# Customisation of Paillier Homomorphic Encryption for Efficient Binary Biometric Feature Vector Matching

Georg M. Penn, Gerhard Pötzelsberger, Martin Rohde, Andreas Uhl
University of Salzburg, Department of Computer Sciences
J.-Haringerstr.2, 5020 Salzburg, Austria
andreas.uhl@sbg.ac.at

**Abstract:** Computing the Hamming weight between binary biometric feature vectors in a homomorphic encryption domain can be rather inefficient due to the required bit-wise encryption. A biometric matching technique more efficient than the Goldwasser-Micali approach is proposed based on exploiting Paillier's capability of encrypting messages larger than one bit at a time. The efficiency is documented in an iris identification context.

## 1 Introduction

In (distributed) biometric authentication, biometric sample (or template) data is sent from the acquisition device to the authentication component and can eventually be read by an eavesdropper on the channel. Also, biometric enrollment template (or even sample) databases can be compromised and the data misused in fraudulent manner. However, data corresponding to several biometric modalities has to be assumed to be public, as fingerprints, for instance, are left on every object a person touches, and pictures of a persons' facial or even periocular area can easily be taken without notice (or can be taken from Facebook or Flickr). It is therefore not (only) the biometric data itself that needs protection, but essentially as well the relationship between the identity and the biometric feature of an individual. Therefore, there is a strong need for keeping private data confidential during biometric authentication [SSW10]. As the relationship between the person's identity and her biometric trait must be hidden, comparison of the enrolled template and a newly captured sample has to take place in an encrypted domain [BCI+07]. This also prevents misuse of eventually leaked transmitted samples or template databases. However, the sample differs from the template stored in the database. Hence, using traditional cryptographic methods such as hash-values is not suitable in this scenario.

To cope with these demands, various flavours of template protection schemes have been developed, termed biometric cryptosystems and cancelable biometrics [RU11]. While these techniques provide sufficient computational efficiency for practical employment, most approaches are restricted to verification and many security concerns have arisen. As an alternative approach, matching in an homomorphic encrypted domain [AMFF+13] has been suggested – while providing satisfying security and suited for identification applications in principle, the low computational efficiency prevents its usage in large-scale

identification scenarios.

In this paper, we introduce a more efficient matching process for generic binary template vectors by exploiting Paillier's capability of encrypting messages larger than one bit at a time. We experimentally evaluate and compare the corresponding key generation, encryption, and matching mechanisms of the proposed scheme to those of Paillier and Goldwasser-Micali with respect to computational efficiency. Finally, an iris-based identification approach is assessed using the proposed technique. Section 2 reviews prior work on using homomorphic encryption in privacy-preserving biometric matching and defines the assumed authentication architecture. In Section 3 we discuss how binary (feature) vectors can be compared in encrypted domains by XORing and computing the Hamming weight (efficiently). Section 4 presents experimental results and in Section 5 we conclude the paper.

## 2 Biometric System Architectures using Homomorphic Encryption

There are basically two different types of architectures described in literature. A simple client-server architecture, where the client presents the sample to the server, which is then compared to one or more templates in the database. The second architecture features an additional matching component (matcher) which separates the matching process from the general client-server communication.

Both architectures obtain their privacy preserving property by distributing the verification/identification process among several independent components and it is essential that no single component is able to track a user and/or can gain any sensitive relationship information between a person's identity and her biometric trait. For the client-server architecture it is thus necessary that the client is also involved in computing the final matching result and therefore it is vital that relevant client-side parts of the calculation are kept secret. The client-server-matcher architecture introduces an additional matching component which reduces the computational workload of the client.

**Client-Server Architectures** Barni et al. [BBC$^+$10] introduce a generic identification protocol for privacy preserving Fingercode authentication employing the Paillier and Elliptic Curve ElGamal encryption schemes. Biometric template matching is done in an encrypted domain on the server which then sends the encrypted results back to the client which decrypts these results using the private key of the underlying homomorphic encryption scheme. Upmanyu et al. [UNSJ10] suggest to utilise the homomorphic properties of RSA. They build on a basic client-server architecture, but additionally rely on a trusted third party enrollment server. During enrollment, the client, which also holds the RSA private key, sends its public key along with its identity and its biometric samples to the enrollment server. The enrollment server in turn uses the provided samples to calculate a so called *authenticating classifier* and an individual threshold. The classifier is then encrypted and passed on to the regular server together with the client's ID and its public key. The actual authentication takes place over a twofold communication phase between the client and the server. Sarier [Sar10] designs a biometric verification protocol based on

