Outsource the Ciphertext Decryption of Inner Product Predicate Encryption Scheme Based on Prime Order Bilinear Map

Xingbing Fu[1], Xunyun Nie[1], and Fagen Li[2]

(Corresponding author: Xingbing Fu)

School of Information and Software Engineering, University of Electronic Science and Technology of China[1]

No. 4, North Jianshe Road, Chenghua District, Chengdu, Sichuan 610054, P.R. China

School of Computer Science and Engineering, University of Electronic Science and Technology of China[2]

No.2006, Xiyuan Avenue, West Hi-tech Zone, Chengdu 611731, P.R. China

(Email: fuxbuestc@126.com)

(Received Jan. 20, 2016; revised and accepted Mar 11 & Apr. 17, 2016)

Abstract

In the private index predicate encryption scheme, the ciphertext not only hides the message, but also hides the attributes. Predicate encryption scheme can enforce the fine grained access control over the encrypted data and perform selective search on the encrypted data. However, the main efficiency drawback of predicate encryption scheme is that the size of the ciphertext and the time required to decrypt it scale with the complexity of the predicate. In this work, we proposed a novel inner product predicate encryption scheme with verifiable outsourced ciphertext decryption based on prime order bilinear group, which significantly reduces the overhead of the data consumer. In the proposed scheme, the data consumer provides the cloud server with a transformation key with which the cloud server transforms the ciphertext associated with the attributes which satisfy the predicate associated with the private key into a simple and short ciphertext, and thus it significantly reduces the time for the data consumer to decrypt the ciphertext, whereas the cloud server does not know the underlying plaintext message for any data consumer; simultaneously, the data consumer can check whether the transformation done by the cloud server is correct to verify the correctness of the transformation.

Keywords: Decryption outsourcing, inner product predicate encryption, prime order bilinear map, RCCA security

1 Introduction

Predicate encryption which can enforce fine grained access control over the encrypted data and perform the selective search on the encrypted data is a novel public key encryption paradigm. In predicate encryption scheme, the private key corresponds to the predicate, and the ciphertext is associated with the attribute set. The private key $PriKey_f$ corresponding to the predicate f can decrypt the ciphertext associated with the attribute A, if and only if $f(A) = 1$.

The traditional public key encryption scheme is coarse-grained: the sender encrypt the message m using the public key PK, only if the owner of private key associated with the public key PK can decrypt the ciphertext to recover the plaintext message m. This scheme is suitable for point-to-point communication, and the encrypted data are sent to the recipient whom the sender is known to in advance. Recently, with the advent of cloud computing, the data owner may want to store their sensitive data in the cloud server such that the sensitive data are accessible to the data consumers anytime and anywhere. However, the cloud servers are honest and curious: on one hand, it performs the various services for data owners and data consumers according to the protocol requirements; on the other hand, it may sell the sensitive data belonging to the data owner to his competitors to obtain the economic benefits. Furthermore, the adversary may want to obtain the sensitive data to do damage to the data owner. Therefore, the data owner encrypts the data to preserve the privacy of data, then he outsources the encrypted data to the cloud server to store such that the authorized consumers can access them. If the traditional encryption scheme is used, the data consumer is not able to search the encrypted data stored in the cloud server. In fact, the data consumer needs to download the encrypted data first, decrypts them, and searches them. When the big complex data are processed in the cloud computing environment, this method will bring about the huge processing overhead and communication overhead.

Boneh et al. [2] first investigated this problem, and they first introduced the encryption scheme that supports the equality tests. In this scheme, the owner of the public key can calculate the trapdoor information K_m for any message m, and the K_m allows for the server storing the data to test whether a given ciphertext encrypts message m on the condition that any additional information is not obtained. They used this scheme to encrypt the e-mails which were stored in the server such that the data consumer only downloaded the e-mail messages with a given subject without downloading the whole messages and decrypting them. Goyal et al. [11] proposed the key policy attribute based encryption scheme. In this scheme, the ciphertext is associated with the attribute sets, and the private key is associated with the predicate. The private key can decrypt the ciphertext if and only if the attribute set associated with the ciphertext satisfies the predicate associated with the key. Their scheme employs secret sharing scheme, hence, their scheme is expressive. Due to expressiveness of attribute based encryption schemes, they are suitable for many cloud computing and cloud storage applications. Lee, Chung, and Hwang [15] surveyed attribute based encryption scheme of access control in cloud environments. Chung, Liu, and Hwang [7] surveyed attribute-based proxy re-encryption scheme in cloud environments. However, these schemes only achieve payload hiding, that is to say, in these schemes, only the plaintext privacy is guaranteed, whereas the attribute set associated with the ciphertext is public, and not hidden, i.e., the privacy of attribute is not guaranteed. In some highly sensitive environments, not only the privacy of message is guaranteed, but also the privacy of attribute is guaranteed. That is to say, the attribute hiding should be guaranteed. For example, in the personal health record, oncologist attribute suggests that someone or person associated with him has tumour. To meet this requirement, Boneh and Waters [3] proposed hidden vector encryption scheme. Their scheme supports conjunctions, subsets and range query. However, their scheme does not support delegation. Shi and Waters [18] proposed hierarchical hidden vector encryption scheme that supports delegation. However, both schemes do not support disjunction query. In order to support disjunction query, polynomial equation, and inner product calculation, Katz et al. proposed inner product predicate encryption scheme, KSW scheme [14]. However, these three schemes are based on composite order bilinear group.Since group operations and especially bilinear map are prohibitively slow on composite order elliptic curves: a Tate pairing on a composite order elliptic curve whose group order is 1024 bits is roughly 50 times slower than the same pairing on a comparable prime order elliptic curve [17]. With the security levels increasing, this performance gap will become worse. To obtain the same security level, in contrast with composite order elliptic curve group, prime order elliptic curve group requires less order. In order to solve the inefficiency on composite order elliptic curve group, Iovino and Persiano [13] proposed hidden vector encryption scheme based on prime order bilinear group. However, their scheme only supports conjunction calculation, and does not support disjunction calculation. In order to improve efficiency, Freeman [8] proposed inner product predicate encryption scheme, and their scheme obtains the same functionality as KSW scheme [14], whereas their scheme is implemented on prime order elliptic curve group. Both KSW scheme [14] and Freeman scheme [8] are only chosen plaintext secure against attack (**CPA**), and are not chosen ciphertext secure against attack (**CCA**), even not replayable chosen ciphertext secure against attack (**RCCA**). In this work, we employ Freeman scheme [8] as a building block, and attempt to improve security and efficiency.

