Biometric Identification over Encrypted Data Made Feasible

Michael Adjedj[1,2], Julien Bringer[1], Hervé Chabanne[1,3], and Bruno Kindarji[1,3]

[1] Sagem Sécurité, Osny, France
[2] Université Bordeaux I, UFR de Mathématiques, Bordeaux, France
[3] Institut Telecom, Telecom ParisTech, Paris, France

Abstract. Realising a biometric identification scheme with the constraint of storing only encrypted data is an exciting challenge. Whereas a recent cryptographic primitive described by Bringer *et al.* and named Error-Tolerant Searchable Encryption achieves such a goal, the associated construction is not scalable to large databases. This paper shows how to move away from the model of Bringer *et al.*, and proposes to use Symmetric Searchable Encryption (SSE) as the baseline for biometric identification. The use of symmetric cryptography enables to achieve reasonable computational costs for each identification request.

This paper also provides a realistic security model for this problem, which is stronger than the one for SSE. In particular, the construction for biometric identification is resilient to statistical attacks, an aspect yet to be considered in the previous constructions of SSE.

As a practical example, parameters for the realisation of our scheme are provided in the case of iris recognition.

Keywords: Identification, Biometrics, Searchable Encryption.

1 Introduction

Biometric recognition systems are based on the unicity of some biological trait every human being carries along. For instance, it is possible to verify if a given individual is the one he claims to be (*Authentication*), or to find someone's identity using his biometrics (*Identification*).

In most cases, identification is done by comparing a new acquisition of the biometric trait (a biometric **template**) with a database that is stored on a server. The server can be outsourced, and we do not want it to learn more information than it ought to. If the database is not encrypted, or if the server first decrypts the data before comparing the traits, this leads to privacy leakage, as the server then learns personal information on the people that use the biometric system.

One of the inherent problems of working with some biometric templates is their fuzziness. Two captures by the same sensor, of the same biometric trait of the same person, are in most cases significantly different. The standard way to deal with this fuzziness is to use a matching function, which is able to tell whether two templates come from the same biometric trait.

Using only traditional matching algorithms, the server has to execute $\mathcal{O}(N)$ matching comparisons to find a match among N biometric samples. This is infeasible in a satisfactory computation time for large databases. Unfortunately, matching algorithms that compare encrypted templates are rare – and expensive.

Moreover, doing an identification implies to look for the template among a collection that is the closest to the one presented at a sensor. Instead of doing all the traditional operations in the encrypted domain, we choose to do both the matching and the search at the same time. Combining encryption with search capabilities is a cryptographic primitive called **searchable encryption**.

To the best of our knowledge, the only construction that achieves biometric identification with encrypted biometric data is [5,6]. However the privacy of this construction is based on the use of a Private Information Retrieval protocol [8] and asymmetric cryptography; and such a protocol's computational complexity is always (at least) linear in the size of the database. To avoid this pitfall, we focus in this paper on symmetric cryptography.

Recent works on Symmetric Searchable Encryption [2,3,7,9,12,17] provide schemes with constant-time access to servers; the price to pay is a leakage of the **search pattern**: the server can tell whether a word was queried twice and can even recover links between documents and words. This enables to infer relations between requests and for instance to determine, after a statistical survey, the queried word. We formalize this advantage in the adversarial model stated in Section 4.2. In particular, Condition 3 is a barrier to statistical attacks. To cope with this classical weakness, we introduce a way to protect the access pattern on the server's side.

This paper solves the issue of preserving privacy in a biometric identification system. As opposed to previous work, the computational cost for this purpose is quite low, making use of recent advances in the fields of similarity searching and secure querying on a remote server. In the end, we perform biometric identification over a wholly encrypted database, in such a way that the server does not have an advantage over the users' privacy.

2 Basic Concepts

For a given biometric, let B be the set of all possible biometric features, *i.e.* data captured by a biometric sensor. Consider a user $\mathcal{U}$, his biometric trait is noted β. From a measurement of β using a sensor, the so-called **biometric template** is computed after feature extraction and is noted b ($b \in B$). A matching algorithm is a function $m : B \times B \to \mathbb{R}$, which computes a similarity score between two templates. Let b and b' be the results of two measurements from the same biometric. Then, with a high probability, their matching score $m(b, b')$ is **small**. We say that b and b' constitute a **matching pair**. Otherwise, when they are from different biometrics, with a high probability, their matching score is **large**. In practice, some thresholds are chosen λ_{min}, λ_{max} and the score is considered **small** (resp. **large**) if it is smaller (resp. greater) than λ_{min} (resp. λ_{max}). Depending on the values fixed for λ_{min} and λ_{max}, errors eventually occur: 1. The system declares two templates obtained from different users as a matching pair; this is called a False Acceptance (**FA**). 2. The system states that two templates extracted from the same user do not match; this is the False Reject case (**FR**).

At registration, a user chooses a pseudonym, also called an **identity**. A **biometric identification system** recognizes a person among others. On input b_{new}, the system returns a set of identities (corresponding to a templates set $\{b_{ref}\}$), such that all matching scores between b_{new} and the b_{ref}'s are small. This means that b_{new} and b_{ref} possibly correspond to the same person.

We restrict ourselves to the case where biometric templates are in the Hamming space $B = \{0,1\}^n$ with the Hamming distance d (e.g. IrisCode [10]). Two templates b, b' of a same user $\mathcal{U}$ are with a high probability at distance $d(b,b') < \lambda_{min}$. Similarly, when b and b' comes from different users, they are with a high probability at distance $d(b,b') > \lambda_{max}$. In this case, and in the rest of the paper, the matching algorithm consists in evaluating a Hamming distance.

3 Useful Tools

3.1 Locality-Sensitive Hashing

For two different inputs with a small matching score, *i.e.* close in the sense of the Hamming distance, Locality-Sensitive Hash families output, with a high probability, the same value. We use them to decrease disparities between two similar templates.

