b-it

Bonn-Aachen International Center
for Information Technology

Seminar: Biometry & Security Winter 2009/2010

Report

Secure Biometric Authentication With Improved Accuracy

Author: Filip Müllers (Matrikel-Nummer: 1608402)

Version: 30.01.2010

Contents

0 Introduction

In the seminar "Biometry & Security" we dealt with different types of cryptosystems. One of the actual systems is described in the paper "Secure Biometric Authentication with Improved Accuracy"[1] written by Manual Barbosa, Thierry Brouard, Stéphane Cauchy and Simão Melo de Sousa and will provide the foundation for this work. The described system is a hybrid protocol for cryptographically secure biometric authentication, which uses state-of-the-art identification classifiers and "strong cryptographic security based on the Paillier public key encryption" ([BBCM08]).

The remaining of this report is organized as follows. Firstly I will give reasons for the decision of selected techniques. This will be the motivation in chapter 1. Then I will present the protocol in chapter 2 and distinguish it from a similar protocol provided by Bringer et al[2]. Chapter 3 will discuss the Pattern Recognition System in general and in chapter 4 we will see Support Vector Machines as concrete classifiers. The Paillier-based algorithm implementation will be examined in chapter 5 before I will summarize the system in chapter 6.

1 Motivation

Imagine that you want to use biometric authentication to access privileged resources over the internet. In this scenario you have to cope with different problems. The service provider must not get the opportunity to associate your identity with your biometric data. According to this the biometric data has to be stored in another server. This raises another issue, the insecure communication channel. The authors present a hybrid protocol for cryptographically secure biometric authentication, without transmission of user identities during the whole authentication process. In this context "hybrid" means a cooperation of different servers for data acquisition and feature recognition. The insecurity of the communication channel will be eliminated by using the Paillier public key encryption.
In comparison to other protocols the provided protocol will also improve the accuracy of authentication.

Biometric Data	Bit Length	Fuzzy Error	Classifier Error
Key stroke	12	48%	1.8%
Voice	46	20%	5%
Signature	40	28%	5%
Face	120	5%	0.6%
Fingerprint	128	17%	8%
Iris	140	5%	5%

(Tab. 1) adapted from [BBCM08]

As you can easily see in Tab. 1, "advanced [pattern] classifiers consistently outperform simple distance-based (fuzzy) classification techniques" and bring improvements especially for behavioural biometric data. The authors give reasons for this characteristics: "Fuzzy-pattern recognition components can [only] deal with acquisition variability [and] not with user variability [while] advanced classifiers are build on the assumption that two users may produce close measurements". ([BBCM08]). To achieve higher accuracy the provided protocol takes the benefit of the Support Vector Machines as an advanced classifier, which can optimally minimize the error risk. An Identification-classifier does not need to know, who the user claims to be in order to determine if he/she belongs to the set of valid users.

1 ([BBCM08])

2 [BCIPTZ07]

2 Protocol

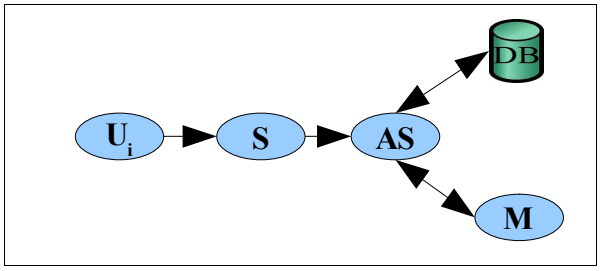
The proposed Protocol rests upon the one provided by Bringer *et al.* and will be denoted as "Bringer-Protocol" in the following.

2.1.1 Bringer-Protocol

The Bringer-Protocol uses the Goldwasser-Micali encryption scheme that is an "asymmetric key encryption algorithm developed by Shafi Goldwasser and Silvio Micali in 1982 [and takes benefit of its homomorphic properties. Classification is done with] the Hamming distance between fresh biometric readings and stored biometric profiles" ([wEnGM10]). As the authors wish for their own protocol, the Bringer-Protocol uses distributed server-functionality to hide the association between biometric data b_i and user identities ID_i to gain higher privacy protection. Therefore it has three server-sided components:

- an authentication service **AS** that knows the user's claimed identity and wants to verify it
- a database **DB** with a total number of N records, that stores the biometric data b_i, where $b_i = (b_{i,1}, b_{i,2},..., b_{i,M})$ is a binary vector of the Dimension M, without knowing the corresponding ID_i and
- a matching service **M**, that authenticates users without making an association between ID_i and b_i

"These Services are assumed to be honest-but-curious" ([BCIPTZ07]).