However, the current predicate encryption schemes, whether they are based on composite order bilinear group or on prime order bilinear group, have the common drawbacks: the ciphertext size and time to decrypt it scale with the size of predicate. When the data consumer employs the resource-restrained device to manage and query the private data stored on his device, the increasing requirement is to outsource calculation to the cloud server where you pay as you use.

How can securely outsource the decryption of ciphertext? A naive method is that the data consumer sends his private key $PriKey$ to the cloud server, the cloud server decrypts the ciphertext which is requested by the data consumer, and then it sends the data decrypted to the data consumer, which requires that outsourcing service is fully trusted. In fact, the cloud server provider can employ the data consumer's private key $PriKey$ to decrypt the ciphertext which will be sent to the other data consumers to recover the plaintext message to obtain the economic benefits. Furthermore, the data decrypted are transmitted in the clear; once the attacker captures the data, the confidentiality of data is compromised. The second method is the outsourcing techniques [6, 9] based on fully homomorphism encryption [10]. These schemes outsource the computation to the cloud server, such that not only the input privacy is guaranteed, but also the decryption keys and messages privacy are guaranteed. However, fully homomorphism encryption schemes and those schemes based on fully homomorphism encryption schemes are not suitable for outsourcing due to inefficiency. If secure pairing outsourcing techniques are employed, then pairing calculations are outsourced to servers. However, the scheme [5] requires the client to calculate multiple exponentiations in the target group where every outsourced pairing is performed. These exponentiations are too expensive and the overhead of the client scales with the predicate size. Furthermore, every pairing operation in the original scheme has four pairings which will be done by the proxy such that the client's bandwidth requirements increase as well. Given the aforementioned drawbacks, our scheme outsources the decryption of inner product predicate encryption ciphertext to the cloud server to perform, and imposes the minimal overhead on the data consumer.

In this work, we proposed a novel scheme that securely and efficiently outsources the decryption of the inner product predicate encryption ciphertext. The proposed scheme significantly reduces the overhead of the data consumers. In this scheme, the data consumer provides the cloud server with a transformation key such that the cloud server can transform the inner product predicate encryption ciphertext into a simple and short ciphertext without the cloud server knowledging the data consumer's plaintext data, and simultaneously the transformation done by the cloud server can be verified to guarantee that the transformation done by the cloud server is correct. The proposed scheme significantly saves the client bandwidth and the local calculation time: the size of the ciphertext transformed is much smaller than the size of the original ciphertext, and the time to decrypt the transformed ciphertext is much less than the time to decrypt the original ciphertext. Therefore, the resource-restrained device consumes less power. Our scheme is secure against the malicious cloud server as well. Furthermore, our scheme achieves **RCCA** security.

The remainders of our paper are organized as follows: We discuss related work in Section 2. We introduce preliminaries in Section 3. We present the syntax, and security model of the proposed scheme in Section 4. We present the architecture of the proposed scheme in Section 5. We present the scheme construction in Section 6. We give the security proof of the proposed scheme in Section 7. The performance of the proposed scheme is evaluated in Section 8. We draw the conclusion in Section 9.

2 Related Work

In this section, we give the related work as follows: interactive verifiable calculation, bilinear pairing delegation, and proxy re-encryption.

Interactive Verifiable Computation. Interactive verifiable computation [6, 9] enables the resource-restrained devices with weak computation to outsource the computation on the functions to a server, the server returns the result of computations to the client, and gives to it the non-interactive proof that the computation on the functions is correct. Since these schemes [6, 9] outsource the computation to the cloud server, and protect the privacy of input data. However, these schemes are based on fully homomorphism encryption. The overheads of these schemes are so large that they cannot be applied to the cloud computing system. Parno et al. [16] proposed the verifiable computation from attribute based encryption. Their scheme obtains the public delegation and public verification. They proposed the multi-function verifiable computation scheme as well, and this scheme is based on attribute based encryption scheme with outsourced decryption ciphertext due to Green et al. [12]. However, these schemes are focused on the delegation of general functions, not on the efficiency of the problems. Furthermore, these schemes are based on attribute based encryption which only achieve payload hiding, that is to say, the privacy of message is guaranteed, whereas the attributes are public. In some highly sensitive settings, the attributes are required to be hidden. Predicate encryption schemes obtain the attribute hiding.

Pairing Outsourcing. Chevallier-Mames et al. [5] proposed the pairing outsourcing which enables a client to outsource the pairing computation to another entity. However, this scheme still requires the client to calculate multiple exponentiations in the target group where every outsourced pairing is performed. If their scheme is employed to outsource the decryption of predicate encryption ciphertext, the overhead of the client will be proportional to the size of the predicate.

Proxy Re-Encryption. The proposed scheme shows that the client allows the cloud server to transform the predicate encryption ciphertext on m into a simple and short ciphertext, whereas the cloud server acting as the proxy does not learn the underlying plaintext message m. This method is similar to proxy re-encryption [1], in which a semi-trusted proxy is given a proxy key which allows it to transform a ciphertext under one public key into a ciphertext of the same message under another public key without learning the underlying plaintext message. However, in the traditional proxy re-encryption scheme, the correctness of the transformation done by the proxy is not guaranteed. Since the proxy can replace the encryption of m under the delegator's public key with the encryption of another message m' under the delegator's public key, and then employs the proxy keys to transform the latter into an encryption of m' under the delegatee's public key, which brings about reducing the significant computation to perform the other computation services to obtain the economic advantages.

3 Preliminaries

3.1 Bilinear Group Generator

A bilinear group generator is an algorithm $\mathcal{G}$ [8] which takes in a security parameter κ, and outputs five abelian groups G, G_1, U, U_1 and G_T, where $G_1 \subset G$, $U_1 \subset U$. In each group, efficient group operation and random samples are performed. The algorithm outputs efficient computable map $e : G \times U \to G_T$ which has the following properties:

Bilinearity: For any $g_1, g_2 \in G$, $u_1, u_2 \in U$, $e(g_1g_2, u_1u_2) = e(g_1, u_1)e(g_1, u_2)e(g_2, u_1)e(g_2, u_2)$;

Non-degenerate: For all $g \in G$, for all $u \in U$, if $e(g, u) = 1$, then $g = 1$.

