A new Approach for Private Searches on Public-Key Encrypted Data

Amar SIAD

LAGA, UMR 7539, CNRS, Department of Mathematics, University of Paris XIII
and University Paris of VIII, 2 rue de la Liberté 93526 SAINT-DENIS, FRANCE
siad@math.univ-paris13.fr

Abstract. Public-key Encryption with Keyword Search (PEKS) allows authorized users to search by keywords on encrypted data by generating trapdoors for the desired keywords. Basically there are two requirements for the process of trapdoors generation. On one hand, it is important to allow users to privately search on encrypted data without revealing keywords to the Trapdoor Generation Entity $\mathcal{TGE}$, and solutions proposed are based on the use of blinding protocols between user and $\mathcal{TGE}$. On the other hand, in some applications it is necessary to prevent $\mathcal{TGE}$ from generating trapdoors on behalf of a legitimate user, this is done by splitting the $\mathcal{TGE}$ and generating trapdoors in distributed manner. Until now, these two requirements in PEKS were considered separately and many solutions were proposed dealing with each one. However, no solution is known to satisfy the two requirements at the same time.
In this paper we present a new approach to deal with this issue. First, we extend the notion of anonymous Identity-Based Encryption with distributed Private Key Generator to handle the blind key-extraction protocols as in Blind IBE. We call such new schemes blind anonymous (n, t)-IBE, and we give concrete construction of such schemes. Second, we generically transform the blind anonymous (n, t)-IBE into a threshold PEKS with oblivious keyword search satisfying the requirements specified above.

Keywords: Threshold Searchable Encryption, Public key Encryption with Keyword Search, Blind Identity-Based Encryption.

1 Introduction

Public-key encryption with keyword search. The first formalization of the concept of public-key encryption with keyword search (PEKS) was proposed by Boneh, et al. [2], the authors presented an application that allows user Alice to have the email server extract encrypted emails that contain a particular keyword by providing a trapdoor corresponding to that keyword, while the email server and other parties do not learn anything else about the email. PEKS is very useful, it provides mechanism for searching on encrypted data for public key cryptosystems. Furthermore, it has a very close connection to anonymous identity-based encryption as defined in [2]. This connection has been studied more thoroughly by Abdalla et al. [1]. For this reason, most work (including ours) on asymmetric searchable encryption has a direct use for identity-based encryption, and vice versa.

After [2] a flurry of PEKS schemes were proposed [9,12,10,16], each one deals with a special scenario application. We focus our interest on special schemes [12,10,16] proposing mechanisms for generating trapdoors. Camenisch et al. [10] introduced the concept of public key encryption with oblivious keyword search (PEOKS) which uses Blind key extraction protocols to hide keyword from the $\mathcal{TGE}$. They extended the definition of PEKS to incorporate the encryption of a secret message when computing a searchable encryption. This secret message can contain a symmetric key, which allows PEKS to be used directly in settings such as [9]. Hiding keywords from $\mathcal{TGE}$ allows generation of trapdoors in blind manner. However, $\mathcal{TGE}$ can always try to derive all possible trapdoors according to a certain "dictionary". It seems that there is not much can be done to protect the scheme against a strong adversary like the $\mathcal{TGE}$ in this situation. Furthermore, as noticed by the authors, using PEOKS in practice needs an additional trusted authority which is responsible of certifying keywords.

Wang et al. [12] constructed a threshold keywords search scheme, using Shamir verifiable secret sharing, in which users encrypt their data and upload it to the server with secure indexes, then a subset of users should collaborate to generate a trapdoor for the target keywords. Decryption also is done in threshold manner, and secret keys used for decryption and trapdoor generation are chosen and distributed by a centralized dealer. The scheme presented is based on the random oracle IBE of Boneh and Franklin [17] and no formal relation between the PEKS scheme and the underlying IBE scheme is given. Furthermore, the given scheme does not support the feature of hiding keywords from $\mathcal{TGE}$. In the same category, Siad [16] showed how to generically construct threshold PEKS from anonymous IBE with distributed Private Key Generator and presented a practical architecture for deployment of TPEKS. However, as in [12] the proposed solution does not hide keywords from the $\mathcal{TGE}$. Furthermore, no concrete construction is given.

Identity-based encryption with distributed PKG. A well known approach to design distributed protocols is to use multi-party computation. Although results obtained in general multi-party computation guarantee feasibility, they cannot be directly applied without affecting computation complexity. In contrast, threshold protocols for specific tasks should be developed at hand and are therefore much more efficient and practical.

In the identity-based encryption settings, there have been many proposals. Boneh and Franklin [17] were the first to suggest the distribution of the master secret key, their scheme have the advantage of being straightforward to distribute. Geisler and Smart [4] proposed a distributed version of sakai-kasahara based systems which requires for each ID-based key a secure multi-party computation to be performed amongst the servers and presented an implementation in the semi-honest model. The drawback oft he proposed solution is that it lacks a formal security analysis. Recently, Kate and Goldberg [6] considered IBE schemes with more complex user's private key structure (distribution of such schemes requires inversion in the exponent, addition, and multiplication of shares). The authors developed a distributed private-key generators for three IBE schemes along with the security proofs in the random oracle model. Another interesting work [3,5] with an application in the IBE settings, proposes a different approach in which the $\mathcal{PKG}$ is replaced by a distributed set of users, each one of them holding a small piece of the master secret in the form of a short password. The authors defined ideal functionalities for distributed public-key generation and virtual private-key computation in the UC framework and constructed efficient protocols that securely realize them.

