Verifiable Attribute-Based Proxy Re-encryption with Non-repudiation Based on Blockchain

Yaorui He[1], Ting Liang[1], Pei Huang[1], and Zhe Xia[1,2(✉)]

[1] School of Computer Science and Artificial Intelligence, Wuhan University of Technology, Wuhan 430070, China
xiazhe@whut.edu.cn

[2] Hubei Key Laboratory of Transportation Internet of Things, Wuhan University of Technology, Wuhan 430071, China

Abstract. With the development of Blockchain, cloud computing, and artificial intelligence, smart transportation is highly likely to drive urban transportation in the direction of intelligence and digitalization. However, when connected vehicle devices share data in edge networks, there are security and privacy issues, e.g. leakage of sensitive information will cause serious threats. The separation of user data ownership and physical control requires users to impose access control on the outsourced data. Existing privacy protection schemes still suffer from problems such as a lack of supervision of centralized third-party cloud servers, which face the challenge of data leakage. In this paper, Blockchain and proxy re-encryption techniques are used to address these challenges. To enable sharing of the outsourced data in a secure way, Attribute-Based Proxy Re-encryption (ABPRE) is a popular technique. At the same time, verifiability enables the shared user to verify that the re-encrypted ciphertext returned by the server is correct. In addition, the introduction of a decentralized Blockchain provides distributed storage and it is tamper-proof for enhancing the trustworthiness of the connected vehicle devices as well as the security of the communication between the entities. The security and performance analyses demonstrate that our proposed scheme can achieve the desirable security features and it is efficient for protecting data privacy.

Keywords: Blockchain · Proxy Re-Encryption · Privacy-Preserving Data-Sharing

1 Introduction

The Intelligent Transportation System(ITS) [15] is a new information technology that integrates the Internet of Things (IoT), spatial awareness, cloud computing, and mobile Internet in the field of transportation. The ITS structure can be divided into three layers: the perception layer, the network layer, and the

J. Chen and Z. Xia (Eds.), BlockTEA 2023, LNICST 577, pp. 115–134, 2024.
https://doi.org/10.1007/978-3-031-60037-1_7

application layer. Among them, the perception layer includes vehicle nodes and the sensors configured around them, such as GPS and cameras, to collect vehicle driving data in real-time. The network layer is responsible for transmitting information and realizing data transmission and sharing between entities such as vehicle-to-vehicle (V2V) and vehicle-to-infrastructure(V2I) through wireless communications such as Zigbee, WiFi, and DSRC [13]. Service providers collect vehicle data and analyze and process them through cloud computing technology, and provide services to users at the application layer to improve the traffic environment and urban operation efficiency.

Data sharing and cooperation are key to achieving smart transportation and the Internet of Vehicles(IoV). Communication networks between vehicles (e.g. V2V) can share information such as real-time traffic conditions, vehicle location, and speed. Through direct data sharing, vehicles understand each other's status, helping to optimize route selection, avoid congestion, and provide early warning and emergency response, among other things [3]. Data sharing between vehicle manufacturers and related service providers allows third-party developers and applications to access and utilize vehicle data, facilitating innovation and the development of various connected vehicle applications such as navigation software, vehicle-sharing services, etc. Open data interfaces can follow standardized data formats and protocols to ensure data interoperability and security [18]. In general, participants can enter into data cooperation and sharing agreements that specify how data is to be shared permissions, and privacy protection [2]. This can involve cooperation between vehicle manufacturers, transportation authorities, city planners, insurance companies, and other parties. These data-sharing and cooperation methods can be achieved through various technical means, such as V2V, V2I, IoT technologies, cloud computing, etc.

While cloud-assisted IoV data sharing offers many conveniences and innovative features, it also raises a number of security and privacy concerns. Data from vehicles is transferred to cloud servers for processing and storage, which leads to having to take the risks associated with data centralization. In the event of an attack or failure of the cloud server, a large amount of vehicle data may be lost or unavailable [12]. Also, vehicle data contains personal sensitive information such as driving habits and destination information of the vehicle owner, which is at risk of data leakage, data tampering, or unauthorized access [21]. With the separation of data ownership and storage, data owners have a strong incentive to maintain control over their use and access to shared data.

To address these issues, a number of security and privacy protection measures are required, such as

- Data privacy: The vehicle data is encrypted before uploading to the cloud to ensure data privacy.
- Access control: Restrict access to vehicle data by other entities and authorize only to trusted service providers and authorized users.
- Compliance review: Ensure that data is shared in compliance with applicable laws and regulations and that privacy protection requirements are observed.

– Security audits: Audits and vulnerability scans for cloud-assisted Internet of Vehicles security to identify and fix potential security issues in a timely manner.

In this paper, Blockchain and proxy re-encryption technologies are integrated to address the above privacy and security problems. Blockchain is a decentralized ledger that ensures security and trustworthiness of data through cryptographic means. Blockchain uses techniques such as public key encryption and hash functions to verify and protect the integrity of data, while consistency of the ledger is ensured by consensus algorithms. Blockchain is best known in the field of cryptocurrencies, such as Bitcoin [16] and Ether, but also has many other application areas, such as supply chain management, smart contracts, etc. Blockchain and the Internet of Vehicles are two different but widely used technology areas. Their technological convergence can bring many advantages to the intelligent transportation system and automotive industry [5]. Proxy Re-Encryption [1] is a novel technique to allow a third party to transform some encrypted data from one public key to another public key without learning the underneathing plaintext. In IoV, attribute-based proxy re-encryption schemes can help enable secure and privacy-preserving data sharing. However, existing privacy protection schemes still suffer from problems such as a lack of supervision of centralized third-party cloud servers [9]. Since ciphertext processing consumes cloud computing resources, some inactive cloud servers may even return the wrong ciphertext to save computing resources.

The integration of Blockchain and proxy re-encryption can enhance data protection and access control. Access and sharing of data on the Blockchain is controlled through the use of proxy re-encryption, allowing only specific authorized users to decrypt and access the data. At the same time, the Blockchain ensures that the data stored by the user is not tampered with. This combination provides a more robust permission management and privacy protection mechanism, while still retaining the decentralized and trustworthy characteristics of the Blockchain.

The main contributions of this paper are as follows.

1. First of all, this paper adopts a privacy protection scheme based on proxy re-encryption to realize the data sharing of the Internet of Vehicles, which can simplify the sharing process between entities, and users do not need to download the ciphertext to decrypt it before re-encrypting and uploading it.
2. Secondly, the scheme proposed in this paper provides verifiability, which means that users can directly verify the legitimacy of the re-encrypted ciphertext without having to perform complex calculations.
3. Finally, this paper combines Blockchain technology to achieve persistent storage, provide evidence for arbitration, and solve the problems that centralized cloud servers are vulnerable to single-point attacks and data tampering.

