An Efficient Searchable Encryption Scheme and Its Application in Network Forensics

Xiaodong Lin[1], Rongxing Lu[2], Kevin Foxton[1], and Xuemin (Sherman) Shen[2]

[1] Faculty of Business and Information Technology, University of Ontario Institute of Technology, Oshawa, Ontario, Canada L1H 7K4
{xiaodong.lin,kevin.foxton}@uoit.ca

[2] Department of Electrical and Computer Engineering, University of Waterloo, Waterloo, Ontario, Canada N2L 3G1
{rxlu,xshen}@bbcr.uwaterloo.ca

Abstract. Searchable encryption allows an encrypter to send a message, in an encrypted form, to a decryptor who can delegate to a third party to search the encrypted message for keywords without losing encrypted message content's privacy. In this paper, based on the bilinear pairings, we propose a new efficient searchable encryption scheme, and use the provable security technique to formally prove its security in the random oracle model. Since some time-consuming operations can be pre-computed, the proposed scheme is very efficient. Therefore, it is particularly suitable for time-critical applications, such as network forensics scenarios, especial when the content is encrypted due to privacy concerns.

Keywords: Searchable encryption, Network forensics, Provable security, Efficiency.

1 Introduction

Network forensics is a newly emerging forensics technology aiming at the capture, recording, and analysis of network events. This is done in order to discover the source of security attacks or other incidents occurring in networked systems [1]. There has been a growing interest in this field of forensics in recent years. Network forensics can help provide evidence to investigators to track back and prosecute the attack perpetrators by monitoring network traffic, determining a traffic anomaly, and ascertaining the attacks [2]. However, as an important element of a network investigation, network forensics is only applicable to environment where network security policies such as authentication, firewall, and intrusion detection systems have already been deployed. Large-volume traffic storage units are necessary as well, in order to hold the large amount of network information that is gathered during network operations. Once a perpetrator attacks a networked system, network forensics should immediately be launched by investigating the traffic data kept in the data storage units.

In order for effective network forensics, the storage units are required to maintain a complete record of all network traffic; unfortunately this slows down the investigation due to the amount of data that needs to be reviewed. In addition, to meet the security and privacy goals of a network, the network traffic needs to be encrypted and not removable

X. Lai et al. (Eds.): E-Forensics 2010, LNICST 56, pp. 66–78, 2011.

from the storage units. The network architecture needs to be setup in such way so that if an attacker compromises the storage unit, they still cannot view or edit the data's plaintext. Since the policy on storing traffic data in an encrypted manner produces negative effects on the efficiency of an investigation; we therefore need to determine how to efficiently make a post-mortem investigation on a large volume of encrypted traffic data. This is an ongoing challenge in the network forensics field.

Boneh et al. first introduced the concept of searchable encryption in 2004 [3]. They state that it is possible for an encryptor to send an encrypted message, in its encrypted form, to a decryptor who has the rights to decrypt the message, and that receiving decryptor can delegate to a third party to search for keywords in the encrypted message without losing the confidentiality of the message's content. Due to this promising feature, searchable encryption has been very active and many searchable encryption schemes have been proposed in recent years [4,5,6,7,8,9,10,11]. Obviously, searchable encryption can be applied in data forensics so that an authorized party can help collect the required encrypted evidence without the loss of confidentiality of the information. Before putting searchable encryption into use in data forensics, the efficiency issue must be resolved. For example, a large volume of network traffic could simultaneously come into a network/system; an encryptor should be able to quickly encrypt the network traffic and store it on storage units. However, many previously reported searchable encryption schemes require time-consuming pairing and MapToPoint hash operations [12] during the encryption process, which make them inefficient for data forensics scenarios. In this paper, motivated by the above mentioned points, we propose a new efficient searchable encryption scheme based on bilinear pairing. Due to its ability to handle some of the time-consuming operations in advance, and only requiring one point multiplication during real-time encryption, the proposed scheme is particularly suitable for data forensics applications. Specifically, the contributions of this paper are twofold:

- We propose an efficient searchable encryption scheme based on bilinear pairing, and use the provable security technique to formally prove its security through the use of the random oracle model [13].
- Due to the proposed scheme's efficiency in terms of the speed of encryption, we also discuss how to apply it to data forensics scenarios to resolve the challenging issue of data privacy while effectively locating valuable forensic data of interest.

The remainder of this paper is organized as follows. In Section 2, we review several related works on public key based searchable encryption. In Section 3, we formalize the definition of public key based searchable encryption and its corresponding security model. In Section 4, we review bilinear pairing and the complexity assumption, which is the basis of our proposed scheme. We present our efficient public key based searchable encryption scheme based on bilinear pairing, together with its formal security proof and efficiency analysis in Section 5. We discuss how to apply the proposed scheme in several network forensics scenarios that require the preservation of information confidentiality in Section 6. Finally, we draw our conclusions in Section 7.

2 Related Work

Recently, many research works on public key based searchable encryption have been appeared in literature [3,4,5,6,7,8,9,10,11]. The pioneering work of public-key based searchable encryption scheme is due to Boneh et al [3], where an entity, which is granted with some search capability, can search for encrypted keywords without revealing the content of the original data. Shortly after Boneh et al's work [3], Golle et al. [4] propose some provably secure schemes to allow for conjunctive keywords queries on encrypted data, and Park et al. [5] also propose public key encryption with conjunctive field keyword search in 2004. In 2005, Abdalla et al [6] further discuss the consistency property of searchable encryption, and give a generic construction by transforming an anonymous identity-based encryption scheme. In 2007, Boneh and Waters [7] extend the searchable encryption scheme to support conjunctive, subset, and range queries on encrypted data. Both Fuhr and Paillier [8] and Zhang et al. [9] investigate how to combine searchable encryption and public key encryption in a generic way. In [10], Hwang and Lee study the public key encryption with conjunctive keyword search and its extension to a multi-user system. In 2008, Bao et al. [11] further systematically study searchable encryption in a practical multi-user setting.

