Integrated PKE and PEKS - Stronger Security Notions and New Constructions

Yu Chen[1], Jiang Zhang[2], Zhenfeng Zhang[1], Dongdai Lin[1]

[1]State Key Laboratory of Information Security (SKLOIS),
Institute of Information Engineering, Chinese Academy of Sciences, Beijing 100095, China
{chenyu,zhangzhenfeng,ddlin}@iie.ac.cn
[2]Institute of Software, Chinese Academy of Sciences, Beijing 100190, China
{zhangjiang@is.iscas.ac.cn}@is.iscas.ac.cn

Abstract. In this paper we investigate the security for integrated public-key encryption (PKE) and public-key encryption with keyword search (PEKS) schemes. We observe that the security notions for integrated PKE and PEKS schemes considered in the existing literature are not strong enough to capture practical attacks, thus define a new notion named joint CCA-security which is shown to be stronger than the previous ones. We also propose two simple and efficient constructions of jointly CCA-secure integrated PKE and PEKS schemes from anonymous (hierarchical) identity-based encryption schemes. Besides, we review the consistency for PEKS schemes and improve previous results.

Key words: PKE-PEKS, joint security, consistency, collision-freeness, robustness

1 Introduction

Public-key encryption with keyword search (PEKS) [BCOP04] is an useful primitive which allows one to delegate to a third party the capability of "searching on public-key encrypted data" without impacting the privacy. It has found numerous applications in various fields such as the design of spam filter and searchable cloud storage. We illustrate this mechanism more precisely with the following example. Let (pk_A, sk_A) be Alice's public/secret key pair. Alice wishes her email gateway can identify if the incoming email contains a particular keyword w. To do so, Alice sends the gateway a token t_w for w. When Bob sends Alice an email m labeled by a keyword w privately, he first encrypts m under pk_A using a standard PKE scheme, then "encrypts" w using a PEKS scheme. The final encrypted email is of the form $\mathsf{PKE}(pk_A, m)||\mathsf{PEKS}(pk_A, w)$. Upon receiving an encrypted email, the email gateway uses its token t_w to test if the email is labeled by w. Except the testing result, the gateway learns nothing else about this email and the keywords.

Chosen-Ciphertext Security for PEKS. The security requirement for PEKS schemes is inherently limited to semantic security, namely the indistinguishability against chosen-plaintext attack (IND-PEKS-CPA). In the PEKS setting, the term "plaintext" in fact means "keyword". We will slightly abuse this abbreviation where it is clear from the context. In the IND-PEKS-CPA security experiment defined in [BCOP04], the adversary is only given access to a token oracle. Obviously, this security experiment fails to capture the attacks from more powerful adversaries, say, who can inject packets into the network and observe actions taken based on them. For instance, in practice, an adversary can send a PEKS ciphertext s to a gateway holding token t_w, then learns useful information from the routing results. Thus it is necessary to consider a stronger security notion for PEKS, namely the IND-PEKS-CCA security [ABN10]. In the corresponding security experiment, beside the token oracle, the adversary is also given access to a test oracle which can decides if a PEKS ciphertext s encrypts a keyword w.

Integrated PKE and PEKS. PEKS is introduced to provide searchable functionality for PKE. As pointed out by [BCOP04, BSNS06, ZI07, ABN10], due to lack of data retrieval function, a PEKS scheme is only meaningful when coupled with a PKE scheme. For this reason, a more powerful primitive named integrated PKE and PEKS scheme[1] is suggested, which combines PEKS with PKE to encrypt non-searchable but recoverable message. Naturally, it is of practical interest to consider the PKE and PEKS in the joint sense rather than separately. From here on, we refer to the integrated PKE and PEKS scheme as PKE-PEKS scheme, and denote the integrated PKE-PEKS ciphertext of (m, w) by $c||s$, where c is the PKE ciphertext of message m and s is the PEKS ciphertext of keyword w. Loosely speaking, we say a PKE-PEKS scheme is jointly secure if it processes data privacy (the confidentiality of m) and keyword privacy (the confidentiality of w) simultaneously. However, building a jointly secure PKE-PEKS scheme in strong sense is not an easy task. As envisioned by [BSNS06, ZI07, ABN10], the straightforward approach of concatenating ciphertexts of PKE and PEKS works fine for joint CPA-security, but is insufficient for joint CCA-security. Note that for PKE, chosen-ciphertext security is closely related to the notion of non-malleability [DDN00], which captures an adversary's inability that given a ciphertext c to output a different ciphertext c' such that the plaintexts m, m' underlying these two ciphertexts are "meaningfully related". In what follows, we will use the notion of non-malleability to analyze previous constructions and illustrate our approaches.

1.1 Known Solutions and Their Limitations

Baek *et al.* [BSNS06] first considered the problem of combining PKE and PEKS in a secure manner. They also gave a joint security notion for PKE-PEKS scheme. However, as pointed out in [ZI07], their notion is not complete since it only considers the data privacy, but neglects the keyword privacy. Besides, their notion is restricted for that in the challenge stage the target keyword w must differ from the target messages m_0 and m_1. In the same paper, two solutions meeting their security notion are proposed in the random oracle model. One is a concrete scheme based on the REACT version of ElGamal [OP01] (serve as the PKE component) and BDOP-PEKS [BCOP04] (serve as the PEKS component). The main idea is exploring the randomness reuse technique to bind the PKE ciphertext and the PEKS ciphertext together and using a hash function (act as a MAC) to protect the integrity of the overall ciphertext. The other is a generic construction based on a OW-PCA (one-wayness against plaintext checking attack) secure PKE scheme and a PEKS scheme equipped with the same public/secret key pair as the PKE scheme. These requirements may be too stringent for a generic construction.

Be aware of the incompleteness of Baek *et al.*'s security notion, Zhang and Imai [ZI07] gave another security notion which captures both data privacy (IND-PKE-CCA) and keyword privacy (IND-PEKS-CPA). They also gave a generic construction based on two independent primitives: a tag-based CCA-secure PKE scheme and a CPA-secure PEKS scheme. To bind them together, the PEKS ciphertext s is used as a part of the tag for the PKE scheme. The public/secret key pair of the resulting PKE-PEKS scheme is a direct composition of the two corresponding public/secret key pairs belong to the underlying PKE scheme and PEKS scheme, respectively. However, their construction merely performs unidirectional binding, not bidirectional binding. Though the PKE-PEKS ciphertext is non-malleable with respect to the PKE ciphertext, it is still malleable with respect to the PEKS ciphertext. This explains why their PKE-PEKS construction [ZI07] has IND-PKE-CCA security for data privacy but only IND-PEKS-CPA security for keyword privacy. Besides, in order to achieve joint security (the main principle is to reduce mutual dependency), their PKE-PEKS construction resorts to *key separation strategy*,

[1] Integrated PKE and PEKS scheme is also known as combined PKE/PEKS scheme.

i.e., using different keys for different cryptographic operations. As a result, the resulting PKE-PEKS construction suffers from double key size, which could be critical in resource constrained applications.

Abdalla *et al.* [ABN10] noted that the security notions for PKE-PEKS schemes considered in [BSNS06, ZI07] are not strong enough since the adversary is not given access to a test oracle. They introduced a new security notion that the adversary can access to both a decryption oracle and a test oracle. We remark that this security notion is still not strong enough since the adversary might also have access to a token oracle. They also sketched how to construct a PKE-PEKS scheme satisfying their security notion in the standard model with the techniques of [DK05]. That is, choose a tag-based CCA-secure PKE scheme and a tag-based CCA-secure PEKS scheme[2], then bind the PKE ciphertext and the PEKS ciphertext together by using the verification key of a one-time signature scheme as a common tag, and append a signature of the two ciphertexts under the corresponding signing key. It is worth to note that the PKE primitive and PEKS primitive are already CCA-secure themselves, which might make the resulting PKE-PEKS scheme a bit expensive. Besides, their construction also suffers from double key size due to the adoption of key seperation strategy.

1.2 Our Contributions

Opposed to key seperation strategy, key reuse strategy dictates using the same key for multiple cryptographic operation [HP01]. In the context of PKE-PEKS scheme, similar to the case of joint encryption and signature analyzed in [PSST11], using the same key pair for both PKE and PEKS primitives can offer us at least two practical advantages: 1) reduce storage requirements for certificates as well as key pairs; 2) reduce the cost of public key certification, and the time taken for public key verification. These savings may be critical in resource-constrained applications.

Motivated by the above, we focus on building jointly CCA-secure PKE-PEKS schemes with single key pair, a problem of which, as we discussed before, there currently exists no satisfactory solution.

We first formalize the joint CCA-security notion for PKE-PEKS schemes in Section 4 by incorporating the IND-PKE-CCA security and the IND-PEKS-CCA security together. The resulting joint CCA-security notion is stronger than previous ones [BSNS06, ZI07, ABN10].

We then present two generic constructions of jointly CCA-secure PKE-PEKS schemes from (hierarchical) IBE schemes[3] in Section 5 and Section 6. More precisely, the first construction is based on any two-level HIBE scheme that is anonymous against chosen-plaintext attack (ANO-HIBE-CPA), while the second construction is based on any IBE scheme that is anonymous and weakly robust against chosen-ciphertext attack (ANO-IBE-CCA and WROB-CCA). The two constructions embody the same main idea that applying the BCHK transform [BCHK07] and the BDOP transform [BCOP04, ABC$^+$08] to the underlying (H)IBE scheme in an interweaving manner. Our constructions have several advantages over previous results. Firstly, our constructions are jointly CCA-secure in the strongest sense so far. Secondly, our constructions use a single key pair for both PKE and PEKS operation, thus enjoy the advantages brought by key reuse as described above. Thirdly, our constructions are mainly derived from one (H)IBE scheme, thus the major operations of PKE and PEKS are actually implemented in one (H)IBE scheme. This enables us to achieve better practical efficiency and compact cryptographic code. So far, a number of anonymous (H)IBE schemes based on various

[2] In [ABN10], tag-based is refered to as label-based [Sho01].

[3] Our constructions also make use of a one-time signature scheme, but it can be derived from one-way functions which in turn implied by CPA-secure encryption.

assumptions are known, such as [HL02, BF03, BGH07, GPV08] in the random oracle model and [Gen06, CIP10, ABB10, CHKP10, SC11] in the standard model; Instantiating our constructions from these (H)IBE schemes yields jointly CCA-secure PKE-PEKS schemes based on the same assumptions.

As another contribution, we revisit the notion of consistency for PEKS in Section 7. For a PEKS scheme derived from the BDOP transform, we prove that it is consistent as long as the underlying IBE scheme is weak collision-freeness. Previous result [ABN10] has to assume the underlying IBE scheme has strong robustness. which is strictly stronger notion than weak collision-freeness. We also present the enhanced notions of consistency for PEKS schemes and collision-freeness for encryption schemes. Interestingly, most schemes satisfying the original notions also satisfy the enhanced ones without any modification.

1.3 Related Work

Boneh *et al.* [BCOP04] first proposed the concept of PEKS and discussed the connection between PEKS and IBE, that is, PEKS implies IBE. Also a transform from anonymous IBE scheme to PEKS scheme (known as the BDOP transform) is implicitly presented in [BCOP04] without a formal proof. Shortly afterwards, Abdalla *et al.* [ABC$^+$08] revised the BDOP transform and provided a formal proof. Their revised transform is known as the new BDOP transform. In the same paper, they also thoroughly studied the consistency of PEKS.

Subsequent works are divided into two main directions. One thread focuses on proposing new constructions and security models. Di Crescenzo and Saraswat [CS07] constructed a PEKS scheme based on anonymous version of Cocks IBE [Coc01] in the random oracle model. As stated above, Baek *et al.* [BSNS06], Zhang *et al.* [ZI07], and Abdalla *et al.* [ABN10] gave their own security notions and constructions for PKE-PEKS schemes. The other thread focused on extending the basic concept of PEKS. Boneh and Waters [BW07] constructed a PKE scheme that supports conjunctive, subset, and range queries over the keywords. Boneh *et al.* [BKOI07] showed how to create a PKE scheme that allows PIR (Private Information Retrieval) searching. Fuhr and Paillier [FP07] introduced the concept of decryptable PEKS which allows the user to retrieve the keywords. They also gave a concrete construction in the random oracle model. Hofheinz and Weinreb [HW08] then gave a decryptable PEKS scheme in the standard model.

2 Overview of Our Approach

Before sketching our approach, we first describe our definition of joint CCA-security for PKE-PEKS schemes more clearly, then identify the main challenges in constructing jointly CCA-secure PKE-PEKS schemes.

2.1 Joint CCA-security for PKE-PEKS

The standard security notion for PKE is IND-PKE-CCA security (cf. definition in Appendix D.1), while the strong security notion so far for PEKS is IND-PEKS-CCA security (cf. definition in Appendix D.2). Towards an as-strong-as-possible joint security, we require that a PKE-PEKS scheme retain IND-PKE-CCA security (with respect to data privacy) in the presence of additional unrestricted token oracle and test oracle, and simultaneously retain IND-PEKS-CCA security (with respect to keyword privacy) in the presence of an additional unrestricted decryption oracle.

2.2 The Main Challenges

Minimizing the Size of Keys. A natural solution to resolve this challenge is adopting the *key reuse strategy*, namely using a single key pair for both PKE operation and PEKS operation. Thus the primary difficulty is obtaining a PKE scheme and a PEKS scheme sharing the same key pair. Moreover, in our joint CCA-security notion for PKE-PEKS, the adversary against data privacy has unrestricted access to an additional token oracle and an additional test oracle, while the adversary against keyword privacy has unrestricted access to an additional decryption oracle. It is very likely that the PKE component and PEKS component interacts with one another badly when they use the same key pair, and thus compromise the data privacy or the keyword privacy, leading to the corruption of joint security. Therefore, delicate technique should be used to dismiss such bad interaction due to mutual dependency. Perhaps for precise this concern, constructions in [ZI07] and [ABN10] followed key separation strategy, which easily avoids the possible mutual undermining since the PKE operation and PEKS operation are essentially independent. However, it make the resulting constructions suffer from double key pair.

Achieving the Joint CCA-security. To obtain joint CCA-security is not an easy job. Intuitively, the PKE-PEKS ciphertext should be non-malleable with respect to both PKE ciphertext and PEKS ciphertext. Several subtle issues may arise when binding a PKE scheme and a PEKS scheme together to pursue joint CCA-security. One issue is that the binding should be bidirectional, otherwise the resulting PKE-PEKS could not be joint CCA-secure since the overall ciphertext is malleable with respect to either PKE ciphertext or PEKS ciphertext. For example, if one binds a CCA-secure PKE and CCA-secure PEKS in the way like [ZI07], the binding is only unidirectional. The other issue is that both the user and the gateway must be able to check the well-formedness of the PKE-PEKS ciphertext. Note that in PKE-PEKS systems, decryption capability of the user is strictly stronger than that of the gateway. In some cases, the gateway may not be able to check the well-formedness of the ciphertext without the knowledge of secret key. Therefore, although the PKE-PEKS ciphertext is non-malleable in the view of the user, it may still be "malleable" to the gateway, thus violate the keyword privacy. We provide such an example in Remark 2 (the first attempt of using MAC to instead one-time signature).