a simple client server architecture, where the secret (private) key is not needed and thus is not stored anywhere. As Sarier distinguishes between ordered and unordered sets of biometrics he uses ElGamal for ordered sets and RSA for unordered sets as his underlying homomorphic encryption schemes. The biometric features are separated into stable and a non-stable parts [BSW07]. The security of the design is based on the underlying homomorphic encryption scheme, an associated zero knowledge proof (ZKP), and a tamper proof smart card, which stores the non-stable parts of the user's biometric template along with the parameters of the biometric template extraction method. Kuchi et al. [KNON09] suggest a similar server-client approach using the Fujisaki-Okamoto commitment (which is additively homomorphic) and a cryptographic protocol for proving that a committed value lies in a specific interval without revealing the very value [Bou00]. Again, a tamper proof smart card holds vital information for the matching process, so that there is no need to store the secret (private) key anywhere in the system.

**Client-Server-Matcher Architectures** Bringer et al. [BCI$^+$07] introduce an application of the Goldwasser-Micali cryptosystem to biometric authentication. Their system consists of an authentication server, which stores the user's identity and deals with a user's service requests, a database, which stores the plaintext biometric templates of a user and a matcher, which holds the Goldwasser-Micali public/private key pair. The matcher aids the authentication server in making a decision whether a user's authentication request is accepted or rejected. No secret information storage is required at the client side and with the introduction of the matcher, access to the user's biometric information is effectively limited. The matcher does not store any biometric information, hence, compromise of the server does not leak any information to an outside attacker. Although the biometric templates are stored in plaintext in the database, however, without any relevant identity information, no sensitive relationship information may be obtained if the database is compromised. Lou et al. [LCY09] describe a similar solution for anonymous biometric access control applying the Paillier encryption scheme. In addition to the client and biometric server components they utilise a "commodity server" which assists the biometric server in the matching process. However, no information about the templates and the sample is leaked to the commodity server and the actual identification is performed with the help of privately computing the Hamming distance. Although the Hamming distance is computed entirely in the encrypted domain, the enrolled templates are stored in plaintext in the database. The outcome of matching is a single bit which is passed from the commodity server to the biometric server indicating if the user is legitimate (this is done in multiple rounds to prevent attacks). Rane et al. [RSV09] propose another (similar) identification protocol employing the Paillier encryption system. Barbosa et al. [BBCS08] present a secure biometric verification scheme which relies on Support Vector Machine (SVM) classifiers instead of the Hamming distance. Again the verification server (matcher) owns the private key of the underlying homomorphic cryptosystem, in this case the Paillier encryption scheme. Privacy is ensured since no user identities are transmitted at any point in the protocol execution.

## 2.1 The Architecture Considered

Our architecture consists of two parts, the enrollment phase and the identification phase. For both phases, we require a tamper proof sensor which provides the system with samples of the target trait. We also presume that a feature vector (template) of each sample is extracted and is available to the system.

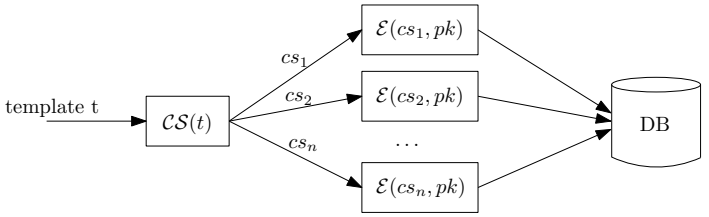



Figure 1: Enrollment architecture (see Section 3 for symbol definitions)

For the enrollment phase we capture and store one (or several) templates of each individual to be recognised by the system. For iris templates (as used in Section 4.2) we additionally define a function $\mathcal{CS}$ which takes a template $t$ as its parameter and performs cyclic shifts over that template. These cyclic shifts are used to balance possible inconsistencies while capturing the sample thus fundamentally improving the matching rates [RUW13]. As we can see from Figure 1, $n$ denotes the number of shifted templates stored in the database.