Our contribution. First, we formalize the concept of blind anonymous (n,t)-IBE[1] as an extension of anonymous IBE schemes with distributed PKG for which we add a blinding phase. In this context, anonymous means that the ciphertext does not leak the identity under which it was encrypted [1] and blind means that a user can request the decryption key for a given identity without the PKGs learning the identity [23]. Second, we give a concrete construction of a blind anonymous (n,t)-IBE along with the security proofs. Our construction focuses only on the blinding protocol under the standard definitions of security following the ideal/real world paradigm in the standalone setting [13] and we discuss its extension to the universal composability framework [14]. Finally, we give an application of the blind anonymous (n,t)-IBE in the setting of PEKS. We show how to generically transform a blind anonymous (n,t)-IBE into a threshold PEKS with oblivious keyword search and we described a system architecture for real world deployment of TPEOKS to build a public key encrypted database with distributed and blinded generation of trapdoors.

2 Preliminaries

2.1 Identity-Based Encryption

An identity-based encryption (IBE) scheme supports two types of players: a single Private Key Generator $\mathcal{PKG}$ and multiple users. An IBE scheme consists of algorithms (IBESetup, IBEKeyGen, IBEExtract, IBEEnc, IBEDec).
Anonymous (n,t)-IBE. Abdella et al. [1] defined anonymity against user attacks for IBE similarly to semantic security. The attacker's goal is to distinguish the intended recipient of a ciphertext between two chosen identities. An anonymous IBE scheme with distributed $\mathcal{PKG}$ ((n,t)-IBE) extends a regular anonymous IBE scheme by distributing the master secret key of the $\mathcal{PKG}$ among n

[1] A related concept of our blind anonymous (n,t)-IBE was proposed by Siad and Amara [20]. However, their security model is based on the use of properties of p-signatures [7] and their protocol is less efficient than ours since it relies on a general two-party protocol and no formal analysis was given. Furthermore, the authors does not relate their scheme to the concept of searchable encryption.

$\mathcal{PKG}$s. In addition to IBEEnc and IBEDecr algorithms which remain unchangeable, an (n, t)-IBE scheme consists of the following algorithms:

- **IBEGlobalSetup**: Off-line bootstrapping procedure run by any untrusted entity to generate global parameters $\mathcal{GP}$. This includes the threshold parameters, description of the groups and their generators, and the bilinear pairing.
- (n, t)**-IBEKeyGen**: Distributed protocol run between n players takes global parameters $\mathcal{GP}$ as input and outputs for each player a share sk_i of a master secret key msk and the master public key mpk.
- (n, t)**-IBEExtract**: Distributed protocol that takes as input, for each player, the user identity id, the public key mpk and the master-secret share sk_i and outputs a verifiable private-key share $d_{id}^{(i)}$. The user computes her private key d_{id} after verifying the received shares $d_{id}^{(i)}$ from at least $t + 1$ $\mathcal{PKG}$.

CPA-security and user-anonymity for (n, t)-IBE ((n, t)-IBE-IND-CPA, (n, t)-IBE-ANON-CPA) are defined [16]. In addition to the key extraction oracle $\mathcal{O}_{Extract(.)}$ a new oracle $\mathcal{O}_{Corrupt(t,.)}$ is defined to allow the adversary to choose t players to corrupt. Furthermore, (n, t)-IBE-IND-CPA and (n, t)-IBE-ANON-CPA can be combined in one game called (n, t)-IBE-IND-ANON-CPA security.

2.2 Public-key Encryption with Keyword Search

Definition 1. ***(PEKS)*** *[10] A non-interactive public key encryption with keyword search scheme consists of the following polynomial time randomized algorithms:*

- ***KeyGen(**1^k**):** generates public/private key pair pk, sk.*
- ***PEKS(**pk, W, M**):** produces a searchable encryption $S_{W,M}$ for keyword W and message M.*
- ***Trapdoor(**sk, W**)** : produces a trapdoor T_W for keyword W.*
- ***Test(**$S_W, T_{\hat{W}}$**):** outputs M if $\hat{W} = W$ and $\perp$ otherwise.*

In PEKS, the party holding the secret key sk (called Trapdoor Generation Entity $\mathcal{TGE}$) runs the Trapdoor algorithm to obtain the trapdoor T_w for a keyword W. Threshold Public key encryption with keyword search (TPEKS) is an extension of PEKS in which a user U performing a search can generate trapdoor shares $T_{i,w}$ from the trapdoor generation entities $\mathcal{TGE}$s in a distributed manner and combine them to compute the final trapdoor.

Definition 2. ***(TPEKS)*** *A threshold public key encryption with keyword search scheme consists of the following polynomial time randomized algorithms:* (1) ***Setup(**1^k**):** generates $\mathcal{GP}$, the parameters of the scheme;* (2) (n, t)***-KeyGen(**$\mathcal{GP}$**):** generate the master key pair msk/mpk, each player P_i gets its own share sk_i of msk;* (3) ***TPEKS(**$\mathcal{GP}, W, M$**):** produces a searchable encryption $S_{W,M}$ for keyword W and message M;* (4) (n, t)***-Trapdoor(**W**):** produces shares $T_{i,W}$ of the trapdoor T_W for W;* (5) ***Test(**$S_{W,M}, T_{\hat{W}}$**):** outputs M if $\hat{W} = W$ and $\perp$ otherwise.*

Proof of Knowledge. We use the following notation for proofs of knowledge of discrete logarithms and proofs of the validity of statements about discrete logarithms. PoK$\big\{(a,b) : \mathcal{C} = g^a h^b\big\}$. This proof denotes a "zero-knowledge Proof of Knowledge of integers a, b such that the equality $\mathcal{C} = g^a h^b$ holds". The convention used here is that values in the parenthesis denote values known only to the prover, while all other values are known to the verifier.