2.3 Our Approach

We first focus on obtaining a PKE component and a PEKS component equipped with the same key pair. Our starting point is a CPA-secure and anonymous IBE scheme (see definition in Appendix D.3) consisting of algorithms Setup, $\mathsf{Extract}$, $\mathsf{Encrypt}$, $\mathsf{Decrypt}$. On the one hand, the BCHK transform [BCHK07] (cf. Appendix D.5) shows how to build a CCA-secure PKE scheme from a CPA-secure IBE scheme. Interestingly, a simple modification of the BCHK transform also allows one to create a CPA-secure PKE scheme with essentially no overhead as compared to the underlying IBE scheme, which works as follows: for key generation one runs Setup and uses the resulting master public/secret key pair as the public/secret key pair (pk, sk). To encrypt a message m under pk, one chooses a random "identity" c_1, then sets the ciphertext c to be (c_1, c_2), where $c_2 \leftarrow \mathsf{Encrypt}(pk, c_1, m)$. To recover the message from a ciphertext $c = (c_1, c_2)$, one computes $m \leftarrow \mathsf{Decrypt}(dk, c_2)$, where $dk \leftarrow \mathsf{Extract}(sk, c_1)$. The resulting PKE scheme is essentially tag-based where c_1 serves as the tag. On the other hand, the BDOP transform [BCOP04, ABC$^+$08] (cf. Appendix D.6) shows how to build a PEKS scheme from an anonymous IBE scheme, which works as follows: key generation is performed the same as the BCHK transform. To generate a token for a keyword w, user computes $t_w \leftarrow \mathsf{Extract}(sk, w)$, where t_w is acturally a private key for "identity" w. To encrypt a keyword w under pk, one picks a value s_1 from the message space of the underlying IBE scheme, then computes $s_2 \leftarrow \mathsf{Encrypt}(pk, w, s_1)$ and sets the final ciphertext s to be (s_1, s_2). The gateway identifies if s encrypts the keyword w

by testing if $s_1 = \mathsf{Decrypt}(t_w, s_2)$. Thus, applying the modified BCHK transform and the BDOP transform to a CPA-secure and anonymous IBE scheme simultaneously yields a PKE scheme and a PEKS scheme with the same key pair. By directly combining them together, we obtain a basic PKE-PEKS construction that uses a single key pair for both PKE encryption and PEKS encryption. Next, we show how to change it to a jointly CCA-secure one.

Securely Reuse the Keys. The basic PKE-PEKS construction exactly follows the key reuse strategy. However, as we envisioned before, key reuse is not without its problem. In the basic construction, a token for keyword w is exactly a private key for "identity" w, then for a PKE-PEKS ciphertext $u = (c, s)$ of (m, w), where $c = (c_1, \mathsf{Encrypt}(pk, c_1, m))$ is the PKE encryption of message m and $s = (s_1, \mathsf{Encrypt}(pk, w, s_1))$ is the PEKS encryption of keyword w, an adversary against data privacy can simply decrypt c by querying the token for keyword "c_1". The reason underlie such attack is the tag space of the PKE component and the keyword space of the PEKS component overlap each other, and consequently the private keys and tokens are meaningfully related. We resolve this problem by using a bit prefix to provide domain separation between the private keys and tokens, that is, mapping tag c_1 to "identity" $0||c_1$ while mapping keyword w to "identity" $1||w$. Our use of bit prefix trick to partitioning the identity space is reminiscent of prior work in the context of joint security of encryption and signature [PSST11]. The difference is that prior work uses bit prefix to separate the private keys and signatures.

Securely Bind PKE and PEKS. After securely reusing the key pair, we then focus on securely binding. In the above basic construction, both the PKE component and PEKS component are only CPA-secure, and there is no binding between them. Thus it can not resist chosen-ciphertext attack. The initial attempt is: generate a one-time signature key pair (vk, sk), create the PKE ciphertext c of m under tag vk, create the PEKS ciphertext s of w as before, then sign the concatenate ciphertext $c||s$ using sk and append the signature σ. However, the overall ciphertext is still malleable with respect to the PEKS ciphertext. e.g. for a ciphertext $(vk, c||s, \sigma)$, an adversary can simply generate a new signature pair (vk', sk'), then creates a new valid ciphertext $(vk', c'||s, \sigma')$. This is because the PEKS operation is still independent to the PKE operation. We solve this problem by setting vk as a part of input of PEKS encryption.

One idea is encoding vk to the "identity" for the PEKS encryption, namely setting keyword w as the first level identity and using vk as the second level identity. In this way the PKE ciphertext and the PEKS ciphertext are mutually binded together under a common value vk. Coupled with the one-time signature, the final PKE-PEKS ciphertext is non-malleable with respect to both the PKE ciphertext and the PEKS ciphertext (see Section 5 for details). The advantage of this construction is that the joint CCA-security immediately follows from the CPA security and anonymity of the underlying HIBE scheme. The crux is that in the security proof, the use of one-time signature forces the adversary's decryption queries and test queries to differ from the challenge ciphertext in a special way. The disadvantage is that the underlying IBE scheme should be hierarchical to accommodate vk as the second level identity.

The other idea is encoding vk to the "message" for the PEKS encryption. We emphasize that in such construction, the overall ciphertext is still malleable with respect to the PEKS ciphertext. This is because the PEKS ciphertext might be malleable with respect to "message" vk if the underlying IBE scheme is merely anonymous against chosen-plaintext attack. Therefore, to guarantee the joint CCA-security, the underlying IBE scheme should be at least anonymous against chosen-ciphertext attack (ANO-IBE-CCA). Somewhat surprisingly, ANO-IBE-CCA anonymity is still inadequate. Unlike the case of our first construction, the use of one-time signature here cannot force the adversary's test queries to differ from the challenge ciphertext, since now vk acts as message in the PEKS encryption. Consequently, when reducing the keyword privacy to the ANO-IBE-CCA anonymity of the underlying IBE scheme, the simulator has to handle the test query related to the challenge ciphertext itself. Some extra but natural property of the IBE scheme is required to ensure the correctness of simulation. As we will elaborate in the proof of Lemma B.2, to guarantee the joint CCA-security, the IBE scheme has to be anonymous and weakly robust against CCA-attack.

3 Definitions

Notation and conventions. For a finite set S, we use $x \overset{R}{\leftarrow} S$ to denote that x is sampled from S uniformly at random. Let κ denote the security parameter. $\perp$ is a distinguished symbol which falls outside the message space. $x||y$ is the string concatenation of x and y. A probabilistic polynomial-time (PPT) algorithm $\mathcal{A}$ is a randomized algorithm that runs in time polynomial in κ. If $\mathcal{A}$ is a randomized algorithm, we write $y \leftarrow \mathcal{A}(x_1, \ldots, x_n; r)$ to indicate that $\mathcal{A}$ outputs y on inputs $(x_1, \ldots, x_n)$ and random coins r. Sometimes for brevity, we omit r when it is not necessary to make explicit the random coins $\mathcal{A}$ uses. Particularly, for probabilistic encryption schemes, we say c an honest ciphertext if it is generated by running the encryption algorithm under fresh random coins. In this paper, all the security experiments are played between an adversary $\mathcal{A}$ and a challenger $\mathcal{CH}$. The advantage of the adversary is defined over the random coins used by $\mathcal{A}$ and $\mathcal{CH}$.

In our constructions, we will mainly use a (H)IBE scheme and a one-time signature scheme as the underlying primitives. We include the definitions and security notions of them in Appendix D.3 and D.4, respectively.

4 Integrated PKE-PEKS

A PKE-PEKS scheme [BSNS06, ZI07, ABN10] consists of the following five PPT algorithms:

- $\mathsf{KeyGen}(\kappa)$: take as input a security parameter κ, output a public/secret key pair (pk_A, sk_A). Let M be the message space, W be the keyword space, and U be the ciphertext space. We assume pk_A is used as an implicit input for algorithms $\mathsf{Decrypt}$, $\mathsf{TokenGen}$, as well as Test.
- $\mathsf{Encrypt}(pk_A, m, w)$: take as input a public key pk_A, a message $m \in M$ and a keyword $w \in W$, output a PKE-PEKS ciphertext u.
- $\mathsf{Decrypt}(sk_A, u)$: take as input a secret key sk_A and a PKE-PEKS ciphertext $u \in U$, output the plaintext $m \in M$ or a reject symbol $\perp$ indicating u is invalid.
- $\mathsf{TokenGen}(sk_A, w)$: take as input a secret key sk_A and a keyword $w \in W$, output a token t_w.
- $\mathsf{Test}(t_w, u)$: take as input a token t_w for keyword w and a PKE-PEKS ciphertext $u \in U$ which encrypts keyword w' under pk_A, output 1 if $w' = w$ and 0 otherwise.

Correctness. For any $(pk_A, sk_A) \leftarrow \mathsf{KeyGen}(\kappa)$, any $m \in M$, $w \in W$ and $t_w \leftarrow \mathsf{TokenGen}(sk_A, w)$, $\mathsf{Decrypt}(sk_A, \mathsf{Encrypt}(pk_A, m, w)) = m$ and $\mathsf{Test}(t_w, \mathsf{Encrypt}(pk_A, m, w)) = 1$.
Consistency. For any $(pk_A, sk_A) \leftarrow \mathsf{KeyGen}(\kappa)$, any $m \in M$, two distinct keywords $w, w' \in W$ and $t_w \leftarrow \mathsf{TokenGen}(sk_A, w)$, $\mathsf{Test}(t_w, \mathsf{Encrypt}(pk_A, m, w')) = 0$ holds overwhelmingly.

4.1 Joint CCA-security for PKE-PEKS

We consider data privacy and keyword privacy for PKE-PEKS schemes in joint sense as follows.

Data Privacy for PKE-PEKS schemes is defined via the following experiment.
Setup: $\mathcal{CH}$ runs $\mathsf{KeyGen}(\kappa)$ and gives $\mathcal{A}$ the public key pk_A.
Phase 1: $\mathcal{A}$ adaptively makes three types of queries:

– Decryption query $\langle u \rangle$: $\mathcal{CH}$ responds with $m \leftarrow \mathsf{Decrypt}(sk_A, u)$.
– Token query $\langle w \rangle$: $\mathcal{CH}$ responds with $t_w \leftarrow \mathsf{TokenGen}(sk_A, w)$.
– Test query $\langle u, w \rangle$: $\mathcal{CH}$ responds with $b \leftarrow \mathsf{Test}(u, t_w)$ where $t_w \leftarrow \mathsf{TokenGen}(sk_A, w)$.

Challenge: $\mathcal{A}$ outputs two message m_0 and m_1 and a keyword w^*. $\mathcal{CH}$ picks a random bit $b \in \{0,1\}$ and sends $u^* \leftarrow \mathsf{Encrypt}(pk_A, m_b, w^*)$ to $\mathcal{A}$ as the challenge ciphertext.
Phase 2: $\mathcal{A}$ continues to issue more decryption queries $\langle u \rangle$ subject to the restriction that $u \neq u^*$, more token queries $\langle w \rangle$ and more test queries $\langle u, w \rangle$ with no restriction (token query $\langle w^* \rangle$ and test query $\langle u^*, w^* \rangle$ are allowed!). $\mathcal{CH}$ responds the same way as in Phase 1.
Guess: $\mathcal{A}$ outputs $b' \in \{0,1\}$ of b and wins if $b' = b$.

We define its advantage as $\mathrm{Adv}_{\mathcal{A}}(\kappa) = |\Pr[b' = b] - 1/2|$.

Definition 4.1 *A PKE-PEKS scheme has $(t, Q_d, Q_w, Q_t, \epsilon)$ data privacy if for all t-time PPT adversary making at most Q_d decryption queries and Q_w token queries and Q_t test queries have advantage at most ϵ in the above experiment.*

Keyword Privacy for PKE-PEKS schemes is defined via the following experiment.
Setup: $\mathcal{CH}$ runs $\mathsf{KeyGen}(\kappa)$ and gives $\mathcal{A}$ the public key pk_A.
Phase 1: Same as that in the experiment for data privacy.
Challenge: $\mathcal{A}$ outputs two keywords w_0^* and w_1^* subject to the restriction that the two keywords had not been asked for tokens in Phase 1, and a message m. $\mathcal{CH}$ picks a random bit $b \in \{0,1\}$ and sends $c^* \leftarrow \mathsf{Encrypt}(pk_A, m, w_b)$ to $\mathcal{A}$ as the challenge ciphertext.
Phase 2: $\mathcal{A}$ continues to adaptively make more token queries $\langle w \rangle$ subject to the restriction that $w \neq w_0^*, w_1^*$, more test queries $\langle u, w \rangle$ subject to the restriction that $(u, w) \neq (u^*, w_0^*), (u^*, w_1^*)$, and more decryption queries with no restriction ($\langle u^* \rangle$ is allowed!).
Guess: $\mathcal{A}$ outputs a guess $b' \in \{0,1\}$ of b and wins if $b' = b$.

We define $\mathcal{A}$'s advantage as $\mathrm{Adv}_{\mathcal{A}}(\kappa) = |\Pr[b' = b] - 1/2|$.

Definition 4.2 *A PKE-PEKS scheme has $(t, Q_d, Q_w, Q_t, \epsilon)$ keyword privacy if for all t-time PPT adversary making at most Q_d decryption queries and Q_w token queries and Q_t test queries have advantage at most ϵ in the above experiment.*

Definition 4.3 *We say a PKE-PEKS scheme is jointly CCA-secure if it has the keyword privacy and data privacy defined as above simultaneously.*

Our joint CCA-security notion for PKE-PEKS schemes is stronger than previous ones considered in [BSNS06, ZI07, ABN10], since it embodies both IND-PKE-CCA security and IND-PEKS-CCA security in the joint sense. As analyzed earlier, the PKE-PEKS constructions proposed in [BSNS06] and [ZI07] are both insecure in our joint CCA-security notion.

5 A Generic Construction from HIBE

We first show how to construct a PKE-PEKS scheme from a two-level HIBE scheme with algorithms (Setup, $\mathsf{Extract}$, Derive, $\mathsf{Encrypt}$, $\mathsf{Decrypt}$). In our construction we also make use of a strong one-time signature scheme OT with algorithms (KeyGen, Sign, Verify) (cf. definition in Appendix D.4). Without loss of generality, we assume that the message space of the HIBE scheme is $\{0,1\}^n$, the identity space of the HIBE scheme is $\{0,1\}^*$ for each level, and the verification key space of the signature scheme is $\{0,1\}^n$, where $n = n(\kappa)$ is a polynomially bounded function.

$\mathsf{KeyGen}(\kappa)$: Run $(mpk, msk) \leftarrow \text{HIBE}.\mathsf{Setup}(\kappa)$, return $(pk_A, sk_A) \leftarrow (mpk, msk)$.

```
Encrypt(pk_A, m, w):
 1. Run $(vk, sk) \leftarrow \text{OT}.\mathsf{KeyGen}(\kappa)$.
 2. Encrypt message $m$ using level-1 identity $0||vk$ as $c \leftarrow \text{HIBE}.\mathsf{Encrypt}(pk_A, 0||vk, m)$.
 3. Encrypt $vk$ using level-2 identity $(1||w, vk)$ as $s \leftarrow \text{HIBE}.\mathsf{Encrypt}(pk_A, (1||w, vk), vk)$.
 4. Compute $\sigma \leftarrow \text{OT}.\mathsf{Sign}(sk, c||s)$, return the final ciphertext $u = (vk, c, s, \sigma)$.
```

```
Decrypt(sk_A, u):
 1. Parse $u$ as $(vk, c, s, \sigma)$.
 2. If $\text{OT}.\mathsf{Verify}(vk, c||s, \sigma) = 1$, then compute $dk \leftarrow \text{HIBE}.\mathsf{Extract}(sk_A, 0||vk)$,
        return $m \leftarrow \text{HIBE}.\mathsf{Decrypt}(dk, c)$.
    Else return $\perp$.
```

$\mathsf{TokenGen}(sk_A, w)$: compute $t_1 \leftarrow \text{HIBE}.\mathsf{Extract}(sk_A, 1||w)$, $t_2 \leftarrow w$, return $t_w = (t_1, t_2)$.

```
Test(t_w, u):
 1. Parse $t_w$ as $(t_1, t_2)$, $u$ as $(vk, c, s, \sigma)$.
 2. If $\text{OT}.\mathsf{Verify}(vk, c||s, \sigma) = 1$, then compute $dk \leftarrow \text{HIBE}.\mathsf{Derive}(t_1, (1||t_2, vk))$,
        return 1 if $vk = \text{HIBE}.\mathsf{Decrypt}(dk, s)$ and 0 otherwise.
    Else return 0.
```

This completes the description of our first construction. The keyword space W of the resulting PKE-PEKS scheme is $\{0,1\}^*$. The correctness of the PKE-PEKS scheme follows readily from that of the underlying HIBE scheme. We show how to instantiate our first construction from pairing and lattice in Appendix C.1 and Appendix C.2.

