \prod_{z \in S_x} F_z^{\Delta i, S'_x(0)} \\ &= \prod_{z \in S_x} \left(e(g,g)^{uid \cdot USK \cdot Q_z(0)}\right)^{\Delta i, S'_x(0)} \\ &= \prod_{z \in S_x} \left(e(g,g)^{uid \cdot USK \cdot Q_{parent(z)}(child(z))}\right)^{\Delta i, S'_x(0)} \\ &= \prod_{z \in S_x} \left(e(g,g)^{uid \cdot USK \cdot Q_x(i)}\right)^{\Delta i, S'_x(0)} \\ &= e(g,g)^{uid \cdot USK \cdot Q_x(0)} \end{aligned} \tag{8}$$

where x represents the currently traversed node. S_x denote any set of child nodes of scale t_x. For all leaf nodes $z \in S_x$, transfer $DN(CT', SK_{id}, KEK, z) \rightarrow F_z$. When user does not satisfy the set of attribute groups G_x, $DN(CT', SK_{id}, KEK, x)$ will not be calculated and will be skipped. Finally, algorithm calculates $PDCT$:

$$PDCT = (F_{root}, \widetilde{C}) = (e(g,g)^{uid \cdot USK \cdot s}, Me(g,g)^{as}); \tag{9}$$

- Decryption algorithm $(PDCT, USK) \rightarrow \overline{M}$: The algorithm takes pre-decrypted information $PDCT$ and user data private key USK as input. Algorithm calculates $\overline{M}$:

$$\overline{M} = \frac{\widetilde{C} \cdot A^{\frac{1}{USK}}}{e(C, D)} = \frac{Me(g,g)^{as}}{\frac{e\left(g^{\beta s}, g^{\frac{\alpha + uid}{\beta}}\right)}{e(g,g)^{uid \cdot s}}}; \tag{10}$$

- Update $KEK(id, att_x) \rightarrow KEK$: The algorithm takes user id and revoked attribute att_x as input. Algorithm randomly selects σ_x, and updates data key pair DSK, DPK:

$$\begin{aligned} DSK &= \{t_i | 1 \le i \le n, i \ne x\} \cup \{t_x^* = t_x \cdot \sigma_x\}, \\ DPK &= \{T_i | 1 \le i \le n, i \ne x\} \cup \left\{T_x{}^* = T_x^{\sigma_x}\right\}. \end{aligned} \tag{11}$$

where $\sigma_x \in Z_p^*$. For $u_k \in G_x$, the algorithm calculates user attribute group encryption information KEK:

$$\begin{aligned} \overline{\varphi_x} &= Path(u_k) \cap Mincs(G_x), \\ kek_x &= (kek_x)^{\sigma_x}, KEK_x = (kek_x)^{\theta_{j'}}. \end{aligned} \tag{12}$$

where, $\theta_{j'}$ is a value corresponding to the node $\overline{\varphi_x}$;

- Update ciphertext $(CT') \rightarrow CT*$: The algorithm takes ciphertext CT' as input. Algorithm randomly selects s', k_y', and updates $CT*$:

$$\begin{aligned} \widetilde{C} &= \widetilde{C} \cdot e(g,g)^{\alpha s'}, C = C \cdot h^{s'}, C_y' = g^{Q_y(0)} \cdot g^{k_y'}, \\ Hdr &= \begin{cases} \left\{\overline{v_j'}, E\left(k_y\right) = g^{k_y' \cdot \theta_{j'} / t_x}\right\} & \overline{v_j'} \in Mincs(G_x) \\ \left\{v_j', E\left(k_y\right) = g^{k_y \cdot \theta_j / t_i}\right\} & v_j' \in Mincs(G_i), i \ne x \end{cases}. \end{aligned} \tag{13}$$

where $s', k_y' \in Z_p^*$.

5. Security Analysis

Theorem 1. *If the CDH assumption holds in Group $\mathbb{G}$, then no CPA attacker can selectively corrupt the scheme in polynomial time with a non-negligible advantage.*

Proof. Assuming that attacker $\mathcal{A}$ can selectively break the scheme proposed in this paper with a non-negligible advantage $Adv_{\mathcal{A}} = \varepsilon$ after executing q_1 times Type-1 and q_2 times Type-2 key queries, then a challenger $\mathcal{C}$ can be constructed to break the CDH assumption with a non-negligible advantage $Adv_C = \varepsilon / (q_1 \cdot q_2)$. This section describes the IND-CPA security model of the scheme, which is a game between a challenger and an attacker. The flow is as follows:

Init: The challenger $\mathcal{C}$ inputs a random CDH challenge $A = g^{Z_1}$ and $B = g^{Z_2}$. The attacker $\mathcal{A}$ selects the access structure T^* and attribute att_x^* to be sent to $\mathcal{C}$, where att_x^* is a required attribute to satisfy T^*.