The remainder of the paper is organized as follows: Sect. 2 reviews the state of the art of relevant research addressing the problem considered in this paper. Section 3 presents the cryptographic primitives and basic definitions involved in

implementing the scheme proposed in this paper. Section 4 states the various models and definitions involved in the scheme of this paper. Section 5 shows the details of the proposed scheme and the protocol flow. Section 6 proves the security of the scheme in this paper. Section 7 provides a comparative analysis of the schemes related to this paper. Section 8 concludes the paper and provides an outlook on future developments.

2 Related Work

Proxy Re-Encryption allows a third-party, called the proxy server, to transform ciphertexts without revealing the original key [4]. This technique can be used to securely share encrypted data and has important applications in privacy protection and data security.

Proxy re-encryption techniques can realize a variety of functions, such as transformation and decryption of proxy re-encryption, delegation, and revocation of proxy re-encryption. Various proxy re-encryption schemes have been proposed, including identity-based schemes [8,10,19], certificate-less schemes [20], and attribute-based schemes [14] etc. Researchers endeavor to propose proxy re-encryption schemes with strong security guarantees and verify them with formal security proofs [11]. In order to detect whether the proxy is malicious, Ohata et al. [17] proposed a mechanism to verify the output of the proxy, but it has not considered fine-grained access control. Liang et al. [14] introduced the concept of proxy re-encryption into attribute-based encryption schemes, thereby achieving flexible control over encrypted data. Ge et al. [9] proposed a verifiable and fair ciphertext-policy attribute-based proxy re-encryption scheme to prevent malicious cloud servers from forging ciphertext as their malicious behavior can be identified. PRE is widely used in cloud computing, data sharing, and multi-party secure computing. Researchers are committed to improving the efficiency, security, and functionality of proxy re-encryption to meet the needs of different application scenarios.

Overall, proxy re-encryption techniques are an active research area, and researchers are continuously improving and enhancing their efficiency and security. With the increased demand for data security and privacy protection, proxy re-encryption techniques are widely used in more application scenarios.

3 Preliminaries

This section introduces the cryptographic primitives needed to construct our scheme as well as the Blockchain basics to make it easier for the reader to understand our work.

3.1 Attribute-Based Proxy Re-Encryption

A proxy re-encryption scheme is a cryptographic technique used to implement the transformation and re-encryption of ciphertext without exposing the original key. This technique can be used to securely share encrypted data and has

important applications in privacy protection and data security. The following are the basic elements of the proxy re-encryption scheme.

The PRE scheme involves multiple entities, including:

- delegator: It is the data owner that encrypts the plaintext using the public key.
- proxy: It is an intermediate third party that performs the re-encryption operation to convert the ciphertext from the delegator's public key to the delegatee's public key.
- delegatee: It is the intended recipient of the message, who has the corresponding private key to decrypt the re-encrypted ciphertext.

An ABPRE scheme is a tuple of PPT algorithms (SETUP, KEYGEN, RKGEN, ENC, REENC, DEC).

1. $SETUP(1^k) \rightarrow (params, mk)$: takes as inputs a security parameter 1^k, and it outputs the public parameter $params$ and the master key mk.
2. $KEYGEN(S, mk) \rightarrow (usk)$: takes as inputs an attribute set S and the master key mk, and it outputs a secret key usk.
3. $ENC(AS, m) \rightarrow (C)$: takes as inputs an access structure AS and a message m, and it outputs a ciphertext C.
4. $RKGEN(usk, AS) \rightarrow (rk)$: takes as inputs a secret key usk and an access structure AS, and it outputs a re-key rk.
5. $REENC(rk, C) \rightarrow (C')$: takes as inputs a re-key rk and a ciphertext C, and it checks whether the access structure of C is satisfied by the attribute set in rk. If yes, it outputs a re-encrypted ciphertext C', or it outputs "reject" otherwise.
6. $DEC(usk, C) \rightarrow (m)$: takes as inputs a secret key usk and a ciphertext C, and it checks if the access structure of C is satisfied by the attribute set in usk. If yes, it outputs a message m in the message space, or it outputs "reject" otherwise.

Correctness. This property captures two requirements, i.e. for any message m in the message space, the following two equations should hold:

1. $DEC(usk_S, ENC(AS, m)) = m$;
2. $DEC(usk_{S'}, REENC(rk_{AS \rightarrow AS'}, C)) = m$.

where S satisfies AS, S' satisfies AS', mk is the master key, C is the ciphertext associate with the message m and the access structure AS'.

3.2 Blockchain

Blockchain technology is a distributed ledger, and its main features are as follows:

1. Distributed ledger: Blockchain is a decentralized ledger that is maintained and shared by multiple entities. Each has a complete copy of all transaction records and block information. Distributed ledgers are characterized by decentralization, openness and transparency, security, and trust.

2. Block and chain: the block is a basic unit in a Blockchain, containing a series of transaction records as well as some metadata. Each block contains a hash value that points to the previous block, forming a chain structure. And this is why it is called Blockchain. New blocks are inserted to the end of the chain by a specific consensus algorithm and its main feature is tamper-proof.
3. Distributed Consensus: Blockchain solves the problem of consistency among participants using distributed consensus algorithms. The consensus algorithm ensures that all participants agree on the state of the ledger without a trusted authority. Proof of Work (PoW) and Proof of Stake (PoS) are commonly used consensus algorithms.
4. Cryptography: Blockchain uses cryptographic techniques, such as digital signature and hash function, to ensure the security and privacy protection of data. Hash functions are used to generate unique identifiers for blocks, and digital signatures are used to prove the legitimacy of transactions.
5. Decentralized applications: Blockchain can support decentralized applications (DApps), which are applications built on the Blockchain that perform various functions, such as digital asset trading, smart contract execution, data storage, validation, etc. Decentralized applications enable trust and security without the need for intermediaries.
6. Smart Contracts: They are automated contracts executed on the Blockchain according to some predefined rules and conditions. Smart contracts can automatically execute and validate transactions without a trusted third party. They can also implement complex logic such as asset transfers, conditional payments, voting, etc.

The practical applications of Blockchain technology are very wide, covering a wide range of fields such as smart transportation, smart healthcare, and industrial Internet of Things. With the development and innovation of technology, Blockchain will continue to drive the digital economy and social change.

4 Models and Definitions

4.1 System Model

A verifiable attribute-based proxy re-encryption system includes the following entities, key generates center, data owner, original recipient, a cloud server that acts as a proxy, and a shared user. As shown in Fig. 1.

- Key Generation Center: KGC is responsible for establishing the cryptosystem, publishing the system's public parameters, and saving the master key. It accepts user (DO, DR, SU) registration requests, generates user private keys, and sends them to users via secure channels.
- Data Owners: DO is responsible for collecting plaintext data, which needs to be encrypted to protect user privacy before uploading to the Blockchain.
- Data Recipient: DR can access the ciphertext on the Blockchain and it can use its private key to recover the plaintext. Moreover, DR can share the ciphertext with other users.

