PVSAE: A Public Verifiable Searchable Encryption Service Framework for Outsourced Encrypted Data

Rui Zhang*, Rui Xue*, Ting Yu†‡, and Ling Liu§
*SKLOIS, Institute of Information Engineering, Chinese Academy of Sciences, Beijing, China
Email: zhangrui,xuerui@iie.ac.cn
†Qatar Computing Research Institute and Hamad Bin Khalifa University, Doha, Qatar, Email: tyu@qf.org.qa
§College of Computing, Georgia Institute of Technology, Atlanta, GA, USA, Email: lingliu@cc.gatech.edu

***Abstract*—Outsource encrypted data is a popular trend for storing sensitive data in third party clouds. Many cloud applications need privacy preserving data encryption services with two capabilities: On one hand, they need querying over encrypted data in Web based data hosting services. On the other hand, they also need to keep the query keywords and associated search operations private such that data hosting service providers cannot gain access to unauthorized content or trace and infer sensitive data stored in the third party data hosting servers. In this paper we present a novel service oriented framework for verifiable searchable asymmetric encryption, called PVSAE. PVSAE offers strong support for outsourced encrypted data with two formal security properties in terms of IND-CKA security and search pattern privacy. Our framework supports two concrete PVSAE schemes. The first scheme ℓ-PVSAE is based on the ℓ-dimensional vectors and achieves strong security notions, namely statistical IND-CKA security and statistical search pattern privacy. The second scheme 3-PVSAE is a light-weight version based on 3-dimensional vectors. 3-PVSAE maintains the strong security properties and offers higher efficiency for search over encrypted data compared with existing verifiable searchable asymmetric encryption schemes. We experimentally evaluate the proposed PVSAE schemes and show that they not only offer strong security but also are practical and deployable.**

***Index Terms*—searchable asymmetric encryption; verifiable computation; IND-CKA security; search pattern privacy;**

I. Introduction

With the proliferation of a new breed of Web-enabled big data services and applications that store and process data at remote service providers, the search over encrypted data has emerged as an important research problem of growing interests. In a typical storage-as-a-service scenario, a query generated at the client side is rewritten into a transformed representation such that it can be evaluated directly over encrypted data at the remote cloud service provider. The returned results are typically processed at the client after decryption to obtain the final answers.

In this work, we focus on the public-key scenario, where a data owner encrypts index with data user's public key and the data user can issue queries with his own private key, so that the data owner can authorize queries to data user without sharing the secret key through a secure channel.

In addition, to prevent misuse and abuse of outsourced data stored in cloud data centers, such as intentional data removal or malicious intent, the capability to allow data users to verify the correctness of their searching results returned by the cloud server becomes critical for data owners to outsource the sensitive data into third party cloud storage hosting services. Moreover, many cloud applications require to allow their users to verify whether the cloud has faithfully executed their search operations or not, while keeping both outsourced data encrypted and the queries over encrypted data private [1], [2].

Based on the above introduction, the desirable security properties of queries over encrypted data in the public-key setting are as follows: (1) Confidentiality: the cloud cannot learn anything about the data as well as the query; (2) Correctness: the results must be sound and complete (items returned satisfy the query and all such items are returned); (3) Public verifiability: anyone (not necessarily the data user originating the search query) can check the cloud's responses.

However, to the best of our knowledge, no existing public-key encryption with keyword search (PEKS) schemes can achieve all three properties at the same time. Although Zheng *et al*. [3] and Liu *et al*. [4] respectively proposed the first verifiable attribute-based keyword search (VABKS) schemes, their schemes only provide private verification, that is, only the data user with secret key can verify whether or not the cloud has faithfully executed the keyword search operation.

In this paper, we propose a novel Public Verifiable Searchable Asymmetric Encryption service framework, called PVSAE, to satisfy all the above requirements. Our schemes allow a data owner to control the search of its outsourced encrypted data by building the secure index with data user's public key, while allowing the legitimate data user to outsource the costly search operations to the cloud and anyone with verification key can verify whether or not the cloud has faithfully returned the right result. Most importantly, the process of search and verification will not leak any information about the query and the outsourced data.

A. System Model

We construct our schemes on the inverted index data structure, which is an index data structure storing a mapping from content, such as keywords to a set of files. Let $D = \{d_1, d_2, \cdots, d_m\}$ be the set of files to be stored in an untrusted cloud, where $|D| = m$ is the total number of files. Each file d_i contains a set of keywords. Let $W = \{w_1, w_2, \cdots, w_n\}$ be the set of keywords in D, where $|W| = n$ is the total number

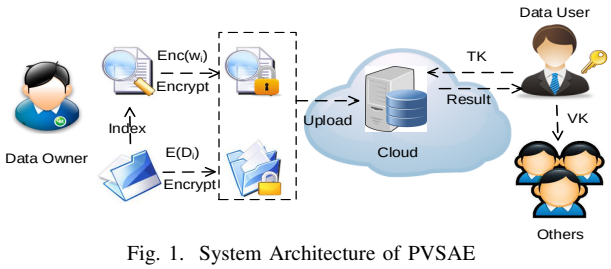


Fig. 1. System Architecture of PVSAE

of keywords. $D_{i\in[n]} \subseteq D$ denotes the set of files containing keyword w_i.

Figure 1 shows the overall system architecture. To outsource a set of files to the cloud, the data owner builds a secure searchable index for the file sets and then uploads the encrypted files, together with the secure index, to the cloud. Note that the keywords and the sets of files are respectively encrypted by encryption algorithms Enc and E, where Enc is a public verifiable searchable asymmetric encryption scheme proposed in this paper and E can be any semantically secure asymmetric encryption scheme. The secure index is expressed as $I = \{I_{w_1}, \cdots, I_{w_n}\} = \{\mathsf{Enc}(w_1), \mathsf{Enc}(w_2), \cdots, \mathsf{Enc}(w_n)\}$. A data user who with the secret key can issue a query with token TK for the files that containing keyword w to the cloud. In addition, the data user generates a verification key VK for this query. Upon receiving the search token, the cloud executes the search algorithm over the secure index and returned the matched files to the data user as the search result. Anyone obtains the verification key VK and returned result can verify the correctness of search operation of the cloud.

B. Security Model

Unlike the security model of traditional searchable encryption schemes as in [5], [6], we adopt the "malicious" model for the cloud, which is untrusted by users in the following sense: (1) The cloud may delete a part of encrypted files or index to save storage space. (2) The cloud may not use the input that it is provided when performing keyword search. For example, to save its computation or download bandwidth, the cloud may execute only a fraction of queries and return a fraction of the search result. In addition, the cloud may forge the search result to cheat the users. (3) The cloud is considered curiously, it may try to analyze data in its storage and message flows in order to learn additional information.

C. Design Goal

Based on the "malicious" model of the cloud, our design bears the following security and performance goals.