Setup: The challenger $\mathcal{C}$ generates $PK = \left(g, e(g,g)^{\alpha}, h = g^{\beta}\right)$, $MSK = (\beta, g^{\alpha})$, $\overline{DPK} = \{T_i | 1 \le i \le n, i \ne x\} \cup \left\{\overline{T_x^*}\right\}$, and $\overline{DSK} = \{t_i | 1 \le i \le n, i \ne x\} \cup \left\{\overline{t_x^*}\right\}$.

Among them, $\overline{T_x^*} = (T_x^*)^{z_1}$ is a theoretical value. Then $\mathcal{C}$ sends PK to $\mathcal{A}$.

Phase 1: The attacker $\mathcal{A}$ can request two types of keys. The challenger $\mathcal{C}$ initializes two empty lists L_1, L_2 to record the requested key.

Type-1: The attributes set S_1 of user u_1 satisfies the access structure T^*, but the attribute att_x^* are revoked. The challenger $\mathcal{C}$ calculates SK_{u_1}, kek and KEK, then sends them to $\mathcal{A}$:

$$\begin{aligned} SK_{u_1} &= (D, D', USK_1) = (g^{\frac{\alpha + u_1 \cdot Z_2}{\beta}}, g^{u_1 \cdot Z_2 \cdot USK_1}, USK_1), \\ kek &= \begin{cases} \left\{att_i, kek_i = g^{t_i \cdot u_1 \cdot USK_1}\right\} & att_i \in S_1, i \ne x \\ \left\{att_x, kek_x^* = g^{t_x^* \cdot u_1 \cdot USK_1 \cdot Z_2}\right\} & att_x^* \in S_1 \end{cases}, \\ KEK_{S_1} &= \{KEK_i\}_{i \ne x} \cup \left\{KEK_x^* = (kek_x^*)^{1/\theta^*} = B^{t_x^* \cdot u_1 \cdot USK / \theta^*}\right\}. \end{aligned} \tag{14}$$

Type-2: The attributes set S_2 of user u_2 has attributes att_x^*, but does not satisfy the access structure T^*. The challenger $\mathcal{C}$ calculates SK_{u_2}, kek and KEK, then sends them to $\mathcal{A}$. The calculation here is like Type-1.

Challenge: The attacker $\mathcal{A}$ submits two messages of equal length, M_0 and M_1, and randomly selects $b = \{0,1\}$. The challenger $\mathcal{C}$ will calculate $\widetilde{C} = M_b e(g,g)^{\alpha s}$, $C = h^s$ and C'_y (or C_x^*). For $i \neq x, C'_y = g^{Q_y(0)} \cdot g^{k_y}$. For att_x^*, $C_x^* = g^{Q_y(0)} \cdot A^{k_x}$. Then, $\mathcal{C}$ calculates CT_b and Hdr^*:

$$CT_b = \left\{ \widetilde{C}, C, \{C_x^*\} \cup \left\{ C'_y \right\}_{i \neq x} \right\},$$
$$Hdr^* = \begin{cases} \left\{ v_j^*, E(k_x^*) = g^{k_x^* \cdot \theta_j^* / t_x^*} \right\} v_j^* \in Mincs(G_x) \\ \left\{ v_j, E(k_y) = g^{k_y \cdot \theta_j / t_i} \right\} v_j \in Mincs(G_i), i \neq x \end{cases}. \quad (15)$$

Finally, $\mathcal{C}$ sends CT_b and Hdr^* to $\mathcal{A}$.

Phase 2: The attacker $\mathcal{A}$ is allowed to request keys as in Phase 1.

Guess: The attacker $\mathcal{A}$ outputs $b' \in \{0,1\}$ as the prediction. Assume the attacker's advantage $Adv_A = \left| P_r[b' = b] - \frac{1}{2} \right| = \varepsilon$, and the challenger $\mathcal{C}$ chooses a key from L_1 and L_2. Exist as follows:

$$\frac{e(D', C_x^*)}{e(KEK_x^*, E(k_x^*))} = \frac{e\left(g^{u_1 \cdot z_2 \cdot USK_1}, g^{Q_y(0)} \cdot A^{k_x} \right)}{e\left(A^{t_x^* \cdot u_2 \cdot USK_2 / \theta_j^*}, g^{k_x \cdot \theta_j^* / t_x^*} \right)} \quad (16)$$

