# Efficient Privacy Preserving Predicate Encryption with Fine-grained Searchable Capability for Cloud Storage

Xu An Wang[1,3], Fatos Xhafa[2], Weiyi Cai[3], Jianfeng Ma[4], Fushan Wei[4,5]

[1] *School of Telecommunications Engineering, Xidian University, P. R. China*
[2] *Department of Computer Science, Technical University of Catalonia, Spain*
[3] *Engineering University of Chinese Armed Police Force, P. R. China*
[4] *School of Cyber Engineering, Xidian University, P. R. China*
[5] *The PLA Information Engineering University, P. R. China*
`wangxazjd@163.com, fatos@cs.upc.edu`

## Abstract

With the fast development in Cloud storage technologies and ever increasing use of Cloud data centres, data privacy has become a must. Indeed, Cloud data centres store each time more sensitive data such as personal data, organizational and enterprise data, transactional data, etc. A commonplace solution is data encryption, however, achieving privacy and confidentiality with flexible searchable capability is a challenging issue. In this article, we show how to construct an efficient predicate encryption with fine-grained searchable capability. Predicate Encryption (PE) can achieve more sophisticated and flexible functionality compared to traditional public key encryption. We propose an efficient predicate encryption scheme by utilizing the dual system encryption technique, which can also be proved to be *IND-AH-CPA* (indistinguishable under chosen plain-text attack for attribute-hiding) secure without random oracle. We also carefully analyze the relationship between predicate encryption and searchable encryption. To that end, we introduce a new notion of Public-Key Encryption with Fine-grained Keyword Search (PEFKS). Our results show that an IND-AH-CPA secure PE scheme can be used to construct an IND-PEFKS-CPA (indistinguishable under chosen plain-text attack for public-key encryption with fine-grained keyword search) secure PEFKS scheme. A new transformation of PE-to-PEFKS is also proposed and used to construct an efficient PEFKS scheme based on the transformation from the proposed PE scheme. Finally, we design a new framework for supporting privacy preserving predicate encryption with fine-grained searchable capability for Cloud storage. Compared to most prominent frameworks, our framework satisfies more features altogether and can serve as a basis for developing such frameworks for Cloud data centres.

*Keywords:* Predicate Encryption, Public-Key Encryption with Fine-grained Keyword Search, Data Privacy, Data Confidentiality, Cloud Storage, Cloud Data Search Framework.

## 1. Introduction

Cloud computing is penetrating each time more in all everyday activities of people, enterprises, banks, businesses, organizations and alike. Indeed, Cloud computing is a disruptive technology changing very fast the landscape of very large scale computing and embracing all kinds of human activity. This fast penetration is explained by the many features of Cloud platforms, offered through *everything as a service*, namely, software, hardware, applications, data storage, etc., at reasonable costs. This is especially relevant to Small and Medium Size (SMEs) enterprises allowing flexibility, agility through tailorable solutions to their needs and alleviating the burden of maintenance and development of their proper private Cloud. There is however one important obstacle for massive Cloud adoption: the control over data at the Cloud! By using Cloud solutions it is assumed that at the end data is controlled by the Cloud owners, despite advances in the definition of SLA (Service Level Agreement) to protect users' data. Notorious examples of such loose of data control are the data outsourcing not only from SMEs but even from big corporations, which prefer data outsourcing to reduce costs of proper data centres. The challenging issue is therefore how to balance the benefits of the Cloud computing with ensuring sensitive valuable data from not being searched, accessed, stolen or leaked by malicious entities. Unfortunately, to date there is less progress in data protection than other Cloud services.

A reasonable and yet practical way to address the data privacy issue at Cloud would be to first encrypt the data sets and then outsource them to the Cloud [1]. Assuming that this can be done by the entity that outsources the data, the data security and protection would then rely on the strength of the encryption algorithm. Although there is a significant progress in achieving stronger encryption algorithms, the development of such algorithms should go hand-by-hand with the ability of other algorithms to process the data. For instance, it should be possible to run data mining algorithms directly on the cipher-texts. In other words, the development of new generation encryption algorithms should not compromise the ability to process and analyze the data. To address such issues, the cryptographic community is devoting many efforts to develop more advanced cryptographic techniques, such as fully homomorphic encryption. Homomorphic encryption has a nice feature, namely, it can support meaningful operations on the cipher-texts just like on the plain-texts, such as "*addition*", "*multiplication*", "*xor*", "*comparison*" operations and even more complex computations are possible. The drawback however is that according to the state of the art, these algorithms are not very fast to deal with very large data sets and large data streams, yet they offer solutions to cope in practice with the data privacy issue at Cloud for many real life applications.

### *1.1. Motivation*

Given that it is desirable to preserve the ability of data processing, analysis and mining algorithms, a foremost requirement for the encrypted data would be to ensure a very basic operation of such algorithms, namely, an efficient and flexible searching on the cipher-texts. The problem that arises here is that when the data sets are encrypted by the data owner, the resulted cipher-texts will loose the original data sets' natural indexing structure and therefore hinders an efficient and secure search. One way to overcome this problem, and thus ensure efficient and secure search, is to use *hybrid encryption with keyword search* (hereafter denoted, HEKS), which consists of public key encryption with keyword search and the underlying symmetric encryption. However, this technique has two shortcomings:

1. First, several natural but complex searching queries cannot be supported. For example, let us suppose that Corporation A outsources its data sets in encrypted form to the Cloud, and later would like to execute a query search on its cipher-texts with the search pattern ((“*Finance Department Related File*” **OR** “*CEO Charlie Signed File*”) **AND** “*All Technique Supporting Department Employees' Agreement File*”). Unfortunately, the traditional hybrid encryption with keyword search cannot implement easily this kind of search.
2. Secondly, privacy preserving searching queries cannot be implemented either. Referring again to the above example, the Cloud search engine may know the search keywords “*Finance Department Related File*” or “*CEO Charlie Signed File*” or “*Technique Supporting Department Employees' Agreement File*”, however, in some situations these keywords are very sensitive and might not be known by the Cloud search engine. Corporation A needs therefore a mechanism to implement privacy preserving search, although attribute based encryption and attribute based keyword search can implement complex search patterns such as the above example, they cannot implement them in a privacy preserving manner.

In this paper, in order to overcome the above shortcomings, we propose an efficient predicate encryption (PE) and show how to use it to implement a public key encryption with fine-grained searchable capability mechanism (PEFKS). Additionally, by combing PEFKS and PE into an integral encryption system, we are able to design a privacy preserving framework supporting an efficient predicate encryption with fine-grained searchable capability.

### *1.2. Overview of PE and PEKS definition and concepts*

Let us briefly recall here some main definitions and concepts on PE and PEKS.

#### *Encryption (PE)*

Traditional public key encryption aims to achieve a one-to-one secure communication, data is encrypted under a particular established public key, say belonging to a person, and then the cipher-text can be decrypted by using the corresponding private key, whereby this person can only get the entire plain-text or nothing. This feature is sufficient when the structure of Internet is simple such as standard client-server applications of recent past, but for many emerging applications like secure transactional Cloud systems, it is in fact insufficient to satisfy the very demanding requirements for flexibility under complex applications. In 2008, Katz, Sahai and Waters [2] proposed the concept of Predicate Encryption (PE), which is a very flexible encryption scheme. Given in input the cipher-text

$CT_I$ and secret key $Key_f$, PE can evaluate more sophisticated and flexible functionality $F : Key_f \times CT_I \rightarrow \{0,1\}^*$. For predicate encryption, a predicate is embedded in a key and a cipher-text corresponds to an attribute vector. A cipher-text with attribute vector $I$ can be decrypted by the secret key $sk_f$ if and only if $f(I) = 1$. PE actually is a generalization of several recent new encryption paradigms, such as Identity-Based Encryption (IBE), Attribute-Based Encryption (ABE), Hidden-Vector Encryption (HVE), all of them aiming at extending fine-grained encryption capability. Attribute-hiding (AH), which is essential for predicate encryption, is stronger than payload-hiding. For attribute-hiding, the associated attribute as well as the plain-text will be hidden from a cipher-text, while for payload-hiding, only the cipher-text conceals the plain-text. Currently, propelled by Cloud computation and its security issues, there has been increased attention from security community to advance on the efficient outsource computation at the Cloud [3, 4, 5, 6]. These predicate encryption schemes can have broad application to secure Cloud storage for their properties of payload-hiding and attribute-hiding.