Each of the resulting reference templates $cs_1, \ldots, cs_n$ is then encrypted and stored in the database. Dependent on the cryptosystem encryption is performed using either the Goldwasser-Micali or the Paillier public key. Furthermore, if we use the conventional Paillier encryption scheme (see Section 3.1), also the plaintext template must be stored in the database (since it is required for the encrypted $xor(\oplus)$ operation).

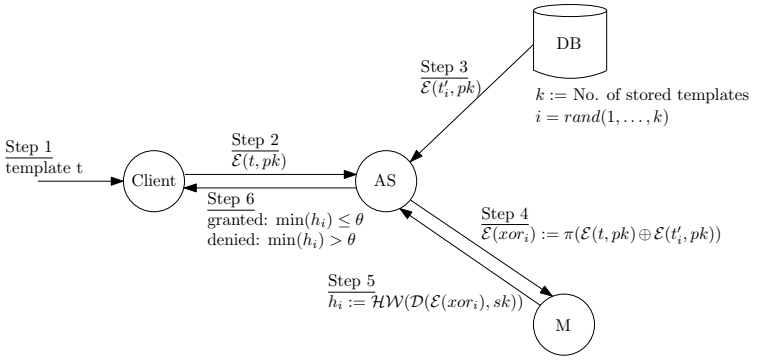



Figure 2: Identification architecture (see Section 3 for symbol definitions)

In the identification phase (depicted in Figure 2), the authentication request is initiated at the client side by extracting a template from the sample of the individual requesting authentication. This template $t$ is encrypted with the according public key at the client side

and is transfered to the server component. The server component consists of three independent parts, the authentication server $AS$, the database server $DB$ (storing the encrypted reference templates) and a matcher $M$.

The authentication server $AS$ deals with the client's service request and fetches all enrolled templates $t'_i, i = 1, \ldots, k$ in random order from the $DB$, where $k$ denotes the number of templates stored in the database. Each enrolled template is then $xor$-ed against the encrypted template $t$ provided by the client. Regardless of the underlying $xor$-algorithm, the calculation of the Hamming weight cannot be accomplished in the encrypted domain using partially homomorphic encryption techniques (see Section 3). For this purpose, the third server component (the matcher $M$), being *honest-but-curious* and in possession of the private key, is employed to compute the Hamming weight from the encrypted binary strings. It has to be ensured that $M$ cannot learn anything about template relations from the binary strings, since being eventually able to eavesdrop the template data being transmitted from the client (encrypted with $M$'s public key), $M$ would get enough information to profile and track persons against their will. Thus, these strings (resulting from the $xor$ operations) are permuted. Permutation is done using a permutation function $\pi$ and bitstrings are sent to the matcher $M$. As the permutation does not change the number of zeros and ones in the binary string, the permutation has no effect on the resulting Hamming weight. The matcher decrypts the permuted string using the private key and calculates the Hamming weight $h_i$, which is sent back to $AS$. If the minimum of all Hamming weights $h_1, \ldots, h_k$ falls within a predefined threshold $\theta$, access to the system is granted, otherwise access is denied. There are also techniques for secure comparisons without revealing any of the involved quantities to either $AS$ or $M$ [DGK08, LCY09, BCP13], but these are not considered in our experiments.

## 3 XORing Homomorphicly Encrypted Binary Feature Vectors

Homomorphic encryption schemes are special cases of asymmetric cryptosystems. As in asymmetric systems there is a public key which is used for encryption and a private key for decryption. Additionally, specific algebraic operations performed on a plaintext are equivalent to other (possibly different) algebraic operations performed on the ciphertext. These encryption schemes can be additively and/or multiplicatively homomorphic.

For an additively homomorphic property an algebraic operation on ciphertext values corresponds to an addition of the corresponding plaintext values. Multiplicative homomorphism generally means that an algebraic operation on ciphertext values relates to a multiplication of the plaintext values.

If a cryptosystem is either additively or multiplicatively homomorphic (not both) it is called a partially homomorphic system (e.g. Unpadded RSA, ElGamal, Paillier, Goldwasser-Micali), whereas schemes which are both additively and multiplicatively homomorphic are called fully homomorphic (see e.g. [Gen09]).

### 3.1 Paillier and Goldwasser-Micali Encryption

For both probabilistic cryptographic techniques, we briefly discuss the respective algorithms for public and private key generation ($\mathcal{K}_P$ and $\mathcal{K}_{GM}$), data encryption ($\mathcal{E}_P$ and $\mathcal{E}_{GM}$), and decryption ($\mathcal{D}_P$ and $\mathcal{D}_{GM}$) together with their homomorphic properties. First, two large prime numbers $p$ and $q$ are generated and the product $n = pq$ is computed.