3.2 Cancelling Pairing

In the proposed scheme, bilinear group generator $\mathcal{G}$ is from the prime order bilinear group generator $\mathcal{P}$, and the pairing e on the product groups is defined as any nontrivial linear combination of the componentwise pairings on the underlying prime order group. In the scheme based on composite order bilinear group [14], if the two group elements g, u have the co-prime order, then $e(g, u) = 1$, which implies that the two subgroups generated by g, u can encode different types of information, and these two components will remain distinct after the pairing operation.

Definition 1. *Let $\mathcal{G}$ be a bilinear group generator [8]. If $\mathcal{G}$ outputs $G_1, \cdots, G_q \subset G$ and $U_1, \cdots, U_q \subset U$, such that:*

1) $G \cong G_1 \times \cdots \times G_q$ and $U \cong U_1 \times \cdots \times U_q$;

2) Whenever $g_i \in G_i$ and $u_j \in U_j$, $i \neq j$, $e(g_i, u_j) = 1$, then $\mathcal{G}$ is q cancelling.

4 Inner Product Predicate Encryption Scheme with Outsourced Ciphertext Decryption Based on Prime Order Bilinear Group

In this section, we give the syntax and security model of inner product predicate encryption scheme with outsourced ciphertext decryption based on prime order bilinear group.

4.1 Syntax

Let the attribute set be S, and the class of predicate $\mathbb{F}$. Inner product predicate encryption scheme with outsourced ciphertext decryption based on prime order bilinear group comprises the six algorithms as follows:

$Setup(1^k, l) \to (PK, MS)$**:** The **Setup** algorithm is run by the trusted authority. This algorithm takes in a security parameter 1^κ and a positive integer l, which is attribute and predicate vector length, and outputs the public key PK and the master key MS.

$Encrypt(PK, m, A) \to CT$**:** The **Encrypt** algorithm is run by the data owner. It takes in the public key PK, message m and attribute A, and outputs the ciphertext $CT = Encrypt(PK, m, A)$.

$PriKeyGen(PK, MS, f) \to PriKey_f$**:** The private key generation algorithm **PriKeyGen** is run by the trusted authority. It takes in PK, the master key MS and predicate $f \in F \subseteq \mathbb{F}$, and outputs the private key $PriKey_f$.

$OutKeyGen(PK, PriKey_f) \to (TK_f, SK_f)$**:** The Outsourced Private Key Generation algorithm **OutKeyGen** is run by the data consumer. It takes in the public key PK and the private key $PriKey_f$, and outputs the transformation key TK_f and the private keys SK_f. The transformation key TK_f is public, and the data consumer sends the transformation key TK_f to the cloud server which employs TK_f to partially decrypt the original ciphertext. The data consumer keeps the secret key SK_f private and employs SK_f to decrypt the ciphertext that is partially decrypted. It is the data consumer, not the trusted authority that runs the Outsourced Private Key Generation algorithm **OutKeyGen**, which avoids the costly online request for the trusted authority.

$Transform(PK, TK_f, CT) \to \widetilde{CT}$**:** The ciphertext **Transform** algorithm is run by the cloud server acting as the proxy. It takes in the public key PK, the transformation key TK_f for the predicate f and the original ciphertext CT, and outputs the partially decrypted ciphertexts $\widetilde{CT}$ if $f(A) = 1$, else the error symbol $\perp$.

$OutDecrypt(PK, SK_f, \widetilde{CT}) \to \{m, \perp\}$**:** The Outsourced Decryption **OutDecrypt** algorithm is run by the data consumer. It takes in the secret key SK_f and the partially decrypted ciphertexts $\widetilde{CT}$; if $f(A) = 1$, then it outputs the message m, else the error symbol $\perp$.

Correctness. For any security parameters κ, any public keys PK and master secrets MS generated by **Setup**, any $f \in \mathbb{F}$, any private keys

$$\begin{aligned}
PriKey_f &\leftarrow \textbf{PriKeyGen}(PK, MS, f),\\
(TK_f, SK_f) &\leftarrow \textbf{OutKeyGen}(PK, PriKey_f),\\
CT &\leftarrow \textbf{Encrypt}(PK, m, A),\\
\widetilde{CT} &\leftarrow \textbf{Transform}(PK, TK_f, CT),
\end{aligned}$$

for all attributes $\overrightarrow{y} \in S$, the following propositions hold:

1) If $f(\overrightarrow{y}) = 1$, then **OutDecrypt**$(PK, SK_f,$ **Transform**$(PK, TK_f, Encrypt(PK, m, A)))$ $\to m$;

2) If $f(\overrightarrow{y}) = 0$, then **OutDecrypt**$(PK, SK_f,$ **Transform**$(PK, TK_f, Encrypt(PK, m, A)))$ $\to \perp$.

4.2 Security Model of Inner Product Predicate Encryption Scheme with Outsourced Ciphertext Decryption Based on Prime Order Bilinear Map

Since security against the adaptive chosen ciphertext attack (**CCA**) requires that any bit of the ciphertext should not be altered, which makes the requirement too strong, and the outsourced goal is to compress the ciphertext size, therefore, our scheme adopts the replayable adaptive chosen ciphertext attack (**RCCA**) security due to [4]. **RCCA** security allows the ciphertext to be altered provided that the underlying message is not changed in a meaningful way. The security model for inner product predicate encryption scheme with outsourced ciphertext decryption based on prime order bilinear map is described between a challenger and an attacker:

Initialization. The attacker declares the challenge attributes $\overrightarrow{y}, \overrightarrow{z} \in S$, and gives them to the challenger.

Setup. The challenger runs **Setup**$(1^k, l)$ to generate the public key PK and the master secret MS. The challenger defines the value N, and gives it and the public key PK to the attacker.

Query Phase 1. The challenger initializes an empty set T, and an empty set F. The attacker adaptively makes the following queries:

The private key query: On input the predicate f, the challenger runs $PriKey_f \leftarrow$ **PrikeyGen**(PK, MS, f), and sets $F = F \bigcup \{f\}$ with the restriction that $f.\overrightarrow{y} = 0$ holds if and only if $f.\overrightarrow{z} = 0$. It returns $PriKey_f$ to the attacker.