In order to obtain a fully secure PE, the dual system encryption technique introduced by Waters [7] can be useful. In a dual encryption system, there are normal keys, semi-normal keys, normal cipher-texts and semi-normal cipher-texts. We only use the semi-functional keys and semi-functional cipher-texts in games of the security proof, while the normal keys and normal cipher-texts are used in the real system. The security proof organized as a sequence of games which can be proved to be indistinguishable. For the first game, the keys and cipher-texts are assumed normal, while in the second game, the cipher-text is semi-normal and the keys remain normal. In subsequent games, the attacker can only get semi-functional keys one by one. When reaching to the final game, all the keys are semi-functional, and thus they are no useful for decrypting a semi-functional cipher-text, therefore it is relatively easy to prove the security.

*Public-Key Encryption with Keyword Search (PEKS)*

On the other hand, the concept of Public-Key Encryption with Keyword Search (PEKS) was first proposed by Boneh *et al.* [8]. To exemplify its definition, consider the following scenario: the email server can intelligently deliver one's emails according to some keywords attached to the emails. Therefore, some trapdoors for the keywords are generated by the user and then are sent to the server, thereafter these keywords in the emails can be tested by the server and in case the test outputs true, the corresponding email will be sent to the user. Abadalla *et al.* [9] proposed that the consistency and security properties must be satisfied by a practical PEKS. The first property states that only if the trapdoor and the cipher-text are matched, the decryption can be succeeded. The second property states that, the keywords will be hidden by the cipher-text unless the trapdoor is given. In [8] was proved that PEKS implying anonymous IBE, while in [9] was proved that a secure and consistent PEKS can be constructed by an anonymous IBE. Therefore, the anonymous IBE and PEKS were shown to have a tight connection.

### *1.3. Our Contribution*

In this paper, the connection between anonymous IBE and PEKS is further investigated. More precisely, we focus on the relationship between PE and PEKS schemes. Our results are as follows:

1. We present a fully secure predicate encryption for the class of inner-product predicates without random oracles. Compared to previous state of the art systems, our proposal has several improved properties. First, the full security of our construction can be proved based on simple assumptions by dual encryption system. Secondly, compared to Katz *et al.*'s scheme [2], the computational cost of our scheme is halved, thus being more efficient. As a matter of fact, for each attribute in the cipher-text and user's key, there is only one group element and for each attribute in the decryption algorithm, it only requires one pairing operation, resulting overall in a much more efficient scheme.
2. In previous searchable encryption schemes, only the presence of one keyword in the cipher-text can be tested by the server. Our Public-Key Encryption with Fine-grained Keyword Search (PEFKS) not only can test whether multiple keywords are present in the cipher-text, but also can evaluate the relations of the keywords, such as equal, disjunction/conjunction. One may argue that, by adding some relations of keywords, only from single keyword search such complex relations can be obtained, however this does not hold, since some unnecessary information can be derived by the server. We also discuss the consistency property and the formal security model of PEFKS through the challenge and the adversary's interactive game.
3. We prove that IND-AH-CPA secure predicate encryption has a tight connection with IND-PEFKS-CPA secure PEFKS and thereof propose a transformation from PE to PEFKS (PE-to-PEFKS), which is an efficient and practical solution for many Cloud applications. From our predicate encryption scheme, we are able to construct an efficient PEFKS scheme.
4. Finally, we present a privacy preserving framework for implementing efficient predicate encryption with fine-grained searchable capability and provide analysis of its security. To the best of our knowledge, this is the first framework simultaneously having privacy preserving, confidentiality and efficiently supporting fine-grained searchable properties.

### *1.4. Related Work*

In 2008, Katz, Sahai and Waters in [2] proposed the notion of predicate encryption as a generalization of IBE. In their predicate encryption, a vector $\bar{v} \in Z_p^n \backslash \{\bar{0}\}$ is associated with a predicate $f$ and attribute $\bar{x} \in Z_p^n \backslash \{\bar{0}\}$ is associated with the key. If $\bar{x}\bar{v} = 0$, then $f_{\bar{v}}(\bar{x}) = 1$, else $f_{\bar{v}}(\bar{x}) = 0$. Their construction achieved thus attribute-hiding property. However this scheme was not efficient and can only be proved to be selectively secure in the IND-AH-CPA game. Shi and Waters [10] defined delegation for predicate encryption, and proposed a new security model for delegation. They presented an efficient construction, which can support conjunctive queries. Likewise, their scheme can only achieve selective and CPA security. A fully secure (H)PE scheme for inner-product predicates was proposed by Lewko *et al.* [11] in the standard model based on the dual system methodology. But their scheme is only CPA secure and inefficient. Recently, Zhang *et al.* [26, 13, 14, 15, 16, 17] proposed functional encryption with various interesting properties, Attrapadung *et al.* [18] proposed interesting conversions among several types of predicate encryption and discussed their applications to ABE with various compactness tradeoffs.

Boneh *et al.* [8] first gave the concept of public-key encryption with keyword search (PEKS). Some constructions of PEKS have been reported and they also proved that a secure IBE can be implied by a PEKS scheme. It was left as an open problem how to construct PEKS from IBE. In [9], Abdalla *et al.* did further work on PEKS. They made two important contributions. First, computational, statistical and perfect consistency are formulated via an experiment involving an adversary and the scheme. Second, a transformation from an anonymous IBE to a secure PEKS that guaranteed consistency and security was shown. There have been many research work on single keyword search, which clearly do not cover the case, which arises in many practical situations, when the search pattern is made of multiple keywords. On the other hand, recently, there are reported several techniques to efficiently implement secure search on confidential Cloud storage. Li *et al.* [19] proposed a novel way to enable efficient fuzzy keyword search over encrypted data for Cloud storage. Recently, Cao *et al.* [20] described a privacy preserving multi-keyword text search protocol in the Cloud setting supporting similarly-based ranking. Zheng *et al.* [21] proposed the VABKS system, which can achieve verifiable attribute-based keyword search over outsourced encrypted data. In Infocom'14, an attribute-based keyword search scheme has been given by Sun *et al.* [22], in which the data owner's search delegation right can be protected. Recently an efficient verifiable conjunctive keyword search scheme was proposed by Sun *et al.* [23] for large dynamic encrypted Cloud data. However until now there is not much research work on how to implement preserving privacy encryption and fine-grained searching simultaneously.

### *1.5. Paper's organization*

The rest of the paper is organized as follows. In Section 2, we give some preliminaries. We present our proposal on PE and analyze its security in Section 3. In Section 4, we present PE-to-PEFKS transformation and construct a concrete PEFKS scheme. In Section 5, we present our privacy preserving framework for implementing efficient predicate encryption with fine-grained searchable capability and sketch the analysis on its security. Finally, we end the paper in Section 6 with some conclusions and an outlook for future work.

## 2. Preliminaries

To facilitate the reading of the paper, we give some preliminaries, basic definitions and terminology. We introduce the notion of predicate encryption, mainly focusing on the class of inner-product predicates and further in this section we give the definition of public-key encryption supporting fine-grained keywords search and the security model. In the last subsection, we describe the mathematical tools, composite order bilinear groups and the related complexity assumption.

### *2.1. Predicate Encryption*

An inner-product predicate encryption scheme supports functionality $F : Key_{f_{\bar{v}}} \times CT_{\bar{x}} \rightarrow \{0,1\}^*$, where $\bar{x} \in Z_p^n \backslash \{\bar{0}\}$ and $\bar{v} \in Z_p^n \backslash \{\bar{0}\}$. If $\bar{x}\bar{v} = 0$, then $f_{\bar{v}}(\bar{x}) = 1$, else $f_{\bar{v}}(\bar{x}) = 0$. We denote the cipher-text space as $C$ and the message space as $M$.