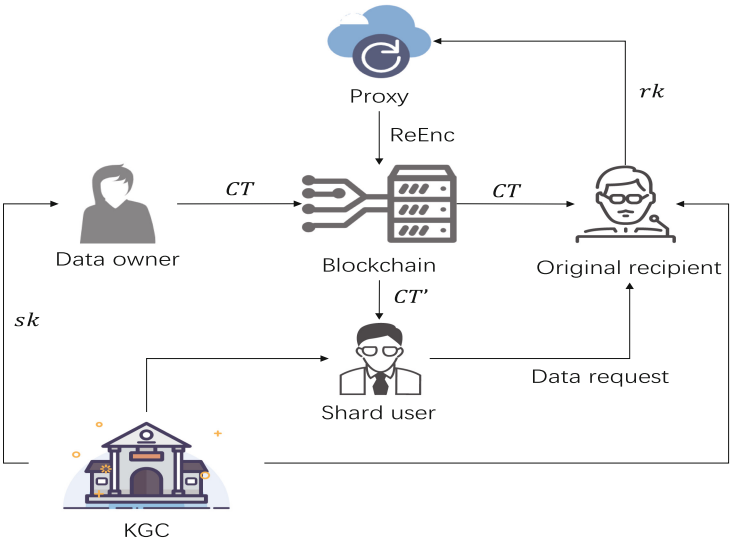


Fig. 1. System model

- Proxy: The Proxy is responsible for handling data-sharing requests. The DR generates the re-encryption key, which is used by the Proxy to re-encrypt the original ciphertext and re-upload the processing result to the Blockchain.
- Shared User: SU can request access to the ciphertext on the Blockchain, first he needs to make an access request to the original data recipient, and after being authorized, he can download the re-encrypted ciphertext from the Blockchain and decrypt it with his own private key.

4.2 Threat Model

In our threat model, the DR and SU are both assumed to be honest-but-curious.

- An external adversary without a valid key, including a proxy, attempts to obtain information through the ciphertext data on the Blockchain.
- Proxy and unauthorized shared users may conspire to attempt to obtain other data information, even private keys, from the original recipient.
- Proxy may reuse previous re-encrypted ciphertexts, or randomly select re-encrypted ciphertexts to save costs.

4.3 Security Requirements

The following security requirements are considered in our proposed scheme.

Selective-Structure Chosen Plaintext Security. An ABPRE scheme is said to be secure against selective-structure chosen plaintext attack, if no PPT adversary $\mathcal{A}$ can win the SS-CPA game with non-negligible advantage.

Init: $\mathcal{A}$ chooses an access structure AS^* and sends it to the challenger, who then runs $SETUP(1^k)$ and sends the public parameter $params$ back to $\mathcal{A}$. The challenger keeps the master key mk private.

Phase 1: The adversary $\mathcal{A}$ issues the following queries to the oracles:

- Key generation query $\mathcal{O}kg$: Once receiving an attribute set S, if S does not satisfy AS^*, it outputs a secret key $usk = KEYGEN(S, mk)$, or it outputs "reject" otherwise.
- Re-key generation query $\mathcal{O}rkg$: Once receiving an attribute set S and an access structure AS, if S does not satisfy AS^*, it outputs a re-key $rk = RKGEN(KEYGEN(S, mk), AS)$, or it outputs "reject" otherwise.
- Re-encryption query $\mathcal{O}re$: Once receiving an attribute set S, an access structure AS and a ciphertext C, if S does not satisfy AS^* and S satisfies the access structure of C, it outputs a ciphertext $C' = REENC(RKGEN(KEYGEN(S, mk), AS), C)$, or it outputs "reject" otherwise.

Challenge: When the above phase is over, $\mathcal{A}$ outputs two messages m_0, m_1 with equal length. The challenger randomly chooses $b \in \{0, 1\}$ and encrypts m_b with AS^*. Then, the ciphertext C^* is sent to $\mathcal{A}$.

Phase 2: The same as in Phase 1.

Guess: $\mathcal{A}$ guesses a bit $b' \in \{0, 1\}$, and it wins the game if $b' = b$.

$\mathcal{A}$'s advantage in SS-CPA game is defined as follows:

$$Adv_{SS-CPA,\mathcal{A}}^{\mathcal{O}kg,\mathcal{O}rkg,\mathcal{O}ree} = |Pr[b' = b] - 1/2| \tag{1}$$

Key Security for ABPRE. An ABPRE scheme enjoys the key security property if no PPT adversary $\mathcal{A}$ can win the following MKS game with a non-negligible advantage.

Init: $\mathcal{A}$ randomly chooses an attribute set S^* for the challenge and sends it to the challenger, who then runs $SETUP(1^k)$ and sends the public parameters $params$ back to $\mathcal{A}$. The challenger keeps the master-key mk private.

Queries: $\mathcal{A}$ issues the following queries to the oracles:

- Key generation query $\mathcal{O}kg$: Once receiving an attribute set S, if $S \neq S^*$, it outputs a secret key $usk = KEYGEN(S, mk)$, or it outputs "reject" otherwise.
- Re-key generation query $\mathcal{O}rkg$: Once receiving an attribute set S and an access structure AS, it outputs $rk = RKGEN(KEYGEN(S, mk), AS)$.

Output: Note that re-encryption query and decryption query are straightforward, because the re-key can be generated by querying the re-key generation oracle. Finally, $\mathcal{A}$ outputs the secret key usk^* for attribute set $S*$.

$\mathcal{A}$'s advantage for the above MKS game is defined as

$$Adv_{MKS,\mathcal{A}}^{\mathcal{O}kg,\mathcal{O}rkg} = |Pr[\mathcal{A}\ succceeds]| \tag{2}$$

Verifiability. An ABPRE scheme satisfies the verifiability property if no PPT adversary $\mathcal{A}$ can win the Ver game with a non-negligible advantage.

Init: The challenger runs $SETUP(1^k)$ and gives $\mathcal{A}$ the system parameters $params$. It keeps the corresponding master-key mk to itself.

Phase 1: The adversary $\mathcal{A}$ issues queries to the oracles:

- Key generation query $\mathcal{O}kg$: Once receiving an attribute set S, it outputs a secret key $usk = KEYGEN(S, mk)$.
- Re-key generation query $\mathcal{O}rkg$: Once receiving an attribute set S and an access structure AS, it outputs a re-key $rk = RKGEN(KEYGEN(S, mk), AS)$.
- Claim query $\mathcal{O}claim$: Suppose the claim query is issued on (CT, rk, CT'). The challenger outputs $CLAIM(CT, rk, CT')$.

Challenge: $\mathcal{A}$ chooses a message m^* and some access policy (AS^*, ρ^*), and sends them to challenger, who then computes $CT^* = Enc(m^*; (AS^*; \rho^*))$ and sends CT^* back to $\mathcal{A}$.

