BLAC: A Blockchain-based Lightweight Access Control Scheme in Vehicular Social Networks

Yuting Zuo[1,2][0000−0002−2783−8351], Li Xu[1,2][0000−0002−8972−3373], Yuexin Zhang[1,2], Chenbin Zhao[3], and Zhaozhe Kang[1,2]

[1] Fujian Normal University, Fuzhou, China
[2] Fujian provincial Key Laboratory of Network Security and Cryptology
{qbx20210079,qbx20210078}@yjs.fjnu.edu.cn, {xuli,yxzhang}@fjnu.edu.cn
[3] Wuhan University, Wuhan, China
chenbinzhao96@163.com

Abstract. Vehicular Social Networks (VSNs) rely on data shared by users to provide convenient services. Data is outsourced to the cloud server and the distributed roadside unit in VSNs. However, roadside unit has limited resources, so that data sharing process is inefficient and is vulnerable to security threats, such as illegal access, tampering attack and collusion attack. In this article, to overcome the shortcomings of security, we define a chain tolerance semi-trusted model to describe the credibility of distributed group based on the anti tampering feature of blockchain. We further propose a Blockchain-based Lightweight Access Control scheme in VSNs that resist tampering and collusion attacks, called BLAC. To overcome the shortcomings of efficiency, we design a ciphertext piece storage algorithm and a recovery one to achieve lightweight storage cost. In the decryption operation, we separate a predecryption algorithm based on outsourcing to achieve lightweight decryption computation cost on the user side. Finally, we present the formal security analyses and the simulation experiments for BLAC, and compare the results of experiments with existing relevant schemes. The security analyses show that our scheme is secure, and the results of experiments show that our scheme is lightweight and practical.

Keywords: Vehicular social networks · Blockchain · Access control · Lightweight.

1 Introduction

Vehicular Social Networks (VSNs) are the integration of social networks and Vehicular Ad hoc Networks [31]. With the rapid development of Internet, Artificial Intelligence and other technologies [17], VSNs offer many diverse services, e.g., the selection of suitable carpools, intelligent suggestions on travel routes, alerts on traffic conditions, etc.

The above services rely on widely deployed infrastructures. In VSNs, Road-Side Unit (RSU) provides instant communication, real-time road sharing and

temporary data storage [19]. Based on these infrastructures, users form a virtual community and share data [12].

However, illegal access is a serious threat to data sharing [20], so that secure access control is considerable necessary for outsourced data [24]. Therefore, before uploading the ciphertext data, the data owner can independently set access permissions. For instance, a data owner defines that other users need to simultaneously satisify the att_1, att_2 and att_3 to access the data. Then, the data owner sets the access policy as $att_1 \wedge att_2 \wedge att_3$.

Ciphertext-Policy Attribute-Based Encryption (CP-ABE) [3] effectively implements the above requirements. In CP-ABE, the access policy is embeded into ciphertext. The decryption key is related to the attribute set of user. The user can decrypt the ciphertext through the attribute key if and only if attribute set of user satisfies the access policy.

1.1 Motivations

Collusion Resistance is essential to CP-ABE [3]. That is, users cannot combine attribute keys to decrypt ciphertext. Particularly, it is necessary to consider fault-tolerant consensus and tamper-proof for secure access control in distributed VSNs. Blockchain technology promotes the reliability and credibility of distributed RSU [14]. Blockchain facilitates the development of a secure, trusted and distributed intelligent transport ecosystem. Moreover, blockchain resists attacks initiated by a small number of malicious nodes. The consortium blockchain [6] achieves a trade-off between security and performance, which is more suitable for data sharing in VSNs.

In addition, it is necessary to consider the cost of RSU and users. Cloud Server (CS) provides strong computation and storage capabilities. We utilize CS to assist RSU in storing data, and outsource pre-decryption operation of the user into the server in order to relieve the computation overhead of the user.

The above motivation inspires us to consider the following design goals in VSNs: (1) The proposed scheme should achieve the basic secure access control. (2) The proposed scheme enables to tolerate the malicious RSU for the distributed storage and secure consensus. (3) The proposed scheme should balance the security and efficiency, and achieves to minimize the storage cost for the RSU and reduce the computation and storage costs for the user.

1.2 Contributions

We summarize the contributions of our paper as follows:

1. We define a Chain Tolerance Semi-Trusted model, called CTST, which estimates the credibility of group based on the anti tampering feature of blockchain. In addition, we propose a Blockchain-based Lightweight Access Control (BLAC) scheme based on CTST, and this scheme can resist the tampering attack and collusion attack.

2. To promote the efficiency, we design a novel ciphertext piece storage and recovery algorithms for achieving lightweight storage cost of RSU. Furthermore, we separate a pre-decryption algorithm and outsource it to the server to relieve the computation overhead of the user.
3. We present a formal security analysis in terms of confidentiality, collusion attack, and tampering attack. Moreover, we also conduct comprehensive simulation experiments and provide the comparisons with existing works. The experimental results show that our scheme is lightweight and practical.

Organizaiton: The rest of this article is organized as follows. In Section **??**, we review the related work. In Section 3, we introduce some preliminaries for the following construction. We describe system model and design goals in Section 4. The construction of BLAC scheme is proposed in Section 5. We describe the security and performance analysis of our scheme in Section 6 and 7. Finally, the conclusion is given in Section 8.

2 Related Work

To achieve the more fine-grained access control, the CP-ABE [3] seems to be a pretty good cryptographic primitive. It is well known that CP-ABE cryptography primitives have high computational overhead. Some schemes with the outsourced decryption operation [5, 16, 30] were proposed to promote the efficiency. The references [9, 28, 29] further proposed schemes equipped with Linear Secret-Sharing Scheme (LSSS) [2] to make the CP-ABE scheme more expressive.

With the rapid rise of blockchain technology, access control faces new opportunities and challenges. Blockchain provides secure data management services for VSNs [1], such as computation offloading [25], secure storage [14]. Wang *et al.* [21] employed the incentive mechanism in the blockchain to construct a credit-based reputation model. Kang *et al.* [11] designed a distributed vehicular blockchain to achieve secure and efficient data sharing. Wang *et al.* [22] proposed a secure private data sharing scheme and used smart contracts to realise access control and usage track of data. However, attackers can attack the database directly so that system-level access control is ineffective against such attacks.