Four algorithms Setup, KeyGen, Encrypt, and Decrypt for predicate encryption are defined as follows:

1. Setup$(1^\lambda) \rightarrow PK, MK)$. Given the security parameter $1^\lambda$, the public parameters $PK$ and the corresponding master key $MK$ are given out as the outputs;
2. KeyGen$(MK, \bar{v}) \rightarrow sk_{\bar{v}}$. This algorithm takes as input the master key $MK$ and a predicate vector $\bar{v}$, and outputs a user key $sk_{\bar{v}}$;
3. Encrypt$(PK, \bar{x}, m) \rightarrow c$. This algorithm takes as input an attribute vector $\bar{x}$ and the public parameters $PK$, and outputs a message $m \in M$ and a cipher-text $c \in C$;
4. Decrypt$(sk_{\bar{v}}, c) \rightarrow m$ or $\perp$. This algorithm takes as input the user key $sk_{\bar{v}}$ and a cipher-text $c$, and outputs the plain-text $m$ if and only if $f_{\bar{v}}(\bar{x}) = 1$.

In [11], IND-AH-CPA security for PE systems is defined by Lewko *et al.* via the following game. In subsection 4.1, for the construction of IND-PKES-CPA scheme, we will see that the IND-AH-CPA security definition is sufficient.

1. Setup. The challenger runs this algorithm and the adversary will get the public parameters;
2. Phase1. The adversary is allowed to adaptively issue queries for private keys on various predicates vector $\bar{v}$;
3. Challenge. Two equal length messages $m_0$ and $m_1$ and two attribute vectors $\bar{x_0}$, $\bar{x_1}$ are submitted by the adversary where $f_{\bar{v}}(\bar{x_0}) \neq 1$ and $f_{\bar{v}}(\bar{x_1}) \neq 1$ for all the key queried in Phase 1. The challenger flips a random coin $b$ and then encrypts $m_b$ with $\bar{x_b}$, which is $c^*$. Then the adversary will get it as the challenge cipher-text;
4. Phase2. In this phase, adaptive queries for private keys can be continued to be issued, except the key query for predicate $f_{\bar{v}}(\bar{x_0}) = 1$ and $f_{\bar{v}}(\bar{x_1}) = 1$;
5. Guess. Finally a guess $b'$ of $b$ is given out by the adversary.

In this game$|Pr[b = b'] - 1/2|$ is defined as the advantage of an IND-AH-CPA adversary.

**Definition 1.** *If for all polynomial time adversaries, they only have at most a negligible advantage in the above security game, then the predicate encryption scheme is IND-AH-CPA secure.*

### *2.2. Public-key Encryption with Fine-grained Keywords Search*

For one single message, there may exist many keywords in practice, such as conjunction predicates "urgent and business" in emails. Also the server often needs to search the emails with disjunction predicates like "family or company". This requirement can be satisfied by the public-key encryption with fine-grained keywords search (PEFKS). In PEFKS, a user can define more fine-grained relationship of keywords for the cipher-texts, which significantly extend appropriately in practice. PEFKS can be defined by the following algorithms.

1. KG$(1^k) \rightarrow (pk, sk)$, given the security parameter $\lambda$ as input, this algorithm outputs a secret key $sk$ and the corresponding public key $pk$; this is the key generation algorithm.
2. Td$(sk, \bar{w})$, this algorithm takes the secret key $sk$ and keywords vector $\bar{w}$ as input, and $t_w$ for keywords vector $\bar{w}$ are given out as the outputs; we call this algorithm as the trapdoor generation algorithm.
3. PEFKS$(pk, \bar{x}) \rightarrow \delta)$, this algorithm takes public key $pk$ and keywords vector $\bar{x}$ as input, and outputs $\delta$ for keywords vector $\bar{x}$; we call this algorithm as the encryption algorithm.
4. Test $(t_w, \delta) \rightarrow \{0, 1\}$, this algorithm takes in input the cipher-text $\delta$ for keywords vector $\bar{x}$, and the trapdoor $t_w$ for keywords vector $\bar{w}$, and outputs 1 if $\bar{w}\bar{x} = 0$, otherwise outputs 0; we call this algorithm as the search (or test/verification) algorithm.

Consistency. The consistency notion is defined by an experiment involving an adversary, in analogy with the definition of [9]. The experiment $Exp_{PEFKS,L}^{PEFKS-CONSISTENT}(l)$ is as follows: $(pk, sk) \leftarrow KG(1^l)$, $\bar{w}, \bar{x} \leftarrow L(pk)$, $\bar{w}, \bar{x} \neq 0$, $t_w \leftarrow Td(sk, \bar{w})$, $s \leftarrow PEFKS(pk, \bar{x})$, if $\bar{w} \neq 0$ and $Test(t_{\bar{w}}, s) = 1$ then return 1 else return 0. The advantage of $L$ is defined as $Adv_{PEFKS,L}^{PEFKS-CONSISTENT}(l) = Pr[Exp_{PEFKS,L}^{PEFKS-CONSISTENT} = 1]$

If the advantage is 0 for all adversaries $L$(computationally unrestricted), we call the scheme as perfectly consistent; if the advantage is negligible for all adversaries $L$(computationally unrestricted), we call the scheme as statistically consistent; if the advantage holds for all polynomial time adversaries $L$, we call the scheme as computational consistent. Computational consistency is adequate in practice, although it is weaker than statistical consistency and perfect consistency.

Security. We define the semantic security model (IND-PKFES-CPA) for the PEFKS.

1. Setup. The $KG$ algorithm is run by the challenger to get $(pk, sk)$, where $pk$ is given to the adversary;
2. Phase 1. In this phase, the adversary can adaptively issue queries on various keywords vector $\bar{w}$ to get the corresponding trapdoors $t_{\bar{w}}$;
3. Challenge. Two keywords vectors $\bar{x_0}, \bar{x_1}$ are submitted by the adversary where $\bar{x_0} \cdot \bar{w} \neq 0$ and $\bar{x_1} \cdot \bar{w} \neq 0$ for all previous queried keys during Phase 1. The challenger flips a random coin $b$, and the adversary is given $\delta^* \leftarrow PEFKS(pk, \bar{x}_b)$;
4. Phase 2. Like Phase 1, adaptive queries are issued by the adversary except that $\bar{x_0} \cdot \bar{w} = 0$ and $\bar{x_1} \cdot \bar{w} = 0$;
5. Guess. A guess $b'$ of $b$ is finally outputted by the adversary.
   In this game the advantage of the IND-PEFKS-CPA adversary is defined as $|Pr[b' = b] - 1/2|$.

**Definition 2.** *In this security game, if at most a negligible advantage can be achieved by all polynomial time adversaries, we call the PEFKS scheme as IND-PEFKS-CPA secure.*

*2.3. Assumption*

Our scheme relies heavily on composite order bilinear groups. In 2005, Boneh, Goh, and Nissim [24] first introduced the composite order bilinear groups for constructing homomorphic encryption. Elliptic curves over finite fields can be used to instantiate the composite order bilinear groups. Usually we require that the elliptic curve group order $N$ should be hard to factor, which can be of 1024 bits.

There exists a composite order bilinear group generation algorithm $\mathcal{G}$ for the elliptic curve group, we can also find an algorithm taking a security parameter $1^\lambda$ as input and let $(N = p_1p_2p_3, G, G_T, e)$ be the output, where $G$ and $G_T$ are cyclic groups of order $N$, $p_1, p_2, p_3$ are distinct primes and $e : G \times G \to G_T$ is a bilinear pairing such that:

1. Bilinear: for all $a, b \in Z_p$ and $u, v \in G$, there exist $e(u^a, v^b) = e(u, v)^{ab}$.
2. Non-degeneracy: we can efficiently find a $g$ such that $e(g, g)$ has order $N$ in $G_T$.

The subgroups of $G$ with order $p_2p_3$, $p_1p_3$, $p_1p_2$ are denoted as $G_{p_1}, G_{p_2}, G_{p_3}$ , satisfy the orthogonality property [25], namely, when $h_i \in G_{p_i}h_j \in G_{p_j}i \neq j$, then $e(h_i, h_j)$ is $G_T$'s identity element. We use this property in our construction next.