3 Formal Definition and Security Model

Blind anonymous (n,t)-IBE. A blind anonymous (n,t)-IBE scheme consists of the same players as the anonymous (n,t)-IBE, together with the same algorithms IBEGlobalSetup, (n,t)-IBEKeyGen, (n,t)-IBEExtract, IBEEnc, IBEDec, a secure commitment scheme $\mathcal{COM}$=(Setup,Commit), and the protocol (n,t)-IBEBlindExtract defined as follows:

- $\overline{\pi}_{Blind}$: The blinding protocol $\overline{\pi}_{Blind}$ is composed of a sequence of two-party protocols
  π_i : $((\mathcal{PKG}_i(\mathcal{GP}, sk_i, \mathcal{C}), \mathcal{U}(\mathcal{GP}, id, open_{id})) \rightarrow (\overline{id}_i, nothing))$ run between the user with identity id, opening information $open_{id}$ as private inputs and $\mathcal{PKG}_i$ with secret key sk_i as private input. If $\mathcal{C} = commit(id, open_{id})$ then the $\mathcal{PKG}_i$ gets the blinded identity $\overline{id}_i$ and the users gets nothing. Otherwise both parties output $\perp$.
- (n,t)**-IBEExtract** the same as in (n,t)-IBE. Yet, the players takes as input the resulting blinded identity from the protocol $\overline{\pi}_{Blind}$.

An alternative and more natural definition is to require the Blind anonymous (n,t)-IBE be an extension of a regular IBE scheme and to require that (n,t)-IBEBlindExtract to be a secure multi-party computation protocol in which the user enters his identity and a commitment to it, and in which the parties representing the distributed key generation authority enter their key shares. This definition covers blindness and does not leak any information to PKGs about the user's identity. However, as we prefer modular composition of protocols and we assume existence of (n,t)-IBE schemes we prefer the first definition and we require that protocols $\pi_1, ..., \pi_n$ satisfy security properties of two-party computation in both semi-honest and malicious adversary models. Furthermore, using generic tools may be inefficient, since we seek to optimize this specific computation, we present an efficient implementation of these protocols.

3.1 Security model

In two-party computation, two parties want to jointly compute a function of their own secret inputs. The standard way of defining security in this setting is through the ideal/real simulation paradigm which provides strong security guarantees like privacy, correctness, etc. and has an important property which is the composition theorem that guarantees certain behaviour when the secure protocol is used as a sub-protocol in another larger protocol. However, in many other settings including our case, it may be sufficient to guarantee privacy only. Indeed, one can use privacy only definition of security in the malicious model in similar way as it is defined for oblivious transfer in [18]. Yet, as noticed by the authors, no general composition theorem is known for the definition that follows privacy-only approach. As our blinding protocols should operate as sub-protocols in the extraction protocol we preferred the first approach to use the advantage of the composition theorem. Indeed, we define security of protocol π_i according to the real/ideal paradigm. First, we define an ideal model where a trusted party is used to compute the function (the blinded identity in our case) for the two parties in "perfect security". Then, we construct a real protocol which is said secure if no adversary can do more harm in the real world execution than in the ideal world execution.

Real world execution. in the real world, the blinding protocol π_i run between the user $\mathcal{U}$ and the $\mathcal{PKG}_i$ as follows: If $\mathcal{PKG}_i$ is corrupted then π_i is run between $\mathcal{U}$ and adversary $\mathcal{A}$ corrupting $\mathcal{PKG}_i$: π_i $(\mathcal{A}(\mathcal{GP}, \mathcal{C}, state), \mathcal{U}(\mathcal{GP}, id, open_{id}))$. In case $\mathcal{U}$ is corrupted then π_i is run between $\mathcal{PKG}_i$ and adversary $\mathcal{A}$ corrupting U as follows: π_i $(\mathcal{PKG}_i(\mathcal{GP}, sk_i, \mathcal{C}), \mathcal{A}(\mathcal{GP}, sate))$ with $\mathcal{PKG}_i$.

Ideal world execution. Ideal world execution is defined by means of a functionality $\mathcal{F}_{Blind}$ that allows one corruption of either user or $\mathcal{PKG}$ and not both of them at the same time. $\mathcal{F}_{Blind}$ is intended to capture blinding protocols for an arbitrary IBE scheme (and not a specific scheme). Thus, we will use the following function as black-box parameter in our definition of the functionality:
BlindedId: $(sk_i, id) \mapsto \overline{id}_i$: Given an identity id and secret key sk_i, this function computes the blinded identity $\overline{id}_i$.

$\mathcal{F}_{Blind}$: This functionality is given a security parameter k, public commitment $\mathcal{C}$ to the identity, and runs with two parties $\mathcal{PKG}_i, \mathcal{U}$, and an adversary $\mathcal{S}$. $\mathcal{F}_{Blind}$ is also given a function **BlindedId**: $(sk_i, id) \mapsto \overline{id}_i$, where $\overline{id}_i$ is the blinded identity.

- **Inputs.** $\mathcal{PKG}_i$'s private input is sk_i and $\mathcal{U}$'s private input is $(id, open_{id})$. $\mathcal{S}$ sees inputs and outputs of corrupted party which he may change.
- **Computation.** $\mathcal{F}_{Blind}$ receives inputs of honest parties and compute $\overline{id}_i =$ **BlindedId** (sk_i, id) if $\mathcal{C} = commit(id, open_{id})$.
  - If $\mathcal{PKG}_i$ is corrupted send $\overline{id}_i$ to $\mathcal{S}$, send $\perp$ to $\mathcal{U}$,
  - If $\mathcal{U}$ is corrupted send $\perp$ to $\mathcal{S}$, send $\overline{id}_i$ to $\mathcal{PKG}_i$.
- **Output.** Honest $\mathcal{PKG}_i$ sets his output to $\overline{id}_i$. $\mathcal{U}$ sets his output to $\perp$.