The above shortcomings inspire extensive prospects for the introduction of blockchain technology in data access control field in VSNs. Liang *et al.* [13] used CP-ABE to achieve flexible access control on blockchain. However, they did not give the concrete structure of CP-ABE scheme. Pu *et al.* [18] proposed data secure sharing scheme to support data recovery in edge servers. The scheme combines blockchain with the traditional CP-ABE based on the access tree structure [3]. Yao *et al.* [27] improved the He *et al.*'s scheme [10] and proposed a lightweight data sharing scheme based on the LSSS. However, they did not consider reducing the computation cost of vehicles. Yang *et al.* [26] proposed a secure data access control scheme with accountability. They used "on-chain/off-chain" structure to reduce the cost of on-chain calculation. However, off-chain computing is expensive for vehicles. Fan *et al.* [8] combined CP-ABE and consortium

blockchain to manage user attributes. Decryption outsourcing reduces the computation cost of the requester. However, Fan *et al.*'s proposal did not take into account optimizing storage cost. In summary, We give the comparisons with the existing related works as shown in Table 1.

Table 1. Comparison of functions between different schemes

Scheme	Access Structure	CTST	Reduce Storage	Computation Outsourcing	Lightweight	Resist Collusion	Resist Tamper
[16]	Tree	✗	✗	✓	✗	✓	✗
[30]	Tree	✗	✗	✓	✗	✓	✗
[18]	Tree	✗	✓	✗	✗	✓	✓
[9]	LSSS	✗	✗	✗	✗	✓	✗
[28]	LSSS	✗	✗	✗	✗	✓	✗
[27]	LSSS	✗	✗	✗	✓	✗	✗
[26]	LSSS	✗	✗	✗	✗	✓	✓
[8]	LSSS	✗	✗	✓	✗	✓	✓
Ours	**LSSS**	✔	✔	✔	✔	✔	✔

3 Preliminaries

3.1 Bilinear Map

Let $\mathbb{G}_1$ and $\mathbb{G}_2$ be two multiplicative cyclic groups of prime order p. $e : \mathbb{G}_1 \times \mathbb{G}_1 \rightarrow \mathbb{G}_2$ is a bilinear map with the following properties [23]:

1. **Bilinearity:** for all $u, v \in \mathbb{G}_1$ and $a, b \in \mathbb{Z}_p$, we have $e(u^a, v^b) = e(u^b, v^a)$.
2. **Non-degeneracy:** $e(g, g) \neq 1$.
3. **Computability:** it is easy to compute $e(u, v)$ for any $u, v \in \mathbb{G}_1$.

3.2 Linear Secret-Sharing Scheme

A linear secret-sharing scheme [2] includes two stages: **Share** and **Reconstruct**. The specific algorithm details are described as follows:

1. **Share** : The shares for each party form a vector over $\mathbb{Z}_p$. There exists a matrix M with l rows and n columns called the share-generating matrix for Π. For all $i = 1, ..., l$, the i-th row of M we let the function ρ defined the party labeling row i as $\rho(i)$. When we consider the column vector $\boldsymbol{v} = (s, y_2, ..., y_n)$, where $s \in \mathbb{Z}_p$ is the secret to be shared, and $y_2, ..., y_n \in \mathbb{Z}_p$ are randomly chosen, then $M\boldsymbol{v}$ is the vector of l shares of the secret s according to Π. The share $(M\boldsymbol{v})_i$ belongs to $\rho(i)$.
2. **Reconstruct** : Let $\mathcal{S} \in \mathbb{A}$ be any authorized set, and let $I \subset \{1, 2, ..., l\}$ be defined as $I = \{i : \rho(i) \in \mathcal{S}\}$. Then, there exist constants $\{\omega_i \in \mathbb{Z}_p\}_{i \in I}$ such that, if $\{\lambda_i = (M\boldsymbol{v})_i\}$ are valid shares of any secret s according to Π. We have $\sum_{i \in I} \omega_i \lambda_i = s$.

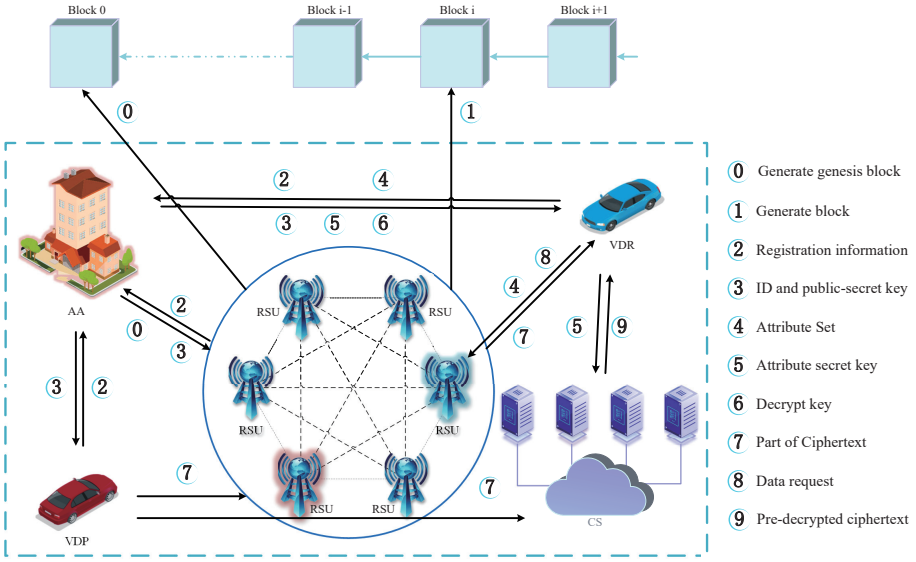


Fig. 1. System Model of BLAC Scheme

3.3 Practical Byzantine Fault Tolerance

We review the Practical Byzantine Fault Tolerance (PBFT) [4] incluidng the following five stages:

1. **Request:** VU sends a message to the Nearest RSU (NRSU). Then the NRSU broadcasts the messages to other RSU in the entire network.
2. **Pre-Prepare:** Sort node in RSU collects and verifies the messages. After the verification passed, it sorts and packs messages into a list. Finally, the list is broadcasted to other RSU.
3. **Prepare:** RSU generates a hash value for the received message list, which is verified, and then broadcasts the hash value to other RSU.
4. **Commit:** If a RSU receives $n - \lfloor \frac{n}{3} \rfloor$ values equal to its own value from other RSU, the RSU broadcasts commitment to other RSU.
5. **Reply:** RSU receives commitment more than $n - \lfloor \frac{n}{3} \rfloor$, it packs list into the latest block lB, and records it on the local blockchain.

4 System Model And Design Goals

In this section, we first give the system model of BLAC scheme, introduce the capabilities of each entity in the model. Then we give the security goals and propose the new model, called Chain Tolerance Semi-Trusted Model.