- **Data Privacy:** Data privacy is a basic requirement which requires the data to be outsourced should not be revealed to any unauthorized parties including cloud service providers. Typically, it can be guaranteed by public-key encryption algorithms. The user who has the secret key can effectively decrypt the encoded data after retrieving them from the cloud.
- **Keyword Privacy:** If the cloud deduces any association between frequent keywords and encrypted dataset from the index, it may learn the main content of a file. Therefore, searchable index should be constructed in such a way that prevents the cloud from performing such kind of association attacks. In PEKS schemes [5]–[8], this kind of security is called security against chosen keyword attack (CKA). In our schemes, the randomness of encrypted index guarantees to prevent such attacks from the cloud.
- **Search Pattern Privacy:** Data users usually prefer to keep their query from being exposed to others, i.e., the keyword indicated by the corresponding token. In the literature [8], this kind of leakage is called search pattern. Namely search patterns reveal whether the same search was performed in the past or not. Like existing PEKS schemes [5]–[8], our schemes use randomly generated tokens to guarantee the privacy of a users search pattern. Note that, randomizing token generation algorithm only contributes to defend outside adversaries of the cloud but not inner adversaries (e.g., cloud administrators), because the entry of index touched in each search process discloses the search pattern as well.
- **Public Verifiability:** Since the cloud may not honestly perform keyword search and return the result. The data user needs to verify the correctness of both the search process and the returned result. Moreover, in the storage-as-a-service settings, other users expect to be convinced of the correctness of cloud service [1], [2], while keeping the queried keyword and the stored data private. Therefore, privacy-preserving public verification is significant in this scenario.
- **Efficiency:** Search efficiency and verification efficiency, are the two important indicators to measure a public verifiable keyword search scheme. Search efficiency determines the response time and quality of cloud service. Verification efficiency meaning that users should be able to check the proof by requiring significantly fewer resources than those that are needed to compute in the cloud. By reducing the dimension of vectors used in our constructions, our 3-PVSAE scheme is more efficient than existing verifiable keyword search schemes in the sense of search and verification.

D. Our Contributions

In this paper, we propose a novel service oriented framework for verifiable searchable asymmetric encryption. To the best of our knowledge, our solutions are the first to achieve both public verifiability and information-theoretical security for searchable asymmetric encryption.

- First, we describe the approaches of constructing PVSAE for inverted index-based encrypted data. We develop a framework and its security definitions of PVSAE in terms of IND-CKA security and search pattern privacy.
- Second, we propose a concrete construction for PVSAE with ℓ-dimensional vectors, called ℓ-PVSAE. ℓ-PVSAE can provide public verification and offers stronger notions of security: statistical IND-CKA security and statistical

search pattern privacy. Then, we put forward a light-weight version of the PVSAE scheme with 3-dimensional vectors, called 3-PVSAE. 3-PVSAE scheme maintains the strong security properties and is highly efficient and faster for querying over encrypted data in the cloud environment, compared with existing verifiable attribute-based keyword search schemes.
- Finally, we analytically and experimentally show that ℓ-PVSAE and 3-PVSAE schemes are not only offering strong security but also practical and deployable for querying over encrypted data in the cloud environment.

II. Previous Work

In this section we briefly review the relevant techniques including public-key encryption with keyword search and verifiable searchable encryption.

A. Public-Key Encryption with Keyword Search

Boneh *et al.* [5] first presented the framework of public-key encryption with keyword search (PEKS). They formally defined its security and showed a general transformation from anonymous identity-based encryption (AIBE) to PEKS. Abdalla *et al.* [6] defined consistency in PEKS and gave an improved transformation from AIBE to PEKS. The conventional security for PEKS, called IND-CKA security, requires that the searchable index does not leak any information about the keyword. This security, however, gives no guarantee about leakage of the keyword from the search token.

The privacy of keywords from search tokens has been discussed in the symmetric-key setting [9] and the interactive public-key setting [10]. Shen *et al.* [9] presented a security notion, *predicate privacy*, to ensure that tokens reveal no information about the encoded query predicate. Camenisch *et al.* [10] presented an extended notion of PEKS, called public-key encryption with oblivious keyword search (PEOKS), in which a user can obtain the search token from the secret key holder without revealing the keyword. Motivated by the need for providing predicate privacy in public-key searchable encryption, Blundo *et al.* [7] presented a predicate encryption scheme with partial public key, and define the notion of *token security* to ensure the privacy of attributes from a token. Subsequently, Boneh *et al.* [11] put forward a new notion, *function privacy*, in function encryption. Their notion asks that decryption keys reveal essentially no information on their corresponding identities and thus yields the first public-key searchable encryption schemes that are provably keyword private. However, their schemes are function private only for identities that are high unpredictable (with high min-entropy).

Nishioka [8] first formalized token privacy within the framework of PEKS, which is called *search pattern privacy*, that is, when two tokens are given, it is hard to guess whether they correspond to the same keyword. Then, Arriaga *et al*. [12] proposed the security notion of *key unlinkability* for IBE, which leads to the guarantee of search token privacy in PEKS. However, their schemes are all constructed over the groups of composite order, which require larger parameter size and much longer time to complete a pairing than over a prime-order elliptic curve [13].

B. Verifiable Searchable Encryption

Considering the cloud may not honestly execute the search operation and return the correct result, Kurosawa *et al.* first proposed a verifiable searchable symmetric encryption (VSSE) scheme [14], in which they use the message authenticate code (MAC) to protect the integrity of the searching results. Inspired with their work, several works on VSSE schemes [15]–[19] are put forward to enhance VSSE in recent years. However, verifiable searchable encryption in public-key setting is rarely studied.

Recently, Zheng *et al.* [3] proposed the first verifiable attribute-based keyword search (VABKS) scheme. Then, Liu *et al.* [4] gave another construction based on key policy attribute-based keyword search (KP-ABKS). However, these two schemes only provide private verification. While many cloud applications require to allow their users to public verify whether the cloud has faithfully executed search operations or not, so that users can reasonably assess the quality of service [1], [2].

III. Preliminaries

In this part we simply explain the cryptographic tools used in our constructions.

A. Notations

For an integer $n \in \mathbb{N}$ we denote by $[n]$ the set $\{1, \cdots, n\}$, and by U_n the uniform distribution over the set $\{0,1\}^n$. For a finite set S we denote by $x \leftarrow S$ the process of sampling a value x according to the distribution over S, and by $x \xleftarrow{R} S$ the random choose process of a value x from the uniform distribution over S. We denote by $\vec{x}$ a vector $(x_1, \cdots, x_{|\vec{x}|})$, where $|\vec{x}|$ and $x_i (1 \le i \le |\vec{x}|)$ respectively denote the number of elements and the i-th element of vector $\vec{x}$. Let $\vec{x}^{\mathsf{T}}$ be the transposition of vector $\vec{x}$. We overload the notation g^M to matrices: we let $g^M \in \mathbb{G}^{\ell \times \ell}$ denote the matrix defined as $(g^M)_{i,j} = g^{M_{i,j}}$, where $1 \le i, j \le \ell$. Let M^{T} be the transposition of matrix M. Let M^{-1} be the inverse matrix of M. Let $g^{\vec{x} \cdot M}$ denote the product defined as $(g^{x_1 M_{1,1} + \cdots + x_\ell M_{\ell,1}}, \cdots, g^{x_1 M_{1,\ell} + \cdots + x_\ell M_{\ell,\ell}})$ and $g^{M \cdot \vec{x}^{\mathsf{T}}}$ denote the product defined as $(g^{M_{1,1}x_1 + \cdots + M_{1,\ell}x_\ell}, \cdots, g^{M_{\ell,1}x_1 + \cdots + M_{\ell,\ell}x_\ell})$. Let $(g^{\vec{x}})^{\vec{y}} = (g^{\vec{x}^{\mathsf{T}}})^{\vec{y}} = (g^{\vec{x}})^{\vec{y}^{\mathsf{T}}}$ denote the exponentiation defined as $\prod_{i=1}^{\ell} (g^{x_i})^{y_i}$. The scheme is parameterized by the security parameter λ. A function ϵ is *negligible* if for every polynomial $p(\cdot)$ there exist an N such that for all integers $n > N$ it holds that $\epsilon(n) < \frac{1}{p(n)}$.