Now we formulate three complexity assumptions based on which we will prove security of our scheme. These subgroup assumptions are extensions of assumptions in [25]. We introduce an additional term $h \in G_{p_1}$, which will not add advantage to $A$ since the challenge tuples are independent of $h$.

Assumption 1: The following distribution is defined by given a group generator $\mathcal{G}$:

$$\begin{aligned}
&\mathcal{G} = (N = p_1p_2p_3, G, G_T, e)\\
&g, h \leftarrow_R G_{p_1}, X_3 \leftarrow_R G_{p_3}\\
&D = (G, g, h, X_3)\\
&T_0 \leftarrow_R G_{p_1}, T_1 \leftarrow_R G_{p_1p_2}
\end{aligned}$$

$Adv1_{G,A}(\lambda) := |Pr[A(D, T_0) = 0] - Pr[A(D, T_1) = 0]|$ is defined as the advantage of an algorithm $A$ in breaking Assumption 1.

**Definition 3.** *For any polynomial time algorithm $A$, if $Adv1_{G,A}(\lambda)$ is a negligible function of $1^\lambda$, we say that $\mathcal{G}$ satisfies Assumption 1.*

Assumption 2: The following distribution is defined by given a group generator $\mathcal{G}$:

$$\begin{aligned}
&\mathcal{G} = (N = p_1p_2p_3, G, G_T, e)\\
&g, h, X_1 \leftarrow_R G_{p_1}, X_2, Y_2 \leftarrow_R G_{p_2}, X_3, Y_3 \leftarrow_R G_{p_3},\\
&D = (G, g, h, X_3, X_1X_2, Y_2Y_3)\\
&T_0 \leftarrow_R G_{p_1p_3}, T_1 \leftarrow_R G
\end{aligned}$$

The advantage of an algorithm $A$ in breaking Assumption 2 can be defined as: $Adv2_{G,A}(\lambda) = |Pr[A(D, T_0) = 0] - Pr[A(D, T_1) = 0]|$.

**Definition 4.** *If for any polynomial time algorithm $A$, $Adv2_{G,A}(\lambda)$ is a negligible function of $1^\lambda$, then we say that $\mathcal{G}$ satisfies Assumption 2*

Assumption 3: The following distribution is defined by given a group generator $\mathcal{G}$:

$$
\begin{aligned}
&G = (N = p_1p_2p_3, G, G_T, e), r \in Z_N \\
&g, h \leftarrow G_{p_1}, X_2, Y_2, Z_2 \leftarrow G_{p_2}, X_3 \leftarrow G_{p_3}, \\
&D = (G, g, X_3, Z_2, g^r X_2, hY_2) \\
&T_0 = e(g,h)^r, T_1 \leftarrow G_T
\end{aligned}
$$

$Adv3_{G,A}(\lambda) = |Pr[A(D,T_0) = 0] - Pr[A(D,T_1) = 0]|$ is defined as the advantage of an algorithm $A$ in breaking Assumption 3.

**Definition 5.** *If for any polynomial time algorithm $A$, $Adv3_{G,A}(\lambda)$ is a negligible function of $1^\lambda$, then we say that $\mathcal{G}$ satisfies Assumption 3.*

## 3. Efficient Predicate Encryption

We construct an IND-AH-CPA secure inner-product predicate encryption scheme $F = \{f_{\bar{v}} | \bar{v} \in Z_p^n \backslash \{\bar{0}\}\}$, with $f_{\bar{v}}(\bar{x}) = 1$ if $\bar{x}\bar{v} = 0 \bmod p_1$ is this class of predicates. In our construction, for encryption and decryption we use the subgroup $G_{p_1}$; for key randomization, we use $G_{p_3}$; $G_{p_2}$ will not be used in real encryption scheme, but only for semi-functional keys and semi-functional cipher-text. The predicates of the user and attributes of the cipher-text are all expressed as vectors. It should be noted here that we require each element of the vector cannot be 0, and $\sum_{i=1,\cdots,n} x_i \neq 0$.

1. Setup($1^\lambda$). The KGC first runs $\mathcal{G}(1^\lambda)$ to get output $\mathcal{G} = (N = p_1p_2p_3, G, G_T, e)$. It then choose random generators $g, h \in G_{p_1}$, $X_3 \in G_{p_3}$ and random $a \in Z_N$, $t_i \in Z_N$, $i = 1, \cdots, n$, where $a \neq t_i$. The output public parameters and master key are

$$
\begin{aligned}
&PK = \{g, h, g_1 = g^a, \{T_i = g^{t_i}\}_{i=1,\cdots,n}\}, \\
&MK = \{a, \{t_i\}_{i=1,\cdots,n}\}
\end{aligned}
$$

2. KeyGen($MK, \bar{v}$). For user with predicate vector $\bar{v} = \{v_1, \cdots, v_n\}$, the KGC runs this algorithm to output a user's private key. First, random values $s \in Z_N$, and $W_i \in G_{p_3}, i = 1, \cdots, n$ are chosen, then it creates the private key as follows:

$$sk_{\bar{v}} = \{\{d_i = (hg^{sv_i}W_i)^{1/(a-t_i)}\}_{i=1,\cdots,n}\}$$

3. Encrypt($PK, \bar{x}, m$). For message $m \in M$ with sum not equal to 0 and attribute vector $\bar{x}$, the sender chooses random $r \in Z_N$ and encrypts as follows:

$$c = \{c_0 = me(g,h)^{-r\sum_{i=1,\cdots,n} x_i}, \{c_i = (g_1 T_i^{-1})^{rx_i}\}_{i=1,\cdots,n}\}$$

4. $\mathsf{Decrypt}(sk_{\bar{v}}, c)$. The cipher-texts are decrypted as follows:

$$c_0 \cdot \prod_{i=1,\cdots,n} e(c_i, d_i)$$

Correctness. If the cipher-text is well-formed, then we can see that correctness holds:

$$\begin{aligned}&c_0 \cdot \prod_{i=1,\cdots,n} e(c_i, d_i) = me(g,h)^{-r\sum_{i=1,\cdot,n} x_i}\\ &\cdot \prod_{i=1,\cdots,n} e((g_1^r T_i^{-r})^{x_i}, (hg^{sv_i}W_i)^{1/(a-t_i)}) = me(g,h)^{-r\sum_{i=1,\cdot,n} x_i} e(g,h)^{r\sum_{i=1,\cdot,n} x_i}\\ &\cdot e(g,h)^{sr\sum_{i=1,\cdot,n} x_i v_i} = me(g,g)^{sr\sum_{i=1,\cdot,n} x_i v_i}\end{aligned}$$

If $\bar{x}\bar{v} \neq 0 \bmod p_1$, then the decryption result will be a random element in the group of $G_T$. If $\bar{x}\bar{v} = 0 \bmod p_1$, namely $f_{\bar{v}}(\bar{x}) = 1$, the decryption result will be $m$.

### *3.1. Security*

Like [25, 7], we use the dual system encryption methodology to prove the security. Two additional structures: semi-functional cipher-texts and semi-functional keys are used in our proof, but will not be used in the real system.

Semi-functional cipher-text. Let $g_2$ denote a generator of $G_{p_2}$. $c \in Z_N$, and $\{z_i \in Z_N\}_{i=1,\cdots,n}$ are random selected. A semi-functional cipher-text consists of the following:

$$\{c_i = (g^{r_1 x_i} g_2^{cz_i})^{a-t_i}\}_{i=1,\cdots,n}$$

Semi-functional Key. Let $g_2$ denote a generator of $G_{p_2}$. $d \in Z_N$ and $\{y_i \in Z_N\}_{i=1,\cdots,n}$ are random selected. A semi-functional key consists of the following:

$$\{d_i = (hg^{sv_i}W_i g_2^{dy_i})^{1/(a-t_i)}\}_{i=1,\cdots,n}$$

Both normal and semi-functional cipher-texts can be decrypted by a normal key, while both normal and semi-functional keys can decrypt a normal cipher-text. When a semi-functional cipher-text is decrypted by a semi-functional key, there exists an additional term $e(g_2, g_2)^{cd\sum_{i=1,\cdots,n} y_i z_i}$. Notice that when decrypting a semi-functional cipher-text, a semi-functional key, which satisfying $\sum_{i=1,\cdots,n} y_i z_i = 0$, can be used to decrypt successfully.