4.1 System Model

Fig. 1 shows the system model of our BLAC scheme. There are five types of entities: Attribute Authority (AA), Vehicle Data Publisher (VDP), Vehicle Data

Requester (VDR), RoadSide Unit (RSU) and Cloud Server (CS). In particular, VDP and VDR are collectively referred to Vehicle User (VU).

1. **AA:** It generates the public parameters and secert keys for the entities, and provides registration services. It generates the attribute secret key for VDR. In particular, it generates the genesis block and broadcasts it to every RSU.
2. **VDP:** It defines the concrete access policy, which is used to encrypt the message, and then outsources the ciphertexts to every RSU and the CS.
3. **VDR:** It requests encrypted data from the system. It owns the attribute set and the corresponding attribute secret key. It successfully decrypts the ciphertext when a subset of its attributes satisfies the access policy.
4. **RSU:** They maintain the blockchain, store ciphertext pieces and provide services for VUs.
5. **CS:** It stores ciphertext related to the access structure and assists VDR in pre-decrypting the ciphertext.

4.2 Security and performance Goals

We define that AA and VDP are fully trusted. VDR is generally a semi-trusted entity, which is honest but curious about some privacy informations. Malicious VDR means that it colludes with other VDR. The RSU group is CTST and easily hijacked by external attackers. Thus, some RSU may not comply with the contracts and publish false information. In addition, malicious RSU will conspire to restore ciphertext. CS is a semi-trusted entity. In particular, we consider the computation and storage cost of the participants. We further describe the security and performance goals as following:

1. **Confidentiality:** It means that the VDR can successfully decrypt the ciphertext if the attribute set satisfies the access policy.
2. **Resist Tamper:** It means that the scheme should resist malicious RSU tampering with stored ciphertext.
3. **Resist Collusion:** It means that VDR does not decrypt the ciphertext by colluding with others, and malicious RSU does not recover the ciphertext by colluding with other RSU in the group.
4. **Lightweight:** It means that the AA and UV consume the low computation cost, and VU and RSU request the low storage cost.

4.3 Chain Tolerance Semi-Trusted Model

Existing trust models focus on the reliability of a single entity and are not suitable for distributed scenarios. Thus, we define Chain Tolerance Semi-Trusted (CTST) in Definition 1, a new universal model to estimate the credibility of group.

Definition 1. *Let* $\mathbb{P} = \{P_1, P_2, ..., P_n\}$ *be a group of participants with the same role. There are three trusted status for the participants in* $\mathbb{P}$*: fully trusted, semi-trusted and untrusted (malicious).* $\mathbb{D} = \{D_1, D_2, ..., D_t\} \subseteq \mathbb{P}$ *is the discriminant*

Fig. 2. Node Credibility Comparison in blockchain

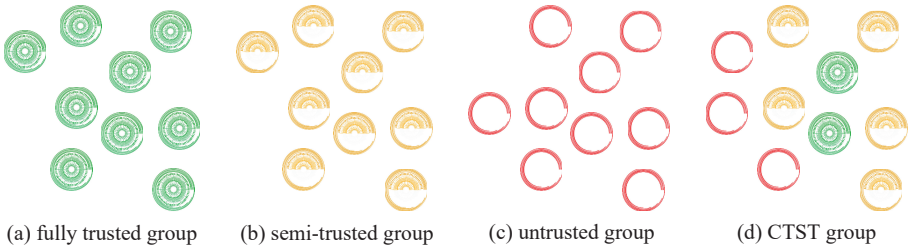


Fig. 3. Group Credibility Comparison in VSNs

subset of $\mathbb{P}$ *with* t *participants, where* $t = \lceil \frac{n}{3} \rceil$. $\mathbb{P}$ *is Chain Tolerance Semi-Trusted (CTST) if and only if* $\forall\ \mathbb{D} \subseteq \mathbb{P}$, $\exists D_i \in \mathbb{D}$, D_i *is fully trusted or semi-trusted participant.*

Fig.2 describes the credibility of nodes in the blockchain. The different nodes represent different levels of credibility, such as a fully trusted node, a semi-trusted node, and untrusted node.

Fig.3 provides four groups of participants with different levels of credibility. A group of participants $\mathbb{P}$ is fully trusted (Fig. 3(a)), if and only if all participants $\{P_1, P_2, ..., P_n\}$ in this group are fully trusted. A CTST group (Fig.3(d)) allows less than $n/3$ participants in the group to be untrusted. CTST requires schemes to tolerate more malicious participants than semi-trusted model. CTST characterizes distributed RSU in VSNs more accurately.

5 The Proposed Scheme

We present the correct construction of our blockchain-based lightweight access control scheme in this section, called BLAC. The details of relevant contracts are shown in Appendix A.

5.1 Construction of BLAC

We define that $\lambda \in N$ is the security parameter. Let $\mathbb{G}_1$ and $\mathbb{G}_2$ be two cyclic groups with the same prime order p, and g is the generator of $\mathbb{G}_1$. $e : \mathbb{G}_1 \times \mathbb{G}_1 \rightarrow \mathbb{G}_2$ is a bilinear map. Let $H : \{0,1\}^* \rightarrow \mathbb{G}_1$ be a hash function cryptographic primitive. Our BLAC scheme includes five stages as follows:

1. **Setup:** AA chooses a random number $\alpha \in \mathbb{Z}_p$ and computes the master secret key $MSK = g^\alpha$. Then AA generates $e(g,g)^\alpha$ as public parameter. AA generates secret key $SK_{rsu} \in \mathbb{Z}_p$, and outputs public key $PK_{rsu} = g^{SKrsu}$ for RSU. We define that the public parameters $PP = \{\mathbb{G}_1, \mathbb{G}_2, e(g,g)^\alpha, g, H\}$.

