Secure searchable public key encryption scheme against keyword guessing attacks

Hyun Sook Rhee[1a], **Willy Susilo**[1b], **and Hyun-Jeong Kim**[2c]

[1] *University of Wollongong, Australia*

[2] *Korea Environmental Council in Europe (KECE), Belgium*

a) *hyunsook.rhee@gmail.com*

b) *wsusilo@uow.edu.au*

c) *security@kece.eu*

Abstract: Byun *et al.* firstly proposed off-line keyword guessing (KG) attacks and proved that some searchable public key encryption (PEKS) schemes are insecure against these attacks. They supposed an open problem on how to construct PEKS schemes secure against keyword guessing attacks. In this letter, we answer this question affirmatively.

Keywords: Public key encryption with keyword search, Keyword guessing attacks

Classification: Storage technology

References

[1] M. Abdalla, M. Bellare, D. Catalano, E. Kiltz, T. Kohno, T. Lange, J. Malone-Lee, G. Neven, P. Paillier, and H. Shi, "Searchable encryption revisited: consistency properties, relation to anonymous IBE, and extensions," *Proc. Crypto'05*, LNCS 3621, 2005.

[2] J. Baek, R. Safavi-Naini, and W. Susilo, "Public key encryption with keyword search revisited," *Proc. ACIS'06*, 2006.

[3] D. Boneh, G. D. Crescenzo, R. Ostrovsky, and G. Persiano, "Public key encryption with keyword search," *Proc. Eurocrypt'04*, LNCS 3027, 2004.

[4] J. W. Byun, H. S. Rhee, H. A. Park, and D. H. Lee, "Off-Line Keyword Guessing Attacks on Recent Keyword Search Schemes over Encrypted Data," *Proc. SDM'06*, LNCS 4165, 2006.

[5] I. R. Jeong, J. O. Kwon, and D. H. Lee, "Constructiong PEKS schemes secure against keyword guessing attacks is possible?," *Elsevier's Computer Communications*, vol. 32, 2009.

[6] H. S. Rhee, J. H. Park, W. Susilo, and D. H. Lee, "Improved searchable public key encryption with designated tester," *Proc. ASIACCS'09* (to appear), 2009.

© IEICE 2009
DOI: 10.1587/elex.6.237
Received December 19, 2008
Accepted January 13, 2009
Published March 10, 2009

1 Introduction

A searchable public encryption scheme (or PEKS for short) used on email routing system was first introduced by Boneh *et al.* [3] to provide emails privacy. In a PEKS scheme, an email sender, Bob, generates an encrypted email (comprises of an encrypted email body and an encrypted list of keywords) by using the public key of an email receiver, Alice, and sends the encrypted email to Alice through an email server. Bob can use a standard public-key encryption scheme (PKE) to encrypt the body of the email, and a PEKS scheme to encrypt each keyword in the list of keywords. To search the encrypted emails on the email server, Alice provides the server with an encrypted keyword known as a *trapdoor* (generated by her secret key). Then the server tests which keywords appended to encrypted emails are identical to the keyword of the trapdoor without revealing any information about the encrypted list of keywords.

Boneh *et al.* considered the security for a PEKS ciphertext (the list of keywords encrypted by PEKS) against an active attacker who is able to obtain trapdoors for any keyword chosen by himself in the sense of semantic-security [3]. This security ensures that a PEKS ciphertext reveals no information without the trapdoor related to the given PEKS ciphertext. However, even though the security of a PEKS ciphertext is guaranteed, anyone who has obtained several trapdoors about unknown keywords can store and use the trapdoors to classify all captured encrypted emails. Baek *et al.* [2] proposed a searchable public key encryption for designated tester (dPEKS) to solve this problem. In a dPEKS scheme, only the designated server can test which dPEKS ciphertext is related with a given trapdoor by using his private key.

