Noname manuscript No.
(will be inserted by the editor)

A Verifiable Hidden Policy CP-ABE with Decryption Testing Scheme and its Application in VANET

Yang Zhao · Hu Xiong

Received: date / Accepted: date

Abstract As a new kind of cryptographic primitive, attribute-based encryption(ABE) is widely used in various complex scenarios because it has the characteristics of access control while encrypting messages. However, the existing CP-ABE encryption schemes have some inherent defects, such as lack of privacy-preserving and inefficient decryption. These weak points make them difficult to play a role in scenarios with high real-time and high data confidentiality requirements, for example, vehicle ad hoc network(VANET). Therefore, we proposed A Verifiable Hidden Policy CP-ABE with Decryption Testing Scheme(or VHPDT, for short). It has the following features: hidden policy to privacy-preserving and outsourced decryption testing can verify the correctness of the decrypted result. Further, we apply it into VANET.

Keywords CP-ABE;hidden policy;verifiable decryption testing;VANET

1 Introduction

With the rapid development of cloud computing and the Internet of Things, more and more people are willing to use cloud platform as a medium for data transmission and sharing. Meanwhile, innumerable applications are established to satisfy and facilitate life needs of human beings. However, the premise of all this is that the cloud computing platform can provide a secure, trusted

Yang Zhao
University of Electronic Science and Technology of China, Chengdu, 610054 , China.
E-mail: zhaoyang@uestc.edu.cn

✉ Hu Xiong
University of Electronic Science and Technology of China, Chengdu, 610054 , China.
Guangxi Colleges and Universities Key Laboratory of cloud computing and complex systems, Guilin University of Electronic Technology, Guilin, 541004, China.
E-mail: xionghu.uestc@gmail.com

environment. Not only can it provide fine-grained access control service for users, it also ensures the privacy of users.

Since attribute-based encryption (ABE) has the characteristics of one-to-many encryption and fine-grained access control, the traditional solution to the above situation is ABE. Specifically, there are usually three entities in an ABE scheme, the data owner, the data consumer and the authority. Authority generates the public key, the master secret key and the private key. Data owner utilizes the public key to encrypt data along with the attributes. Data user can decrypt ciphertext with secret key, and the data decryption is possible if and only if at least d component of the attributes in the encrypted data match with attributes in secret key.

Despite the ABE scheme seems feasible, there are still some meticulous and in-depth questions we need to consider. Fig. 1 shows a typical VANET system architecture. For a VANET system, it can be divided into three parts: **OBU** (On-Board Unit), **RSU** (Road Side Unit) and **CSP** (Cloud Server Provider). **OBU** is abstracted from vehicle sensor and serves as information transmission node in the system. The **RSU** is set on both sides of the road to provide services for information transmission in system and **CSP** is a completely secure and trustworthy cloud server. In Fig. 1, we assume that **Truck1** will send a message to **Car3** via **CSP**. Refer to standard or most ABE schemes, the following steps will be executed. **Truck1** first encrypts the data to generate ciphertext that will be sent to **CSP** along with the access structure. Due to the inherent defects of ABE, it is worth nothing that the access structure here exists in plaintext. The **CSP** then transmits the ciphertext to the recipient **Car3**. After receiving the ciphertext, **Car3** decrypts the ciphertext with its own private key. The above steps are normal, but we consider the following situation. If the ciphertext is intercepted by a malicious attacker during transmission, although the malicious attacker cannot read the specific information of the ciphertext, he may obtain some sensitive information through the access structure transmitted in plaintext format. For example, in Fig. 1, the malicious attacker may get the information that is sent to **Alice** and her license plate is **Z08C06**. This is information disclosure obviously. It is unacceptable to users because it can lead to unpredictable consequences such as tracking, telecom fraud and so on.

1.1 Our Contribution

In this paper, we build a verifiable hidden policy CP-ABE with decryption testing scheme (or VHPDT, for short), which is an extension of the typical ABE scheme with a tree-based access structure, so as to achieve secure, efficient, flexible and fine-grained access control. If VHPDT is applied to VANET, privacy-presenting widehat exist on all message transmission paths, and after our optimization, decryption will be more efficient compared to existing schemes, giving the user a friendly experience.

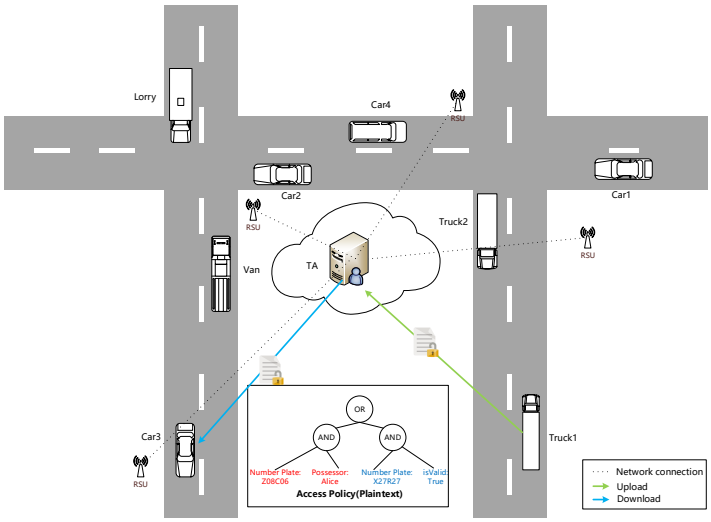

Fig. 1 VANET Structure

1.2 Related Work

Sahai and Waters proposed the concept of fuzzy Identify-Based-Encryption(IBE) initially, which was the prototype of ABE. Later, based on IBE, KP-ABE and CP-ABE as the variants of ABE were proposed. This paper focuses on the three aspects of CP-ABE: hidden policy, verifiable outsourcing schemes and applications in VANET. To realize the function of CP-ABE privacy protection, in 2008, Nishide et al. proposed a partially hidden CP-ABE scheme. They used AND gate on multi-valued attributes with wild-card access structures and used inner product predicate encryption techniques to hide the information of access structure. Later, Lai et al. improved the Nishide et al. scheme and proposed a fully secure hidden policy scheme with the same access structure used in Nishide's scheme. The problem with this scheme is that the size of the ciphertext grows with the number of attributes which leads to higher computational costs. Li et al. used dual-system cryptography to achieve a completely secure access structure hiding scheme over a composite order group. Hur proposed an attribute-based encryption scheme for hidden policy supporting arbitrary monotonic expressions and applied it into smart grid field. But the weak points are that the user has a long length of secret key in the scheme, and no security proof is given in paper. Xu et al. proposed a CP-ABE scheme that supports access structure hiding based on the tree access structure, which further improved the expressiveness of the access strategy but efficiency is the bottleneck of the scheme. In 2016, Phuong et al. proposed a new scheme to overcome the ciphertext size problem in hidden strategies. However, the cumbersome user private key calculation process and complex encryption and decryption operations limit its application.