Differencing from the above works, we investigate a provably secure and efficient searchable encryption scheme and apply it to network forensics. Specifically, our proposed scheme does not require any costly MapToPoint hash operations [12], and supports pre-computation to improve the efficiency.

3 Definition and Security Model

3.1 Notations

Let $\mathbb{N} = \{1, 2, 3, \ldots\}$ denote the set of natural numbers. If $l \in \mathbb{N}$, then 1^l is the string of l 1s. If x, y are two strings, then $|x|$ is the length of x and $x\|y$ is the concatenation of x and y. If S is a finite set, $s \xleftarrow{R} S$ denotes sampling an element x uniformly at random from S. And if $\mathcal{A}$ is a randomized algorithm, $y \leftarrow \mathcal{A}(x_1, x_2, \ldots)$ means that $\mathcal{A}$ has inputs $x_1, x_2, \ldots$ and outputs y.

3.2 Definition and Security Model of Searchable Encryption

Informally, a searchable encryption ($\mathcal{SE}$) allows a receiver to delegate some search capability to a third-party so that the latter can help the receiver to search some keywords in an encrypted message without losing the message content's privacy. According to [3], a $\mathcal{SE}$ can be formally defined as follows.

Definition 1. *(Searchable Encryption) A searchable encryption ($\mathcal{SE}$) scheme consists of the following polynomial time algorithms:* SETUP*,* KGEN*,* PEKS*,* TRAPDOOR*, and* TEST*, where*

– SETUP*(l): Given the security parameter l, this algorithm generates the system parameter $params$.*

- KGEN$(params)$: *Given the system parameters* $params$*, this algorithm generates a pair of public and private keys* (pk, sk)*.*
- PEKS$(params, pk, w)$: *On input of the system parameters* $params$*, a public key* pk*, and a word* $w \in \{0,1\}^l$*, this algorithm produces a searchable encryption* C *of* w*.*
- TRAPDOOR$(params, sk, w)$: *On input of the system parameters* $params$*, a private key* sk*, and a word* w*, this algorithm produces a trapdoor* S_w *with respect to* w*.*
- TEST$(params, s_w, C)$: *On input of the system parameters* $params$*, a searchable encryption ciphertext* $C = \text{PEKS}(pk, w)$*, and a trapdoor* $S_{w'} = \text{TRAPDOOR}(sk, w')$*, this algorithm outputs "*`Yes`*" if* $w = w'$ *and "*`No`*" otherwise.*

Next, we define the security of $\mathcal{SE}$ in the sense of semantic-security under the adaptively chosen keyword attacks (IND-CKA), which ensures that $C = \text{PEKS}(pk, w)$ does not reveal any information about the keyword w unless S_w is available [3]. Especially, we consider the following interaction game run between an adversary $\mathcal{A}$ and a challenger. First, the adversary $\mathcal{A}$ is fed with the system parameters and public key, and can adaptively ask the challenger for the key trapdoor S_w for any keyword $w \in \{0,1\}^l$ of his choice. At a certain time, the adversary $\mathcal{A}$ chooses two un-queried keywords $w_0^\star, w_1^\star \in \{0,1\}^l$, on which it wishes to be challenged. The challenger flips a coin $b \in \{0,1\}$ and returns $C^\star = \text{PEKS}(pk, w_b^\star)$ to $\mathcal{A}$. The adversary $\mathcal{A}$ can continue to make key trapdoor query for any keyword $w \notin \{w_0^\star, w_1^\star\}$. Eventually, $\mathcal{A}$ outputs its guess $b' \in \{0,1\}$ on b and wins the game if $b = b'$.

Definition 2. *(IND-CKA Security) Let* l *and* t *be integers and* ϵ *be a real in* $[0,1]$*, and* $\mathcal{SE}$ *a secure searchable encryption scheme with security parameter* l*. Let* $\mathcal{A}$ *be an IND-CKA adversary, which is allowed to access the key trapdoor oracle* $\mathcal{O}_K$ *(and random oracle* $\mathcal{O}_H$ *in the random oracle model), against the semantic security of* $\mathcal{SE}$*. We consider the following random experiment:*

$$
\begin{array}{l}
\texttt{Experiment } \mathbf{Exp}^{\textit{IND-CKA}}_{\mathcal{SE},\mathcal{A}}(l) \\
\quad params \xleftarrow{R} \text{SETUP}(l) \\
\quad (pk, sk) \xleftarrow{R} \text{KGEN}(params) \\
\quad (w_0^\star, w_1^\star) \leftarrow \mathcal{A}^{\mathcal{O}_K(,\mathcal{O}_H)}(params, pk) \\
\quad b \xleftarrow{R} \{0,1\}, C^\star \leftarrow \text{PEKS}(pk, w_b^\star) \\
\quad b' \leftarrow \mathcal{A}^{\mathcal{O}_K(,\mathcal{O}_H)}(params, pk, C^\star) \\
\quad \texttt{if } b = b' \texttt{ then return } \ b^* \leftarrow 1 \texttt{ else } b^* \leftarrow 0 \\
\quad \texttt{return } b^*
\end{array}
$$

We define the success probability of $\mathcal{A}$ *via*

$$\mathbf{Succ}^{\textit{IND-CKA}}_{\mathcal{SE},\mathcal{A}}(l) = 2\Pr\left[\mathbf{Exp}^{\textit{IND-CKA}}_{\mathcal{SE},\mathcal{A}}(l)\right] - 1 = 2\Pr[b = b'] - 1$$

$\mathcal{SE}$ *is said to be* (l, t, ϵ)*-IND-CKA secure, if no adversary* $\mathcal{A}$ *running in time* t *has a success* $\mathbf{Succ}^{\textit{IND-CKA}}_{\mathcal{SE},\mathcal{A}}(l) \geq \epsilon$*.*

4 Bilinear Pairing and Complexity Assumptions

In this section, we briefly review the necessary facts about bilinear pairing and the complexity assumptions used in our scheme.