**Paillier**: For key generation, $\lambda = lcm(p-1, q-1)$ ($lcm$ = least common multiple) is computed and a random integer $g \in \mathbb{Z}^*_{n^2}$ is selected. In our implementation we choose for simplicity $g = n + 1$, as this ensures that the integer $g$ has the required property, i.e. $n + 1 \in \mathbb{Z}^*_{n^2}$ [Pai99]. These steps result in the following public/private key pair: $pk_P$: $(n, g)$ and $sk_P$: $(\lambda)$.

For encryption, we take a message $m$ with $m \in \mathbb{Z}_n$ and select a random integer $r \in \mathbb{Z}_n$. The ciphertext $c$ is obtained by computing

$$c = g^m \cdot r^n \bmod n^2 \ .$$

Note that the ciphertext is modulated by $n^2$ which increases the bit-size of the ciphertext compared to the plaintext at least by a factor of 2. For decryption, the plaintext message $m$ is restored by calculating

$$m = \frac{L(c^\lambda \bmod n^2)}{L(g^\lambda \bmod n^2)} \bmod n,$$

where $L(u) = \frac{u-1}{n}$. In our implementation we use the following equivalence:

$$m = L(c^\lambda \bmod n^2) \cdot \lambda^{-1} \bmod n$$

The Paillier encryption scheme allows two operations in the encrypted domain due to its additively homomorphic property. For any messages $m_1, m_2 \in \mathbb{Z}_n$:

$$\mathcal{D}_P(\mathcal{E}_P(m_1) \cdot \mathcal{E}_P(m_2) \bmod n^2) = m_1 + m_2 \bmod n$$

$$\mathcal{D}_P(\mathcal{E}_P(m_1)^{m_2} \bmod n^2) = m_1 \cdot m_2 \bmod n$$

**Goldwasser-Micali**: For key generation, a quadratic non-residue $x$ is required such that its Jacobi symbol $\left(\frac{x}{n}\right)_J$ equals $+1$ [GM82]. If $p$ and $q$ are selected to ensure that $n$ is a Blum integer then $x = n - 1$ is guaranteed to have this property. These steps result in the following public/private key pair: $pk_{GM}$: $(x, n)$ and $sk_{GM}$: $(p, q)$.

For encryption, a message $m$ is interpreted as a string of bits, i.e. $m = (m[1], \ldots, m[k])$, $m[1], \ldots, m[k] \in \{0, 1\}$, where $k$ represents the number of bits in the message $m$. The ciphertext $c = (c[1], \ldots, c[k])$ is computed by

$$c[i] = y_i^2 x^{m[i]} \bmod n, i = 1, \ldots, k$$

where $y_i \in \mathbb{Z}^*_n$ is picked randomly for each $m[i]$. Due to the bit by bit encryption, the encryption algorithm leads to a substantial data expansion of the ciphertext. For decrypting

the ciphertext $c = (c[1], \ldots, c[k]), k = |c|$ we need to determine if $c[i]$ is a quadratic residue $\mathrm{mod} n$ for $i = 1, \ldots, k : m[i] = \begin{cases} 0, & c[i] \text{ is a quadratic residue } \mathrm{mod} n \\ 1, & c[i] \text{ is a quadratic non-residue } \mathrm{mod} n \end{cases}$

For this decision, knowing the values of $p$ and $q$, the following relation is exploited [GM82]: Let $c_p = c[i] \bmod p$ and $c_q = c[i] \bmod q$. Then

$$\left(\frac{x}{p}\right)_J = c_p^{\left(\frac{p-1}{2}\right)} \bmod p \text{ and } \left(\frac{x}{q}\right)_J = c_q^{\left(\frac{q-1}{2}\right)} \bmod q$$

If $c_p^{\left(\frac{p-1}{2}\right)} \equiv 1 \bmod p$ and $c_q^{\left(\frac{q-1}{2}\right)} \equiv 1 \bmod q$, then $c[i]$ is a quadratic residue $\bmod\ n$.