(Scheme 1) adapted from [BCIPTZ07]

The Bringer-Protocol works in two phases, the "Enrolment" and the "Verification". The "Enrolment" is a setup-phase and has do be executed before any user can try to authenticate in the "Verification", which is the actual operation-phase.

Enrolment:

- a user U_i registers its biometric template (b_i,i) at the DB and its Identity (Id_i,i) at the AS, where i is the index of the record b_i in the DB
- M generates a keypair (pk, sk) for the Goldwasser-Micali scheme (K,E,D)[3]

Verification:

- The Sensor S captures fresh b'_i from the user U_i and sends $E(b'_i, pk)$ and ID_i to the AS
- AS gets the index i from ID_i and sends $E(t_j, pk)$ $(1 \leqslant j \leqslant N)$ to the DB,where $\begin{cases} t_j = 1 & if\ j = i \\ t_j = 0 & else \end{cases}$
- for every $(1 \leqslant k \leqslant M)$, the DB computes $E(b_{i,k}, pk)$ and sends them to the AS
- AS computes a vector $v_k = E(b'_{i,k}, pk)E(b_{i,k}, pk) \bmod n = E(b'_{i,k} \oplus b_{i,k}, pk)$
- and permutes it into λ_k before passing it to M
- M decrypts λ_k , checks if Hamming-weight of $(D(\lambda_1, sk), D(\lambda_2, sk), ..., D(\lambda_M, sk)) \leqslant d$ and sends the result back to the AS.
- AS accepts or rejects the user depending on M's result.

3 K = Key-generation, E=Encryption, D=Decryption

2.1.2 Vulnerability

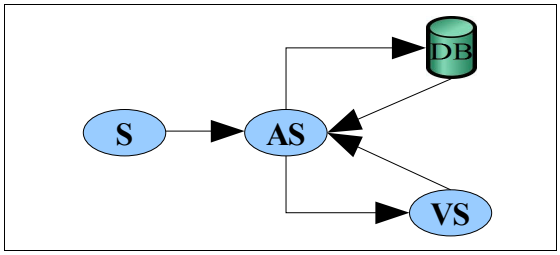
The Goldwasser-Micali cryptosystem is proven to be semantically secure, in other words for a computationally-bounded adversary it is infeasible to get significant information about the plaintext from a given ciphertext and the corresponding public encryption key.
Unfortunately the Bringer-Protocol is not shielded from a simple attack executed by the Authentication Service AS. AS can recover the user profile data that the Database DB has used in its prior calculations with exhaustive search. For this procedure the following steps are necessary:

- Let E be the Goldwasser-Micali encryption algorithm and i the index of the user that wants to authenticate. AS calculates the vector $(E(t_1, pk), \ldots, E(t_N, pk))$, where N is the number of users and $\begin{cases} t_j = 1 & \text{if j = i} \\ t_j = 0 & \text{else} \end{cases}$
- AS calculates $E(b_{i,k}, pk) = \prod_{j=1}^{N} E(t_j, pk)^{b_{j,k}} \bmod n$, for $1 \leqslant k \leqslant M$, where $(b_{i,1}, \ldots, b_{i,M})$ is the users biometric profile
- By calculating $\frac{E(b_{i,k}, pk)}{\prod_{j \in J} E(t_j, pk)} \bmod n$, $\forall$ subsets $J \subset \{1 \ldots N\}$ without i AS can try to figure out whether $b_{i,k}$ is 1 or 0. The $E(t_j, pk)$ are the same as the ones passed originally to the DB. If the calculation leads to 1 then $b_{i,k} = 0$, but if it leads to $E(t_i, pk)$ then $b_{i,k} = 1$

After recovering a biometric profile, the AS is able to determine on its own which features belong to a user, without executing the protocol. This attack is only practicable for a small number of users N, because the complexity is exponential in N and therefore computing time grows very fast. Nevertheless this insecurity can be avoided by re-randomization of the cypher-texts before passing them on from the DB to the AS. In Chapter 5.1.2 a concrete implementation of this patch will be shown concerning the authors protocol.