Having defined the real and ideal models, we should now define security which asserts that a real-world protocol emulates the ideal-world (which run with a trusted party). More formally, adversaries in ideal model are able to simulate adversaries in real model. We define $\mathcal{PKG}$ security and user security as follows:
$\mathcal{PKG}$ **Security (Corrupt user).** for every probabilistic polynomial-time adver-

sary $\mathcal{A}$ for the real model, there exists a probabilistic polynomial-time adversary $\mathcal{S}$ for the ideal model such that for every auxiliary input $z \in \{0,1\}^*$:

$$\mathrm{REAL}_{\mathcal{A}(z),\mathcal{PKG}}(id, open_{id}, sk_i) \overset{c}{\approx} \mathrm{IDEAL}_{\mathcal{S}(z),\mathcal{PKG}}(id, open_{id}, sk_i)$$

User Security (Corrupt $\mathcal{PKG}$): for every probabilistic polynomial-time adversary $\mathcal{A}$ for the real model, there exists a probabilistic polynomial-time adversary $\mathcal{S}$ for the ideal model such that for every auxiliary input $z \in \{0,1\}^*$:

$$\mathrm{REAL}_{\mathcal{A}(z),\mathcal{U}}(id, open_{id}, sk_i) \overset{c}{\approx} \mathrm{IDEAL}_{\mathcal{S}(z),\mathcal{U}}(id, open_{id}, sk_i).$$

Definition 3. *The blinding protocol $\overline{\pi}_{Blind}$ is secure if each one of its subprotocols π_i for $i = 1, ..., n$ satisfies $\mathcal{PKG}$ Security and User Security properties.*

Definition 4. ***(Security for blind anonymous** (n,t)**-IBE)***
Let Π an anonymous (n,t)-IBE scheme. A blind anonymous (n,t)-IBE scheme $=$ (Π, (n,t)-IBEBlindExtract($\overline{\pi}_{Blind}$,(n,t)-IBEExtract), $\mathcal{COM}$) is (n,t)-IBE-IND-CPA (resp. (n,t)-IBE-ANON-CPA) secure in the static malicious model if and only if: (1) The underlying Π is (n,t)-IBE-IND-CPA (resp. (n,t)-IBE-ANON-CPA) secure, (2) $\mathcal{COM}$ is a secure commitment scheme, and (3) $\overline{\pi}_{Blind}$ is secure (in the sense of definition 3).

4 Construction

We present in this section an efficient implementation of a blinding protocol for Gentry [19] scheme using ideas from protocols developed in [8]. The choice of Gentry scheme is motivated by many factors. Firstly, we require an anonymous scheme which is secure in the standard model and sufficiently efficient, this requirement excludes many distributed schemes like Sakai-Kasahara in [20,4,6], Boneh-Franklin in [6]. Secondly, we require a scheme for which there exists a distributed version that is relatively efficient and practical, since Gentry's scheme shares similarities with Sakai-Kasahara, it can be distributed the same way as in [4,6], this requirement excludes also the scheme of Boyen-Waters [15] modified in [10] due to its complex structure of user's private-keys which makes the distribution of these schemes much more difficult.
Let $\mathcal{COM} = (Setup, Commit)$ be a secure commitment scheme and let Π(KeyGen, Enc, Dec) be an additively homomorphic and semantically secure encryption scheme, such that $Enc(e) \otimes r = Enc(er)$, and $Enc(a) \oplus Enc(b) = Enc(a+b)$. We require also Π to be verifiable, that is there exist an efficient proofs of knowledge on encrypted messages (e.g. the content of a ciphertext belongs to a given interval). We construct the blinding protocol as follows:

1. $\mathcal{PKG}_i$ generates a key pair (pk_{hom}, sk_{hom}) for the homomorphic scheme Π by running algorithm $(pk_{hom}, sk_{hom}) \leftarrow$KeyGen$(1^k)$. Then computes $c_1 = Enc(pk_{hom}, \alpha_i)$ and sends c_1, pk_{hom} to the user $\mathcal{U}$ and engages with her in an interactive zero-knowledge proof $PoK_1 = PoK\{(\alpha_i) : c_1 = Enc(\alpha_i)\}$ that c_1 encrypts to a message in $[0, p]$. If the proof fails abort.

2. The user $\mathcal{U}$ randomly chooses $\rho_1 \xleftarrow{R} \mathbb{Z}_p$, $\rho_2 \xleftarrow{R} [0, 2^k p]$, and computes $c_2 = ((c_1 \oplus Enc(pk_{hom}, -id)) \otimes \rho_1) \oplus Enc(pk_{hom}, r\rho_2)$. Then sends c_2 and $\mathcal{C} = commit(id, open_{id})$ to $\mathcal{PKG}_i$. (see section 5 for how $\mathcal{C}$ is used to handle stateful randomness).
3. $\mathcal{U}$ and $\mathcal{PKG}_i$ perform an interactive zero-knowledge proof $PoK_2 = PoK\{ (id, open_{id}, \rho_1, \upsilon, \mu) : c_2 = ((c_1 \oplus \upsilon) \otimes \rho_1) \oplus \mu \wedge \mathcal{C} = commit(id, open_{id})\}$ in which $\mathcal{U}$ shows that c_2 has been computed correctly using the identity id committed to in $\mathcal{C}$ and ρ_1, ρ_2 are in the appropriate range. If the proof fails abort.
4. $\mathcal{PKG}_i$ decrypts c_2 and does a single modulo p reduction to obtain $\overline{id}_i = Dec(sk_{hom}, c_2) \bmod p = (\alpha_i - id)\rho_1$.

Theorem 1. *Assume that The $\Pi(KeyGen, Enc, Dec)$ is a homomorphic semantically secure encryption scheme relative to addition, and that $\mathcal{COM} = (Setup, Commit)$ is a secure commitment scheme. Then, the above protocol is a secure blinding protocol for Gentry scheme that satisfies User security and $\mathcal{PKG}$ security properties.*

Proof. The proof is divided into two parts, one to show that the protocol meets the $\mathcal{PKG}$ security property, and a second to show User security.