Only if $A^{u_2} = g^{z_1 z_2 u_1}$, the calculation is established, then $\mathcal{C}$ calculates $g^{z_1 z_2} = (KEK_x^*)^{\theta_j^* / (u_1 \cdot t_x^*)}$. If $\mathcal{C}$ does not terminate the game, suppose that after q_1 Type-1 and q_2 Type-2 key queries, the probability that $\mathcal{C}$ correctly chooses from the two lists is $\frac{1}{q_1 \cdot q_2}$. Therefore, the advantage of $\mathcal{C}$ to break the CDH assumption is $\frac{\varepsilon}{q_1 \cdot q_2}$. Therefore, the $\mathcal{C}$ can break the CDH assumption in polynomial time with a non-negligible advantage. The proof is over. □

6. Experiment and Analysis

6.1. Comparison

This section compares this scheme's features and computational cost with some schemes [23–25]. These schemes are recent schemes in VANET and are all based on CP-ABE. Table 2 compares the features between our ORCP-ABE scheme and relevant schemes in recent years. The access policy is the implementation of the access policy in each scheme. "Outsourcing Calculation" refers to whether the scheme supports outsourcing part of the computation in the encryption or decryption process to the RSU for completion. "Attribute Assignment" refers to whether the scheme implements the classification and assigns weights to attributes in the access structure based on the attribute relationships. "Attribute Revocation" refers to whether the scheme can provide specific attribute revocation for malicious users. As shown in Table 2, our scheme implements attribute assignment and supports attribute revocation at the user level. Our scheme is flexible in dealing with malicious users by revoking only some of their attributes.

Table 2. Comparison of features.

Scheme	Access Policy	Outsourcing Calculation	Attribute Assignment	Attribute Revocation
[23]	TREE	Yes	No	No
[24]	TREE	Yes	No	No
[25]	LSSS	Yes	No	No
Ours	TREE	Yes	Yes	Yes

Table 3 illustrates the comparison of computational cost. The values in Table 3 are calculated by mathematical formulas and codes after reproduction. We use $|S|$, and $|T|$ to represent the number of attributes of the user attribute set S and access structure T, respectively. $|I|$ represents the number of attributes satisfying the access structure T and $|L|$ represents the number of rows of the access matrix. $|R|$ is the number of ciphertexts involved in the attribute revocation.E and E_T represent exponential operations on $\mathbb{G}$ and $\mathbb{G}_T$, respectively, and P represents bilinear pairing operations. As shown in Table 3, our scheme is more efficient than the other three schemes in the encryption phase since we simplify the access structure by attribute assignment.

Table 3. Comparison of computational cost.

Scheme	Key Generation	User Encryption	Outsourced Encryption	User Decryption	Outsourcing Decryption	Attribute Revocation
[23]	$(4+\|S\|)E$	$P+4E$	$(2+2\|T\|)E$	$P+2E_T$	$2\|S\|P+\|I\|E$	/
[24]	$(2+3\|S\|)E$	$P+(4+\|T\|)E$	$(2+2\|T\|)E$	$P+2E_T$	$2\|S\|P+\|I\|E$	/
[25]	$(1+6\|S\|)E$	$\|L\|P+E_T+(1+6\|L\|)E$	/	E_T	$(6+\|L\|)P+\|L\|E$	/
Ours	$(4+\|S\|)E$	$P+E$	$(2+3\|T\|)E$	$P+2E_T$	$2\|S\|P+\|I\|E$	$\|R\|P+(4+2\|R\|)E$

6.2. Experimental Simulation

6.2.1. Experimental Environment

We implemented our ORCP-ABE scheme in Python, and experiments were run on Ubuntu operating system with Intel core i7 3.00GHz and 8GB 2133MHZ LPDDR3 RAM. We set the size of an element in $\mathbb{G}$ and $\mathbb{G}_T$ to 512 bits and established a simulation experiment based on the Charm–Crypto Library V0.50.

6.2.2. Computational Cost

We added 50 unique attributes to the system attribute set. During the key generation phase, users with various attribute sets are randomly generated. For each user, we evaluated the cost of key generation. The average results are shown in Figure 5. We use a zip file of size 1MB as encrypted data in the encryption phase. Then, we tested the encryption cost of the access tree with a different number of attributes. The average results are illustrated in Figure 6. In the decryption phase, we test the decryption cost using different users that satisfy the access structure, and the average results are illustrated in Figure 7. In the attribute revocation phase, we tested the overhead of revoking a single attribute to update a different number of ciphertexts and the overhead of revoking multiple attributes to update a single ciphertext. The results are illustrated in Figure 8.