Theorem 5.1 *The above PKE-PEKS construction is jointly CCA-secure, provided that the underlying HIBE scheme is IND-HIBE-CPA secure in selective-identity sense and ANO-HIBE-CPA anonymous at level 1 and the signature scheme is one-time sEUF-CMA secure.*

Due to the space limit, we defer the proof to Appendix A.

6 A Generic Construction from IBE

We now show how to construct a PKE-PEKS scheme from an IBE scheme with algorithms (Setup, $\mathsf{Extract}$, $\mathsf{Encrypt}$, $\mathsf{Decrypt}$). As in the first construction, we also make use of a strong one-time signature scheme, and assume that the message space of the IBE scheme is $\{0,1\}^n$, the identity space of the IBE scheme is $\{0,1\}^*$, and the verification key space of the signature scheme is $\{0,1\}^n$.

$\mathsf{KeyGen}(\kappa)$: Run $(mpk, msk) \leftarrow \text{IBE}.\mathsf{Setup}(\kappa)$, return $(pk_A, sk_A) \leftarrow (mpk, msk)$.

```
Encrypt(pk_A, m, w):
 1. Run $(vk, sk) \leftarrow \text{OT}.\mathsf{KeyGen}(\kappa)$.
 2. Encrypt message $m$ using identity $0||vk$ as $c \leftarrow \text{IBE}.\mathsf{Encrypt}(pk_A, 0||vk, m)$.
 3. Encrypt keyword $w$ using identity $1||w$ as $s \leftarrow \text{IBE}.\mathsf{Encrypt}(pk_A, 1||w, vk)$.
 4. Compute $\sigma \leftarrow \text{OT}.\mathsf{Sign}(sk, c||s)$, return the final ciphertext $u = (vk, c, s, \sigma)$.
```

```
Decrypt(sk_A, u):
 1. Parse $u$ as $(vk, c, s, \sigma)$.
 2. If $\text{OT}.\mathsf{Verify}(vk, c||s, \sigma) = 1$,
        then compute $dk \leftarrow \text{IBE}.\mathsf{Extract}(sk_A, 0||vk)$, return $m \leftarrow \text{IBE}.\mathsf{Decrypt}(dk, c)$.
    Else return $\perp$.
```

$\mathsf{TokenGen}(sk_A, w)$: return $t_w \leftarrow \text{IBE}.\mathsf{Extract}(sk_A, 1||w)$.

$\mathsf{Test}(t_w, u)$:

```
1. Parse $u$ as $(vk, c, s, \sigma)$.
2. If $\text{OT}.\mathsf{Verify}(vk, c||s, \sigma) = 1$,
        return 1 if $vk = \text{IBE}.\mathsf{Decrypt}(t_w, s)$ and 0 otherwise.
   Else return 0.
```

This completes the description of our second construction. Similar to the our first construction, the keyword space W of the resulting PKE-PEKS scheme is $\{0,1\}^*$, and the correctness follows readily from that of the underlying IBE scheme. We show how to instantiate our second construction from pairing in Appendix C.3.

Theorem 6.1 *The above PKE-PEKS construction is jointly CCA-secure, provided that the underlying IBE scheme is IND-IBE-CPA secure in the selective-identity sense and ANO-IBE-CCA anonymous and WROB-CCA robust and the signature scheme is one-time sEUF-CMA secure.*

Due to the space limit, we defer the proof to Appendix B.

Remark 1. In our first construction, we use vk as the "message" for the PEKS encryption to further reduce the size of the overall ciphertext. In our second constrcution, the PKE component only need to be CPA-secure. Thus we may achieve better efficiency by employing the selective-identity CPA-secure version of the underlying IBE scheme to build the PKE component. In both of our two constructions, since the user knows the "master secret key" sk_A of the underlying (H)IBE scheme, then it is very likely to speed up the decryption by directly using sk_A. See the instantiation of presented in Appendix C.3 for such an example.

Remark 2. It seems unlikely to improve our constructions by adapting the optimization method suggested by [BCHK07] which replaces the one-time signature scheme by a combination of a message-authentication code (MAC) and a weak form of commitment. We sketch the reasons as below. One attempt is to encrypt the decommitment string dec in the PKE part, the MAC computed over the PKE-PEKS ciphertext can only be verified by the user holding the secret key sk_A. The gateway is unable to check the well-formedness of the PKE-PEKS ciphertexts since lacking of sk_A. The other attempt is to encrypt dec in the PEKS part, the gateway can check the MAC now since now it can recover the MAC key k. However, with k it can also manipulate the PKE-PEKS ciphertext and thus breaks the data privacy. We omit the details here.

7 Consistency for PEKS Revisited

For a cryptographic primitive, except its primary security requirement, we also need to pay attention to its consistency requirement, which ensures that the primitive fulfills its basic function. As to PEKS, its consistency requirement set by [BCOP04] is that for two keywords w, w' and a PEKS ciphertext s of w, $\mathsf{Test}(t_{w'}, s)$ returns 1 if $w = w'$ and returns 0 if $w \neq w'$. This condition can be divided into two parts. The former might be viewed as an analogy of correctness for IBE (decryption reverses encryption), which indicates that $\mathsf{Test}(t_w, s) = 1$ when s encrypts w. Adopting the treatment of [ABC$^+$08], we view it as a built-in property for PEKS and reserve the term consistency to the latter, namely $\mathsf{Test}(t_{w'}, s) = 0$ when s encrypts w and $w \neq w'$. Abdalla *et al.* [ABC$^+$08] formally defined the perfect, statistical, and computational consistency for PEKS. In this paper, we are only interested in computational consistency.

7.1 Consistency for PEKS

The original notion of consistency for PEKS [ABC$^+$08] is defined by the following experiment:

Initialize: $\mathcal{CH}$ runs $\mathsf{KeyGen}(\kappa)$ to generate (pk_A, sk_A) and gives pk_A to $\mathcal{A}$.
Finalize: $\mathcal{A}$ outputs two distinct keywords w and w'. $\mathcal{CH}$ honestly generates a PEKS ciphertext $s \leftarrow \mathsf{PEKS}(pk_A, w)$. $\mathcal{A}$ wins if $\mathsf{Test}(t_{w'}, s) = 1$, where $t_{w'} \leftarrow \mathsf{TokenGen}(sk_A, w')$.

$\mathcal{A}$'s advantage is defined as $\Pr[\mathcal{A}$ wins$]$. A PEKS scheme is said to be consistent if no PPT adversary has non-negligible advantage in the above experiment.

7.2 Robustness for IBE

Abdalla *et al.* [ABN10] formally studied the *robustness* for encryption schemes. Intuitively, in the IBE setting the robustness requires that a ciphertext c does not decrypt to a valid plaintext under the private keys for two distinct identities id and id'. The original notion of robustness for IBE is defined by the following experiment:

Initialize: $\mathcal{CH}$ runs $\mathsf{Setup}(\kappa)$ to generate (mpk, msk) and gives mpk to $\mathcal{A}$.
Phase 1: $\mathcal{A}$ may adaptively make two types of queries:

- Private key extraction queries $\langle id \rangle$: $\mathcal{CH}$ responds with $dk \leftarrow \mathsf{Extract}(msk, id)$.
- Decryption queries $\langle c, id \rangle$: $\mathcal{CH}$ responds with $m \leftarrow \mathsf{Decrypt}(dk, c)$ where $dk \leftarrow \mathsf{Extract}(msk, id)$.

Finalize (weak robustness): $\mathcal{A}$ outputs two distinct identities id and id' subject to the restriction that they had not been asked for private keys in Phase 1, and a message $m \in M$. $\mathcal{CH}$ honestly generates an IBE ciphertext $c \leftarrow \mathsf{Encrypt}(mpk, id, m)$. $\mathcal{A}$ wins if $m' \neq \perp$, where $m' \leftarrow \mathsf{Decrypt}(dk', c)$ for $dk' \leftarrow \mathsf{Extract}(msk, id')$.
Finalize (strong robustness): $\mathcal{A}$ outputs two distinct identities id and id' subject to the restriction that they had not been asked for private keys in Phase 1, and a ciphertext c. Note that c may not be an honest encryption, say, generated using adversarially chosen random coins. $\mathcal{A}$ wins if $m \neq \perp$ and $m' \neq \perp$, where $m \leftarrow \mathsf{Decrypt}(dk, c)$ for $dk \leftarrow \mathsf{Extract}(msk, id)$ and $m' \leftarrow \mathsf{Decrypt}(dk', c)$ for $dk' \leftarrow \mathsf{Extract}(msk, id')$.

Both weak and strong robustness are considered under chose-plaintext or chosen-ciphertext attacks in [ABN10], yielding: WROB-CPA, WROB-CCA, SROB-CPA, SROB-CCA. In case of CPA attack, the decryption queries are not allowed. $\mathcal{A}$'s advantage in the above {WROB, SROB}-{CPA, CCA} experiment is defined as $\Pr[\mathcal{A}$ wins$]$. An IBE scheme is said to be {WROB, SROB}-{CPA, CCA} if no PPT adversary has non-negligible advantage in the corresponding experiment.

7.3 Collision-freeness for IBE

Mohassel [Moh10] introduced the *collision-freeness* for encryption schemes, which is a natural relaxation of robustness. Intuitively, in the IBE setting the collision-freeness requires that a ciphertext does not decrypt to the same message under the private keys for two distinct identities. The original notion of collision-freeness for IBE is defined by the following experiment:

Initialize: same as the experiment for robustness.
Phase 1: same as the experiment for robustness.
Finalize (weak collision-freeness): $\mathcal{A}$ outputs two distinct identities id and id' subject to the restriction that they had not been asked for private keys in Phase 1, and a message $m \in M$. $\mathcal{CH}$ honestly generates an IBE ciphertext $c \leftarrow \mathsf{Encrypt}(mpk, id, m)$. $\mathcal{A}$ wins if $m = m'$, where $m' \leftarrow \mathsf{Decrypt}(dk', c)$ for $dk' \leftarrow \mathsf{Extract}(msk, id')$.

Finalize (strong collision-freeness): $\mathcal{A}$ outputs two distinct identities id and id' subject to the restriction that they had not been asked for private keys in Phase 1, and a ciphertext c. Note that c may not be an honest encryption. $\mathcal{A}$ wins if $m = m'$, where $m \leftarrow \mathsf{Decrypt}(dk, c)$ for $dk \leftarrow \mathsf{Extract}(msk, id)$ and $m' \leftarrow \mathsf{Decrypt}(dk', c)$ for $dk' \leftarrow \mathsf{Extract}(msk, id')$.

Similar to the notion of robustness, both weak and strong collision-freeness are considered under chosen-plaintext or chosen-ciphertext attacks [Moh10], yielding: WCFR-CPA, WCFR-CCA, SCFR-CPA, SCFR-CCA. In case of CPA attack, the decryption queries are not allowed. $\mathcal{A}$'s advantage in the above {WCFR, SCFR}-{CPA, CCA} experiment is defined as $\Pr[\mathcal{A} \text{ wins}]$. An IBE scheme is said to be {WCFR, SCFR}-{CPA, CCA} if no PPT adversary has non-negligible advantage in the corresponding experiment.

7.4 Improvment on Previous Result

Addalla *et al.* [ABC+08] noticed that the original BDOP transform generally does not provide consistency, and thus proposed the new BDOP transform which is consistency-providing instead. After studying the robustness for encryption schemes, Abdalla *et al.* [ABN10] realized that the original BDOP transform can guarantee the consistency of the resulting PEKS scheme if the underlying IBE scheme is SROB-CPA.

However, the strong robustness seems to be overkill for consistency. Note that Mohassel [Moh10] has pointed out that collision-freeness can be a sufficient requirement instead of robustness in some scenarios in practice. It is compelling to know if we can base the consistency for PEKS on the collision-freeness for IBE. In the case of consistency for PEKS schemes, we get the following result:

Theorem 7.1 *For a PEKS scheme constructed using the BDOP transform, it is consistent if the underlying IBE scheme is weak collision-free against chosen-plaintext attack.*

Proof. Suppose there is an adversary $\mathcal{A}$ has non-negligible advantage against the consistency of the PEKS scheme, we can build an adversary $\mathcal{B}$ breaking the WCFR-CPA property of the underlying IBE scheme as follows:

Initialize: $\mathcal{B}$ is given (mpk, msk) of the underlying IBE scheme. $\mathcal{B}$ forwards mpk to $\mathcal{A}$ as pk_A.
Finalize: $\mathcal{A}$ outputs two distinct keywords w and w'. $\mathcal{B}$ outputs $(id = w, id' = w', m)$, where m is set to be the pre-fixed value specified by the BDOP transform. Let c be an ciphertext of m under id which is honestly generated by $\mathcal{B}$'s challenger, then $s = (m, c)$ is also an honestly-generated PEKS ciphertext.

Since $\mathsf{Test}(t_{w'}, s) = 1$ is equivalent to $\mathsf{Decrypt}(c, dk_{id'}) = m$, thus $\mathcal{B}$ breaks the WCFR-CPA property of the underlying IBE scheme as long as $\mathcal{A}$ breaks the consistency of the PEKS scheme. This prove the theorem. $\square$

It is straightforward to verify that this theorem also holds for the new BDOP transform. Note that Mohassel [Moh10] has shown that WCFR-CPA is weaker than SROB-CPA, thus our result improves the previous result due to [ABN10].

7.5 Enhanced Notions of Consistency and Collision-freeness

Abdalla *et al.* [ABC+08] formulated the consistency for PEKS more like a security condition, via a security experiment involving an adversary. Just as they indicated, the adversary against consistency is not very "adversarial": it is modeled to capture some kind of worst case but not malicious behavior. They also noticed that the basic notion of consistency can be made stronger by giving the adversary a token oracle and/or a test oracle. Indeed, we would go so far as to say the consistency should retain even the secret key sk_A is exposed to the adversary. This interpretation agrees with the initial description of consistency for PEKS [BCOP04]. In our sense, consistency, like correctness, should also be viewed as a built-in property for PEKS.

Enhanced Notion of Consistency. We formally define the enhanced notion of consistency for PEKS as follows:
Initialize: $\mathcal{CH}$ runs $\mathsf{KeyGen}(\kappa)$ to generate (pk_A, sk_A) and gives both pk_A and sk_A to $\mathcal{A}$.
Finalize: $\mathcal{A}$ submits two distinct keywords w and w'. $\mathcal{CH}$ honestly generates a PEKS ciphertext $s \leftarrow \mathsf{PEKS}(pk_A, w)$. $\mathcal{A}$ wins if $\mathsf{Test}(t_{w'}, s) = 1$, where $t_{w'} = \mathsf{TokenGen}(sk_A, w')$.