Definition 1 (Locality-Sensitive Hashing [14]). *Let (B, d_B) be a metric space, U a set of smaller dimensionality. Let $r_1, r_2 \in \mathbb{R}$, $p_1, p_2 \in [0,1]$ such that $p_1 > p_2$.*

*A family $H = \{h_1, \ldots, h_\mu\}$, $h_i : B \to U$ is (r_1, r_2, p_1, p_2)-**LSH**, if*

$$\forall h \in H,\ x, x' \in B \begin{cases} Pr[h(x) = h(x') \,|\, d_B(x,x') < r_1] > p_1 \\ Pr[h(x) = h(x') \,|\, d_B(x,x') > r_2] < p_2 \end{cases}$$

Some examples of LSH families are given in [5,13,14]. Another example of practical use is given in Section 5.1.

Remark 1. With regard to Definition 1, LSH hash functions have no cryptographic property. *Per se*, the security of our construction does not rely on the use of LSH functions.

3.2 Symmetric Searchable Encryption – SSE

Searchable Encryption is described as follows.

- A client $\mathcal{U}$ has a collection of documents consisting of sequences of words.
- He encrypts the whole collection along with some indexing data.
- He stores the result on a (remote) server.

The server should be able to return all documents which contain a particular keyword, without learning anything about the aforementioned keyword.

Let $\Delta = \{\omega_1, \cdots, \omega_d\}$ be the set of d distinct words (typically a dictionnary). A *document* $D \in \Delta^*$ is a sequence of words of Δ. The *identifier* $\mathtt{id}(D)$ is a bitstring that uniquely identifies the document D (e.g. its memory address). A *collection* $\mathcal{D} = (D_1, \cdots, D_n)$ is a set of n documents. $\mathcal{D}(\omega)$ denotes the lexicographically ordered list of identifiers of documents which contains the word ω.

For efficiency reasons, we only focus on the **symmetric** searchable encryption paradigm.

Definition 2 (Symmetric Searchable Encryption Scheme [9]). *A Symmetric Searchable Encryption scheme is a collection of four polynomial-time algorithms* `Keygen`, `BuildIndex`, `Trapdoor`, `Search` *such that:*

`Keygen` *(1^ℓ) is a probabilistic key generation algorithm, run by the client to setup the scheme. It takes a security parameter ℓ and returns a secret key K.*

`BuildIndex` *$(K, \mathcal{D})$ is a (possibly probabilistic) algorithm run by the client to compute the index $\mathcal{I}_\mathcal{D}$ of the collection $\mathcal{D}$. It takes as entry a secret key K and a collection of documents $\mathcal{D}$. The index returned allows the server to search for any keyword appearing in $\mathcal{D}$.*

`Trapdoor` *(K, ω) is a deterministic algorithm which generates a trapdoor T_ω for a given word ω under the secret key K. It is perfomed by the client whenever he wants to search securely for all the documents where ω occurs.*

`Search` *$(\mathcal{I}_\mathcal{D}, T_w)$ is run by the server to search in the entire collection $\mathcal{D}$ for all the documents identifiers where the queried word ω appears. It returns $\mathcal{D}(\omega)$.*

These primitives give a functional aspect of what Symmetric Searchable Encryption provides. The associated security model is described in [9], and briefly depicted in Appendix A.1. The goal is to achieve *Adaptive Indistinguishability*, a security property stating that an adversary does not get information on the content of the registered documents. More precisely, if two different collections are registered, with constraints on the number of words per document, an adversary cannot distinguish between two sequences of search requests.

Remark 2. A noticeable construction of a scheme adaptively indistinguishable was also provided in [9] (cf. Appendix A.2), and inspired our identification data structure. Although this scheme is proved secure in their model, this does not cover statistical attacks where an adversary tries to break the confidentiality of the documents or the words based on statistics about the queried words and the index (cf. Remark 5).

4 Fast and Secure Biometric Identification

Our construction does not simply mix a SSE scheme with a LSH family. Indeed, we ensure the security of our biometric identification protocol against statistical attacks, which is an improvement with respect to a direct combination of SSE with LSH.

4.1 Our Idea in a Nutshell

Our Biometric Identification process has two phases: a **search phase** which carries out every request on the database $\mathcal{DB}$ and sends back to the sensor client $\mathcal{SC}$ the search result, an **identification phase** which treats data extracted from search results to proceed to the identification. The search phase is constructed following the principle of the SSE scheme from [9]. The following entities interact:

- Human users $\mathcal{U}_i$: a set of N users who register their biometrics.
- Sensor client $\mathcal{SC}$: a device that captures the biometric data and extracts its characteristics to output the biometric template. It also sends queries to the server to identify a user.
- The server: replies to queries sent by $\mathcal{SC}$ by providing a set of search results and owns a database $\mathcal{DB}$ to store the data related to the registered users.

Remark 3. We consider that $\mathcal{SC}$ is honest and trusted by all other components. In particular, $\mathcal{SC}$ is the only entity which is in possession of the cryptographic key material used in the protocol. To justify this assumption, we emphasize that the object of this paper is to provide a solution to the secure storage of reference templates, but not to provide an end-to-end architecture. See Remark 6 for details on key management.

We provide the three following methods:

1. `Initialize`(1^ℓ): It produces the parameters $\mathcal{K}$ of the system, according to a security parameter ℓ. $\mathcal{K}$ must contain secret keys sk used to encrypt the identities, and K used in the SSE scheme.
2. `Enrolment`$(b_1, \ldots, b_N, ID_1, \ldots, ID_N, \mathcal{K})$: It registers a set of users with their biometric characteristics. For a user $\mathcal{U}_i$, it needs a biometric sample b_i and his identity ID_i. This returns an index $\mathcal{I}$.
3. `Identification`$(\mathcal{K}, b)$: It takes as input a newly captured template b and it returns a set of identities for which the associated templates are close to b. See Conditions 1 and 2, Section 4.2.