Bilinear Pairing. Let $\mathbb{G}$ be a cyclic additive group generated by P, whose order is a large prime q, and $\mathbb{G}_T$ be a cyclic multiplicative group with the same order q. An *admissible* bilinear pairing $e : \mathbb{G} \times \mathbb{G} \to \mathbb{G}_T$ is a map with the following properties:

1. *Bilinearity:* For all $P, Q \in \mathbb{G}$ and any $a, b \in \mathbb{Z}_q^*$, we have $e(aP, bQ) = e(P, Q)^{ab}$;
2. *Non-degeneracy:* There exists $P, Q \in \mathbb{G}$ such that $e(P, Q) \neq 1_{\mathbb{G}_T}$;
3. *Computability:* There is an efficient algorithm to compute $e(P, Q)$ for all $P, Q \in \mathbb{G}$.

Such an *admissible* bilinear pairing $e : \mathbb{G} \times \mathbb{G} \to \mathbb{G}_T$ can be implemented by the modified Weil or Tate pairings [12].

Complexity Assumptions. In the following, we define the quantitative notion of the complexity of the problems underlying the proposed scheme, namely the collusion attack algorithm with k traitors (k-CAA) Problem [14] and the decisional collusion attack algorithm with k traitors (k-DCAA) Problem.

Definition 3. *(k-CAA Problem) Let $(e, \mathbb{G}, \mathbb{G}_T, q, P)$ be a bilinear pairing tuple. The k-CAA Problem in $\mathbb{G}$ is as follows: for an integer k, and $x \in \mathbb{Z}_q$, given*

$$\left\{ P, Q = xP, h_1, h_2, \cdots, h_k \in \mathbb{Z}_q, \frac{1}{h_1 + x} P, \frac{1}{h_2 + x} P, \cdots, \frac{1}{h_k + x} P \right\}$$

to compute $\frac{1}{h^* + x} P$ for some $h^* \notin \{h_1, h_2, \cdots, h_k\}$.

Definition 4. *(k-CAA Assumption) Let $(e, \mathbb{G}, \mathbb{G}_T, q, P)$ be a bilinear pairing tuple, and $\mathcal{A}$ be an adversary that takes an input of $P, Q = xP, h_1, h_2, \cdots, h_k \in \mathbb{Z}_q$, $\frac{1}{h_1+x} P$, $\frac{1}{h_2+x} P$, $\cdots$, $\frac{1}{h_k+x} P$ for some unknown $x \in \mathbb{Z}_q^*$, and returns a new tuple $(h^*, \frac{1}{h^*+x} P)$ where $h^* \notin \{h_1, h_2, \cdots, h_k\}$. We consider the following random experiment.*

$\texttt{Experiment}\ \mathbf{Exp}_{\mathcal{A}}^{\mathrm{k-CAA}}$
$x \xleftarrow{R} \mathbb{Z}_q^*$,
$(h^*, \alpha) \leftarrow \mathcal{A}\left(P, Q = xP, h_1, h_2, \cdots, h_k \in \mathbb{Z}_q, \frac{1}{h_1+x} P, \frac{1}{h_2+x} P, \cdots, \frac{1}{h_k+x} P\right)$
$\texttt{if}\ \alpha = \frac{1}{h^*+x} P\ \texttt{then}\ b \leftarrow 1\ \texttt{else}\ b \leftarrow 0$
$\texttt{return}\ b$

We define the corresponding success probability of $\mathcal{A}$ in solving the k-CAA problem via

$$\mathbf{Succ}_{\mathcal{A}}^{\mathrm{k-CAA}} = \Pr\left[\mathbf{Exp}_{\mathcal{A}}^{\mathrm{k-CAA}} = 1\right]$$

Let $\tau \in \mathbb{N}$ and $\epsilon \in [0, 1]$. We say that the k-CAA is (τ, ϵ)-secure if no polynomial algorithm $\mathcal{A}$ running in time τ has success $\mathbf{Succ}_{\mathcal{A}}^{\mathrm{k-CAA}} \geq \epsilon$.

Definition 5. *(k-DCAA Problem) Let* $(e, \mathbb{G}, \mathbb{G}_T, q, P)$ *be a bilinear pairing tuple. The k-DCAA Problem in* $\mathbb{G}$ *is as follows: for an integer* k*, and* $x \in \mathbb{Z}_q$*, given*

$$\left\{P, Q = xP, h_1, h_2, \cdots, h_k, h^* \in \mathbb{Z}_q, \frac{1}{h_1+x}P, \frac{1}{h_2+x}P, \cdots, \frac{1}{h_k+x}P, T \in \mathbb{G}_T\right\}$$

to decide whether $T = e(P,P)^{\frac{1}{h^*+x}}$ *or a random element* R *drawn from* $\mathbb{G}_T$*.*

Definition 6. *(k-DCAA Assumption) Let* $(e, \mathbb{G}, \mathbb{G}_T, q, P)$ *be a bilinear pairing tuple, and* $\mathcal{A}$ *be an adversary that takes an input of* $P, Q = xP, h_1, h_2, \cdots, h_k, h^* \in \mathbb{Z}_q, \frac{1}{h_1+x}P, \frac{1}{h_2+x}P, \cdots, \frac{1}{h_k+x}P, T \in \mathbb{G}_T$ *for unknown* $x \in \mathbb{Z}_q^*$*, and returns a bit* $b' \in \{0,1\}$*. We consider the following random experiments.*