Byun *et al.* firstly defined off-line keyword guessing (KG) attacks and showed that the PEKS scheme [3] is insecure against KG attacks. Since email users (a sender and a receiver) usually query commonly-used keywords which have low entropy, KG attacks are meaningful. If an attacker $\mathcal{A}$ who can guess the keyword w of the given trapdoor T_w obtains the replied message, PEKS/dPEKS ciphertexts, by an email server, $\mathcal{A}$ can know that not only the relation between the PEKS/dPEKS ciphertexts and the trapdoor T_w but also the keyword about PEKS/dPEKS ciphertexts by using KG attacks. As a result, if the security for a PEKS/dPEKS ciphertext is not provided against KG attacks, the security for PEKS/dPEKS ciphertext cannot also be guaranteed in the sense of semantic-security.

Jeong *et al.* [5] pointed that the consistency implies insecurity of a PEKS scheme against KG attacks: Abdalla *et al.* [1] defined the notion of consistency in a PEKS scheme, which means that if the keyword w used in generating a PEKS/dEPKS ciphertext by a sender is not identical to the keyword w' ($w \neq w'$) used in generating a trapdoor by a receiver, the probability of that a server determines $w = w'$ is negligible[1]. However, although a PEKS (dPEKS) scheme satisfies the consistency, it is possible to construct a secure PEKS scheme against KG attacks. If a keyword guessing attacker $\mathcal{A}$ cannot

[1]If h is negligible, for any constant k, there exists N such that $h(n) < 1/n^k$ for $n > N$.

© IEICE 2009
DOI: 10.1587/elex.6.237
Received December 19, 2008
Accepted January 13, 2009
Published March 10, 2009

guess the keyword w of a given trapdoor T_w in a PEKS (dPEKS) scheme, the scheme is secure against KG attacks.

In this letter, we construct a new secure dPEKS scheme against KG attacks. We show that our dPEKS scheme also satisfies the consistency and the security of dPEKS ciphertext in the sense of semantic-security. Our scheme is one solution for the open problem proposed by Byun *et al.* [4].

2 Preliminaries

Let G and G_T be groups, where the computational Diffie-Hellman (CDH) problem is hard. Let λ be a security parameter and Z be a set of integers. Suppose that g is a generator of G and $e(g, g)$ is a generator of G_T. Let $H_1 : \{0,1\}^* \rightarrow G$ and $H_2 : G_T \rightarrow \{0,1\}^\lambda$ be hash functions. A bilinear map $e : G \times G \rightarrow G_T$ satisfies the following properties.

- Bilinear : for all $u, v \in G$ and $a, b \in Z$, we have $e(u^a, v^b) = e(u, v)^{ab}$.
- Non-degenerate : $e(g, g) \neq 1$.

We say that G is a bilinear group if the group action in G can be computed efficiently, and there exist a group G_T and an efficiently computable bilinear map $e : G \times G \rightarrow G_T$.

2.1 Review of PEKS/dPEKS Scheme

Let $\mathcal{KS}$ be a keyword space where the size is bounded by some polynomial.

PEKS Scheme. Boneh *et al.*'s PEKS scheme [3] works as follows:
• $\mathsf{KeyGen}(\lambda)$: Given a security parameter λ, it picks a random $\alpha \in Z_p^*$ and outputs $pk = (pk_1, pk_2) = (g, g^x)$ and $sk = x$.
• $\mathsf{PEKS}(pk, w)$: This algorithm picks a random $r \in Z_p^*$ and outputs a ciphertext $C = [A,\ B] = [g^r, H_2(e(H_1(w), pk_2^r))]$, where $w \in \mathcal{KS}$.
• $\mathsf{Trapdoor}(sk, w)$: This algorithm outputs a trapdoor $T_w = H_1(w)^{sk}$.
• $\mathsf{Test}(C, T_w)$: This algorithm checks if $B = H_2(e(A, T_w))$. If this equality is satisfied, then output "1". Otherwise, output "0".