2.2 Hybrid Authentication Protocol

As already mentioned the authors protocol is based on the Bringer-Protocol and so it is quite similar. As you can see in (Scheme 2) there again are three server-sided components:



(Scheme 2) adapted from [BBCM08]

1. The Authentication Service **AS** that organizes the whole procedure. A successful authentication means that the **AS** learns the user's identity, so it should learn nothing about the user's biometric data.
2. The Database Server **DB** that securely stores the users' profiles and executes the pre-decision part of the classification cl. Because the **DB** knows the biometric data it should learn nothing about the user's identity.
3. Verification Server **VS** that takes the output from DB and computes a decision D. The **VS** shouldn't learn anything about user's identity, too, or correlate or trace authentication runs from a given unknown user.

On the client-side there is only again the Sensor **S** which should provide a liveness test and the ability to capture user's biometric data and extract it into a binary string. Further the sensor must be able to perform cryptographic operation, e.g. public key encryption, to encrypt the binary string. The three server-sided components distribute the server functionality again and no single entity can associate a user's identity with the user's biometric data.

2.2.1 Algorithms

After the data collection procedure which is comparable to the enrolment stage of the Bringer-Protocol, each server provides a set of static data. This procedure is assumed to be realized in a secure way, so that only the respective servers get the belonging static data and no information is leaked.
AS contains a list of system user identities $ID_i \in \{0,1\}^*$, where i will be used as the system user identifier uid . The whole set of data is called $\mathbf{AS_{data}}$. DB includes the biometric classification data for the users and permits a pre-classification of all these system users with their $uids$, called $\mathbf{DB_{data}}$. VS has extra information to obtain a verdict from the obfuscated pre-decision classification, named $\mathbf{VS_{data}}$.
The following probabilistic polynomial time algorithms are executed during an authentication run:

Participant	Algorithm
VS	(params, k_d) ← Gen(1k)
S	auth ← S(v_{ID}, params)
DB	class ← Classify(params, auth, DB_{data})
AS	(sclass, π) ← Shuffle(params,class, AS_{data})
VS	d ← Decide(sclass, params, kd, VS_{data})
AS	ID / ⊥ ← Identify(d, π, AS_{data})

(Tab. 2) adapted from [BBCM08]

According to (Tab. 2) authentication in this protocol is gained in these steps:

0. The key generation algorithm **Gen** is executed by the VS. It publishes the resulting set of public *params* and stores the secret key k_d.
1. S encrypts fresh biometric data v_{ID} using algorithm **S** and initializes the authentication request *auth*.
2. a) AS receives this request and passes it on to DB.
   b) DB returns encrypted classification information *class* by calculating the pre-decision classification **Classify.**
3. The algorithm **Shuffle** is invoked by the AS to make it impossible to associate a decision from previous authentication runs. At this it outputs the scrambled version of *class*, namely *sclass* and stores the descrambling key π.
4. AS uses its private-key k_d and *sclass* for the final decision *d* using **Decide**.
5. Resulting from *d* AS uses **Identify**, the descrambling key π and AS_{Data} to recover the user's real identity or $\perp$.

The soundness condition is that the server-side system and AS in particular "produces a correct decision [and] recognises whether a new feature belongs to a valid user, and determines the correct identity" ([BBCM08]).
The probability *Pr* for all fresh features v_{ID} has to be sufficiently close to a value for practical use as an authentication protocol.

$$\Pr\left[\text{Identify}(d, \pi, AS_{data}) = \text{ID or } \perp \;\middle|\; \begin{array}{l} (params, k_d) \leftarrow \text{Gen}(1^k) \\ auth \leftarrow \text{S}(v_{ID}, params) \\ class \leftarrow \text{Classify}(params, auth, DB_{data}) \\ (sclass, \pi) \leftarrow \text{Shuffle}(params, class, AS_{data}) \\ d \leftarrow \text{Decide}(sclass, params, k_d, VS_{data}) \end{array}\right]$$

2.3 Security Model

The authors assume the servers do not collude and follow the protocol rules, but may use gained information to subvert these requirements. Even so the protocol has to be secure in two ways:
Firstly, concerning privacy neither a service nor a passive attacker may get enough information to reconstruct a pair of identity and feature of a single user.
Also it should not be possible to distinguish whether two auth-requests belong to the same person. This will be shown in the game (Exp. 1), which deals with the feature indistinguishability f^{fIND}. The adversaries adv ϵ {AS, DB, VS, Eve}, with Eve is an eavesdropper, are the input in this game together with two biometric readings, v_0, which is a fresh reading from a valid system-user *ID* and v_1, a reading from any user. The adversary must be unable to distinguish between which of two features belongs to a certain system user.