In order to solve the issue of low decryption efficiency, Green et al. firstly introduced the notion of ABE with outsourced decryption. Afterwards, Lai et.al optimized Green's scheme and added verification function to guarantee the correctness of the transformation done by the cloud server. It perfectly settles the issues including low efficiency and correctness of transformation ciphertext, while there are parallel instances in the encryption and decryption algorithms, it can increase communication overhead. Lin et al. used key derivation function(KDF) and random function techniques to successfully reduce excessive communication overhead.

Consider the practical applications of CP-ABE in VANET. Huang et al. were the first to introduce CP-ABE into VANET, and some other schemes haver been proposed since then. But how to ensure that messages are disseminated in VANET in an efficient, flexible and secure manner is still a challenge.

1.3 Organization

The remainder of this article consists of the followings. In section2, we list some preliminaries that are used in our scheme. Then, the concrete scheme and an EMR system are entirely demonstrated in section3. Section4 proposes the security proof of our scheme. Finally, in section5, the performance analysis of our scheme is illustrated.

2 Preliminaries

2.1 Access Structure

The same access structure as [**??**] we adopt. Let $P_1, P_2, ..., P_n$ be a set of parties. A collection $\mathbb{A} \subseteq 2^{\{P_1,P_2,...,P_n\}}$ is monotone if $\forall B, C$: if $B \in \mathbb{A}$ and $B \subseteq C$ then $C \in \mathbb{A}$. An access structure (respectively, monotone access structure) is a collection(respectively, monotone collection) $\mathbb{A}$ of non-empty subsets of $\{P_1, P_2, ..., P_n\}$, i.e., $\mathbb{A} \subseteq 2^{\{P_1,P_2,...,P_n\}} \setminus \{\varnothing\}$. The sets in $\mathbb{A}$ are called the authorized sets, and the sets not in $\mathbb{A}$ are called the unauthorized sets.

In our scheme, the P_i are replaced by attributes. We define the collection $\mathbb{A} = \{a_1, a_2, .., a_n\}$ contains all of the authorized attributes in system where n is the number of authorized attributes. For $\forall i \in \mathbb{Z}_N$, the value set of attribute a_i is $\mathbb{V}_i = \{v_{i,1}, v_{i,2}, ..., v_{i,n_i}\}$, similarly, n_i is the amount of elements of $\mathbb{V}_i$. The collection $\mathbb{L} = \{l_1 = v_{1,t_1}, l_2 = v_{2,t_2}, ..., l_k = v_{k,t_k}\}$ is a subset of system attributes set $\mathbb{A}$ and it is used to denote the collection of user attributes where k is the number of user attributes.

2.2 Composite Order Bilinear Groups

This paper utilizes a bilinear group with a product of three unique prime numbers. Assume that the group generation algorithm $\mathcal{G}$ has a security input

parameter 1^λ and it outputs a tuple $(p, q, r, \mathbb{G}, \mathbb{G}_T, e)$ where p, q, r are distinct primes$(p, q, r > 2^\lambda)$, $\mathbb{G}$ and $\mathbb{G}_T$ are cyclic groups of order $N = pqr$. $\mathbb{G}_p, \mathbb{G}_q, \mathbb{G}_r$ whose order are p, q, r separately are the subgroups of group $\mathbb{G}$. The $e : \mathbb{G} \times \mathbb{G} \to \mathbb{G}_T$ is a map with the following requirements:

- Bilinear: $\forall g, h \in \mathbb{G}, \forall a, b \in \mathbb{Z}_N, e(g^a, h^b) = e(g, h)^{ab}$;
- Non-degenerate: $\exists g \in \mathbb{G}$ such that $e(g, g)$ has the order N in $\mathbb{G}_T$;
- Computable: $\mathbb{G}$ and $\mathbb{G}_T$ are bilinear groups if the group operations in $\mathbb{G}, \mathbb{G}_T$ and the bilinear map $e : \mathbb{G} \times \mathbb{G} \to \mathbb{G}_T$ are both computable in polynomial time.

We use $\mathbb{G}_p$ and $\mathbb{G}_q$ to denote the subgroups of $\mathbb{G}$ with order p and q respectively. Note that the following two properties:

1). $\forall h_p \in \mathbb{G}_p, \forall h_q \in \mathbb{G}_q, e(h_p, h_q) = 1$;
2). $\forall h_p \in \mathbb{G}_p, \forall h_q \in \mathbb{G}_q, \forall a, b, c, d \in \mathbb{Z}_N$,
$e(h_p^a h_q^b, h_p^c h_q^d) = e(h_p, h_q)^{ab} = e(h_p, h_q)^{cd}$.

2.3 Complexity Assumptions

Now we cite the complexity assumptions we use in this paper. The assumption is just the subgroup decision problem in the case where the group order is the product of three prime orders. Note that the assumptions have two characteristics, non-interactive and fixed size like in **??**. Let $\mathbb{G}_{pq}$ indicates the subgroup with order pq.

Assumption 1 *We define the following distributions:*
$\mathbb{G} = (N = pqr, \mathbb{G}, \mathbb{G}_T, e) \leftarrow \mathcal{G}(1^\lambda), g_p \xleftarrow{R} \mathbb{G}_p, g_r \xleftarrow{R} \mathbb{G}_r, D = (\mathbb{G}, g_p, g_r)$,
$T_1 \xleftarrow{R} \mathbb{G}_{pq}, T_2 \xleftarrow{R} \mathbb{G}_p$

Definition 1
We define the advantage of algorithm $\mathcal{A}$ to break Assumption1 as: $Adv^1_{g,\mathcal{A}(\lambda)} = |Pr[\mathcal{A}(D, T_1) = 1] - Pr[\mathcal{A}(D, T_2) = 1]|$.