Phase 2: The same as in Phase 1.

Guess: $\mathcal{A}$ outputs an attribute set S^* as well as a re-encrypted ciphertext CT'^*, where $R(S^*; (AS^*; \rho^*)) = 1$. $\mathcal{A}$ wins the game if $Dec_{re}(AS^*; CT^*; CT'^*) \notin \{m^*, \perp\}$.

$\mathcal{A}$'s advantage of winning the above game is defined as

$$Adv_{\mathcal{A}}^{Ver} = |Pr[\mathcal{A}\ succceeds]| \tag{3}$$

4.4 Assumptions

Bilinear Map. Let G_1 and G_2 be two multiplicative cyclic groups of prime order p. Let g be a generator of G_1 and e be a bilinear map, $e : G_1 \times G_1 \rightarrow G_2$. The bilinear map e has the following properties:

1. Bilinearity: If for all $u, v \in G_1$ and $a, b \in Z_p$, we have $e(u^a, v^b) = e(u, v)^{ab}$, Or $e(u \cdot v, w) = e(u, w) \cdot e(v, w)$ and $e(u, v \cdot w) = e(u, v) \cdot e(u, w)$, then the mapping is said to be bilinear.

2. Non-degeneracy: The mapping does not map all pairs of elements in $G_1 \times G_1$ to unit elements in G_2. Since G_1, and G_2 are all groups of order prime, this implies that: if g is the generating element of G_1, then $e(g, g)$ is the generating element of G_2.
3. Computability: For any $u, v \in G_1$, an efficient algorithm exists to compute $e(u, v)$.

Decisional Parallel Bilinear Diffie-Hellman Exponent Assumption. Denote (e, G_1, G_2, p, g) as a bilinear pairing, randomly choose $a, s, b_1, ..., b_q \in Z_p$ and g is a generator of G_1. Denote y as

$$g, g^s, g^a, ..., g^{(a^q)}, g^{(a^{q+2})}, ..., g^{(a^{2q})},$$
$$\forall_{1 \le j \le q}, g^{s \cdot b_j}, g^{a/b_j}, ..., g^{(a^q/b_j)}, g^{(a^{q+2}/b_j)}, g^{(a^{2q}/b_j)},$$
$$\forall_{1 \le j,k \le q, k \ne j}, g^{a \cdot s \cdot b_k/b_j}, g^{(a^q \cdot s \cdot b_k/b_j)}$$

the q-parallel BDHE assumption requires that for any PPT adversary $\mathcal{A}$, its advantage of distinguishing $e(g, g)^{a^{q+1} \cdot s} \in G_2$ from a random element in G_2 is negligible. Formally, the advantage of $\mathcal{A}$ is

$$Adv^{BDHE} = |Pr[\mathcal{B}(y, T = e(g, g)^{a^{q+1} \cdot s}) = 0] - Pr[\mathcal{B}(y, T = R) = 0]|.$$

Discrete Logarithm Assumption. Denote (e, G_1, G_2, p, g) as a bilinear pairing, for a tuple $(e, G_1, G_2, p, g, g^\beta)$ where $g \in G_1, \beta \in Z_p^*$, the discrete logarithm assumption requires that for any PPT adversary $\mathcal{A}$, its advantage of computing the integer β is negligible. Formally, the advantage of $\mathcal{A}$ is

$$Adv^{DL} = Pr[\mathcal{A}(e, G_1, G_2, p, g, g^\beta) = \beta].$$

5 Proposal Scheme

In this section, we first introduce a verifiable attribute-based proxy re-encryption scheme, which mainly consists of eight algorithms. First, the authorization center (KGC) generates the public parameters and keeps the master key secret. Participants then register (DO, DR, SU) with the authorization center. This requires these participants to authenticate their identities, and then they can obtain the private keys associated with their attributes. The data owner collects the data, encrypts it according to the access policies, and uploads it to the Blockchain. Therefore, only the users whose attributes satisfy the access policies can access the data. Moreover, shared users can request access to the data from the data owner. The data owner validates the request, generates the re-encryption key and sends it to the proxy server. The authorized proxy server can perform a re-encryption operation on the original ciphertext and upload the re-encrypted ciphertext to the Blockchain. Shared users can access the re-encrypted ciphertext using their own private keys. The shared user can verify the correctness of the ciphertext after decryption. To describe our proposed scheme clearer, we give the description of used notations in Table 1.

Table 1. Symbols and descriptions

Notation	Description
U, S	Attribute set
$AS = (\mathbb{A}, \rho)$	Access structure
$params$	System public parameters
mk	Master key, kept by the KGC
sk_S	Private key for S
m	Plaintext
CT, CT'	Ciphertext
$rk_{AS \to AS'}$	Re-encryption key from AS to AS'

5.1 Concrete Structure

The attribute-based proxy re-encryption scheme extends the attribute-based encryption scheme to provide proxy re-encryption capability while protecting the user's key and plaintext, consisting of eight algorithms, including

$$(SETUP, KEYGEN, ENC, DEC, RKGEN, REENC, DEC_{re}, CLAIM).$$

$SETUP(1^k, U) \to (params, mk)$*:* On input of security parameters and an attribute set, the KGC generates system public parameters and Keeps the master key secret.

- Generate a tuple (e, G_1, G_2, g, p), where G_1, G_2 are two finite cyclic group of order p and g is the generator of G_1. The message space is denoted as $\{0,1\}^k$
- Choose $\alpha, a \in Z_p^*, f_1, f_2, ..., f_U, u, v, h \in G_1$ randomly.
- Select two hash function $H_1 : G_2 \to \{0,1\}^{2k}, H_2 : \{0,1\}^* \to Z_p^*$.
- Compute $mk = g^\alpha$.
- Publish system public parameters.

$$params = \{e, G_1, G_2, g, f_1, f_2, ..., f_U, u, v, h, g^a, e(g,g)^\alpha, H_1, H_2\}$$

$KEYGEN(S, mk) \to (sk_S)$*:* Once receiving a user attribute set and the master key, the KGC generates the user's private key.

- Choose $s \in Z_P^*$ randomly.
- Compute $sk_S = (S, K_1 = g^\alpha g^{as}, K_2 = g^s, \{K_x = f_x^s\}_{\forall x \in S})$.

$ENC(AS, m) \to (CT)$*:* Once receiving an access structure and a plaintext, the data owner generates the corresponding ciphertext.

- Choose $R \in \{0,1\}^k$ randomly and computes $r = H_2(m||R)$.