Definition 3. *In our proposal,* keywords *are evaluations of LSH functions on templates, concatenated with the index of the considered function, i.e.* $h_i(b)||i$, *for* $i \in [1, \mu]$ *where* b *is the captured template of a user.*

Identifiers *are the encryptions of the* identities *of the registered users. We have,* $\mathtt{id}(\mathcal{U}_i) = \mathcal{E}_{sk}(ID_i)$ for $i \in [1, N]$ *where* $\mathcal{E}_{sk}$ *is an encryption function with the secret key* sk*, and* ID_i *is the identity of the user* $\mathcal{U}_i$.

The interaction between the server and $\mathcal{SC}$ defines the identification view, required for the security experiments. It consists of the encrypted identities of the registered users, and informations sent by $\mathcal{SC}$ when a user $\mathcal{U}$ is being identified.

Definition 4 (Identification View). *The* identification view *under the secret keys* K *and* sk *is defined as*

$$IdV_{K,sk}(b') = (\mathcal{I}, T_{h_1(b')||1}, \ldots, T_{h_\mu(b')||\mu}, \mathcal{E}_{sk}(ID_1), \ldots, \mathcal{E}_{sk}(ID_N))$$

where b' is a freshly captured template from $\mathcal{U}$.

4.2 Security Requirements

We assume that the Hamming space $B = \{0,1\}^n$ is such that $n \geq \ell$, where ℓ is the security parameter. A function f is said to be negligible if for all non-constant polynomial P, and for all sufficiently large k, we have $f(k) < \frac{1}{|P(k)|}$. In the sequel, a probability is negligible if it is negligible in ℓ.

First of all it is important that the scheme actually works, *i.e.* that the retrieval of the identity of a registered user gives the correct result. This can be formalized by the two following conditions.

Condition 1 (Completeness). *The system is* complete *if for all $b' \in B$, the result of* `Identification`(b') *contains the set of identities for which the associated templates b_i are close to b' (ie. $d(b', b_i) < \lambda_{min}$), except for a negligible probability.*

Condition 2 (Soundness). *The system is sound if, for each template b' such that $d(b', b_i) > \lambda_{max}$,* `Identification`$(b')$ *is the empty set $\emptyset$, except with negligible probability.*

To avoid statistical attacks, we do not want the database to infer relations between different identities. This is modeled by the following condition.

Condition 3 (Adaptive Confidentiality). *An identification system achieves adaptive confidentiality if the advantage $\mathtt{Adv}_{\mathcal{A}} = |\Pr(b_0 = b_0') - \frac{1}{|B|}|$ of any polynomial-time adaptive adversary is negligible in the next experiment, where $\mathcal{A} = (\mathcal{A}_1, \mathcal{A}_2)$ is an opponent taking the place of the server, and $\mathcal{C}$ is a challenger at $\mathcal{SC}$'s side.*

$$\begin{array}{|lll}
1.\ \mathcal{K} & \overset{R}{\longleftarrow} \mathtt{Initialize}(1^\ell) & (\mathcal{C}) \\
2.\ b_1, \ldots, b_N & \longleftarrow B & (\mathcal{A}) \\
3.\ \mathcal{I}_1 & \longleftarrow \mathtt{Enrolment}(b_1, \ldots, b_N) & (\mathcal{C}) \\
4.\ b, IdV_{K,sk}(b) & \longleftarrow \mathcal{A}_1^{\mathtt{Identification}}(\mathcal{I}_1) & (\mathcal{A}) \\
5.\ b_0 & \overset{R}{\longleftarrow} B & (\mathcal{C}) \\
\quad \textit{such that } \forall i \in [1,N], d(b_0, b_i) > \lambda_{max} & & \\
6.\ \mathcal{I}_2 & \longleftarrow \mathtt{Enrolment}(b_0, b_1, \ldots, b_N) & \\
6'.\ b, IdV_{K,sk}(b) & \longleftarrow \mathcal{A}_1^{\mathtt{Identification}}(\mathcal{I}_1, \mathcal{I}_2) & (\mathcal{A}) \\
7.\ b_0' & \longleftarrow \mathcal{A}_2(\mathcal{I}_1, \mathcal{I}_2, b, IdV_{K,sk}(b), IdV_{K,sk}(b_0)) &
\end{array}$$

`Enrolment`$(b_1, \ldots, b_N)$ *stands for* `Enrolment`$(b_1, \ldots, b_N, ID_1, \ldots, ID_N, K, sk)$.

In this game, the attacker is allowed to set a templates database $b_1, \ldots, b_N$ of its choice (2.). Then the challenger creates the database by enrolling the whole collection (3.), and the adversary can make a polynomial number of identifications (using the method `Identification`) of the templates of his choice (4.). The challenger then picks a random template b_0 (5.) and it recreates the database $\mathcal{I}_2$ (6.). The attacker is allowed once again to make a polynomial number of identifications from the templates of its choice (6′.) and he is challenged to retrieve the initial template b_0 (7.), given the knowledge of $\mathcal{I}_1, \mathcal{I}_2$, and the views of the identifications.

This condition expresses the confidentiality of the enrolled templates, even if the adversary has access to the index and to identification views, which may give him the possibility to construct a statistical model on it.