$\mathcal{PKG}$ **security.** For any real-world adversary $\mathcal{A}$ (corrupting the user) we construct a simulator $\mathcal{S}$ such that the "real" and "ideal" executions are computationally indistinguishable. More formally, for any distinguisher $\mathcal{D}$ there exists a negligible function $\nu(.)$ such that:
Pr[$\mathcal{D}$(REAL$_{\mathcal{A}(z),\mathcal{PKG}}(id, sk_i)$)]-Pr[$\mathcal{D}$(IDEAL$_{\mathcal{S}(z),\mathcal{PKG}}(id, sk_i)$)] $\leq v(k)$.
Note that the simulator $\mathcal{S}$ can rewind an instance of the adversary $\mathcal{A}$ that he runs internally. He simulates the communication between $\mathcal{D}$ and $\mathcal{A}$ by passing $\mathcal{D}$'s input to $\mathcal{A}$ and $\mathcal{A}$'s output to $\mathcal{D}$. To construct a simulator $\mathcal{S}$, we consider the following sequence of games. We start from the first game representing the real world experiment and modify elements via a series of games until we arrive to the last game which represents the ideal world execution. Let Adv [Game i] be $\mathcal{D}$'s advantage in distinguishing the output of Game i from the Real distribution, we define also a set of negligible functions $\nu_1(.), ..., \nu_n(.)$ where $\nu_i(.)$ is the i^{th} function for the i^{th} game.
Game 0: In this game the honest real-world $\mathcal{PKG}$ interacts with the adversary $\mathcal{A}$ as in the real world protocol. It is clear that Adv [Game 0] = 0.
Game 1: $\mathcal{A}$ interacts with $\mathcal{S}'$ which behaves as the real protocol for steps 1 and 2, but then uses the knowledge extractor for PoK_2 to extract from $\mathcal{A}$ the values ρ_1,id, $open_{id}$ and outputs $\perp$ if the knowledge extractor fails or $\mathcal{C} \neq commit(id, open_{id})$. Since this extractor fails with probability negligible in k, then Adv[Game 1] - Adv [Game 0] $\leq \nu_1(k)$.
Game 2: $\mathcal{A}$ interacts with $\mathcal{S}$. In this game, we replace c_1 by the value $c_1^* = Enc(0)$. The difference from Game 1 is equal to $\mathcal{D}$'s advantage in distinguishing the encryption of the value 0 from an encryption of a random value. Adv[Game

2] - Adv [Game 1] $\leq$ Adv[Enc]. By the security of the encryption scheme we have Adv[Enc] $\leq \nu_2(k)$. Thus Adv[Game 2] - Adv [Game 1] $\leq \nu_2(k)$.
The ideal-world simulator $\mathcal{S}$ is an algorithm that performs all of the changes between the games above, and runs internally $\mathcal{A}$, it is described as follows:

1. First, $\mathcal{S}$ honestly generates a key pair $(pk_{hom}, sk_{hom}) \leftarrow$KeyGen$(1^k)$, computes $c_1^* = Enc(pk_{hom}, 0)$, and sends c_1, pk_{hom} to $\mathcal{A}$. $\mathcal{S}$ uses the zero-knowledge simulator to simulate the ZK proof PoK_1 with $\mathcal{A}$.
2. $\mathcal{S}$ receives from $\mathcal{A}$ random values c_2 and $\mathcal{C}$.
3. $\mathcal{S}$ acts as the verifier for the proof that c_2 was computed correctly. If the proof fails to verify, $\mathcal{S}$ aborts. Otherwise, $\mathcal{S}$ runs the proof of knowledge extraction algorithm to extract $(id, open_{id}, \rho_1)$ and checks that $\mathcal{C} = commit(id, open_{id})$ if it isn't it aborts.
4. Next, $\mathcal{S}$ submits id, ρ_1 to the trusted party, who returns the valid blinded identity $\overline{id}_i = (\alpha_i + id)\rho_1$. Finally, $\mathcal{S}$ returns $\perp$ to $\mathcal{A}$.

Summing the differences between the above games, it is clear that Adv [Game 2] is negligible, and therefore no p.p.t. algorithm can distinguish the distribution of Game 2 from Game 0. Thus,
Pr[$\mathcal{D}$(REAL$_{\mathcal{A}(z),\mathcal{PKG}}(id, sk_i)$)]-Pr[$\mathcal{D}$(IDEAL$_{\mathcal{S}(z),\mathcal{PKG}}(id, sk_i)$)] $\leq \nu(k)$. □

User security. For any real-world adversary $\mathcal{A}$ (corrupting the PKG) we construct a simulator $\mathcal{S}$ such that, for any distinguisher $\mathcal{D}$:
Pr[$\mathcal{D}$(REAL$_{\mathcal{A}(z),\mathcal{U}}(id, sk_i)$)]-Pr[$\mathcal{D}$(IDEAL$_{\mathcal{S}(z),\mathcal{U}}(id, sk_i)$)] $\leq \nu(k)$.