The transformation key query: On input the predicate f, the challenger searches $(f, PriKey_f, TK_f, SK_f)$ to see whether they lie in table T. If so, it returns the transformation key TK_f; else, it runs $PriKey_f \leftarrow$ **PriKeyGen**(PK, MS, f), $(TK_f, SK_f) \leftarrow$ **OutKeyGen**$(PK, PriKey_f)$, and stores $(f, PriKey_f, TK_f, SK_f)$ in table T. It returns the transformation key TK_f to the attacker.

The outsourced decryption query: On input the predicate f, the challenger searches the entry $(f, PriKey_f, TK_f, SK_f)$ to see whether they lies in the table T. If so, it runs $m \leftarrow$ **OutDecrypt**$(PK, SK_f, \widetilde{CT})$, and returns it to the attacker; if not, it returns $\perp$.

Challenge. The attacker submits the two message m_0 and m_1 of the equal length. If $f(\overrightarrow{y}) = f(\overrightarrow{z}) = 1$, then $m_0 = m_1$. The challenger picks a random fairly binary coin $\beta \in \{0, 1\}$. If $\beta = 0$, then it gives $CT =$ **Encrypt**$(PK, m_0, \overrightarrow{y})$ to the attacker; else it gives $CT =$ **Encrypt**$(PK, m_1, \overrightarrow{z})$ to the attacker.

Query Phase 2. The same as **Query Phase 1** with the restriction that the attacker cannot:

1) Trivially obtain a private key that decrypt the challenge ciphertext. That is to say, it cannot issue the query that the attributes associated with the ciphertext satisfy the predicate associated with the private key.

2) Trivially issue decryption query. The decryption query is the same as the **Query Phase 1**.

Guess. The attacker outputs a guess β' of β.

In this game, the advantage of the attacker is defined as

$$Adv_{\mathcal{A}} = |Pr\{\beta' = \beta\} - \frac{1}{2}|.$$

Definition 2. *If all probabilistic polynomial time attackers have the negligible advantage in the aforementioned **RCCA** security games, then inner product predicate encryption scheme with outsourced ciphertext decryption based on prime order bilinear group is **RCCA** secure.*

CPA Security. If decryption oracles in **Query Phase 1** and **Query Phase 2** are removed, then this scheme is secure against chosen plaintext attack **CPA**.

Selective Security. An inner product predicate encryption scheme with outsourced ciphertext decryption based on prime order bilinear group is selectively secure if an Init stage is added before *Setup*, in which the attacker declares the challenge attributes.

5 The Architecture of Inner Product Predicate Encryption Scheme with Outsourced Ciphertexts Decryption Based on Prime Order Bilinear Map

We proposed an inner product predicate encryption scheme with outsourced ciphertexts decryption based on prime order bilinear map whose architecture is illustrated in Figure 1. In the architecture, the cloud server stores the inner product predicate encryption (IPPE) ciphertext CT; when the client employed by the data consumer attempts to decrypt the ciphertext CT, if the ciphertext CT is found that it is not partially decrypted, then it sends the ciphertext CT and the transformation key TK_f to the cloud server which employs the transformation key TK_f to run the **Transform** algorithm to output the partially decrypted ciphertext $\widetilde{CT}$. The cloud server sends $\widetilde{CT}$ to the client which employs the secret key SK_f to decrypt the partially decrypted ciphertext $\widetilde{CT}$ to obtain the plaintext message.

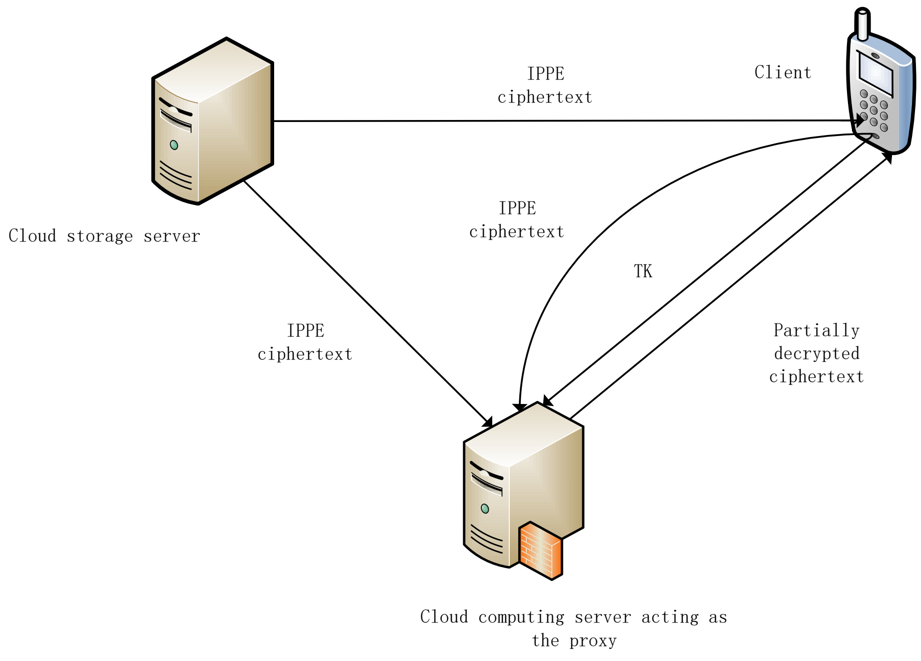

Figure 1: Architecture of inner product predicate encryption scheme with outsourced ciphertexts decryption

6 The Construction of Outsourced Ciphertext Decryption of Inner Product Predicate Encryption Scheme Based on Prime Order Bilinear Map

Our scheme employs the asymmetric bilinear pairing generator to improve the efficiency.

Setup$(1^k, l)$**:** The **Setup** algorithm is run by the trusted authority. It takes in a security parameter 1^κ and a positive integer l that is attribute and predicate vector length,$\mathcal{G}(\kappa)$ is a 3-cancelling bilinear group generator, and outputs the groups G_i and U_i which have the prime order exponents p_i respectively, where $i = 1, \cdots, 3$. Define the hash functions as follows: $H_1 : G_T \times \{0,1\}^n \to \mathbb{Z}_{p_1}$, $H_2 : G_T \to \{0,1\}^n$(n is the length of message bit), and $H_3 : \{0,1\}^n \to \mathbb{Z}_{p_1}$. Perform the following:

Step 1: Calculate $(G, G_1, G_2, G_3, U, U_1, U_2, U_3, G_T) \xleftarrow{\$} \mathcal{G}(\kappa)$.