$$
\begin{array}{l}
\mathrm{Exp}_{\beta}^{fIND}(adv, v_0, v_1) \\
\quad (params, k_d) \leftarrow \mathrm{Gen}(1^k) \\
\quad auth \leftarrow \mathrm{S}(v_0, params) \\
\quad class \leftarrow \mathrm{Classify}(params, auth, DB_{data}) \\
\quad (sclass, \pi) \leftarrow \mathrm{Shuffle}(params, class, AS_{data}) \\
\quad d \leftarrow \mathrm{Decide}(sclass, k_d, VS_{data}) \\
\quad r \leftarrow \mathrm{Identify}(d, \pi, AS_{data}) \\
\quad \mathrm{Return}(v_\beta, view_{adv}) \\
\\
view_{AS} := (auth, class, sclass, \pi, d, r, AS_{data}, params) \\
view_{DB} := (auth, class, DB_{data}, params) \\
view_{VS} := (sclass, d, VS_{data}, k_d, params) \\
view_{Eve} := (auth, class, sclass, d, params)
\end{array}
$$

(Exp. 1)

It is required that, for all ID in the set of valid system-users and all *adv* the following distributions be computationally indistinguishable (≡): $\{(ID, \mathrm{Exp}_{\beta=1}^{fIND}(adv, v_0, v_1))\} \equiv \{(ID, \mathrm{Exp}_{\beta=0}^{fIND}(adv, v_0, v_1))\}$
The advantage Adv^{fIND}(adv) is the absolute value of the deviation from ½ in the probability that *adv* guesses β .

Secondly, concerning untraceability no possible adversary, except the AS, may get enough information to recognize a previously authenticated user. This user indistinguishability u^{IND} will be shown in the game (Exp. 2), where the set of possible adversaries is adv ϵ {DB, VS, Eve}. This game takes two fresh readings v_0, v_1 from valid users *uid* and *uid'*.

$$
\begin{array}{l}
\mathrm{Exp}_{\beta}^{uIND}(adv, v_0, v_1) \\
\quad (params, k_d) \leftarrow \mathrm{Gen}(1^k) \\
\quad auth \leftarrow \mathrm{S}(v_0, params) \\
\quad class \leftarrow \mathrm{Classify}(params, auth, DB_{data}) \\
\quad (sclass, \pi) \leftarrow \mathrm{Shuffle}(params, class, AS_{data}) \\
\quad d \leftarrow \mathrm{Decide}(sclass, k_d, VS_{data}) \\
\quad r \leftarrow \mathrm{Identify}(d, \pi, AS_{data}) \\
\quad \mathrm{Return}(view_{adv}) \\
\\
view_{AS} := \text{see (Exp. 1)}
\end{array}
$$

(Exp. 2)

It is required that, for all valid users with identifiers *uid* & *uid'* and all *adv* the following distributions be

computationally indistinguishable (≡): $\{(\text{uid}, \text{uid}', \text{Exp}_{\beta=1}^{uIND}(adv, v_0, v_1))\} \equiv \{(\text{uid}, \text{uid}', \text{Exp}_{\beta=1}^{uIND}(adv, v_0, v_1))\}$
The advantage $\text{Adv}^{\text{uIND}}(\text{adv})$ is the absolute value of the deviation from ½ in the probability that *adv* guesses β.

3 Biometric Authentication System

To explain the biometric authentication system it is essential to understand the pattern recognition system, which is the basis of this protocol. Therefore a definition for a pattern recognition system for the purpose that will lead to a distinction of biometric authentication and identification in 3.1 will be given. After this the necessary steps for the authentication system in 3.2 will be shown.

3.1 Pattern Recognition System

A Pattern recognition system for biometric identification is a 5-tupel $\Gamma = \{b, SU, \rho_b, D \circ cl, w_{SU}^*\}$, where

- $-b \in B$ B is a complete set of biometric parameters
- $-SU \subset U$ SU is set of systemusers, U is the univers of possible users
- $-\rho_b : U \to V$ is an Application illustratingan adequate sensor, V is the sensor-dependent space of biometric features
- $-w_{SU}^* \in W$ W is a sensor-dependent set of possible users profiles (n-tupel of realnumbers)
- $-D \circ cl$ is the classifier with a pre-decision processing stage cl and a decision stage D

The system wants to recover a pre-image of a feature $\rho_b(u)$ for $u \in SU$ using prior knowledge of a users profile $w_{SU}^* \in W$ and the two classifier stages.