The additively homomorphic property for any $m_1, m_2 \in \{0, 1\}$ is ($\oplus$ denoting $xor$)

$$\mathcal{D}_{GM}(\mathcal{E}_{GM}(m_1) \ \cdot \ \mathcal{E}_{GM}(m_2)) = m_1 \oplus m_2$$

In other words, if $c_1$ and $c_2$ are the encrypted values of $m_1$ and $m_2$, $(c_1 \cdot c_2) \bmod n$ will be an encryption of $m_1 \oplus m_2$.

### 3.2 Customising Paillier Encryption

To compare two binary biometric templates $m_1$ and $m_2$ we usually calculate the Hamming distance $h$ of the two binary strings by $xor$-ing $m' = m_1 \oplus m_2$ and then computing the Hamming Weight $HW$ of the resulting string (by essentially counting the 1 bits in $m'$): $h = \mathcal{HW}(m')$.

Hence, in order to compute $h$ in a privacy preserving way, we first need to find a way of $xor$-ing two bit-strings in the encrypted domain. For the Goldwasser-Micali encryption scheme we can directly exploit its homomorphic property as described above. For the Paillier cryptosystem however, calculating the $xor$ of two encrypted binary strings is not as trivial. Thus we will have to look at the process of $xor$-ing two bit-strings more closely.

Let $m_1 = (m_1[1] \ldots m_1[k]), m_2 = (m_2[1] \ldots m_2[k])$ be two binary strings of length $k$. Then

$$m_1 \oplus m_2 = m_1[i] + m_2[i] - 2m_1[i]m_2[i], i = 1, \ldots, k\,.$$

As a consequence, we have to use bit-by-bit encryption of the Paillier scheme and can finally perform the encrypted $xor$ as follows:

$$\tilde{m_2}[i] = -2m_2[i] \bmod n$$

$$\mathcal{E}_P(m_1[i] \oplus m_2[i]) = \mathcal{E}_P(m_1[i]) \cdot \mathcal{E}_P(m_2[i]) \cdot (\mathcal{E}_P(m_1[i]))^{\tilde{m_2}[i]} \bmod n^2$$

where $n$ is part of the public key for the Paillier cryptosystem and all encryption steps also use the public key.

However, encrypting only a single bit is very inefficient as the Paillier scheme is designed to encrypt messages of length $m < n$. To improve performance we propose a different

algorithm for $xor$-ing two binary $m_1, m_2$ in the encrypted domain which exploits Paillier's property to encrypt messages of length $m < n$. For the remainder of this paper we refer to this algorithm as *Paillier Chunkwise*.

**Corollary 1** *Before encrypting $m_1$ and $m_2$, these messages are up-sampled as follows:*
$m_{j,up}[i] = \begin{cases} m_j[\frac{i}{2}], & 2 \mid i \\ 0, & otherwise \end{cases} \quad i = 1 \ldots 2 \cdot length(m),\ j \in \{1, 2\}.$

*Then the encrypted result of $xoring$ $m_1$ and $m_2$ can be computed as (for $i = 1 \ldots length(m)$):*

$$\mathcal{E}_P(m_1[i] \oplus m_2[i]) = (\mathcal{E}_P(m_{1,up}) \cdot \mathcal{E}_P(m_{2,up}) \bmod n^2)[2i] \ .$$

*This means that when upsampling the binary data of the $m_j$ before encryption, the binary positions of the result of the multiplication of the decimal numbers corresponding to upsampled $m_j$ are equal to the encrypted result of $xor$-ing the $m_j$'s binary positions.*

**Proof**: The operation supported by the homomorphic property by the Paillier scheme is the addition in the non-encrypted domain. Thus, to be able to exploit this, the binary representation of the result of the addition of two decimal numbers should be related to the binary result of bitwise $xor$-ing the binary representation of the two decimal numbers. Indeed we have to show that, if both $m_j$ are up-sampled as described, the even binary positions of the result of the addition of the two decimal numbers corresponding to the $m_{j,up}$ are identical to the bits representing the result of the bitwise $xor$ of the original $m_j$:

$$m_1[i] \oplus m_2[i] = (m_{1,up} + m_{2,up})[2i]$$

Now directly exploiting the Paillier additively homomorphic property leads to the desired result as stated above. **QED**.