$\texttt{Experiment}\ \mathbf{Exp}_{\mathcal{A}}^{\mathrm{k-DCAA}}$
$x, h_1, h_2, \cdots, h_k, h \xleftarrow{R} \mathbb{Z}_q; R \xleftarrow{R} \mathbb{G}_T$
$\widetilde{b} \leftarrow \{0,1\}$
if $\widetilde{b} = 0$*, then* $T = e(P,P)^{\frac{1}{h^*+x}}$*; else if* $\widetilde{b} = 1$ *then* $T = R$
$\widetilde{b}' \leftarrow \mathcal{A}\left(P, Q = xP, h_1, h_2, \cdots, h_k, h \in \mathbb{Z}_q, \frac{1}{h_1+x}P, \frac{1}{h_2+x}P, \cdots, \frac{1}{h_k+x}P, T\right)$
$\texttt{return}\ 1$ *if* $\widetilde{b}' = \widetilde{b}$*,* 0 *otherwise*

We then define the advantage of $\mathcal{A}$ *via*

$$\mathbf{Adv}_{\mathcal{A}}^{\mathrm{k-DCAA}} = \left|\Pr\left[\mathbf{Exp}_{\mathcal{A}}^{\mathrm{k-DCAA}} = 1 | \widetilde{b} = 0\right] - \Pr\left[\mathbf{Exp}_{\mathcal{A}}^{\mathrm{k-DCAA}} = 1 | \widetilde{b} = 1\right]\right| \geq \epsilon$$

Let $\tau \in \mathbb{N}$ *and* $\epsilon \in [0,1]$*. We say that the k-DCAA is* (τ, ϵ)*-secure if no adversary* $\mathcal{A}$ *running in time* τ *has an advantage* $\mathbf{Adv}_{\mathcal{A}}^{\mathrm{k-DCAA}} \geq \epsilon$*.*

5 New Searchable Encryption Scheme

In this section, we will present our efficient searchable encryption scheme based on bilinear pairing, followed by its security proof and performance analysis.

5.1 Description of The Proposed Scheme

Our searchable encryption ($\mathcal{SE}$) scheme mainly consists of five algorithms, namely SETUP, KGEN, PEKS, TRAPDOOR and TEST, as shown in Fig. 1.

SETUP. Given the security parameter l, 5-tuple bilinear pairing parameters $(e, \mathbb{G}, \mathbb{G}_T, q, P)$ are first chosen such that $|q| = l$. Then, a secure cryptographic hash function $\mathcal{H}$ is also chosen, where $\mathcal{H} : \{0,1\}^l \rightarrow \mathbb{Z}_q^*$. In the end, the system parameters $params = (e, \mathbb{G}, \mathbb{G}_T, q, P, \mathcal{H})$ are published.

KGEN. Given the system parameters $params = (e, \mathbb{G}, \mathbb{G}_T, q, P, \mathcal{H})$, choose a random number $x \in \mathbb{Z}_q^*$ as the private key, and compute the corresponding public key $Y = xP$.

PEKS. Given a key $w \in \{0,1\}^l$ and the public key Y, choose a random number $r \in \mathbb{Z}_q^*$, and execute the following steps:

SETUP	KGEN
SETUP(l) $\rightarrow$system parameters $params = (e, \mathbb{G}, \mathbb{G}_T, q, P, \mathcal{H})$	system parameters $params \rightarrow$ private key $x \in \mathbb{Z}_q^*$ public key $Y = xP$
PEKS	TRAPDOOR
for a keyword $w \in \{0,1\}^l$ choose a random number $r \in \mathbb{Z}_q^*$ $\alpha = r \cdot (Y + \mathcal{H}(w)P)$, $\beta = e(P,P)^r$ $C = (\alpha, \beta)$	trapdoor for keyword w: $S_w = \frac{1}{x+\mathcal{H}(w)}P$
	TEST
	test if $\beta = e(\alpha, S_w)$ if so, output "Yes"; if not, output "No".

Fig. 1. Proposed searchable encryption ($\mathcal{SE}$) scheme

- compute (α, β) such that $\alpha = r \cdot (Y + \mathcal{H}(w)P)$, $\beta = e(P,P)^r$,
- set the ciphertext $C = (\alpha, \beta)$.

TRAPDOOR. Given the keyword $w \in \{0,1\}^l$ and the public and private key pairs (Y, x), compute the keyword w's trapdoor $S_w = \frac{1}{x+\mathcal{H}(w)}P$.

TEST. Given the ciphertext $C = (\alpha, \beta)$ and the keyword w's trapdoor $S_w = \frac{1}{x+\mathcal{H}(w)}P$, check if $\beta = e(\alpha, S_w)$. If the equation holds, "Yes" is output; otherwise, "No" is output. The correctness is as follows,

$$e(\alpha, S_w) = e\left(r \cdot (Y + \mathcal{H}(w)P), \frac{1}{x+\mathcal{H}(w)}P\right) = e\left(xP + \mathcal{H}(w)P, \frac{1}{x+\mathcal{H}(w)}P\right)^r$$
$$= e(P,P)^r = \beta$$

Consistency. Since $\mathcal{H}()$ is a secure hash function, the probability that $\mathcal{H}(w_0) = \mathcal{H}(w_1)$ can be negligible for any two keywords $w_0, w_1 \in \{0,1\}^l$ and $w_0 \neq w_1$. Therefore, $S_{w_0} = \frac{1}{x+\mathcal{H}(w_0)}P \neq \frac{1}{x+\mathcal{H}(w_1)}P = S_{w_1}$, and the TEST algorithm outputs "Yes" on input of a trapdoor for w_0 and a $\mathcal{SE}$ ciphertext C of w_1 is negligible. As a result, the consistency follows.

5.2 Security Proof

In the following theorem, we will prove that the ciphertext $C = (\alpha, \beta)$ is IND-CKA-secure in the random oracle model, where the hash function $\mathcal{H}$ is modelled as random oracle [13].