dPEKS Scheme. Baek *et al.*'s dPKES scheme [2] works as follows :
• $\mathsf{Global\ Setup}(\lambda)$: Given a security parameter λ, it returns a global parameter $\mathcal{GP} = (G,\ G_T,\ e,\ H_1(\cdot),\ H_2(\cdot),\ g,\ \mathcal{KS})$.
• $\mathsf{KeyGen}_{Server}(\mathcal{GP})$: This algorithm randomly chooses $\alpha \in Z_p^*$ and $Q \in G$ and returns $sk_s = (\mathcal{GP}, \alpha)$ and $pk_s = (\mathcal{GP}, Q, Z) = (\mathcal{GP}, Q, g^\alpha)$ as the server's secret and public key respectively.
• $\mathsf{KeyGen}_{Receiver}(\mathcal{GP})$: This algorithm randomly chooses $x \in Z_p^*$ and returns $sk_{_R} = x$ and $pk_{_R} = g^x$ as the receiver's secret and public key respectively.
• $\mathsf{dPEKS}(pk_{_R}, pk_s, w)$: This algorithm picks a random $r \in Z_p^*$ and outputs a ciphertext $C = [g^r, H_2(k)]$, where $k = e(Q, Z)^r \cdot e(H_1(w), pk_{_R})^r$ and $w \in \mathcal{KS}$.
• $\mathsf{Trapdoor}(sk_{_R}, w)$: This algorithm outputs $T_w = H_1(w)^x$, where $w \in \mathcal{KS}$.
• $\mathsf{dTest}(C, sk_s, T_w)$: This algorithm checks if $B = H_2(e(Q^\alpha + T_w, A))$. If this equality is satisfied, then output "1". Otherwise, output "0".

© IEICE 2009
DOI: 10.1587/elex.6.237
Received December 19, 2008
Accepted January 13, 2009
Published March 10, 2009

2.2 Consistency in dPEKS

We define the notion of consistency in a dPEKS scheme, which is similar to the notion of consistency in a PEKS scheme from [1]. Suppose there exists an adversary $\mathcal{A}$ that wants to make consistency fail. The consistency is formally defined as follows:

```
Experiment $\mathbf{Exp}^{\mathsf{dpeks-cons}}_{\mathbf{dPEKS},\mathcal{A}}(k)$
    $(pk_R, sk_R) \leftarrow \mathsf{KeyGen}_{\mathsf{Receiver}}(1^k)$ ; $(pk_S, sk_S) \leftarrow \mathsf{KeyGen}_{\mathsf{Server}}(1^k)$
    Pick random oracles $H_1$ and $H_2$
    $(w, w') \leftarrow \mathcal{A}(pk_R, pk_S)$
    $C \leftarrow \mathsf{dPEKS}^{H_1,H_2}(pk_R, pk_S, w)$ ; $T_{w'} \leftarrow \mathsf{Trapdoor}^{H_1}(sk_R, w')$
    if $w \neq w'$ and $\mathsf{dTest}(T'_w, sk_S, C) = 1$, then return 1 else return 0.
```

We define the advantage of $\mathcal{A}$ as

$$\mathbf{Adv}^{\mathsf{dpeks-cons}}_{\mathbf{dPEKS},\mathcal{A}}(k) = \Pr[\mathbf{Exp}^{\mathsf{dpeks\text{-}cons}}_{\mathbf{dPEKS},\mathcal{A}}(k) = 1] \ ,$$

where the probability is taken over all possible coin flips of all the algorithms involved, and over all possible choices of random oracles H_1 and H_2. The scheme is said to be *computationally consistent* if it is negligible for all polynomial time adversaries $\mathcal{A}$ to win the above experiment.

3 PEKS Schemes versus KG attacks

We review KG attacks [4] on Boneh *et al.*'s PEKS scheme, show that Baek *et al.*'s dPEKS scheme is insecure against KG attacks, and explain the reasons why PEKS and dPEKS schemes are insecure against KG attacks.