2. **UserReg:** AA generates the unique identity vu and public-secret key pair $\{PK_{vu} = g^{SK_{vu}}, SK_{vu}\}$ for VU. Then, AA transmits SK_{vu} to VU through a secure channel and broadcasts $\{H(vu), PK_{vu}\}$ to RSU.
3. **KeyGen:** Then, AA generates attribute secret key ASK_{vdr} and decrypt key DK_{vdr} for every VDR based on the attribute sets Att_{vdr}. The key generation process of VDR is shown in Algorithm 1. Among them, the function $attNum()$ represents the number of attributes in the attribute set.
4. **Encryption:** VDP selects the message $\mathcal{M}$ and sets an access policy (M, ρ) for $\mathcal{M}$, where M is an $l \times n$ matrix. Then, VDP encrypts $\mathcal{M}$ and outputs the ciphertext CT. The specific encryption process is shown in Algorithm 2.
5. **Decryption:** We divide decryption into Pre-Decryptoin (Pre-Dec) and Finally Decryptoin (Fin-Dec) algorithms:
   (1) **Pre-Dec**: CS obtains the authorization set I and corresponding attribute secret keys $\{\{K_{i,1}\}_{i\in I}, K_2\}$ of VDR. It obtains CT and $\{\omega_i\}_{i\in I}$ that satisfies $\sum_{i\in I} \omega_i M_i = (1, 0, ..., 0)$. Then, CS generates pre-decrypted ciphertext $\tilde{C} = \prod_{i\in I} \left(\frac{e(K_{i,1}, C_i^1)}{e(K_i, C_i^2)}\right)^{\omega_i} = e(g, g)^{\alpha s DK_{vdr}}$.
   (2) **Fin-Dec**: VDR obtains the original message by calculating $\mathcal{M} = \frac{C}{\tilde{C}^{DK_{vdr}^{-1}}}$.

Correctness: Our scheme ensures the correctness of the calculation of $\tilde{C}$. Since $i \in I$, the Att_i in the key $K_{i,1}$ is the same as the $\rho(i)$ in the ciphertext C_i^2. Therefore, we use H_i to represent $H(Att_i)$ and $H(\rho(i))$ in Equation (1).

$$
\begin{aligned}
\tilde{C} &= \prod_{i\in I} \left(\frac{e(K_{i,1}, C_i^1)}{e(K_2, C_i^2)}\right)^{\omega_i} = \prod_{i\in I} \left(\frac{e(g^{\alpha DK_{vdr}} \cdot H_i^{r_{vdr}}, g^{\lambda_{\rho(i)}})}{e(g^{r_{vdr}}, H_i^{\lambda_{\rho(i)}})}\right)^{\omega_i} \\
&= \prod_{i\in I} \left(\frac{e(g^{\alpha DK_{vdr}}, g^{\lambda_{\rho(i)}}) \cdot e(H_i^{r_{vdr}}, g^{\lambda_{\rho(i)}})}{e(g^{r_{vdr}}, H_i^{\lambda_{\rho(i)}})}\right)^{\omega_i} \\
&= \prod_{i\in I} e(g^{\alpha DK_{vdr}}, g^{\lambda_{\rho(i)}})^{\omega_i} = e(g^{\alpha DK_{vdr}}, g)^{\sum_{i\in I} \lambda_{\rho(i)} \cdot \omega_i} \\
&= e(g, g)^{\alpha s DK_{vdr}}
\end{aligned}
\tag{1}
$$

Algorithm 1 KeyGen

Input: MSK, Att_{vdr}
Output: Attribute secret key ASK_{vdr} and decrypt key DK_{vdr} of VDR

Choose $\{DK_{vdr}, r_{vdr}\} \in \mathbb{Z}_p$ randomly
Calculate $K = MSK^{DK_{vdr}}$
Let $z = attNum(Att_{vdr})$
Calculate $K_2 = g^{r_{vdr}}$
for each $i \in [1, z]$ **do**
    Calculate $K_{i,1} = K \cdot H(Att_i)^{r_{vdr}}$
return $ASK_{vdr} = \{\{K_{i,1}\}_{i\in[1,z]}, K_2\}$ and DK_{vdr} of VDR

Algorithm 2 Encryption

Input: PP, (M, ρ), $\mathcal{M}$
Output: Ciphertext CT

Choose the secret $s \in \mathbb{Z}_p$ randomly
Choose $y_2, y_3, ..., y_n \in \mathbb{Z}_p$ randomly
Let $\boldsymbol{v} = (s, y_2, y_3, ..., y_n) \in \mathbb{Z}_p^n$
for each $i \in [1, l]$ **do**
    $\lambda_{\rho(i)} = M_i \cdot \boldsymbol{v}$
    $C_i^1 = g^{\lambda_{\rho(i)}}$, $C_i^2 = H(\rho(i))^{\lambda_{\rho(i)}}$
Calculate $C = \mathcal{M} e(g, g)^{\alpha s}$
return The ciphertext $CT = \{C, (M, \rho), \{C_i^1, C_i^2\}_{i\in[1,l]}\}$

5.2 Ciphertext Piece Storage and Recovery

We design the ciphertext piece storage and recovery algorithms for secure and efficient storage. Data storage contract 5 and recovery contract 6 is shown in Appendix A.

1. **Ciphertext Piece Storage:** C is divided into n pieces $\{C_1, C_2, ..., C_n\}$. VDR recovers C iff it acquire at least t correct pieces, where $\frac{n}{3} < t < \frac{2n}{3}$. The core algorithm is shown as Algorithm 3.
2. **Ciphertext Recovery:** The Nearest RSU (NRSU) of VDR broadcasts the access requests received from the VDR to other RSU. If Att_{vdr} satisfies the access policy, RSU sends C_i to the VDR. Finally, VDR gets at least t correct pieces $\{C_{k_i}\}_{i\in[1,t],k_i\in[1,n]}$ to recover C. The ciphertext recovery process is shown as Equation (2).

$$\begin{bmatrix} s_1 \\ s_2 \\ ... \\ s_t \end{bmatrix} = \begin{bmatrix} x_{k_1}^{t-1} & x_{k_1}^{t-2} & ... & x_{k_1} & 1 \\ x_{k_2}^{t-1} & x_{k_2}^{t-2} & ... & x_{k_2} & 1 \\ ... & ... & ... & ... & \\ x_{k_t}^{t-1} & x_{k_t}^{t-2} & ... & x_{k_t} & 1 \end{bmatrix}^{-1} \times \begin{bmatrix} y_{k_1} \\ y_{k_2} \\ ... \\ y_{k_t} \end{bmatrix} \tag{2}$$

Algorithm 3 Ciphertext Piece Storage

Input: $C = [C_{Lt}, C_{Rt}]$, n, t
Output: Ciphertext pieces $\{C_i\}_{i\in[1,n]}$
Let $t_r = t/2$, $t_l = t - t_r$
Let $len = C_{Lt}.size(), l = 0$
for each $i \in [1, t_l]$ **do**
  $k = \lceil (len - l)/(t_l - i + 1) \rceil$
  $s_i = int(C_{Lt}.substr(l, k))$
  $l = l + k$
Let $len = C_{Rt}.size(), l = 0$
for each $i \in [1, t_r]$ **do**
  $k = \lceil (len - l)/(t_r - i + 1) \rceil$
  $s_{t_l+i} = int(C_{Rt}.substr(l, k))$
  $l = l + k$
for each $i \in [1, n]$ **do**
  Let $y_i = 0$ and choose $x_i \in \mathbb{Z}_p$
  for each $j \in [1, t]$ **do**
    Calculate $y_i = y_i \cdot x_i + s_j$
return $\{C_i = (x_i, y_i)\}_{i\in[1,n]}$

6 Security Analysis

In this section, we analyze the security of our scheme in terms of confidentiality, tampering resistance, and collusion resistance.

