




# Secure Processing of Biometric Signals in Malicious Setting



Giulia Droandi

Ph.D Thesis in Engineering and Science of Information  
University of Siena





UNIVERSITÀ DEGLI STUDI DI SIENA  
FACOLTÀ DI INGEGNERIA  
DIPARTIMENTO DI INGEGNERIA DELL'INFORMAZIONE E SCIENZE  
MATEMATICHE



# Secure Processing of Biometric Signals in Malicious Setting

Giulia Droandi

*Ph.D Thesis in Engineering and Science of Information  
XXVII cycle*

*Supervisor*

Prof. Mauro Barni

*Co-Supervisor*

Riccardo Lazzeretti

*External reviewers*

Prof. Giuseppe Bianchi

Prof. Fabio Scotti

*Examination Committee*

Prof. Giuseppe Bianchi

Prof. Simone Rinaldi

Prof. Fabio Scotti

---

SIENA  
OCTOBER 19, 2018

---

# Contents

<b>Acknowledgements</b>	<b>xi</b>
<b>Abstract</b>	<b>xiii</b>
<b>1 Introduction</b>	<b>1</b>
1.1 Biometric recognition . . . . .	1
1.2 Privacy and security . . . . .	4
1.3 Signal processing in the encrypted domain . . . . .	6
1.4 State of the art . . . . .	7
1.5 Contributions . . . . .	9
1.6 Publication list . . . . .	11
1.7 Outline . . . . .	11
<b>2 Cryptographic Tools</b>	<b>13</b>
2.1 Security Models . . . . .	14
2.2 Basic SMPC Primitives . . . . .	16
2.2.1 Oblivious transfer . . . . .	16
2.2.2 Garbled circuits . . . . .	17
2.2.3 Secret sharing . . . . .	19
2.3 Homomorphic encryption . . . . .	19
2.3.1 Somewhat and fully homomorphic encryption . . . . .	21
2.3.2 Somewhat homomorphic encryption on integers . . . . .	22
2.4 SPDZ . . . . .	26

2.5	Hybrid protocols . . . . .	33
<b>3</b>	<b>Privacy Preserving Biometrics Matching</b>	<b>35</b>
3.1	Biometric recognition in plain domain . . . . .	35
3.2	Iris recognition in the encrypted domain . . . . .	38
3.3	Face recognition in the encrypted domain . . . . .	42
3.4	Fingerprint recognition in the encrypted domain . . . . .	45
3.5	Multimodal Biometrics . . . . .	47
3.6	Biometric recognition in malicious setting . . . . .	51
<b>4</b>	<b>Biometrics Recognition Based on Somewhat Homomorphic Encryption</b>	<b>53</b>
4.1	Representing negative numbers in SHE . . . . .	55
4.2	General Protocol . . . . .	56
4.2.1	Authentication protocol . . . . .	56
4.2.2	Identification protocol . . . . .	57
4.3	Computation of the Hamming distance under SHE . . . . .	58
4.4	Computation of the Squared Euclidean distance under SHE . . . . .	60
4.5	Blinding and comparison . . . . .	61
4.6	Complexity of the proposed protocols . . . . .	63
4.7	Experimental results . . . . .	64
4.7.1	Authentication protocol . . . . .	66
4.7.2	Identification Protocol . . . . .	68
<b>5</b>	<b>Multi-Modal Biometrics authentication in the Malicious Setting</b>	<b>71</b>
5.1	Iris authentication protocol in SPDZ . . . . .	74
5.1.1	Hamming distance and comparison in SPDZ . . . . .	74
5.1.2	Complexity . . . . .	75
5.2	Face authentication protocol in SPDZ . . . . .	76
5.2.1	Euclidean distance in SPDZ . . . . .	76
5.2.2	Complexity . . . . .	77
5.3	SEMBA: the fusion protocol in SPDZ . . . . .	77
5.3.1	Complexity of SEMBA . . . . .	79
5.3.2	Security . . . . .	80