We consider series of games starting from the real experiment until we arrive at the ideal execution. Let Adv [Game i] be $\mathcal{D}$'s advantage in distinguishing the output of Game i from the Real distribution.
Game 0: In the first game the honest real-world user interacts with the adversary $\mathcal{A}$ as in the real world protocol. Clearly Adv [Game 0] = 0.
Game 1: in this game, $\mathcal{A}$ interacts with an alternative simulator $\mathcal{S}'$ which runs as in the real protocol, but the zero-knowledge proof-of-knowledge PoK_2 is replaced by a simulated proof. We have Adv[Game 1]- Adv[Game 0]$\leq$ Adv[ZK], where Adv[ZK] is the advantage of an adversary in breaking the security of the zero-knowledge proof system which is negligible. Thus, Adv[Game 1]- Adv[Game 0]$\leq \nu_1(k)$.
Game 2: In this game, $\mathcal{A}$ interacts with $\mathcal{S}''$. The difference from Game 1 is that the computing of c_2 is replaced by $c_2' = x \oplus c_1$ and the commitment $\mathcal{C}$ to id is replaced by a commitment $\mathcal{C}^*$ to x, where x is a value chosen at random. The difference between Game 2 and Game 1 is equal to D's advantage in distinguishing distribution of c_2' from the one of c_2 and $\mathcal{C}^*$ from $\mathcal{C}$. We observe that c_2' is distributed statistically close to the uniform distribution, since x is random and c_1 is an encryption of a value chosen at random in $\mathcal{Z}_p$. In the same way the value c_2 computed in the real protocol is distributed statistically close to the uniform distribution. Finally, the security of the commitment scheme guarantees that no adversary can distinguish $\mathcal{C}^*$ from the valid commitment $\mathcal{C}$. Thus Adv[Game 2]-Adv[Game 1] $\leq \nu_2(k)$.

Game 3: In the final game, we employ the knowledge extractor to extract from $\mathcal{A}$ the value α_i. If the knowledge extractor fails, set $\mathcal{S}$ output to $\perp$. Let $Pr[error]$ be the probability that the knowledge extractor fails, then Adv[Game 3] - Adv [Game 2] $\leq$ Pr[$error$]. We know that Pr[$error$]$\leq$ $\nu_3(k)$, Thus Adv[Game 3]-Adv[Game 2]$\leq$ $\nu_3(k)$

The ideal-world simulator $\mathcal{S}$ is an algorithm that performs all of the changes between the games above, and runs internally $\mathcal{A}$, it is described as follows:

1. $\mathcal{S}$ receives c_1, pk_{hom} from $\mathcal{A}$ and acts as verifier for ZK proof that c_1 is correctly computed. If the proof fails $\mathcal{S}$ aborts. Otherwise $\mathcal{S}$ runs proof of knowledge extractor to extract α_i from the proof. $\mathcal{S}$ submits α_i to the trusted party and gets $\perp$.
2. $\mathcal{S}$ chooses random values $x, open_x$, and computes $c_2' = c_1 \oplus Enc(x)$, $\mathcal{C}^* = commit(x, open_x)$ and sends $c_2', \mathcal{C}^*$ to $\mathcal{A}$. Then, $\mathcal{S}$ uses the simulator for the zero knowledge proof to interact with the adversary for a proof of knowledge of x.

Summing the differences between the above games, we get Adv[Game 3] - Adv[Game 0] is negligible, and therefore no p.p.t. algorithm can distinguish the distribution of Game 3 from Game 0. Thus, Pr[$\mathcal{D}$(REAL$_{\mathcal{A}(z),\mathcal{U}}(id, sk_i)$)]-Pr[$\mathcal{D}$(IDEAL$_{\mathcal{S}(z),\mathcal{U}}(id, sk_i)$)] $\leq \nu(k)$ □

4.1 Extension to the universal composition model

Universal composition framework [14] guarantees security under concurrent execution. A protocol that is universally composable (UC secure) maintains its security even when it is run in an arbitrary environment and with other protocols. Compared to the security model of [13] used in our work, the UC framework introduces an additional entity called the environment $\mathcal{Z}$ which interacts with the adversary $\mathcal{A}$ in an arbitrary way. An immediate consequence of this free interaction between $\mathcal{Z}$ and $\mathcal{A}$ is that it imposes a restriction in the security proofs when constructing the simulator $\mathcal{S}$. The latter cannot use rewinding technique anymore. The protocol presented above is secure in the non-concurrent model, it is possible to extend it to the concurrent model, since the UC framework preserves the overall structure of the approach taken in [13]. Thus, to construct a UC-secure version of our protocol we should consider the following differences:

1. **Commitment schemes.** We should replace the commitment scheme with a universally composable scheme that UC realizes functionality $\mathcal{F}_{COM}$. Our protocol can use any UC-secure commitment scheme, in particular the efficient scheme proposed by Damgard and Neilson [25] can be used. We notice that UC-secure commitment protocol requires an additional trusted setup assumption such as a common reference string.
2. **Rewinding techniques.** The only places in the security proof of the blinding protocol in which the simulator uses rewinding are within the zero-knowledge proofs. This fact can be used to modify the protocol in such a way it will be possible to construct a straight-line simulator which is enough

to move to universal composability model. Thus, we should replace the zero-knowledge proofs in our protocol with new zero-knowledge proofs that have non-rewinding simulators.

3. **Reactive tasks.** Another difference between the two models, is that the UC framework allows capturing not only secure function evaluation but also reactive tasks, which is obtained by replacing the trusted party by an ideal functionality. However, Goldreich [24] showed that the definition of security in [13] can be extended to the case of reactive tasks in a straightforward way by allowing the trusted party to obtain inputs and send outputs in phases (for more details see discussion in [24] section 7).
4. **Homomorphic encryption.** Another element to consider is the encryption scheme, although we used a generic scheme which can be efficiently instantiated by Paillier scheme [22], it remains to investigate schemes of practical use which are secure in the UC framework. We can use as a starting point the work of Prabhakaran and Rosulek [21] that showed that indistinguishability-based security definitions of homomorphic encryption imply a natural definition of security in the Universal Composition framework.

5 Threshold PEKS with oblivious keyword search

We follow a generic transformation in [1,10,16] to transform our blind anonymous (n,t)-IBE into TPEKS with oblivious keyword search. Notice that Gentry's scheme uses stateful randomness in the secret key generation phase. However, as noted in [19], this requirement can be overcome easily and generically to get a stateless key-authority by using a pseudo-random function $f(id)$. In case of blinding protocols, this becomes more difficult since the $\mathcal{PKG}$ does not have access to the identity list. Hence, we handle this requirement as follows: since the blinding protocols require an external entity for certifying identities by the generation of certificates on the committed identities as in [11] or anonymous credentials *cred* as in [10], a simple solution of stateful randomness in this case consists of adding a pseudo-random function $f \in \mathcal{F}$ and always use random coins derived from $f(com)$, where *com* is a commitment to the identity signed by the certifcate (resp. credential). However, we should make the assumption that this new entity does not collude with $\mathcal{PKG}$s and generates always the same commitment for the same identity.

