Generic Combination of Public Key Encryption with Keyword Search and Public Key Encryption

Rui Zhang and Hideki Imai

Research Center for Information Security (RCIS)
National Institute of Advanced Industrial Science and Technology (AIST)
{r-zhang,h-imai}@aist.go.jp.

Abstract. In this paper, we study the problem of secure integrating public key encryption with keyword search (PEKS) with public key data encryption (PKE). We argue the previous security model is not complete regarding keyword privacy and the previous constructions are secure only in the random oracle model. We solve these problems by first defining a new security model, then give a generic construction which is secure in the new security model without random oracles. Our construction is based on secure PEKS and tag-KEM/DEM schemes and achieves modular design. We also give some applications and extensions for our construction. For example, instantiate our construction with proper components, we have a concrete scheme without random oracles, whose performance is even competitive to the previous schemes with random oracles.

1 Introduction

Public key encryption with key word search (PEKS) [7] is very useful to provide the functionality of "searching on encrypted data" for public key cryptosystems. For instance, it can be used to build a gateway to route an encrypted email without knowing the content. We briefly review this mechanism here. Let (pk, sk) be Alice's public/secret key pair. Bob encrypts his message (email body) m with a public key encryption (PKE) scheme under Alice's public key pk and let's call the encrypted email σ. Bob also encrypts a keyword w using PEKS, under Alice's public key pk and let's call the encrypted keyword τ. The resulting ciphertext $c = \tau||\sigma$ will be sent to Alice's email server. Alice is able to specify a few keywords, and upon receiving a trapdoor t_w associated with a keyword w from Alice, the server can check whether τ encrypts w. Then if the keyword is "urgent", the server sends c to Alice's mobile phone, and if the keyword is "lunch", the server sends c to Alice's desktop to be read later. The security of PEKS is that the server should not know anything beyond the keyword. Readers are recommended to refer [7] and the references thereafter for details.

The security requirements discussed in [7, 1] have considered semantic security of encryption of keywords against a powerful adversary that adaptively corrupts gateways. Since a PEKS scheme cannot be used alone but have to be paired with a public key encryption (PKE) scheme, we have to consider the security of the whole system rather than separate components. Hereafter we refer the integrated scheme as PEKS/PKE. Unfortunately, secure PEKS and secure PKE schemes may not remain secure when they are composed together, which was pointed out by Baek, Safavi-Naini and Susilo [3]. Basically, they gave a counterexample as follows: When an adversary observes a PEKS/PKE ciphertext $\tau||\sigma$, it can produce

another valid ciphertext $\tau'||\sigma$, where τ' is a valid tag under different keyword. Querying $\tau'||\sigma$ to a decryption oracle, the adversary obtains the plaintext m.

We remark that the above attack is realistic in practice, since for most encrypted email systems, headers of an email remain even after routing, and the decryption is done without integrity check on the header. On the other hand, for keyword privacy, nothing was considered against chosen keyword/ciphertext attack before this work.

Known Solutions and Their Limitations. As mentioned in [3], a trivial solution may be simply appending an authentication tag generated from a message authentication code (MAC), with a shared key between the sender and receiver. While it works, the solution destroys the asymmetric nature of public key encryption. Another possible solution is to attach a signature on the ciphertext. However, this requires the sender has a pair of verification/signing keys, which is not applicable for many practical scenarios.

Additionally, two solutions were given in [3], assuming that a MAC is provided by the PKE component. One is based on the Boneh-Franklin identity based encryption (IBE) [8] as a PEKS [7] with ElGamal [16] as a PKE. The other is a generic construction based on a PEKS and a PKE with a MAC. The intuition behind both constructions is borrowed from REACT [22], where a MAC is used to protect the integrity of both parts of the ciphertext. However, to prove their security, the authors of [3] have to assume the hash functions are random oracles [6] and the underlying PKE is secure against plaintext checking attack (PCA), which is inherent in all variants of REACT. These requirements may be too stringent, and it is desirable to have other solutions, better without random oracles.

1.1 Our Contributions

Formal Security Model of PEKS/PKE. Authors of [3] have given a security model on data privacy of PEKS/PKE against adaptive chosen keyword attack and chosen ciphertext attack, however, it is not clear which attack model is posed on keyword privacy. Actually, no concrete discussions were given regarding this point in [3].

Here we show an example with no keyword privacy at all when the attack model of [3] is considered. To see this, one just appends the keyword as a part of the ciphertext of data encryption scheme. It is easily verified that this doesn't violate the data privacy of the PEKS/PKE scheme, as long as the keyword is chosen independent from the encrypted message, but the scheme is not a secure PEKS/PKE scheme since it leaks the information of the keyword. It seems that keyword privacy has been assumed to remain even after compositions by [3].

We thus conclude the previous security model of PEKS/PKE is not complete regarding keyword privacy, however, we emphasize that the two concrete constructions proposed in [3] are secure. In this paper, we formalize the requirement of keyword privacy for secure PEKS/PKE schemes.

Generic Construction of PEKS/PKE. Principally, the design of PEKS/PKE schemes without assuming random oracles is not new, e.g., one first put together PEKS and PKE

components (each without random oracles), then applies non-interactive zero-knowledge proof of "well-formness" for this integration, but this is only theoretical and very inefficient. When speaking of practical schemes, all known constructions have to assume random oracles. It is well-known that a scheme with a security proof in the random oracle model implies no security in the real world [11], therefore, it is desirable to build proofs without random oracles. In this paper, we present such a generic construction.

Interesting Extensions. We also give some applications and extensions of the generic construction. For example, instantiating the above construction with concrete components, one obtains various PEKS/PKE schemes with many good properties. For instance, combining a PEKS scheme from the Gentry IBE [18], and the Kurosawa-Desmedt tag-KEM/DEM [20], we have a PEKS/PKE scheme secure without random oracles. The scheme is quite efficient, which is even comparable to previous constructions with random oracles. In fact, a secure PEKS/PKE is achievable from a variety of assumptions, e.g., from the Waters IBE [28] using asymmetric pairing [10], however, our scheme from the Gentry IBE provides better efficiency.