6.1 Confidentiality

The confidentiality proof of our scheme relies on a mathematical difficultyassumption similar to that of scheme [23]. In particular, we analyze that our scheme satisfies confidentiality through Theorem 1.

Theorem 1. *A single entity whose attributes do not satisfy the access policy cannot decrypt the data even if the full ciphertext is obtained.*

Proof. VDR cannot get correct $\omega_i \in \mathbb{Z}_p$ if attribute set of VDR does not satisfies the access policy. Suppose VDR generates illegal $\omega_i' \in \mathbb{Z}_p$ and tries to decrypt. Due to lack of necessary attributes (e.g., attribute $j \in I$), VDR cannot calculate Equation (3) correctly in the Equation (1). That is, for the VDR whose attributes do not satisfy the access policy, the Equation (3) is invalid.

$$\frac{e(H(Att_i)^{r_{vdr}}, g^{\lambda_{\rho(i)}})}{e(g^{r_{vdr}}, H(\rho(i))^{\lambda_{\rho(i)}})} = 1 \tag{3}$$

In addition, since $\omega_i' \in \mathbb{Z}_p$ is not related to VDR attributes, the value of $e(g^{\alpha DK_{vdr}}, g)^{\sum_{i\in I} \lambda_{\rho(i)} \cdot \omega_i'}$ in Equation (1) cannot be calculated correctly, namely, $\sum_{i\in I} \lambda_{\rho(i)} \cdot \omega_i' \neq s$.

CS can obtain the ASK_{vdr} and $\omega_i \in \mathbb{Z}_p$ in Decryption stage. $\tilde{C}$ can be calculated by CS. However, CS cannot obtain the DK_{vdr}, it cannot decrypt the message. RSU gets no more valuable information than CS.

In summary, our scheme satisfies confidentiality.

6.2 Tampering Resistance

Theorem 2. *Our BLAC scheme resists tamper even if malicious RSU provides incorrect data.*

Proof. We consider that the RSU group is Chain Tolerance Semi-Trusted, which means that less than one-third of RSU may send error message. We analyze the tampering resistance from the following three stages: In Setup stage, we set the interaction threshold between AA and RSU as *thresh*. The Setup stage will prevent malicious RSU from joining the system. In Encryption stage, RSU verifies the correctness of the data broadcasted by other RSU. RSU stores the data when it gets correct data more than 1/3 of the total. Since the group of RSU is Chain Tolerance Semi-Trusted, it is guaranteed that the algorithm 5 can perform correctly. It will not affect the correctness of the scheme even if some RSU are malicious. In Decryption stage, VDR recovers complete data only if at least t correct data pieces are obtained. Because the group of RSU is CTST, VDR can obtain at least 2/3 of the correct pieces. Since $t < \frac{2n}{3}$, VDR is able to recover C correctly.

6.3 Collusion Resistance

Our scheme considers two types of collusion attacks. Theorem 3 proves that our scheme resists malicious VDR colluding to decrypt the ciphertext, and Theorem 4 proves that our scheme resists malicious RSU colluding to recover C.

Theorem 3. *A CP-ABE scheme is resistant to collusion attack, even if all VDR combine their keys can not decrypt correctly.*

Proof. We consider that the malicious VDR does not satify the access policy as a result of lacking an attribute $j \in I$.

Malicious VDR gets $\{K_o H(Att_j)^{r_o}, g^{r_o}\} \in ASK_o$ of other VDR, where $K_o = MSK^{DK_o}$. Then, it obtains $\{\omega_i\}_{i \in I}$ that satisfies $\sum_{i \in I} \omega_i M_i = (1, 0, ..., 0)$. Then, $\tilde{C}$ is calculated as Equation (4).

$$\begin{aligned}
\tilde{C} &= \left(\frac{e(K_{j,1}, C_j^1)}{e(g^{r_o}, C_j^2)}\right)^{\omega_j} \prod_{i \in I, i \neq j} \left(\frac{e(K_{i,1}, C_i^1)}{e(K_2, C_i^2)}\right)^{\omega_i} \\
&= e(g^{\alpha DK_o}, g^{\lambda_{\rho(j)}})^{\omega_j} \prod_{i \in I, i \neq j} e(g^{\alpha DK_{mvdr}}, g^{\lambda_{\rho(i)}})^{\omega_i} \\
&\neq e(g^{\alpha DK_{mvdr}}, g^{\lambda_{\rho(j)}})^{\omega_j} \prod_{i \in I, i \neq j} e(g^{\alpha DK_{mvdr}}, g^{\lambda_{\rho(i)}})^{\omega_i} \\
&= \prod_{i \in I} e(g^{\alpha DK_{mvdr}}, g^{\lambda_{\rho(i)}})^{\omega_i} \\
&= e(g, g)^{\alpha s DK_{mvdr}}
\end{aligned} \tag{4}$$

Therefore, malicious VDR cannot collude with other VDR when it lacks necessary attributes.

Theorem 4. *Malicious RSU in the RSU group cannot collude to recover C, if this group is CTST.*

Proof. Let n and t denote the total number of RSU and the recovery threshold of C, Malicious RSU can only obtain less than 1/3 pieces $\{(x_i, y_i)\}_{i \in [1,t']}$ of C. It try to solve a system of equations (5) to get the full ciphertext:

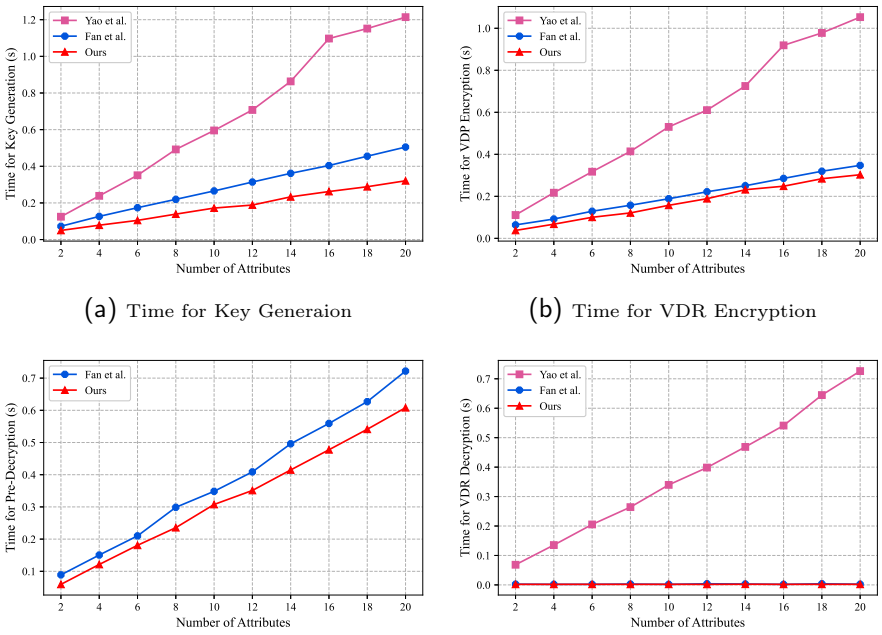
$$\begin{cases}
s_1 x_1^{t-1} + s_2 x_1^{t-2} + ... + s_{t-1} x_1 + s_t = y_1 \\
s_1 x_2^{t-1} + s_2 x_2^{t-2} + ... + s_{t-1} x_2 + s_t = y_2 \\
... \\
s_1 x_{t'}^{t-1} + s_2 x_{t'}^{t-2} + ... + s_{t-1} x_{t'} + s_t = y_{t'}
\end{cases} \tag{5}$$

Recall that in our scheme, ciphertext recovery threshold $t > n/3 > t'$. That is, the system of equations (5) has no unique solution. Thus, even if all malicious RSU collude, C cannot be recovered correctly.

7 Performance Analysis

Yao *et al.* [27] and Fan *et al.* [8] used CP-ABE to realize efficient data access control in VSNs. In this section, we compare our scheme with the above two schemes. To ensure fair comparisons, we simulate three schemes in the same environment with the same access structure and message.

We simulate our blockchain environment by using Hyperledger Fabric [7] and implement our scheme based on the Pairing Based Cryptography (PBC) library [15] in VS2015. We use the symmetric elliptic curve α-curve with 512-bit field size and 160-bit group order. We execute our scheme on an Intel(R) Core (TM) i5, with 8 GB RAM running Windows 10 64-bit system.



(a) Time for Key Generaion (b) Time for VDR Encryption

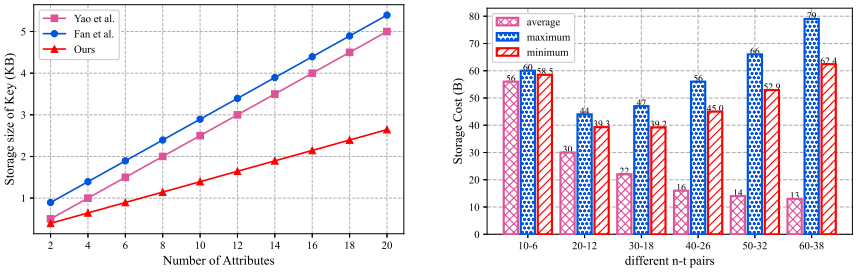
(c) Time for Pre-Dec (d) Time for VDR Decryption

Fig. 4. Computational Cost of Different Stages

7.1 Computational Cost

As depicted in Fig. 4, We compare the key generation time, encryption time, and decryption time of Yao *et al.*'s scheme, Fan *et al.*'s scheme and our scheme.

1. In KeyGen stage, as depicted in Fig. 4(a), Yao *et al.*'s scheme has more complicated public parameters, so more calculations are required. Compared to our scheme, Fan et al.'s scheme generates additional decryption public-private key pairs for the user. In particular, both parts of the attribute key in the comparison scheme increase linearly with increasing attributes, whereas part of the attribute key in our scheme is constant.
2. In Encryption stage, as depicted in Fig. 4(b), Yao *et al.*'s scheme performs a large of encryption operations in $C_i^3 = \Pi_{i=0}^{l} h_{j,i}^{\lambda^{\rho(i)}}$ due to the complex public parameters. In addition, with the increase of attributes, the computation cost of this scheme will increase significantly, which is much higher than the other two schemes. Fan *et al.*'s scheme additionally calculates $C = h^s$ for the decryption stage. This scheme takes an average of 0.0266s longer than ours.
3. In Decryption stage, Fan *et al.*'s scheme and ours support decryption outsourcing. As shown in Fig. 4(c), Fan *et al.* perform an additional pairing and division operations. In the Pre-Dec process, our scheme is more efficient. Fin-Dec process is shown in Fig. 4(d), VDR performs all decryption operations



(a) Storage Cost of VDR's Keys (b) Storage Cost of CT piece

Fig. 5. Computational Cost of Keys and Ciphertext

in Yao *et al.*'s scheme, which has the most expensive decryption cost. The computation cost of VDR is constant in Fan *et al.*'s scheme and ours.

7.2 Storage Cost

We compared the storage cost of the three schemes at different stages.

1. In Setup stage, Yao *et al.* additionally generate the public key $\{h_{1,1}, ..., h_{l,N}\}$ and Fan *et al.* additionally calculate $\{f = g^{1/\beta}, h = g^{\beta}\}$. Therefore, our scheme has the lowest storage cost in this stage.
2. In KeyGen stage, as depicted in Fig. 5 (a), Fan *et al.*'s scheme additionally stores $\{D_{AA} = g^{\frac{y_i+\beta}{\beta r_{uid}}}\}$ than Yao *et al.*'s scheme. Since only $\{K_{i,1}\}$ in ASK increases linearly with the attributes, the storage cost of our scheme is nearly half of Yao *et al.*'s scheme.
3. In Encryption stage, Fan *et al.* additionally store $C = h^s$.
4. In Decryption stage, we store pieces of C instead of full C in different RSU. We assume that there are 30 RSUs maintaining the blockchain and $t = 18$. As shown in Table 2, each RSU requires only 39.2B of storage space on average. While the comparison schemes require 312B storage space.

Furthermore, for our scheme, we calculate the storage cost with different numbers (denote as n) of RSU and recovery thresholds (denote as t) in Table 3 and Fig. 5 (b). The results show that piece storage can greatly reduce the storage cost of ciphertext.

In the three schemes, the communication cost mainly exists in the process of key distribution and ciphertext transmission. The communication cost is similar to the storage cost for keys and ciphertexts. Therefore, we no longer analyze the communication cost.