- Cl takes a feature and pre-established profile information and returns classification data, e.g. confidence intervals, distances, etc.
- The decision stage D makes the final decision using an appropriate criterion, e.g. pre-defined threshold, majority rules, etc.

The sytem will classify as follows:

$$\forall u \in SU, D(cl(\rho_b(u), w_{SU}^*)) = u \qquad \forall u \in U / SU, D(cl(\rho_b(u), w_{SU}^*)) = \perp$$

For a set of users SU, such that |SU| > 1 the system is a biometric identification system, for |SU| = 1 it is a biometric authentication system. You can use a biometric authentication system for identification, by trying all possible users in a database. In order to avoid the transmission of the user's claimed identity, a biometric identification system is used. This way "the authentication protocol can be implemented by transmitting only the user's biometric data" ([BBCM08]).

3.2 Authentication System

The authentication system is a two stage procedure. The first stage is the "Enrolment", that can be seen as a set-up stage. The second is the actual operation stage and is named "Generalisation".

- The Enrolment consists of two sub-steps:
  - The acquisition and feature extraction step, where a reference set of feature values $\rho_b(u)(\forall u \in SU)$ is constructed, this is called the training set.
  - The learning step uses the training set to construct the users' profile w_{SU}^* .
- The Generalisation has two sub-steps, too:
  - The acquisition and feature extraction step collects a feature $v = \rho_b(unknown)$ for an unknown user.
  - The decision step uses the classifier *cl* and profile data w^* to determine who of the system users is *unknown.*

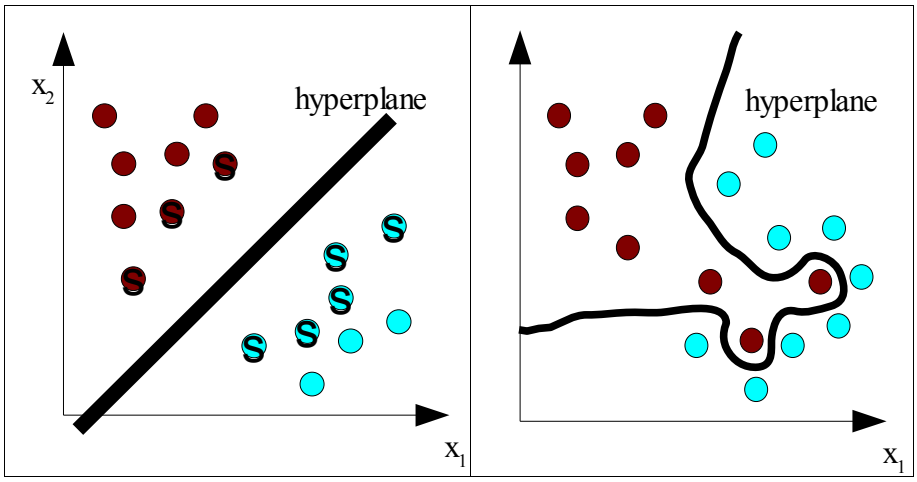
$$\{u \in SU, \perp\} \leftarrow D(cl(v, w_{SU}^*))$$

4 The Support Vector Machine Classifier

In this Chapter the attention is turned away from the theoretical systems to more concrete implementations. Support Vector Machines (SVMs) provide a new approach to the problem of pattern recognition. In comparison to other approaches such as neural networks: “SVM training always finds a global minimum, and their simple geometric interpretation provides fertile ground for further investigation” ([Bur98]).
The Chapter is divided into two parts. In 4.1 the basic ideas behind Support Vector Machines will be explained and in 4.2 a classifier for the by the authors meant purpose will be presented, that works in the SVM way.

4.1 Support Vector Machine

The origin of support vector machines is the machine learning field. So it is not a machine in the classical meaning, it is a mathematical method for pattern matching. In simple words, we have a training set with examples, each marked as belonging to one of two classes, “the SVM training algorithm builds a model that predicts whether a new example [is part of one class or the other]” ([wEnSV10]). The given examples are represented as vectors in a vector-space and the separate classes are divided by a clear borderline, that has the maximum distance to the examples of both classes. The borderline divides two hyperplanes, one for each class. Intuitively it is not necessary to use all the examples to build these hyperplane, the training-vectors that are farther from the borderline and lie behind vectors that a nearer to the other hyperplane do not have an effect on the position of the hyperplane. The vectors that do affect the hyperplane are called support vectors. In (Graphic 1) these special vectors are marked with an S.