1.2 Related Work

Public key encryption (PKE) is an important primitive in modern cryptography which guarantees privacy of communications. The standard security notion for PKE is indistinguishability against adaptively chosen ciphertext attack (IND-CCA) [19, 21, 23, 15, 5]. While it is comparatively easy to build CCA-secure schemes assuming random oracles [6], to have CCA-secure schemes such that security reduction without random oracles is not easy. Only theoretical constructions of CCA-secure PKE schemes [21, 15, 24] were known before Cramer and Shoup gave the first practical solution [14]. Another recent approach was proposed by Boneh, Canetti, Halevi and Katz [12, 9] based on identity based encryption (IBE).

An IBE scheme is a public key encryption scheme where any string can be the public key of a user, say, the identity of a user. It was advocated by Shamir [25], whose original intuition was to simplify the management of public key certificates. It has been an open problem to construct full-fledged IBE schemes for many years until [8], when Boneh and Franklin proposed the first IBE scheme based on pairings. Cocks [13] independently proposed another IBE scheme based on decisional quadratic residue problem. Public key encryption with keyword search (PEKS) was proposed in [7]. It was shown that to build a PEKS with exponential keyword space is at least as hard as build an identity based encryption (IBE) [7].

Another security notion for public key encryption is key privacy [4], which captures an adversary's inability to know a receiver's identity from a given ciphertext. For identity based encryption, this was studied under the name "anonymity" [1] (the precise definition postponed to Appendix A). Basically, public key encryption schemes with key privacy provides the functionality of PEKS, however, currently anonymous PKE schemes only provide polynomially bounded keyword space [7, 1], and one may need anonymous IBE schemes for a keyword space of exponential size.

2 Preliminary

In this section, we give some notations and definitions.

Notations. If x is a string, let $|x|$ denotes its length, while if S is a set then $|S|$ denotes its size. If S is a set then $s \leftarrow S$ denotes the operation of picking an element s of S uniformly at random. We write $z \leftarrow \mathcal{A}(x, y, \ldots)$ to indicate that $\mathcal{A}$ is an algorithm with inputs $(x, y, \ldots)$ and an output z. Denote $x||y$ as the string concatenation of x and y. If $k \in \mathbb{N}$, a function $f(k)$ is negligible if $\exists\ k_0 \in \mathbb{N}$, $\forall\ k > k_0$, $f(k) < 1/k^c$, where $c > 0$ is a constant.

2.1 Public Key Encryption

A public key encryption scheme consists of three algorithms $\mathcal{PKE} = (\mathsf{PKEkg}, \mathsf{PKEenc}, \mathsf{PKEdec})$.

PKEkg: a randomized algorithm, taking a security parameter k as input, generates a public key pk and a corresponding secret key sk, denoted as $(pk, sk) \leftarrow \mathsf{PKEkg}(1^k)$.

PKEenc: a possibly randomized algorithm, taking a public key pk, and a plaintext m taken from the message space as input, with internal coin flipping, outputs a ciphertext c, denoted as $c \leftarrow \mathsf{PKEenc}(pk, m)$.

PKEdec: a deterministic algorithm, taking a secret key sk and a ciphertext c as input, outputs the corresponding m, or "$\perp$" (indicating invalid ciphertext), denoted as $m \leftarrow \mathsf{PKEdec}(sk, c)$.

We require a PKE scheme should satisfy the standard correctness requirement, namely for all $(pk, sk) \leftarrow \mathsf{PKEkg}(1^k)$ and all m, $\mathsf{PKEdec}(sk, \mathsf{PKEenc}(pk, m)) = m$.

Data Privacy. We say a public key encryption scheme is (ϵ, q, T)-IND-CCA secure, if the advantage of any adversary $\mathcal{A}$ with at most q queries to a decryption oracle $\mathcal{DO}$, is at most ϵ within time T in the following experiment.

$$\mathrm{Adv}^{\mathsf{ind\text{-}cca}}_{\mathcal{PKE},\mathcal{A}}(k) = |\Pr[(pk, sk) \leftarrow \mathsf{PKEkg}(1^k); (m_0, m_1, s) \leftarrow \mathcal{A}^{\mathcal{DO}}(pk); \\ b \leftarrow \{0,1\}; c^* \leftarrow \mathsf{PKEenc}(pk, m_b); b' \leftarrow \mathcal{A}^{\mathcal{DO}}(c^*, s) : b' = b] - 1/2|$$

where $\mathcal{DO}$ returns the corresponding decryption result on a query on ciphertext c, whereas $\mathcal{A}$ is forbidden to query c^* to $\mathcal{DO}$. We say a PKE scheme is IND-CCA-secure, if for polynomially bounded q and T, ϵ is negligible.

2.2 Tag-KEM/DEM

Shoup introduced key encapsulation mechanism (KEM) and data encapsulation mechanism (DEM) [27], to deal with efficient hybrid encryption. Tag-KEM/DEM is a form of KEM which also takes as input a tag, which was introduced in [2]. A tag-KEM is a generalization of KEM/DEM, and together with a passively secure DEM, it can be easily be extended to threshold settings.

Tag-KEM. Our definition of tag-KEM runs parallel with [2]. A tag-KEM consists of four algorithms $\mathcal{TK} = (\mathsf{TKkg}, \mathsf{TKkey}, \mathsf{TKenc}, \mathsf{TKdec})$.

TKkg: a randomized algorithm, taking a security parameter k as input, generates a public key pk and a secret key sk, denoted as $(pk, sk) \leftarrow \mathsf{TKkg}(1^k)$.

TKkey: a randomized algorithm, taking a public key pk as input, outputs a random session key $dk \in \mathcal{K}_D$, where $\mathcal{K}_D$ is a key space, and internal state information η, denoted as $(dk, \eta) \leftarrow \mathsf{TKkey}(pk)$.

TKenc: a possible randomized algorithm, taking the internal state η and a tag λ as input, encrypts dk (embedded in η) into ψ, denoted as $\psi \leftarrow \mathsf{TKenc}(\eta, \lambda)$.

TKdec: a deterministic algorithm, taking a secret key sk, a ciphertext ψ and a tag λ as input, recovers dk from ψ and λ, denoted as $dk \leftarrow \mathsf{TKdec}(sk, \psi, \lambda)$. We require $\mathsf{TKdec}(\psi, \lambda) = dk$ must hold for any sk, dk, ψ and λ, associated by the above three algorithms. The algorithm outputs "$\perp$" when encountering an error.

Additionally, we require that given a public key pk, a tag λ, and an internal state η for the encryption algorithm TKenc, the session key dk of a tag-KEM should be uniquely decided. We call this property uniqueness of tag-KEM/DEM.