B. Collision-Resistant Hash Function

A function H is *collision resistant* if it is infeasible for any probabilistic polynomial-time (PPT) algorithm to find a collision in H.

Definition 1: A hash function $\Pi' = (\mathsf{Gen}, H)$ is *collision resistant* if for all PPT adversaries $\mathcal{A}$ there exists a negligible function ϵ such that

$$\Pr[\mathsf{Gen}(1^\lambda) \to H, \mathcal{A}(H) \to x, x' : x' \neq x, H(x') = H(x)] \leq \epsilon(\lambda)$$

C. Bilinear Map

Let GroupGen be a PPT algorithm that takes as input a security parameter 1^λ, and outputs $(\mathbb{G}, \mathbb{G}_T, q, g, e)$, where $\mathbb{G}$ and $\mathbb{G}_T$ are groups of prime order q, $\mathbb{G}$ is generated by $g \in \mathbb{G}$, and q is a λ-bit prime number. Let $\mathbb{G}_T$ be a (different) group of order q. A bilinear map $e : \mathbb{G} \times \mathbb{G} \to \mathbb{G}_T$ has the following properties. (1) *Bilinearity:* for all $g_1, g_2 \in \mathbb{G}$, $a, b \in \mathbb{Z}$ it holds that $e(g_1^a, g_2^b) = e(g_1, g_2)^{ab}$; (2) *Non-degeneracy:* $e(g, g) \neq 1$; (3) It follows that $g_T \stackrel{\mathsf{def}}{=} e(g, g)$ generates $\mathbb{G}_T$.

Let $\vec{x} = \{x_1, \cdots, x_\ell\}$ and $\vec{y} = \{y_1, \cdots, y_\ell\}$ be two ℓ-dimensional vectors. The bilinear map of $e(g_1^{\vec{x}}, g_2^{\vec{y}^\mathsf{T}})$ is computed as follows:

$$\begin{aligned} e(g_1^{\vec{x}}, g_2^{\vec{y}^\mathsf{T}}) &= e(g_1^{x_1}, g_2^{y_1}) \cdot e(g_1^{x_2}, g_2^{y_2}) \cdots e(g_1^{x_\ell}, g_2^{y_\ell}) \\ &= e(g_1, g_2)^{x_1 y_1 + \cdots + x_\ell y_\ell} \end{aligned}$$

D. The Discrete Logarithm Assumption

The discrete logarithm assumption is that the assumption that the discrete logarithm problem is hard for GroupGen, where this is defined as follows:

Definition 2: The *discrete logarithm problem is hard for* GroupGen if the following is negligible for all PPT adversaries $\mathcal{A}$:

$$\Pr[(\mathbb{G}, q, g) \leftarrow \mathsf{GroupGen}(1^\lambda); h \leftarrow \mathbb{G}; x \leftarrow \mathcal{A}(\mathbb{G}, q, g, h) : g^x = h].$$

E. Existentially Unforgeable Signature Scheme

A signature scheme is a tuple of three PPT algorithms $(\mathsf{SGen}, \mathsf{Sign}, \mathsf{SVerify})$. For an adversary $\mathcal{A}$, we define the advantage function $\mathsf{Adv}_{\mathsf{EU}}(\lambda)$ to be:

$$\Pr\left[\begin{array}{c} \mathsf{SVerify}_{vk_s}(m', \sigma') = 1 \\ \text{and } m' \text{ is not queried} \end{array} : \begin{array}{c} (vk_s, sk_s) \leftarrow \mathsf{SGen}(1^\lambda) \\ (m', \sigma') \leftarrow \mathcal{A}^{\mathsf{Sign}_{sk_s}(\cdot)}(vk_s) \end{array}\right]$$

A signature scheme is *existentially unforgeable under an adaptive chosen-message attack* if for all PPT adversary $\mathcal{A}$, the advantage $\mathsf{Adv}_{\mathsf{EU}}(\lambda)$ is a negligible function in λ.

IV. Framework and Definitions

We introduce the definition and security notions of PVSAE in this section.

A. The Framework of PVSAE

To outsource data, data owner associates a set of keywords $W = \{w_1, \cdots, w_n\}$ with each file and creates an index $I = \{I_{w_1}, \cdots, I_{w_n}\}$ with data user's public key PK, then sends the encrypted index to the cloud along with the encrypted data. The data user knowing the secret key MK may query the database for a query function $F(w)$. The cloud computes $F(w)$ on the index to return a search result $\theta_{F(w)}$. Any party holding the verification key $VK_{F,w}$ can verify the correctness of the result. In this paper, we focus on the keyword exact match search, that is, $F(w) = w$, so we directly use w for the query function $F(w)$ in the following parts.

Let $\Pi = (\mathsf{Setup}, \mathsf{Enc}, \mathsf{TokenGen}, \mathsf{Query}, \mathsf{Verify})$ be a PVSAE scheme over the set of keywords $\mathcal{W}_\lambda$ consists of the following PPT algorithms as follows:

- $\mathsf{Setup}(1^\lambda) \to (PP, PK, MK)$: On input the security parameter 1^λ, output public parameters PP and a key pair (PK, MK).
- $\mathsf{Enc}(W, PK, PP) \to I$: On input the keywords set $W = \{w_1, \cdots, w_n\} \subseteq \mathcal{W}$, the public key PK and public parameters PP, output a searchable encrypted index I.
- $\mathsf{TokenGen}(w, MK, PP) \to (TK_w, VK_w)$: On input a search keyword w, the secret key MK and public parameters PP, output a search token TK_w and a verification key VK_w.
- $\mathsf{Query}(TK_w, I, PP) \to \theta_{w_i} \cup \perp$: On input a search token TK_w, the searchable encrypted index I and public parameters PP, output the search result θ_{w_i} or $\perp$.
- $\mathsf{Verify}(\theta_{w_i}, VK_w, PP) \to 1 \cup 0$: On input the returned result θ_{w_i}, a verification key VK_w and public parameters PP, output 1 or 0.

Correctness. The correctness of a PVSAE scheme can be defined as follows:

Definition 3 (Correctness): For all λ, all $w \in W$, letting $(PP, PK, MK) \leftarrow \mathsf{Setup}(1^\lambda)$, if $I \leftarrow \mathsf{Enc}(W, PK, PP)$ and $(TK_w, VK_w) \leftarrow \mathsf{TokenGen}(w, MK, PP)$, then $\mathsf{Query}(TK_w, I, PP) = \theta_{w_i}$ and $\mathsf{Verify}(\theta_{w_i}, VK_w, PP) = 1$.

Security against Chosen-Keyword Attack. The basic notion of security against chosen-keyword attack (CKA) asks that without seeing the related search token $\mathcal{A}$ cannot learn any information about keywords from the encrypted index.