3.1 Insecure PEKS/dPEKS Scheme against KG attacks

Let $w \in \mathcal{KS}$ be an unknown keyword and T_w be the trapdoor of w. Let $\mathcal{A}$ be an outside attacker where the running time is bounded by t which is polynomial in a security parameter k. We assume that $\mathcal{A}$ can determine which keyword was used in generating a given trapdoor.

PEKS Scheme [4]. (1) $\mathcal{A}$ guesses an appropriate keyword $w' \in \mathcal{KS}$ and computes $H_1(w')$. (2) $\mathcal{A}$ checks if the equality $e(pk_R, H_1(w')) = e(g, T_w)$ is satisfied. If so, the guessed keyword is valid. Otherwise, go to (1). Suppose that $T_w = H_1(w)^x$ and $w = w'$. The equation $e(pk_R, H_1(w')) = e(g, H_1(w')^x)$ $= e(g, T_w)$ is satisfied.

dPEKS Scheme. We show that Baek *et al.*'s dPEKS scheme is not secure against KG attacks. Let $pk_s = (\mathcal{GP}, Q, Z) = (\mathcal{GP}, Q, g^\alpha)$ and $pk_R = (pk_s, Y) = (pk_s, g^x)$ be the server's and receiver's public keys. Suppose that an attacker $\mathcal{A}$ is given a ciphertext C and a trapdoor T_w such that $\mathsf{dTest}(C, T_w) = 1$, and C and T_w are made with a keyword w in $\mathcal{KS}$. $\mathcal{A}$ can determine which keyword is used in generating C and T_w as follows: (1) $\mathcal{A}$ guesses a keyword $w' \in \mathcal{KS}$ and computes $H_1(w')$. (2) $\mathcal{A}$ checks if $e(H_1(w'), Y) = e(T_w, g)$. If so, the guessed keyword w' is a valid keyword. Otherwise, go to (1).

© IEICE 2009
DOI: 10.1587/elex.6.237
Received December 19, 2008
Accepted January 13, 2009
Published March 10, 2009

3.2 The Analysis of the PEKS/dPEKS Scheme against KG attacks

Boneh *et al.*'s PEKS scheme [3] satisfies the following functionalities:
(1) *Generating a PEKS ciphertext should be easy*: It should be possible for everyone to generate a PEKS ciphertext $C = \mathsf{PEKS}(w, pk_A)$ where w is a keyword and pk_A is a receiver's public key.
(2) *Capability of searching should be restricted*: It should be possible for only the receiver to generate a trapdoor T_w with a keyword w and his/her secret key sk_A.
(3) *Consistency should be satisfied*: To provide a correct routing configuration of the email server, it should be satisfied that if $w \neq w'$ then $\Pr[\mathsf{Test}(C, T_{w'}) = 1]$ is negligible, where $T_{w'} = \mathsf{Trapdoor}(w', sk_A)$, $C = \mathsf{PEKS}(w, pk_A)$, and w and w' are unknown keywords.

When a trapdoor T_w about an unknown keyword w is given in Boneh *et al.*'s PEKS scheme, an attacker $\mathcal{A}$ can try to guess a keyword w' of a keyword space: we can generally assume that the size of the keyword space is polynomial and a keyword has low entropy. Then, on the base of the above functionalities, $\mathcal{A}$ can generate the PEKS ciphertext $C' = \mathsf{PEKS}(w', pk_A)$ and check if $\mathsf{Test}(C', T_w) = 1$ using the PEKS and Test algorithms. If $\mathcal{A}$ gets the keyword w' satisfying the equality of $\mathsf{Test}(C', T_w) = 1$ then $\mathcal{A}$ can win in the KG attacks on Boneh *et al.*'s PEKS scheme. In particular, Byun *et al.*'s KG attacks on Boneh *et al.*'s PEKS scheme considers the special case of the above process, in which the first component g^r of a ciphertext $C = [g^r, H_2(e(pk_R, H_1(w)^r))]$ is fixed as g^1, *i.e.*, $r = 1$.