Step 2: Pick $g_i \xleftarrow{\$} G_i$ and $u_i \xleftarrow{\$} U_i$.

Step 3: Pick $\mu \xleftarrow{\$} \mathbb{Z}_{p_1}$ and $t_0 \xleftarrow{\$} \mathbb{Z}_{p_3}$.

Step 4: For $j = 1, \cdots, l$, pick $\eta_{1,j}, \eta_{2,j} \xleftarrow{\$} \mathbb{Z}_{p_1}$ and $\sigma_{1,j}, \sigma_{2,j} \xleftarrow{\$} \mathbb{Z}_{p_3}$.

Step 5: Output the public parameters $(g_1, g_3, W = g_2 g_3^{t_0}, e(g_1, u_1)^\mu, \{B_{1,j} = g_1^{\eta_{1,j}} g_3^{\sigma_{1,j}}, B_{2,j} = g_1^{\eta_{2,j}} g_3^{\sigma_{2,j}}\}_{j=1}^l)$.

The master secret $MS = (u_1,\ u_2,\ u_3,\ u_1^{-\mu},\ \{\eta_{1,j},\ \eta_{2,j}\}_{j=1}^l)$.

Encrypt$(PK, m, \overrightarrow{y})$**:** Let $\overrightarrow{y} = (y_1, \cdots, y_l) \in \mathbb{Z}_N^l$ is an attribute vector, $m \in \{0,1\}^n$ is a message whose length is n. The **Encrypt** algorithm picks a random $\gamma \in G_T$, calculates $s = H_1(\gamma, m)$, $d = H_2(\gamma)$ and $\tau = H_3(d)$, and picks $\delta, \theta \xleftarrow{\$} \mathbb{Z}_N$ and random values $\xi_{1,j}, \xi_{2,j} \xleftarrow{\$} \mathbb{Z}_N$, where $j = 1, \cdots, l$. The ciphertext CT is published as:

$$CT = (E', E_b, \widehat{E}, \tau, \{E_{1,j}, E_{2,j}\}_{j=1}^l).$$

Here,

$$\begin{cases} E' &= \gamma e(g_1, u_1)^{\mu s}, \\ E_b &= g_1^s, \\ \widehat{E} &= m \oplus d, \\ \tau &= H_3(d), \\ E_{1,j} &= B_{1,j}^s W^{\delta y_j} g_3^{\xi_{1,j}}, \\ E_{2,j} &= B_{2,j}^s W^{\theta y_j} g_3^{\xi_{2,j}} \end{cases}$$

PriKeyGen(PK, MS, f)**:** Let $f = (f_1, \cdots, f_l) \in \mathbb{Z}_N^l$ is the predicate vector. The **PriKeyGen** algorithm picks $t_{1,j}, t_{2,j} \in \mathbb{Z}_N$, where $j = 1, \cdots, l$, and $\omega_1, \omega_2, \psi_1, \psi_2 \in \mathbb{Z}_N$. It outputs the private key corresponding to f:

$$PriKey_f = (K_b, \{K_{1,j}, K_{2,j}\}_{j=1}^l).$$

Here,

$$\begin{cases} K_b &= u_1^{-\mu} u_2^{\psi_1} u_3^{\psi_2} \prod_{j=1}^{l} u_1^{-t_{1,j}\eta_{1,j}-t_{2,j}\eta_{2,j}} \\ K_{1,j} &= u_1^{t_{1,j}} u_2^{\omega_1 f_j} \\ K_{2,j} &= u_1^{t_{2,j}} u_2^{\omega_2 f_j} \end{cases}$$

$OutKeyGen(PK, PriKey_f)$: The Outsourced Private Key Generation **OutKeyGen** algorithm runs **PriKeyGen**(PK, MS, f) algorithm to obtain $\widetilde{PriKey_f} = (\widetilde{K_b} = u_1^{-\mu} u_2^{\psi_1} u_3^{\psi_2} \prod_{j=1}^{l} u_1^{-t_{1,j}\eta_{1,j}-t_{2,j}\eta_{2,j}}, \{\widetilde{K_{1,j}} = u_1^{t_{1,j}} u_2^{\omega_1 f_j}, \widetilde{K_{2,j}} = u_1^{t_{2,j}} u_2^{\omega_2 f_j}\}_{j=1}^{l})$. It picks a random $x \in \mathbb{Z}_{p_1}^*$, and it sets the transformation key TK_f as:

$$\begin{aligned} TK_f &= (PK, OK_b, OK_{1,j}, OK_{2,j}). \\ OK_b &= \widetilde{K_b}^{\frac{1}{x}}, \\ OK_{1,j} &= \widetilde{K_{1,j}}^{\frac{1}{x}}, \\ OK_{2,j} &= \widetilde{K_{2,j}}^{\frac{1}{x}}. \end{aligned}$$

The private key $DecryptKey$ is $(x, TK_f) = (SK_f, TK_f)$.

$Transform(PK, TK_f, CT)$**:** The **Transform** algorithm takes in TK_f associated with the predicate f and the ciphertext CT associated with the attributes. If the attributes $\overrightarrow{y}$ associated with the ciphertext do not satisfy the predicate f, then the **Transform** algorithm returns $\perp$, else it calculates:

$$\begin{aligned} & e(E_b, OK_b) \prod_{j=1}^{l} e(E_{1,j}, OK_{1,j}) e(E_{2,j}, OK_{2,j}) \\ =\; & e(g_1^s, (u_1^{-\mu} u_2^{\psi_1} u_3^{\psi_2} \prod_{j=1}^{l} u_1^{-t_{1,j}\eta_{1,j}-t_{2,j}\eta_{2,j}})^{\frac{1}{x}}) \\ & \prod_{j=1}^{l} e(B_{1,j}^s W^{\delta y_j} g_3^{\xi_{1,j}}, (u_1^{t_{1,j}} u_2^{\omega_1 f_j})^{\frac{1}{x}}) \\ & e(B_{2,j}^s W^{\theta y_j} g_3^{\xi_{2,j}}, (u_1^{t_{2,j}} u_2^{\omega_2 f_j})^{\frac{1}{x}}) \\ =\; & e(g_1^s, u_1^{-\mu})^{\frac{1}{x}} e(g_1^s, (\prod_{j=1}^{l} u_1^{-t_{1,j}\eta_{1,j}-t_{2,j}\eta_{2,j}})^{\frac{1}{x}}) \\ & \prod_{j=1}^{l} e(g_1^{s\eta_{1,j}} g_2^{\delta y_j}, (u_1^{t_{1,j}} u_2^{\omega_1 f_j})^{\frac{1}{x}}) \\ & e(g_1^{s\eta_{2,j}} g_2^{\theta y_j}, (u_1^{t_{2,j}} u_2^{\omega_2 f_j})^{\frac{1}{x}}) \\ =\; & e(g_1^s, u_1^{-\mu})^{\frac{1}{x}} e(g_1^s, (\prod_{j=1}^{l} u_1^{-t_{1,j}\eta_{1,j}-t_{2,j}\eta_{2,j}})^{\frac{1}{x}}) \\ & \prod_{j=1}^{l} e(g_1^{s\eta_{1,j}} g_2^{\delta y_j}, (u_1^{t_{1,j}} u_2^{\omega_1 f_j})^{\frac{1}{x}}) \\ & e(g_1^{s\eta_{2,j}} g_2^{\theta y_j}, (u_1^{t_{2,j}} u_2^{\omega_2 f_j})^{\frac{1}{x}}) \end{aligned}$$

$$\begin{aligned} =\; & e(g_1^s, u_1^{-\mu})^{\frac{1}{x}} e(g_1^s, (\prod_{j=1}^{l} u_1^{-t_{1,j}\eta_{1,j}-t_{2,j}\eta_{2,j}})^{\frac{1}{x}}) \\ & e(g_1^s, (\prod_{j=1}^{l} u_1^{t_{1,j}\eta_{1,j}+t_{2,j}\eta_{2,j}})^{\frac{1}{x}}) \\ & \prod_{j=1}^{l} e(g_2^{\delta y_j}, (u_2^{\omega_1 f_j})^{\frac{1}{x}}) e(g_2^{\theta y_j}, (u_2^{\omega_2 f_j})^{\frac{1}{x}}) \\ =\; & e(g_1^s, u_1^{-\mu})^{\frac{1}{x}} \prod_{j=1}^{l} e(g_2^{\delta y_j}, (u_2^{\omega_1 f_j})^{\frac{1}{x}}) e(g_2^{\theta y_j}, (u_2^{\omega_2 f_j})^{\frac{1}{x}}) \\ =\; & e(g_1, u_1)^{\frac{-\mu s}{x}} \prod_{j=1}^{l} e(g_2, u_2)^{\frac{1}{x}\delta y_j \omega_1 f_j} e(g_2, u_2)^{\frac{1}{x}\theta y_j \omega_2 f_j}) \\ =\; & e(g_1, u_1)^{\frac{-\mu s}{x}} \prod_{j=1}^{l} e(g_2, u_2)^{\frac{1}{x}\delta \omega_1 y_j f_j} e(g_2, u_2)^{\frac{1}{x}\theta \omega_2 y_j f_j}) \\ =\; & e(g_1, u_1)^{\frac{-\mu s}{x}} \prod_{j=1}^{l} e(g_2, u_2)^{\frac{1}{x}(\delta\omega_1+\theta\omega_2)(y_j f_j)} \\ =\; & e(g_1, u_1)^{\frac{-\mu s}{x}} e(g_2, u_2)^{\frac{1}{x}(\delta\omega_1+\theta\omega_2)\sum_{j=1}^{l} <y_j, f_j>} \\ =\; & e(g_1, u_1)^{\frac{-\mu s}{x}} \text{ (If } <\overrightarrow{y}, f>= 0). \end{aligned}$$

If $<\overrightarrow{y}, f>= 0$, then the **Transform** algorithm outputs the result as $= e(g_1, u_1)^{\frac{-\mu s}{x}}$; else it returns the error symbol $\perp$.

The **Transform** algorithm outputs the partially decrypted ciphertext:

$$\begin{aligned} \widetilde{CT} &= (E', \widehat{E}, \tau, E_e). \\ \widehat{E} &= m \oplus d, \\ \tau &= H_3(d), \\ E_e &= e(g_1, u_1)^{\frac{-\mu s}{x}}. \end{aligned}$$

OutDecrypt$(PK,\ DecryptKey,\ \widetilde{CT}) \rightarrow \{m,\ \perp\}$. The Outsourced Decryption **OutDecrypt** algorithm takes in the public key PK, the private key $DecryptKey = (x, TK_f)$ and the partially decrypted ciphertext $\widetilde{CT}$. If the ciphertext is not partially decrypted, then **OutDecrypt** algorithm first runs the **Transform**(PK, TK_f, CT) algorithm. If the **Transform**(PK, TK_f, CT) algorithm outputs $\perp$, then **OutDecrypt**$(PK, DecryptKey, \widetilde{CT})$ algorithm outputs $\perp$, else it takes in $\widetilde{CT}$ and calculates

$$\begin{aligned} \gamma &= E' E_e^x, \\ d &= H_2(\gamma), \\ m &= \widehat{E} \oplus H_2(\gamma), \\ \tau' &= H_3(d). \end{aligned}$$

This algorithm checks that $\tau' \overset{?}{=} \tau$. If so, it will show that the transformation done by the server is correct. It outputs the message m.

If the ciphertext is partially decrypted by the cloud server, then **OutDecrypt** algorithm requires one exponentiation, one hash operation and XOR operation to obtain the message m, and no pairing operation. Since the malicious cloud server cannot obtain the message, our scheme is secure against it. Through the hash function H_3, the data consumer can decide whether the transformation done by the cloud server is correct.

7 Proof of Security

Suppose there is a **PPT** attacker $\mathcal{A}$ that attacks the proposed scheme in the selective **RCCA** security model with the advantage ϵ. We build a simulator $\mathcal{B}$ which attacks the [8] scheme in the selective **CPA** security model with the advantage ϵ. Freeman scheme [8] is proven secure under the two assumptions.

Initialization. The simulator $\mathcal{B}$ runs the attacker $\mathcal{A}$ which declares the challenge attributes $(\overrightarrow{y}^*, \overrightarrow{z}^*)$ that the simulator $\mathcal{B}$ sends to the Freeman scheme [8] challenger as the challenge attributes on which it wished to be challenged.