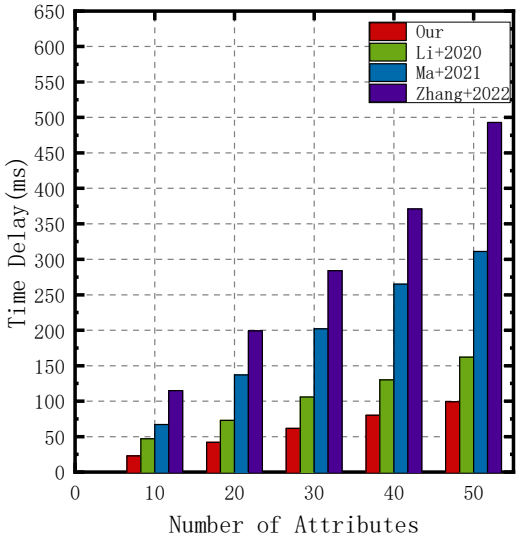

Figure 5. Cost of key generation [23–25].

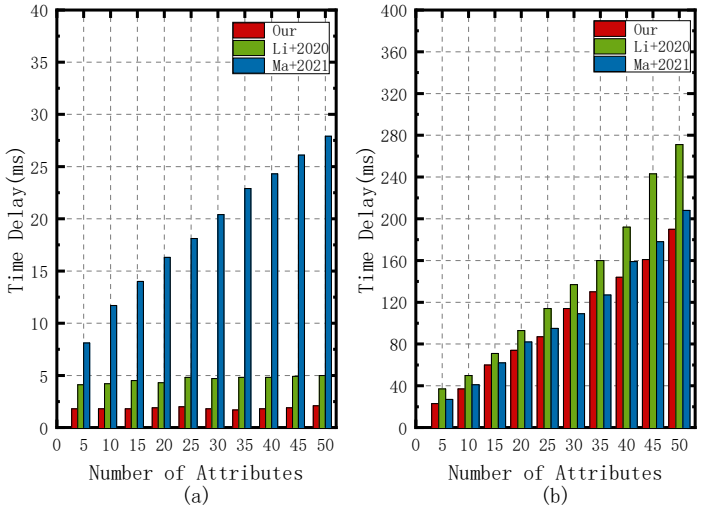

Figure 6. (**a**) Cost of user encryption. (**b**) Cost of RSU encryption [23,24].

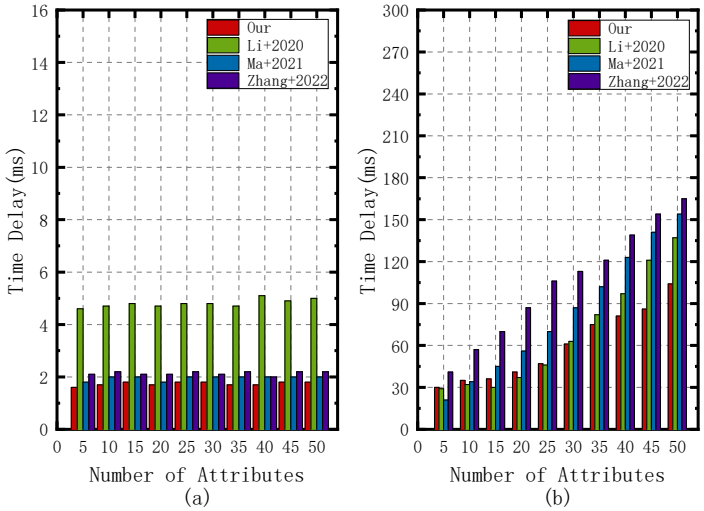

Figure 7. (**a**) Cost of user decryption (**b**) Cost of RSU decryption [23–25].

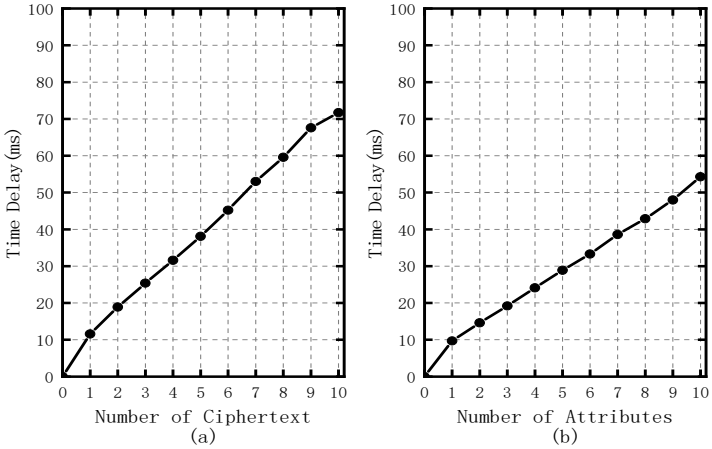

Figure 8. (**a**) Cost of single attribute revocation (**b**) Cost of multi-attribute revocation.