Theorem 1. *(IND-CKA Security) Let $k \in \mathbb{N}$ be an integer, and $\mathcal{A}$ be an adversary against the proposed $\mathcal{SE}$ scheme in the random oracle model, where the hash function $\mathcal{H}$ behaves as random oracle. Assume that $\mathcal{A}$ has the success probability $\mathbf{Succ}^{ind\text{-}cka}_{\mathrm{SE},\mathcal{A}} \geq \epsilon$ to break the indistinguishability of the ciphertext $C = (\alpha, \beta)$ within the running time τ, after $q_H = k + 2$ and $q_K \leq k$ queries to the random oracle $\mathcal{O}_H$ and the key trapdoor oracle $\mathcal{O}_K$, respectively. Then, there exist $\epsilon' \in [0,1]$ and $\tau' \in \mathbb{N}$ as follows*

$$\epsilon' = \mathbf{Adv}^{\mathrm{k-DCAA}}_{\mathcal{A}}(\tau') \geq \frac{\epsilon}{q_H(q_H - 1)}, \qquad \tau' \leq \tau + \Theta(.) \tag{1}$$

such that the k-DCAA problem can be solved with probability ϵ' within time τ', where $\Theta(.)$ is the time complexity for the simulation.

Proof. We define a sequence of games $\mathbf{Game}_0$, $\mathbf{Game}_1$, $\cdots$ of modified attacks starting from the actual adversary $\mathcal{A}$ [15]. All the games operate on the same underlying probability space: the system parameters $params = (e, \mathbb{G}, \mathbb{G}_T, q, P, \mathcal{H})$ and public key $Y = xP$, the coin tosses of $\mathcal{A}$. Let $(P, xP, h_1, h_2, \cdots, h_k, h^* \in \mathbb{Z}_q^*, \frac{1}{h_1+x}P, \frac{1}{h_2+x}P, \cdots, \frac{1}{h_k+x}P, T \in \mathbb{G}_T)$ be a random instance of k-DCAA problem, we will use these incremental games to reduce the k-DCAA instance to the adversary $\mathcal{A}$ against the IND-CKA security of the ciphertext $C = (\alpha, \beta)$ in the proposed $\mathcal{SE}$ scheme.

$\mathbf{Game}_0$: This is a real attack game. In the game, the adversary $\mathcal{A}$ is fed with the system parameters $params = (e, \mathbb{G}, \mathbb{G}_T, q, P, \mathcal{H})$ and public key $Y = xP$. In the first phase, the adversary $\mathcal{A}$ can access to the random oracle $\mathcal{O}_H$ and the key trapdoor oracle $\mathcal{O}_K$ for any input. At some point, the adversary $\mathcal{A}$ chooses a pair of keywords $(w_0^\star, w_1^\star) \in \{0,1\}^l$. Then, we flip a coin $b \in \{0,1\}$ and produce the message $w^\star = w_b^\star$'s ciphertext $C^\star = (\alpha^\star, \beta^\star)$ as the challenge to the adversary $\mathcal{A}$. The challenge comes from the public key Y and one random number $r^\star \in \mathbb{Z}_q^*$, and $\alpha^\star = r^\star \cdot (Y + \mathcal{H}(w^\star)P)$, $\beta^\star = e(P,P)^{r^\star}$. In the second stage, the adversary $\mathcal{A}$ is still allowed to access to the random oracle $\mathcal{O}_H$, and the key trapdoor oracle $\mathcal{O}_K$ for any input, except the challenge $(w_0^\star, w_1^\star)$. Finally, the adversary $\mathcal{A}$ outputs a bit $b' \in \{0,1\}$. In any $\mathbf{Game}_j$, we denote by $\mathbf{Guess}_j$ the event $b = b'$. Then, by definition, we have

$$\epsilon \leq \mathbf{Succ}_{\mathrm{SE},\mathcal{A}}^{\text{ind-cka}} = 2\Pr[b = b']_{\mathbf{Game}_0} - 1 = 2\Pr[\mathbf{Guess}_0] - 1 \tag{2}$$

$\mathbf{Game}_1$: In the simulation, we know the adversary $\mathcal{A}$ makes a total of $q_H = k + 2$ queries on $\mathcal{O}_H$, two of which are the queries of the challenge $(w_0^\star, w_1^\star)$. In this game, we consider that we successfully guess the challenge $(w_0^\star, w_1^\star)$ from q_H queries $(w_1, w_2, \cdots, w_{q_H})$ in advance, then the probability of successful guessing $(w_0^\star, w_1^\star)$ is $1/\binom{q_H}{2} = \frac{2}{q_H(q_H-1)}$. Then, in this game, we have

$$\begin{aligned} \frac{2}{q_H(q_H-1)}\mathbf{Succ}_{\mathrm{SE},\mathcal{A}}^{\text{ind-cka}} &= 2\Pr[b = b']_{\mathbf{Game}_1} - 1 = 2\Pr[\mathbf{Guess}_1] - 1, \\ \Pr[\mathbf{Guess}_1] &= \frac{1}{q_H(q_H-1)} \cdot \mathbf{Succ}_{\mathrm{SE},\mathcal{A}}^{\text{ind-cka}} + \frac{1}{2} \geq \frac{\epsilon}{q_H(q_H-1)} + \frac{1}{2} \end{aligned} \tag{3}$$

$\mathbf{Game}_2$: In this game, we simulate the random oracle $\mathcal{O}_H$ and the key trapdoor oracle $\mathcal{O}_K$, by maintaining the lists $\mathcal{H}$-List and $\mathcal{K}$-List to deal with the identical queries. In addition, we also simulate the way that the challenges $C^\star$ is generated as the challenger would do. The detailed simulation in this game is described in Fig. 2. Because the distribution of $(params, Y)$ is unchanged in the eye of the adversary $\mathcal{A}$, the simulation is perfect, and we have

$$\Pr[\mathbf{Guess}_2] = \Pr[\mathbf{Guess}_1] \tag{4}$$

$\mathbf{Game}_3$: In this game, we modify the rule Key-Gen in the key trapdoor oracle $\mathcal{O}_K$ simulation without resorting to the private key x.