Assumption 2 *We define the following distributions:*
$\mathbb{G} = (N = pqr, \mathbb{G}, \mathbb{G}_T, e) \leftarrow \mathcal{G}(1^\lambda), g_p, X_1 \xleftarrow{R} \mathbb{G}_p, X_2 \xleftarrow{R} \mathbb{G}_q, g_r \xleftarrow{R} \mathbb{G}_r, D = (\mathbb{G}, g_p, g_r, X_1, X_2)$,
$T_1 \xleftarrow{R} \mathbb{G}_{pq}, T_2 \xleftarrow{R} \mathbb{G}_p$

Definition 2
We define the advantage of algorithm $\mathcal{A}$ to break Assumption2 as: $Adv^2_{g,\mathcal{A}(\lambda)} = |Pr[\mathcal{A}(D, T_1) = 1] - Pr[\mathcal{A}(D, T_2) = 1]|$.

Assumption 3 *We define the following distributions:*
$\mathbb{G} = (N = pqr, \mathbb{G}, \mathbb{G}_T, e) \leftarrow \mathcal{G}(1^\lambda), \omega, s \in \mathbb{Z}_N, g_p \xleftarrow{R} \mathbb{G}_p, X_2, Y, Z_2 \xleftarrow{R} \mathbb{G}_q, g_r \xleftarrow{R} \mathbb{G}_r, D = (\mathbb{G}, g_p, g_p^\omega X_2, g_p^s Y_2, Z_1 Z_2, g_r)$,
$T_1 \xleftarrow{R} e(g_p, g_p)^{\omega s}, T_2 \xleftarrow{R} \mathbb{G}_p$

Definition 3
We define the advantage of algorithm $\mathcal{A}$ to break Assumption3 as: $Adv^3_{g,\mathcal{A}(\lambda)} = |Pr[\mathcal{A}(D, T_1) = 1] - Pr[\mathcal{A}(D, T_2) = 1]|$.

Definition 4 *If algorithm $\mathcal{A}, Adv^1_{g,\mathcal{A}(\lambda)}, Adv^2_{g,\mathcal{A}(\lambda)}, Adv^3_{g,\mathcal{A}(\lambda)}$ can be ignored in any polynomial time, the $\mathcal{G}$ will satisfy Assumption1∼3.*

2.4 Security Model

We will give a adaptively-CCA security model for our scheme. The security game between adversary $\mathcal{A}$ and Challenger $\mathcal{C}$ is described as followings:

- **Setup:** The challenger $\mathcal{C}$ runs initialization algorithm $Setup(1^\lambda)$ to generates public key PK and master secret key MSK. Then $\mathcal{C}$ stores MSK and transfers PK to $\mathcal{A}$.
- **Phase1:** $\mathcal{A}$ asks $\mathcal{C}$ secret key of attribute sets L adaptively. $\mathcal{C}$ generates the secret key SK_L by running $KeyGen$ algorithm and transfers SK_L to $\mathbb{A}$. Note that $\mathbb{A}$ can repeatedly ask the secret key multiple times.
- **Challenge:** $\mathcal{A}$ submits two messages m_0, m_1 which they have the same length and two access structures T_0, T_1. Note that any attribute set isn't satisfied with T_0 and T_1. $\mathcal{C}$ flips a random coin $b \in \{0, 1\}$ and then calculates $c^* = Encrypt(PK, m_b, T_b)$. Finally, $\mathcal{C}$ sends $c*$ to $\mathcal{A}$ as challenge ciphertext.
- **Phase2:** Phase2 is the same as Phase1.
- **Guess:** The $\mathcal{A}$ outputs a guess b' of b where $b' \in \{0, 1\}$.

If $b' = b$, the adversary wins this security game. And the advantage of adversary in this security game is defined as $Adv_{\mathcal{A}} = |Pr[b' = b] - \frac{1}{2}|$.

2.5 CP-ABE Scheme

The ciphertext policy attribute-based encryption(CP-ABE) scheme consists of four fundamental algorithms: Setup, Encrypt, Key Generation and Decrypt.

- Setup(k). The setup algorithm takes no input other than the security parameter k. It outputs the public parameters PK and a master key MK.
- Key Generation(MK, S). The key generation algorithm takes the master key MK and a set of attributes S that describe the key as input. It outputs a private key SK.
- Encrypt(PK, M, A). The encryption algorithm takes the public parameters PK, a message M and an access structure A over the universe of attributes as input. The algorithm will encrypt M and produce a ciphertext C_T, so that only a user that possesses a set of attributes that satisfies the access structure will be able to decrypt the message. We will assume that the ciphertext implicitly contains A.

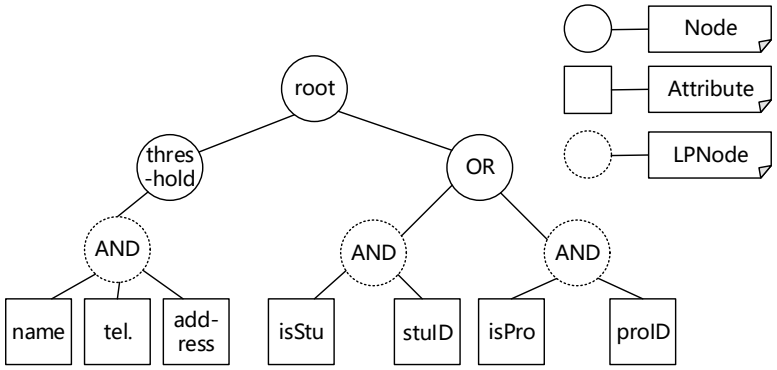

Fig. 2 Access Tree

- Decrypt(PK, C_T, SK). The decryption algorithm takes the public parameters PK, a ciphertext C_T which contains an access policy A, and a private key SK as input. If the attributes set satisfies the access structure A, the algorithm will decrypt the ciphertext and return a message M, otherwise return the error symbol.

3 Our Construction

In this section, we will introduce our scheme in detail. A novel access structure based on tree structure is presented firstly and then the main scheme will be described.