$\mathcal{A}$'s advantage is defined as $\Pr[\mathcal{A}$ wins$]$. A PEKS scheme is consistent in the enhanced sense if no PPT adversary $\mathcal{A}$ can win the above experiment with non-negligible probability.

Enhanced Notion of Collision-freeness. For the same reasoning, the original notion of collision-freeness can also be enhanced. Particularly, in the IBE setting the enhanced collision-freeness can be defined analogously as above. Namely, no adversary can break the collision-freeness with non-negligible probability even it knows the master secret key msk. As an analogy of Theorem 7.1, we claim that a PEKS scheme is enhanced consistent if the underlying IBE scheme is enhanced collision-free. We omit the proof here due to its straightforwardness.

Schemes such as ElGamal PKE [ElG85] and the Boyen-Waters IBE [BW06] are shown to be strong collision-free in [Moh10]. Interestingly, they are also strong collision-free in the enhanced sense. Moreover, we find that many encryption schemes are at least weak collision-freeness in the enhanced sense without any modification, such as Cramer-Shoup PKE [CS02], DHIES [ABR01], Boneh-Franklin IBE [BF03], Sakai-Kasahara IBE [SK03], Boneh-Boyen IBE [BB04a], and Waters IBE [Wat05]. For many encryption schemes, the collision-freeness inherently rely on their internal structure and the collision-resistance of hash functions being used, but not build on the hardness of the underlying assumptions. Thereby, we think the enhanced notion of collision-freeness is a reasonable strengthen. It might be fairly viewed as a built-in property for PKE/IBE. The same observation also holds for the case of robustness, thus we can enhance the notion of robustness similarly. We would like to note that the notion of enhanced robustness is independently proposed by Farshim *et al.* [FLPQ12], which they refer to as unrestricted strong robustness.

7.6 Consistency for PKE-PEKS

The notion of consistency for PEKS schemes extends naturally to that for PKE-PEKS schemes. For a similar reduction we did in the proof of Theorem 7.1, it is straightforward to verify our two constructions presented in Section 5 and Section 6 are consistent if the underlying (H)IBE scheme is weakly collision-free. We omit the details here.

8 Extensions

Keyword Hiding Property. As noticed in [BW07, HW08, Nis12], the IND-PEKS-CPA/CCA notions only formalize the security that searchable ciphertext does not reveal any information about the keyword, but neglects to formalize the keyword privacy against gateway, namely a token does not reveal its corresponding keyword. We refer to such privacy as keyword hiding property, which may be desired in some cases. For instance, in the email example from the introduction, Alice may not want to reveal her interested keyword to the gateway. We emphasize that, in the non-interactive setting[4], one can not expect indistinguishability[5] but only one-wayness. This is because given two different keywords w_0 and w_1 and a token t_{w_b}, one can always learn b by using encrypt-then-test method [HW08, Nis12].

In our first constrcution, token t_w for keyword w is of the form $(dk_{1||w}, w)$ and thus t_w exposes the whole keyword w in plain. This design is crucial for our purpose, since the gateway might need the information of w to generate a private key for a "temporary identity" $(1||w, vk)$, then uses it to run the Test algorithm. In fact, we can provide a simple and generic patch to achieve *somewhat* keyword hiding property for any PEKS scheme that such property is obviously absent. That is, for a PEKS scheme Π with keyword space W, we compile it to Π' with keyword space W' by using $H(w)$ in the place of w, where H is a collision resistant hash function from W to W'. It is easy to verify that the patched PEKS scheme preserve its original keyword privacy (IND-PEKS-CPA or IND-PEKS-CCA) and consistency due to the collision resistance of H and acquires somewhat keyword hiding property due to the one-wayness of H, which in turn based on the collision resistance of H. We also remark that similar methods have been suggested by [HW08, Nis12], in which H is required to be one-way. However, one-wayness alone can ensure somewhat keyword hiding property, but it is insufficient to preserve the original keyword privacy and consistency.

Integrated IBE and IBEKS. Abdalla *et al.* [ABC$^+$08] put forwarded the concept of Identity-Based Encryption with Keyword Search (IBEKS). Naturally, we can define the integrated (H)IBE and (H)IBEKS scheme and its joint security notion analogously. Both our constructions extends to give transforms from (H)IBE with appropriate properties to jointly CCA-secure integrated (H)IBE-(H)IBEKS scheme.

Bibliography

[ABB10] Shweta Agrawal, Dan Boneh, and Xavier Boyen. Efficient lattice (h)ibe in the standard model. In *Advances in Cryptology - EUROCRYPT 2010*, volume 6110 of *Lecture Notes in Computer Science*, pages 553–572. Springer, 2010.

[ABC$^+$08] Michel Abdalla, Mihir Bellare, Dario Catalano, Eike Kiltz, Tadayoshi Kohno, Tanja Lange, John Malone-Lee, Gregory Neven, Pascal Paillier, and Haixia Shi. Searchable encryption revisited: Consistency properties, relation to anonymous ibe, and extensions. *J. Cryptology*, 21(3):350–391, 2008.

[ABN10] Michel Abdalla, Mihir Bellare, and Gregory Neven. Robust encryption. In *TCC 2010*, volume 5978 of *Lecture Notes in Computer Science*, pages 480–497. Springer, 2010.

[ABR01] Michel Abdalla, Mihir Bellare, and Phillip Rogaway. The oracle diffie-hellman assumptions and an analysis of dhies. In *Topics in Cryptology - CT-RSA 2001*, volume 2020 of *LNCS*, pages 143–158, 2001.

[BB04a] Dan Boneh and Xavier Boyen. Efficient selective-id secure identity based encryption without random oracles. In *Advances in Cryptology - EUROCRYPT 2004*, volume 3027 of *Lecture Notes in Computer Science*, pages 223–238, 2004.

[BB04b] Dan Boneh and Xavier Boyen. Short signatures without random oracles. In *Advances in Cryptology - EUROCRYPT 2004*, volume 3027 of *Lecture Notes in Computer Science*, pages 56–73, 2004.

[4] This means both token generation and keyword search are done in a non-interactive manner.

[5] Indistinguishability is possible in the interactive setting. Boneh *et al* [BKOI07] constructed a PEKS scheme allowing Private Information Retrieval [KO97], which can hide all the information including the keyword in a semantically-secure way.

[BB08] Dan Boneh and Xavier Boyen. Short signatures without random oracles and the sdh assumption in bilinear groups. *J. Cryptology*, 21(2):149–177, 2008.

[BCHK07] Dan Boneh, Ran Canetti, Shai Halevi, and Jonathan Katz. Chosen-ciphertext security from identity-based encryption. *SIAM J. Comput.*, 36(5):1301–1328, 2007.

[BCOP04] Dan Boneh, Giovanni Di Crescenzo, Rafail Ostrovsky, and Giuseppe Persiano. Public key encryption with keyword search. In *Advances in Cryptology - EUROCRYPT 2004*, volume 3621 of *Lecture Notes in Computer Science*, pages 506–522, 2004.

[BF03] Dan Boneh and Matthew K. Franklin. Identity-based encryption from the weil pairing. *SIAM Journal on Computation*, 32:586–615, 2003.

[BGH07] Dan Boneh, Craig Gentry, and Michael Hamburg. Space-efficient identity based encryption without pairings. In *48th Annual IEEE Symposium on Foundations of Computer Science (FOCS 2007)*, pages 647–657. IEEE Computer Society, 2007.

[BKOI07] Dan Boneh, Eyal Kushilevitz, Rafail Ostrovsky, and William E. Skeith III. Public key encryption that allows pir queries. In *Advances in Cryptology - CRYPTO 2007*, volume 4622 of *Lecture Notes in Computer Science*, pages 50–67. Springer, 2007.

[Boy10] Xavier Boyen. Lattice mixing and vanishing trapdoors: A framework for fully secure short signatures and more. In *Public Key Cryptography - PKC 2010*, volume 6056 of *Lecture Notes in Computer Science*, pages 499–517. Springer, 2010.

[BSNS06] Joonsang Baek, Reihaneh Safavi-Naini, and Willy Susilo. On the integration of public key data encryption and public key encryption with keyword search. In *Information Security, 9th International Conference, ISC 2006*, volume 4176 of *Lecture Notes in Computer Science*, pages 217–232. Springer, 2006.

[BW06] Xavier Boyen and Brent Waters. Anonymous hierarchical identity-based encryption (without random oracles). In *Advances in Cryptology - CRYPTO 2006*, volume 4117 of *Lecture Notes in Computer Science*, pages 290–307. Springer, 2006.

[BW07] Dan Boneh and Brent Waters. Conjunctive, subset, and range queries on encrypted data. In *Theory of Cryptography, 4th Theory of Cryptography Conference, TCC 2007*, volume 4392 of *Lecture Notes in Computer Science*, pages 535–554. Springer, 2007.

[CHKP10] David Cash, Dennis Hofheinz, Eike Kiltz, and Chris Peikert. Bonsai trees, or how to delegate a lattice basis. In *Advances in Cryptology - EUROCRYPT 2010*, volume 6110 of *Lecture Notes in Computer Science*, pages 523–552. Springer, 2010.

[CIP10] Angelo De Caro, Vincenzo Iovino, and Giuseppe Persiano. Fully secure anonymous hibe and secret-key anonymous ibe with short ciphertexts. In *4th International Conference - Pairing-Based Cryptography - Pairing 2010*, volume 6487 of *Lecture Notes in Computer Science*, pages 347–366. Springer, 2010.

[Coc01] Clifford Cocks. An indentity based encryption scheme based on quadratic residues. In *Institute of Mathematics and Its Applications International Conference on Cryptigraphy and Coding Proceedings of IMA 2001*, volume 2260 of *Lecture Notes in Computer Science*, pages 360–363, 2001.

[CS02] Ronald Cramer and Victor Shoup. Universal hash proofs and a paradigm for adaptive chosen ciphertext secure public-key encryption. In *Advances in Cryptology - EUROCRYPT 2002*, volume 2332 of *LNCS*, pages 45–64. Springer, 2002.

[CS07] Giovanni Di Crescenzo and Vishal Saraswat. Public key encryption with searchable keywords based on jacobi symbols. In *Progress in Cryptology - INDOCRYPT 2007*, volume 4859 of *Lecture Notes in Computer Science*, pages 282–296. Springer, 2007.

[DDN00] Danny Dolev, Cynthia Dwork, and Moni Naor. Nonmalleable cryptography. *SIAM J. Comput.*, 30(2):391–437, 2000.

[DK05] Yevgeniy Dodis and Jonathan Katz. Chosen-ciphertext security of multiple encryption. In *TCC 2005*, volume 3378 of *Lecture Notes in Computer Science*, pages 188–209. Springer, 2005.

[ElG85] Taher ElGamal. A public key cryptosystem and a signature scheme based on discrete logarithms. *IEEE Transactions on Information Theory*, 31:469–472, 1985.

[FLPQ12] Pooya Farshim, Benoît Libert, Kenneth G. Paterson, and Elizabeth A. Quaglia. Robust encryption, revisited. Cryptology ePrint Archive, Report 2012/673, 2012. `http://eprint.iacr.org/`.

[FP07] Thomas Fuhr and Pascal Paillier. Decryptable searchable encryption. In *Provable Security, First International Conference, ProvSec 2007*, volume 4784 of *Lecture Notes in Computer Science*, pages 228–236. Springer, 2007.

[Gen06] Craig Gentry. Practical identity-based encryption without random oracles. In *Advances in Cryptology - EUROCRYPT 2006*, volume 4004 of *Lecture Notes in Computer Science*, pages 445–464, 2006.

[GPV08] Craig Gentry, Chris Peikert, and Vinod Vaikuntanathan. Trapdoors for hard lattices and new cryptographic constructions. In *Proceedings of the 40th Annual ACM Symposium on Theory of Computing, STOC*, pages 197–206. ACM, 2008.

[HL02] Jeremy Horwitz and Ben Lynn. Toward hierarchical identity-based encryption. In *Advances in Cryptology - EUROCRYPT 2002*, volume 2322 of *Lecture Notes in Computer Science*, pages 466–481, 2002.

[HP01] Stuart Haber and Benny Pinkas. Securely combining public-key cryptosystems. In *ACM Conference on Computer and Communications Security*, pages 215–224. ACM, 2001.

[HW08] Dennis Hofheinz and Enav Weinreb. Searchable encryption with decryption in the standard model. IACR Cryptology ePrint Archive, Report 2008/423, 2008. `http://eprint.iacr.org/2008/423`.

[KO97] Eyal Kushilevitz and Rafail Ostrovsky. Replication is not needed: Single database, computationally-private information retrieval. In *38th Annual Symposium on Foundations of Computer Science, FOCS 1997*, pages 364–373. IEEE Computer Society, 1997.

[KR00] Hugo Krawczyk and Tal Rabin. Chameleon signatures. In *Proceedings of the Network and Distributed System Security Symposium, NDSS 2000*. The Internet Society, 2000.

[Moh10] Payman Mohassel. A closer look at anonymity and robustness in encryption schemes. In *Advances in Cryptology - ASIACRYPT 2010*, volume 6477 of *Lecture Notes in Computer Science*, pages 501–518. Springer, 2010.

[MP12] Daniele Micciancio and Chris Peikert. Trapdoors for lattices: Simpler, tighter, faster, smaller. In *Advances in Cryptology - EUROCRYPT 2012*, volume 7237 of *Lecture Notes in Computer Science*, pages 700–718. Springer, 2012.

[Nis12] Mototsugu Nishioka. Perfect keyword privacy in peks systems. In *Provable Security - 6th International Conference, ProvSec 2012*, volume 7496 of *Lecture Notes in Computer Science*, pages 175–192. Springer, 2012.

[OP01] Tatsuaki Okamoto and David Pointcheval. The gap-problems: A new class of problems for the security of cryptographic schemes. In *Public Key Cryptography - PKC 2001*, volume 1992 of *LNCS*, pages 104–118, 2001.

[PSST11] Kenneth G. Paterson, Jacob C. N. Schuldt, Martijn Stam, and Susan Thomson. On the joint security of encryption and signature, revisited. In *Advances in Cryptology - ASIACRYPT 2011*, volume 7073 of *Lecture Notes in Computer Science*, pages 161–178. Springer, 2011.

[RS91] Charles Rackoff and Daniel R. Simon. Non-interactive zero-knowledge proof of knowledge and chosen ciphertext attack. In *Advances in Cryptology - CRYPTO 1991*, volume 576 of *LNCS*, pages 433–444, 1991.

[SC11] Jae Hong Seo and Jung Hee Cheon. Fully secure anonymous hierarchical identity-based encryption with constant size ciphertexts. IACR Cryptology ePrint Archive, Report 2011/021, 2011. `http://eprint.iacr.org/2011/021`.

[Sho01] Victor Shoup. A proposal for an iso standard for public key encryption. Cryptology ePrint Archive, Report 2001/112, 2001. `http://eprint.iacr.org/2001/112`.

[SK03] Ryuichi Sakai and Masao Kasahara. Id based cryptosystems with pairing on elliptic curve. Cryptology ePrint Archive, Report 2003/054, 2003. `http://eprint.iacr.org/2003/054`.

[Wat05] Brent Waters. Efficient identity-based encryption without random oracles. In *Advances in Cryptology - EUROCRYPT 2005*, volume 3494 of *Lecture Notes in Computer Science*, pages 114–127, 2005.