By using a sequence of games, we prove the security of our system based on the above assumptions. Both cipher-texts and keys are normal in $Game_{real}$, while in the second game $Game_0$, all keys are normal and the cipher-text is semi-functional. The first $k$ key queries are semi-functional and the remaining are normal during Game $Game_k$. When reaching to the final game $Game_{final}$, the challenge cipher-text is a semi-functional encryption of a random message, while all of the key queries are semi-functional. These games are indistinguishable based on the following lemmas.

**Lemma 1.** *If for a polynomial time adversary $\mathcal{A}$, there exists $Adv_{\mathcal{A}}^{Game_{real}} - Adv_{\mathcal{A}}^{Game_0} = \epsilon$, then with advantage $\epsilon$, we can construct a polynomial time simulator $\mathcal{B}$ breaking Assumption 1.*

Proof. $\mathcal{B}$ is given a challenge sample of Assumption 1 as an input of the Setup algorithm, $(\mathcal{G}, g, h, X_3, T)$. $\mathcal{B}$ chooses random value $a \in Z_N$, $t_i \in Z_N$, $i = 1 \cdots, n$. The public parameters are the same as Setup algorithm. $\mathcal{B}$ will simulate $Game_{real}$ and $Game_0$ with $\mathcal{A}$.

For the key queries $\bar{v}$, since the KeyGen algorithm $\mathcal{B}$ knows the $MK$, by using this secret value, it can generate a normal key.

As for the challenge $(m_0, m_1)$ and $(\bar{x_0}, \bar{x_1})$, $\mathcal{B}$ embed the Assumption 1 into the challenge cipher-text. A random coin $b$ is first flipped and set as:

$$c^* = \{c_0 = m_b e(T, h)^{-\sum_{i=1,\cdots,n} x_i^b}, \{c_i = T^{x_i^b(a-t_i)}\}_{i=1,\cdots,n}\}$$

If $T \in G_{p_1}$, namely $T = g^r$, then this is a correct normal cipher-text. If $T \in G_{p_1 p_2}$, namely $T = g^r g_2^c$, we implicitly set $z_i = x_i$. However from conclusion of the Chinese Remainder Theorem, the values of $x_i \bmod p_1$ are independent from $z_i \bmod p_2$. The cipher-text is a semi-functional cipher-text with properly distribution. $\mathcal{A}$ can be used by $\mathcal{B}$ to gain advantage $\epsilon$ in breaking Assumption 1.

**Lemma 2.** *If for a polynomial time adversary $\mathcal{A}$, there exists $Adv_{\mathcal{A}}^{Game_{k-1}} - Adv_{\mathcal{A}}^{Game_k} = \epsilon$, then with advantage $\epsilon$, we can construct a polynomial time simulator $\mathcal{B}$ breaking Assumption 2.*

Proof. $(\mathcal{G}g, h, X_3, X_1X_2, Y_2Y_3, T)$ as a challenge sample of Assumption 2 is given to $\mathcal{B}$. Similarly as in the proof of Lemma 1, the public parameters are generated in the same way. $\mathcal{B}$ simulates $Game_{k-1}$ and $Game_k$ with $\mathcal{A}$.

Upon receiving key queries $\bar{v}$, for queries larger than $k$ $\mathcal{B}$ forms normal keys, for queries less than $k$ $\mathcal{B}$ forms semi-functional keys, for the $k$th query $\mathcal{B}$ forms either normal or semi-functional.

By using $MK$, $\mathcal{B}$ can generate normal keys by running the KeyGen algorithm, for queries larger than $k$. For the queries less than $k$, $\mathcal{B}$ chooses random values $s, d \in Z_N$, then the following is defined as the semi-functional key:

$$\{d_i = (hg^{sv_i} W_i (Y_2Y_3)^{dy_i})^{1/(a-t_i)}\}_{i=1,\cdots,n}$$

$\mathcal{B}$ uses the value of $\mathcal{T}$ in the challenge for the $k$th key query. The key is set as:

$$\{d_i = (hT^{v_i} W_i)^{1/(a-t_i)}\}_{i=1,\cdots,n}$$

If $T \in G_{p_1} G_{p_3}$, then this is a properly distributed normal key. If $T \in G_{p_1 p_2 p_3}$, namely $T = g^s g_2^d g_3^f$, we implicitly set $y_i = v_i$. This is a semi-functional key due to the Chinese Remainder Theorem.

Now, we discuss whether $\mathcal{B}$ can distinguish the simulated $k$th query, which is semi-functional in $Game_k$ and normal in $Game_{k-1}$ by itself. If $\mathcal{B}$ itself has constructed a valid semi-functional cipher-text with $\bar{x}\bar{v} = 0$, since $\mathcal{B}$ doesn't know the factor of $N$ and $X_1X_2$ is the only one can be used for semi-functional cipher-text, the simulated cipher-text must contain it. It implies that $z_i = x_i$. Then we have $\bar{z}\bar{y} = \bar{x}\bar{v} = 0$. Decryption still works fine. Therefore, $\mathcal{B}$, without ability to distinguish the simulated $k$th key from semi-functional and functional, can only use the output of $\mathcal{A}$ to break the Assumption.

As for the challenge $(m_0, m_1)$ and $(\bar{x_0}, \bar{x_1})$, $\mathcal{B}$ flips a random coin $b$, and sets: $c^* = \{c_0 = m_b e(X_1X_2, h)^{-\sum_{i=1\cdots n} x_i^b}, \{c_i = (X_1X_2)^{x_i^b(a-t_i)}\}_{i=1,\cdots,n}\}$

If $X_1X_2 = g^r g_2^c$, we then set $z_i = x_i$. Following the rule of the Chinese Remainder Theorem, these values are also actually uncorrelated in the subgroups $p_1$, $p_2$. This is a properly distributed semi-functional cipher-text.

The output of $\mathcal{A}$ can be used by $\mathcal{B}$ to break Assumption 2 with advantage $\epsilon$.

**Lemma 3.** *A polynomial time simulator $\mathcal{B}$ with advantage $\epsilon$ can be constructed to break Assumption 3, if there is a polynomial time adversary $\mathcal{A}$ such that $Adv_{\mathcal{A}}^{Game_q} - Adv_{\mathcal{A}}^{Game_{final}} = \epsilon$.*

Proof. $(\mathcal{G}, g, X_3, Z_2, g^r X_2, hY_2, T)$ as a challenge sample of Assumption 3, is given to $\mathcal{B}$. $\mathcal{B}$ chooses random values $a \in Z_N$, $t_i \in Z_N$, $i = 1 \cdots, n$. The public parameters are set as: $PK = (g, hY_2, g_1 = g^a, \{T_i = g^{t_i}\}_{i=1,\cdots,n})$. For it is hard to find a factor of $N$, $hY_2$ will seem indistinguishable from $h$ to $\mathcal{A}$. $\mathcal{B}$ simulate $Game_q$ and $Game_{final}$ as follows.

As for the key queries $\bar{v}$, $\mathcal{B}$ chooses random $s, y_i' \in Z_N$, and sets the semi-functional key as:

$$\{d_i = (hY_2 g^{sv_i} Z_2^{y_i'} W_i)^{1/(a-t_i)}\}_{i=1,\cdots,n}$$

Let $Y_2 = g_2^f, Z_2 = g_2^d$, we implicitly set $y_i = y_i' + f/d$. This is a semi-functional cipher-text.

As for the challenge $(m_0, m_1)$ and $(\bar{x_0}, \bar{x_1})$, $\mathcal{B}$ will embed into the challenge cipher-text the Assumption 3. It flips a random coin $b$, and sets: $c^* = \{c_0 = m_b T^{-\sum_{i=1\cdots n} x_i^b}, \{c_i = (g^r X_2)^{x_i^b(a-t_i)}\}_{i=1,\cdots,n}\}$

If $T = e(g, h)^r$, it is a valid semi-functional cipher-text. If $T \in G_T$ , this is a semi-functional encryption for a random message, and this simulation of $Game_{final}$ is perfect.