## Contents

---

5.4	Preliminary tests in plain . . . . .	81
5.4.1	Iris . . . . .	82
5.4.2	Face . . . . .	83
5.4.3	Multimodal system . . . . .	85
5.5	Results of tests on SEMBA . . . . .	86
<b>6</b>	<b>Conclusion</b>	<b>91</b>
6.1	Highlights for future reserch . . . . .	92
	<b>Bibliography</b>	<b>95</b>
	<b>Index</b>	<b>109</b>



---

## List of Figures

3.1	Biometric recognition system main steps. . . . .	36
3.2	Authentication protocol . . . . .	37
3.3	Identification protocol . . . . .	37
3.4	Runtime complexity of eigenfaces . . . . .	43
3.5	The possible sources of information in a multibiometric system. . . . .	49
4.1	Secure biometric recognition system . . . . .	54
4.2	General privacy protection protocol in SHE . . . . .	57
4.3	Authentication protocol in SHE . . . . .	58
4.4	Identification protocol in SHE . . . . .	59
5.1	General scheme of our fusion protocol. . . . .	78





---

## List of Tables

2.1	Parameters recap . . . . .	24
2.2	Linear operations in SPDZ . . . . .	28
2.3	Computational complexity in SPDZ . . . . .	33
3.1	Performances in [1] . . . . .	40
3.2	Complexity of privacy preserving face recognition in [2] . . . . .	43
3.3	Online performances in [3] . . . . .	46
4.1	Sizes of cryptosystem parameters . . . . .	64
4.2	The averaged time results needed for a single encryption, decryption, key generation and multiplication. . . . .	65
4.3	Complexity in authentication protocol . . . . .	66
4.4	Iris protocol execution time . . . . .	67
4.5	Fingerprint protocol execution time . . . . .	68
4.6	Identification protocol execution time . . . . .	69
5.1	Correspondence table between binary and integer operations . . . . .	75
5.2	SPDZ complexity . . . . .	80
5.3	Iriscode EER in plain . . . . .	83
5.4	Eigenfaces EER in plain . . . . .	84
5.5	Multimodal EER in plain . . . . .	84
5.6	EER dependance from $\alpha$ variation in plain . . . . .	85
5.7	Setting of multimodal protocol . . . . .	87

5.8	Communication complexity . . . . .	88
5.9	Iris protocol time in SPDZ system. . . . .	88

---

## List of Algorithms

2.1	SPDZ MACCheck . . . . .	29
2.2	SPDZ multiplication . . . . .	30
2.3	SPDZ square . . . . .	30
2.4	SPDZ comparison . . . . .	32



---

## Acknowledgements

I'd like to thanks Francesco, without whom I would have not finished this work. I take this opportunity to express gratitude to my PhD supervisor Mauro Barni and to Riccardo Lazzeretti for their help and support.



---

## Abstract

In the digital and interconnected world we live in, establishing the identity of any individual is a pressing need. Home banking, on line shopping, and social care web sites are only few examples of services where proof of identity is fundamental. Such a process can be based on *what you know* (i.g. a password), on *what you possess* (i.g. the key of a house or an ID card) or on *what you are* (ID-based, i.g. biometrics). In this thesis we focus on biometrics. Biometric recognition, or simply biometrics, refers to “the automated recognition of individuals based on behavioral and biological characteristics” (ISO/IEC JTC1 SC37). This method of recognition has the advantage that it does not need the memorisation of any password or the possession of any token, at the same time, however, biometrics cannot be changed if compromised in any way, hence calling for the adoption of suitable protection mechanisms.

In this thesis we study the development of privacy preserving protocols for biometric recognition. This is a new research field for which a number of solutions have been proposed in recent years. For efficiency reasons, the majority of those solutions are secure only against a *passive* adversary, that is an adversary that does not deviate from the protocol, yet tries to infer as much information as possible from the data exchanged during the protocol.