Setup. The simulator $\mathcal{B}$ obtains the Freeman [8] public parameters which are sent to the attacker $\mathcal{A}$ as the public parameters.

Query Phase 1. The simulator $\mathcal{B}$ initializes empty tables T, T_1, T_2, T_3 and an empty set F. The adversary's queries are answered by the simulator $\mathcal{B}$ as follows.

Random Oracle Hash $H_1 : G_T \times \{0,1\}^n \to \mathbb{Z}_{p_1}$**:** If there is an entry (γ, m, s) in the table T_1, then it returns s, else, it picks a random value $s \in \mathbb{Z}_p$, records (γ, m, s) in table T_1, and returns s.

Random Oracle Hash $H_2 : G_T \to \{0,1\}^n$**:** If there is an entry (γ, d) in table T_2, then it returns d, else it picks a random value $d \in \{0,1\}^n$, records (γ, d) in table T_2, and returns d.

Random Oracle Hash $H_3 : \{0,1\}^n \to \mathbb{Z}_{p_1}$**:** If there is an entry (τ, d) in table T_3, then it returns τ, else it picks a random value $d \in \{0,1\}^n$, records (τ, d) in table T_3, and returns τ.

The simulator $\mathcal{B}$ proceeds as follows: If the challenge attributes satisfy the predicate, the transformation key is constructed as follows: Call the private key generation algorithm of the Freeman scheme [8] to obtain the private key associated with the predicate f as $(\widetilde{K_b}, \widetilde{K_{1,j}}, \widetilde{K_{2,j}})$. The algorithm picks a random value $x \in \mathbb{Z}_{p_1}^*$, sets the transformation key as $TK_f = (PK, OK_b = \widetilde{K_b}^{\frac{1}{x}}, OK_{1,j} = \widetilde{K_{1,j}}^{\frac{1}{x}}, OK_{2,j} = \widetilde{K_{2,j}}^{\frac{1}{x}})$, stores it in table T, and returns TK_f to the attacker, else it returns $\perp$.

The attacker cannot issue the private key query where the ciphertext attributes satisfy the predicate. If there is an entry $(f, PriKey_f, TK_f, SK_f)$ in table T, then the simulator $\mathcal{B}$ obtains $(f, PriKey_f, TK_f, SK_f)$, set $F = F\bigcup\{f\}$, and returns the private key $PriKey_f$ to the attacker, else it returns $\perp$.

Decryption Oracle: Assume that all ciphertext inputs to the oracle are the partially decrypted ciphertext. The simulator $\mathcal{B}$ and the attacker $\mathcal{A}$ have access to the transformation key TK_f, so they can execute the transformation operation.Let $CT = (E' = \gamma e(g_1, u_1)^{\mu s}, E_b = g_1^s, \widehat{E} = m \oplus d, \tau = H_3(d), \{E_{1,j} = B_{1,j}^s W^{\delta y_j} g_3^{\xi_{1,j}}, E_{2,j} = B_{2,j}^s W^{\theta y_j} g_3^{\xi_{2,j}}\}_{j=1}^l)$ be associated with the attribute vector $\overrightarrow{y}$. From table T, $(f, PriKey_f, TK_f, SK_f)$ is obtained. If no entry exists or the attributes associated with the ciphertext do not satisfy the predicate, then it returns $\perp$ to the attacker $\mathcal{A}$.

If the attributes associated with the ciphertext do satisfy the predicate, proceed as follows: Parse $PriKey_f = (SK_f, TK_f) = (x, TK_f)$, and calculate $\gamma = E' \backslash E_e^x$.

Test if $E' = \gamma e(g_1, u_1)^{\mu s}, \widehat{E} = m \oplus d$. If so, it outputs the message m, else it returns $\perp$ to the attacker.

Challenge. The attacker $\mathcal{A}$ submits the two messages m_0^*, m_1^* of equal length, and the simulator $\mathcal{B}$ proceeds as follows:

Step 1. The simulator $\mathcal{B}$ picks random messages $(\gamma_0, \gamma_1) \in G_T$, and passes them to the challenger in Freeman scheme [8] to obtain the ciphertext $CT = (E' = \gamma e(g_1, u_1)^{\mu s}, E_b = g_1^s, \widehat{E} = m \oplus d, \tau = H_3(d), \{E_{1,j} = B_{1,j}^s W^{\delta y_j} g_3^{\xi_{1,j}}, E_{2,j} = B_{2,j}^s W^{\theta y_j} g_3^{\xi_{2,j}}\}_{j=1}^l)$.

Step 2. The simulator $\mathcal{B}$ picks the random value $\widehat{E'} \in \{0,1\}^n$.

Step 3. The simulator $\mathcal{B}$ sends to the attacker $\mathcal{A}$ the challenger ciphertext $CT^* = (E', E_b, \widehat{E}', \tau = H_3(d), \{E_{1,j}, E_{2,j}\}_{j=1}^l)$.

Query Phase 2. The simulator $\mathcal{B}$ continues to answer the queries as **Query Phase 1**, except that if the response to the decryption query is m_0^*,m_1^*, then the simulator $\mathcal{B}$ responds with `test`.

Guess. The attacker $\mathcal{A}$ returns a bit β or abort, and the simulator $\mathcal{B}$ does not respond. The simulator $\mathcal{B}$ search through tables T_1 and T_2 to see if γ_0 or γ_1 appears as the first element of any entry. If both γ_0 and γ_1 or neither appear, the simulator $\mathcal{B}$ returns a random bit as a guess. If γ_β appears, then the simulator $\mathcal{B}$ returns β as its guess.