In summary, in terms of computation cost, our scheme reduces the key generation time of AA, the encryption time of VDP and the decryption time of VDR. In terms of storage cost, our scheme reduces the key storage for VDR and the ciphertext storage for each RSU. Therefore, our scheme is lightweight.

Table 2. Comparison of storage cost of C in different schemes

Yao *et al.* [27]	**Fan *et al.* [8]**	**BLAC**		
		minimum	maximum	average
312 B	312 B	22 B	47 B	39.2 B

Table 3. Comparison of storage cost in different n-t pairs

n-t / value	10-6	20-12	30-18	40-26	50-32	60-38
minimum	56 B	30 B	22 B	16 B	14 B	13 B
maximum	60 B	44 B	47 B	56 B	66 B	79 B
average	58.5 B	39.3 B	39.2 B	45.0 B	52.9 B	62.4 B

8 Conclusion

In this article, we proposed a new CTST model to estimate the credibility of group. Then, we proposed the BLAC scheme for secure and lightweight data access control so as to resist tampering attack and collusion attack. To overcome the shortcomings of efficiency, We designed ciphertext piece storage and recovery algorithms to realize lightweight data storage cost, and outsourced a pre-decryption algorithm to the CS to reduce the computation cost for the RSU. The finally security analysis and experiment results show that our scheme is secure and lightweight.

Acknowledgements The authors of this article would like to thank the editor. This work is supposed by the National Natural Science Foundation of China (NOs. U1905211, 61902289).

References

1. Alladi, T., Chamola, V., Sahu, N., Venkatesh, V., Goyal, A., Guizani, M.: A comprehensive survey on the applications of blockchain for securing vehicular networks. IEEE Communications Surveys & Tutorials (2022)
2. Beimel, A.: Secure schemes for secret sharing and key distribution. Phd Thesis Israel Institute of Technology Technion (1996)
3. Bethencourt, J., Sahai, A., Waters, B.: Ciphertext-policy attribute-based encryption. In: 2007 IEEE Symposium on Security and Privacy (SP'07). pp. 321–334. IEEE (2007)
4. Castro, M., Liskov, B., et al.: Practical byzantine fault tolerance. In: OSDI. vol. 99, pp. 173–186 (1999)
5. De, S.J., Ruj, S.: Efficient decentralized attribute based access control for mobile clouds. IEEE Transactions on Cloud Computing **8**(1), 124–137 (2017)

6. Dib, O., Brousmiche, K.L., Durand, A., Thea, E., Hamida, E.B.: Consortium blockchains: Overview, applications and challenges. International Journal On Advances in Telecommunications **11**(1&2), 51–64 (2018)
7. Fabric, H.: Hyperledger Fabric, `https://www.hyperledger.org/projects/fabric`, (2017)
8. Fan, K., Pan, Q., Zhang, K., Bai, Y., Sun, S., Li, H., Yang, Y.: A secure and verifiable data sharing scheme based on blockchain in vehicular social networks. IEEE Transactions on Vehicular Technology **69**(6), 5826–5835 (2020)
9. Han, D., Pan, N., Li, K.C.: A traceable and revocable ciphertext-policy attribute-based encryption scheme based on privacy protection. IEEE Transactions on Dependable and Secure Computing (2020)
10. He, Q., Zhang, N., Wei, Y., Zhang, Y.: Lightweight attribute based encryption scheme for mobile cloud assisted cyber-physical systems. Computer Networks **140**, 163–173 (2018)
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 of Things Journal **6**(3), 4660–4670 (2019)
12. Khowaja, S.A., Khuwaja, P., Dev, K., Lee, I.H., Khan, W.U., Wang, W., Qureshi, N.M.F., Magarini, M.: A secure data sharing scheme in community segmented vehicular social networks for 6g. IEEE Transactions on Industrial Informatics **19**(1), 890–899 (2022)
13. Liang, W., Yang, Y., Yang, C., Hu, Y., Xie, S., Li, K.C., Cao, J.: Pdpchain: A consortium blockchain-based privacy protection scheme for personal data. IEEE Transactions on Reliability (2022)
14. Lu, Y., Zhang, J., Qi, Y., Qi, S., Zheng, Y., Liu, Y., Song, H., Wei, W.: Accelerating at the edge: A storage-elastic blockchain for latency-sensitive vehicular edge computing. IEEE Transactions on Intelligent Transportation Systems **23**(8), 11862–11876 (2021)
15. Lynn, B.: The Pairing-Based Cryptography (PBC) library, `https://crypto.stanford.edu/pbc/`, (2013, Jun 14)
16. Nasiraee, H., Ashouri-Talouki, M.: Privacy-preserving distributed data access control for cloudiot. IEEE Transactions on Dependable and Secure Computing **19**(4), 2476–2487 (2021)
17. Posner, J., Tseng, L., Aloqaily, M., Jararweh, Y.: Federated learning in vehicular networks: Opportunities and solutions. IEEE Network **35**(2), 152–159 (2021)
18. Pu, Y., Hu, C., Deng, S., Alrawais, A.: R^2PEDS: a recoverable and revocable privacy-preserving edge data sharing scheme. IEEE Internet of Things Journal **7**(9), 8077–8089 (2020)
19. Pu, Y., Xiang, T., Hu, C., Alrawais, A., Yan, H.: An efficient blockchain-based privacy preserving scheme for vehicular social networks. Information Sciences **540**, 308–324 (2020)
20. Qiu, J., Tian, Z., Du, C., Zuo, Q., Su, S., Fang, B.: A survey on access control in the age of internet of things. IEEE Internet of Things Journal **7**(6), 4682–4696 (2020)
21. Wang, Y., Su, Z., Zhang, K., Benslimane, A.: Challenges and solutions in autonomous driving: A blockchain approach. IEEE Network **34**(4), 218–226 (2020)
22. Wang, Y., Su, Z., Zhang, N., Chen, J., Sun, X., Ye, Z., Zhou, Z.: Spds: A secure and auditable private data sharing scheme for smart grid based on blockchain. IEEE Transactions on Industrial Informatics **17**(11), 7688–7699 (2021)