- Random select a vector $\vec{\mu} = (r, y_2, ..., y_n) \in Z_p^n$, where $y_2, ..., y_n \in Z_p$. For each row $\mathbb{A}_j$ of $\mathbb{A}$, computes $\lambda_j = \vec{\mu} \cdot \mathbb{A}_j, j \in [1, l]$.
- Randomly chooses $r_j \in Z_p$ for each $j \in [1, l]$. Computes

$$\begin{aligned} C = (m||R) \oplus H_1(e(g,g)^{\alpha r}), C_1 = g^r, C_2 = h^r, \\ \{C_{3,j} = g^{a\lambda_j} f_{\rho(j)}^{-r_j}, C_{4,j} = g^{r_j}\}_{\forall j \in [1,l]}, \\ \bar{C} = u^{H_2(m)} v^{H_2(R)} \end{aligned} \tag{4}$$

- Output $CT = \{(\mathbb{A}, \rho), C, C_1, C_2, \{C_{3,j}, C_{4,j}\}_{j \in [1,l]}, \bar{C}\}$

$DEC(sk_S, CT) \rightarrow (m/\perp)$: Once receiving the user's private key and ciphertext, the recipient computes the plaintext or it outputs reject otherwise.

- If $R(S, AS) = 0$, output $\perp$,
- Otherwise, let $J = \{j : \rho(j) \in S\}$, finds elements $\theta_j \in Z_p^*$ such that $\sum_{j \in J} \theta_j \cdot A_j = (1, 0, ..., 0)$.
- Compute

$$\Delta = \frac{e(K_1, C_1)}{\prod_{j \in J} (e(K_2, C_{3,j}) \cdot e(K_{\rho(j)}, C_{4,j}))^{\theta_j}}.$$

$$m||R = C \oplus H_1(\Delta).$$

- Outputs m if $\bar{C} = u^{H_2(m)} v^{H_2(R)}$. Otherwise, outputs $\perp$.

$RKGEN(sk_S, AS') \rightarrow (rk_{AS \rightarrow AS'})$: Once receiving the user's private key and an access policy, the recipient generates a re-encryption key.

- Randomly chooses $\chi \in G_2, \delta \in Z_p^*$.
- Computes

$$\begin{aligned} rk_0 = e(g,g)^{\alpha H_2(\chi)}, rk_1 = K_1^{H_2(\chi)} \cdot h^{\delta}, \\ rk_2 = g^{\delta}, rk_3 = K_2^{H_2(\chi)}, \\ \{rk_{4,j} = K_x^{H_2(\chi)}\}_{\forall x \in S}, \\ rk_5 = ENC(AS', \chi). \end{aligned} \tag{5}$$

- Output the re-encryption key as $rk_{AS \rightarrow AS'} = (rk_0, rk_1, rk_2, rk_3, \{rk_{4,x}\}_{x \in S}, rk_5)$.

$REENC(rk_{AS \rightarrow AS'}, CT) \rightarrow (CT')$: Once receiving an original ciphertext and a re-encryption key, the proxy generates the re-encrypted ciphertext.

- If $R(S, AS) = 0$, output $\perp$,
- Otherwise, let $J = \{j : \rho(j) \in S\}$, finds elements $\theta_j \in Z_p^*$ such that $\sum_{j \in J} \theta_j \cdot A_j = (1, 0, ..., 0)$.

– Compute

$$C_1' = \frac{e(rk_1, C_1) \cdot e(rk_2, C_2)^{-1}}{\prod_{j \in J} (e(rk_3, C_{3,j}) \cdot e(rk_{4,\rho(j)}, C_{4,j}))^{\theta_j}}.$$

– Sets $C' = C, \bar{C}' = \bar{C}, C_2' = rk_5, C_3' = rk_0$.
– Output the re-encrypted ciphertext $CT' = (C', \bar{C}', C_1', C_2', C_3')$.

$DEC_{re}(sk_{S'}, CT') \rightarrow (m)$*:* Once receiving a private key and a re-encrypted ciphertext, the shared user computes the plaintext or it outputs reject otherwise.

– First decrypt χ from C_2' by running $DEC(sk_{S'}, C_2')$.
– Then computes

$$m||R = C' \oplus H_1(C_1'^{1/H_2(\chi)}).$$

$CLAIM(CT, rk_{AS \rightarrow AS'}, CT') \rightarrow (True/False)$*:* Once receiving an original ciphertext, re-encryption key, re-encrypted ciphertext, and proof published by a shared user, the verifier announces whether it is correct or not.

– Suppose the re-encrypted ciphertext is claimed to be wrong, where the proof is $\pi = (m, R)$.
– Check whether

$$\bar{C}' = \bar{C} = u^{H_2(m)} v^{H_2(R)}$$

$$C_1' = C_3'^r$$

– If the above equations hold, outputs $false$. Otherwise, outputs $true$.

5.2 Correctness of the Proposal Scheme

In this section, we prove the correctness of algorithms DEC and DEC_{re}, which support our proposed scheme in basis.

on input sk and CT, when $R(S, AS) = 1$, we have

$$\begin{aligned} \Delta &= \frac{e(K_1, C_1)}{\prod_{j \in J} (e(K_2, C_{3,j}) \cdot (K_{\rho(j)}, C_{4,j}))^{\theta_j}} \\ &= \frac{e(g^\alpha g^{as}, g^r)}{\prod_{j \in J} (e(g^s, g^{a\lambda_j} f_{\rho(j)}^{-r_j}) \cdot (f_{\rho(j)}^s, g^{r_j}))^{\theta_j}} \\ &= \frac{e(g,g)^{\alpha r} \cdot e(g^{as}, g^r)}{e(g^s, g^a)^{\sum_{j \in J} \lambda_j \theta_j}} \\ &= e(g,g)^{\alpha r} \end{aligned} \tag{6}$$

$$\begin{aligned} C \oplus H_1(\Delta) &= [(m||R) \oplus H_1(e(g,g)^{\alpha r})] \oplus H_1(e(g,g)^{\alpha r}) \\ &= m||R \end{aligned} \tag{7}$$

On input sk and CT', we have

$$\begin{aligned} C_1' &= \frac{e(rk_1, C_1) \cdot e(rk_2, C_2)^{-1}}{\prod_{j \in J} (e(rk_3, C_{3,j}) \cdot e(rk_{4,\rho(j)}, C_{4,j}))^{\theta_j}} \\ &= \frac{e(K_1^{H_2(\chi)} \cdot h^{\delta}, C_1) \cdot e(g^{\delta}, h^r)}{\prod_{j \in J} (e(K_2^{H_2(\chi)}, C_{3,j}) \cdot (K_{\rho(j)}^{H_2(\chi)}, C_{4,j}))^{\theta_j}} \\ &= \frac{e(K_1^{H_2(\chi)}, C_1)}{\prod_{j \in J} (e(K_2^{H_2(\chi)}, C_{3,j}) \cdot (K_{\rho(j)}^{H_2(\chi)}, C_{4,j}))^{\theta_j}} \\ &= e(g,g)^{\alpha r H_2(\chi)} \end{aligned} \tag{8}$$

$$\begin{aligned} C' \oplus H_1(C_1'^{1/H_2(\chi)}) &= [(m||R) \oplus H_1(e(g,g)^{\alpha r})] \oplus H_1(e(g,g)^{\alpha r H_2(\chi)/H_2(\chi)}) \\ &= m||R \end{aligned} \tag{9}$$

6 Security Analysis

In this section, we analyze the security of the proposed scheme. Specifically, following the security requirements discussed earlier, our analysis focuses on Selective-Structure Chosen Plaintext Security, Key Security, and Verifiability for the proposed scheme.