(Graphic 1) (Graphic 2)

In this simple example the two classes (blue and red) are linear separable, but in general this is not the case. As is illustrated in Graphic 2 the SVM can also cope with non-linear separable examples. Therefore a kernel-function K is used that builds the hyperplane. The Idea of this kernel-function is to compute maximum borderline in a higher vector-space. In a sufficiently high vector-space every example-set becomes linear separable. During the inverse transformation to the original dimension vector-space the linear hyperplane becomes a non-linear one, but still divides the vectors into two classes (Graphic 2).

4.2. Classifier

As described in 4.1 a basic support vector machine can distinguish, if a feature belongs to one of two classes (binary problem). The goal is to extend the classification capabilities to |SU| classes. For such a multi-class SVM the single multi-class problem is reduced into multiple binary problems as in 4.1. Each binary classifier, “produces an output function that gives relatively large values for examples from the positive class and relatively

small values for examples belonging to the negative class" ([wEnSV10]). The authors prefer a one-against-all strategy, so that the classifier with the highest output for a new instance assigns the class.
According to this we train such binary classifiers for each user $u \in SU$ using the remaining users SU/u as the rejected class. For our purpose we define a SVM as a |SU|-class identification classifier and the classifier function like this: $cl_{\text{SVM}} : \mathbb{V}_{\text{SVM}} \times \mathbb{W}_{\text{SVM}} \rightarrow \mathbb{N}^{|\text{SU}|}$

$cl_{\text{SVM}}(v, w^*_{|SU|}) := (cl^{(1)}_{\text{SVM}}(v, w^*_{|SU|}), \ldots, cl^{(|SU|)}_{\text{SVM}}(v, w^*_{|SU|}))$,
where v is a fresh biometric reading, $w^*_{|SU|}$ contains $(\alpha_{i,j}, SV_{i,j})$ for $1 \leqslant i \leqslant S, 1 \leqslant j \leqslant |SU|$
and $cl^{(j)}_{\text{SVM}}(v, w^*_{|SU|}) := \sum_{i=1}^{S} \alpha_{i,j} K(v, SV_{i,j})$
$K(a,b)$ is simplified by the authors as the "scalar product between a and b in the initial space" ([BBCM08]):
$K(a,b) = \sum_{l=1}^{k} a_l b_l$

4.2.1 Learning Stage

In the learning stage the SVM determines an outer and an inner hyperplane in the k-dim features space. The hyperplanes are expressed as linear combination of the support vectors $SV_{i,j} \in \mathbb{V}_{\text{SVM}}$, i = 1...S, j=1...|SU|, which are known samples. These $SV_{i,j}$ are weighted with $\alpha_{i,j} \in \mathbb{N}$ coefficients:

$$\mathbb{V}_{\text{SVM}} = \mathbb{N}^k \text{ and } \mathbb{W}_{\text{SVM}} = (\mathbb{N} \times \mathbb{V})^{S \times |\text{SU}|}$$

4.2.2 Processing Stage

For authentication the SVM evaluates the distance of a fresh feature v to the hyperplanes using a scalar product. In the non-linear separable case the kernel-function K is used as described in 4.1. This way the computational cost is reduced "compared to a basic projection followed by the scalar product" ([BBCM08]), because K calculates it in a single step.
After that the decision D is calculated by finding the index of the maximum positive scalar contained in the vector $cl_{\text{SVM}}(v, w^*)$.

$$D_{\text{SVM}}(cl_{\text{SVM}}(v, w^*)) := \begin{cases} d \leftarrow argmax_{j=1}^{|SU|}(cl^{(j)}_{\text{SVM}}(v, w^*)) \\ \textbf{IF } cl^{d}_{\text{SVM}}(v, w^*) > 0 \textbf{ THEN return } d \\ \textbf{ELSE return } \perp \end{cases}$$

5 Algorithm Implementation

As the given Cryptosystem is based on the Paillier Cryptosystem, it is introduced in chapter 5.1.1 before the concrete implementation of the presented algorithms is shown in 5.1.2. A Security analysis is sketched in 5.2.