Figure 5 shows that the time overhead of key generation increases as the number of user attributes increases. To know the key generation efficiency of this scheme, we tested the schemes [23–25]. When the number of user attributes is 10, the overhead of this scheme is 24 ms, and the overheads of schemes [23–25] are 42 ms, 67 ms, and 115 ms, respectively.

When the number of user attributes reaches 50, the overhead of the scheme [25] is nearly 500 ms, while the overhead of this scheme is only 99 ms. When the number of user attributes is large, the overhead of this scheme is the smallest.

As shown in Figure 6a, the encryption overhead at the user side of this scheme and scheme [23] is not affected by the number of attributes in the access structure. While the encryption overhead at the user side of the scheme [24] increases with the number of attributes in the access structure. The overhead of this scheme is about 2 ms, and the scheme [23] is about 4.5 ms. As shown in Figure 6b, the encryption overhead at the outer packet side of the three schemes increases with the number of attributes in the access structure. When the number of attributes of the access structure is 5, the overhead of this scheme is 21 ms, and schemes [23,24] are 37 ms and 27 ms, respectively. When the number of attributes of the access structure reaches 50, the overhead of this scheme is 190 ms which is better than schemes [23,24]. There is no outsourcing in the encryption phase of the scheme [25], so it is not compared here.

As shown in Figure 7a, the decryption overhead at the user side of each scheme is not affected by the number of user attributes. The decryption overhead of this scheme is maintained at 1.8 ms, while the overheads of schemes [23–25] are 4.3 ms, 1.8 ms, and 2.1 ms, respectively. As shown in Figure 7b, the decryption overhead of each scheme increases with the number of user attributes at the outer packet side. However, the decryption overheads of each scheme are very similar. When the number of user attributes is more than 30, the decryption overhead of this scheme is slightly less than the other schemes.

As shown in Figure 8a, when revoking a single attribute, the revocation overhead increases with the number of updated ciphertexts. As shown in Figure 8b, when the revocation of a single attribute is updated with only one cipher, the overhead revocation increases with the number of attributes. From the above results, attribute revocation is mainly affected by the number of updated ciphertexts. When the number of updated ciphertexts required for an attribute is small, the overhead of revoking a single attribute is less than 100 ms.

6.2.3. Transaction Cost and Transmission Rate

In blockchain simulation phase, we built the FISCO BCOS [42] consortium blockchain using the build_chain.sh scripts and deployed the IPFS command line version on numerous servers with Ubuntu 20.04 system environment. The consortium blockchain uses PBFT consensus. Therefore, we set the initial number of consensus nodes to 4 to satisfy the 3f+1 requirement [43]. We evaluated the performance of processing a single transaction with different numbers of nodes. Furthermore, we evaluated the model's transactional concurrency capabilities. The file size in the test transaction is 1 MB. The experimental results are shown in Figure 9.

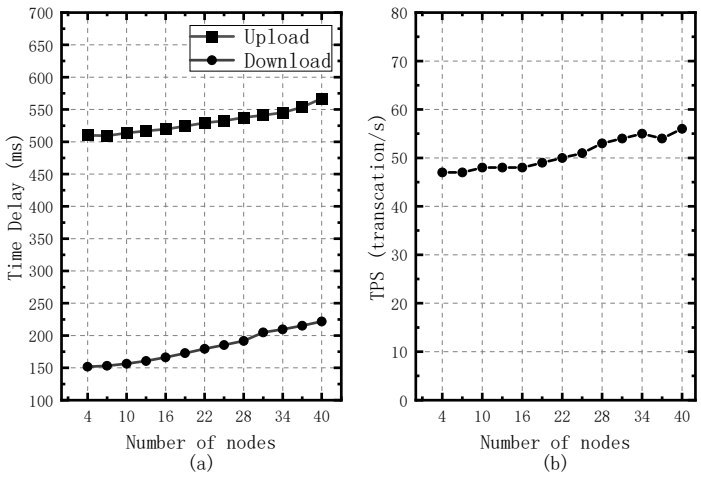

Figure 9. (**a**) Cost of transaction (**b**) Throughput.

As shown in Figure 9a, there are two types of test transactions: data upload confirmation and data download confirmation. The results show that the latency of the two types of transactions increases linearly with the increase of the number of nodes. The increase in delay is mainly due to the increased communication overhead caused by adding nodes in the blockchain to reach a consensus. In addition, the delay of data upload is about 360 ms higher than that of data download. The data upload delay is mainly the overhead of data encryption and metadata storage, and the data download delay is mainly the overhead of data download and data decryption. The difference in latency is because the metadata storage overhead in IPFS is much greater than the data download overhead. As shown in Figure 9b, the transaction throughput of the model is positively linearly related to nodes. The number of nodes is initially 4, and the throughput of the model is 47 transaction/s. When the number of nodes increases to 34, the throughput is 54 transaction/s. The throughput of the model grows slowly. After the number of nodes reaches 34, the throughput of the model roughly stabilizes between 54–56 transaction/s.