The output of $\mathcal{A}$ can be used by $\mathcal{B}$ to gain advantage $\epsilon$ in breaking Assumption 3.

**Theorem 1.** *Our PE system is IND-AH-CPA secure based on the Assumptions 1, 2, and 3.*

Proof. According to the previous lemmas, $Game_{real}$ is indistinguishable from $Game_{final}$ based Assumptions 1, 2, and 3. The challenge cipher-text is independent with $b$ for $Game_{final}$. Therefore, our predicate encryption scheme is IND-AH-CPA secure.

Table 1: Comparisons of computational efficiency with other schemes

| Schemes | Key Size | Cipher-text Size | Encryption time | Decryption time | Security |
|---|---|---|---|---|---|
| Katz [2] | $2(n+1)g$ | $2(n+1)g + 1g_T$ | $(4n+2)p$ | $(2n+1)e$ | Selective |
| Lewko [11] | $Ng$ | $Ng + 1g_T$ | $Np$ | $Ne$ | adaptive |
| Ours | $ng$ | $ng + 1g_T$ | $(1+n)p$ | $ne$ | adaptive |

### *3.2. Efficiency*

On cipher-text size, private key size, and computation time for decryption and encryption, our scheme has better efficiency compared to [2, 11], we can see the results in Table 1.

Let us note here $N \geq 2n + 3$. $g$ denotes one group element in $G$ and $g_T$ denotes one group element in $G_T$. $p$ and $e$ represents the modular power operation and pairing operation. The cost of our scheme is only a half of [2, 11] fromTable 1. Only approximately one group element in $G$ is contained in the cipher-text for each attribute, while two in [2]. Only one group elements in $G$ for each attribute for the user's private keys, while two in [2]. $e(g, h)$ and $(g_i T_i^{-1})$ can be pre-computed for the encryption, so there do not exist any pairing operation. For each attribute, it only requires one power operation, while more than two in [11] and four in [2]. For the decryption, one pairing needs to be implemented for every each attribute, while two in [2] and more than two in [11] are needed. Therefore our construction is more efficient.

## 4. PE-TO-PEFKS Transformation

In [8], Boneh *et al.* proved that an IND-ID-CCA secure IBE could be constructed from a secure PEKS, they also derived that from a secure IBE it was difficult to construct a PEKS, which was left as an open problem. Abdalla *et al.* [9] solved this open problem in 2005, they found a general way to construct an IND-PEKS-CPA secure and computationally consistent PEKS from any IND-ANO-CPA secure IBE. But only whether the keyword in the cipher-text being matched to the trapdoor with corresponding keyword can be tested for this kind of PEKS. Aiming to generalize their work, we propose a general way to transform an IND-AH-CPA secure PE into a secure PEFKS scheme.

The following steps define the PE-to-PEFKS transformation:

1. $KG(1^\lambda)$ use Setup$(1^\lambda)$ to generate $(pk, sk)$;
2. $Td(sk, \bar{w})$ use KeyGen algorithm to get $t_{\bar{w}}$, then it deliver $t_{\bar{w}}$ to the server;
3. First, choose a random element $R$, $PEFKS(pk, \bar{x})$ use $\mathsf{Encrypt}(PK, \bar{x}, R) \rightarrow c$ to encrypt keywords $\bar{x}$, and set $\delta = (R, c)$;
4. If Decrypt$(sk_{\bar{v}}, c) \rightarrow R$, then Test $(t_{\bar{w}}, \delta) \rightarrow 1$. Otherwise Test $(t_{\bar{w}}, \delta) \rightarrow 0$.

Our scheme's consistency and security can be reduced to the security of predicate encryption. We give the formal result and proof in Theorem 2.

**Theorem 2.** *The transformed PEFKS is IND-PEFKS-CPA secure and computational consistency, if PE is IND-AH-CPA secure.*

Proof. Assume adversary $\mathcal{L}_1$ can successfully attack the computational consistency of PEFKS and let $\mathcal{A}$ be a polynomial time adversary of PE. In the key queries phase, $\mathcal{A}$ runs $\mathcal{L}_1(pk)$ to get predicate vector $\bar{v'}$ and attribute vector $\bar{x}$ such that $\bar{x}\bar{v'} \neq 0$ but Test still output 1. $\mathcal{A}$ also get $(R_0, R_1)$, which can be used to break the computational consistency for $\mathcal{L}_1$. $\mathcal{A}$ then issue query for the challenge phase, $(R_0, R_1)$ and $\bar{x}$, and the challenge cipher-text $c^*$ encrypting $R_b$ under $\bar{x}$ is given out as the result. $\mathcal{A}$ makes key query for $\bar{v'}$, and runs $Decrypt(sk_{\bar{v'},c^*})$ to find $b$. The data privacy property of PE scheme can be broken, namely

$$Adv^{PEFKS-CONSISTENCY}_{\sum_{PEFKS},\mathcal{L}_1}(\lambda) \leq Adv^{PE-IND-CPA}_{\sum_{PE},\mathcal{A}}(\lambda)$$

Assuming there is a polynomial time adversary $\mathcal{L}_1$ that can break the IND-PEFKS-CPA security of PEFKS. Let $\mathcal{A}$ be a polynomial time adversary of PE. In the key queries phase, $\mathcal{A}$ runs $\mathcal{L}_2(pk)$ to get challenge attribute vectors $\bar{x_0}, \bar{x_1}$ with R. Given the challenge cipher-text $c^*$ encrypting $R$ under $\bar{x_b}$, $\mathcal{A}$ runs $\mathcal{L}_2$ to find $b$. During this phase, $\mathcal{A}$ answers any trapdoor query of $\mathcal{L}_2$ via its key queries. The attribute hiding property of PE scheme can be broken, namely

$$Adv^{PEFKS-IND-CPA}_{\sum_{PEFKS},\mathcal{L}_2}(\lambda) \leq Adv^{PE-IAH-CPA}_{\sum_{PE},\mathcal{A}}(\lambda)$$

### *4.1. Our PEFKS scheme*

We construct an IND-PEFKS-CPA secure PEFKS based on one efficient PE. The PEFKS runs as follows:

1. $\mathsf{KG}(1^\lambda)$: KG$(1^\lambda)$ runs exactly the Setup$(1^\lambda)$ algorithm

$$pk = PK = \{g, h, g_1 = g^a, \{T_i = g^{t_i}\}_{i=1,\cdots,n}\}, sk = MK = \{a, \{t_i\}_{i=1,\cdots,n}\}$$

2. $\mathsf{Td}(sk, \bar{w})$: Td$(sk, \bar{w})$ runs exactly the KeyGen$(MK, \bar{w})$ algorithm

$$t_w = \{\{d_i = (hg^{sw_i}W_i)^{1/(a-t_i)}\}_{i=1,\cdots,n}\}$$

3. $\mathsf{PEFKS}(\bar{pk}, \bar{x})$: By first choosing a random element $R$, the sender runs the Encrypt$(PK, \bar{x}, R)$ to encrypt keyword vector $\bar{x}$, and get $c$ as

$$c = \{c_0 = Re(g,h)^{-r\sum_{i=1,\cdots,n} x_i}, \{c_i = (g_1 T_i^{-1})^{rx_i}\}_{i=1,\cdots,n}\}$$

   The cipher-text consists of $\delta = (R, c)$;

4. $\mathsf{Test}(t_{\bar{w}}, \delta)$: The server computes $c_0 \prod_{i=1,\cdots,n} e(c_i, d_i)$. If it equals to $R$, namely $\bar{w}\bar{x} = 0$, the server sets Test$(t_{\bar{w}}, \delta)$=1. Otherwise it sets Test$(t_{\bar{w}}, \delta)$=0.

### *4.2. Discussion on PEFKS*

Only equal relation can be supported from previous PEKS schemes. The new notion of PEFKS generalizes the concept of traditional searchable encryption. More sophisticated and flexible relations between the encryption-keyword and trapdoor-keyword can be provided.