3.1 Access Tree

To enhance the expression capability of access structure, we utilize tree structure in our scheme. Let $\mathcal{T}$ represents a hierarchical tree. The inner nodes of $\mathcal{T}$ are described the logic relationship of different access policies, including *AND, OR, Threshold*. Besides, the leaf nodes of $\mathcal{T}$ represent attribute marked $a_i = v_{i,t_i} (i \in \mathbb{Z}_n, t_i \in \mathbb{Z}_{n_i})$. For each leaf node, we can always seek its parent node and called it *LPNode* for simplicity. In our tree structure *LPNode* can only express the *AND* relationship. Hence, it is obvious that a kind of attribute value can only appear once under each *LPNode*. Here is an access tree example in Figure2.

3.2 Our Scheme

Our scheme is consisted of the following algorithm:

– Setup(1^λ): TA runs algorithm $\mathcal{G}(1^\lambda)$ to product four elements including $N = p \cdot q \cdot r, \mathbb{G}, \mathbb{G}_T, e$. The subgroups $\mathbb{G}_p, \mathbb{G}_q, \mathbb{G}_r$ are from group $\mathbb{G}$. Note that, the subgroup $\mathbb{G}_p$ is used to subsequent steps. Considering security of the whole scheme, the subgroup $\mathbb{G}_r$ is used to generate random parameters. The subgroup $\mathbb{G}_q$ is mainly used to security analysis. And g_p, g_q, g_r are the generators of three subgroups $\mathbb{G}_p, \mathbb{G}_q, \mathbb{G}_r$. $\mathbb{U} = \{a_1, a_2, \ldots, a_n\}$ expresses all the attributes in the whole system. For each attribute, TA randomly chooses $a_{i,t_i} \in \mathbb{Z}_N, R_{i,t_i} \in \mathbb{Z}_n$ and calculates $A_{i,t_i} = g_p^{a_{i,t_i}} R_{i,t_i}$. In addition, TA generates three more parameters $\omega, \bar{\omega} \in \mathbb{Z}_N/0$ and $R_0 \in \mathbb{G}_r$. Two more parameters $u, v \in \mathbb{G}_p$ and a key derivation function **KDF** with output length ℓ are chosen by TA. Last, TA computes $A_0 = g_p R_0, Y = e(g_p, g_p)^\omega, \bar{Y} = e(g_p, g_p)^{\bar{\omega}}$. The public key PK and master secret key MSK are defined as formula(1) and (2):

$$PK = \left(A_0, g_r, \{A_{i,t_i}\}_{1 \le t_i \le n_i, 1 \le i \le n}, Y, \bar{Y}, u, v, \mathbf{KDF}, \ell\right) \tag{1}$$

$$MSK = \langle g_p, \{a_{i,t_i}\}_{1 \le t_i \le n_i, 1 \le i \le n}, \omega, \bar{\omega} \rangle \tag{2}$$

– KeyGen(MSK,PK,L): For $\forall i \in \mathbb{Z}_k$, the DO randomly selects $d_i \in \mathbb{Z}_N$. Therefore, the SK_L is like the format of formula(3):

$$SK_L = \langle D_0 = g_p^{\omega - \sum_{i=1}^k d_i}, \bar{D}_0 = g_p^{\bar{\omega} - \sum_{i=1}^k d_i}, \{D_j = g_p^{\frac{d_i}{a_{i,t_i}}}\}_{1 \le i \le k} \rangle \tag{3}$$

– Encrypt(PK,m,T):

a). DO selects random values $R_0' \in \mathbb{G}_r$ and $s \in \mathbb{Z}_N$ to encrypt the message $m \in \mathbb{G}$ as follows:

$$C_0 = A_0^s R_0', \ \tilde{C} = m Y^s \tag{4}$$

b). DO sets a session key $\mathbf{DK} = e(g_p, g_p)^{\omega s}$, and computes $\mathbf{KDF}(\mathbf{DK}, \ell) = VKEY \parallel \eta$ where η is a random number and $VKEY$ is used to verify. Then $\widehat{C} = u^{H(VKEY)} v^{H(\eta)}$.

c). Let secret s be assigned in access tree and the root node of $\mathcal{T}$ is set to be s. All child nodes except for leaf nodes are marked un-assigned status and root node is assigned status. Then, for each unassigned node, do the following operations recursively:

- If the node logic operator is *AND*, the first $u-1$ child nodes are set to s_i where s_i is randomly generated in $\mathbb{Z}_N$, and the last child node is set to $s_u = s - \sum_{i=1}^{u-1} s_i$;
- If the node logic operator is *OR*, the child nodes of this node are set s;
- If the node logic operator is *Threshold* with the threshold value h, it will arbitrarily generate a polynomial f that satisfies $f(0) = s$ and the degree of polynomial is $h-1$. For the i-th node, the value is assigned $f(i)$.

d). Suppose that a *LPNode* in access tree has a shared secret s_α. Then calculate a ciphertext component for each attribute in system according to formula(5), where $s_{i,t_i} \in \mathbb{Z}_N$ is a random value and R'_{i,t_i} is also

a random parameter produced from $\mathbb{G}_r$. Therefore, the ciphertext of *LPNode* is like formula(6):

$$C_{i,t_i} = \begin{cases} A_{i,t_i}^{s_{i,t_i}} R'_{i,t_i}, & L_i \in L_\alpha \text{ and } v_{i,t_i} \notin L_\alpha \\ A_{i,t_i}^{s_\alpha} R'_{i,t_i}, & \text{otherwise} \end{cases} \tag{5}$$

$$C_\alpha = \langle H_\alpha = \bar{Y}^{s_\alpha}, \bar{C}_\alpha = A_0^{s_\alpha} R_\alpha, \{C_{i,t_i}\}_{i \le t_i \le n_i, 1 \le i \le n} \rangle \tag{6}$$

where $R_\alpha \in \mathbb{G}_r$. Ultimately, output the final ciphertext:

$$CT = \langle \tilde{C}, C_0, \{C_\alpha\}_{\forall \alpha \ in \ T}, \widehat{C} \rangle \tag{7}$$