Definition 4 (IND-CKA security): A PVSAE scheme $\Pi = (\mathsf{Setup}, \mathsf{Enc}, \mathsf{TokenGen}, \mathsf{Query}, \mathsf{Verify})$ is *IND-CKA secure* if for any PPT adversary $\mathcal{A}$, there exists a negligible function $\epsilon(\lambda)$ such that

$$\mathsf{Adv}^{\mathrm{cka}}_{\Pi,\mathcal{A}}(\lambda) \stackrel{\mathsf{def}}{=} \left|\Pr\left[\mathsf{Expt}^{(0)}_{\mathrm{cka},\Pi,\mathcal{A}}(\lambda) = 1\right] - \Pr\left[\mathsf{Expt}^{(1)}_{\mathrm{cka},\Pi,\mathcal{A}}(\lambda) = 1\right]\right| \leq \epsilon(\lambda),$$

the experiment $\mathsf{Expt}^{(b)}_{\mathrm{cka},\Pi,\mathcal{A}}(\lambda)$ is defined as follows.

$\mathsf{Expt}^{(b)}_{\mathrm{cka},\Pi,\mathcal{A}}(\lambda)$
1: $(PP, PK, MK) \leftarrow \mathsf{Setup}(1^\lambda)$.
2: $(W_0^*, W_1^*, state) \leftarrow \mathcal{A}(1^\lambda, PK, PP)$, where $|W_0^*| = |W_1^*| = n$.
3: $I_{W_b^*} \leftarrow \mathsf{Enc}(W_b^*, PK, PP)$, where $b \xleftarrow{R} \{0, 1\}$.
4: $b' \leftarrow \mathcal{A}^{\mathsf{TokenGen}(\cdot, MK, PP)}(I_{W_b^*}, state)$, where $b' \in \{0, 1\}$.
5: For all token generation queries w_i,
if $\mathsf{Query}(TK_{w_i}, I_{W_0^*}, PP) = \mathsf{Query}(TK_{w_i}, I_{W_1^*}, PP)$
then output b', otherwise output $\perp$.

Remark. In addition, such a scheme is *statistically* IND-CKA secure if the above holds for any *computationally-unbounded* adversary.

Search Pattern Privacy. Given a search token, $\mathcal{A}$ should not be able to infer any information about the queried keyword.

Definition 5 (search pattern security): A PVSAE scheme $\Pi = (\mathsf{Setup}, \mathsf{Enc}, \mathsf{TokenGen}, \mathsf{Query}, \mathsf{Verify})$ is *search pattern secure* if for any PPT adversary $\mathcal{A}$, there exists a negligible function $\epsilon(\lambda)$ such that

$$\mathsf{Adv}^{\mathrm{ssp}}_{\Pi,\mathcal{A}}(\lambda) \stackrel{\mathsf{def}}{=} \left|\Pr\left[\mathsf{Expt}^{(0)}_{\mathrm{spp},\Pi,\mathcal{A}}(\lambda) = 1\right] - \left[\mathsf{Expt}^{(1)}_{\mathrm{spp},\Pi,\mathcal{A}}(\lambda) = 1\right]\right| \leq \epsilon(\lambda),$$

the experiment $\mathsf{Expt}^{(b)}_{\mathrm{spp},\Pi,\mathcal{A}}(\lambda)$ is defined as follows.

$\mathsf{Expt}^{(b)}_{\mathrm{spp},\Pi,\mathcal{A}}(\lambda)$
1: $(w_0^*, w_1^*) \leftarrow W_\lambda$.
2: $(PP, PK, MK) \leftarrow \mathsf{Setup}(1^\lambda)$.
3: $(TK_{w_0^*}, VK_{w_0^*}) \leftarrow \mathsf{TokenGen}(w_0^*, MK, PP)$.
4: $(TK_{w_b^*}, VK_{w_b^*}) \leftarrow \mathsf{TokenGen}(w_b^*, MK, PP)$,where $b \xleftarrow{R} \{0,1\}$.
5: $b' \leftarrow \mathcal{A}^{\mathsf{TokenGen}(\cdot,MK,PP)}(TK_{w_b^*}, VK_{w_b^*}, state)$, output b'.

Remark. Like the Definition 4, such a scheme is *statistically* search pattern private if the above holds for any *computationally-unbounded* adversary.

Public Verifiability. $\mathcal{A}$ should not be able to return an incorrect result without being detected by the verifier.

Definition 6 (public verifiability): A PVSAE scheme $\Pi = (\mathsf{Setup}, \mathsf{Enc}, \mathsf{TokenGen}, \mathsf{Query}, \mathsf{Verify})$ is *public verifiable* if for any PPT adversary $\mathcal{A}$, there exists a negligible function $\epsilon(\lambda)$ such that

$$\mathsf{Adv}^{\mathrm{pubverif}}_{\Pi,\mathcal{A}}(\lambda) \stackrel{\mathsf{def}}{=} \left|\Pr\left[\mathsf{Expt}_{\mathrm{pubverif},\Pi,\mathcal{A}}(\lambda)=1\right]\right| \leq \epsilon(\lambda),$$

the experiment $\mathsf{Expt}_{\mathrm{pubverif},\Pi,\mathcal{A}}(\lambda)$ is defined as follows.

$\mathsf{Expt}_{\mathrm{pubverif},\Pi,\mathcal{A}}(\lambda)$
1: $(PP, PK, MK) \leftarrow \mathsf{Setup}(1^\lambda)$.
2: $w \leftarrow \mathcal{A}(1^\lambda, PK, PP)$.
3: $(TK_w, VK_w) \leftarrow \mathsf{TokenGen}(w, MK, PP)$.
4: $\theta_{\hat{w}} \leftarrow \mathcal{A}^{\mathsf{TokenGen}(\cdot,MK,PP)}(TK_w, VK_w, PP)$,
5: $1 \leftarrow \mathsf{Verify}(\theta_{\hat{w}}, VK_w, PP)$.
if $\hat{w} \neq \perp$ and $\hat{w} \neq w$ output 1, otherwise output 0.

V. The Constructions of PVSAE

In this section, we first construct an ℓ-PVSAE scheme based on ℓ-dimensional vectors. Then we show a light-weight and efficient version called 3-PVSAE.

A. The ℓ-PVSAE Scheme

Let $\Pi^\ell = (\mathsf{Setup}, \mathsf{Enc}, \mathsf{TokenGen}, \mathsf{Query}, \mathsf{Verify})$ be our ℓ-PVSAE scheme. The detail construction of each algorithm is as follows.