Condition 4 (Non-adaptive Indistinguishability). *We say that a biometric identification system achieves indistinguishability if the advantage* $\mathtt{Adv}_{\mathcal{A}} = |\Pr(e = e') - \frac{1}{2}|$ *of any polynomial-time adversary* $\mathcal{A} = (\mathcal{A}_1, \mathcal{A}_2)$ *is negligible in the following game:*

$$\left|\begin{array}{lll} 1.\ b_1, \ldots, b_N & \overset{R}{\longleftarrow} B & (\mathcal{C}) \\ 2.\ b^{(0)}, b^{(1)} & \longleftarrow \mathcal{A}_1(\mathcal{C}(IdV_{K,sk})) & (\mathcal{A}) \\ 3.\ e & \overset{R}{\longleftarrow} \{0,1\} & (\mathcal{C}) \\ 4.\ e' & \longleftarrow \mathcal{A}_2(IdV_{K,sk}(b^{(e)})) & (\mathcal{A}) \end{array}\right.$$

where $\mathcal{A}_1(\mathcal{C}(IdV_{K,sk}))$ *stands for the fact that the adversary accesses to the identification view produced when* $\mathcal{C}$ *executes a polynomial number of identification requests, without knowing the input randomly chosen by the challenger.*

This experiment is executed as follows: The challenger first creates a set of templates $b_1, \ldots, b_N$ (1.), and executes a polynomial number of identification requests. The adversary has access to all the identification views (2.). The attacker then chooses two templates for which he believes he has an advantage (2.), and the challenger picks at random one of them and executes its identification (3.). The attacker is finally challenged to determine which template the challenger chose (4.).

4.3 Our Identification Protocol

$\underline{\mathtt{Initialize}}(1^\ell)$:

- We choose an IND-CPA symmetric encryption scheme $(\mathcal{G}, \mathcal{E}, \mathcal{D})$.
- We use the Symmetric Searchable Encryption scheme from [9] (see Appendix A.2 for the construction detail) out of which we pick the functions (`Keygen`, `Trapdoor`, `Search`) and adapt them to our needs.
- We fix a threshold $0 < \lambda \leq \frac{1}{2}$.
- Let $H = (h_1, \ldots, h_\mu)$ be a $(\lambda_{min}, \lambda_{max}, p_1, p_2)$- LSH family, $\mu \geq \ell$.
- Let $K = \mathtt{KeyGen}(1^\ell)$, and $sk = \mathcal{G}(1^\ell)$.
- Let π_K be the pseudo-random permutation indexed by the key K used in the SSE scheme.

Output $\mathcal{K} = (h_1, \ldots, h_\mu, K, sk, \lambda)$.

$\underline{\mathtt{Enrolment}}(b_1, \ldots, b_N, ID_1, \ldots, ID_N, \mathcal{K})$: Consider N users $\mathcal{U}_1, \ldots, \mathcal{U}_N$ to be enrolled. Their template are denoted by b_i, and their identity ID_i, $i \in [1, N]$. We recall that in our construction, the words we consider are the $h_i(b)||i$, $i \in$

```
Enrolment($b_1, \ldots, b_N, ID_1, \ldots, ID_N, \mathcal{K}$):

  • Initialization:
      - build $\Delta = \{h_i(b_k)||i; \quad i \in [1, \mu],\ k \in [1, N]\}$
      - for each $\omega_{i_k} \in \Delta$, build $\mathcal{D}(\omega_{i_k})$ the set of identifiers of users
        $\mathcal{U}_{k'}$ such that $h_i(b_{k'})||i = \omega_{i_k}$
      - compute $\mathtt{max} = \max_{\omega \in \Delta}(|\mathcal{D}(\omega)|)$ and $m = \mathtt{max} \cdot |\Delta|$
  • Build look-up table T:
      - for each $\omega_i \in \Delta$
            for $1 \leq j \leq |\mathcal{D}(\omega_i)|$
                set T $[\pi_K(\omega_i \parallel j)] = \mathcal{E}_{sk}(ID_{i,j})$
      - if $m' = \sum_{\omega_i \in \Delta} |\mathcal{D}(\omega_i)| < m$ ,then set the remaining $(m - m')$
        entries of T to random values.
  • Output $\mathcal{I} = \mathtt{T}$
```

Fig. 1. Enrolment Procedure

$[1, \mu],\ b \in B$, where h_i is one of the chosen LSH functions, and b is a reference template from a registered user.

We alter the `BuildIndex` algorithm of the SSE scheme into `Enrolment` to take into account the need for identification (cf. Figure 1).

Remark 4. Our scheme stores identifiers encrypted by an IND-CPA scheme so that no relation between the entries could be found by observing the index $\mathcal{I}$. This prevents inferring statistics from the $\mathcal{DB}$ content. Proposition 3 formalizes this intuition.

`Identification`($\mathcal{K}, b'$):
Search phase: When a user $\mathcal{U}$ wants to be identified, $\mathcal{SC}$ captures his biometric trait in a template b'. $\mathcal{SC}$ evaluates each LSH function on b' to compute $\omega_{k_i} = h_i(b')||i,\ i \in [1, \mu]$ and sends to the server the trapdoors:

$$T_{\omega_{k_i}} = \mathtt{Trapdoor}(K, \omega_{k_i}) = (\pi_K(\omega_{k_i}||1), \ldots, \pi_K(\omega_{k_i}||\max))$$

The server executes the `Search` algorithm on the different trapdoors $T_{\omega_{k_i}}$ – each call to $\mathtt{Search}(t_1, \ldots, t_{\max})$ returns $\mathtt{T}[t_1], \ldots \mathtt{T}[t_{\max}]$ – and sends to $\mathcal{SC}$ the array $\mathcal{ID}(b')$ which corresponds to all the search results:

$$\mathcal{ID}(b') = \begin{bmatrix} \mathcal{E}_{sk}(ID_{k_1,1}) & \cdots & \mathcal{E}_{sk}(ID_{k_1,\mathtt{max}}) \\ \vdots & \ddots & \vdots \\ \mathcal{E}_{sk}(ID_{k_\mu,1}) & \cdots & \mathcal{E}_{sk}(ID_{k_\mu,\mathtt{max}}) \end{bmatrix}$$

where each row is made of the output of $\mathtt{Search}(T_{\omega_{k_i}})$. It may happen that a virtual address $\pi_K\big(h_i(b')||i||j\big)$ is invalid, in this case the server sends `NULL` instead of an identifier.