– GenTk$_{out}$(PK,SK): DO selects a random factor $z \in \mathbb{Z}_N$ and does the following operations: $D_i^* = D_i^z, \bar{D}_0^* = \bar{D}_0^z, D_0^* = D_0^z, H_\alpha^* = H_\alpha^z$. Finally, the DO output the TK as the formula(8) and CT to outsourced server by security channel.

$$TK = \langle D_i^*, \bar{D}_0^*, H_\alpha^* \rangle \tag{8}$$

– Transform$_{out}$(CT,TK,PK):
  • Testing: In this process, outsourced server runs the following defined functions. If the node is a *LPNode*, it will execute *PreDecNode(α)*. Similarly, if the node is a normal node, it will execute *PreDecNode(β)* where u is the number of child of node β and h is the threshold value of node β.

$$PreDecNode(\alpha) = \prod_{i=1}^{k} e(C_{i,t_i}, D_i^*) \tag{9}$$

PreDecNode(β)

$$= \begin{cases} \prod_{i=1}^{h} PreDecNode(Child(\beta, i))^{\Delta_{i,\beta}(0)}, & Option(\beta) = Threshold \\ \prod_{i=1}^{u} PreDecNode(Child(\beta, i)), & Option(\beta) = AND \\ PreDecNode(Child(\beta, i)), & Option(\beta) = OR \end{cases} \tag{10}$$

Evidently θ_α will be a random value, if the *LPNode* cannot satisfy attributes. When θ_α is a random value, it indicates this *LPNode* is an error node. Hence, there is no necessary to participate in the subsequent decryption phase.

The outsourced server calculates $\theta_\alpha = \frac{H_\alpha^*}{e(\bar{C}_\alpha, \bar{D_0}^*) PreDecNode(\alpha)}$. If the *LPNode* satisfies the attributes represented by each leaf node under it, then it can obtain $PreDecNode(\alpha) = e(g_p, g_p)^{zds_\alpha}$. Further, the outsourced server performs the following calculations:

$$\begin{aligned} \theta_\alpha &= \frac{H_\alpha^*}{e(\bar{C}_\alpha, \bar{D_0}^*) PreDecNode(\alpha)} \\ &= \frac{\bar{Y}^{zs_\alpha}}{e(A_0^{s_\alpha} R_\alpha, g_p^{z(\bar{\omega}-d)}) e(g_p, g_p)^{zds_\alpha}} \\ &= \frac{e(g_p, g_p)^{\bar{\omega} z s_\alpha}}{e(g_p, g_p)^{z s_\alpha (\bar{\omega}-d)} e(g_p, g_p)^{zds_\alpha}} \\ &= 1 \end{aligned} \tag{11}$$

- PreDecrypt(CT,TK,PK): The next description operation is completed by executing formula(12) when $\theta_\alpha = 1$:

$$\begin{aligned} X &= \frac{1}{e(C_0, D_0^*)PreDecNode(root(T))} \\ &= \frac{1}{e(g_p^s R_0^s R_0', g_p^{z(\omega-d)})e(g_p, g_p)^{dsz}} \\ &= \frac{1}{e(g_p, g_p)^{sz(\omega-d)}e(g_p, g_p)^{dsz}} \\ &= \frac{1}{e(g_p, g_p)^{sz\omega}} \end{aligned} \tag{12}$$

  At last, outsourced server sets $T_1 = \widehat{C}$ and transfers $CT' = (T_1, X)$ to DataUser.

– Decrypt(CT',SK,CT): DataUser receives CT' sent by outsourced server and use SK to decrypt ciphertext. The decryption process is described as formula(13):

$$\frac{\tilde{C}}{(X)^{\frac{1}{z}}} = \frac{mY^s}{e(g_p.g_p)^{s\omega}} = \frac{me(g_p.g_p)^{s\omega}}{e(g_p.g_p)^{s\omega}} = m \tag{13}$$

– Verify(CT,CT'): DataUser verifies the correctness of result generated from outsourced server. The detail procedures are as followings:

a). If $T_1 = \widehat{C}$ holds, let $\mathbf{DF} = X^{-\frac{1}{z}} = e(g_p, g_p)^{s\omega}$ and $\mathbf{KDF}(\mathbf{DK}, \ell) = VKEY \parallel \eta$; otherwise, output the terminator $\perp$;

b). If $u^{H(VKEY)}v^{H(\eta)} = \widehat{C}$ holds, it denotes the testing results and outsourced decryption are proper, furthermore, DataUser can calculate a right message m; otherwise, output the terminator $\perp$;

3.3 Application in cloud computing

In this subsection, we introduce the practical application of our scheme. In the cloud computing environment, we apply the solution to the personal health record(PHR) system. The figure3 is the system architecture diagram of our scheme. For the sake of convenience, we will first introduce the components in our PHR system. Our system consists of the following five parts:

– PHR Provider: The PHR provider can be considered as input to the entire system. The PHR provider encrypts its personal health record using a public key and an expressive access policy. The encrypted personal health record is called encrypted data and uploaded to the database for storage.

– Trusted Authority: The credible authority will be divided into two parts, AAS and PKS are part of it, and KGC is another part. It is worth mentioning that whether AAS, PKS or KGC, they are all trustworthy. In other words, they are security servers for the whole system.

    - KGC: KGC is an abbreviation for Key Generation Center. In the system, KGC plays the important role of generating security parameters and public keys. Meanwhile, KGC sends the public keys to PHR providers and PHR consumers.
    - AAS and PKS: AAS means attribute authority server and PKS denotes private key server. AAS and PKS can be viewed as a tiny parallel system, which is a collaborative relationship. AAS authenticates the PHR consumers attributes and then requires PKS to generate the secret key for PHR consumers.
- DB Cluster: DB cluster is a group of database. The main task of DB cluster is to store encrypted PHR data. DB cluster consists of multiple machines for data security and system stability.
- PHR Consumer: PHR consumers play the role of obtaining encrypted PHR data and requesting CSP to decrypt data.
- CSP: CSP stands for cloud server provider. Like the Trusted Authority section above, the CSP will be divided into two parts. The first part is the Testing server and the second part is the PreDec server. Similarly, testing server and predec server form a parallel system to decrypt ciphertext.
    - Testing Server: Testing server runs the testing algorithm to detect whether the PHR consumers' attributes satisfy the access structure required to decrypt the ciphertext.
    - PreDec Server: After running the testing algorithm, the PreDec server runs the PreDec algorithm to perform a partial decryption operation, and sends the decrypted result to the PHR consumer.