- $\mathsf{Setup}(1^\lambda) \rightarrow (PP, PK, MK)$. The algorithm samples $(\mathbb{G}, \mathbb{G}_T, q, g, e) \leftarrow \mathsf{GroupGen}(1^\lambda)$, where $\mathbb{G}$ is a cyclic group of prime order q, an ℓ-bit vector $\vec{S} \xleftarrow{R} \{0,1\}^\ell$, two full rank matrices $M', M'' \xleftarrow{R} \mathbb{Z}_q^{\ell\times\ell}$, and a collision-resistant hash function $H : \{0,1\}^* \rightarrow \mathbb{Z}_q$. It then sets $PP = (\mathbb{G}, \mathbb{G}_T, q, g, e, H, \vec{S})$ as public parameters, $PK = (g^{M'^\mathsf{T}}, g^{M''^\mathsf{T}})$ as the public key and $MK = (M'^\mathsf{T}, M''^\mathsf{T})$ as the secret key. Note that M', M'' are invertible with all but a negligible probability.
- $\mathsf{Enc}(W, PK, PP) \rightarrow ((vk_s, sk_s), I, \sigma)$. With a collection of keywords $W = \{w_1, \cdots, w_n\}$, the data owner encrypts it with the public key PK and public parameters PP, and outputs a signature and verification key-pair (vk_s, sk_s), the encrypted index $I = \{I_{w_1}, \cdots, I_{w_n}\}$ and a signature set $\sigma = \{\sigma_{w_1}, \cdots, \sigma_{w_n}\}$, where each element I_{w_i} is generated as follows.
  1) Generate an ℓ-dimensional vector $\vec{p_i} = (p_{i,1}, \cdots, p_{i,\ell}) = (H(w_i), r_{i,1}, \cdots, r_{i,\ell-2}, 1)$, where $r_{i,j\in[1,\ell-2]} \xleftarrow{R} \mathbb{Z}_q$.
  2) Split $\vec{p_i}$ to two vectors $\vec{p_i}'$ and $\vec{p_i}''$ with the splitting indicator $\vec{S}$ as follows. For $1 \leq j \leq \ell$,
  $$\begin{cases} p'_{i,j} = p''_{i,j} = p_{i,j} \pmod q, & S_j = 0 \\ p'_{i,j} + p''_{i,j} = p_{i,j} \pmod q, & S_j = 1 \end{cases} \quad (1)$$
  For the second case, first choose $p'_{i,j} \xleftarrow{R} \mathbb{Z}_q$, then set $p''_{i,j} = p_{i,j} - p'_{i,j} \pmod q$.
  3) Encrypt these two vectors as $I_{w_i} = (I_{w_i,1}, I_{w_i,2}) = (g^{M'^\mathsf{T}\vec{p_i}'^\mathsf{T}}, g^{M''^\mathsf{T}\vec{p_i}''^\mathsf{T}})$.

  Then the data owner runs $\mathsf{SGen}(PP) \rightarrow (vk_s, sk_s)$ and computes $\mathsf{Sign}_{sk_s}(I_{w_i} || \mathsf{E}(D_{w_i})) \rightarrow \sigma_{w_i}$ to generate a signature for each item of index. Finally, the data owner sends I, σ to the cloud and makes vk_s public.
- $\mathsf{TokenGen}(w, MK, PP) \rightarrow (TK_w, VK_w)$. With the queried keyword w, the secret key MK and public parameters PP, the data user creates search token TK_w and verification key VK_w for keyword w as follows.
  1) Generate an ℓ-dimensional vector $\vec{t} = (t_1, \cdots, t_\ell) = (a, 0, \cdots, 0, c)$, where $a, c \xleftarrow{R} \mathbb{Z}_q$.
  2) Split $\vec{t}$ to two vectors $\vec{t}'$ and $\vec{t}''$ with the splitting indicator $\vec{S}$ as follows. For $i = 1$ to ℓ,
  $$\begin{cases} t'_i = t''_i = t_i \pmod q, & S_i = 1 \\ t'_i + t''_i = t_i \pmod q, & S_i = 0 \end{cases} \quad (2)$$
  For the second case, first choose $t'_i \xleftarrow{R} \mathbb{Z}_q$, then set $t''_i = t_i - t'_i \pmod q$.
  3) Compute $\beta = H(w)$ and output $TK_w = (TK_{w,1}, TK_{w,2}, TK_{w,3}) = \left(g^{\vec{t}'M'^{-1}\mathsf{T}}, g^{\vec{t}''M''^{-1}\mathsf{T}}, g^{a\beta+c}\right)$ and $VK_w = (VK_{w,1}, VK_{w,2}, VK_{w,3}) = \left(\vec{t}'M'^{-1}\mathsf{T}, \vec{t}''M''^{-1}\mathsf{T}, g^{a\beta+c}\right)$ for w.
- $\mathsf{Query}(TK_w, I, PP) \rightarrow \theta_{w_i} \cup \perp$. With the search token TK_w, encrypted index I and public parameters PP, the cloud tests TK_w with each item of encrypted index by checking whether the following equation holds. If Equation (3) holds it returns the search result $\theta_{w_i} = (I_{w_i}, \mathsf{E}(D_{w_i}), \sigma_{w_i})$; otherwise it returns $\perp$.
$$e(TK_{w,1}, I_{w_i,1}) \cdot e(TK_{w,2}, I_{w_i,2}) \stackrel{?}{=} e(TK_{w,3}, g) \quad (3)$$
- $\mathsf{Verify}(\theta_{w_i}, VK_w, vk_s, PP) \rightarrow 1 \cup 0$. With the θ_{w_i}, verification key VK_w, signature verification key vk_s and public parameters PP, anyone can verifies whether the Equation (4) holds. If Equation (4) holds and $\mathsf{SVerify}_{vk_s}(I_{w_i} || \mathsf{E}(D_{w_i}), \sigma_{w_i}) = 1$ it outputs 1; otherwise it outputs 0.
$$I_{w_i,1}^{VK_{w,1}} \cdot I_{w_i,2}^{VK_{w,2}} \stackrel{?}{=} VK_{w,3} \quad (4)$$

Correctness. We state the following theorem about the correctness of our ℓ-PVSAE construction.

Theorem 1: If each algorithm is performed correctly, our ℓ-PVSAE scheme Π^{ℓ} satisfies the correctness as defined in Definition 3.

Proof: For the query correctness, with a search token TK_w and an encrypted index I, the left side of Equation (3) can be computed as follows.

$$\begin{aligned} & e(TK_{w,1}, I_{w_i,1}) \cdot e(TK_{w,2}, I_{w_i,2}) \\ = \; & e(g^{\vec{t}' M'^{-1\mathsf{T}}}, g^{M'^{\mathsf{T}} \vec{p_i}'^{\mathsf{T}}}) \cdot e(g^{\vec{t}'' M'^{-1\mathsf{T}}}, g^{M''^{\mathsf{T}} \vec{p_i}''^{\mathsf{T}}}) \\ = \; & e(g,g)^{\vec{t}' \vec{p_i}'^{\mathsf{T}} + \vec{t}'' \vec{p_i}''^{\mathsf{T}}} \\ = \; & e(TK_{w,3}, g) \end{aligned}$$

For the verification correctness, with a verification key VK_w and a returned search result $\theta_{w_i} = (I_{w_i}, \mathsf{E}(D_{w_i}), \sigma_{w_i})$, the left side of Equation (4) can be computed as follows.

$$\begin{aligned} & I_{w_i,1}^{VK_{w,1}} \cdot I_{w_i,2}^{VK_{w,2}} \\ = \; & \left(g^{M'^{\mathsf{T}} \vec{p_i}'^{\mathsf{T}}}\right)^{\vec{t}' M'^{-1\mathsf{T}}} \cdot \left(g^{M''^{\mathsf{T}} \vec{p_i}''^{\mathsf{T}}}\right)^{\vec{t}'' M''^{-1\mathsf{T}}} \\ = \; & g^{\vec{t}' (M' M'^{-1})^{\mathsf{T}} \vec{p_i}'^{\mathsf{T}} + \vec{t}'' (M'' M''^{-1})^{\mathsf{T}} \vec{p_i}''^{\mathsf{T}}} \\ = \; & VK_{w,3} \end{aligned}$$

Besides, $\mathsf{SVerify}_{vk_s}(I_{w_i} || \mathsf{E}(D_{w_i}), \sigma_{w_i}) = 1$ guarantees the binding and unforgeability of I_{w_i} and the corresponding encrypted data $\mathsf{E}(D_{w_i})$. ∎

B. The 3-PVSAE scheme

To improve the efficiency of ℓ-PVSAE scheme, we propose a light-weight version of ℓ-PVSAE in which we set $\ell = 3$.