23. Waters, B.: Ciphertext-policy attribute-based encryption: An expressive, efficient, and provably secure realization. In: International Workshop on Public Key Cryptography. pp. 53–70. Springer (2011)
24. Wenxiu, D., Yan, Z., Deng, R.H.: Privacy-preserving data processing with flexible access control. IEEE Transactions on Dependable and Secure Computing **17**(2), 363–376 (2020)
25. Xu, S., Guo, C., Hu, R.Q., Qian, Y.: Blockchain-inspired secure computation offloading in a vehicular cloud network. IEEE Internet of Things Journal **9**(16), 14723–14740 (2021)
26. Yang, W., Guan, Z., Wu, L., Du, X., Guizani, M.: Secure data access control with fair accountability in smart grid data sharing: An edge blockchain approach. IEEE Internet of Things Journal **8**(10), 8632–8643 (2021)
27. Yao, Y., Chang, X., Mišić, J., Mišić, V.B.: Lightweight and privacy-preserving id-as-a-service provisioning in vehicular cloud computing. IEEE Transactions on Vehicular Technology **69**(2), 2185–2194 (2020)
28. Zhang, W., Zhang, Z., Xiong, H., Qin, Z.: Phas-hekr-cp-abe: partially policy-hidden cp-abe with highly efficient key revocation in cloud data sharing system. Journal of Ambient Intelligence and Humanized Computing pp. 1–15 (2022)
29. Zhao, C., Xu, L., Li, J., Fang, H., Zhang, Y.: Toward secure and privacy-preserving cloud data sharing: Online/offline multiauthority cp-abe with hidden policy. IEEE Systems Journal **16**(3), 4804–4815 (2022)
30. Zhong, H., Zhou, Y., Zhang, Q., Xu, Y., Cui, J.: An efficient and outsourcing-supported attribute-based access control scheme for edge-enabled smart healthcare. Future Generation Computer Systems **115**, 486–496 (2021)
31. Zhong, Y., Hua, K., Li, P., Deng, D., Liu, X., Chen, Y.: Dynamic periodic location encounter network analysis for vehicular social networks. IEEE Transactions on Vehicular Technology **70**(8), 7453–7463 (2021)

A Pseudocode of the contract

We design contract pseudocodes of the BLAC scheme.

In Setup stage, system initialization contract 4 shows the process of RSU registering identities and reaching consensus to generate the genesis block. In Encryption stage, the process of VDP generating pieces and storing them in different RSU is shown in data storage contract 5. In Decryption stage, VDR gets pieces and recovers CT in data recovery contract 6.

Algorithm 4 System Initialization Contract

$k = 0, kC = 0, L_{rsu} = \varnothing$
AA generates PP and MSK
AA sets $thresh$ and $k = 0$
AA generates rsu and $\{PK_{rsu}, SK_{rsu}\}$ for RSU
AA generates $H_P = H(PP)$
AA broadcasts PP and H_P to RSU
RSU calculates $H'_P = H(PP)$
RSU calculates $Sig_{rsu} = H(rsu)^{SK_{rsu}}$ to AA
while $k < thresh$ **do**
    if $H_P \neq H'_P$ **then**
        RSU sends 0 to AA
        AA sends PP and H_P to RSU
        RSU calculates $H'_P = H(PP)$
        $k = k + 1$
    if $e(g, Sig_{rsu}) \neq e(PK_{rsu}, H(rsu))$ **then**
        AA sends 0 to RSU
        RSU calculates $Sig_{rsu} = H(rsu)^{SK_{rsu}}$ to AA
        $k = k + 1$
    if $H_P = H'_P$ and $e(g, Sig_{rsu}) = e(PK_{rsu}, H(rsu))$ **then**
        Add $\{rsu, PK_{rsu}\}$ to the list L_{rsu}
        RSU broadcasts H'_P
        $k = 0$
        break
AA generates Genesis Block $B = \{PP, L_{rsu}, time\}$ to RSU
for each $rsu \in L_{rsu}$ **do**
    rsu receives $\{H'_P\}$
    for each $h \in \{H'_P\}$ **do**
        if $H_P = h$ **then**
            $kC = kC + 1$
            if $kC > N_{RSU}/3$ **then**
                rsu records B
                $kC = 0$
                break

Algorithm 5 Data Storage Contract

Let lB refer to the current latest block in Blockchain
VDP sets the threshold t and generates pieces $\{C_i\}_{i\in[1,n]}$
Let $C' = \{(M,\rho), \{C_i^1, C_i^2\}_{i\in[1,l]}\}$
VDP calculates $HC' = H(C')$ and $HCT = H(CT)$
VDP stores $\{(M,\rho), \{C_i^1, C_i^2\}_{i\in[1,l]}, HC', HCT\}$ in the CS
for each $i \in [1,n]$ **do**
    VDP calculates $Sig_{vdp}(C_i) = H(C_i)^{SK_{vdp}}$
    VDP sends $\{Sig_{vdp}(C_i), C_i, (M,\rho), NRSU\}$ to the i-th RSU
    if $e(g, Sig_{vdp}(C_i)) = e(PK_{vdp}, H(C_i))$ **then**
        Let $B_i = \{HCT, t, HC_i, rsu\}$
        The i-th RSU broadcasts B_i to the other RSU
VDP specifies the Nearest RSU (NRSU)
NRSU builds block $B = \{HCT, HC', t, \{HC_i, rsu\}_{i\in[1,n]}\}$
NRSU gets the hash of the current latest block HlB
NRSU calculates $HB_{NRSU} = H(B||time||HlB)$
NRSU generates $lB' = \{B, time, HB_{NRSU}\}$
NRSU broadcasts lB' to other RSU
for each $rsu \in L_{rsu}$ **do**
    Let $num = 0$
    for each $i \in [1,n]$ **do**
        rsu calculates $HB' = H(B||time||HlB)$
        if $\{HC_i, rsu\} \in B$ and $HB' = HB_{NRSU}$ **then**
            $num = num + 1$
            rsu stores $\{HCT, C_i\}$ and broadcasts *accept*
        else
            rsu broadcasts *reject*
    if $num > n/3$ **then**
        rsu uploads lB' as latest block

Algorithm 6 Data Recovery Contract

Let $nC = 0$ refer to the number of the correct C_i received
VDR sends vdr, Att_{vdr}, HCT to the NRSU
VDR receives $\{HCT, HC', t, \{HC_i, rsu\}_{i\in[1,n]}\}$ from the blockchain
NRSU receives $(M,\rho) \in C'$ from the CS
if $I = \{i|\rho(i) \in Att_{vdr}\}$ meets the access policy (M,ρ) **then**
    NRSU broadcasts $\{HCT, I\}$ to other RSU
else
    return $\perp$
RSU receives $(M,\rho) \in C'$ from the CS
if I meets (M,ρ) **then**
    RSU sends C_i to VDR
while $nC < t$ **do**
    VDR receives C_i from the i-th RSU
    if $H(C_i) \in \{HC_i, rsu\}$ **then**
        $nC = nC + 1$
VDR obtions $\omega_i \in \mathbb{Z}_p$ associated with I
VDR recovers C and gets CT