Security Notion. We define the security of tag-KEM as indistinguishability against adaptive chosen ciphertext attack (IND-TK-CCA). We say a tag-KEM scheme is (ϵ, q, T)-IND-TK-CCA secure, if the advantage of any adversary $\mathcal{A}$ with at most q queries to a decryption oracle $\mathcal{DO}$, is at most ϵ within time T in the following experiment.

$$\begin{aligned}\mathrm{Adv}^{\mathsf{ind\text{-}tk\text{-}cca}}_{\mathcal{TK},\mathcal{A}}(k) = |\Pr[(pk, sk) \leftarrow \mathsf{TKkg}(1^k); b \leftarrow \{0,1\};\\ dk_0 \leftarrow \mathcal{K}_D; (\eta, dk_1) \leftarrow \mathsf{TKkey}(pk); (\lambda^*, s) \leftarrow \mathcal{A}^{\mathcal{DO}}(pk, dk_b);\\ \psi^* \leftarrow \mathsf{TKenc}(\eta, \lambda^*); b' \leftarrow \mathcal{A}^{\mathcal{DO}}(\psi^*, s) : b' = b] - 1/2|\end{aligned}$$

where $\mathcal{DO}$ returns corresponding dk on input (ψ, λ), and $\mathcal{A}$ cannot query (ψ^*, λ^*) to $\mathcal{DO}$. We say a tag-KEM is IND-TK-CCA-secure, if for polynomially bounded q and T, ϵ is negligible.

DEM. A DEM consists of two deterministic algorithms, $\mathcal{DEM} = (\mathsf{DEMenc}, \mathsf{DEMdec})$, which is associated with a key space and a plaintext space defined by a security parameter k.

DEMenc: taking a symmetric key $dk \in \mathcal{K}_D$, where $\mathcal{K}_D$ is defined by k and a plaintext $m \in \{0,1\}^*$ as input, outputs a ciphertext χ, denoted as $\chi \leftarrow \mathsf{DEMenc}(dk, m)$.

DEMdec: taking a symmetric key $dk \in \mathcal{K}_D$ and a ciphertext χ as input, outputs a plaintext m, denoted as $m \leftarrow \mathsf{DEMdec}(dk, \chi)$.

We require that for all m and all dk, $\mathsf{DEMdec}(dk, \mathsf{DEMenc}(dk, m)) = m$.

Semantic Security. We only require passive security for DEM. We say a DEM scheme is (ϵ, T)-semantically secure, if the advantage of any adversary $\mathcal{A}$, is at most ϵ within time T in the following experiment.

$$\mathrm{Adv}^{\mathsf{ss}}_{\mathcal{DEM},\mathcal{A}}(k) = |\Pr[b \leftarrow \{0,1\}; dk \leftarrow \mathcal{K}_D; (m_0, m_1, s) \leftarrow \mathcal{A}(1^k); \\ \chi \leftarrow \mathsf{DEMenc}(dk, m_b); b' \leftarrow \mathcal{A}(\chi, s) : b' = b] - 1/2|$$

We say a DEM scheme is (ϵ, T)-semantically secure, if for polynomially bounded T, ϵ is negligible.

2.3 PEKS

A public key encryption with keyword search (PEKS) scheme [7, 1] consists of four algorithms $\mathcal{PEKS} = (\mathsf{PEKSkg}, \mathsf{PEKSenc}, \mathsf{PEKStd}, \mathsf{PEKStest})$.

PEKSkg: a randomized algorithm, taking a security parameter k as input, the probabilistic key generation algorithm generates a public key pk and a secret key sk, denoted as $(pk, sk) \leftarrow \mathsf{PEKSkg}(1^k)$.

PEKSenc: a possibly randomized algorithm, taking a public key pk and a keyword w as input, computes a ciphertext τ, denoted as $\tau \leftarrow \mathsf{PEKSenc}(pk, w)$.

PEKStd: a possibly randomized algorithm, taking a secret key sk and a keyword w as input, computes a trapdoor t_w, denoted as $t_w \leftarrow \mathsf{PEKStd}(sk, t_w)$.

PEKStest: a deterministic algorithm, taking a trapdoor t_w and a ciphertext τ as input, tests whether c encrypts w and outputs a bit b, with 1 meaning "`yes`" and 0 meaning "`no`", denoted as $b \leftarrow \mathsf{PEKStest}(t_w, \tau)$.

Here we assume there is only one receiver (one public key) in the system, and it is straightforward to extend the above definitions to multi-user settings.

Consistency. Several flavors of consistency were discussed in [1], and we only define computational consistency here, since this notion suffices for most practical applications. A PEKS scheme is said to be computationally consistent, if the advantage is negligible for all computationally bounded adversary $\mathcal{A}$ in the following experiment.

$$\mathrm{Adv}^{\mathsf{peks\text{-}consist}}_{\mathcal{PEKS},\mathcal{A}}(k) = \Pr[(pk, sk) \leftarrow \mathsf{PEKSkg}(1^k); (w, w') \leftarrow \mathcal{A}(pk); \\ t_{w'} \leftarrow \mathsf{PEKStd}(sk, w'); \tau^* \leftarrow \mathsf{PEKSenc}(pk, w) : \mathsf{PEKStest}(t_{w'}, c^*) = 1]$$

Keyword Privacy. We define indistinguishability of keywords against adaptive chosen keywords attack (IK-CKA), as considered in [7, 1]. We say a PEKS scheme is (ϵ, q, T)-IK-CKA secure, if the advantage of any adversary $\mathcal{A}$ with at most q queries to a trapdoor generation oracle $\mathcal{TO}$, is at most ϵ within time T in the following experiment.

$$\mathrm{Adv}^{\mathsf{ik\text{-}cka/cca}}_{\mathcal{PEKS},\mathcal{A}}(k) = |\Pr[(pk, sk) \leftarrow \mathsf{PEKSkg}(1^k); (w_0, w_1, s) \leftarrow \mathcal{A}^{\mathcal{TO}}(pk); \\ b \leftarrow \{0,1\}; \tau^* \leftarrow \mathsf{PEKSenc}(pk, w_b); b' \leftarrow \mathcal{A}^{\mathcal{TO}}(\tau^*, s) : b' = b] - 1/2|$$

where $\mathcal{TO}$ is a trapdoor oracle, returns the corresponding trapdoor t_w upon a query on keyword w, whereas $\mathcal{A}$ cannot query w_0 or w_1 to $\mathcal{TO}$. A PEKS scheme is said to be IK-CKA-secure, if for polynomially bounded q and T, ϵ is negligible.