In Baek *et al.*'s dPEKS scheme, when the trapdoor T_w about an unknown keyword w is given, only an email server chosen by an email sender can process dTest algorithm. However, an attacker $\mathcal{A}$ wins in the KG attacks on the dPEKS scheme by using the same process. The reason lies on that the structure of a trapdoor in the dPEKS scheme is same to one in the PEKS scheme. When the trapdoor T_w about an unknown keyword w is given, $\mathcal{A}$ guesses a keyword w' and generates the PEKS ciphertext C' instead of the dPEKS ciphertext. Then, $\mathcal{A}$ can check the equality of $\mathsf{Test}(C', T_w) = 1$ instead of dTest algorithm. To protect the dPEKS scheme from KG attacks, the structure of a trapdoor should be changed. To this end, we suggest a dPEKS scheme with a new structure of a trapdoor.

4 Secure dPEKS Scheme against KG Attacks

Our secure dPEKS scheme against KG attacks works as follows:
• $\mathsf{Global\ Setup}(\lambda)$: Given a security parameter λ, it returns a global parameter $\mathcal{GP} = (G, G_T, e, H_1(\cdot), H_2(\cdot), g, \mathcal{KS})$, where $\mathcal{KS}$ is a keyword space.
• $\mathsf{KeyGen}_{Server}(\mathcal{GP})$: This algorithm randomly chooses $\alpha \in Z_p^*$ and $Q \in G$, and returns $sk_s = \alpha$ and $pk_s = (\mathcal{GP}, Q, y_s) = (\mathcal{GP}, Q, g^\alpha)$ as a server's secret and public keys, respectively.

© IEICE 2009
DOI: 10.1587/elex.6.237
Received December 19, 2008
Accepted January 13, 2009
Published March 10, 2009

• $\mathsf{KeyGen}_{Receiver}(\mathcal{GP})$: This algorithm randomly chooses $x \in Z_p^*$ and returns $sk_R = x$ and $pk_R = g^x$ as a receiver's secret and public keys, respectively.
• $\mathsf{dPEKS}(pk_R, pk_S, w)$: This algorithm picks a random value $r \in Z_p^*$ and outputs $C = [A, B] = [(pk_R)^r, H_2(e(y_s, H_1(w)^r))]$, where $w \in \mathcal{KS}$.
• $\mathsf{Trapdoor}(sk_R, w)$: This algorithm outputs $T_w = [T_1, T_2] = [y_s^{r'},\ H_1(w)^{1/x} \cdot g^{r'}]$, where $w \in \mathcal{KS}$.
• $\mathsf{dTest}(C, T_w)$: This algorithm computes $\mathcal{T} = (T_2)^\alpha/(T_1)^{\alpha^2}$ and checks if $B = H_2(e(A, \mathcal{T}))$. If the equality is satisfied, then output "1"; otherwise, output "0".

Consistency and Security. Since the security of our dPEKS scheme is similar to one in [6], we can identically show the security of a dPEKS ciphertext in same manner in [6]. We omit the proof of the security of a dPEKS ciphertext. We show that our scheme is computationally consistent and is secure against KG attacks on the base that a discrete logarithm problem is hard.

Theorem 1. *Our dPEKS scheme satisfies computationally consistency.*
Proof. Suppose $\mathcal{A}$ is any polynomial time adversary attacking the computational consistency of our dPEKS scheme. Let w and w' be keywords that $\mathcal{A}$ returns in the consistency experiment, and assume without loss of generality that $w \neq w'$. Let $r \in Z_p^*$ be a random value, $A = e(H_1(w), y_s)^r$, and $A' = e(H_1(w'), y_s)^r$. If $w \neq w'$ and $H_2(A) = H_2(A')$, then $\mathcal{A}$ wins correctly.