6.1 Selective-Structure Chosen Plaintext Security

Theorem 1. *Our scheme satisfies semantically secure assuming that the q-parallel BDHE assumption holds.*

Proof. During the data uploading phase, the confidentiality of the data owner is achieved based on our encryption algorithm, which is modified from a ciphertext-policy attribute-based encryption scheme. Assume there exists a PPT adversary $\mathcal{A}$ who can break the Selective-Structure Chosen Plaintext Security in our scheme with a non-negligible probability ϵ, then a simulator $\mathcal{B}$ can be constructed that can solve the q-parallel BDHE assumption with a non-negligible advantage. Recall that $\mathcal{B}$ is given a q-parallel BDHE instance $(\vec{v}, T)$, and its goal is to decide whether T equals to $e(g,g)^{r \cdot \alpha^{q+1}}$ or T is randomly chosen from G_2.

In the data-sharing phase, the original data recipient generates a re-encryption key to send to the proxy server. In this case, the proxy key consists of two parts: a valid key that has been blinded, and a blinding factor that is encrypted under the shared user access policy. The blinding factor has the same privacy protection as plaintext messages, and its security is provided by the CP-ABE scheme. The proxy decrypts the original ciphertext using the blinded key and computes the decryption factor in the blinded state. Nevertheless, the proxy still cannot get any information about the plaintext.

6.2 Key Security

Collusion resistance refers to the property of a system or protocol that prevents participants from collaborating in a way that undermines the intended security or fairness guarantees. It ensures that even if the proxy and shard user conspires together, they cannot manipulate the system to gain unfair advantages or obtains the original data recipient's private key. In our proxy re-encryption scheme, the re-encryption key is masked by h^{δ}. If the proxy colludes with the shared user, they can compute the blinding factor χ to un-blind the user's private key. However, thanks to the protection of h^{δ}, the adversary still cannot obtain the original valid key. Therefore, our scheme can resist collusion attacks and achieves key security.

6.3 Verifiability

Theorem 2. *The proposed scheme satisfies the verifiable property assuming the discrete logarithm assumption holds.*

Proof. In the scheme proposed in this paper, the ciphertext and the re-encrypted ciphertext are stored on the Blockchain. Using the tamper-proof nature of the Blockchain, the data can be guaranteed not to be tampered with. Meanwhile, the proxy re-encryption scheme used in this paper adds a digital commitment to the message content in the encryption algorithm as well as an authentication mechanism, which makes the verifier believe that the correct re-encryption algorithm computes the ciphertext and that the proxy cannot pass the authentication by using other ciphertexts. Suppose a PPT adversary $\mathcal{A}$ can violate verifiability of our proposed scheme with a non-negligible advantage ϵ, then a simulator $\mathcal{B}$ can be constructed that can solve the discrete logarithm problem with a non-negligible advantage. Recall that $\mathcal{B}$ is given a discrete logarithm instance $(e, G_1, G_2, p, g, g^{\beta})$, its goal is to output the value β.

7 Performance Analysis

In this section, we present the computation costs and communication overheads of the proposed scheme, and compare it with some related schemes, such as [7] and [6].

7.1 Functionality Comparison

The functionality comparison between our scheme and other related works is shown in Table 2. "✓" indicates that the feature can be implemented while "✗" signifies that the scheme fails to realize this functionality.

Fine-grained access control indicates that a data owner can specify a class of recipients to access the data whose attributes conform to a certain access policy. That is one-to-many access control. And the scheme of [7] belongs to identity-based encryption, which is suitable for one-to-one encryption. All of the

Table 2. Functionality comparison of our scheme and related works.

Functionality	[7]	[6]	Our scheme
Fine-Grained Access Control	✗	✓	✓
Data Sharing	✓	✓	✓
Verifiability	✗	✓	✓
Collusion-Resistance	✓	✓	✓
Non-repudiation	✓	✓	✓

above schemes combine proxy re-encryption techniques to realize data sharing, where the original data recipient generates the re-encryption key and undergoes a re-encryption operation by an incompletely trustworthy third-party proxy server, which realizes the conversion of ciphertexts without exposing the plaintext information. Verifiability prevents lazy proxies from dishonestly performing re-encryption algorithms to save costs while randomly selecting messages from the ciphertext space to ensure the interests of shared users. There is no effective authentication mechanism provided in [7]. The user can only repeat the re-encryption algorithm to compare the ciphertexts. The collusion-resistance property secures the key of the original data recipient and prevents curious proxy and shared users from colluding to compute the complete user's private key. Non-repudiation guarantees users' rights and interests through the tamper-proof nature of Blockchain technology, providing a credible source of information when users have disputes with proxy.

7.2 Theoretical Analysis

We theoretically analyze and compare the computation and storage cost of our works and other related works in Table 4 and Table 3. We mainly focus on the most time-consuming calculations in bilinear groups, including exponentiation operation, multiplication operation, and bilinear pairings. We set $\mathcal{P}$, $\mathcal{E}$, and $\mathcal{M}$ to represent the computation cost of a bilinear pairing, a modular exponentiation, and a modular multiplication operation, respectively. The size of a single group element in G_1 and G_2 are correspondingly denoted as $|G_1|$ and $|G_2|$.

Table 3. Storage costs for messages in each phase of the protocol.

Scheme	private key	ciphertext	re-key	re-encrypted ciphertext
[7]	$\|G_1\|$	$\|G_1\| + \|G_2\|$	$2\|G_1\| + \|G_2\|$	$2\|G_1\| + 2\|G_2\|$
[6]	$(3n+2)\|G_1\|$	$(4n+2)\|G_1\| + \|G_2\|$	$(2n+5)\|G_1\|$	$(n+4)\|G_1\| + \|G_2\|$
Our scheme	$(n+2)\|G_1\|$	$(2n+3)\|G_1\| + \|G_2\|$	$(3n+4)\|G_1\| + 2\|G_2\|$	$(2n+2)\|G_1\| + 3\|G_2\|$

Storage Costs. There are four types of messages in our scheme. During the registration phase, KGC generates a user's secret key to send to the user with

Table 4. Computation costs for algorithms in each phase of the protocol.