2.4 Bilinear Groups

We review some facts about bilinear groups for future use. Let $\mathbb{G}_1$ and $\mathbb{G}_T$ be two multiplicative cyclic groups of prime order p and g be a generator of $\mathbb{G}_1$. A bilinear map $e : \mathbb{G}_1 \times \mathbb{G}_1 \rightarrow \mathbb{G}_T$ satisfies the following properties: (i) *Bilinearity*: For all $x, y \in \mathbb{G}_1$ and $a, b \in \mathbb{Z}$, $e(x^a, y^b) = e(x, y)^{ab}$. (ii) *Non-degeneracy*: $e(g, g) \neq 1$. (iii) *Computability*: There is an efficient algorithm to compute $e(x, y)$ for any $x, y \in \mathbb{G}_1$.

3 Our Model of PEKS/PKE

In this section, we give the syntax and security definitions for PEKS/PKE schemes. The advantage of our model is that we have notational convenience to define keyword privacy and data privacy.

3.1 PEKS/PKE

We focus on the integration of PEKS/PKE. A PEKS/PKE scheme consists of five algorithms $\mathcal{PEKS}/\mathcal{PKE} = (\mathsf{Kg}, \mathsf{Enc}, \mathsf{Dec}, \mathsf{Td}, \mathsf{Test})$.

Kg: a randomized algorithm, taking a security parameter k as input, generates a public key pk and a secret key sk, denoted as $(pk, sk) \leftarrow \mathsf{Kg}(1^k)$.

Enc: a possibly randomized algorithm, taking a public key pk, a keyword w and a plaintext m as input, outputs a PEKS/PKE ciphertext c, denoted as $c \leftarrow \mathsf{Enc}(pk, w, m)$.

Dec: a deterministic algorithm, taking a secret key sk and a PEKS/PKE ciphertext c, outputs the decryption result m (or "$\perp$" if c is invalid). We denote this as $m \leftarrow \mathsf{Dec}(sk, c)$.

Td: a possibly randomized algorithm, taking a secret key sk and a keyword w as input, computes a trapdoor t_w for keyword w, denoted as $t_w \leftarrow \mathsf{Td}(sk, w)$.

Test: a deterministic algorithm, tests whether a given PEKS/PKE ciphertext c encrypts keyword w, and outputs a bit b, with 1 meaning "`yes`" and 0 meaning "`no`", denoted as $b \leftarrow \mathsf{Test}(t_w, c)$.

Our model simplifies the one in [3]. In the encryption algorithm Enc, we don't explicitly require a tag in the ciphertext, since otherwise the security definition should additionally consider the tag. We remark the model is general enough because the tag can be regarded as a part of the ciphertext.

Consistency. A PEKS/PKE scheme is said to be computationally consistent, if the advantage is negligible for all computationally bounded adversary $\mathcal{A}$ in the following experiment.

$$\mathrm{Adv}^{\mathsf{peks/pke\text{-}consist}}_{\mathcal{PEKS}/\mathcal{PKE},\mathcal{A}}(k) = \Pr[(pk, sk) \leftarrow \mathsf{Kg}(1^k); (w, w', m) \leftarrow \mathcal{A}(pk); \\ t_{w'} \leftarrow \mathsf{Td}(sk, w'); c^* \leftarrow \mathsf{Enc}(pk, m, w) : \mathsf{Test}(t_{w'}, c^*) = 1]$$

3.2 Security Notions

We consider two security requirements, keyword privacy, namely, indistinguishability of keywords against adaptive chosen keyword attack and chosen ciphertext attack (IK-CKA/CCA), and data privacy, namely, indistinguishability of ciphertexts against adaptive chosen keyword attack and chosen ciphertext attack (IND-CKA/CCA). Note that in a PEKS/PKE scheme, PEKS and PKE are both regarded as components of the whole system.

Principally, the adversary is given two oracles, a trapdoor generation oracle $\mathcal{TO}$, that on a keyword w, generates the corresponding trapdoor t_w and a decryption oracle that on a ciphertext c, returns the corresponding plaintext m.

Keyword Privacy. We say a PEKS/PKE scheme is (ϵ, q_t, q_d, T)-IK-CKA/CCA secure, if the advantage of any adversary $\mathcal{A}$ with at most q_t queries to a trapdoor generation oracle $\mathcal{TO}$, at most q_d queries to a decryption oracle $\mathcal{DO}$, is at most ϵ within time T in the following experiment.

$$\mathrm{Adv}^{\mathsf{ik\text{-}cka/cca}}_{\mathcal{PEKS}/\mathcal{PKE},\mathcal{A}}(k) = |\Pr[(pk, sk) \leftarrow \mathsf{Kg}(1^k); (w_0, w_1, m, s) \leftarrow \mathcal{A}^{\mathcal{TO},\mathcal{DO}}(pk); \\ b \leftarrow \{0,1\}; c^* \leftarrow \mathsf{Enc}(pk, m, w_b); b' \leftarrow \mathcal{A}^{\mathcal{TO},\mathcal{DO}}(c^*, s) : b' = b] - 1/2|$$

where $\mathcal{TO}$ is a trapdoor oracle, upon a query on keyword w returns the corresponding trapdoor t_w, and $\mathcal{DO}$ is a decryption oracle, upon a query on ciphertext c returns the corresponding plaintext, whereas $\mathcal{A}$ cannot query w_0 or w_1 to $\mathcal{TO}$. We say a PEKS/PKE is IK-CKA/CCA-secure, if for polynomially bounded q_t, q_d and T, ϵ is negligible.