Since H_1 and H_2 are random oracles and the probability of hash collisions is relatively small, it is not easy that $\mathcal{A}$ obtains a non-negligible advantage about the dPEKS-consistent. Suppose that q_i $(i = 1, 2)$ is the number of the queries of $\mathcal{A}$ to H_i and E_1 is the event that there exist distinct queries $w, w' \in \{0,1\}^*$ such that $H_1(w) = H_1(w')$ and E_2 is the event that there exist $x, x' \in G_T$ $(x \neq x')$ such that $H_2(x) = H_2(x')$. If the hypothesis $|G| > 2^k$ $(2^k < p < 2^{k+1})$ of the group G is used, $\Pr[E_1] < (q_1+2)^2/|G| < (q_1+2)^2/2^k$, $\Pr[E_2] < (q_1+2)^2/2^k$ and $\mathbf{Adv}^{\mathsf{dpeks-cons}}_{\mathbf{dPEKS},\mathcal{A}}(k) \leq \Pr[E_1] + \Pr[E_2] + \Pr[\mathbf{Exp}^{\mathsf{dpeks-cons}}_{\mathbf{dPEKS},\mathcal{A}}(k) = 1 \wedge \overline{E}_1 \wedge \overline{E}_2]$. If $\Pr[\mathbf{Exp}^{\mathsf{dpeks-cons}}_{\mathbf{dPEKS},\mathcal{A}}(k) = 1 \wedge \overline{E_1} \wedge \overline{E_2}]$ is negligible, then the proof is completed. If $\overline{E}_1 \wedge \overline{E}_2$ and $H_1(w) \neq H_1(w')$, then there exists $a, a' \in Z_p$ such that $H_1(w) = g^a$ and $H_1(w') = g^{a'}$. Also, $e(g,g)$ is a generator of G_T of prime order p, $e(g,g)^r$ is a generator of G_T and p is not divide with r, for every $r \in Z_p$. Hence, $(e(g,g)^r)^{a\alpha} \neq (e(g,g)^r)^{a'\alpha}$ where $\alpha \in Z_p^*$ is the server's unknown private key.

Theorem 2. *Our dPEKS scheme is secure against KG attacks.*
Proof. Suppose $\mathcal{B}$ is a KG attacker in our scheme with advantage ϵ. Assume that $T_w = [T_1, T_2]$ is a trapdoor. To obtain a correct keyword $w \in \mathcal{KS}$ from the given T_w, it should be possible that $\mathcal{B}$ gets $H_1(w)^{1/x}$ or $H_1(w)$ from T_w. Since a discrete logarithm problem is hard, $\mathcal{B}$ cannot easily get the unknown r' or $\alpha \in Z_p^*$ from $T_2 = y_s^{r'}$, where $y_s = g^\alpha$. Furthermore, even though $\mathcal{B}$ can compute $e(y_s, T_2)/e(g, T_1) = e(y_s, H_1(w)^{1/x} \cdot g^{r'})/e(g, y_s^{r'}) = e(y_s, H_1(w)^{1/x})$, $\mathcal{B}$ cannot guess $w \in \mathcal{KS}$ such that $e(y_s, H_1(w)^{1/x})$ without a knowledge of a receiver's secret key x or a server's secret key α. Therefore, it is hard that $\mathcal{B}$

© IEICE 2009
DOI: 10.1587/elex.6.237
Received December 19, 2008
Accepted January 13, 2009
Published March 10, 2009

guesses $H_1(w)^{1/x}$ or $H_1(w)$ from T_w.

5 Conclusion

We proposed that a secure dPEKS scheme against KG attacks. This result affirmatively answers the open problem proposed in Byun *et al.* [4].

Acknowledgments

This work was supported by the Korea Research Foundation Grant funded by the Korean Government [KRF-2008-357-D00258].

DOI: 10.1587/elex.6.237
Received December 19, 2008
Accepted January 13, 2009
Published March 10, 2009