**Lemma 1** *Let $m_1[i]$ and $m_2[i]$ be two binary strings, $m_{1,up}$ and $m_{2,up}$ the decimal numbers corresponding to the upsampled versions of $m_1$ and $m_2$. Then*

$$m_1[i] \oplus m_2[i] = (m_{1,up} + m_{2,up})[2i] \ .$$

**Proof**: The addition of two decimal numbers $m_1$ and $m_2$ in their binary representation can be accomplished by $(m_1 + m_2)[i] = m_1[i] + m_2[i]\ \forall i$ and adding some carry bit at the i+1th position if necessary. Due to the definition of the up-sampling process the even positions of the up-sampled sequences $m_{j,up}[2i]$ contain the original bits of $m_j[i]$. The odd positions $m_{j,up}[2i+1]$ are zero and thus, the (binary) sum of $m_{1,up}[2i+1]+m_{2,up}[2i+1]$ is zero as well. Adding the even positions, i.e. $m_{1,up}[2i] + m_{2,up}[2i]$, four cases need to be considered:

1. $m_{1,up}[2i] = 0 \ \wedge \ m_{2,up}[2i] = 0 \implies m_{1,up}[2i] + m_{2,up}[2i] = 0 = m_{1,up}[2i] \otimes m_{2,up}[2i]$
2. $m_{1,up}[2i] = 1 \ \wedge \ m_{2,up}[2i] = 0 \implies m_{1,up}[2i] + m_{2,up}[2i] = 1 = m_{1,up}[2i] \otimes m_{2,up}[2i]$

3. $m_{1,up}[2i] = 0 \wedge m_{2,up}[2i] = 1 \implies m_{1,up}[2i] + m_{2,up}[2i] = 1 = m_{1,up}[2i] \otimes m_{2,up}[2i]$
4. $m_{1,up}[2i] = 1 \wedge m_{2,up}[2i] = 1 \implies m_{1,up}[2i] + m_{2,up}[2i] = 0 = m_{1,up}[2i] \otimes m_{2,up}[2i]$ and a carry bit has to be set resulting in $m_{1,up}[2i+1] + m_{2,up}[2i+1] = 1$. Since originally all odd positions are zero, there is no further carry bit and thus no further influence on positions [i+2].

Thus all four cases lead to the desired result. **QED**.

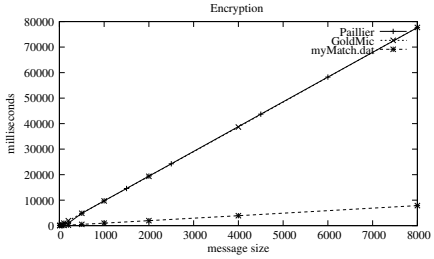
Regardless of the underlying $xor$-algorithm, the calculation of $HW$ cannot be accomplished in the encrypted domain. As described before, this is done on the matcher $M$, using the private key of the employed respective encryption scheme. The bitwise permutation of the binary string $m'$ to ensure privacy when transfered to the matcher can be conduced for Paillier and Goldwasser-Micali encryption in straightforward manner, however, for Paillier Chunkwise the permutation can only be carried out on the encrypted chunks. Therefore, the privacy preserving property of the Paillier Chunkwise encryption scheme declines the smaller the input data and therefore the fewer chunks are available for permutation (thus, this issue needs to be considered when setting up the system, where we face a tradeoff between efficiency (large chunks) and privacy (small but numerous chunks)).

# 4 Experiments

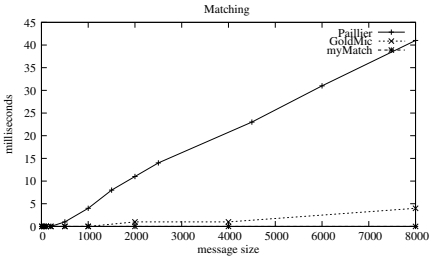
## 4.1 Assessment of Privacy-preserving Components

All algorithms are implemented in C using the GNU MPL. First experiments are executed on an Intel Core Duo T2050 with 1600MHz. Time for key generation is not considered since it can be done in a preprocessing stage, the prime numbers used for key generation have 25 bits each (to limit experimentation effort) and chunk-size is set to 80 bits. We use random binary $m_j$ and repeat each experiment 40 times averaging the results. Message size is given in bits in the plots. For all three encryption techniques (i.e. Paillier (P), Goldwasser-Micali (GM), Pallier Chunkwise (PC), denoted as "myMatch" in the plots), we compare timings of encrypting a feature vector (including up-sampling for PC and done on the client), timings of matching the binary strings in the encrypted domain (done on the $AS$), and timings of decrypting, summing the set bit positions and comparisons to a threshold (including the extraction of the even bit positions for PC and done on the $M$).