Data Privacy. We say a PEKS/PKE scheme is (ϵ, q_t, q_d, T)-IK-CKA/CCA secure, if the advantage of any adversary $\mathcal{A}$ with at most q_t queries to a trapdoor generation oracle $\mathcal{TO}$, at most q_d queries to a decryption oracle $\mathcal{DO}$, is at most ϵ within time T in the following experiment.

$$\mathrm{Adv}^{\mathsf{ind\text{-}cka/cca}}_{\mathcal{PEKS}/\mathcal{PKE},\mathcal{A}}(k) = |\Pr[(pk, sk) \leftarrow \mathsf{Kg}(1^k); (w, m_0, m_1, s) \leftarrow \mathcal{A}^{\mathcal{TO},\mathcal{DO}}(pk); \\ b \leftarrow \{0,1\}; c^* \leftarrow \mathsf{Enc}(pk, w, m_b); b' \leftarrow \mathcal{A}^{\mathcal{TO},\mathcal{DO}}(c^*, s) : b' = b] - 1/2|$$

where $\mathcal{TO}$ is a trapdoor oracle, upon a query on keyword w returns the corresponding trapdoor t_w, and $\mathcal{DO}$ is a decryption oracle, upon a query on ciphertext c returns the corresponding plaintext, whereas $\mathcal{A}$ cannot query c^* to $\mathcal{DO}$. We say a PEKS/PKE is IK-CKA/CCA-secure, if for polynomially bounded q_t, q_d and T, ϵ is negligible.

4 A Generic Construction of Secure PEKS/PKE

We need two ingredients for our generic construction, one is an IK-CKA secure PEKS scheme, and the other is an IND-TK-CCA secure tag-KEM/DEM. The main idea is to regard the ciphertext of PEKS as a proportion of the tag for tag-KEM/DEM. The tag-KEM/DEM framework covers almost all the known PEK schemes (see [2] for details), and can be built flexibly from a variety of assumptions. Since a tag is a natural component for a tag-KEM/DEM scheme, the structures of both parts persist. Another advantage of our methodology is that a tag-KEM/DEM can be easily extended to threshold settings, since the DEM only require passive security. We give our construction in Figure 1.

$\mathsf{Kg}(1^k)$	$\mathsf{Dec}(sk, c)$
$(pk_1, sk_1) \leftarrow \mathsf{PEKSkg}(1^k)$;	$sk = (sk_1, sk_2)$;
$(pk_2, sk_2) \leftarrow \mathsf{TKkg}(1^k)$;	$c = (\tau, \psi, \chi)$;
$pk = (pk_1, pk_2)$;	$dk \leftarrow \mathsf{TKdec}(sk_2, \psi, \tau\|\|\chi)$;
$sk = (sk_1, sk_2)$;	$m \leftarrow \mathsf{DEMdec}(dk, \chi)$;
return (pk, sk);	return m;
$\mathsf{Enc}(pk, w, m)$	$\mathsf{Td}(sk, w)$
$pk = (pk_1, pk_2)$;	$sk = (sk_1, sk_2)$;
$\tau \leftarrow \mathsf{PEKSenc}(pk_1, w)$;	$t_w \leftarrow \mathsf{PEKStd}(sk_1, w)$;
$(dk, \eta) \leftarrow \mathsf{TKkey}(pk_2)$;	return t_w;
$\chi \leftarrow \mathsf{DEMenc}(dk, m)$;	
$\lambda \leftarrow (\tau\|\|\chi)$;	$\mathsf{Test}(t_w, c)$
$\psi \leftarrow \mathsf{TKenc}(\eta, \lambda)$;	$c = (\tau, \psi, \chi)$;
$c \leftarrow (\tau, \psi, \chi)$;	$b \leftarrow \mathsf{PEKStest}(t_w, \tau)$;
return c;	return b;

For each algorithm of PEKS/PKE, we require it should terminate and return "$\perp$" (denoting "abnormal termination"), if any of its sub-algorithms terminates abnormally.

Fig. 1. Generic Construction of PEKS/PKE

It is easily verified that if both the PEKS and PKE used in the construction are consistent, the resulting PEKS/PKE is consistent. We focus on the keyword privacy and data privacy of the construction.

Theorem 1. *The construction of PEKS/PKS shown in Figure 1 is* IK-CKA/CCA*-secure and* IND-CKA/CCA *secure, provided that the underlying PEKS scheme is* IK-CKA*-secure, the tag-KEM scheme is* IK-TK-CCA*-secure and the DEM scheme is semantically secure.*

Intuitions. First, notice that a PEKS scheme aims at providing keywords privacy, while "naturally", ciphertext χ of the tag-KEM/DEM will not leak information of the keywords. By uniqueness property of tag-KEM, ψ is determined once the public key pk_2, the tag $\lambda = (\psi||\chi)$ and internal state η of the tag-KEM are determined, and will be independent from a keyword w if τ doesn't leak information on w. Moreover, because dk only depends on pk_2 and internal random coin-flipping of TKkey, and the algorithm DEMenc is deterministic, we have χ is also

independent from w. Finally, we conclude, if τ doesn't leak information on w, neither will ψ and χ. On the other hand, from our construction, τ does not depend on m, thus leaks no information on m. Additionally, taking τ as a part of the tag provides integrity guarantee also for τ, i.e., any adversary cannot gain advantage in obtaining knowledge on a plaintext m by modifying this part. Otherwise, the adversary breaks indistinguishability of session key for the tag-KEM. We elaborate the above discussions in two lemmas.

Lemma 1. *The construction shown in Figure 1 is* $(\epsilon_K + \epsilon_D, q_t, q_d, T_K + T_D)$-IND-CKA/CCA *secure, provided that the tag-KEM scheme is* (ϵ_K, q_d, T_K)-IK-TK-CCA*-secure and the DEM scheme is* (ϵ_D, T_D)*-semantically secure.*

Proof. First, notice that τ is independent from m_b from the encryption algorithm. Assume there is an IND-CKA/CCA adversary $\mathcal{A}$, we can build an adversary $\mathcal{B}$ against IND-TK-CKA/CCA of tag-KEM or semantic security of DEM. $\mathcal{A}$ flips a fair coin, and runs in either of the following modes.

Mode 0 (Adversary against tag-KEM/DEM): $\mathcal{A}$ generates a pair of public/secret keys for PEKS. Since $\mathcal{A}$ has the secret key, trapdoor queries are handled perfectly. Decryption queries are forwarded to $\mathcal{A}$'s own decryption oracle and are also handled perfectly. For challenge, after receiving a pair of plaintext (m_0, m_1) and a keyword w from $\mathcal{B}$ and dk_b from its own challenger, $\mathcal{A}$ chooses uniformly $\beta \leftarrow \{0,1\}$ and computes $\tau = \mathsf{PEKSenc}(pk_1, w)$, and computes $\chi \leftarrow \mathsf{DEMenc}(dk_b, m_\beta)$, where $b \leftarrow \{0,1\}$. $\mathcal{A}$ then sets $\lambda = (\tau, \chi)$ as the tag to its challenger. After receiving its challenge ψ, $\mathcal{A}$ gives $\mathcal{B}$ the challenge $c = (\tau, \psi, \chi)$. When $\mathcal{B}$ outputs a guess on β, $\mathcal{A}$ checks whether this equals to β. $\mathcal{A}$ outputs 1 if yes and otherwise, 0. It is easily verified when dk_b is the real session key, then the challenge for $\mathcal{B}$ is valid and $\mathcal{A}$ will succeed at least the probability as $\mathcal{B}$. If dk_b is a random session key, β is perfectly hiding from $\mathcal{B}$, and $\mathcal{B}$'s probability in guessing b is exactly 1/2. Summarize above discussions, we have the success probability of $\mathcal{A}$ is at least that of $\mathcal{B}$.