Table 1: Comparison of our scheme and freeman scheme

References	**Security Level**	**Original Ciphertext Size**	**Time to Decrypt the Original Ciphertext**	**Outsourced Ciphertext Size**	**Time to Decrypt the Outsourced Ciphertext**
Freeman Scheme [8]	CPA security	$\|G_T\| + \|G_1\| + \|G_1\|l\|G_2\|(l+1)\|G_3\|$	$\leq (1+2l)C_e$	Not Supported	Not Supported
Our Scheme	**RCCA** security	$\|G_T\| + \|G_1\| + \|G_1\|l\|G_2\|(l+1)\|G_3\| + n + 1H_2 + 1H_3$	$\leq (1+2l)C_e$	$\|G_T\| + n + 1H_2 + 1H_3$	$E_T + 1H_2 + 1H_3$

8 Performance Evaluation

As illustrated in Table 1 which depicts the outsourced ciphertext decryption of inner product predicate encryption scheme based on prime order bilinear group, where l denotes the attribute and predicate vector length, n is the length of message bits in **RCCA** scheme, aH_2, bH_3 denotes a times H_2 operation,b times H_3 operation, or the bit number of hash operation, respectively, and cC_e, $||$, and E_T denotes c times bilinear map, the cardinality of the set, and exponentiation. As seen from the Table 1, in contrast with Freeman scheme [8], in the proposed scheme, the data consumer requires $E_T + 1H_2 + 1H_3$ operation, outsourcing significantly reduces the time to decrypt for the data consumer, and compresses the ciphertext size, such that the overhead of the data consumer is significantly reduced. Furthermore, our scheme achieves security against **RCCA**, whereas Freeman scheme [8] is only secure against **CPA**.

9 Conclusion

In predicate encryption scheme, the ciphertext size and time to decrypt it scale with the complexity of the predicate. If predicate encryption scheme is employed in the resource constrained devices to achieve fine grained access control over the encrypted data, it will drain the battery. In this work, we propose inner product predicate encryption scheme based on prime order bilinear group that outsources decryption of ciphertext to the cloud server, which significantly reduces the ciphertext size and time to decrypt it, whereas the cloud server does not learn the underlying plaintext message. Therefore, outsourced decryption has the obvious advantages.

10 Acknowledgments

This work was supported by national science and technology support program of China (No.2014BAH11F02), the National Science Foundation of China (No. 61402376 and No.61373163) and the science and technology foundation of Sichuan Province of China (No. 2014GZ0109). The authors gratefully acknowledge Chaosheng Feng and the anonymous reviewers for their valuable comments.

References

[1] G. Ateniese, K. Fu, M. Green, and S. Hohenberger, "Improved proxy re-encryption schemes with applications to secure distributed storage," *ACM Transactions on Information and System Security*, vol. 9, no. 1, pp. 1–30, 2006.

[2] D. Boneh, G. D. Crescenzo, R. Ostrovsky, and G. Persiano, "Public key encryption with keyword search," in *Advances in Cryptology (EUROCRYPT'04)*, LNCS 3027, pp. 506–522, 2004.

[3] D. Boneh and B. Waters, "Conjunctive, subset, and range queries on encrypted data," in *Theory of Cryptography Conference (TCC'07)*, pp. 535–554, Springer, 2007.

[4] R. Canetti, H. Krawczyk, and J. B. Nielsen, "Relaxing chosen-ciphertext security," in *Advances in Cryptology (CRYPTO'03)*, pp. 565–582, Springer, 2003.

[5] B. Chevallier-Mames, J. Coron, N. McCullagh, D. Naccache, and M. Scott, "Secure delegation of elliptic-curve pairing," in *International Conference on Smart Card Research and Advanced Applications*, pp. 24–35, Springer, 2010.

[6] K. M. Chung, Y. Kalai, and S. P. Vadhan, "Improved delegation of computation using fully homomorphic encryption," in *Advances in Cryptology (CRYPTO'10)*, pp. 483–501, Springer, 2010.

[7] P. S. Chung, C. W. Liu, and M. S. Hwang, "A study of attribute-based proxy re-encryption scheme in cloud environments," *International Journal of Network Security*, vol. 16, no. 1, pp. 1–13, Jan. 2014.

[8] D. M. Freeman, "Converting pairing-based cryptosystems from composite order groups to prime-order groups," in *Advances in Cryptology (EUROCRYPT'10)*, pp. 44–61, Springer, 2010.

[9] R. Gennaro, C. Gentry, and B. Parno, "Non-interactive verifiable computing: Outsourcing computation to untrusted workers," in *Advances in Cryptology (CRYPTO'10)*, pp. 465–482, Springer, 2010.

[10] C. Gentry, "Fully homomorphic encryption using ideal lattices," in *Proceedings of the Forty-first Annual ACM Symposium on Theory of Computing (STOC'09)*, pp. 169–178, 2009.

[11] V. Goyal, O. Pandey, A. Sahai, and B. Waters, "Attribute-based encryption for fine-grained access control of encrypted data," in *Proceedings of the 13th ACM Conference on Computer and Communications Security (CCS'06)*, pp. 89–98, 2006.

[12] M. Green, S. Hohenberger, and B.Waters, "Outsourcing the decryption of abe ciphertexts," in *USENIX Security Symposium*, pp. 354–372, 2011.

[13] V. Iovino and G. Persiano, "Hidden-vector encryption with groups of prime order," in *Pairing-Based Cryptography (Pairing'08)*, LNCS 5209, pp. 75–88, Springer, 2008.

[14] J. Katz, A. Sahai, and B. Waters, "Predicate encryption supporting disjunction, polynomial equations, and inner products," in *Advances in Cryptology (EUROCRYPT'08)*, pp. 146–162, Springer, 2008.

[15] C. C. Lee, P. S. Chung, and M. S. Hwang, "A survey on attribute-based encryption schemes of access control in cloud environments," *International Journal of Network Security*, vol. 15, no. 4, pp. 231–240, July 2013.

[16] B. Parno, M. Raykova, and V. Vaikuntanathan, "How to delegate and verify in public: Verifiable computation from attribute-based encryption," in *Theory of Cryptography (TCC'12)*, LNCS 7194, pp. 422–439, Springer, 2012.

[17] M. Scott., *Personal Communication*, 2009.

[18] E. Shi and B. Waters, "Delegating capabilities in predicate encryption systems," in *International Colloquium on Automata, Languages, and Programming (ICALP'08)*, pp. 560–578, Springer, 2008.

Xingbing Fu is a lecturer, he received his M.S. degree from Southwest University in 2007. He is currently a PhD Candidate School of Information and Software Engineering, University of Electronic Science and Technology of China. His research interests are information security, cloud computing, and cryptography.

Xuyun Nie received the Ph.D. degree in Information security from Graduate university of Chinese Academy of Sciences, Beijing, China, in 2007. He is presently an Associate Professor at University of Electronic Science and Technology of China (UESTC). His research interests include cryptography and information security.

Fagen Li received the Ph.D. degree in Cryptography from Xidian University, Xi'an, P.R. China in 2007. His research interests include cryptography and network security.