Identification phase: $\mathcal{SC}$ decrypts all received identifiers and determines the number of occurrences of each identity to output the list of the ones that appear more than $\lambda\mu$ times, *i.e.* the list of identities $\{ID(\mathcal{U}_l)\}$ that verify this inequality: $\sum_{i=1}^{\mu}\sum_{j=1}^{\mathtt{max}} \mathbb{1}_{ID(\mathcal{U}_l)}\left(ID_{k_i,j}\right) > \lambda\mu$, where $\mathbb{1}$ is the indicator function. If the result is still ambiguous after that the identity that appeared the most was selected, an empirical rule is applied.

4.4 Security Properties

We use Shannon's entropy, $\mathcal{H}_2(\lambda) = \lambda \cdot log\frac{1}{\lambda} + (1-\lambda)\cdot log\frac{1}{1-\lambda}$

Proposition 1 (Completeness). *Provided that H is a $(\lambda_{min}, \lambda_{max}, p_1, p_2)$-LSH family, for $1 - p_1 \leq \frac{1}{4^{\mathcal{H}_2(\lambda)+c}}$, with $c \geq 1$, our scheme is complete.*

Proposition 2 (Soundness). *Provided that H is a $(\lambda_{min}, \lambda_{max}, p_1, p_2)$-LSH family, for $p_2 \leq \frac{1}{2^{\frac{1}{\lambda}+c}}$, with $c \geq 1$, our scheme is sound.*

The proofs of Propositions 1 and 2 are given in Appendix B. The underlying idea is that computing the μ LSH functions separates the close and the distant template pairs.

Proposition 3 (Adaptive Confidentiality). *Provided that the underlying encryption scheme $(\mathcal{G}, \mathcal{E}, \mathcal{D})$ is a IND-CPA secure scheme, our construction ensures the templates confidentiality.*

Sketch of Proof. The adversary $\mathcal{A}$ is allowed to execute some identification requests. If $\mathcal{A}$ is able to reconstruct the template b_0, then he can infer links between the enrolled b_i and the associated identification result $\mathcal{ID}(b_0)$.

Due to the IND-CPA security of $(\mathcal{G}, \mathcal{E}, \mathcal{D})$, a simulator can simulate the array $\mathcal{ID}(b)$ during the second enrolment phase in the following way: when it receives for the first time a set of trapdoors $\{T_{h_1(b||1)}, \ldots, T_{h_\mu(b||\mu)}\}$, for a template b, it picks up a random array of size $\mu \cdot \mathtt{max}$ and stores the correspondence between the trapdoors and this array. When the adversary sends the same trapdoors, the same result is sent back by the simulator. This way, an adversary who can make links between informations contained in the array $\mathcal{ID}(b)$, can also infer links between random identifiers, which is impossible. Thus the property. $\square$

Proposition 4 (Non-Adaptive Indistinguishability). *Provided that π_K is a pseudo-random permutation depending on a secret key K, and that $(\mathcal{G}, \mathcal{E}, \mathcal{D})$ is semantically secure, our construction ensures the non-adaptive indistinguishability.*

Sketch of Proof. This property is mainly a consequence of the semantic security of the SSE scheme we consider. Indeed, for π_K a pseudo-random permutation, a simulator can simulate the trapdoors sent by the sensor client during an identification, and it can also simulate the server's response because of the semantic security of the symmetric encryption scheme used. $\square$

Remark 5. We emphasize that the aforementioned properties define an adequate description of what resistance against statistical attacks would be.

A scheme, that would be no more than a combination of the SSE scheme described in [9] with the use of LSH functions, would not be resistant against these methods. An adversary is, in that setting, able to retrieve – and compare – the identifiers of the users enrolled, and thus infer knowledge on the identity of a user that did not proceed to identification.

Similarly, if the identifiers are not encrypted, an attacker who observes the views of identification can gather statistics on the identification of the different users. This enables him to link the identity of users - as some are more likely to be identified than others - with the response of the server. Moreover, he can manage a very general statistical attack in that case: by learning the relation between identities and keywords (i.e. LSH values of biometric data), he can even reconstruct unknown templates.

Note that our technique to thwart statistical attacks is quite general and can be reused in other contexts.

5 Practical Considerations

5.1 Choosing a LSH Family

To explain that our scheme meets the usual needs for a practical deployment of a biometric identification system, let us consider the case of biometric iris recognition as a practical example. A well-known method to represent iris is to use Daugman's IrisCode [10] algorithm. The outcome is a 4096-bit vector, where half of them carry information held in the iris, and the other part is a "mask" of the significant bits.

In [13], a method to increase the efficiency of iris identification is described. By applying specific projection functions into $\{0,1\}^{10}$, the search among an iris dataset can be accelerated. The projected values of an iriscode serve in a pre-filtering step. Experiments successfully reported in [13] on a UAE database, containing $N = 632500$ iris records, show that it decreases the number of necessary matching to 41 instead of N to achieve identification. The authors determine a biometric template b as a candidate for b' if b and b' give the same evaluations with at least three functions; see [13] for more details. Their functions can be in fact considered as a $(\lambda_{min}, \lambda_{max}, (1-\frac{\lambda_{min}}{2048})^{10}, (1-\frac{\lambda_{max}}{2048})^{10})$-LSH family and we can thus apply our construction without degrading their results. This shows the actual practicality of our scheme.