▷ **Rule** Key-Gen$^{(3)}$
look up the item $\frac{1}{h+x}P$ in $\{\frac{1}{h_1+x}P, \frac{1}{h_2+x}P, \cdots, \frac{1}{h_k+x}P\}$
set $S_w = \frac{1}{h+x}P$
answer S_w and add (w, S_w) to $\mathcal{K}$-List

Because q_K, the total key trapdoor query number, is less than or equal to k, the item $S_w = \frac{1}{h+x}P$ always can be found in the simulation due to the k-DCAA problem. Therefore, these two games $\mathbf{Game}_3$ and $\mathbf{Game}_2$ are perfectly indistinguishable, and we have

$$\Pr[\mathbf{Guess}_3] = \Pr[\mathbf{Guess}_2] \tag{5}$$

$\mathbf{Game}_4$: In this game, we manufacture the challenge $C^\star = (\alpha^\star, \beta^\star)$ by embedding the k-DCAA challenge $(h^*, T \in \mathbb{G}_T)$ in the simulation. Specifically, after flipping $b \in \{0,1\}$ and choosing $r^\star \in \mathbb{Z}_q^*$, we modify the rule Chal in the Challenger simulation and the rule No-H in the $\mathcal{O}_H$ simulation.

▷ **Rule** Chal$^{(4)}$
- $\alpha^\star = r^\star P, \beta^\star = T^{r^\star}$
- set the ciphertext $C^\star = (\alpha^\star, \beta^\star)$

▷ **Rule** No-H$^{(4)}$
- if $w \notin (w_0^\star, w_1^\star)$
  - randomly choose a fresh h from the set $\mathbb{H} = \{h_1, h_2, \cdots, h_k\}$
  - the record (w, h) will be added in $\mathcal{H}$-List
- else if $w \in (w_0^\star, w_1^\star)$
  - if $w = w_b^\star$
    - set $h = h^*$, the record (w, h) will be added in $\mathcal{H}$-List
  - else if $w = w_{b-1}^\star$
    - randomly choose a fresh random number h from $\mathbb{Z}_q^*/(\mathbb{H} \cup \{h^*\})$
    - the record (w, h) will be added in $\mathcal{H}$-List

Based on the above revised rules, if T in the k-DCAA challenge is actually $e(P,P)^{\frac{1}{h^*+x}}$, i.e., $\widetilde{b} = 0$ in the `Experiment` $\mathbf{Exp}_{\mathcal{A}}^{\mathrm{k-DCAA}}$, we know that

$$C^\star = \left(\alpha^\star = r^\star P, \beta^\star = T^{r^\star} = e(P,P)^{\frac{r^\star}{h^*+x}}\right)$$

is a valid ciphertext, which will pass the Test equation $\beta^\star = e(\alpha^\star, S_{w_b^\star})$, where $S_{w_b^\star} = T = e(P,P)^{\frac{1}{h^*+x}}$. Therefore, we have

$$\Pr[\mathbf{Guess}_4 | \widetilde{b} = 0] = \Pr[\mathbf{Guess}_3]. \tag{6}$$

and

$$\Pr\left[\mathbf{Exp}_{\mathcal{A}}^{\mathrm{k-DCAA}} = 1 | \widetilde{b} = 0\right] = \Pr[\mathbf{Guess}_4 | \widetilde{b} = 0] \tag{7}$$

If T in the k-DCAA challenge is a random element in $\mathbb{G}_T$ other than $e(P,P)^{\frac{1}{h^*+x}}$, i.e., $\widetilde{b} = 1$ in the `Experiment` $\mathbf{Exp}_{\mathcal{A}}^{\mathrm{DBDH}}$, $C^\star = \left(\alpha^\star = r^\star P, \beta^\star = T^{r^\star}\right)$ is not a valid ciphertext, and thus is independent on b. Therefore, we will have

$$\Pr\left[\mathbf{Exp}_{\mathcal{A}}^{\mathrm{k-DCAA}} = 1 | \widetilde{b} = 1\right] = \Pr[\mathbf{Guess}_4 | \widetilde{b} = 1] = \frac{1}{2}. \tag{8}$$

As a result, from Eqs. (3)-(8), we have

$$\begin{aligned}\epsilon' &= \mathbf{Adv}_{\mathcal{A}}^{\mathrm{k-DCAA}} \\ &= \left|\Pr\left[\mathbf{Exp}_{\mathcal{A}}^{\mathrm{k-DCAA}} = 1|\widetilde{b}=0\right] - \Pr\left[\mathbf{Exp}_{\mathcal{A}}^{\mathrm{k-DCAA}} = 1|\widetilde{b}=1\right]\right| \\ &\geq \left|\frac{\epsilon}{q_H(q_H-1)} + \frac{1}{2} - \frac{1}{2}\right| = \frac{\epsilon}{q_H(q_H-1)}\end{aligned} \tag{9}$$

In addition, we can obtain the claimed bound for $\tau' \leq \tau + \Theta(.)$ in the sequence games. Thus, the proof is completed. □

Query to Oracle $\mathcal{O}_H$

Query $\mathcal{H}(w)$: if a record (w, h) has already appeared in $\mathcal{H}$-List, the answer is returned with the value of h.

Otherwise the answer h is defined according to the following rule:

▷ **Rule** No-H$^{(2)}$

if $w \notin (w_0^\star, w_1^\star)$
randomly choose a fresh h from the set $\mathbb{H} = \{h_1, h_2, \cdots, h_k\}$
the record (w, h) will be added in $\mathcal{H}$-List
else if $w \in (w_0^\star, w_1^\star)$
randomly choose a fresh random number h from $\mathbb{Z}_q^*/(\mathbb{H} \cup \{h^*\})$
the record (w, h) will be added in $\mathcal{H}$-List

Query to Oracle $\mathcal{O}_K$

Query $\mathcal{O}_K(w)$: if a record (w, S_w) has already appeared in $\mathcal{K}$-List, the answer is returned with S_w.