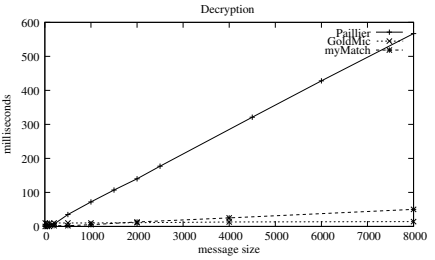
While encryption is very time consuming (and nearly identical expensive, see Fig. 3.a) for P and GM due to the required bit-by-bit operation mode, PC is much more efficient. For matching, P is clearly least efficient as expected due to the computation of the inverse element and the higher number of required multiplications. GM is less efficient compared to the proposed PC since more multiplications are required (in PC more bits are tracted by a single multiplication). Finally, for decryption, GM is the fastest technique since determining if a number is a quadratic residual can be done very efficiently using the MPL. On the other hand, the extraction of single bits as required by PC is rather inefficient, thus,




(a) Encryption




(b) Matching




(c) Decryption

Figure 3: Timings for different processing stages

PC is ranked second in this stage. The bit-by-bit decryption as done by P is clearly the least efficient approach.

### 4.2 Iris Recognition in the Encrypted Domain

In order to experimentally evaluate the overall performance in realistic actual use for privacy-preserving biometric matching, we set up an iris identification experiment using data from the CASIA V3 Interval database. For these experiments an Intel Core i7-2600 4x3400MHz PC with 4GB DDR3 RAM is used and key length / chunk-size are set to 1024 / 31 bits, respectively. For feature extraction and matching, software from the University of Salzburg USIT Toolkit is used (available from `www.wavelab.at/sources/Rathgeb12e/` [RUW13]) implementing a context adaptive Hough Transform for segmentation and the algorithm of Ma et al. [MTWZ04] for extracting binary templates of length 10240 bits. All the templates of a subject are stored (in pre-encrypted manner) in the DB using 17 different cyclic shifts. In Table 1 we present timings of identification attempts including encryption of the template to be identified, matching in the encrypted domain, and final decryption and comparison of HW result to a threshold.

In the table, we display timing results of single identification attempts in which either only a single person was enrolled in the $DB$ (thus simulating the computational cost of a verification procedure) or 30 subjects were enrolled. As already to be expected from the results of the last subsection, PC significantly outperforms both P and GM: The proposed

Table 1: Time performance (seconds): Iris Identification

| Scheme | verification | 30 subjects enrolled |
|---|---|---|
| Unencrypted | $< 0.1$ | 2.5 |
| Goldwasser-Micali (GM) | 22.8 | 682.0 |
| Paillier (P) | 5591.9 | 162000.0 |
| Paillier Chunkwise (PC) | 3.0 | 90.0 |

scheme is faster by a factor of more than 7 as compared to GM and faster by several orders of magnitude compared to P. However, inspite of these improvements PC is still slower by a factor of 36 as compared to unencrypted application.

## 5 Conclusions

We have proposed a variant of Paillier homomorphic encryption clearly outperforming the Goldwasser-Micali approach in privacy-preserving biometric matching of binary templates. While obtaining significant improvements, the increase in computational cost as compared to the unencrypted case is still too large for large scale deployment.

## References


[AMFF+13] C. Aguilar-Melchor, S. Fau, C. Fontaine, G. Cogniat, and R. Sirdey. Recent advances in homomorphic encryption. *IEEE Signal Processing Magazine*, 2(30):108–117, 2013.

[BBC+10] M. Barni, T. Bianchi, D. Catalano, M. Di Raimondo, R. Donida Labati, P. Failla, D. Fiore, R. Lazzeretti, V. Piuri, F. Scotti, and A. Piva. Privacy-preserving fingercode authentication. In *Proceedings of the 12th ACM Workshop on Multimedia and Security*, MM&Sec '10, pages 231–240, New York, NY, USA, 2010. ACM.

[BBCS08] M. Barbosa, T. Brouard, S. Cauchie, and S. M. Sousa. Secure Biometric Authentication with Improved Accuracy. In *Proceedings of the 13th Australasian Conference on Information Security and Privacy*, ACISP '08, pages 21–36, Berlin, Heidelberg, 2008. Springer-Verlag.