In fact, PEFKS can include previous PEKS as subclasses, that means that PEFKS supports equal relation. E.g. in previous PEKS, the encrypted keyword vector is set as $\bar{x} = (w', -1)$ and the keyword vector is set as $\bar{w} = (1, w)$. If $w = w'$, namely $\bar{w}\bar{x} = 0$, correctness and security follow.

PEFKS fully provides multiple keywords search that are expressed by conjunctive or disjunctive logical connectives.

- For a disjunctive logical connective, "$w_1$ and $w_2$" which corresponds to the polynomial evaluation $p = r(w_1 - x_1) + (w_2 - x_2)$, the keyword vector is set as $\bar{w} = (rw_1, -r, w_2, -1)$. If $\bar{x} = (1, x_1, 1, x_2)$ and $p = 0$, the Test will be evaluated to 1.

- For a conjunctive logical connective, "$w_1$ or $w_2$" which corresponds to the polynomial evaluation $p = r(w_1 - x_1)(w_2 - x_2)$, the keyword vector is set as $\bar{w} = (w_2w_1, -w_1, -w_2, -1)$. If $\bar{x} = (1, x_1, x_2, x_1x_2)$ and $p = 0$, the Test will be evaluated to 1.

More complex combinations for boolean formulas can be extended by conjunctive or disjunctive logical connectives. More general polynomial evaluation $p = w_0 + w_1x + \cdots + w_dx^d$ can also be extended by the above polynomial evaluation.

Here we describe a simple application PEFKS's. If the email is appended with keywords "business and urgent", the email server will deliver the email immediately to the users. "urgent" and "business" can be defined as some values in the system. The user sets the trapdoor $t_{\bar{w}}$ as $\bar{w} = (w_2w_1, -w_1, -w_2, -1)$, where $w_1$ denotes "urgent" and $w_2$ denotes "business". If "urgent and business" are the associated keywords with the email, a sender encrypts the email and appends with the cipher-text $\delta$ of keywords vector $\bar{x} = (1, w_1, w_2, w_1w_2)$. The server then can test whether this email is "urgent and business" by running $\text{Test}(t_{\bar{w}}, \delta)$.

### *4.3. Comparison of features from various schemes*

In this subsection, we give the features comparison between our scheme and other related schemes, the results are summarized in Table 2.

From Table 2 we can see that our scheme can simultaneously support privacy-preserving encryption, fine-grained search capability and multi-keywords search, while other schemes do not support these three properties simultaneously. It should be noted however that our proposal cannot achieve verifiability of the search results, which is target our future work.

Table 2: Features Comparisons

| Schemes | Privacy-preserving encryption | Fine-grained search | Multi keywords search | Verifiable search results |
|---|---|---|---|---|
| Li *et al.* [19] | No | Yes | Yes | No |
| Cao *et al.* [20] | Yes | No | Yes | No |
| Zheng *et al.* [21] | No | Yes | No | Yes |
| Sun *et al.* [22] | No | Yes | No | No |
| Sun *et al.* [23] | No | Yes | Yes | Yes |
| Ours | Yes | Yes | Yes | No |

## 5. A Framework for Privacy Preserving Predicate Encryption with Fine-grained Searchable Capability for Cloud Storage

Our framework for privacy preserving predicate encryption with fine-grained searchable capability for Cloud storage is described in Fig. 1:

1. Data owner Alice first encrypts her data sets as follows: she first uses our predicate encryption to encrypt the encapsulated key $K$, then she uses the block cipher like AES to encrypt the data sets by using the encapsulated key $K$.
2. She establishes the searching index for the encrypted files. She first chooses the keyword vector $\bar{w}$ corresponding to each data item and encodes its corresponding encrypted keyword vector $\bar{x}$ (these encodings can also follow a setup rule defined for the whole system in the setup phase). Notice here that the keyword vector is different from the encrypted keyword vector, they consist of an inner-product relationship. Then she publishes the encrypted keyword vector and the keyword vector. Finally she encrypts the corresponding encrypted keyword vector $\bar{x}$ by using the PEFKS mechanism and get the corresponding index header file for that data item.
3. She outsources the corresponding index header, the encrypted encapsulated key and the encrypted data item as a whole data packet to the Cloud. It should be noted here that the encrypted data item can be larger than the first two parts, in this way the system's efficiency is significantly improved.
4. After outsourcing these encrypted data sets, in order to enable the Cloud can search on these cipher-texts, Alice now sets up the search trapdoor for the Cloud. Here we use a proxy at the Cloud to implement the search task, in practical applications this proxy can be regarded as a very powerful server or information management system. Alice outsources the search trapdoor corresponding to the keyword vector $\bar{w}$ to the Cloud. Note here that this search trapdoor can be first obtained by Alice from querying the PKG.
5. All of the above steps can be seen as **Phase 1**. The next steps can be seen as Phase 2. In Phase 2, the data user Bob (also can include Alice herself) wants to search on the cipher-texts. He first queries from the Cloud with the keyword vector $\bar{w}$.

6. Now the Cloud (the proxy) first finds the search trapdoor associated with the keyword vector $\bar{w}$. Then it uses this trapdoor to implement the Test algorithm of PEFKS on the first header of each cipher-text data packet. If the cipher-texts contains this keyword, the Test algorithm will return 1, otherwise it will return 0. In this way, the Cloud can implement this searching and return the correct search results to the users.

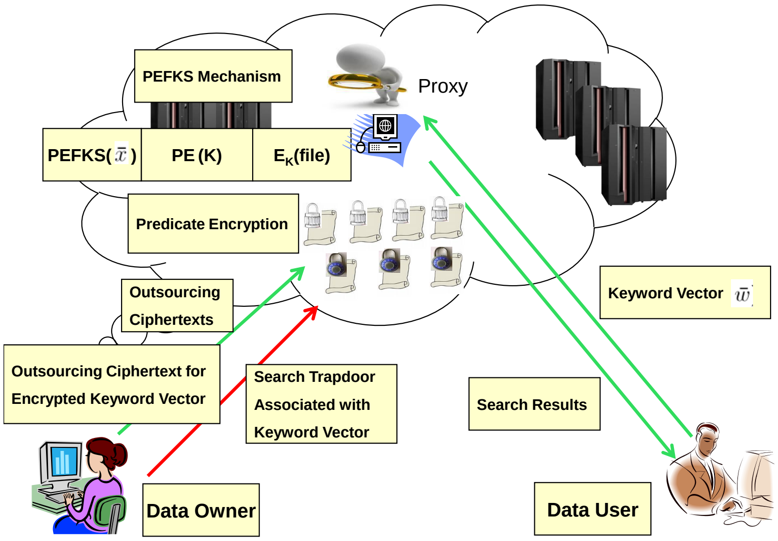

Figure 1: The framework

Below we sketch the analysis of the security properties of this framework:

1. *Confidential property.* In our framework, all the data sets are encrypted by hybrid encryption with keyword search. Concretely, the data sets are encrypted by the block cipher, the encapsulated keys are encrypted by the predicate encryption, and the keyword vector are encrypted by the public key encryption with fine grained search mechanism. All the Cloud knows are the cipher-texts, it implements searching on the cipher-texts, thus our framework can achieve the confidential property.
2. *Privacy preserving property.* In our framework, data sets are encrypted by the block cipher and the encapsulated keys are encrypted by the predicate encryption. As it is known, predicate encryption is a privacy preserving encryption mechanism. Compared to attribute based encryption, the cipher-texts in predicate encryption not only conceals the payload but also conceals the attributes. Furthermore, our PEFKS mechanism also provides some extent privacy preserving property. Unlike the traditional public key encryption with keyword search, the keyword vector for the search trapdoor generation and the encrypted keyword vector are different. The encoding for the keyword vector and its dual encrypted keyword vector provides some extent of privacy preserving property for the inner-product relationship, for example, there are many solutions of $\bar{y}$ for a fixed $\bar{x}$ if $\bar{x} \cdot \bar{y} = 0$.