Given a blind anonymous (n,t)-IBE scheme (IBEGlobalSetup, (n,t)-IBEKeyGen, (n,t)-IBEBlindExtract, IBEEnc, IBEDec, $\mathcal{COM}$) which is (n,t)-IBE-ANON-CPA secure, the TPEOKS scheme with oblivious keyword search is as follows:

1. **Setup(1^k)**: On input a security parameter k, run IBEGlobalSetup(1^k) to generate global parameters $\mathcal{GP}$.
2. (n,t)**-KeyGen($\mathcal{GP}$)**: On input global parameters $\mathcal{GP}$ run the distributed key generation algorithm (n,t)-IBEKeyGen($\mathcal{GP}$) to generate shares of the master secret key *msk* and the master public key *mpk*.
3. **TPEKS($\mathcal{GP}, W, M$)**: On input a keyword W and a message M, this algorithm picks a random value $C_2 \in \{0,1\}^k$ and computes the cyphertext

$C_1 = IBEEnc(\mathcal{GP}, W, C_2\|M)$. As a result, the algorithm outputs the tuple $S_{W,M} = (C_1, C_2)$.

4. (n,t)**-BlindTrapdoor**: To generate a trapdoor T_W associated with the keyword W, the user $\mathcal{U}$ uses a commitment to her keyword $\mathcal{C} = commit(W, open_W)$ and run (n,t)-IBEBlindExtract with $\mathcal{TGE}$s where user takes as private inputs $W, open_W$ and $\mathcal{TGE}_i$'s private input is sk_i. Finally, the user gets a blinded trapdoor shares $T_{\overline{W}_i}$ which she can transform into T_W by combining and unblinding at least $t+1$ shares.
5. **Test(**$S_{W,M}, T_{\hat{W}}$**)**: On input the searchable encryption $S_{W,M}$ and the trapdoor $T_{\hat{W}}$, it outputs M if $C_2\|M = IBEDec(T_{\hat{W}}, C_1)$ and $\perp$ otherwise.

Definition 5. *Let Υ be a TPEKS scheme. The TPEOKS scheme defined by (Υ,(n,t)-BlindTrapdoor, $\mathcal{COM}$) is secure if and only if: (1) the underlying Υ is secure TPEKS scheme, (2) $\mathcal{COM}$ is a secure commitment scheme, and (3) (n,t)-BlindTrapdoor is instantiated using $\overline{\pi}_{Blind}$ protocol which is secure (see definition 3).*

5.1 System architecture for TPEOKS

We briefly describe an architecture that enables the use of threshold trapdoor generation in a blinded manner. We propose an architecture that extends the one presented in [10]. It is composed of composed of four entities: Trapdoor Generation Entities $\mathcal{TGE}s$, data holders, keywords certifying authority $\mathcal{KCA}$, and the investigator. The overall architecture works as follows:

First, Trapdoor Generation Entities $\mathcal{TGE}s$ run jointly algorithm (n,t)-KeyGen to generate master public key mpk and shares sk_i of the master secret key. Second, data holders encrypt each data record using a random symmetric key K and generate a searchable encryption $S_{W,K}$ for a keyword W by running TPEKS$(\mathcal{GP}, W, K)$. The generated searchable encryption is stored with the data record in the database. The investigator requests authorisation from $\mathcal{KCA}$ to perform a search on a given database for a particular keyword W and gets back a credential *cred* with attribute W. Next, the investigator presents the credential *cred* to $\mathcal{TGE}s$ and gets trapdoor shares by jointly running (n,t)-BlindTrapdoor with $\mathcal{TGE}s$, once combined and unblinded she obtains a trapdoor that matches a searchable encryption. Then she runs algorithm Test$(S_{W,K}, T_{\hat{W}})$ and returns the symmetric key K that allows her to decrypt the record associated with the searchable encryption $S_{W,K}$.

Notice that it is possible to separate the ability of searching for data records described by a keyword and the ability of decryption to allow better properties like delegation of search to a third party. To do this, we can proceed as follows: first, we omit the private-key K from the algorithm TPEKS$(\mathcal{GP}, W, K)$ and add an encryption of K encrypted under the public key belonging to the $\mathcal{TGE}s$. Test algorithm will return only true/false response whether or not a given data record is described by a keyword, and the investigator should submit the encrypted key K to $\mathcal{TGE}s$ which act as threshold decryption servers. $\mathcal{TGE}s$ return decryption shares to the investigator, the latter combines the obtained shares to compute the symmetric key K and proceeds to the decryption of the data record.

6 Conclusion

In this paper, we have presented a new searchable encryption scheme, namely threshold public-key encryption with oblivious keyword search TPEOKS. Unlike all previous schemes, our solution presents the advantage of allowing users to privately search by keywords on encrypted data and at the same time prevents the Trapdoor Generation Entity $\mathcal{TGE}$ from generating trapdoors on behalf of a genuine user, which was an issue until now. In particular, we have used an approach that merges together TPEKS [16] and PEOKS [10] in one framework.

In order to implement TPEOKS, we have defined blind anonymous (n, t)-IBE schemes and provided a generic transformation from an anonymous (n, t)-IBE into a blind anonymous (n, t)-IBE. We have also developed a concrete construction of such schemes along with the security analysis in the standalone settings and we discussed the extension of the developed protocol to handle concurrent executions. Next, we have showed how to transform a blind anonymous (n, t)-IBE into TPEOKS generically. Finally, we described a system architecture for real world application of TPEOKS.