Scheme	$KEYGEN$	ENC	DEC	$RKGEN$	$REENC$	DEC_{re}	$CLAIM$
[7]	$\mathcal{E}$	$\mathcal{P}+2\mathcal{E}+\mathcal{M}$	$\mathcal{P}+\mathcal{E}$	$\mathcal{P}+2\mathcal{E}+2\mathcal{M}$	$\mathcal{P}+\mathcal{M}$	$2\mathcal{P}+2\mathcal{M}$	$\mathcal{P}+\mathcal{M}$
[6]	$(4n+2)\mathcal{E}+(2n+1)\mathcal{M}$	$(5n+3)\mathcal{E}+(n+1)\mathcal{M}$	$(2n+1)\mathcal{P}+n\mathcal{E}+2n\mathcal{M}$	$6\mathcal{E}+3\mathcal{M}$	$(3n+1)\mathcal{P}+n\mathcal{E}+(3n+2)\mathcal{M}$	$3\mathcal{P}+3\mathcal{M}$	$n\mathcal{P}+n\mathcal{E}$
Our scheme	$(n+2)\mathcal{E}+\mathcal{M}$	$(3n+5)\mathcal{E}+(2n+1)\mathcal{M}$	$(2n+1)\mathcal{P}+n\mathcal{E}+2n\mathcal{M}$	$(4n+8)\mathcal{E}+(n+1)\mathcal{M}$	$(2n+2)\mathcal{P}+n\mathcal{E}+(2n+1)\mathcal{M}$	$(2n+1)\mathcal{P}+(n+1)\mathcal{E}+2n\mathcal{M}$	$3\mathcal{E}+\mathcal{M}$

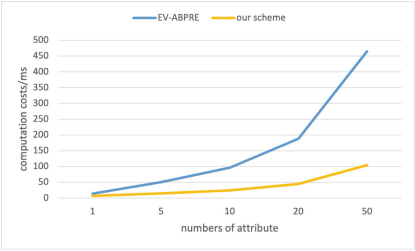
a key size of $(n+2)|G_1|$. The data owner protects the data using an encryption algorithm that generates a ciphertext of size $(2n+3)|G_1|+|G_2|$. When data is shared, the original data recipient first generates a re-encryption key with size $(3n+4)|G_1|+2|G_2|$. Then the agent performs the re-encryption operation to generate the re-encryption ciphertext of size $(2n+2)|G_1|+3|G_2|$. [6] and our scheme are attribute-based encryption schemes where the user's secret key and ciphertext are associated with attributes as well as access policies, respectively. The message length increases linearly with the number of attributes, enabling fine-grained one-to-many access control. While [7] provides identity-based encryption, the complexity of its storage overhead will be the same as ours when the identity is converted to user attributes. Since [7] belongs to identity-based encryption and [6] re-encrypts attribute-based ciphertext agents into identity-based cryptosystems, their re-encryption key has a slight advantage over our scheme, which approximates the size of the re-encryption key in our scheme to be twice as large as theirs. However, they add extra parameters to the user's private key and ciphertext in order to provide verifiability, resulting in all other message lengths being larger than ours. Therefore, the scheme proposed in this paper has a low storage overhead when implementing fine-grained access control and data sharing.

Computation Costs. In the proposed scheme, KGC performs the $KEYGEN$ algorithm to compute the user's private key, which costs $(n+2)Te+Tm$. During the data uploading phase, the data owner encrypts its data by executing the ENC algorithm, which costs $(3n+5)Te+(2n+1)Tm$. Original data recipient access data by executing the DEC algorithm, which costs $(2n+1)Tp+nTe+2nTm$. DR generates a re-encryption key by executing the $RKGEN$ algorithm, which costs $(4n+8)Te+(n+1)Tm$. And proxy executing the $REENC$ algorithm to compute a re-encrypted ciphertext, which costs $(2n+2)Tp+nTe+(2n+1)Tm$. Then, SU requests and decrypts ciphertext by executing the DEC_{re} algorithm, which costs $(2n+1)Tp+(n+1)Te+2nTm$. The $CLAIM$ algorithm is run when a shared user verifies a re-encrypted ciphertext, which costs $3Te+Tm$. Comparing the attribute-based scheme [6] and our scheme, the encryption and decryption algorithms have the same computational complexity, but in the verification phase of re-encrypted ciphertexts, our proposed scheme is efficient.

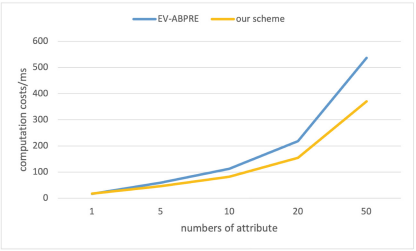
7.3 Experimental Analysis

In order to evaluate the execution efficiency of the scheme proposed in this paper, we design experiments to test the time overhead required for each phase

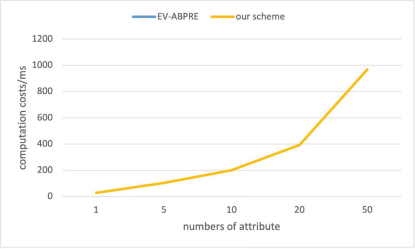
of the scheme. We conducted the experiment in Java running on the PC with one 2.30 GHz Intel Core i5, 16.0-GB memory, and Windows 11 system. We use an elliptic curve of type A and set the security parameter to 160bit.



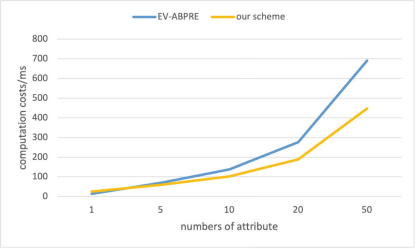
(a) Computation costs of KEYGEN.



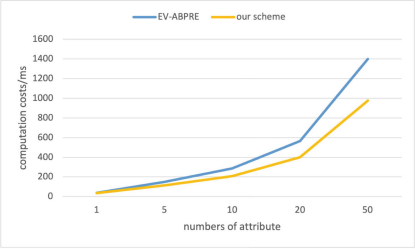
(b) Computation costs of ENC.



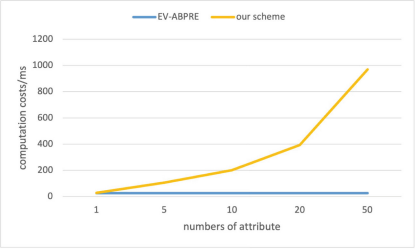
(c) Computation costs of DEC.



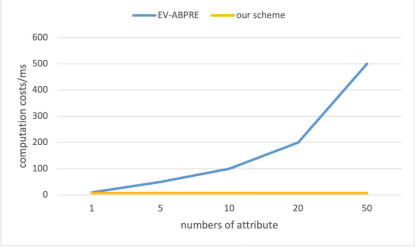
(d) Computation costs of RKGEN.



(e) Computation costs of REENC.



(f) Computation costs of DECre.



(g) Computation costs of CLAIM.

Fig. 2. Computation costs of our schemes.

Figure 2 compares the execution time of our proposed scheme against a related scheme [6].