[ZI07] Rui Zhang and Hideki Imai. Generic combination of public key encryption with keyword search and public key encryption. In *Cryptology and Network Security, 6th International Conference, CANS 2007*, volume 4856 of *Lecture Notes in Computer Science*, pages 159–174. Springer, 2007.

A Proof of Theorem 5.1

We prove this theorem by the following two lemmas.

Lemma A.1 *Assume the HIBE scheme is $(t_1, Q_{e_1}, \epsilon_1)$ IND-HIBE-CPA secure in selective-identity sense and the signature scheme is (t_3, ϵ_3) one-time sEUF-CMA secure. Then the PKE-PEKS scheme has $(t, Q_d, Q_w, Q_t, \epsilon)$ data privacy such that*

$$\epsilon \leq \epsilon_1 + \frac{1}{2}\epsilon_3, Q_d + Q_w + Q_t \leq Q_{e_1}, t \leq \min\{t_1, t_3\} - T_{kg} - T_{sig} - (Q_d + Q_t)(T_v + T_d) - Q_w T_{dk}$$

where T_{kg}, T_{sig}, and T_v are the maximum times for key generation, signing, and verification in the signature scheme, while T_{dk}, and T_d are the maximum times for private key extraction, and decryption time in the HIBE scheme.

Proof. Suppose there is an adversary $\mathcal{A}$ that has advantage ϵ against the data privacy of the PKE-PEKS construction. We construct an algorithm $\mathcal{B}$ against the selective-identity IND-HIBE-CPA security of the underlying HIBE scheme.

Setup: $\mathcal{B}$ runs OT.$\mathsf{KeyGen}(\kappa)$ to generate (vk^*, sk^*), then commits the target identity $id^* = 0||vk^*$ to its own challenger. Given mpk of the HIBE scheme, $\mathcal{B}$ sends mpk to $\mathcal{A}$ as pk_A.
Phase 1: Upon receiving the decryption queries, token queries, and test queries issued by $\mathcal{A}$, $\mathcal{B}$ responds as follows:

– Decryption query $\langle u \rangle$: $\mathcal{B}$ parses u as (vk, c, s, σ). If OT.$\mathsf{Verify}(vk, c||s, \sigma) = 0$ then $\mathcal{B}$ rejects this decryption query with $\perp$. Otherwise $\mathcal{B}$ proceeds as follows:
  - Case 1: $vk = vk^*$
    A forgery has been made against OT, call this event Forge. If Forge occurs, $\mathcal{B}$ aborts and outputs a random bit.
  - Case 2: $vk \neq vk^*$
    $\mathcal{B}$ obtains a private key dk for "identity" $0||vk$ (by issuing the private key extraction query $\langle 0||vk \rangle$ to its own challenger), then responds with HIBE.$\mathsf{Decrypt}(dk, c)$.

– Token query $\langle w \rangle$: $\mathcal{B}$ issues the private key extraction query $\langle 1||w \rangle$ to its own challenger and forwards the reply with w to $\mathcal{A}$.
– Test query $\langle u, w \rangle$: $\mathcal{B}$ first obtains a token t_w for w in the same way as it answers the token query $\langle w \rangle$, then responds with $\mathsf{Test}(t_w, u)$.

Challenge: $\mathcal{A}$ outputs two messages m_0 and m_1 and keyword w where it want to be challenged on. $\mathcal{B}$ proceeds as follows:

1. Submit two messages m_0 and m_1 to its own challenger, and get the encryption c^* of m_b under identity $id^* = 0||vk^*$, where b is a random bit chosen by $\mathcal{B}$'s challenger.
2. Compute $s^* \leftarrow \text{HIBE.}\mathsf{Encrypt}(pk_A, (1||w, vk^*), vk^*)$.
3. Compute $\sigma^* \leftarrow \text{OT.}\mathsf{Sign}(sk^*, c^*||s^*)$,
4. Send $u^* = (vk^*, c^*, s^*, \sigma^*)$ to $\mathcal{A}$ as the challenge ciphertext.

Phase 2: $\mathcal{A}$ can adaptively issue more decryption queries, token queries, and test queries. $\mathcal{B}$ proceeds the decryption queries in the same way as in Phase 1. $\mathcal{B}$ can answer all the token queries and test queries correctly since the private key queries of the form $\langle 1||w \rangle$ are always permitted by $\mathcal{B}$'s challenger.
Guess: As soon as $\mathcal{A}$ outputs its guess b' for b, $\mathcal{B}$ outputs b' to its own challenger.

$\mathcal{B}$ represents a legal strategy for attacking selective-identity IND-HIBE-CPA security of the underlying HIBE scheme. Provide the event Forge does not occur, $\mathcal{B}$ provides a perfect simulation for $\mathcal{A}$ so $\mathcal{B}$ succeeds with the same probability against the selective-identity IND-HIBE-CPA security of the HIBE scheme as $\mathcal{A}$ against the data privacy of the PKE-PEKS construction. If Forge does occur then $\mathcal{B}$ outputs a random bit and succeeds with probability 1/2. Let SuccB denote the event of $\mathcal{B}$ outputting the correct bit in the selective-identity IND-HIBE-CPA security experiment and SuccA denote the event of $\mathcal{A}$ outputting the correct bit in the data privacy experiment for PKE-PEKS, we have:

$$\left|\Pr[\mathsf{SuccB}] - \frac{1}{2}\right| = \left|\Pr[\mathsf{SuccA}]\Pr[\overline{\mathsf{Forge}}] + \frac{1}{2}\Pr[\mathsf{Forge}] - \frac{1}{2}\right|$$

Since the HIBE scheme is (t_1, Q_e, ϵ_1) selective-identity IND-HIBE-CPA secure and the signature scheme is (t_3, ϵ_3) one-time sEUF-CMA secure, thus $|\Pr[\mathsf{SuccB}] - 1/2| \leq \epsilon_1$ and $\Pr[\mathsf{Forge}] \leq \epsilon_3$. It follows that:

$$\begin{aligned}
\epsilon &= \left|\Pr[\mathsf{SuccA}] - \frac{1}{2}\right| \\
&= \left|\Pr[\mathsf{SuccA} \wedge \mathsf{Forge}] + \Pr[\mathsf{SuccA} \wedge \overline{\mathsf{Forge}}] - \frac{1}{2}\Pr[\mathsf{Forge}] + \frac{1}{2}\Pr[\mathsf{Forge}] - \frac{1}{2}\right| \\
&\leq \left|\Pr[\mathsf{SuccA} \wedge \mathsf{Forge}] - \frac{1}{2}\Pr[\mathsf{Forge}]\right| + \left|\Pr[\mathsf{SuccA} \wedge \overline{\mathsf{Forge}}] + \frac{1}{2}\Pr[\mathsf{Forge}] - \frac{1}{2}\right| \\
&\leq \frac{1}{2}\Pr[\mathsf{Forge}] + \left|\Pr[\mathsf{SuccA} \wedge \overline{\mathsf{Forge}}] + \frac{1}{2}\Pr[\mathsf{Forge}] - \frac{1}{2}\right| \\
&= \frac{1}{2}\Pr[\mathsf{Forge}] + \left|\Pr[\mathsf{SuccB}] - \frac{1}{2}\right| \leq \frac{1}{2}\epsilon_3 + \epsilon_1
\end{aligned}$$

The running time of $\mathcal{A}$ is at most $\min\{t_1, t_3\} - T_{kg} - T_{sig} - (Q_d + Q_t)(T_v + T_d) - Q_w T_{dk}$. Throughout the simulation $\mathcal{B}$ issues at most $Q_d + Q_w + Q_t$ private key extraction queries to its own challenger. This proves the Lemma A.1. □

Lemma A.2 *Assume the HIBE scheme is $(t_2, Q_{e_2}, \epsilon_2)$ ANO-HIBE-CPA anonymous at level 1 and the signature scheme is (t_3, ϵ_3) one-time sEUF-CMA secure. Then the PKE-PEKS scheme has $(t, Q_d, Q_w, Q_t, \epsilon)$ keyword privacy such that*

$$\epsilon \leq \epsilon_2 + \frac{1}{2}\epsilon_3, Q_d + Q_w + Q_t \leq Q_{e_2}, t \leq \min\{t_2, t_3\} - T_{kg} - T_{sig} - (Q_d + Q_t)(T_v + T_d) - Q_w T_{dk}$$

where T_{kg}, T_{sig}, T_v, T_{dk}, and T_d are defined as above.

Proof. Suppose there is an adversary $\mathcal{A}$ that has advantage ϵ against the keyword privacy of the PKE-PEKS construction. Then we can construct an algorithm $\mathcal{B}$ that breaks the ANO-HIBE-CPA anonymity at level 1 of the underlying HIBE scheme.

Setup: $\mathcal{B}$ is given mpk of the HIBE scheme. $\mathcal{B}$ forwards it to $\mathcal{A}$ as the public key pk_A.
Phase 1: Upon receiving the token queries, test queries, and decryption queries issued by $\mathcal{A}$, $\mathcal{B}$ responds as follows:

- Token query $\langle w \rangle$: $\mathcal{B}$ issues the private key extraction query $\langle 1||w \rangle$ to its own challenger and forwards the reply with w to $\mathcal{A}$.
- Test query $\langle u, w \rangle$: $\mathcal{B}$ obtains a token t_w for w in the same way as it answers the token query $\langle w \rangle$, then responds with $\mathsf{Test}(t_w, u)$.
- Decryption query $\langle u \rangle$: $\mathcal{B}$ parses u as (vk, c, s, σ). If OT.$\mathsf{Verify}(vk, c||s, \sigma) = 0$, then $\mathcal{B}$ rejects this decryption query with $\perp$. If not, $\mathcal{B}$ obtains a private key dk for "identity" $0||vk$ (by issuing the private key extraction query $\langle 0||vk \rangle$ to its own challenger), then responds with HIBE.$\mathsf{Decrypt}(dk, c)$.

Challenge: When $\mathcal{A}$ outputs a message m, two keywords w_0^* and w_1^* where it want to be challenged on, $\mathcal{B}$ proceeds as follows:

1. Run $(vk^*, sk^*) \leftarrow$ OT.$\mathsf{KeyGen}(\kappa)$, set $id^* = 0||vk^*$.
2. Compute $c^* \leftarrow$ HIBE.$\mathsf{Encrypt}(pk_A, id^*, m)$.
3. Submit vk^* (as the message) and two target identities $(1||w_0^*, vk^*)$ and $(1||w_1^*, vk^*)$ to its own challenger, and get the encryption s^* of vk^* under identity $(1||w_b^*, vk^*)$, where b is a random bit chosen by $\mathcal{B}$'s challenger.
4. Compute $\sigma^* \leftarrow$ OT.$\mathsf{Sign}(sk^*, c^*||s^*)$.
5. Send $u^* = (vk^*, c^*, s^*, \sigma^*)$ to $\mathcal{A}$ as the challenge ciphertext.

Phase 2: $\mathcal{A}$ may issue more token queries, test queries, and decryption queries in Phase 2, $\mathcal{B}$ responds as follows:

- Token query $\langle w \rangle$: as long as $w \neq w_0^*, w_1^*$, $\mathcal{B}$ can always issue the private key extraction query for identity $1||w$, then sends the reply with w to $\mathcal{A}$.
- Test query $\langle u, w \rangle$: test queries $\langle u^*, w_0^* \rangle$ and $\langle u^*, w_1^* \rangle$ will be rejected. $\mathcal{B}$ parses u as (vk, c, s, σ), first checks if OT.$\mathsf{Verify}(vk, c||s, \sigma) = 0$. If so, $\mathcal{B}$ returns 0. If not, when w is not w_0^* or w_1^*, $\mathcal{B}$ obtains a token t_w for w in the same way as it answers the token query $\langle w \rangle$ and then responds with $\mathsf{Test}(t_w, u)$. Otherwise, $\mathcal{B}$ proceeds as follows:
  - Case 1: $vk = vk^*$
    A forgery has been made against OT (since in this stage w is either w_0^* or w_1^*, thus for a legal query we have $u \neq u^*$), call this event Forge. $\mathcal{B}$ aborts and outputs a random bit.
  - Case 2: $vk \neq vk^*$
    $\mathcal{B}$ obtains the private key dk for identity $(1||w, vk)$, returns 1 if $vk =$ HIBE.$\mathsf{Decrypt}(dk, s)$ and 0 otherwise.
- Decryption query $\langle u \rangle$: $\mathcal{B}$ responds to the decryption queries as it did in Phase 1. Since the private key extraction queries of the form $\langle 0||vk \rangle$ are always permitted, $\mathcal{B}$ can answer all the decryption queries correctly.

Guess. As soon as $\mathcal{A}$ outputs its guess b' for b, $\mathcal{B}$ outputs b' to its own challenger.

$\mathcal{B}$ represents a legal strategy for attacking the ANO-HIBE-CPA anonymity of the underlying HIBE scheme. Provide the event Forge does not occur, then $\mathcal{A}$ is playing exactly the keyword privacy experiment of PKE-PEKS. If Forge does occur then $\mathcal{B}$ outputs a random bit and succeeds with probability 1/2. Let SuccB denote the event of $\mathcal{B}$ outputting the correct bit in the ANO-HIBE-CPA anonymity experiment for the underlying HIBE scheme, and SuccA denote the event of $\mathcal{A}$ succeeds in the above experiment, we have:

$$\left|\Pr[\mathsf{SuccB}] - \frac{1}{2}\right| = \left|\Pr[\mathsf{SuccA}]\Pr[\overline{\mathsf{Forge}}] + \frac{1}{2}\Pr[\mathsf{Forge}] - \frac{1}{2}\right|$$

Since the HIBE scheme is $(t_2, Q_{e_2}, \epsilon_2)$ ANO-HIBE-CPA anonymous at level 1 and the signature scheme is (t_3, ϵ_3) one-time sEUF-CMA secure, thus $|\Pr[\mathsf{SuccB}] - 1/2| \le \epsilon_2$ and $\Pr[\mathsf{Forge}] \le \epsilon_3$. Let $|\Pr[\mathsf{SuccA}] - 1/2| = \epsilon$. For a similar derivation as we did in the proof for Lemma A.1, we have that $\epsilon \le \epsilon_2 + \epsilon_3/2$. The running time of $\mathcal{A}$ is at most $\min\{t_2, t_3\} - T_{kg} - T_{sig} - (Q_d + Q_t)(T_v + T_d) - Q_w T_{dk}$ Throughout the simulation $\mathcal{B}$ issues at most $Q_d + Q_w + Q_t$ private key extraction queries to its own challenger. This proves Lemma A.2. □

Theorem 5.1 immediately follows from Lemma A.1 and Lemma A.2. □

B Proof of Theorem 6.1

We prove this theorem by the following two lemmas.

Lemma B.1 *Assume the IBE scheme is $(t_1, Q_{e_1}, \epsilon_1)$ IND-IBE-CPA secure in selective-identity sense and the signature scheme is (t_3, ϵ_3) one-time sEUF-CMA secure. Then the PKE-PEKS scheme has $(t, Q_d, Q_w, Q_t, \epsilon)$ data privacy such that*

$$\epsilon \le \epsilon_1 + \frac{1}{2}\epsilon_3, Q_d + Q_w + Q_t \le Q_{e_1}, t \le \min\{t_1, t_3\} - T_{kg} - T_{sig} - (Q_d + Q_t)(T_v + T_d) - Q_w T_{dk}$$

where T_{kg}, T_{sig}, and T_v are the maximum times for key generation, signing, and verification in the signature scheme, while T_{dk}, and T_d are the maximum times for private key extraction, and decryption time in the IBE scheme.