On the contrary, in this thesis we look for protocols which are secure against *active* adversaries, that is adversaries that deliberately and arbitrarily deviate from the recognition protocol. Specifically, we propose two possible solutions using signal processing in the encrypted domain’s tools.

First we use a cryptographic scheme belonging to the somewhat homomor-



phic scheme's family and we propose both an *identification* and an *authentication* non-interactive scheme. The first protocol focuses on a one-to-many recognition task: the biometric probe of a specific individual is compared with all the probes contained in a database looking for a positive match. The second protocol, instead, considers a one to one comparison. The new probe of an enrolled individual is compared with the probe of the same individual stored during the enrollment phase.

As a second contribution, we propose SEMBA: a protocol secure against active adversary for multibiometric recognition. In this case we look for a trade-off between efficiency and accuracy by combining information from two biometric traits instead of only one. The protocol relies on SPDZ, a new framework proposed by Damgård et al. which is secure also in the presence of an active adversary.

### 1.1 Biometric recognition

In the digital and increasingly interconnected world we live in, individuals' identity verification is a pressing need. It is becoming more and more important to establish if someone is in some government watch list (i.g. when requiring a VISA) or if he is authorized to use some facilities. Home banking, on line shopping, and social care web sites are only few examples of services where proof of identity is fundamental.

By identity recognition in the following we mean the process of verifying the identity of a user, a device or another entity in a computer network or system [4]. Passwords, security tokens, and biometrics are collectively called *authenticators* and can be used in a recognition protocol. If someone (let's say Alice) wants to log into a system, she has three possible ways to demonstrate her identity. First, identification can be based on *what Alice knows* (knowledge-based recognition). This modality includes passwords and PINs (Personal Identification Numbers). A drawback of this solution is that passwords can be stolen or forgotten and, if shared with someone, become less secure. Moreover an individual usually chooses a password easy to remember but at the same time easy to guess. Second, the system can authorize Alice on the basis of *what she possesses* (object-based). Everyday examples are the key of a house or an ID card. On one hand those items (e.g. house's key) can be lost and whoever found them can enter the service (i.e the house); on the other hand when the owner is aware of the loss, he can change the token (in our example, the lock). Lastly, the system can verify Alice's identity on the base of *what she is* (ID-based). An example is biometric recognition but also driver's license, passport, credit card; those last examples are not directly *Alice* but are based on what she is. The security of those authenticators is

based on the difficulty to forge them [4].

Biometric recognition or simply biometrics [5] refers to “the automated recognition of individuals based on behavioral and biological characteristics” (ISO/IEC JTC1 SC37). Many physical traits (e.g. iris, fingerprints, face, etc.) or behavior characteristic (e.g. gait, signature, etc.) have been used for identity recognition [6]. Even if at the beginning biometrics had been mainly used by law enforcement for identifying criminals (e.g. secure identification of convicts, security clearance for particular jobs, forensics, etc.), in recent years, biometric recognition systems inspired an increasing interest also for civilian applications, since in this way it is possible to establish an identity based on *who you are*, rather than on *what you possess* or *what you can remember* [5,7].

Virtually, any human characteristic can be chosen as biometrics but it should respect some requirements like universality (every one should have it), distinctiveness (it should be different enough to distinguish one person from another), permanence (the characteristic should not change over time), etc. (more details in Section 3.1). In summary the trait should be at the same time easy to acquire, enough different from a person to another and everyone should have it [5].

Any biometric recognition system starts with the acquisition of biometric rough data, then the raw data is processed, and a template is extracted. Then it is possible to match two templates of the same biometric by using a decision function. A match score, usually using Hamming or Euclidean distance, is calculated and then the resulting score is then compared with a certain threshold depending on the function and on the biometrics.