In addition, the experiments tested the performance of data file upload and download. Figure 10 shows the time spent on file transfers of different sizes. The experimental file sizes range from 1 MB to 1024 MB. The experimental results show that the time overhead of transfer increases exponentially with the file size. When the file size is less than 16 MB, the transfer time is less than 5 s for uploading and less than 2 s for downloading. The main factors affecting the transfer time are not the network bandwidth but the data encryption and decryption and the metadata storage. When the file size exceeds 16 MB, the average upload transfer rate is about 12.8 MB/s and the average download transfer rate is about 27.6 MB/s. Therefore, the network bandwidth is the main factor affecting the transfer time.

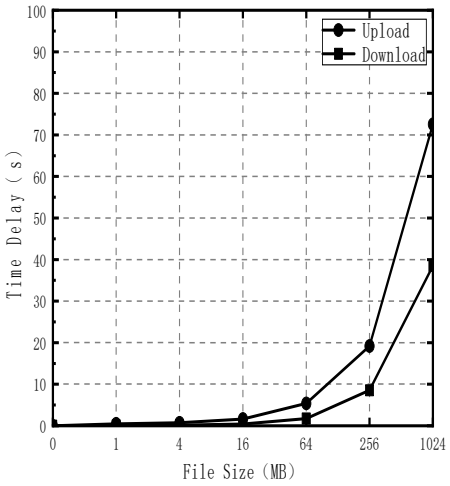

Figure 10. Cost of transmission.

7. Conclusions

This paper studies a safe and efficient data-sharing model for VANET. This model provides a data-sharing platform with privacy-preserving and authorized access by CP-ABE, blockchain, and IPFS. It has no single point of failure and can undo user-level attributes. Experiments show that our scheme has certain advantages compared with other schemes in data encryption and data decryption at the user end. In the scheme, the revocation overhead of a single attribute of a user is relatively low. The transaction processing delay of the model is short and has certain concurrency capabilities. The model is proved to be IND-CPA safe in the game under the CDH assumption. In future work, we will optimize the on-chain information storage. We are considering adopting an editable blockchain to reduce old blocks with invalid information. In addition, we will increase the concurrency performance of our model by improving the consensus mechanism.

Author Contributions: Conceptualization, X.C.; methodology, X.C., Y.C. and K.F.; software, X.C. and X.W.; validation, Y.C.; formal analysis, Y.C.; investigation, X.W.; resources, X.Z. and K.F.; data curation, X.C.; writing—original draft preparation, X.C.; writing—review and editing, Y.C.; visualization, X.W.; supervision, K.F.; project administration, Y.C.; funding acquisition, X.Z. and K.F. All authors have read and agreed to the published version of the manuscript.

Funding: This research was funded by Key Research and Development Project of Hunan Province grant number 2020NK2033 and the APC was funded by project 2020NK2033.

Informed Consent Statement: Not applicable.