During this work, we carefully justified all our design choices along the way, in particular the security model that allows composition of protocols and the IBE scheme considered, other schemes maybe considered in a future work. Furthermore, we will focus on new constructions in the UC framework since we already discussed how to extend our proposed protocol to this framework.

References

1. 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)
2. 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)
3. Abdalla, M., Boyen, X., Chevalier, C., Pointcheval, D.: Distributed public-key cryptography from weak secrets. In: S. Jarecki and G. Tsudik (Eds.): PKC '09. LNCS, vol. 5443, pp. 139–159, Springer-Verlag, (2009)
4. Geisler, M., Smart N.:Distributing the Key Distribution Centre in SakaiKasahara Based Systems. In: M.G. Parker (Eds.): Cryptography and Coding 2009, LNCS 5921, pp. 252262, 2009. Springer-Verlag Berlin Heidelberg (2009)
5. Boyen, X., Chevalier, C., Fuchsbauer, G., Pointcheval, D. :Strong cryptography from weak secrets : Building efficient pke and ibe from distributed passwords. In: Bernstein, Daniel and Lange, Tanja (Ed.): Africacrypt '10, volume 6055 of LNCS, pages 297–315. Springer-Verlag Berlin Heidelberg, (2010)
6. Kate, A., Goldberg I.:Distributed Private-Key Generators for Identity-Based Cryptography. In: J.A. Garay and R. De Prisco (Eds.): SCN 2010, LNCS 6280, pp. 436453, 2010. Springer-Verlag Berlin Heidelberg (2010)
7. Belenkiy, M., Chase, M., Kohlweiss, M., Lysyanskaya, A. : P-signatures and noninteractive anonymous credentials. In R. Canetti, editor, TCC. LNCS, vol. 4948, pp. 356-374. Springer, (2008)

8. Belenkiy, M., Chase, M., Kohlweiss, M., Lysyanskaya, A., Shacham, H. :Randomizable Proofs and Delegatable Anonymous Credentials. In Crypto 09, volume 5677 of LNCS, pp. 108–125. Springer, (2009)
9. Waters, B., Balfanz, D., Durfee, G., Smetters, D.: Building an Encrypted and Searchable Audit Log. In: Proc. of Network and Distributed System Security Symposium (NDSS 2004) (2004)
10. Camenisch, J., Kohlweiss M., Rial A., Sheedy C.: Blind and Anonymous Identity-Based Encryption and Authorised Private Searches on Public Key Encrypted Data. In:S. Jarecki and G. Tsudik (Eds.): PKC 2009, LNCS 5443, pp. 196-214, (2009)
11. Chow, S. M. : Removing Escrow from Identity-Based Encryption New Security Notions and Key Management Techniques. In: Jarecki, Stanislaw and Tsudik, Gene (Eds.): PKC 2009, LNCS 5443, pp. 256-276, 2009. Springer-Verlag Berlin Heidelberg (2009)
12. Wang,P., Wang, H., Pieprzyk J.: Threshold Privacy Preserving Keyword Searches. In: V. Geffert et al. (Eds.): SOFSEM 2008, LNCS 4910, pp. 646-658, 2008. Springer-Verlag Berlin Heidelberg (2008)
13. Canetti, R. : Security and Composition of Multiparty Cryptographic Protocols. In : the Journal of Cryptology, 13(1):143-202, (2000)
14. Canetti, R. : Universally composable security : a new paradigm for cryptographic protocols. In FOCS, pages 136–145, (2001)
15. Boyen, X., Waters, B.: Anonymous hierarchical identity-based encryption (without random oracles). In: Dwork, C. (ed.) CRYPTO 2006. LNCS, vol. 4117, pp. 290307. Springer, Heidelberg (2006)
16. Siad, A.: Anonymous Identity-Based encryption with distributed Private-Key generator and searchable encryption. In NTMS'2012 - Security Track, (2012)
17. 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)
18. Y. Lindell and B. Pinkas. Secure multiparty computation for privacy-preserving data mining. Journal of Privacy and Confidentiality, 1(1), p. 59–98, (2009)
19. Gentry, C.: Practical identity-based encryption without random oracles. In: Vaudenay, S. (ed.) EUROCRYPT 2006. LNCS, vol. 4004, pp. 445-464. Springer, Heidelberg (2006)
20. Siad, A., Amara, M.: Anonymous Key Issuing Protocol for Distributed Sakai-Kasahara Identity-based Scheme. In ICN 2011, pages 35–40, (2011)
21. Prabhakaran, M., Rosulek, M. : Homomorphic Encryption with Chosen-Ciphertext Security. In: Aceto, L., Damgrd, I., Goldberg, L., Halldrsson, M., Inglfsdttir, A., Walukiewicz, I. (eds.): AUTOMATA, LANGUAGES AND PROGRAMMING . LNCS, vol. 5126, pp. 667–678. Springer, Heidelberg (2008)
22. Paillier, P. : Public-key cryptosystems based on composite degree residuosity classes. In EUROCRYPT, LNCS, vol. 1592, pages 223–238, (1999)
23. Green, M., Hohenberger, S.: Blind identity-based encryption and simulatable oblivious transfer. In: Kurosawa, K. (eds.): ASIACRYPT 2007. LNCS, vol. 4833, pp. 265-282. Springer, Heidelberg (2007)
24. Goldreich, O.: Foundations of Cryptography: Volume 2 Basic Applications. Cambridge University Press (2004)
25. Damgard, I., Nielsen, J. B.: Perfect hiding and perfect binding universally composable commitment schemes with constant expansion factor. In Yung, M. (eds.): Advances in Cryptology CRYPTO 02, LNCS, vol. 2442, pp. 3–42, (2002)