5.1.1 Paillier Cryptosystem

The Paillier Public Key Encryption Scheme is a probabilistic asymmetric algorithm consisting of three parts:

1. Key generation: $G_{\text{paillier}}(1^k) = (k_d, k_e)$, where 1^k is the security parameter and the output is the key-pair, public key $k_e = (n,g)$ and secret key $k_d = (\mu,\lambda)$.
   n=pq, where p and q are large random prime numbers and λ = lcm(p-1,q-1). The Algorithm randomly selects $g \in \mathbb{Z}^*_{n^2}$ such that n divides the order of g.

This can be proved by $gcd(L(g^{\lambda} mod\ n^2), n) = 1$, where $L(u) = \frac{u-1}{n}$, because it implies the existence of the multiplicative inverse: $\mu = (L(g^{\lambda} mod\ n^2))^{-1} mod\ n$.

2. Encryption: $\xi_{paillier}(m,k_e) = c$.
   Takes the public key and a message and outputs $c \in \mathbb{Z}_n^2$ the cipher-text calculated by $c = g^m r^n \bmod n^2$, where $r \in \mathbb{Z}_n^*$ is generated uniformly at random.
3. Decryption: $\mathfrak{D}_{paillier}(c,k_d) = m$.
   Recovers the message m by calculating $m = \frac{L(c^{\lambda} mod\ n^2)}{L(g^{\lambda} mod\ n^2)} mod\ n$.

The Paillier Scheme reveals the following homomorphic properties $\forall m_1, m_2 \in \mathbb{Z}_n$ and $k \in \mathbb{N}$, where D denotes the decryption $\mathfrak{D}_{paillier}$ and E denotes the encryption $\xi_{paillier}$:

1. $D(E(m_1)\,E(m_2) mod\ n^2) = m_1 + m_2 mod\ n$
2. $D(E(m_1)\,k \bmod n^2) = k m_1 mod\ n$
3. $D(E(m_1)\,g^{m_2} mod\ n^2) = m_1 + m_2 mod\ n$
4. $\left.\begin{matrix} D(E(m_1)\,m_2 mod\ n^2) \\ D(E(m_2)\,m_1 mod\ n^2) \end{matrix}\right\} = m_1 m_2 mod\ n$

It also provides a useful property that allows to re-randomize a given Paillier Cryptosystem called Self-Blinding $\forall m \in set\ Z_n$ and $r \in \mathbb{N}$:

5. $D(E(m)\,r^n mod\ n^2) = m$ or $D(E(m)\,g^{nr} mod\ n^2) = m$

According to this property "any cipher-text can be publicly changed into another one without affecting the plain-text" ([Pai99]).
The original cryptosystem as shown above does provide semantic security against chosen-plaintext attacks (IND-CPA). Every probabilistic polynomial time adversary has only a negligible advantage over random guessing.

5.1.2 Concrete Implementation

On the basis of the Paillier cryptosystem the authors suggest the following implementations:

1. **Gen**$(1^k) \rightarrow$ (params,k_d):
   The Generation-Algorithm uses the Paillier Scheme and produces the key-pair, where params equals k_e.
2. **S**(v) $\rightarrow$ auth:
   The Sensor-Algorithm takes a fresh feature from an unknown user. Looking at the feature as $v := (v_1, \ldots, v_k) \in \mathbb{Z}_n^k$ [4] the encryptions runs componentwise:
   Output: auth $\leftarrow (\xi_{paillier}(v_1,k_e),\ldots,\xi_{paillier}(v_k,k_e))$
3. **Classify**(auth, DB_{data}, params) $\rightarrow$ class:
   This algorithm computes the pre-decision without decrypting the features in the authentication request auth. It takes the profile data $w_{|SU|}^*$ from DB_{data} and calculates $for\ 1 \leqslant j \leqslant |SU|$:
   $c_j = \prod_{i=1}^{S} \mathring{K}(auth, SV_{i,j})^{\alpha_{i,j}} = \xi_{Paillier}(\sum_{i=1}^{S} \alpha_{i,j} K(v, SV_{i,j}), params)$,
   with $\mathring{K}(auth, SV_{i,j}) := \prod_{l=1}^{k} [auth_j]_l^{[SV_{i,j}]_l}$, where $[.]_l$ denotes the l^{th}-component in a tuple.
   The Authentication Service AS could perform exhaustive search of the profile space, so the Database DB re-randomizes the encrypted pre-decision-data by $class_j = (c_j r_j^n) mod\ n^2$ using the self-Blinding

4 In the Context of SVM the feature space was presented as $\mathbb{V}_{SVM} = \mathbb{N}^k$

property of the Paillier Scheme and produces $class = (class_1, \ldots, class_{|subU|})$.