Figure (2a, 2b, 2d, 2e) shows that our scheme is more efficient than the EV-ABPRE scheme in the key generation, encryption, re-encryption key generation, and re-encryption phases. This is because their scheme needs to compute additional parameters to achieve verifiability. It is shown in Fig. 2c that both schemes have the same decryption efficiency, which is equivalent to performing a traditional CP-ABE decryption. Figure 2f shows that our scheme has a larger computational overhead in performing the decryption of the re-encrypted ciphertext because in [6], the re-encrypted ciphertext is converted to a single identity-based encryption, whose decryption efficiency is independent of the number of attributes. Figure 2g shows that our scheme has a significant advantage in the ciphertext verification phase, and the computational overhead does not grow linearly with the number of attributes increases.

8 Conclusion

This paper applies Blockchain and attribute-based proxy re-encryption to propose a privacy-preserving security model in connected vehicle data sharing. ABPRE allows third-party cloud servers to transform ciphertexts without obtaining plaintext information to achieve user access control to data. At the same time, to solve the problem that centralized cloud servers may have a single point of failure and dishonest cloud servers return wrong ciphertexts to save costs, this paper combines Blockchain technology to provide tamper-proof data storage and a verifiable mechanism of re-encrypted ciphertexts to protect users' interests from being infringed. Security and performance analyses demonstrate that the proposed scheme achieves a good balance between security and efficiency.

References

1. Ateniese, G., Fu, K., Green, M., Hohenberger, S.: Improved proxy re-encryption schemes with applications to secure distributed storage. ACM Trans. Inform. Syst. Secur. (TISSEC) **9**(1), 1–30 (2006)
2. Baker, T., Asim, M., Samwini, H., Shamim, N., Alani, M.M., Buyya, R.: A blockchain-based fog-oriented lightweight framework for smart public vehicular transportation systems. Comput. Netw. **203**, 108676 (2022). https://doi.org/10.1016/j.comnet.2021.108676
3. Bao, Y., Qiu, W., Cheng, X., Sun, J.: Fine-grained data sharing with enhanced privacy protection and dynamic users group service for the IoV. IEEE Trans. Intell. Transp. Syst. 1–15 (2022). https://doi.org/10.1109/TITS.2022.3187980
4. Blaze, Matt, Bleumer, Gerrit, Strauss, Martin: Divertible protocols and atomic proxy cryptography. In: Nyberg, Kaisa (ed.) EUROCRYPT 1998. LNCS, vol. 1403, pp. 127–144. Springer, Heidelberg (1998). https://doi.org/10.1007/BFb0054122
5. Elagin, V., Spirkina, A., Buinevich, M., Vladyko, A.: Technological aspects of blockchain application for vehicle-to-network. Information **11**(10), 465 (2020)

6. Feng, T., Wang, D., Gong, R.: A blockchain-based efficient and verifiable attribute-based proxy re-encryption cloud sharing scheme. Information **14**(5) (2023). https://doi.org/10.3390/info14050281
7. Gao, Y., Chen, Y., Hu, X., Lin, H., Liu, Y., Nie, L.: Blockchain based IIoT data sharing framework for SDN-enabled pervasive edge computing. IEEE Trans. Industr. Inf. **17**(7), 5041–5049 (2021). https://doi.org/10.1109/TII.2020.3012508
8. Ge, C., Liu, Z., Xia, J., Fang, L.: Revocable identity-based broadcast proxy re-encryption for data sharing in clouds. IEEE Trans. Dependable Secure Comput. **18**(3), 1214–1226 (2021). https://doi.org/10.1109/TDSC.2019.2899300
9. Ge, C., Susilo, W., Baek, J., Liu, Z., Xia, J., Fang, L.: A verifiable and fair attribute-based proxy re-encryption scheme for data sharing in clouds. IEEE Trans. Dependable Secure Comput. **19**(5), 2907–2919 (2021)
10. Green, Matthew, Ateniese, Giuseppe: Identity-based proxy re-encryption. In: Katz, Jonathan, Yung, Moti (eds.) ACNS 2007. LNCS, vol. 4521, pp. 288–306. Springer, Heidelberg (2007). https://doi.org/10.1007/978-3-540-72738-5_19
11. Hanaoka, G., et al.: Generic construction of chosen ciphertext secure proxy re-encryption. In: Dunkelman, Orr (ed.) CT-RSA 2012. LNCS, vol. 7178, pp. 349–364. Springer, Heidelberg (2012). https://doi.org/10.1007/978-3-642-27954-6_22
12. Karthikeyan, H., Usha, G.: Real-time DDoS flooding attack detection in intelligent transportation systems. Comput. Electr. Eng. **101**, 107995 (2022). https://doi.org/10.1016/j.compeleceng.2022.107995
13. Kenney, J.B.: Dedicated short-range communications (DSRC) standards in the United States. Proc. IEEE **99**(7), 1162–1182 (2011)
14. Liang, X., Cao, Z., Lin, H., Shao, J.: Attribute based proxy re-encryption with delegating capabilities. In: Proceedings of the 4th International Symposium on Information, Computer, and Communications security, pp. 276–286 (2009)
15. Liu, J., Zhang, L., Li, C., Bai, J., Lv, H., Lv, Z.: Blockchain-based secure communication of intelligent transportation digital twins system. IEEE Trans. Intell. Transp. Syst. **23**(11), 22630–22640 (2022). https://doi.org/10.1109/TITS.2022.3183379
16. Nakamoto, S.: Bitcoin: a peer-to-peer electronic cash system. Decentralized business review (2008)
17. Ohata, S., Kawai, Y., Matsuda, T., Hanaoka, G., Matsuura, K.: Re-encryption verifiability: how to detect malicious activities of a proxy in proxy re-encryption. In: Nyberg, K. (ed.) Topics in Cryptology – CT-RSA 2015, pp. 410–428. Springer, Cham (2015)
18. Panigrahy, S.K., Emany, H.: A survey and tutorial on network optimization for intelligent transport system using the internet of vehicles. Sensors **23**(1), 555 (2023)
19. Shao, J.: Anonymous id-based proxy re-encryption, pp. 364–375 (2012). https://doi.org/10.1007/978-3-642-31448-3_27
20. Xu, L., Wu, X., Zhang, X.: CL-PRE: a certificateless proxy re-encryption scheme for secure data sharing with public cloud. In: Proceedings of the 7th ACM Symposium on Information, Computer and Communications Security. pp. 87–88. ASIACCS 2012. Association for Computing Machinery, New York (2012). https://doi.org/10.1145/2414456.2414507
21. Zavvos, E., Gerding, E.H., Yazdanpanah, V., Maple, C., Stein, S., Schraefel, M.: Privacy and trust in the internet of vehicles. IEEE Trans. Intell. Transp. Syst. **23**(8), 10126–10141 (2022). https://doi.org/10.1109/TITS.2021.3121125