Let $\Pi^3 = (\mathsf{Setup}, \mathsf{Enc}, \mathsf{TokenGen}, \mathsf{Query}, \mathsf{Verify})$ be our 3-PVSAE scheme. The Setup algorithm runs the same as the Setup algorithm of ℓ-PVSAE except that it generates 3-bit vector $\vec{S} \xleftarrow{R} \{0,1\}^3$, two full rank matrices $M', M'' \xleftarrow{R} \mathbb{Z}_q^{3\times 3}$. The Enc algorithm first generates a 3-dimensional vector $\vec{p} = (p_1, p_2, p_3) = (H(w_i), r, 1)$, where $r \xleftarrow{R} \mathbb{Z}_q$. Then it splits and encrypts $\vec{p}$ as the same as the process of Enc algorithm of ℓ-PVSAE. The $\mathsf{TokenGen}$ algorithm acts identical to the $\mathsf{TokenGen}$ algorithm of ℓ-PVSAE except to generate a 3-dimensional vector $\vec{t} = (t_1, t_2, t_3) = (a, 0, c)$, where $a, c \xleftarrow{R} \mathbb{Z}_q$. Query and Verify algorithms are act exactly the same as Query and Verify of ℓ-PVSAE.

Obviously, the correctness of 3-PVSAE can be maintained, since Equation (3) and Equation (4) are still hold if each algorithm is performed correctly.

VI. Analysis of Our PVSAE Schemes

We evaluate the proposed schemes by analyzing their security and efficiency in this section. First, we show how the proposed schemes meet security guarantees defined in section IV-A. Then we give the efficiency analysis and comparisons with existing verifiable public-key searchable schemes.

A. Security

We first prove that our PVSAE schemes achieve strong security notion *statistically IND-CKA security* and *statistically search pattern private*. Then we prove that our PVSAE schemes can provide public verification for returned encrypted index and files.

1) Security against Chosen-Keyword Attack:

Theorem 2: When $\ell \geq 3$ our ℓ-PVSAE scheme is statistically IND-CKA secure based on the discrete logarithm assumption.

Proof: Let $\mathcal{A}$ be a computationally unbounded adversary that makes a polynomial number of queries to the token generation oracle $\mathsf{TokenGen}$. We prove that the distribution of $\mathcal{A}$'s view in the experiment $\mathsf{Expt}^{(0)}_{\mathrm{cka},\Pi,\mathcal{A}}(\lambda)$ is statistically close to the distribution of $\mathcal{A}$'s view in the experiment $\mathsf{Expt}^{(1)}_{\mathrm{cka},\Pi,\mathcal{A}}(\lambda)$ (we refer the reader to Definition 4 for the descriptions of these experiments). We denote these two distributions by $\mathsf{View}^{(0)}_{\mathrm{cka}}$ and $\mathsf{View}^{(1)}_{\mathrm{cka}}$, respectively.

Denote by $W_0^* = \{w_{0,1}^*, \cdots, w_{0,n}^*\}$ and $W_1^* = \{w_{1,1}^*, \cdots, w_{1,n}^*\}$ the two challenge keyword sets. Having already fixed public parameters $PP = (\mathbb{G}, \mathbb{G}_T, q, g, e, H, \vec{S})$ and public key $PK = (g^{M'^{\mathsf{T}}}, g^{M''^{\mathsf{T}}})$, we can assume that $\mathsf{View}^{(b)} = ((I_{w_{1,1}^*}, I_{w_{1,2}^*}), \cdots, (I_{w_{n,1}^*}, I_{w_{n,2}^*})) = ((g^{M'^{\mathsf{T}} \vec{p_1}'^{\mathsf{T}}}, g^{M''^{\mathsf{T}} \vec{p_1}''^{\mathsf{T}}}), \cdots, (g^{M'^{\mathsf{T}} \vec{p_n}'^{\mathsf{T}}}, g^{M''^{\mathsf{T}} \vec{p_n}''^{\mathsf{T}}}))$. Observe that M' and M'' are uniformly chosen from $\mathbb{Z}_q^{\ell\times\ell}$, thus for every $i \in [n]$, the distributions of $g^{M'^{\mathsf{T}} \vec{p_i}'^{\mathsf{T}}}$ and $g^{M''^{\mathsf{T}} \vec{p_i}''^{\mathsf{T}}}$ are uniform as long as $\vec{p_i}', \vec{p_i}'' \neq \vec{0}$. The probability that $\vec{p_i}' = \vec{0}$ is $\Pr[\vec{p_i}' = \vec{0}] \leq \frac{1}{(2q)^{\ell}} \left(\frac{q-1}{q}\right)^{\ell-i-1}$. When $i = \ell - 1$ the probability is maximized, that is $\Pr[\vec{p_i}' = \vec{0}] \leq \frac{1}{(2q)^{\ell}}$. The probability is negligible when q is a large prime.

In summary, the statistical distance between $\mathsf{View}^{(0)}_{\mathrm{cka}}$ and $\mathsf{View}^{(1)}_{\mathrm{cka}}$ is negligible in λ. ∎

2) Search Pattern Privacy:

Theorem 3: when $\ell \geq 3$ our ℓ-PVSAE scheme is statistically search pattern private based on the discrete logarithm assumption.

Proof: Let $\mathcal{A}$ be a computationally unbounded adversary that makes a polynomial number of queries to the token generation oracle $\mathsf{TokenGen}$. We prove that the distribution of $\mathcal{A}$'s view in the experiment $\mathsf{Expt}^{(0)}_{\mathrm{spp},\Pi,\mathcal{A}}(\lambda)$ is statistically close to the distribution of $\mathcal{A}$'s view in the experiment $\mathsf{Expt}^{(1)}_{\mathrm{spp},\Pi,\mathcal{A}}(\lambda)$ (we refer the reader to Definition 5 for the descriptions of these experiments). We denote these two distributions by $\mathsf{View}^{(0)}_{\mathrm{spp}}$ and $\mathsf{View}^{(1)}_{\mathrm{spp}}$, respectively.

Denote by w_0^* and w_1^* the two challenge keywords. Having already fixed hash function H and $MK = (M'^{\mathsf{T}}, M''^{\mathsf{T}})$, we can assume that $\mathsf{View}^{(b)}_{spp} = (TK_{w_b^*}, VK_{w_b^*})$, for $b \in \{0,1\}$, where $w_b^* = w_0^*$ for $b = 0$, $w_b^* = w_1^*$ for $b = 1$, and $a, c \xleftarrow{R} \mathbb{Z}_q$.

The distribution of $(VK_{w_b^*,1}, VK_{w_b^*,2})$ are uniform as long as $\vec{t}', \vec{t}'' \neq \vec{0}$ where $\vec{t}'$ and $\vec{t}''$ are split from $\vec{t} = (a, 0, \cdots, 0, c)$, since M' and M'' are uniformly chosen from $\mathbb{Z}_q^{\ell\times\ell}$. This implies the distribution of $(TK_{w_b^*,1}, TK_{w_b^*,2})$ are also uniform as long as $\vec{t}', \vec{t}'' \neq \vec{0}$. The probability that $\vec{t}' = \vec{0}$ is $\Pr[\vec{t}' = \vec{0}] \leq \frac{1}{2^{\ell} q^2}$. The probability is negligible when q is a large prime.