The family is made by $\mu = 128$ hash functions, which each extracts a 10-bit vector. Our parameter λ can be set to $\lambda = \frac{3}{128}$. According to traditional matching algorithms, we can choose $\lambda_{min} = 0.25 \cdot 2048 = 512$ and $\lambda_{max} = 0.35 \cdot 2048 = 716.8$, which gives the probabilities $p_1 \simeq 0.056$ and $p_2 \simeq 0.013$ (with the notations of Definition 1). The probability of `Identification`(b') *not* returning a template close to b' is given by $\sum_{i=0}^{\lfloor \lambda\mu \rfloor} \binom{\mu}{i} p_1^i \left(1-p_1\right)^{\mu-i} \simeq 0.066$ and the other probability to consider is, for b' far from all the b_i, $\Pr[$`Identification`$(b') \neq$

$\emptyset] = \sum_{i=\lceil \lambda\mu \rceil}^{\mu} \binom{\mu}{i} p_2^i (1-p_2)^{\mu-i} \simeq 0.095$. Note that those probabilities are small, and not negligible, but they can be considered attractive for practical uses (as asserted by the results from [13]).

5.2 Implementation

To check further the feasibility of our scheme, we implemented our scheme and conduced a first empirical evaluation on the ICE 2005 database [15,16] which contains 2953 images from 244 different eyes. The results are similar to those deduced in the previous section from the results of [13]. For instance, the probability that the genuine identity is not in the outputted list of candidates is below 10%.

Remark 6. In addition to this performance consideration, it is important to notice that the deployment of the scheme is quite simple as only the client needs to know the secret keys. So management of the keys is reduced to a distribution to the clients that are allowed to run identification requests onto the remote server.

For a similar reason, the scheme only uses classical cryptographic schemes; therefore it does not suffer of the same weaknesses [1] as some other biometric protection schemes.

5.3 Complexity

We here evaluate the computational complexity of an identification request on the server's side as well as on $\mathcal{SC}$. We note $\kappa(op)$ the cost of operation *op*.

- On the server's side: assuming that we organize the look-up table in a FKS dictionnary [11], a search is made in constant time and the server has μ searches to achieve.
- On $\mathcal{SC}$'s side:

$$\begin{aligned}\kappa(identification) &= \kappa(trapdoors) + \kappa(count)\\ &= \mu.\mathtt{max}.\left[\kappa(hash) + \kappa(encryption) + \kappa(decryption)\right]\end{aligned}$$

  $\kappa(hash)$ is the computational complexity to evaluate a LSH function, and $\kappa(encryption)$ is the one to apply the pseudo-random permutation π_K. The final count needs to compute the number of occurences of each identity, it can be made in computation time linear in the size of the final array, hence the term $\mu.\mathtt{max}.\kappa(decryption)$ (remember that before counting, $\mathcal{SC}$ has to decrypt the search results). If the chosen hash functions map $\{0,1\}^*$ to $\{0,1\}^m$ (for $m \in \mathbb{N}^*$) and assuming that images of these functions are equally distributed, the max value can be bounded by $\mathcal{O}(\frac{N}{2^m})$, where N is the number of registered users. So the overall complexity is $\mathcal{O}\left(\mu\frac{N}{2^m}\right) \cdot [\kappa(hash) + \kappa(encryption) + \kappa(decryption)]$. A traditional identification algorithm would cost $\mathcal{O}(N)$ matching operations; with the parameters given in Section 5.1, our solution is 8 times more efficient, with the additional benefit of the encryption of the data.

Remark 7. The complexity of the construction initially proposed in [6] was globally the same at the client level (modulo the use of asymmetric cryptography rather than symmetric schemes in our case). It consists in computing the LSH images of the freshly acquired template, and in preparing μ PIR queries associated to the hashes. While this computation is costly, it is still doable in reasonable time. However on the server side, $\mathcal{S}$ must compute the PIR replies, and cannot do it in less than a linear time in the database's size (2^m). Indeed, no matter what PIR scheme is used, $\mathcal{S}$ always needs to process the whole database before sending its reply; here we enable secure biometric identification with only μ constant-time operations at $\mathcal{S}$'s side.

Acknowledgements

The authors thank the referees for their comments. This work is partially supported by funding under the Seventh Research Framework Programme of the European Union, Project TURBINE (ICT-2007-216339).

References

1. Ballard, L., Kamara, S., Reiter, M.K.: The practical subtleties of biometric key generation. In: van Oorschot, P.C. (ed.) [18], pp. 61–74
2. Bellare, M., Boldyreva, A., O'Neill, A.: Deterministic and Efficiently Searchable Encryption. In: Menezes, A. (ed.) CRYPTO 2007. LNCS, vol. 4622, pp. 535–552. Springer, Heidelberg (2007)
3. Boneh, D., Di Crescenzo, G., Ostrovsky, R., Persiano, G.: Public Key Encryption with Keyword Search. In: Cachin, C., Camenisch, J.L. (eds.) EUROCRYPT 2004. LNCS, vol. 3027, pp. 506–522. Springer, Heidelberg (2004)
4. Boneh, D., Kushilevitz, E., Ostrovsky, R., Skeith III, W.E.: Public Key Encryption That Allows PIR Queries. In: Menezes, A. (ed.) CRYPTO 2007. LNCS, vol. 4622, pp. 50–67. Springer, Heidelberg (2007)
5. Bringer, J., Chabanne, H., Kindarji, B.: Error-tolerant searchable encryption. In: IEEE International Conference on Communications, 2009. ICC 2009, June 2009, pp. 1–6 (2009)
6. Bringer, J., Chabanne, H., Kindarji, B.: Identification with encrypted biometric data. CoRR abs/0901.1062 (2009) Full version of [5]
7. Chang, Y.C., Mitzenmacher, M.: Privacy Preserving Keyword Searches on Remote Encrypted Data. In: Ioannidis, J., Keromytis, A.D., Yung, M. (eds.) ACNS 2005. LNCS, vol. 3531, pp. 442–455. Springer, Heidelberg (2005)
8. Chor, B., Kushilevitz, E., Goldreich, O., Sudan, M.: Private Information Retrieval. J. ACM 45(6), 965–981 (1998)
9. Curtmola, R., Garay, J., Kamara, S., Ostrovsky, R.: Searchable Symmetric Encryption: Improved Definitions and Efficient Constructions. In: CCS 2006: Proceedings of the 13th ACM conference on Computer and communications security, pp. 79–88. ACM, New York (2006)
10. Daugman, J.: High Confidence Visual Recognition of Persons by a Test of Statistical Independence. IEEE Trans. Pattern Anal. Mach. Intell. 15(11), 1148–1161 (1993)