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

References

1. Yousefi, S.; Mousavi, M.S.; Fathy, M. Vehicular ad hoc networks (VANETs): Challenges and perspectives. In Proceedings of the 6th International Conference on ITS Telecommunications, Chengdu, China, 21–23 June 2006; pp. 761–766.
2. Lin, J.; Yu, W.; Zhang, N.; Yang, X.; Zhang, H.; Zhao, W. A survey on internet of things: Architecture, enabling technologies, security and privacy, and applications. *IEEE Internet Things* **2017**, *4*, 1125–1142. [CrossRef]
3. Engoulou, R.G.; Bellaïche, M.; Pierre, S.; Quintero, A. VANET security surveys. *Comput. Commun.* **2014**, *44*, 1–13. [CrossRef]
4. Al-Sultan, S.; Al-Doori, M.M.; Al-Bayatti, A.H.; Zedan, H. A comprehensive survey on vehicular ad hoc network. *Netw. Comput. Appl.* **2014**, *37*, 380–392. [CrossRef]
5. Deng, J.; Hu, J.L.; Liu, A.C.M.; Wu, J. Research and application of cloud storage. In Proceedings of the 2010 2nd International Workshop on Intelligent Systems and Applications, Wuhan, China, 22–23 May 2010; pp. 1–5.
6. Gao, W.; Hatcher, W.G.; Yu, W. A survey of blockchain: Techniques, applications, and challenges. In Proceedings of the 27th International Conference on Computer Communication and Networks (ICCCN), Hangzhou, China, 30 July–2 August 2018; pp. 1–11.
7. Zaghloul, E.; Li, T.; Mutka, M.W.; Ren, J. Bitcoin and blockchain: Security and privacy. *IEEE Internet Things* **2020**, *7*, 10288–10313. [CrossRef]
8. Jiang, T.; Fang, H.; Wang, H. Blockchain-based internet of vehicles: Distributed network architecture and performance analysis. *IEEE Internet Things* **2018**, *6*, 4640–4649. [CrossRef]
9. Aujla, G.S.; Singh, A.; Singh, M.; Sharma, S.; Kumar, N.; Choo, K.K.R. BloCkEd: Blockchain-based secure data processing framework in edge envisioned V2X environment. *IEEE Trans. Veh. Technol.* **2020**, *69*, 5850–5863. [CrossRef]
10. Daemen, J.; Rijmen, V. *The Design of Rijndael*; Springer: New York, NY, USA, 2002.
11. Kang, J.; Yu, R.; Huang, X.; Wu, M.; Maharjan, S.; Xie, S.; Zhang, Y. Blockchain for secure and efficient data sharing in vehicular edge computing and networks. *IEEE Internet Things* **2018**, *6*, 4660–4670. [CrossRef]
12. Li, M.; Zhu, L.; Lin, X. Efficient and privacy-preserving carpooling using blockchain-assisted vehicular fog computing. *IEEE Internet Things* **2018**, *6*, 4573–4584. [CrossRef]
13. Yao, Y.; Chang, X.; Mišić, J.; Li, L. BLA: Blockchain-assisted lightweight anonymous authentication for distributed vehicular fog services. *IEEE Internet Things* **2019**, *6*, 3775–3784. [CrossRef]
14. Rivest, R.L.; Shamir, A.; Adleman, L. A method for obtaining digital signatures and public-key cryptosystems. *Commun. ACM* **1978**, *21*, 120–126. [CrossRef]
15. Sahai, A.; Waters, B. Fuzzy identity-based encryption. In Proceedings of the Annual International Conference on the Theory and Applications of Cryptographic Techniques, EUROCRYPT, Aarhus, Denmark, 22–26 May 2005; pp. 457–473.
16. Goyal, V.; Pandey, O.; Sahai, A.; Waters, B. Attribute-based encryption for fine-grained access control of encrypted data. In Proceedings of the 13th ACM Conference on Computer and Communications Security, New York, NY, USA, 30 October 2006; pp. 89–98.
17. Bethencourt, J.; Sahai, A.; Waters, B. Ciphertext-policy attribute-based encryption. In Proceedings of the IEEE Symposium on Security and Privacy (SP'07), Berkeley, CA, USA, 20–23 May 2007; pp. 321–334.
18. Benet, J. IPFS-content addressed, versioned, P2P file system. *arXiv* **2014**, arXiv:1407.3561.
19. Mahi, M.J.N.; Chaki, S.; Ahmed, S.; Biswas, M.; Kaiser, S.; Islam, M.S.; Sookhak, M.; Barros, A.; Whaiduzzaman, M. A review on VANET research: Perspective of recent emerging technologies. *IEEE Access* **2022**, *6*, 65760–65783. [CrossRef]
20. Lee, M.; Atkison, T. VANET applications: Past, present, and future. *Veh. Commun.* **2021**, *28*, 100310. [CrossRef]
21. Deng, X.; Gao, T.; Guo, N.; Qi, J.; Zhao, C. PAS: Privacy-Preserving Authentication Scheme Based on SDN for VANETs. *Appl. Sci.* **2022**, *12*, 4791. [CrossRef]
22. Chen, J.; Li, K.; Philip, S.Y. Privacy-Preserving Deep Learning Model for Decentralized VANETs Using Fully Homomorphic Encryption and Blockchain. *IEEE Trans. Intell. Transp. Syst.* **2022**, *23*, 11633–11642. [CrossRef]
23. Li, H.; Pei, L.; Liao, D.; Chen, S.; Zhang, M.; Xu, D. FADB: A fine-grained access control scheme for VANET data based on blockchain. *IEEE Access* **2020**, *8*, 85190–85203. [CrossRef]
24. Ma, J.; Li, T.; Cui, J.; Ying, Z.; Cheng, J. Attribute-based secure announcement sharing among vehicles using blockchain. *IEEE Internet of Things* **2021**, *8*, 10873–10883. [CrossRef]