In addition, we can directly infer that the distribution of $TK_{w_b^*,3}$ and $VK_{w_b^*,3}$ are statistically-close to uniform as $a, c \xleftarrow{R} \mathbb{Z}_q$. This implies the statistical distance between

TABLE I
COMPARISON OF VERIFIABLE PUBLIC-KEY SEARCHABLE ENCRYPTION SCHEMES

Scheme	Security	Search Complexity	Verification Complexity	Public Verification
VABKS [3]	CKA, SPP	$(2S+2)\mathsf{P}+S\mathsf{E_T}$	$2\mathsf{SVerify}+(2S+2)\mathsf{P}+S\mathsf{E_T}$	No
LWMN [4]	CKA, SPP	$(4+S)\mathsf{P}$	$2\mathsf{SVerify}+2\mathsf{E}$	No
ℓ-PVSAE	sCKA, sSPP	$(2\ell+1)\mathsf{P}$	$2\ell\mathsf{E}+\mathsf{SVerify}$	Yes
3-PVSAE	sCKA, sSPP	$7\mathsf{P}$	$6\mathsf{E}+\mathsf{SVerify}$	Yes

S denotes the number of a data user's attributes. P denotes the pairing operation. E denotes the exponentiation operation in $\mathbb{G}$. $\mathsf{E_T}$ denotes the exponentiation operation in $\mathbb{G}_T$. SVerify denotes the verification operation of an existentially unforgeable signature scheme. We ignore multiplication and hash operations because they are much more efficient than the above operations. We write sCKA for statistically chosen keyword attack security and sSPP for statistically search pattern privacy.

$\mathsf{View}_{\mathrm{spp}}^{(0)}$ and $\mathsf{View}_{\mathrm{spp}}^{(1)}$ is negligible in λ. ■

3) Public Verifiability:

Theorem 4: If $H : \{0,1\}^* \rightarrow \mathbb{Z}_q$ is a collision-resistant hash function, our PVSAE schemes achieve public verifiability according to Definition 6.

Proof: To prove our PVSAE schemes achieve public verifiability we can direct discuss the probability that the adversary $\mathcal{A}$ successfully create $\theta_{\hat{w}}$ and $\mathsf{Verify}(\theta_{\hat{w}}, VK_w, PP) = 1$, that is Equation (4) holds. According to Equation (4), we actually need to compute the probability $\Pr[aH(\hat{w})+c = aH(w)+c]$. Since $H : \{0,1\}^* \rightarrow \mathbb{Z}_q$ is a collision-resistant hash function, according to Definition 1, the probability that all PPT adversaries output $w, \hat{w} \in \{0,1\}^\lambda$ such that $\hat{w} \neq w$ and $H(\hat{w}) = H(w)$ is negligible.

Besides, to verify the Equation (4), it also needs to satisfy $\mathsf{SVerify}_{vk_s}(I_w||E(D_w), \sigma_w) = 1$. Since the signature scheme adopted in PVSAE schemes is existentially unforgeable, the advantage that all PPT adversaries successfully forge a signature σ'_w is negligible according to the definition in section III-E. In summary, we have following result.

$$\mathsf{Adv}_{\Pi,\mathcal{A}}^{\mathrm{pubverif}}(\lambda) = \left|\Pr\left[\mathsf{Expt}_{\mathrm{pubverif},\Pi,\mathcal{A}}^{(\lambda)}(\lambda) = 1\right]\right| \leq \epsilon(\lambda)$$

■

B. Efficiency

We evaluate the efficiency of the PVSAE schemes in terms of search and verification and compare with existing schemes in Table I. Note that the search complexity in Table I is for evaluating with a search token and one item of encrypted index. The total search complexity for whole search process is equal to the values in the table multiplied by n, where n is the total number of keywords. We observe that our 3-PVSAE scheme is much more efficient than VAVKS [3] and LWMN [4].

VII. EXPERIMENTAL EVALUATION

A. Dataset and Experimental Setup

For our experiments, we built Email datasets indexed by different number of keywords (i.e., $n = 1000, 2000, \cdots, 10000$). We encrypted the datasets with OAEP+ [20] and encrypted the indexes with ℓ-PVSAE and 3-PVSAE respectively. We adopted RSA-FDH [21] as the secure signature algorithm in our PVSAE schemes. We then executed random queries over these encrypted data.

We implemented our constructions in JAVA with Java Pairing Based Cryptography library (JPBC) [22]. In our implementation, the bilinear map is instantiated as Type A pairing (base field size is 512-bit), which offers a level of security equivalent to 1024-bit DLOG [22].

The algorithms run by the data owner and the data users (i.e., Enc, TokenGen and Verify) were executed on a client machine with Windows 7, Intel i7-4600U 2.70GHz CPU, and 4GB RAM. The algorithms run by the cloud (i.e., Query) were executed on a server machine with Windows 7, Intel i7-3520M 2.90GHz CPU, and 8GB RAM.

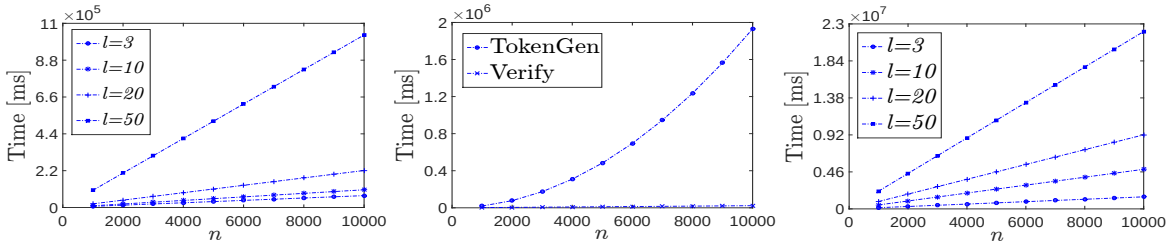
B. Implementation

Figure 2 shows the performance of our PVSAE schemes. Figure 2(a) shows the computation overhead of building whole encrypted index with n keywords. We observe that when $\ell = 3$, to build an index with 10,000 keywords, the data owner only needs to take about 71 seconds. However, when $\ell = 50$, the data owner needs to take about 17 minutes to build an index with 10,000 keywords. Figure 2(b) shows the computation overhead of token generation and verification. Take $n = 10,000$ as an example, the time for generating a search token is about 32 minutes, and the time for verifying the correctness of the search result is only about 23 seconds. Figure 2(c) plots the computation overhead of performing query over the encrypted index with n keywords. We observe that when ℓ is small (say $\ell = 3$) our PVSAE scheme is much efficient for searching over the whole encrypted index.

Figure 3 shows the performance comparison among VABKS [3], LWMN [4] and PVSAE Schemes. In order to facilitate comparison with VABKS [3] and LWMN [4], we set $\ell = S$, where ℓ is the vector dimension used in our construction and S is the number of attributes used in VABKS [3] and LWMN [4]. Figure 3(a) and 3(b) show the computation overhead of search and verification. We observe that our 3-PVSAE scheme are more efficient than VABKS [3] and LWMN [4]. The execution time of performing Query and Verify are only 153.24 milliseconds and 16.74 milliseconds. Figure 3(c) and 3(d) respectively show the storage and communication overhead. We observe that the storage overhead and communication overhead of our schemes are lower than VABKS [3] and LWMN [4].