11. Fredman, M.L., Komlós, J., Szemerédi, E.: Storing a Sparse Table with O(1) Worst Case Access Time. ACM 31 (1984)
12. Goh, E.-J.: Secure Indexes. Cryptology ePrint Archive, Report 2003/216 (2003), http://eprint.iacr.org/2003/216/
13. Hao, F., Daugman, J., Zielinski, P.: A Fast Search Algorithm for a Large Fuzzy Database. IEEE Transactions on Information Forensics and Security 3(2), 203–212 (2008)
14. Indyk, P., Motwani, R.: Approximate Nearest Neighbors: Towards Removing the Curse of Dimensionality. In: Symposium on the Theory of Computing (1998)
15. Liu, X., Bowyer, K.W., Flynn, P.J.: Iris Recognition and Verification Experiments with Improved Segmentation Method. In: Fourth IEEE Workshop on Automatic Identification Advanced Technologies (AutoID), Buffalo, New York, October 17-18 (2005)
16. National Institute of Standards and Technology (NIST). Iris Challenge Evaluation (2005), http://iris.nist.gov/ICE
17. Sedghi, S., van Liesdonk, P., Doumen, J.M., Hartel, P.H., Jonker, W.: Adaptively Secure Computationally Efficient Searchable Symmetric Encryption. Technical Report TR-CTIT-09-13 (April 2009)
18. van Oorschot, P.C. (ed.): Proceedings of the 17th USENIX Security Symposium, San Jose, CA, USA, July 28-August 1. USENIX Association (2008)

A Security Model Associated to Symmetric Searchable Encryption

The following model for Symmetric Searchable Encryption was proposed in [9]. We briefly state the requirements and provide the construction given by the authors to comply with the model.

A.1 Security Model for Symmetric Searchable Encryption

A **history** H_q is an interaction between a client and a server over q queries, consisting of a collection of documents $\mathcal{D}$ and q keywords $\omega_1, \cdots, \omega_q$. Let $\mathcal{D}$ be a collection of n documents $(D_1, \cdots, D_n)$, and let `Enc` be an encryption function. If the documents of $\mathcal{D}$ are stored encrypted by `Enc`, and $H_q = (\mathcal{D}, \omega_1, \cdots, \omega_q)$ is a history over q queries, an adversary's **view** of H_q under the secret key K is defined as

$$V_K(H_q) = (\boldsymbol{id}(D_1), \ldots, \boldsymbol{id}(D_n), \mathtt{Enc}_K(D_1), \ldots, \mathtt{Enc}_K(D_n), \mathcal{I}_{\mathcal{D}}, T_{\omega_1}, \ldots, T_{\omega_q})$$

The History and the View of an interaction determine what did an adversary obtain after a client executed the protocol; an estimation of the information leaked is given by the Trace. Let $H_q = (\mathcal{D}, \omega_1, \ldots, \omega_q)$ be a history over q queries. The **trace** of H_q is the sequence

$$Tr(H_q) = (\boldsymbol{id}(D_1), \ldots, \boldsymbol{id}(D_n), |D_1|, \ldots, |D_n|, \mathcal{D}(\omega_1), \ldots, \mathcal{D}(\omega_q), \Pi_q)$$

where Π_q is a symmetric matrix representing the access pattern, *i.e.* $\Pi_q[i,j] = 1$ if $\omega_i = \omega_j$, and $\Pi_q[i,j] = 0$ otherwise.

For such a scheme, the security definition is the following.

Definition 5 (Adaptive Indistinguishability Security for SSE [9]). *A SSE scheme is said to be* adaptively indistinguishable *if for all $q \in \mathbb{N}$ and for all probabilistic polynomial-time adversaries $\mathcal{A}$, for all traces Tr_q of length q, and for all polynomially sampleable distributions*

$$\mathcal{H}_q = \{H_q \ : \ Tr(H_q) = Tr_q\}$$

(i.e. the set of all histories of trace Tr_q), the advantage $\mathtt{Adv}_{\mathcal{A}} = \left|\Pr[b' = b] - \frac{1}{2}\right|$ of the adversary is negligible.

$\mathtt{Exp}_{\mathcal{A}}^{IND}$

```
1. $K$                                   $\leftarrow \mathtt{Keygen}(1^k)$                                              $(\mathcal{C})$
2. $(\mathcal{D}_0, \mathcal{D}_1)$      $\leftarrow \mathcal{A}$                                                       $(\mathcal{A})$
3. $b$                                   $\overset{R}{\leftarrow} \{0,1\}$                                              $(\mathcal{C})$
4. $(\omega_{1,0}, \omega_{1,1})$        $\leftarrow \mathcal{A}(\mathcal{I}_b)$                                        $(\mathcal{A})$
5. $T_{\omega_{1,b}}$                    $\leftarrow \mathtt{Trapdoor}(K, \omega_{1,b})$                                $(\mathcal{C})$
6. $(\omega_{i+1,0}, \omega_{i+1,1})$    $\leftarrow \mathcal{A}(\mathcal{I}_b, T_{\omega_{1,b}}, \dots, T_{\omega_{i,b}})$   *for* $i = 1, \dots, q-1$   $(\mathcal{A})$
7. $T_{\omega_{i+1,b}}$                  $\leftarrow \mathtt{Trapdoor}(K, \omega_{i+1,b})$                              $(\mathcal{C})$
8. $b'$                                  $\leftarrow \mathcal{A}(V_K(H_b))$                                             $(\mathcal{A})$
```

In this experiment, the attacker begins by choosing two collections of documents (2.), which each contains the same number of keywords; then the challenger follows by flipping a coin b (3.), and the adversary receives the index of one of the collections $\mathcal{D}_b$; he then submits two words $(\omega_{1,0}, \omega_{1,1})$ (4.) and receives the trapdoor for $\omega_{1,b}$ (5.). The process goes on until the adversary has submitted q queries (6. and 7.) and he is challenged to output b (8.).