Mode 1 (Adversary against DEM): For setup, $\mathcal{A}$ generates a pair of public/secret keys for PEKS and tag-KEM. Trapdoor oracle queries and decryption oracle queries can perfectly simulated, since $\mathcal{A}$ has the secret keys. After receiving a pair of plaintext (m_0, m_1) and a keyword w from $\mathcal{B}$, $\mathcal{A}$ outputs (m_0, m_1) to its challenger. After obtains a challenge χ^* from its challenger, $\mathcal{A}$ computes a ciphertext τ of PEKS and a ciphertext ψ with some random $dk \in \mathcal{K}_D$ from a tag $\lambda = (\tau || \psi)$. After $\mathcal{B}$ stops and outputs a guess, $\mathcal{A}$ also outputs the same bit. Since neither τ or ψ contains information on b, $\mathcal{B}$ can only gain advantage by inferring b from χ. One case to mention is that if $\mathcal{B}$ is able to distinguish ψ is not a valid ciphertext, then the result of $\mathcal{B}$ cannot be utilized and $\mathcal{A}$ should abort. However, in this case, we can construct an attack against the tag-KEM. However, according to our assumption, this happens at most ϵ_K. We then have in this case $\mathcal{A}$'s success probability breaking the DEM is at least that of $\mathcal{B}$ plus ϵ_K.

Summarizing the above two cases, we see the advantage of $\mathcal{A}$ is upper-bounded by $\epsilon_K + \epsilon_D$ and the queries and the running time of $\mathcal{A}$ are exactly the same as the claim. □

Lemma 2. *The construction shown in Figure 1 is* $(\epsilon_P, q_t, q_d, T_P)$-IK-CKA/CCA-*secure, provided that the PEKS scheme is* (ϵ_P, q_t, T_P)-IK-CKA-*secure.*

Proof Sketch. The proof for the lemma is quite simple and we only give the sketch. From the algorithms shown in Figure 1, only τ depends on a keyword w, since the tag-KEM/DEM does not even take w as an input. a PEKS adversary $\mathcal{A}$ generates the public/secret keys (pk_2, pk_2) for the tag-KEM/DEM scheme and sets the public key as $pk = (pk_1, pk_2)$, where pk_1 is the public key of its target PEKS scheme. Since $\mathcal{A}$ knows the the secret key of the tag-KEM scheme, all decryption queries from a PEKS/PKE adversary $\mathcal{B}$ can be answered perfectly. For $\mathcal{B}$'s challenge query, $\mathcal{A}$ forwards (w_0, w_1) to its own trapdoor oracle and extracts τ^* from its challenge as the challenge for $\mathcal{B}$, it is easy to verify that τ^* is a valid challenge for $\mathcal{B}$. Then $\mathcal{A}$'s success probability is exactly the same as $\mathcal{B}$. This proves our claim. □

Theorem 1 follows Lemma 1 and Lemma 2 naturally.

5 Applications and Extensions

In this section, we give some possible extensions of our generic construction. In particular, we show a concrete PEKS/PKE scheme, whose security can be proven without random oracles.

5.1 A Concrete Instantiation without Random Oracles

We instantiate our generic construction with an anonymous IBE by Gentry [18], and the Kurosawa-Desmedt tag-KEM/DEM [20]. The resulting PEKS/PKE is secure without random oracles. The scheme is given in Figure 2.

The notion of anonymous IBE is reviewed in Appendix A. The consistency condition is easy verified to be met since it is a straightforward instantiation of BDOP construction of PEKS (based Gentry IBE) and a secure tag-KEM/DEM scheme.

Theorem 2. *The PEKS/PKE scheme shown in Figure 2 is* IND-IK-CKA/CCA-*secure, provided that the Kurosawa-Desmedt tag-KEM/DEM is secure and the Gentry IBE is anonymous.*

The above proof is easily derived from Theorem 1 and known results [2, 18].

Performance. Though our scheme relies on the DADHE assumption that seems strong, however, the scheme is quite efficient in of key size and computation cost. Note that the previous schemes have to adopt a large key size to compensate security loss due to loose security reductions. Moreover, our scheme needs no Map-to-Point computations [8], and it can be further optimized with a trick mentioned in [17] and pre-computations. Consider all these and the fact that our scheme is without random oracles, we conclude our scheme is efficient.

PEKS/PKE without MACs. Our instantiation of tag-KEM/DEM is based on Kurosawa-Desmedt, where a MAC is inevitable. One can use other tag-KEM schemes, e.g., Cramer-Shoup tag-KEM [14, 2], or OAEP+ [26, 2], such that the MAC is not explicitly needed.

$\mathsf{Kg}(1^k)$
$g, h \leftarrow \mathbb{G}_1$;
$z \leftarrow e(g_1, g_2)$;
$\alpha \leftarrow \mathbb{Z}_p$;
$g_1 \leftarrow g^{\alpha}$;
$pk_1 \leftarrow (g, g_1, h)$;
$sk_1 \leftarrow \alpha$;
$z_1, z_2 \leftarrow \mathbb{G}_2$;
$x_1, x_2, y_1, y_2 \leftarrow \mathbb{Z}_p$;
$c \leftarrow z_1^{x_1} z_2^{x_2}$;
$d \leftarrow z_1^{y_1} z_2^{y_2}$;
$pk_2 \leftarrow (c, d, G, F, H)$;
$sk_2 \leftarrow (x_1, x_2, y_1, y_2)$;
$pk \leftarrow (pk_1, pk_2)$;
$sk \leftarrow (sk_1, sk_2)$;
return (pk, sk);