4 Security Proof

In this section, we will give a proof of security of our scheme. The idea of Lai et.al inspires our thinking and based on dual encryption system invented by Waters et.al we completed our proof. Above all, for convenience, we denote the corresponding definitions used in Water's paper. Both ciphertext and private keys can take on one of two indistinguishable forms the *normal* or the *semi-functional* in dual system encryption system. A *normal* private key or ciphertext are generated from key generation or encryption algorithm. As for semi-functional private key, it can decrypt all normally generated ciphertext. However, decryption will fail if one try to decrypt a semi-functional ciphertext with a semi-functional privacy key. Similarly, semi-functional ciphertext can be decrypted only by normal private key. Note that the semi-functional ciphertext and semi-functional private keys are used in our proof process, and you can't search for them in the actual system.
The security of our scheme depends on three hypotheses which are mentioned in section 2.3 and we utilize hybrid argumentation techniques to prove the security of the scheme with the indistinguishability of a series of games. The followings displayed are the components of our proof procedure.

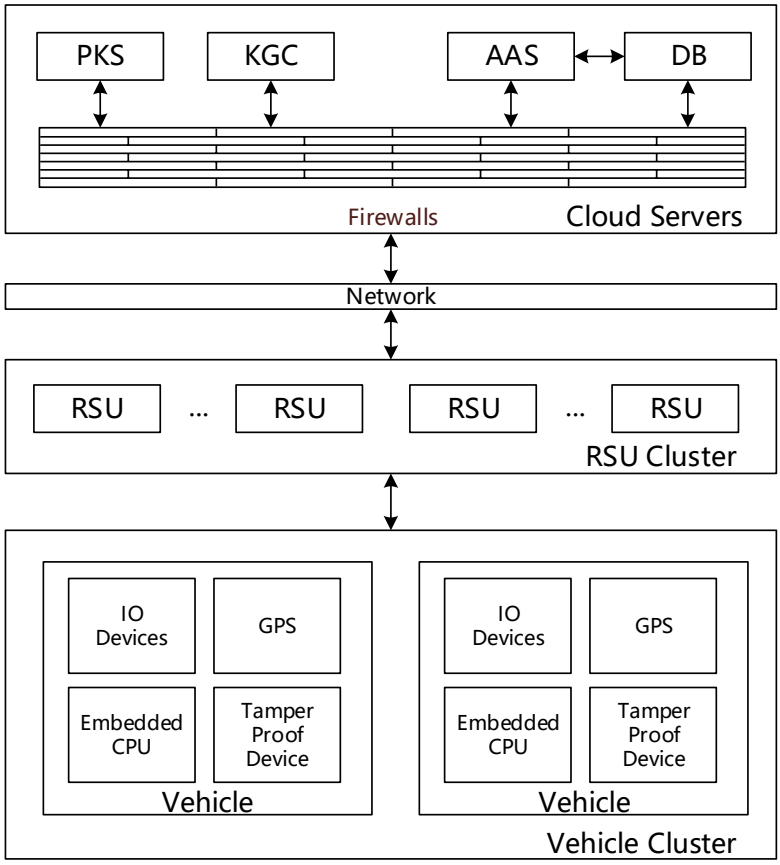

Fig. 3 Frame of VANET System

– Normal ciphertext: The normal ciphertext is like:

$$\langle \tilde{C}', C_0', \{(C_{i,t_i}')_{1\leq t_i\leq n_i, 1\leq i\leq n}\}\rangle \tag{14}$$

– Semi-functional ciphertext: Select a random number $x_0 \in \mathbb{Z}_N/0$ and in each *LINode*, for $\forall t_i (i \in \mathbb{Z}_n/0, t_i \in \mathbb{Z}_{n_i}/0)$, randomly generates $x_{i,t_i} \in \mathbb{Z}_N/0$, then, the semi-functional ciphertext is:

$$\langle \tilde{C} = \tilde{C}', C_0 = C_0' g_q^{x_0}, \{C_{i,t_i} = C_{i,t_i}' g_q^{x_{i,t_i}}\}_{1\leq t_i\leq n_i, 1\leq i\leq n}\rangle \tag{15}$$

– $Game_{real}$: The real game is composed of real private key and ciphertext;
– $Game_0$: The query private key is normal but challenge ciphertext is semi-functional in $Game_0$;
– $Game_{1,k}$: It denotes the challenge ciphertext is semi-functional. The private key of the first queries is semi-functional and the remains are normal.

– $Game_{final}$: The private key of all queries is semi-functional. The challenge ciphertext is a semi-functional ciphertext generated by encrypting a random message.

Lemma 1 *Suppose there exist a probabilistic polynomial time algorithm $\mathcal{A}$ that can win $Gmae_{real}, Game_0$ with advantage $Adv_{\mathcal{A}}^{Game_{real}} - Adv_{\mathcal{A}}^{Game_0} = \varepsilon$. Then we can build a PPT algorithm $\mathcal{C}$ to break Assumption1 with advantage ε.*

Proof Given the distribution in Assumption1, $D = \{\mathbb{G}, g_p, g_r\}$.

– Setup: This phase is the same as Setup algorithm in section 3.3.
– Phase1: Adversary can answer private key of any attribute set.
– Challenge: Adversary selects two messages m_0, m_1 with the same length and two access structures T_0, T_1. Note that any query cannot satisfy access structures T_0, T_1. Then adversary transfers them to challenger. After the challenger receives the parameters from adversary, challenger flips a coin $b \in \{0, 1\}$ and encrypts the received message to m_b by using a access structure T_b. Afterwards, challenger randomly selects some necessary parameters, $s \in \mathbb{Z}_N, R'_0 \in \mathbb{G}_r$ and generates $\tilde{C}^* = m_b e(g_p, X)^{\omega s}, C_0^* = X^s R'_0$ which are components of ciphertext. When $((L_i \in L_\alpha) \wedge (v_{i,t_i} \notin L_\alpha))$, challenger obtains the random value $s_{i,t_i} \in \mathbb{Z}_N$ *and* $s_{i,t_i} \neq s_\alpha$, set $C_{i,t_i}^* = X^{a_{i,t_i} \cdot s_{i,t_i}} R'_{i,t_i}$, otherwise, let $C_{i,t_i}^* = X^{a_{i,t_i} \cdot s_\alpha} R'_{i,t_i}$. Finally, challenger send the completed ciphertext $\langle \tilde{C}^*, C_0^*, \{\{C_{i,t_i}^*\}_{1 \leq i \leq n, 1 \leq t_i \leq n_i}\}_{\forall \alpha} \rangle$ to $\mathcal{A}$.
– Phase2: It is the same as Phase1.
– Guess: Adversary outputs a guess d' to challenger.