4. **Shuffle**(class → (sclass, π):
   Generates a fresh permutation π: {1,…,|SU|} → {1,…,|SU|}, and re-randomizes all cypher-text components in class by calculating $sclass_j = (class_{\pi(j)} r_j^n) \bmod n^2$ and resulting in $sclass = (sclass_1, \ldots, sclass_{|subU|})$.
5. **Decide**(sclass, k_d, VS_{data}) → d:
   The Decision-Algorithm decrypts the components in sclass and classifies with the SVM classifier, giving us d, which is the index in the input vector corresponding to the largest positive scalar, or ⊥.
6. **Identify**(d, π, As_{data}) → ID:
   This algorithm either outputs ⊥ or for authentication runs where d has a concrete value it identfies the user by unshuffling the decision with uid = π^{-1}(d). So the final result is the identity ID or ⊥.

5.2 Security Analysis

It was already dealt with the security of the Bringer Protocol and its weakness were patched. The following Theorems are not proved in the given paper, but the authors refer to the full version of their paper.

1. Theorem:
   "The proposed protocol ensures feature privacy. More precisely, any [probabilistic polinomial-time] adversary has negligible advantage in distinguishing the distributions associated with Exp^{fIND}".
2. Theorem:
   "The proposed protocol ensures user untraceability. More precisely, any [probabilistic polinomial-time] adversary has negligible advantage in distinguishing the distributions associated with Exp^{uIND}".

([BBCM08]).

6 Conclusion

To sum up, the authors proposed a hybrid authentication protocol based on the protocol provided by Bringer *et al.* and improved it in many ways. More precisely, higher authentication accuracy was achieved using state of the art multi-class classifiers from a pattern recognition point of view. With this classifiers namely the Support Vector Machines in combination with an encryption scheme based on the Paillier public key encryption scheme another advantage over other biometric authentication systems was obtained. By using the SVMs as an identification classifier instead of an authentication classifier also a privacy improvement was gained. In this vein no user identities are transmitted at any point of the protocol. The last improvement concerns the lack of security in the protocol provided by Bringer *et al.* which was easily fixed by re-randomization of the output-data from the Database.

The main conclusion that can be drawn is therefore that biometric data may be stored in public servers, but anonymous.

References

[BBCM08] MANUEL BARBOSA, THIERRY BROUARD, STÉPHANE CAUCHIE, SIMÃO MELO DE SOUSA - *Secure Biometric Authentication with Improved Acuracy*, 2008

[BCIPTZ07] JULIEN BRINGER, HERVÉ CHABANNEL, MALIKA IZABACHÉNE, DAVID POINTCHEVAL, QIANG TANG, SÉBASTIEN ZIMMER - *An Application of the Goldwasser Micali Cryptosystem to Biometric Authentication*, 2007

[MTG07] PRAVEER MANSUKHANI, SERGEY TULYAKOV, VENU GOVINDARAJU - *Using Support Vector Machines to Eliminate False Minutiae Matches during Fingerprint Verification*, 2007

[Pai99] PASCAL PAILLIER - *Public-Key Cryptosystems Based on Composite Degree Residuosity Classes*, 1999

[Bur98] CHRISTOPHER J.C. BURGES - *A Tutorial on Support Vector Machines for Pattern Recognition*, 1998

[wEnGM10] WIKIPEDIA.ORG – *Goldwasser-Micali cryptosystem* as in January 2010 http://en.wikipedia.org/wiki/Goldwasser-Micali_cryptosystem

[wEnCi10] WIKIPEDIA.ORG – *Semantic security* as in January 2010 http://en.wikipedia.org/wiki/Semantic_security

[wEnCi10] WIKIPEDIA.ORG – *Ciphertext indistinguishability* as in January 2010 http://en.wikipedia.org/wiki/Ciphertext_indistinguishability

[wEnPC10] WIKIPEDIA.ORG – *Paillier cryptosystem* as in January 2010 http://en.wikipedia.org/wiki/Paillier_cryptosystem

[wDeSV10] WIKIPEDIA.ORG – *Support Vector Machine* as in January 2010 http://de.wikipedia.org/wiki/Support_Vector_Machine

[wEnSV10] WIKIPEDIA.ORG – *Support Vector Machine* as in January 2010 http://en.wikipedia.org/wiki/Support_vector_machine