[BCI+07] J. Bringer, H. Chabanne, M. Izabachène, D. Pointcheval, Q. Tang, and S. Zimmer. An application of the Goldwasser-Micali cryptosystem to biometric authentication. In *Proceedings of the 12th Australasian Conference on Information Security and Privacy*, ACISP'07, pages 96–106, 2007. Springer.

[BCP13] J. Bringer, H. Chabanne, and A. Patey. Privacy-preserving biometric identification using secure multiparty computation. *IEEE Signal Processing Magazine*, 2(30):42–52, 2013.

[Bou00] Fabrice Boudot. Efficient proofs that a committed number lies in an interval. In *Proceedings of the 19th International Conference on Theory and Application of Cryptographic Techniques*, EUROCRYPT'00, pages 431–444, 2000. Springer.

[BSW07] T. E. Boult, W. J. Scheirer, and R. Woodworth. Revocable Fingerprint Biotokens: Accuracy and Security Analysis. *Computer Vision and Pattern Recognition, IEEE Computer Society Conference on*, pages 1–8, 2007.

[DGK08] Ivan Damgard, Martin Geisler, and Mikkel Kroigard. Homomorphic encryption and secure comparison. *International Journal of Applied Cryptography*, 1(1):22–31, 2008.

[Gen09] Craig Gentry. Fully homomorphic encryption using ideal lattices. In *Proceedings of the 41st annual ACM Symposium on Theory of Computing*, STOC '09, pages 169–178, New York, NY, USA, 2009. ACM.

[GM82] Shafi Goldwasser and Silvio Micali. Probabilistic encryption & how to play mental poker keeping secret all partial information. In *Proceedings of the fourteenth annual ACM Symposium on Theory of Computing*, STOC '82, pages 365–377, New York, NY, USA, 1982. ACM.

[KNON09] H. Kikuchi, K. Nagai, W. Ogata, and M. Nishigaki. Privacy-preserving similarity evaluation and application to remote biometrics authentication. *Soft Comput.*, 14:529–536, December 2009.

[LCY09] Y. Luo, S. S. Cheung, and S. Ye. Anonymous biometric access control based on homomorphic encryption. In *Proceedings of the 2009 IEEE International Conference on Multimedia and Expo*, ICME'09, pages 1046–1049, Piscataway, NJ, USA, 2009. IEEE Press.

[MTWZ04] L. Ma, T. Tan, Y. Wang, and D. Zhang. Efficient iris recognition by characterizing key local variations. *IEEE Transactions on Image Processing*, 13:739–750, 2004.

[Pai99] Pascal Paillier. Public-key cryptosystems based on composite degree residuosity classes. In *Proceedings of the 17th International Conference on Theory and Application of Cryptographic Techniques*, EUROCRYPT'99, pages 223–238, Berlin, Heidelberg, 1999. Springer-Verlag.

[RSV09] S. D. Rane, W. Sun, and A. Vetro. Secure distortion computation among untrusting parties using homomorphic encryption. In *Proceedings of the 16th IEEE International Cnference on Image processing*, ICIP'09, pages 1469–1472, Piscataway, USA, 2009. IEEE Press.

[RU11] Christian Rathgeb and Andreas Uhl. A Survey on Biometric Cryptosystems and Cancelable Biometrics. *EURASIP Journal on Information Security*, 2011(3), 2011.

[RUW13] Christian Rathgeb, Andreas Uhl, and Peter Wild. *Iris Recognition: From Segmentation to Template Security*, volume 59 of *Advances in Information Security*. Springer, 2013.

[Sar10] N.D. Sarier. Practical multi-factor biometric remote authentication. In *Biometrics: Theory Applications and Systems (BTAS), 2010 Fourth IEEE International Conference*, BTAS '10, pages 1–6, September 2010.

[SSW10] A.-R. Sadeghi, T. Schneider, and I. Wehrenberg. Efficient privacy-preserving face recognition. In *Proceedings of the International Conference on Information, Security, and Cryptology (ICISC'09)*, volume 5984 of *Springer LNCS*, pages 229–244, 2010.

[UNSJ10] M. Upmanyu, A. M. Namboodiri, K. Srinathan, and C. V. Jawahar. Blind authentication: a secure crypto-biometric verification protocol. *IEEE Trans. Info. For. Sec.*, 5:255–268, June 2010.