Proof. Suppose there is an adversary $\mathcal{A}$ that has advantage against the data privacy of the PKE-PEKS construction. We construct an algorithm $\mathcal{B}$ against the IND-IBE-CPA security in selective-identity sense of the underlying IBE scheme.

Setup: $\mathcal{B}$ runs OT.$\mathsf{KeyGen}(\kappa)$ to generate (vk^*, sk^*), then commits the "target" identity $id^* = 0||vk^*$ to its own challenger. Given mpk of the IBE scheme, $\mathcal{B}$ sends mpk to $\mathcal{A}$ as pk_A.
Phase 1: Upon receiving the decryption queries, token queries, and test queries issued by $\mathcal{A}$, $\mathcal{B}$ responds as follows:

- Decryption query $\langle u \rangle$: $\mathcal{B}$ parses u as (vk, c, s, σ). If OT.$\mathsf{Verify}(vk, c||s, \sigma) = 0$ then $\mathcal{B}$ rejects this decryption query with $\perp$. Otherwise $\mathcal{B}$ proceeds as follows:
  - Case 1: $vk = vk^*$
    A forgery has been made against OT, call this event Forge. If Forge occurs, $\mathcal{B}$ aborts and outputs a random bit.
  - Case 2: $vk \ne vk^*$
    $\mathcal{B}$ obtains a private key dk for "identity" $0||vk$ (by issuing the private key extraction query $\langle 0||vk \rangle$ to its own challenger), then responds with IBE.$\mathsf{Decrypt}(dk, c)$.

– Token query $\langle w \rangle$: $\mathcal{B}$ issues the private key extraction query $\langle 1||w \rangle$ to its own challenger and forwards the reply to $\mathcal{A}$.
– Test query $\langle u, w \rangle$: $\mathcal{B}$ first obtains a token t_w for w in the same way as it answers the token query $\langle w \rangle$, then responds with $\mathsf{Test}(t_w, u)$.

Challenge: $\mathcal{A}$ outputs two messages m_0 and m_1 and keyword w where it want to be challenged on. $\mathcal{B}$ proceeds as follows:

1. Submit two messages m_0 and m_1 to its own challenger, and get the encryption c^* of m_b under identity $id^* = 0||vk^*$, where b is a random bit chosen by $\mathcal{B}$'s advantage.
2. Compute $s^* \leftarrow \text{IBE}.\mathsf{Encrypt}(pk_A, 1||w, vk^*)$.
3. Compute $\sigma^* \leftarrow \text{OT}.\mathsf{Sign}(sk^*, c^*||s^*)$,
4. Send $u^* = (vk^*, c^*, s^*, \sigma^*)$ to $\mathcal{A}$ as the challenge ciphertext.

Phase 2: $\mathcal{A}$ can issue more token queries, more test queries, and more decryption queries. Note that $\mathcal{B}$ can answer all the token queries and test queries correctly since the private key queries of the form $\langle 1||w \rangle$ are always permitted in its security experiment. $\mathcal{B}$ proceeds the decryption queries in the same way as in Phase 1.
Guess: $\mathcal{A}$ outputs its guess b' for b. $\mathcal{B}$ outputs b' to its own challenger.

$\mathcal{B}$ represents a legal strategy for attacking the selective-identity IND-IBE-CPA security of the underlying IBE scheme. Provide the event Forge does not occur, $\mathcal{B}$ provides a perfect simulation for $\mathcal{A}$ so $\mathcal{B}$ succeeds with the same probability against the selective-identity IND-IBE-CPA security of the IBE scheme as $\mathcal{A}$ against the data privacy of the PKE-PEKS construction. If Forge does occur then $\mathcal{B}$ outputs a random bit and succeeds with probability 1/2. Let SuccB denote the event of $\mathcal{B}$ outputting the correct bit in the selective-identity IND-IBE-CPA security experiment and SuccA denote the event of $\mathcal{A}$ outputting the correct bit in the data privacy experiment for PKE-PEKS, we have:

$$\left|\Pr[\mathsf{SuccB}] - \frac{1}{2}\right| = \left|\Pr[\mathsf{SuccA}]\Pr[\overline{\mathsf{Forge}}] + \frac{1}{2}\Pr[\mathsf{Forge}] - \frac{1}{2}\right|$$

Since the IBE scheme is (t_1, Q_e, ϵ_1) IND-IBE-CPA secure and the signature scheme is (t_3, ϵ_3) one-time sEUF-CMA secure, thus $|\Pr[\mathsf{SuccB}] - 1/2| \leq \epsilon_1$ and $\Pr[\mathsf{Forge}] \leq \epsilon_3$. Let $|\Pr[\mathsf{SuccA}] - 1/2| = \epsilon$. For a similar derivation as we did in the proof for Lemma A.1, we have that $\epsilon \leq \epsilon_1 + \epsilon_3/2$. The running time of $\mathcal{A}$ is at most $\min\{t_1, t_3\} - T_{kg} - T_{sig} - (Q_d + Q_t)(T_v + T_d) - Q_w T_{dk}$. Throughout the simulation $\mathcal{B}$ issues at most $Q_d + Q_w + Q_t$ private key extraction queries to its own challenger. This proves the Lemma B.1. □

Lemma B.2 *Assume the IBE scheme is $(t_2, Q_{e_2}, Q_{d_2}, \epsilon_2)$ ANO-IBE-CCA anonymous and WROB-CCA robust, and the signature scheme is (t_3, ϵ_3) one-time sEUF-CMA secure. Then the PKE-PEKS scheme has $(t, Q_d, Q_w, Q_t, \epsilon)$ keyword privacy such that:*

$$\epsilon \leq \epsilon_2 + \frac{1}{2}\epsilon_3, Q_d + Q_w \leq Q_{e_2}, Q_t \leq Q_{d_2}, t \leq \min\{t_2, t_3\} - T_{kg} - T_{sig} - (Q_d + Q_t)(T_v + T_d) - Q_w T_{dk}$$

where T_{kg}, T_{sig}, T_v, T_{dk}, and T_d are defined as above.

Proof. Suppose there is an adversary $\mathcal{A}$ that has advantage ϵ against the keyword privacy of the PKE-PEKS construction. We construct an algorithm $\mathcal{B}$ that breaks the ANO-IBE-CCA anonymity of the underlying IBE scheme.

Setup: $\mathcal{B}$ is given mpk of the IBE scheme. $\mathcal{B}$ forwards it to $\mathcal{A}$ as the public key pk_A.
Phase 1: Upon receiving the token queries, test queries, and decryption queries issued by $\mathcal{A}$, $\mathcal{B}$ responds as follows:

– Token query $\langle w \rangle$: $\mathcal{B}$ issues the private key extraction query $\langle 1||w \rangle$ to its own challenger and forwards the reply to $\mathcal{A}$.
– Test query $\langle u, w \rangle$: $\mathcal{B}$ parses u as (vk, c, s, σ). If OT.$\mathsf{Verify}(vk, c||s, \sigma) = 0$, $\mathcal{B}$ returns 0. Else, $\mathcal{B}$ issues decryption query $\langle 1||w, s \rangle$ to its own challenger. $\mathcal{B}$ returns 1 if the reply is vk and 0 otherwise.
– Decryption query $\langle u \rangle$: $\mathcal{B}$ parses u as (vk, c, s, σ). If OT.$\mathsf{Verify}(vk, c||s, \sigma) = 0$, then $\mathcal{B}$ rejects this decryption query with $\perp$. If not, $\mathcal{B}$ obtains a private key dk for "identity" $0||vk$ (by issuing the private key extraction query $\langle 0||vk \rangle$ to its own challenger), then responds with IBE.$\mathsf{Decrypt}(dk, c)$.

Challenge: When $\mathcal{A}$ outputs a message m, two target keywords w_0^* and w_1^* where it want to be challenged on, $\mathcal{B}$ proceeds as follows:
1. Run $(vk^*, sk^*) \leftarrow$ OT.$\mathsf{KeyGen}(\kappa)$, set $id^* = 0||vk^*$.
2. Compute $c^* \leftarrow$ IBE.$\mathsf{Encrypt}(pk_A, id^*, m)$.
3. Submit vk^* (as the message) and two target identities $1||w_0^*$ and $1||w_1^*$ to its own challenger, and get the response s^*.
4. Compute $\sigma^* \leftarrow$ OT.$\mathsf{Sign}(sk^*, c^*||s^*)$.
5. Send the challenge ciphertext $u^* = (vk^*, c^*, s^*, \sigma^*)$ to $\mathcal{A}$.

Phase 2: $\mathcal{A}$ may issue more token queries, decryption queries, and test queries in Phase 2, $\mathcal{B}$ responds as follows:
– Token query $\langle w \rangle$: as long as $w \neq w_0^*, w_1^*$, $\mathcal{B}$ can always issue the private key extraction query for identity $1||w$, then forwards the reply to $\mathcal{A}$.
– Test query $\langle u, w \rangle$: test queries $\langle u^*, w_0^* \rangle$ and $\langle u^*, w_1^* \rangle$ will be rejected. For a legal test query, $\mathcal{B}$ parses u as (vk, c, s, σ), first checks if OT.$\mathsf{Verify}(vk, c||s, \sigma) = 1$. If not, $\mathcal{B}$ returns 0. If so, when w is not w_0^* or w_1^*, $\mathcal{B}$ issues decryption query $\langle 1||w, s \rangle$ to its own challenger, returns 1 if the reply is vk and 0 if not. Otherwise, $\mathcal{B}$ proceeds as follows:
  - Case 1: $vk = vk^*$
    A forgery has been made against OT (since in this stage w is either w_0^* or w_1^*, thus for a legal query we have $u \neq u^*$), call this event Forge. $\mathcal{B}$ aborts and outputs a random bit.
  - Case 2: $vk \neq vk^*$
    If $s \neq s^*$, $\mathcal{B}$ makes decryption query $\langle 1||w, s \rangle$ to its own challenger, then returns 1 if the reply is vk and 0 otherwise. If $s = s^*$, $\mathcal{B}$ returns 0. The correctness of $\mathcal{B}$'s response for case $s = s^*$ is based on the following fact: if $w = w_b^*$, the decryption of s^* under identity "$1||w$" is exactly vk^* (differs from vk); if $w \neq w_b^*$, the decryption of s^* under identity "$1||w$" is $\perp$ due to the weak robustness of the underlying IBE scheme. Thus the PEKS test fails in both cases.
– Decryption query $\langle u \rangle$: $\mathcal{B}$ responds to the decryption queries as it did in Phase 1. Since the private key extraction queries of the form $\langle 0||vk \rangle$ are always permitted, $\mathcal{B}$ can answer all the decryption queries correctly.

Guess. As soon as $\mathcal{A}$ outputs its guess b' for b, $\mathcal{B}$ outputs b' to its own challenger.

$\mathcal{B}$ represents a legal strategy for attacking the ANO-IBE-CPA anonymity of the underlying IBE scheme. Provide the event Forge does not occur, $\mathcal{B}$ provides a perfect simulation for $\mathcal{A}$ so $\mathcal{B}$ succeeds with the same probability against the ANO-IBE-CCA anonymity of the IBE scheme as $\mathcal{A}$ against the keyword privacy of the PKE-PEKS construction. If Forge does occur then $\mathcal{B}$ outputs a random bit and succeeds with probability 1/2. Let SuccB denote the event of $\mathcal{B}$ outputting the correct bit in the ANO-IBE-CCA anonymity experiment, and SuccA denote the event of $\mathcal{A}$ succeeds in the above experiment, we have that:

$$\left|\Pr[\mathsf{SuccB}] - \frac{1}{2}\right| = \left|\Pr[\mathsf{SuccA}]\Pr[\overline{\mathsf{Forge}}] + \frac{1}{2}\Pr[\mathsf{Forge}] - \frac{1}{2}\right|$$

Since the IBE scheme is $(t_2, Q_{e_2}, Q_{d_2}, \epsilon_2)$ ANO-IBE-CCA anonymous and the signature scheme is (t_3, ϵ_3) one-time sEUF-CMA secure, thus $|\Pr[\mathsf{SuccB}] - 1/2| \leq \epsilon_2$ and $\Pr[\mathsf{Forge}] \leq \epsilon_3$. Let $|\Pr[\mathsf{SuccA}] - 1/2| = \epsilon$. For a similar derivation as we did in the proof for Lemma A.1, we conclude that $\epsilon \leq \epsilon_2 + \epsilon_3/2$. The running time of $\mathcal{A}$ is at most $\min\{t_2, t_3\} - T_{kg} - T_{sig} - (Q_d + Q_t)(T_v + T_d) - Q_w T_{dk}$ Throughout the simulation $\mathcal{B}$ issues at most $Q_d + Q_w$ private key extraction queries and at most Q_t decryption queries to its own challenger. This proves Lemma B.2. □

Theorem 6.1 immediately follows from Lemma B.1 and Lemma B.2. □

C Instantiations

C.1 Pairing-based Instantiation of Construction 1

De Caro *et al.* [CIP10] constructed a CPA-secure and anonymous HIBE in composite bilinear groups in the standard model. We give a two-level version of the HIBE scheme [CIP10] as below:

- $\mathsf{Setup}(\kappa, 2)$: take as input a security parameter κ, generate four primes p_1, p_2, p_3, p_4, two groups $\mathbb{G}$ and $\mathbb{G}_T$ of order $N = p_1 p_2 p_3 p_4$. Let e be a bilinear map from $\mathbb{G}$ to $\mathbb{G}_T$, and $\mathbb{G}_{p_i}$ be the subgroup of $\mathbb{G}$ with order p_i for $1 \leq i \leq 4$. Randomly choose $Y_1, X_1, u_1, u_2 \in \mathbb{G}_{p_1}$, $Y_3 \in \mathbb{G}_{p_3}$, $X_4, Y_4 \in \mathbb{G}_{p_4}$ and $\alpha \in \mathbb{Z}_N$, output the master key pair:
$$mpk = (N, Y_1, Y_3, Y_4, t = X_1 X_4, u_1, u_2, W = e(Y_1, Y_1)^\alpha); \quad msk = (X_1, \alpha)$$
- $\mathsf{Extract}(msk, id)$: take as input msk and an level-1 identity $id = (id_1)$ where $id_1 \in \mathbb{Z}_{p_1}$, randomly choose $r_1, r_2 \in \mathbb{Z}_N$, and $R_{1,1}, R_{1,2}, R_{2,1}, R_{2,2} \in \mathbb{G}_{p_3}$, then compute:
$$K_{1,1} = Y_1^{r_1} R_{1,1}, K_{1,2} = Y_1^{\alpha} (u_1^{id} X_1)^{r_1} R_{1,2}, E_{1,2} = u_2^{r_1} R_{1,2}$$
$$K_{2,1} = Y_1^{r_2} R_{2,1}, \quad K_{2,2} = (u_1^{id} X_1)^{r_2} R_{2,2}, \quad E_{2,2} = u_2^{r_2} R_{2,2}$$
output the private key $dk_{id} = (K_{1,1}, K_{1,2}, E_{1,2}, K_{2,1}, K_{2,2}, E_{2,2})$.
- $\mathsf{Derive}(dk_{id}, id')$: take as input a private key $dk_{id} = (K_{1,1}, K_{1,2}, E_{1,2}, K_{2,1}, K_{2,2}, E_{2,2})$ for a level-1 identity $id = (id_1)$ and a level-2 identity $\tilde{id} = (id_1, id_2)$, randomly chooses $\tilde{r}_1 \in \mathbb{Z}_N$, and and $\tilde{R}_{1,1}, \tilde{R}_{1,2} \in \mathbb{G}_{p_3}$. Then compute
$$K'_{1,1} = K_{1,1}(K_{2,1})^{\tilde{r}_1} \tilde{R}_{1,1}; \quad K'_{1,2} = K_{1,2}(K_{2,2})^{\tilde{r}_1} (E_{1,2})^{id_2} (E_{2,2})^{\tilde{r}_1 id_2} \tilde{R}_{1,2}$$
return the secret key $dk_{id} = (K'_{1,1}, K'_{1,2})$. Note that for a level-2 identity, it's not necessary to generate any other secret elements, say E_i's as in the output of $\mathsf{Extract}$ algorithm, since we don't need to derive a private key for the next level (no such a level exists).
- $\mathsf{Encrypt}(mpk, id, m)$: take as input mpk, an identity $id = (id_1, \ldots, id_j)$ (here $1 \leq j \leq 2$), and a message $m \in \mathbb{G}_T$, choose $s \xleftarrow{R} \mathbb{Z}_N$ and $Z, Z' \xleftarrow{R} \mathbb{G}_{p_4}$, then compute:
$$c_0 = m \cdot W^s, \qquad c_1 = (u_1^{id_1} \cdots u_j^{id_j} t)^s Z, \qquad c_2 = Y_1^s Z'$$
return the ciphertext $c = (c_0, c_1, c_2)$.
- $\mathsf{Decrypt}(dk_{id}, c)$: take as input a private key dk_{id} for identity id and a ciphertext $c = (c_0, c_1, c_2)$, compute:
$$m = c_0 \cdot \frac{e(K_{1,1}, c_1)}{e(K_{1,2}, c_2)}$$
return the message m.