VIII. CONCLUSIONS

We have presented the PVSAE framework for providing a privacy preserving and public verifiable service on inverted index-based encrypted data. This paper is novel from three perspectives: (1) We have presented a framework and formal security definitions for constructing a verifiable searchable asymmetric encryption scheme that supports efficient public verification. (2) Built on the proposed PVSAE framework, we have designed two complimentary PVSAE services, ℓ-PVSAE and 3-PVSAE. Both schemes enjoy strong notions of security, namely statistical IND-CKA security and statistical

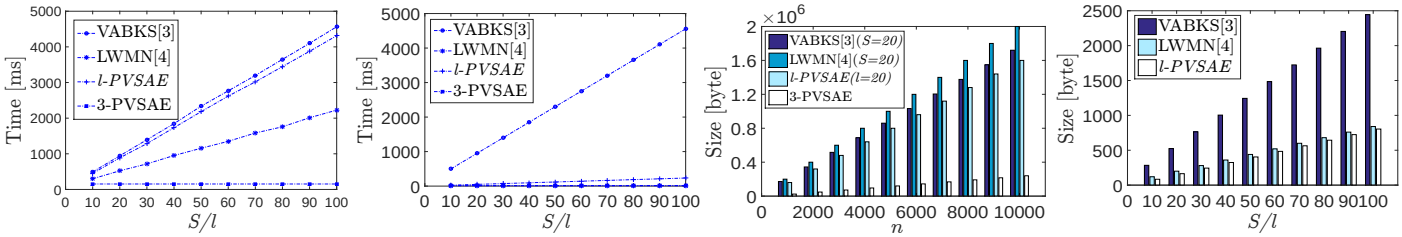


(a) The Computation Overhead of Building Encrypted Indexes

(b) The Computation Overhead of Token Generation and Verification

(c) The Computation Overhead of Querying

Fig. 2. Performance of PVSAE Schemes



(a) The Computation Overhead of Searching One Item of Index

(b) The Computation Overhead of Verification

(c) The Storage Overhead on the Server Side

(d) The Communication Overhead of Search Results

Fig. 3. Performance Comparison among VABKS [3], LWMN [4] and PVSAE Schemes

search pattern privacy, than existing searchable asymmetric encryption schemes. Furthermore, we show that 3-PVSAE offers high efficiency and low storage and communication overhead for searching over encrypted data, compared to existing verifiable attribute-based keyword search schemes.

ACKNOWLEDGMENT

Rui Zhang and Rui Xue are supported by the Strategic Priority Research Program of the Chinese Academy of Sciences, Grants No.XDA06010701, National Natural Science Foundation of China(No.61402471, 61472414). Ling Liu is partially supported by the National Science Foundation under Grants IIS-0905493, CNS-1115375, NSF 1547102, SaTC 1564097, and Intel ISTC on Cloud Computing.

REFERENCES

[1] B. Parno, M. Raykova, and V. Vaikuntanathan, "How to delegate and verify in public: Verifiable computation from attribute-based encryption," in *TCC'12*, ser. LNCS, 2012, vol. 7194, pp. 422–439.

[2] M. Azraoui, K. Elkhiyaoui, M. Önen, and R. Molva, "Publicly verifiable conjunctive keyword search in outsourced databases," Department of Network and Security, EURECOM, Tech. Rep. Research Report RR-15-303, April 3rd 2015.

[3] Q. Zheng, S. Xu, and G. Ateniese, "VABKS: verifiable attribute-based keyword search over outsourced encrypted data," in *INFOCOM'14*, 2014, pp. 522–530.

[4] P. Liu, J. Wang, H. Ma, and H. Nie, "Efficient verifiable public key encryption with keyword search based on KP-ABE," in *BWCCA'14*, 2014, pp. 584–589.

[5] D. Boneh, G. Crescenzo, R. Ostrovsky, and G. Persiano, "Public key encryption with keyword search," in *EUROCRYPT'04*, ser. LNCS, 2004, vol. 3027, pp. 506–522.

[6] M. Abdalla, M. Bellare, D. Catalano, and *et al.*, "Searchable encryption revisited: Consistency properties, relation to anonymous ibe, and extensions," in *CRYPTO'05*, ser. LNCS, 2005, vol. 3621, pp. 205–222.

[7] C. Blundo, V. Iovino, and G. Persiano, "Predicate encryption with partial public keys," in *CANS'10*, ser. LNCS, 2010, vol. 6467, pp. 298–313.

[8] M. Nishioka, "Perfect keyword privacy in peks systems," in *ProvSec'12*, ser. LNCS, 2012, vol. 7496, pp. 175–192.

[9] E. Shen, E. Shi, and B. Waters, "Predicate privacy in encryption systems," in *TCC'09*, 2009, pp. 457–473.

[10] J. Camenisch, M. Kohlweiss, A. Rial, and C. Sheedy, "Blind and anonymous identity-based encryption and authorised private searches on public key encrypted data," in *PKC'09*, ser. LNCS, 2009, vol. 5443, pp. 196–214.

[11] D. Boneh, A. Raghunathan, and G. Segev, "Function-private identity-based encryption: Hiding the function in functional encryption," in *CRYPTO'13*, ser. LNCS, 2013, vol. 8043, pp. 461–478.

[12] A. Arriaga, Q. Tang, and P. Ryan, "Trapdoor privacy in asymmetric searchable encryption schemes," in *AFRICACRYPT'14*, ser. LNCS, 2014, vol. 8469, pp. 31–50.

[13] A. Guillevic, "Comparing the pairing efficiency over composite-order and prime-order elliptic curves," in *ACNS'13*, ser. LNCS, 2013, vol. 7954, pp. 357–372.

[14] K. Kurosawa and Y. Ohtaki, "Uc-secure searchable symmetric encryption," in *FC'12*, ser. LNCS, 2012, vol. 7397, pp. 285–298.

[15] Q. Chai and G. Gong, "Verifiable symmetric searchable encryption for semi-honest-but-curious cloud servers," in *ICC'12*, June 2012, pp. 917–922.

[16] M. Mohamad and G. Poh, "Verifiable structured encryption," in *ISC'13*, ser. LNCS, 2013, vol. 7763, pp. 137–156.

[17] J. Wang, H. Ma, Q. Tang, J. Li, H. Zhu, S. Ma, and X. Chen, "Efficient verifiable fuzzy keyword search over encrypted data in cloud computing," *Comput. Sci. Inf. Syst.*, vol. 10, no. 2, pp. 667–684, 2013.

[18] K. Kurosawa and Y. Ohtaki, "How to update documents verifiably in searchable symmetric encryption," in *CANS'13*, ser. LNCS, 2013, vol. 8257, pp. 309–328.

[19] R. Cheng, J. Yan, C. Guan, F. Zhang, and K. Ren, "Verifiable searchable symmetric encryption from indistinguishability obfuscation," in *ASIACCS'15*, 2015, pp. 621–626.

[20] V. Shoup, "Oaep reconsidered," in *CRYPTO'01*, ser. LNCS, 2001, vol. 2139, pp. 239–259.

[21] M. Bellare and P. Rogaway, "Random oracles are practical: A paradigm for designing efficient protocols," in *CCS'93*, 1993, pp. 62–73.

[22] "The java pairing based cryptography library." [Online]. Available: http://gas.dia.unisa.it/projects/jpbc