Apparently, if $X \in \mathbb{G}_p$ the answer ciphertext $c*$ must be normal ciphertext. Similarly, if $X \in \mathbb{G}_{pq}$, the answer ciphertext $c*$ have semi-function. Therefore, under the assumption that $\mathcal{A}$ have a non-negligible advantage $Adv_{\mathcal{A}}^{Game_{real}} - Adv_{\mathcal{A}}^{Game_0} = \varepsilon$, then challenger also can distinguish elements of $\mathbb{G}_p$ or $\mathbb{G}_{pq}$ with indistinguishable advantage.

Lemma 2 *If there exist a algorithm $\mathcal{A}$ that can distinguish between $Game_{real}$ and $Game_0$ with advantage $Adv_{\mathcal{A}}^{Game_{real}} - Adv_{\mathcal{A}}^{Game_0} = \varepsilon$, then a algorithm $\mathcal{C}$ that can break Assumption2 with identical advantage ε in probabilistic polynomial time is constructed.*

Proof Given the same conditions as Assumption2

– Setup: the similar step as in Lemma1.
– Challenge: In this phase, $\mathcal{A}$ sends the followings that consists of m_0, m_1 with the same bit-length and two access structure T_0, T_1(note that any attribute set can not satisfy them) to challenger. Subsequently, challenger selects a random value $b \in 0, 1$ and use the access structure T_b to encrypt message to generate new ciphertext m_b. Then challenger arbitrarily chooses $s \in \mathbb{Z}_N, R'_0 \in \mathbb{Z}_r$ and produces the semi-functional ciphertext by the corresponding parameters in MSK and assumption conditions $\tilde{C}^* =$

$m_b e(g_p, X_1X_2)^{\omega s}, C_0 = (X_1, X_2)^s R'_0$. Finally, when $((L_i \in L_\alpha) \wedge (v_{i,t_i} \notin L_\alpha))$, let $C^*_{i,t_i} = (X_1X_2)^{a_{i,t_i} s_{i,t_i}} R'_{i,t_i}$ where s_{i,t_i} is a random value and $s_{i,t_i} \in \mathbb{Z}_N$ *and* $s_{i,t_i} \neq s_\alpha$, otherwise, let $C^*_{i,t_i} = (X_1X_2)^{a_{i,t_i} s_\alpha} R'_{i,t_i}$. Hence, the ciphertext $\langle \tilde{C}^*, C^*_0, \{\{C^*_{i,t_i}\}_{1\leq i\leq n, 1\leq t_i\leq n_i}\}_{\forall \alpha}\rangle$ is transmit to $\mathcal{A}$.

– Phase1 and Phase2: Considering the $i-th$ query of $\mathcal{A}$, obviously, the private key query is divided the following three sections:
  - Case1: When $i > k$, challenger can correctly calculate normal private key using MSK and attribute sets from the query of $\mathcal{A}$, then challenger sends the normal private key to $\mathcal{A}$.
  - Case2: When $i < k$, according to the above definition of $Game_{1,k}$, the first $k-1$ private key of $Game_{1,k-1}$ and $Game_{1,k}$ is semi-functional. Therefore, challenger calculates to the semi-functional key and sends it to $\mathcal{A}$.
  - Case3: When $i = k$, for $\forall i \in \mathbb{Z}_k$, challenger randomly chooses $d_i \in \mathbb{Z}_N$ and generates private key $D'_0 = X^{\omega - d}, D'_i = X^{\frac{d_i}{a_{i,t_i}}}$ where $d_i \in \mathbb{Z}_N$,d_i is a random value and sets $d = \sum_{i=1}^{k} d_i$.
– Guess: Adversary outputs a guess d' to challenger.

One situation is that if $X \in \mathbb{G}_p$, let $X = \mathbb{G}_p$, hence, the generated key is normal private key, further, the game is $Game_{1,k-1}$. The other is that $X = g_p g_q^{\frac{y_0}{\omega - d}}, X \in \mathbb{G}_{pq}$, moreover, the generated private key is semi-functional key and the game is $Game_{1,k}$. Therefore, considering the above two situation, if $\mathcal{A}$ has a non-negligible advantage $Ada_{\mathcal{A}}^{Game_{1,k-1}} - Ada_{\mathcal{A}}^{Game_{1,k}} = \varepsilon$, then challenger also has a non-negligible advantage to distinguish the elements between $\mathbb{G}_{pq}$ and $\mathbb{G}_p$.

Lemma 3 *If there exist a probabilistic polynomial time algorithm $\mathcal{A}$ that has the advantage $Adv_{\mathcal{A}}^{Game_{1,u}} - Adv_{\mathcal{A}}^{Game_{final}} = \varepsilon$, then a probabilistic polynomial time algorithm $\mathcal{C}$ that is able to break Assumption3 with the advantage ε can be built.*

Proof Given the distributions in Assumption3, $D = (\mathbb{G}, g_p, g_p^\omega X_2, g_p^s Y_2, Z_1Z_2g_r)$.