The collision-freeness of the above scheme is easy to be verified. As to the candidate of one-time signature, we recommend Boneh and Boyen short signature scheme [BB08], which is sEUF-CMA secure based on the strong Diffie-Hellman (SDH) assumption in the bilinear groups in the standard model. By employing these two schemes as the underlying primitives for our first generic construction, we obtain a pairing-based PKE-PEKS scheme which is jointly CCA-secure and consistent in the standard model.

C.2 Lattice-based Instantiation of Construction 1

Agrawal *et al.* [ABB10] proposed a CPA-secure and anonymous HIBE scheme based on the learning with errors (LWE) assumption in the standard model. Their original scheme encrypts just one bit at one time and is only selective-identity secure and anonymous. As the authors of [ABB10] pointed out, their scheme can be simply extended to support multi-bit encryption by reusing the randomness of the encryption algorithm, and can be bootstrapped to an adaptive-identity secure and anonymous scheme by applying a similar technique used in [Wat05, Boy10].

Before we presenting the two-level version of the HIBE scheme in [ABB10], we first recall several related algorithms on lattices in the sense of their functionality (since we try not to involve too much knowledge on lattices here).

- $\mathsf{TrapGen}(n, q)$: take as input two integers n and q, output a matrix $\mathbf{A} \in \mathbb{Z}_q^{n\times m}$ and a trapdoor $\mathbf{T_A} \in \mathbb{Z}^{m\times m}$, such that $\mathbf{A}\mathbf{T_A} = \mathbf{0} \mod q$ and the norm of $\mathbf{T_A}$ is very short.
- $\mathsf{SampleBasisLeft}(\mathbf{A}, \mathbf{B}, \mathbf{T_A}, \sigma)$: take as input two matrices $\mathbf{A} \in \mathbb{Z}_q^{n\times m_1}$, $\mathbf{B} \in \mathbb{Z}_q^{n\times m_2}$, a trapdoor $\mathbf{T_A} \in \mathbb{Z}^{m_1\times m_1}$ of $\mathbf{A}$, and a real σ, output a trapdoor $\mathbf{T_F}$ of $\mathbf{F} = (\mathbf{A}||\mathbf{B}) \in \mathbb{Z}_q^{n\times(m_1+m_2)}$, where the norm of $\mathbf{T_F}$ is determined by the quality of $\mathbf{T_A}$ (i.e., the smaller the better) and the size of σ.
- $\mathsf{SamplePre}(\mathbf{A}, \mathbf{T_A}, \mathbf{u}, \tau)$: take as input a matrix $\mathbf{A} \in \mathbb{Z}_q^{n\times m}$, a trapdoor $\mathbf{T_A} \in \mathbb{Z}^{m\times m}$ of $\mathbf{A}$, a vector $\mathbf{u} \in \mathbb{Z}_q^n$ and a real τ, output a short vector $\mathbf{e} \in \mathbb{Z}^{m\times m}$ such that $\mathbf{A}\mathbf{e} = \mathbf{u} \mod q$, where the norm of $\mathbf{e}$ critically relies on the quality of $\mathbf{T_A}$ and the size of τ.

For more details about the parameters and the algorithms, we refer the readers to [ABB10].

For simplicity, we assume the level-2 identity space is $\{-1, 1\}^l \times \{-1, 1\}^l$, the adaptive-identity secure and anonymous HIBE encrypting one bit is given below:

- $\mathsf{Setup}(\kappa, 2)$: take as input a security parameter κ, generate a $n \times m$ matrix $\mathbf{A}_0 \in \mathbb{Z}^{n\times m}$ and its trapdoor $\mathbf{T}_{\mathbf{A}_0}$ using the $\mathsf{TrapGen}$ algorithm. Then, randomly choose $2(l + 1)$ matrices $\{\mathbf{A}_i, \{\mathbf{B}_{i,j}\}_{j=1,\dots,l}\}_{i=1,2}$ from $\mathbb{Z}_q^{n\times m}$. Finally, randomly choose a vector $\mathbf{u} \leftarrow \mathbb{Z}_q^n$, output the master key pair:

$$mpk = (\mathbf{A}_0, \{\mathbf{A}_i, \{\mathbf{B}_{i,j}\}_{j=1,\dots,l}\}_{i=1,2}, \mathbf{u})\,; \quad msk = (\mathbf{T}_{\mathbf{A}_0}) \in \mathbb{Z}^{m\times m}$$

- $\mathsf{Extract}(msk, id)$: take as input msk and an identity $id = (id_1)$ where $id_1 = (b_{1,1}, \dots, b_{1,l}) \in \{-1, 1\}^l$, define matrix $\mathbf{F}_{id} = (\mathbf{A}_0||\mathbf{A}_1 + \sum_{j=1}^{l} b_{1,j}\mathbf{B}_{1,j})$, pick a real σ, then compute:

$$dk_{id} \leftarrow \mathsf{SampleBasisLeft}\big(\mathbf{A}_0, \mathbf{A}_1 + \sum_{j=1}^{l} b_{1,j}\mathbf{B}_{1,j}, msk, \sigma\big)$$

  output the trapdoor dk_{id} of $\mathbf{T}_{\mathbf{F}_{id}}$ as the private key for id.
- $\mathsf{Derive}(dk_{id}, id')$: take as input a private key dk_{id} for identity $id = (id_1)$ and an identity $id' = (id_1, id_2)$ where $id_2 = (b_{2,1}, \dots, b_{2,l}) \in \{-1, 1\}^l$, define $\mathbf{F}_{id'} = (\mathbf{F}_{id}||\mathbf{A}_2 + \sum_{j=1}^{l} b_{2,j}\mathbf{B}_{2,j})$, then output the private key $dk_{id'}$ for id' as:

$$dk_{id'} \leftarrow \mathsf{SampleBasisLeft}\big(\mathbf{F}_{id'}, \mathbf{A}_2 + \sum_{j=1}^{l} b_{2,j}\mathbf{B}_{2,j}, dk_{id}, \sigma\big)$$

- $\mathsf{Encrypt}(mpk, id, m)$: take as input mpk, an identity id, and a message $m \in \{0, 1\}$, first define the corresponding matrix $\mathbf{F}_{id}$ as in the algorithm $\mathsf{Extract}$ (i.e, a level-1 identity) or Derive (i.e, a level-2 identity), then randomly choose a vector $\mathbf{s} \in \mathbb{Z}_q^n$ and a matrix $\mathbf{R} \in \{-1, 1\}^{m\times km}$, where $k = 1$ for level-1 identity, and $k = 2$ for level-2 identity. Finally, chooses a noise vector $x \in \chi, \mathbf{y} \in \chi^m$, compute the ciphertext:

$$c = \big(c_0 = \mathbf{u}^T\mathbf{s} + x + m\left\lfloor \frac{q}{2} \right\rfloor, c_1 = \mathbf{F}_{id}^T\mathbf{s} + \mathbf{R}^T\mathbf{y}\big)$$

– $\mathsf{Decrypt}(dk_{id}, c)$: take as input a private key dk_{id} for identity id and a ciphertext $c = (c_0, c_1)$, first set $\tau = \sigma\sqrt{(k+1)m}\omega(\sqrt{\log(km)})$, then compute:

$$\mathbf{e}_{id} \leftarrow \mathsf{SamplePre}(\mathbf{F}_{id}, dk_{id}, \mathbf{u}, \tau)$$

let $w = c_0 - \mathbf{e}_{id}^T c_1$, if $|w - \lfloor \frac{q}{2} \rfloor| < \lfloor \frac{q}{4} \rfloor$ output 1, otherwise output 0.

The CPA-security and anonymity of the above two-level HIBE scheme is implied by the original results of [ABB10], while the collision-freeness is easy to be verified, Very recently, Micciancio and Peikert [MP12] constructed a short signature scheme, which is *static* sEUF-CMA secure based on the small integer solution (SIS) assumption on lattices in the standard model. Using a family of chameleon hash functions, there is a generic transform [KR00] from static sEUF-CMA to sEUF-CMA security. A suitable type of chameleon hash function [CHKP10] has been constructed under a weak hardness-of-SIS assumption. By employing these two schemes as the underlying primitives for our first generic construction, we obtain a lattice-based PKE-PEKS scheme which is jointly CCA-secure and consistent in the standard model.

C.3 Pairing-based Instantiation of Construction 2

We now give an instantiation of our second generic construction by employing Gentry IBE [Gen06] together with Boneh-Boyen signature scheme [BB08].

$\mathsf{KeyGen}(\kappa)$: Pick two groups $\mathbb{G}$ and $\mathbb{G}_T$ of prime order p with a bilinear map $e : \mathbb{G} \times \mathbb{G} \rightarrow \mathbb{G}_T$. Choose random generators $g, h_1, h_2, h_3 \in \mathbb{G}$ and a random $\alpha \in \mathbb{Z}_p$, set $g_1 = g^\alpha$; choose two collision-resistant hash functions $H_1 : \{0,1\}^* \rightarrow \mathbb{Z}_p$ and $H_2 : \{0,1\}^* \rightarrow \mathbb{G}_T$; set the public key $pk_A = (g, g_1, h_1, h_2, h_3, H_1, H_2)$ and the secret key $sk_A = \alpha$.

$\mathsf{Encrypt}(pk_A, m, w)$:

1. Choose a random generator $t \in \mathbb{G}$, and random integers $x, y \xleftarrow{R} \mathbb{Z}_p^*$, compute $u = t^x$, $v = t^y$, and then set $vk = (t, u, v)$, $sk = (x, y)$.
2. Compute $id_0 = H_1(0||vk)$, choose $r_0 \xleftarrow{R} \mathbb{Z}_p$, and compute $c_1 = g_1^{r_0} g^{-r_0 \cdot id_0}$, $c_2 = m \cdot e(g, h_1)^{-r_0}$; set $c = (c_1, c_2)$.
3. Compute $id_1 = H_1(1||w)$, choose $r_1 \xleftarrow{R} \mathbb{Z}_p$, and compute $s_1 = g_1^{r_1} g^{-r_1 \cdot id_1}$, $s_2 = e(g,g)^{r_1}$, $s_3 = H_2(vk) \cdot e(g, h_1)^{-r_1}$, $s_4 = e(g, h_2)^{r_1} e(g, h_3)^{r_1 \beta}$ for $\beta = H_1(s_1, s_2, s_3)$; set $s = (s_1, s_2, s_3, s_4)$.
4. Compute $z = H_1(c||s)$, pick $\sigma_1 \xleftarrow{R} \mathbb{Z}_p \backslash \{-\frac{x+z}{y}\}$, and compute $\sigma_2 = t^{1/(x+z+y\sigma_1)}$, here the inverse is computed modulo p; set the signature $\sigma = (\sigma_1, \sigma_2)$.
5. Return the final ciphertext $u = (vk, c, s, \sigma)$.

$\mathsf{Decrypt}(sk_A, u)$:

1. Parse u as (vk, c, s, σ), where $vk = (t, u, v)$, and $\sigma = (\sigma_1, \sigma_2)$.
2. Compute $z = H_1(c||s)$. If $e(\sigma_2, u \cdot t^z \cdot v^{\sigma_1}) \neq e(t, t)$, return $\perp$. Otherwise, compute $id_0 = H_1(0||vk)$ and return $m = c_2 \cdot e(c_1, h^{1/(\alpha - id_0)})$.

$\mathsf{TrapGen}(sk_A, w)$:

1. Compute $id_1 = H_1(1||w)$, and then choose $dk_{i,1} \xleftarrow{R} \mathbb{Z}_p$, compute $dk_{i,2} = (h_i g^{-dk_{i,1}})^{1/(\alpha - id_1)}$ and set $dk_i = (dk_{i,1}, dk_{i,2})$ for $1 \leq i \leq 3$. If $\alpha = id_1$, the algorithm aborts. Otherwise, return $t_w = (dk_1, dk_2, dk_3)$.

$\mathsf{Test}(t_w, u)$:

1. Parse t_w as (dk_1, dk_2, dk_3), u as (vk, c, s, σ).

2. Compute $z = H_1(c||s)$. If $e(\sigma_2, u \cdot t^z \cdot v^{\sigma_1}) \neq e(t,t)$, return 0. Otherwise, compute $\beta = H_1(s_1, s_2, s_3)$, test if $s_4 = e(s_1, dk_{2,2} dk_{3,2}^{\beta}) s_2^{dk_{2,1} + dk_{3,1}\beta}$. If the check fails, return 0. Else, continue to check if $H_2(vk) = s_3 \cdot e(s_1, dk_{1,2}) s_2^{dk_{1,1}}$. If so, return 1. Else return 0.

The correctness of the above scheme can be easily verified. Since the Gentry IBE [Gen06] is shown to be ANO-IBE-CCA based on the truncated q-ABDHE assumption, the signature scheme [BB04b] is sEUF-CMA secure based on the Strong Diffie-Hellman (SDH) assumption. Besides, Gentry IBE is weak robust against CCA-attack and thus it is also weak collision-free. Therefore, according to Theorem 6.1 this above construction is jointly CCA-secure based on these assumptions in the standard model.

D Review of Standard Definitions

D.1 Public-Key Encryption

A public-key encryption scheme consists of three PPT algorithms as follows:

– $\mathsf{KeyGen}(\kappa)$: take as input a security parameter κ, output a public/secret key pair (pk, sk).
– $\mathsf{Encrypt}(pk, m)$: take as input a public key pk and a message m, output a ciphertext c.
– $\mathsf{Decrypt}(sk, c)$: take as input a secret key sk and a ciphertext c, output a message m.