Otherwise the answer S_w is defined according to the following rules:

▷ **Rule** Key-Init$^{(2)}$

Look up for$(w, h) \in \mathcal{H}$-List
if the record (w, h) is unfound
same as the rule of query to Oracle $\mathcal{O}_H$

▷ **Rule** Key-Gen$^{(2)}$

Use the private key $sk = x$ to compute $S_w = \frac{1}{x+h}P$

Answer S_w and add (w, S_w) to $\mathcal{K}$-List

Challenger

For two keywords $(w_0^\star, w_1^\star) \in \mathbb{Z}_q^*$, flip a coin $b \in \{0, 1\}$ and set $w^\star = w_b^\star$, randomly choose $r^\star \in \mathbb{Z}_q^*$, then answer $C^\star$, where

▷ **Rule** Chal$^{(2)}$

$\alpha^\star = r^\star \cdot (Y + \mathcal{H}(w_b^\star)P)\,, \beta^\star = e(P, P)^{r^\star}$
set the ciphertext $C^\star = (\alpha^\star, \beta^\star)$

Fig. 2. Formal simulation of the IND-CKA game against the proposed $\mathcal{SE}$ scheme

5.3 Efficiency

Our proposed $\mathcal{SE}$ scheme is particularly efficient in terms of the computational costs. As shown in Fig. 1, the PEKS algorithm requires two point multiplications in $\mathbb{G}$ and one pairing operation. Because $\alpha = r \cdot (Y + \mathcal{H}(w)P) = rY + \mathcal{H}(w)(rP)$, the items rY, rP together with $\beta = e(P, P)^r$, which are irrelative to the keyword w, can be pre-computed. Then, only one point multiplication is required at PEKS. In addition, the TRAPDOOR and TEST algorithms also only require one point multiplication, one pairing operation, respectively. Table 1 shows the computational complexity between the scheme in [3] and our proposed scheme, where we consider point multiplication in $\mathbb{G}$, exponentiation in $\mathbb{G}_T$, pairing, and MapToPoint hash operation [12], but omit miscellaneously small computation operations such as point addition and ordinary hash function $\mathcal{H}$ operation. Then, from the figure, we can see our proposed scheme is more efficient, especially when the pre-computation is considered since T_{pmul} is much smaller than $T_{\mathsf{pair}} + T_{\mathsf{m2p}}$ in many software implementations.

Table 1. Computational cost comparisons

	Scheme in [3]	Proposed scheme
PEKS (w.o. precomputation)	$2 \cdot T_{\mathsf{pmul}} + T_{\mathsf{pair}} + T_{\mathsf{m2p}}$	$2 \cdot T_{\mathsf{pmul}} + T_{\mathsf{exp}}$
PEKS (with precomputation)	$T_{\mathsf{pair}} + T_{\mathsf{m2p}}$	T_{pmul}
TRAPDOOR	$T_{\mathsf{pmul}} + T_{\mathsf{m2p}}$	T_{pmul}
TEST	T_{pair}	T_{pair}

T_{pmul}: time cost of point multiplication in $\mathbb{G}$; T_{pair}: time cost of one pairing;
T_{m2p}: time cost of MapToPoint hash; T_{exp}: time cost of exponentiation in $\mathbb{G}_T$

6 Application in Network Forensics

In this section, we discuss how to apply our proposed searchable encryption $\mathcal{SE}$ scheme to network forensics. As shown in Fig. 3, the network forensics system that we consider mainly consists of a top-level administrator, an investigator and two security modules resided in each network service. The network service consists of the user authentication module and the traffic monitoring module, where the user authentication module takes the responsibility for the user authentication, and the traffic monitoring module is monitoring and logging all user activities in the system. In general, network forensics used in a system can be divided into three phases: network user authentication phase, traffic logging phase, and network investigation phase. Each of the phases is detailed as follows:

– Network user authentication phase: when an Internet user with identity U_i visits a network service, the residing user authentication module will authenticate the user. If the user passes the authentication, he can access the service. Otherwise, the user is prohibited from accessing the service.