A.2 SSE Construction

The algorithms that implement the Symmetric Searchable Encryption in [9] are depicted in Figure 2. The scheme is proven indistinguishable against adaptive adversaries.

For this construction, a pseudo-random permutation noted π_K is used, where K is the secret key of the system. The security of this scheme rests on the indistinguishability of this pseudo-random permutation which ensures the indistinguishability of the sent data.

B Detailed Proofs

B.1 Proof of Proposition 1

Let $\mathcal{U}$ be a registered user to be identified, with reference template b and identity $ID(\mathcal{U})$. Let b' be a freshly captured template such that $d(b, b') < \lambda_{min}$. The scheme is complete if the probability for $ID(\mathcal{U})$ not to be returned is negligible, *i.e.* if $ID(\mathcal{U})$ appears less than $\lambda\mu$ times in $\mathcal{ID}(b')$.

Let us consider the event E_i : "$\mathcal{E}_{sk}(ID(\mathcal{U}))$ does not appear in row i of $\mathcal{ID}(b')$". E_i happens if and only if $h_i(b')||i||j \neq h_i(b)||i||j$, which happens with probability $1 - p_1$. Then, the probability for the scheme not to be complete is given

```
Keygen($1^k$): Generate a random key $K \xleftarrow{R} \{0,1\}^k$

BuildIndex ($K$, $\mathcal{D}$):

  • Initialization:
      - scan $\mathcal{D}$ and build $\Delta$, the set of distinct words in $\mathcal{D}$.
      - for each $\omega \in \Delta$, build $\mathcal{D}(\omega)$
      - compute $\texttt{max} = \max_{\omega \in \Delta}(|\mathcal{D}(\omega)|)$ and $m = \texttt{max} \cdot |\Delta|$
  • Build look-up table T :
      - for each $\omega_i \in \Delta$
            for $1 \leq j \leq |\mathcal{D}(\omega_i)|$
                set T $[\pi_K(\omega_i \parallel j)] = \texttt{id}(D_{i,j})$
      - if $m' = \sum_{\omega_i \in \Delta} |\mathcal{D}(\omega_i)| < m$, then set the remaining $(m - m')$
        entries of T to identifiers of documents $\texttt{id}(D_r)$, $r \in \{1, \ldots, n\}$
        such that the same identifier holds for the same number of
        entries.
  • Output $\mathcal{I}_{\mathcal{D}} = \texttt{T}$

Trapdoor ($K, \omega$): Output $T_\omega = (\pi_K(\omega \parallel 1), \ldots, \pi_K(\omega \parallel \texttt{max}))$

Search ($\mathcal{I}_{\mathcal{D}}, T_\omega$): For $1 \leq i \leq \texttt{max}$: retrieve $\texttt{id} = \mathcal{I}_{\mathcal{D}}\,[T_\omega[i]]$
```

Fig. 2. Adaptively secure SSE construction [9]

by: $\Pr[ID(\mathcal{U})$ appears in less than $\lfloor \lambda\mu \rfloor$ positions $] = \sum_{i=0}^{\lfloor\lambda\mu\rfloor} \binom{\mu}{i} p_1^i (1-p_1)^{\mu-i}$. But, considering $1 - p_1 \leq \frac{1}{4^{\mathcal{H}_2(\lambda)+c}}$, we have: $(1-p_1)^{\mu-i} \leq \frac{1}{4^{(\mathcal{H}_2(\lambda)+c)(\mu-i)}} \leq \frac{1}{4^{(\mathcal{H}_2(\lambda)+c)(\frac{\mu}{2})}} = \frac{1}{2^{\mu(\mathcal{H}_2(\lambda)+c)}}$.

Thus, $\sum_{i=0}^{\lfloor\lambda\mu\rfloor} \binom{\mu}{i} p_1^i \,(1-p_1)^{\mu-i} \leq \sum_{i=0}^{\lfloor\lambda\mu\rfloor} \binom{\mu}{i} \,(1-p_1)^{\mu-i} \leq (\lfloor\lambda\mu\rfloor + 1) \cdot \frac{2^{\mu\mathcal{H}_2(\lambda)}}{2^{\mu(\mathcal{H}_2(\lambda)+c)}} \leq (\lfloor\lambda\mu\rfloor + 1) \cdot \frac{1}{2^{c\mu}}$ which is negligible. This proves the result. $\square$

B.2 Proof of Proposition 2

Let b' be a freshly captured template such that $d(b, b') > \lambda_{max}$ for any registered template b. The system returns an identity if and only if one identity appears in at least $\lceil\lambda\mu\rceil$ entries. This implies that for at least $\lceil\lambda\mu\rceil$ LSH functions h, we have, $h(b) = h(b')$. Given a hash function, and regarding the definition of a LSH family, this occurs with a probability p_2. So, $\Pr[\texttt{Identification}(b') \neq \emptyset] = \sum_{i=\lceil\lambda\mu\rceil}^{\mu} \binom{\mu}{i} p_2^i (1-p_2)^{\mu-i} \leq 2^\mu \cdot p_2^{\lambda\mu}$. If $p_2 \leq \frac{1}{2^{\frac{1}{\lambda}+c}}$, this probability is negligible too. This gives the result. $\square$