3. *Efficiency property.* First, as shown above, our predicate encryption is more efficient than some classic predicate encryption schemes. Secondly, the main computation load is encrypting the data sets by using block cipher, which is very efficient. For our framework uses the hybrid encryption with keyword search, thus our framework is much more efficient in practical applications than directly using public key encryption with keyword search.
4. *Ffne-grained searchable property.* As shown above, our system can support complex searching patterns by adapting proper encoding techniques for the encrypted keyword vector and the keyword vector for searching trapdoor, while the traditional public key encryption with keyword search can only support equal searching relationship. Note that the relationship of inner-product zero can be used to construct very expressive relationships.

## 6. Conclusion

Data security and privacy has become a major challenge for Cloud computing applications and Cloud Data Storage systems. Due to continuous data outsourcing to Cloud platforms, there is an increasing need to efficiently implement encryption schemes for data protection yet such schemes should allow flexible searching, processing, analysis and mining of data sets at large scale. In this paper, we have presented an efficient inner-product predicate encryption system. Our construction is IND-AH-CPA (indistinguishable under chosen plain-text attack for attribute-hiding) secure by using the dual system encryption, which is sufficient for the PEFKS scheme resulted from the PE-TO-PEFKS transformation. We showed how to design a framework for privacy preserving predicate encryption with fine-grained searchable capability for Cloud storage. We also presented an analysis for our framework's properties.

There are some interesting open problems that deserve further investigation, such as designing more sophisticated and flexible functionality $F : Key \times CT \to \{0,1\}^*$ than inner-product for PEFKS, exploiting the way of transformation of PEFKS to PE, etc., which we envisage for future work.

## 7. Acknowledgements

Fatos Xhafa's work has been partially supported by funds from the Spanish Ministry for Economy and Competitiveness (MINECO) and the European Union (FEDER funds) under grant COMMAS (ref. TIN2013-46181-C2-1-R). This work is an extended of [26] and was supported by the Key Problem of NFSC-Guangdong Union Foundation (Grant NO. U1135002), National High Technology Research and Development Program (863 Program) (No. 2015AA016007), the Changjiang Scholars and Innovation Research Team in University (Grant NO. IRT 1078), the Major Nature Science Foundation of China (Grant NO. 61370078), Nature Science Foundation of China (Grant NO. 61272492, 61202492, 61402531, 61572390), Natural Science Foundation of Shaanxi Province (Grant No. 2014JM8300).

## References

[1] Huang Y., Dai C., Leu F., You I. A Secure Data Encryption Method Employing a Sequential-Logic Style Mechanism for a Cloud System In *International Journal of Web and Grid Services*, 11(1), pp. 102-124, 2015.

[2] Katz J., Sahai A., Waters B.. Predicate Encryption Supporting Disjunctions, Polynomial Equations, and Inner Products. In *EUROCRYPT 2008*. LNCS 4965, pp. 146-162.

[3] Wang H.. Identity-based distributed provable data possession in multicloud storage. In *IEEE T. Services Computing*, 8(2), pp. 328–340, 2015.

[4] Wang H.. Proxy provable data possession in public Clouds. In *IEEE T. Services Computing*, 6(4), pp. 551–559, 2013.

[5] Wang H., Wu Q., Qin B., Domingo-Ferrer J.. Identity-based remote data possession checking in public Clouds. IET Information Security, 8(2), pp. 114–121, 2014.

[6] Wang H., Wu Q., Qin B., Domingo-Ferrer J.. FRR: Fair remote retrieval of outsourced private medical records in electronic health networks. Journal of Biomedical Informatics, 50, pp. 226-233, 2014.

[7] Waters B.. Dual System Encryption: Realizing Fully Secure IBE and HIBE under Simple Assumptions. In *CRYPTO 2009*, LNCS 5677, pp. 619-636.

[8] Boneh D., Crescenzo G. D., Ostrovsky R., and Persiano G.. Public Key Encryption with Keyword Search. In *Eurocrypt 2004*, LNCS 3027, Springer-Verlag, 2004, pp. 506-522.

[9] Abdalla M., Bellare M., Catalano D., Kiltz E., Kohno T., Lange T., Malone-Lee J., Neven G., Paillier P. and Shi H.. Searchable Encryption Revisited: Consistency Properties, Relation to Anonymous IBE, and Extensions. In *Crypto 2005*, LNCS 3621, Springer-Verlag, 2005, pp. 205-222.

[10] Shi E., Waters B.. Delegating Capabilities in Predicate Encryption Systems. In *ICALP 2008*, LNCS 5126, pp. 560-578.

[11] Lewko A., Okamoto T., Sahai A., Takashima K., Waters B.. Fully Secure Functional Encryption: Attribute-based Encryption and (Hierarchical) Inner Product Encryption. In *EUROCRYPT 2010*, LNCS 6110, pp. 62-91.

[12] Zhang M., Yang B. and Takagi T.. Bounded leakage-resilient functional encryption with hidden vector predicate. In *Computer Journal*, 56(4): 464-477, 2013.

[13] Zhang M., Wang C., Takagi T., and Mu Y.. Functional encryption resilient to hard-to-invert leakage. In *Computer Journal*, 58(4), pp. 735-749, 2015.

[14] Zhang M., Wang C., Morozov K.. LR-FEAD: Leakage-tolerating and attribute-hiding functional encryption mechanism with delegation in affine subspaces. In *Journal of Supercomputing*, 70(3), pp. 1405-1432, 2014

[15] Zhang M., Yang B., Takagi T.. Anonymous spatial encryption under affine space delegation functionality with fully security. In *Information Sciences*, 277, pp. 715-730, 2014

[16] Zhang M., Zhang Y., Su Y., Huang Q., Mu Y.. Attribute-based Hash Proof System under Learning-with-Errors Assumption in Obfuscator-free and Leakage-resilient Enviroments. In *IEEE System Journals*, DOI:10.1109/JSYST.2015.2435518.

[17] Zhang M. and Mu Y.. Token-leakage tolorant and vector obfuscated IPE and application in privacy-preserving two-party point/polynomial evaluations. In *Computer Journal*, DOI: 10.1093/comjnl/bxv065.

[18] Attrapadung N., Hanaoka G., and Yamada S.. Conversions among several classes of predicate encryption and applications to ABE with various compactness tradeoffs. Cryptology ePrint Achieve, https://eprint.iacr.org/2015/431.pdf

[19] Li J., Wang, Q., Wang. C., Cao N., Ren K., Lou. W. Enabling efficient fuzzy keyword search over encrypted data in Cloud computing. In Infocom 2010, pp. 441-445, also available at Cryptology ePrint Achieve, https://eprint.iacr.org/2009/593.pdf

[20] Cao N., Wang C., Li M., Ren K., Lou W. Privacy preserving multi-keyword text search in the Cloud supporting similarly-based ranking. In *IEEE Transactions on Parallel Distribution System*, 25(1), pages 222–233, 2014.

[21] Zheng Q., Xu. S., Ateniese VABKS: verifiable attribute-based keyword search over outsourced encrypted data. Cryptology ePrint Achieve, https://eprint.iacr.org/2013/462.pdf

[22] Sun W., Yu S., Lou W., Hou T., Li. H. Protecting your right: attribute-based keyword search with fine-grained owner-enforced search authorizatioin in the Cloud. In *IEEE Infocom 2014*, 226–234.

[23] Sun W., Liu X., Lou. W., Hou. T., Li. H. Catch you if you lie to me: efficient verifiable conjunctive keyword search over large dynamic encrypted Cloud data. In *IEEE Infocom 2015*, 2110-2118.

[24] Boneh D., Goh E., Nissim K.. Evaluating 2-dnf formulas on cipher-texts. In *TCC 2005*, LNCS , pp. 325-342.

[25] Lewko A., Waters B.. New Techniques for Dual System Encryption and Fully Secure HIBE with Short cipher-texts. In *TCC 2010*, LNCS 5978, pp. 455-479.

[26] Zhang M., Wang X., Yang X., Cai W. Efficient Predicate encryption supporting construction of fine-grained searchable encryption. In *INCoS 2013*, 438–442.