$\mathsf{Dec}(sk, c)$
$c = (c_1, c_2)$, where $c_1 = (u_1, u_2, u_3)$ and $c_2 = (v_1, v_2, v_3)$;
$f \leftarrow v_1^{x_1 + y_1 H(v_1, v_2)} v_2^{x_1 + y_1 H(v_1, v_2)}$;
$(K_1, K_2) \leftarrow F(K)$;
if $v_4 \neq \mathsf{Mac}(K_1, v_3 || c_1)$;
return "$\perp$";
$m \leftarrow v_3 \oplus G(K_2)$;
return m;

$\mathsf{Enc}(pk, w, m)$
$s_1, s_2 \leftarrow \mathbb{Z}_p$;
$u_1 \leftarrow g_1^{s_1} g^{-s_1 w}$;
$u_2 \leftarrow e(g, g)^{s_1}$;
$u_3 \leftarrow H(e(g, h)^{-s_1})$;
$c_1 \leftarrow (u_1, u_2, u_3)$;
$v_1 \leftarrow z_1^{s_2}$;
$v_2 \leftarrow z_2^{s_2}$;
$K \leftarrow c^{s_2} d^{s_2 H(u_1, u_2)}$;
$(K_1, K_2) \leftarrow F(K)$;
$v_3 \leftarrow G(K_2) \oplus m$;
$v_4 \leftarrow \mathsf{Mac}(K_2, v_3 || c_1)$;
$c \leftarrow (c_1, c_2)$;
return c;

$\mathsf{Td}(sk, w)$
$r_w \leftarrow \mathbb{Z}_p$;
$d_w \leftarrow h g^{r_w}$;
$t_w \leftarrow (r_w, d_w)$;
return t_w;

$\mathsf{Test}(t_w, c)$
$c = (c_1, c_2)$, where $c_1 = (u_1, u_2, u_3)$ and $c_2 = (v_1, v_2, v_3)$;
if $u_3 = H(e(u_1, d_w) u_2^{r_w})$;
return 1;
otherwise
return 0;

‡ Let $e : \mathbb{G}_1 \times \mathbb{G}_1 \rightarrow \mathbb{G}_2$ be a bilinear group pair with prime order p. $\mathcal{MAC} = (\mathsf{Mac}, \mathsf{Vrfy})$ is a message authentication code. F is a key derivition function (KDF) [27], G is a pseudorandom generator and H is a collision resistant hash function. Without further descriptions, we simply assume the input domain and output domain match.

Fig. 2. A Concrete Instantiation without Random Oracles

5.2 Other Extensions

PEKS/PKE from General Assumptions. Our generic construction has implicitly assumed an exponential keyword space, thus the constructions of PEKS is restricted to anonymous IBE schemes. In fact, it is possible to base the PEKS on general assumptions, e.g., existence of trapdoor one-way functions, with relaxation to a polynomial keywords space [7].

Randomness Reuse. We have required that the encryption algorithms of PEKS and PKE choose independent randomness in our generic construction, however, one can actually reuse the randomness without harming the security of the scheme. The technique is standard, and the details are omitted here due to space limitation.

PEKS/PKE with Threshold Decryption. Since a tag-KEM can be easily extended to the threshold setting, it is natural to follow the strategy of [2] to have non-interactive threshold decryption for PEKS/PKE.

Multi-Keyword and Multi-Receiver PEKS/PKE. PEKS/PKE with multi-keywords and multi-receivers have been considered in [3]. We remark the same problem of keyword privacy occurs when considering the ciphertext of PKE leaks information on keyword. It is not hard to generate all our above discussions to these settings. The techniques are quite standard, and again, we omit the details here.

Acknowledgement

We thank the anonymous referees of CANS'07 for many helpful comments.

References

1. M. Abdalla, M. Bellare, D. Catalano, E. Kiltz, T. Kohno, T. Lange, J. Malone-Lee, G. Neven, P. Pallier, and H. Shi. Searchable Encryptino Revisited: Consistnecy Properties, Relation to Anonymous IBE, and Extensions. In *Crypto'05*, volume 3621 of *LNCS*, pages 205–222. Springer, 2005.
2. M. Abe, R. Gennaro, and K. Kurosawa. Tag-KEM/DEM: A New Framework for Hybrid Encryption. Cryptology ePrint Archive, http://eprint.iacr.org/2005/027/, 2005.
3. J. Baek, R. Safavi-Naini, and W. Susilo. On the Integration of Public Key Data Encryption and Public Key Encryption with Keyword Search. In *ISC'06*, volume 4176 of *LNCS*, pages 217–232. Springer, 2006.
4. M. Bellare, A. Boldyreva, A. Desai, and D. Pointcheval. Key-Privacy in Public-Key Encryption. In *Asiacrypt'01*, volume 2248 of *LNCS*, pages 566–582. Springer, 2001.
5. M. Bellare, A. Desai, D. Pointcheval, and P. Rogaway. Relations among notions of security for public key encryption schemes. In *Crypto'98*, volume 1462 of *LNCS*, pages 26–45. Springer, 1998.
6. M. Bellare and P. Rogaway. Random oracles are practical: A paradigm for designing efficient protocols. In *ACM CCS'93*, pages 62–73. ACM Press, 1993.
7. D. Boneh, G. Di Crescenzo, R. Ostrovsky, and G. Persiano. Public Key Encryption with Keyword Search. In *Eurocrypt'04*, volume 3027 of *LNCS*, pages 506–522, 2004.
8. D. Boneh and M. Franklin. Identity-Based Encryption from the Weil Pairing. In *CRYPTO'01*, volume 2139 of *LNCS*, pages 213–229. Springer, 2001.
9. D. Boneh and J. Katz. Improved Efficiency for CCA-Secure Cryptosystems Built Using Identity-Based Encryption. In *CT-RSA'05*, volume 3376, pages 87–103. Springer, 2005.
10. X. Boyen and B. Waters. Anonymous Hierarchical Identity Based Encryption (without Random Oracles). In *CRYPTO'06*, volume 4117 of *LNCS*, pages 290–307, 2006.
11. R. Canetti, O. Goldreich, and S. Halevi. The Random Oracle Methodology, Revisited. In *STOC'98*, pages 557 – 594. ACM, 1998. Full availabe at http://eprint.iacr.org/1998/011.pdf.
12. R. Canetti, S. Halevi, and J. Katz. Chosen-Ciphertext Security from Identity-Based Encryption. In *EUROCRYPT'04*, volume 3027 of *LNCS*, pages 207–222. Springer, 2004.
13. C. Cocks. An Identity Based Encryption Scheme Based on Quadratic Residues. In *the 8th IMA international INPROCEEDINGS on cryptography and coding*, volume 2260 of *LNCS*, pages 360–363. Springer, 2001.
14. R. Cramer and V. Shoup. A Practical Public Key Cryptosystem Provably Secure against Adaptive Chosen Ciphertext Attack. In *Crypto'98*, volume 1462 of *LNCS*, pages 13–25. Springer, 1998.
15. D. Dolev, C. Dwork, and M. Naor. Non-Malleable Cryptography. In *STOC'91*, pages 542–552. ACM, 1991.
16. T. ElGamal. A Public Key Cryptosystem and a Signature Scheme Based on Discrete Logarithms. *IEE Transactions on Information Theory*, 31(4):469–472, 1985.
17. R. Gennaro and V. Shoup. A Note on An Encryption Scheme of Kurosawa and Desmedt. Eprint Report 2004/194. Available at http://eprint.iacr.org/2004/194, 2004.
18. C. Gentry. Practical Identity-Based Encryption Without Random Oracles. In *EUROCRYPT'06*, volume 4004 of *LNCS*, pages 445–464. Springer, 2006.