25. Zhang, L.; Zhang, Y.; Wu, Q.; Mu, Y.; Rezaeibagha, F. A Secure and Efficient Decentralized Access Control Scheme Based on Blockchain for Vehicular Social Networks. *IEEE Internet Things* **2022**, *11*, 86. [CrossRef]
26. Nakamoto, S. Bitcoin: A Peer-to-Peer Electronic Cash System. *Decent Bus. Rev.* **2008**, 21260. Available online: https://bitcoin.org/bitcoin.pdf (accessed on 7 December 2022).
27. Shen, B.; Guo, J.; Yang, Y. MedChain: Efficient healthcare data sharing via blockchain. *Appl. Sci.* **2019**, *9*, 1207. [CrossRef]
28. Ren, Y.; Zhu, F.; Qi, J.; Wang, J.; Sangaiah, A.K. Identity management and access control based on blockchain under edge computing for the industrial internet of things. *Appl. Sci.* **2019**, *9*, 2058. [CrossRef]
29. Bodkhe, U.; Tanwar, S.; Parekh, K.; Khanpara, P.; Tyagi, S.; Kumar, N.; Alazab, M. Blockchain for industry 4.0: A comprehensive review. *IEEE Access* **2020**, *8*, 79764–79800. [CrossRef]
30. Shafagh, H.; Burkhalter, L.; Hithnawi, A.; Duquennoy, S. Towards blockchain-based auditable storage and sharing of IoT data. In Proceedings of the 2017 on Cloud Computing Security Workshop, New York, NY, USA, 3 November 2017; pp. 45–50.
31. Zhang, X.D.; Li, R.; Cui, B. A security architecture of VANET based on blockchain and mobile edge computing. In Proceedings of the 2018 1st IEEE International Conference on Hot Information-Centric Networking (HotICN), Shenzhen, China, 15–17 August 2018; pp. 258–259.
32. Javaid, U.; Aman, M.N.; Sikdar, B. DrivMan: Driving trust management and data sharing in VANETS with blockchain and smart contracts. In Proceedings of the 2019 IEEE 89th Vehicular Technology Conference (VTC2019-Spring), Kuala Lumpur, Malaysia, 28 April–1 May 2019; pp. 1–5.
33. Waters, B. Ciphertext-policy attribute-based encryption: An expressive, efficient, and provably secure realization. In Proceedings of the International Workshop on Public Key Cryptography, Taormina, Italy, 6–9 March 2011; pp. 53–70.
34. Green, M.; Hohenberger, S.; Waters, B. Outsourcing the Decryption of ABE Ciphertexts. In Proceedings of the 20th USENIX Security Symposium (USENIX Security 11), San Francisco, CA, USA, 10–12 August 2011.
35. Lewko, A.; Waters, B. Decentralizing attribute-based encryption. In Proceedings of the Annual International Conference on The Theory and Applications of Cryptographic Techniques, EUROCRYPT, Tallinn, Estonia, 15–19 May 2011; pp. 568–588.
36. Premkamal, P.K.; Pasupuleti, S.K.; Alphonse, P.J.A. Dynamic traceable CP-ABE with revocation for outsourced big data in cloud storage. *Commun. Syst.* **2021**, *34*, e4351. [CrossRef]
37. Sethi, K.; Pradhan, A.; Bera, P. PMTER-ABE: A practical multi-authority CP-ABE with traceability, revocation and outsourcing decryption for secure access control in cloud systems. *Clust. Comput.* **2021**, *24*, 1525–1550. [CrossRef]
38. Wu, Y.; Zhang, W.; Xiong, H.; Qin, Z.; Yeh, K.H. Efficient access control with traceability and user revocation in IoT. *Multimed. Tools Appl.* **2021**, *80*, 31487–31508. [CrossRef]
39. Yang, Y.; Sun, J.; Liu, Z.; Qiao, Y. Practical revocable and multi-authority CP-ABE scheme from RLWE for Cloud Computing. *Inf. Secur. Appl.* **2022**, *65*, 103108. [CrossRef]
40. Liu, X.; Zheng, Y.; Li, X. A revocable attribute-based access control system using blockchain. In Proceedings of the 3rd International Conference on Electronic Engineering and Informatics (EEI 2021), Dali, China, 18–20 June 2021; Volume 1971.
41. Hur, J.; Noh, D.K. Attribute-based access control with efficient revocation in data outsourcing systems. *IEEE Trans. Parallel Distrib. Syst.* **2010**, *22*, 1214–1221. [CrossRef]
42. Fisco-Bcos Homepage. Available online: http://www.fisco-bcos.org/ (accessed on 7 December 2022).
43. Castro, M.; Liskov, B. Practical Byzantine fault tolerance and proactive recovery. *ACM Trans. Comput. Syst. TOCS* **2002**, *20*, 398–461. [CrossRef]