The standard security notion for PKE schemes is indistinguishability against adaptive chosen-ciphertext attack [RS91](IND-PKE-CCA), which is defined by the following experiment:

Setup: $\mathcal{CH}$ runs $\mathsf{KeyGen}(\kappa)$ and gives $\mathcal{A}$ the public key pk.
Phase 1: $\mathcal{A}$ adaptively makes decryption queries $\langle c \rangle$, $\mathcal{CH}$ responds with $\mathsf{Decrypt}(sk, c)$.
Challenge: $\mathcal{A}$ submits two messages m_0 and m_1. $\mathcal{CH}$ randomly picks a bit $b \in \{0,1\}$ and sends $c^* = \mathsf{Encrypt}(pk, m_b)$ to $\mathcal{A}$ as the challenge ciphertext.
Phase 2: $\mathcal{A}$ continues to adaptively make decryption queries $\langle c \rangle$ subjected to the restriction that $c \neq c^*$. $\mathcal{CH}$ responds the same way as in Phase 1.
Guess: $\mathcal{A}$ outputs a guess $b' \in \{0,1\}$ of b.

We define $\mathcal{A}$'s advantage as $\mathrm{Adv}_{\mathcal{A},\mathrm{PKE}}^{\mathrm{IND\text{-}CCA}}(\kappa) = |\Pr[b' = b] - 1/2|$.

Definition D.1 *A PKE scheme is (t, Q_d, ϵ) IND-PKE-CCA secure if for all t-time PPT adversary making at most Q_d decryption queries have advantage at most ϵ in the above experiment.*

The IND-PKE-CPA security for PEKS schemes can be defined by a similar experiment without decryption queries.

D.2 Public-Key Encryption with Keyword Search

A non-interactive public-key encryption with keyword search scheme [BCOP04] consists of three PPT algorithms as follows:

– $\mathsf{KeyGen}(\kappa)$: take as input a security parameter κ, output a public/secret key pair (pk_A, sk_A). Let W be the set of all possible keywords.
– $\mathsf{PEKS}(pk_A, w)$: take as input a public key pk_A and a keyword $w \in W$, output a ciphertext s.
– $\mathsf{TokenGen}(sk_A, w)$: take as input a secret key sk_A and a keyword $w \in W$, output a token t_w.
– $\mathsf{Test}(t_w, s)$: take as input a token t_w and a ciphertext $s = \mathsf{PEKS}(pk, w')$, output 1 if $w' = w$ and 0 otherwise.

The IND-PEKS-CPA security for PEKS schemes is defined by the following experiment:

Setup: $\mathcal{CH}$ runs $\mathsf{KeyGen}(\kappa)$ and gives $\mathcal{A}$ the public key pk_A.

Phase 1: $\mathcal{A}$ adaptively makes token queries $\langle w \rangle$. $\mathcal{CH}$ responds with $t_w \leftarrow \mathsf{TokenGen}(sk_A, w)$.
Challenge: $\mathcal{A}$ outputs two distinct keywords $w_0^*, w_1^* \in W$ subject to the restriction that they had not been asked for tokens in Phase 1. $\mathcal{CH}$ picks a random bit $b \in \{0,1\}$ and sends $c^* = \mathsf{PEKS}(pk_A, w_b^*)$ to $\mathcal{A}$ as the challenge ciphertext.
Phase 2: $\mathcal{A}$ continues to adaptively make token queries $\langle w \rangle$ subject to the restriction that $w \neq w_0^*, w_1^*$. $\mathcal{CH}$ responds the same way as in Phase 1.
Guess: $\mathcal{A}$ outputs a guess $b' \in \{0,1\}$ of b.

We define $\mathcal{A}$'s advantage as $\mathrm{Adv}_{\mathcal{A},\mathrm{PEKS}}^{\mathrm{IND\text{-}CPA}}(\kappa) = |\Pr[b' = b] - 1/2|$.

Definition D.2 *A PEKS scheme is (t, Q_w, ϵ) IND-PEKS-CPA secure if for all t-time PPT adversary making at most Q_w token queries have advantage at most ϵ in the above experiment.*

The IND-PEKS-CCA security for PEKS schemes can be defined by a similar experiment by giving the adversary access to an additional test oracle which can determine if c is an encryption of w. To avoid triviality, test queries $\langle c^*, w_0^* \rangle$ and $\langle c^*, w_1^* \rangle$ are not allowed in Phase 2.

Definition D.3 *A PEKS scheme is (t, Q_w, Q_t, ϵ) IND-PEKS-CCA secure if for all t-time PPT adversary making at most Q_w token queries and Q_t test queries have advantage at most ϵ in the IND-PEKS-CCA experiment.*

D.3 Hierarchical Identity-Based Encryption

Hierarchical identity-based encryption (HIBE) [HL02] is a generalization of IBE [BF03] to identities supporting hierarchical structures. In a HIBE scheme, identities are hierarchical and take the form $id = [id_1, id_2, \ldots]$. Each user in the hierarchy can act as a local key-generation authority for all subordinate hierarchical identities. A HIBE scheme consists of five PPT algorithms as follows:

- $\mathsf{Setup}(\kappa, \ell)$: take as input a security parameter κ and a parameter ℓ for the maximum depth of the HIBE, output a master public/secret key pair (mpk, msk). Let I be the identity space, M be the message space, and C be the ciphertext space. We assume mpk is used as an implicit input for algorithms $\mathsf{Extract}$, Derive, as well as $\mathsf{Decrypt}$,
- $\mathsf{Extract}(msk, id)$: take as input msk and an identity $id \in I$, output a private key dk_{id}.
- $\mathsf{Derive}(dk, id')$: take as input a private key dk_{id} for identity $id = (id_1, \ldots, id_{j-1})$ of depth $j-1$ and an identity $id' = (id_1, \ldots, id_j)$ of depth j, output a private key for id'.
- $\mathsf{Encrypt}(mpk, id, m)$: take as input mpk, an identity $id \in I$, and a message $m \in M$, output a ciphertext $c \in C$.
- $\mathsf{Decrypt}(dk, c)$: take as input a private key dk and a ciphertext $c \in C$, output a message $m \in M$ or a reject symbol $\perp$ indicating c is invalid.

The basic security notion for HIBE schemes is indistinguishability against adaptive chosen-plaintext attack (IND-HIBE-CPA), which is defined by the following experiment:

Setup: $\mathcal{CH}$ runs $\mathsf{Setup}(\kappa)$ and gives $\mathcal{A}$ the master public key mpk.
Phase 1: $\mathcal{A}$ adaptively makes private key extraction queries $\langle id \rangle$. $\mathcal{CH}$ responds with $dk_{id} \leftarrow \mathsf{Extract}(msk, id)$.
Challenge: $\mathcal{A}$ outputs two distinct messages m_0, m_1 and an identity id^* subject to the restriction that any prefix of id^* had not been queried for private keys in Phase 1. $\mathcal{CH}$ randomly picks a bit $b \in \{0,1\}$ and sends $c^* = \mathsf{Encrypt}(id^*, m_b)$ to $\mathcal{A}$ as the challenge ciphertext .
Phase 2: $\mathcal{A}$ continues to adaptively make more private key extraction queries $\langle id \rangle$ subject to the restriction that id is not a a prefix of id^*. $\mathcal{CH}$ responds the same way as in Phase 1.
Guess: $\mathcal{A}$ outputs a guess $b \in \{0,1\}$ of b.

We define $\mathcal{A}$'s advantage as $\mathrm{Adv}^{\text{IND-CPA}}_{\mathcal{A},\text{HIBE}}(\kappa) = |\Pr[b = b'] - 1/2|$.

Definition D.4 *A HIBE scheme is (t, Q_e, ϵ) IND-HIBE-CPA secure if for all t-time PPT adversary making at most Q_e private key extraction queries have advantage at most ϵ in the above experiment.*

The selective-identity IND-HIBE-CPA security can be defined similarly, the difference is that the adversary has to commit a target identity id^* before seeing mpk and is not allowed to query private keys for any prefix of id^* throughout the experiment.

Orthogonal to security, anonymity is introduced to capture key privacy, which ensures the ciphertext reveals no information about the recipient's identity. To allow a fine-grained treatment, Abdalla *et al.* [ABC$^+$08] defined anonymity regarding to a set L of levels, meaning that in an experiment an adversary is challenged to distinguished two distinct identities differing only at levels $l \in L$. For ease of notation, we will write l rather than $\{l\}$ when $L = \{l\}$ is a singleton set. Let $\mathrm{diff}(\cdot, \cdot)$ be a function returning the set of levels at which the two input identities differ. We recall the definition of anonymity for HIBE schemes as follows:

Setup: $\mathcal{CH}$ runs $\mathsf{Setup}(\kappa)$ and gives $\mathcal{A}$ the master public key mpk.
Phase 1: $\mathcal{A}$ adaptively makes private key extraction queries $\langle id \rangle$. $\mathcal{CH}$ responds with $dk_{id} \leftarrow \mathsf{Extract}(msk, id)$.
Challenge: $\mathcal{A}$ outputs a message m and two distinct identities id_0^*, id_1^* subject to the restrictions that any prefix of id_0^* and id_1^* had not been asked for private keys and $\mathrm{diff}(id_0^*, id_1^*) \subset L$. $\mathcal{CH}$ picks a random $b \in \{0, 1\}$ and gives $c^* = \mathsf{Encrypt}(mpk, id_b^*, m)$ to $\mathcal{A}$ as the challenge ciphertext.
Phase 2: $\mathcal{A}$ continues to adaptively issue the private key extraction queries for any identity id subject to the restriction that id is not a prefix of id_0^* or id_1^*. $\mathcal{CH}$ responds the same way as in Phase 1.
Guess: $\mathcal{A}$ outputs a guess $b' \in \{0, 1\}$ of b.

We define $\mathcal{A}$'s advantage as $\mathrm{Adv}^{\text{ANO-CPA}}_{\mathcal{A},\text{HIBE}}(\kappa) = |\Pr[b = b'] - 1/2|$.

Definition D.5 *A HIBE scheme is (t, Q_e, ϵ) ANO-HIBE-CPA $[L]$-anonymous if for all t-time PPT adversary making at most Q_e private key extraction queries have advantage at most ϵ in the above experiment.*

When considering anonymity for HIBE schemes in the presence of chosen-ciphertext attack, we obtain the ANO-PEKS-CCA security [ABN10], which is defined by a similar experiment as above by giving the adversary additional access to a decryption oracle subject the natural restriction that decryption queries $\langle id_0^*, c^* \rangle$ and $\langle id_1^*, c^* \rangle$ are not allowed in Phase 2.

Definition D.6 *A HIBE scheme is (t, Q_e, Q_d, ϵ) ANO-HIBE-CCA $[L]$-anonymous if for all t-time PPT adversary making at most Q_e private key extraction queries and at most Q_d decryption queries have advantage at most ϵ in the ANO-HIBE-CCA experiment.*

D.4 Signatures

A signature scheme consists of three PPT algorithms as follows:

- $\mathsf{KeyGen}(\kappa)$: take as input a security parameter κ, output a verification key vk and a signing key sk. We assume for simplicity that vk has length $n = n(\kappa)$.
- $\mathsf{Sign}(sk, m)$: take as input a signing key sk and a message m (in some implicit message space), output a signature σ.
- $\mathsf{Verify}(vk, m, \sigma)$: take as input a verification key vk, a message m, and a signature σ, output 1 indicates "acceptance" and 0 indicates "rejection".

For the correctness of a signature scheme, we require that for all $(vk, sk) \leftarrow \mathsf{KeyGen}(\kappa)$, and all m in the message space, $\mathsf{Verify}(vk, m, \mathsf{Sign}(sk, m)) = 1$ holds. If (σ, m) satisfies $\mathsf{Verify}(vk, m, \sigma) = 1$, then σ is said to be a valid signature for the message m under the verification key pk.

A strong notion of security for signature schemes is strong existential unforgeability under adaptive chosen-message attack (sEUF-CMA), which is defined by the following experiment:

Setup: $\mathcal{CH}$ runs $\mathsf{KeyGen}(\kappa)$. and gives $\mathcal{A}$ the verification key vk.
Phase 1: $\mathcal{A}$ adaptively makes signing queries $\langle m \rangle$. $\mathcal{CH}$ responds with $\sigma \leftarrow \mathsf{Sign}(sk, m)$.
Forgery: $\mathcal{A}$ outputs a message-signature pair (m, σ) and wins if it is not one of the message-signature pairs produced before and $\mathsf{Verify}(vk, m, \sigma) = 1$.

We define $\mathcal{A}$'s advantage as $\mathrm{Adv}_{\mathcal{A},\mathrm{SIG}}^{\mathrm{sEUF\text{-}CMA}} = \Pr[\mathcal{A} \text{ wins}]$.

Definition D.7 *A signature scheme is (t, Q_s, ϵ) sEUF-CMA secure if for all t-time PPT adversary making at most Q_s signing queries have advantage at most ϵ in the above experiment. Particularly, a signature scheme is one-time strongly unforgeable if it is $(t, 1, \epsilon)$ sEUF-CMA secure.*

D.5 BCHK Transform

Boneh *et al.* [BCHK07] proposed a generic method known as the BCHK transform, which can convert a CPA-secure IBE scheme into a CCA-secure PKE scheme by combining the use of a one-time strongly unforgeable signature scheme. We recall this transform as below:

- $\mathsf{KeyGen}(\kappa)$: run $(mpk, msk) \leftarrow \mathrm{IBE}.\mathsf{Setup}(\kappa)$, return $(pk_A, sk_A) := (mpk, msk)$.
- $\mathsf{Encrypt}(pk_A, m)$: run $\mathrm{OT}.\mathsf{KeyGen}(\kappa)$ to generate a verificaton-signing keypair (vk, sk), then compute $e \leftarrow \mathrm{IBE}.\mathsf{Encrypt}(pk_A, vk, m)$, create a signature $\sigma \leftarrow \mathrm{OT}.\mathsf{Sign}(sk, e)$, output the final ciphertext $c = (vk, e, \sigma)$.
- $\mathsf{Decrypt}(pk_A, c)$: parse c as (vk, e, σ), check if $\mathrm{OT}.\mathsf{Verify}(vk, e, \sigma) = 1$. If so, compute $dk \leftarrow \mathsf{Extract}(sk_A, vk)$ and output $m \leftarrow \mathrm{IBE}.\mathsf{Decrypt}(dk, e)$. If not, reject the ciphertext with $\perp$.

D.6 BDOP Transform

Boneh *et al.* [BCOP04] implicitly presented a transform known as the BDOP transform, which can convert an anonymous IBE scheme to a secure PEKS scheme. We recall this transform as below:

- $\mathsf{KeyGen}(\kappa)$: run $(mpk, msk) \leftarrow \mathrm{IBE}.\mathsf{Setup}(\kappa)$, set $(pk_A, sk_A) := (mpk, msk)$.
- $\mathsf{PEKS}(pk_A, w)$: let s_1 be a pre-fixed value from the message space of the underlying IBE scheme, compute $s_2 \leftarrow \mathrm{IBE}.\mathsf{Encrypt}(pk_A, w, s_1)$, output the ciphertext $s = (s_1, s_2)$.
- $\mathsf{TokenGen}(sk_A, w)$: output $t_w \leftarrow \mathrm{IBE}.\mathsf{Extract}(pk_A, sk_A, w)$.
- $\mathsf{Test}(t_w, s)$: output 1 if $s_1 = \mathrm{IBE}.\mathsf{Decrypt}(t_w, s_2)$ and 0 otherwise.

Remark 3. Abdalla *et al.* [ABC$^+$08] revised the transform and gave a formal proof for it. We refer to the revised BDOP transform as the new BDOP transform, which is identical to the original one except that its PEKS algorithm encrypts a random message but not a pre-fixed one.