19. S. Goldwasser and S. Micali. Probabilistic Encryption. *Journal of Computer and System Sciences*, 28(2):270–299, 1984.
20. K. Kurosawa and Y. Desmedt. A New Paradigm of Hybrid Encryption Scheme. In *CRYPTO'04*, volume 3152 of *LNCS*, pages 426–442. Springer, 2004.
21. M. Naor and M. Yung. Public-key Cryptosystems Provably Secure against Chosen Ciphertext Attacks. In *STOC'90*, pages 427–437. ACM, 1990.
22. T. Okamoto and D. Pointcheval. REACT: Rapid Enhanced-security Asymmetric Cryptosystem Transform. In *CT-RSA'01*, volume 159-175 of *LNCS*, pages 159–175. Springer, 2001.
23. C. Rackoff and D.R. Simon. Non-Interactive Zero-Knowledge Proof of Knowledge and Chosen Ciphertext Attack. In *CRYPTO'91*, volume 576 of *LNCS*, pages 433–444. Springer, 1991.
24. A. Sahai. Non-Malleable Non-Interactive Zero Knowledge and Adaptive Chosen-Ciphertext Security. In *FOCS'99*, pages 543–553. IEEE Computer Society, 1999.
25. A. Shamir. Identity-Based Cryptosystems and Signature Schemes. In *CRYPTO'84*, volume 196 of *LNCS*, pages 47–53. Springer, 1984.
26. V. Shoup. OAEP Reconsidered. In *Crypto'01*, volume 2139 of *LNCS*, pages 239–259. Springer, 2001.
27. V. Shoup. ISO 18033-2: An Emerging Standard for Public-Key Encryption (committee draft). Available at http://shoup.net/iso/, June 2004.
28. B. Waters. Efficient Identity-Based Encryption Without Random Oracles. In *EUROCRYPT'05*, volume 3494 of *LNCS*, pages 114–127. Springer, 2005.

A Identity Based Encryption

An identity based encryption (IBE) can be regarded as a special public key encryption, where the receiver's public key can be any string. Compared with traditional public key encryption, an IBE scheme is equipped with an additional extraction algorithm, with a master secret key and an identity as input, outputs a secret key that is capable to decrypt ciphertext corresponding to this identity. An IBE scheme consists of four algorithms $\mathcal{IBE} = (\mathsf{IBEkg}, \mathsf{IBEext}, \mathsf{IBEenc}, \mathsf{IBEdec})$.

IBEkg: a randomized algorithm, taking a security parameter k as the input, outputs a public parameter $params$ and a master secret key msk, denoted as $(params, msk) \leftarrow \mathsf{TBEkg}(1^k)$.

IBEext: a possibly randomized algorithm, takes inputs of $params$, msk and an identity id, outputs a secret key sk_{id} for id, denoted as $sk_{id} \leftarrow \mathsf{IBEext}(params, msk, id)$, in brief $sk_{id} \leftarrow \mathsf{IBEext}(msk, id)$.

IBEext: a possibly randomized algorithm, taking $params$, an identity id and a plaintext m taken from the message space as input, with internal coin flipping r, outputs a ciphertext c, which is denoted as $c \leftarrow \mathsf{IBEenc}(params, id, m, r)$, in brief $c \leftarrow \mathsf{IBEenc}(params, id, m)$.

IBEdec: a deterministic algorithm, taking a secret key sk_{id}, an identity id and a ciphertext c as input, outputs a plaintext m, or a special symbol "$\perp$", which is denoted $m \leftarrow \mathsf{IBEdec}(sk_{id}, id, c)$.

We require for all $(params, msk) \leftarrow \mathsf{IBEkg}(1^k)$, $sk_{id} \leftarrow \mathsf{IBEext}(msk, id)$ and all m, we have $\mathsf{IBEdec}(sk_{id}, id, \mathsf{IBEenc}(params, id, m)) = m$.

Anonymity. We consider anonymity of receiver against adaptively chosen-ID and chosen plaintext attack (AONT-ID-CPA) [1]. We say an identity based encryption is (ϵ, q, T)-IND-sID-CPA-secure if the advantage of any adversary $\mathcal{A}$ is at most ϵ, with access q times to an extraction oracle $\mathcal{EO}$ within time T in the following experiment.

$$\begin{aligned}\mathrm{Adv}^{\mathsf{aont\text{-}id\text{-}cpa}}_{\mathcal{IBE},\mathcal{A}}(k) = \Pr[&(params, msk) \leftarrow \mathsf{IBEkg}(1^k);\\ &(id_0, id_1, m, s) \leftarrow \mathcal{A}^{\mathcal{EO}}(params); b \leftarrow \{0,1\};\\ &c^* \leftarrow \mathsf{IBEenc}(params, id_b, m); b' \leftarrow \mathcal{A}^{\mathcal{EO}}(c^*, s) : b' = b] - 1/2\end{aligned}$$

where $\mathcal{EO}$ returns the corresponding secret key on a query on identity id, whereas $\mathcal{A}$ is forbidden to query (id_0, id_1) at $\mathcal{EO}$. We say an IBE is AONT-ID-CPA-Secure, if for polynomially bounded q and T, ϵ is negligible.