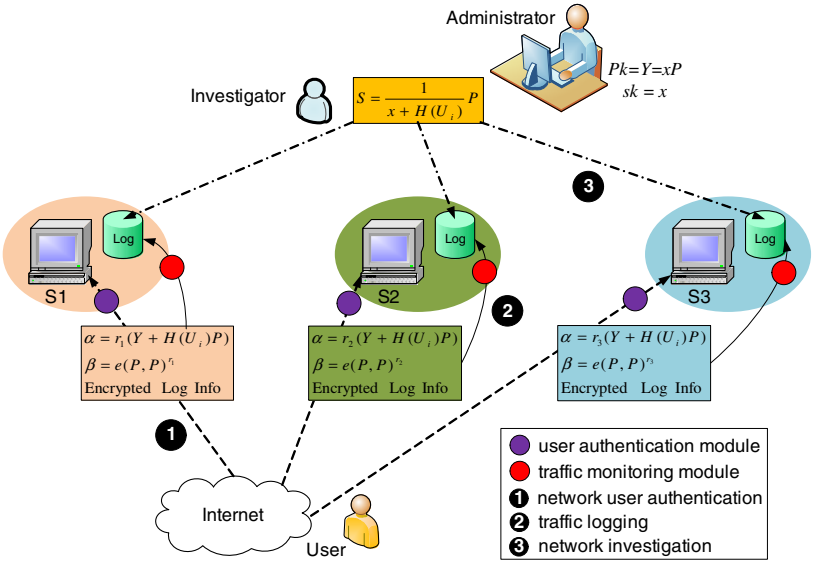

Fig. 3. Network forensics enhanced with searchable encryption

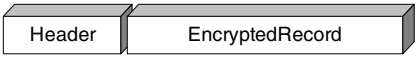

Fig. 4. The format of encrypted record

- Traffic logging phase: when the network service is idle, the traffic monitoring module precomputes a huge number of tuples, each tuple is of the form $(rY, rP, \beta = e(P,P)^r)$, where $r \in \mathbb{Z}_q^*$ and Y is the public key of the administrator. When an authenticated user U_i runs some actions with the service, the traffic monitoring module will pick up a tuple $(rY, rP, \beta = e(P,P)^r)$, compute $\alpha = rY + \mathcal{H}(U_i)rP$, create the logging record in the format as shown in Fig. 4, where Header $:= (\alpha, \beta)$ and EncryptedRecord $:= U_i$'s actions encrypted with the administrator's public key Y. After the user's actions are encrypted, the logged record is stored in the storage units.
- Network investigation phase: once the administrator suspects that an authenticated user U_i could have been compromised by an attacker, he should collect evidence on all actions that U_i did in the past. Therefore, the administrator needs to authorize an investigator to collect the evidences at each service's storage units. However, because U_i is still just under suspicion, the administrator cannot let the investigator know U_i's identity. To address this privacy issue, the administrator grants $S = \frac{1}{x+H(U_i)} P$ to the investigator, and the latter can collect all the required records satisfying $\beta = e(\alpha, S)$. After recovering the collected records from the investigator, the administrator can then do forensics analysis on the data. Obviously, such network forensics enhanced with our proposed searchable encryption can work well in terms of forensics analysis, audit, and privacy preservation.

7 Conclusions

In this paper, we have proposed an efficient searchable encryption ($\mathcal{SE}$) scheme based on bilinear pairings, and have formally shown its security with the provable security technique under k-DCAA assumption. Due to the fact that it supports pre-computation, i.e., only one point multiplication and one pairing are required in PEKS and TEST algorithms, respectively, the proposed scheme is much efficient and particularly suitable to resolve the challenging privacy issues in network forensics.

References

1. Ranum, M.: Network flight recorder, `http://www.ranum.com/`
2. Pilli, E. S., Joshi, R.C., Niyogi, R.: Network forensic frameworks: Survey and research challenges. Digitial Investigation (in press, 2010)
3. Boneh, D., Di Crescenzo, G., Ostrovsky, R., Persiano, G.: Public key encryption with keyword search. In: Cachin, C., Camenisch, J.L. (eds.) EUROCRYPT 2004. LNCS, vol. 3027, pp. 506–522. Springer, Heidelberg (2004)
4. Golle, P., Staddon, J., Waters, B.: Secure conjunctive keyword search over encrypted data. In: Jakobsson, M., Yung, M., Zhou, J. (eds.) ACNS 2004. LNCS, vol. 3089, pp. 31–45. Springer, Heidelberg (2004)
5. Park, D.J., Kim, K., Lee, P.J.: Public key encryption with conjunctive field keyword search. In: Lim, C.H., Yung, M. (eds.) WISA 2004. LNCS, vol. 3325, pp. 73–86. Springer, Heidelberg (2005)
6. Abdalla, M., Bellare, M., Catalano, D., Kiltz, E., Kohno, T., Lange, T., Malone-Lee, J., Neven, G., Paillier, P., Shi, H.: Searchable encryption revisited: Consistency properties, relation to anonymous IBE, and extensions. In: Shoup, V. (ed.) CRYPTO 2005. LNCS, vol. 3621, pp. 205–222. Springer, Heidelberg (2005)
7. Boneh, D., Waters, B.: Conjunctive, subset, and range queries on encrypted data. In: Vadhan, S.P. (ed.) TCC 2007. LNCS, vol. 4392, pp. 535–554. Springer, Heidelberg (2007)
8. Fuhr, T., Paillier, P.: Decryptable searchable encryption. In: Susilo, W., Liu, J.K., Mu, Y. (eds.) ProvSec 2007. LNCS, vol. 4784, pp. 228–236. Springer, Heidelberg (2007)
9. Zhang, R., Imai, H.: Generic combination of public key encryption with keyword search and public key encryption. In: Bao, F., Ling, S., Okamoto, T., Wang, H., Xing, C. (eds.) CANS 2007. LNCS, vol. 4856, pp. 159–174. Springer, Heidelberg (2007)
10. Hwang, Y.-H., Lee, P.J.: Public key encryption with conjunctive keyword search and its extension to a multi-user system. In: Takagi, T., Okamoto, T., Okamoto, E., Okamoto, T. (eds.) Pairing 2007. LNCS, vol. 4575, pp. 2–22. Springer, Heidelberg (2007)
11. Feng Bao, F., Deng, R.H., Ding, X., Yang, Y.: Private query on encrypted data in multi-user settings. In: Chen, L., Mu, Y., Susilo, W. (eds.) ISPEC 2008. LNCS, vol. 4991, pp. 71–85. Springer, Heidelberg (2008)
12. Boneh, D., Franklin, M.: Identity-based encryption from the weil pairing. In: Kilian, J. (ed.) CRYPTO 2001. LNCS, vol. 2139, pp. 213–229. Springer, Heidelberg (2001)
13. Bellare, M., Rogaway, P.: Random Oracles are Practical: A Paradigm for Designing Efficient Protocols. In: ACM Computer and Communications Security Conference, CCS 1993, Fairfax, Virginia, USA, pp. 62–73 (1993)
14. Zhang, F., Safavi-Naini, R., Susilo, W.: An efficient signature scheme from bilinear pairings and its applications. In: Bao, F., Deng, R., Zhou, J. (eds.) PKC 2004. LNCS, vol. 2947, pp. 277–290. Springer, Heidelberg (2004)
15. Shoup, V.: OAEP Reconsidered. Journal of Cryptology 15, 223–249 (2002)