– Setup: Repeat the step in Lemma1.
– Phase1 and Phase 2: In $Game_{1,u}$ and $Game_{final}$, the generated private keys are semi-functional. The challenger produces semi-functional private keys $D'_0 = g_p^\omega X_2 (Z_1Z_2)^{-d}, D'_i = (Z_1Z_2)^{\frac{d_i}{a_{i,t_i}}}$ based on queried attributes sets by $\mathcal{A}$ where $i \in \mathbb{Z}_k, d_i \in \mathbb{Z}_N$ are random values and set $d = \sum_{i=1}^{k} d_i$.
– Challenge: $\mathcal{A}$ arbitrarily selects two messages m_0, m_1 with the same length and two access structures T_0, T_1 that cannot be satisfied by any attribute set. Then $\mathcal{A}$ sends them to challenger. Challenger use access structure T_b to encrypt the message to generate m_b or a random value r. Subsequently, challenger arbitrarily chooses s and do the following operations to produce component of ciphertext. $\tilde{C}^* = m_b T, C^*_0 = g_p^s Y_2 R'_0$. The remain ciphertext components are obtained by the following calculations. When $(L_i \in L_\alpha) \wedge (v_{i,t_i} \notin L_\alpha)$, let $C^*_{i,t_i} = g_p^{a_{i,t_i} s_{i,t_i}} R'_{i,t_i}$ where $s_{i,t_i} \in \mathbb{Z}_N$ *and* $s_{i,t_i} \neq s_\alpha$,

otherwise, $C^*_{i,t_i} = g_p^{a_{i,t_i} s_\alpha} R'_{i,t_i}$. Lastly, the whole ciphertext $\langle \tilde{C}^*, C^*_0, C^*_{i,t_i} \rangle$ is dispatched to $\mathcal{A}$.
- Guess: Challenger outputs a guess marked d' of d.

When $T = e(g_p, g_p)^{\omega s}$, encrypted message m_b is a semi-functional ciphertext, otherwise, let $T = e(g_p, g_p)^{\omega s - x_0}$, the generated semi-functional ciphertext belongs to the random message $m_b e(g_p, g_p)^{\omega s - x_0}$. Therefore, If $\mathcal{A}$ has a non-negligible advantage $Adv_{\mathcal{A}}^{Game_{1,u}} - Adv_{\mathcal{A}}^{Game_{final}} = \varepsilon$, then challenger can also distinguish a random element between $e(g_p, g_p)^{\omega s}$ and $\mathbb{G}_T$ with non-negligible advantage.

Definition 5 *Assume that Assumption1,Assumption2 and Assumption3 hold, then the scheme described in this paper is secure.*

Proof If Assumption1,Assumption2 and Assumption3 hold, $Game_{real}$ and $Game_{finla}$ are indistinguishable, furthermore, $\mathcal{A}$ has a negligible advantage $Adv_{\mathcal{A}}^{Game_{final}}$ in $Game_{final}$, hence, $\mathcal{A}$ have a negligible advantage $Adv_{\mathcal{A}}^{Game_{real}}$ in $Game_{real}$. So, our scheme satisfies security under Chosen Ciphertext Attack(CCA).

5 Performance Analysis

In this section, we reveal the performance of our scheme and it is divided into two sections theoretical analysis and experimental simulation. In the former, we mainly adopt mathematical methods to analyze the efficiency of our scheme and the length of ciphertext, and compare it with other schemes. In order to verify the correctness of the theoretical analysis scheme, we conducted a simulation experiment in the latter.

5.1 Theoretical Analysis

Firstly, let us present the notations appearing in the following. N expresses the number of attributes in the whole system. $T_{\mathbb{G}}, T_{\mathbb{G}_T}$ indicate the cost time of operations of group $\mathbb{G}$ and group $\mathbb{G}_T$. Note that the operations only include exponentiation for convenience. $|*|$ is the bit-length of $*$.

Table1 indicates the properties between our scheme and other related scheme. ✓ means the scheme satisfies this property, conversely, × shows the schemes can not match this property. From Table1, we can get the conclusion that compared with other schemes, our scheme not only has more comprehensive features to improve efficient, testing and outsourcing specifically, but also can verify the correctness of results calculated by cloud servers. Meanwhile, due to the tree structure, our scheme has more expression ability.

5.2 Experimental Simulation

Table 1 PROPERTY COMPARISON TABLE

Scheme	Access Structure	Hidden Policy	Testing	Outsourcing	Verification	Security Model
Nishide	AND-gates	✓	×	×	×	selectively-CCA
Lai	AND-gates	✓	×	×	×	adaptively-CCA
Luan	Tree	×	×	×	×	selectively-CPA
Li	AND-gates	✓	✓	×	×	selectively-CPA
Xu	Tree	✓	×	×	×	selectively-CPA
Ours	Tree	✓	✓	✓	✓	adaptively-CCA

Table 2 Storage of Different Schemes

Scheme	Public Key	Secret Key	Ciphertext
Nishide	$\lvert\mathbb{G}_T\rvert + (2N+1)\lvert\mathbb{G}\rvert$	$(3n+1)\lvert\mathbb{G}\rvert$	$\lvert\mathbb{G}_T\rvert + (2N+1)\lvert\mathbb{G}\rvert$
Lai	$(N+2)\lvert\mathbb{G}\rvert + \lvert\mathbb{G}_T\rvert$	$(n+1)\lvert\mathbb{G}\rvert$	$(N+1)\lvert\mathbb{G}\rvert + \lvert\mathbb{G}_T\rvert$
Our	$2\lvert\mathbb{G}_T\rvert + (N+4)\lvert\mathbb{G}\rvert$	$(2+k)\lvert\mathbb{G}\rvert$	$(2+\lvert\alpha\rvert+\lvert\alpha\rvert N)\lvert\mathbb{G}\rvert + (1+\lvert\alpha\rvert)\lvert\mathbb{G}_T\rvert$

Table 3 Efficient of Different Schemes

Scheme	Encryption	Decryption		
		Transform_{out}		LocalDec
		Testing	PreDec	
Nishide	$(2N+1)\mathbb{G} + \mathbb{G}_T$	–	–	$(3n+1)P + (3n+1)\mathbb{G}_T$
Lai	$(2N+2)\mathbb{G} + \mathbb{G}_T$	–	–	$(n+1)P + (n+3)\mathbb{G}_T$
LiJiGuo	$(N+n)\mathbb{G} + \mathbb{G}_T$	$\mathbb{G}_T + 2P + 2(n-1)\mathbb{G}$	–	$(2n+1)P + (2n+1)\mathbb{G}_T$
Our	$(2N+5)\mathbb{G} + 2\mathbb{G}_T$	$(k\lvert\alpha\rvert + \lvert\alpha\rvert)P + (k\lvert\alpha\rvert + \lvert\alpha\rvert)\mathbb{G}_T$	$P + \mathbb{G}_T$	$2\mathbb{G}_T$