There are two possible scenarios. In the first one, named *identification*, the system, possibly run by a government authority, is interested to know who the biometric template’s owner is between all the individuals in a database of people (for example the database of most wanted criminals, or in the AFIS, Automated Fingerprint Identification System). In identification mode the system validates the person’s identity conducting a one-to-many research in the database. The system looks for a positive match by comparing the new probe with all the others in its database. Identification is particularly important in *negative recognition* applications where the systems establishes whether the person is who denies to be. In this way it is possible to prevent a single

person from using multiple identities. In the second mode, *authentication* or *verification*, the system validates the person identity by comparing the new probe with the one associated to the person and stored into its database. In such scenario an individual claims to be someone using a username, an ID card, or something else, and the system conducts a one-to-one comparison in order to decide if the user is who he claims to be [5].

In an intermediate task, we would only know if someone is or not in a database of people. This modality differs from identification because it has a yes or not answer, while identification's output is the identity of the client.

Using biometrics as recognition token, requires that we also consider biometric errors. Any user can forget or mistype a password, this is an inconvenient resolvable by typing again or requesting a new password. If the error is not due to user fault and he is unable to solve the problem, the situation is more upsetting. In biometric protocols, a dirty capture device, poor lightning, environmental factors are only few example of possible causes of biometric errors [4]. Such errors are usually quantified by two metrics: *False Acceptance Rate* (FAR) and *False Rejection Rate* (FRR). FAR is the likelihood that the biometric recognition system will incorrectly grant an access to an unauthorized user, i.e. it measures the number of false positive errors. FRR is the number of authorized user's instances the system fails to verify, i.e. the number of times the system denies access to a legitimate user. Both errors represent the vulnerability of the system. In a verification task, FRR represents the inconvenience of being incorrectly rejected by the system, while FAR measures how easily an attacker can impersonate a legitimate user [4]. On the contrary, if we want only to know if someone is listed in a database and not who he is, for example in order to grant access to a certain service, FRR measures the vulnerability of the system due to not identify someone in the database while FAR indicates the inconvenience of been misidentified [4]. To determine the accuracy of a biometric system, the *Equal Error Rate* (EER) is also used, it correspond to the error rate when FAR is equal to the FRR. The lower the equal error rate value, the higher the accuracy of the biometric system.

## 1.2 Privacy and security

An authenticator (biometric or not) is effective, beside its relevance to a particular application, if it can resist various types of attacks. Even the more secure system can be attacked by exploiting a human mistake, like writing the password on a piece of paper or choosing a password easy to guess [4]. More in general, authenticators can be attacked at three locations: at the client (i.e. guessing the password), in the transmission channel and at the authentication server.

A biometrical authenticator has the advantage that it does not need to be memorized, but the resulting template (usually a numeric vector) can be guessed as any other password. Moreover biometrics cannot be replaced an unlimited number of times if they are compromised, since they are inherent parts of a person body. At the same time biometrics security does not depend on secrecy [4]. For example face and voice are not secret, while it is difficult to keep fingerprints and iris hidden from anybody. Biometric recognition systems are like the number on a driver's license. What's make it secure is not the number itself but the difficulty to forge the original document [4]. So the security of a biometric system relies on how hard it is to forge a template. Anyway, those systems can incur in several attacks, some of them can be overcome by using cryptography, while others require some security standards of the system.

An attacker (Eve) might try to gain the biometric of a legitimate user (Alice) by attaching a fake biometric capture device to the system. This fake reader, instead of measuring the traits, will only save the biometric and therefore, in the future, Eve could be able to input Alice's captured data as it were her own [8]. If templates are transmitted from a location to another one through an exposed communication link, Eve could masquerade as Alice by simply replacing Alice's template, by intercepting the signal, or by tampering with the templates, if Eve can access directly to them. This kind of violations could be prevented by resorting to cryptography. For example, the capturing device and the system can be provided with a secret value that can authenticate the source of information, or by signing the template in order to prevent any modification during the transmission.

In a different attack, Eve may forge a duplicate of the real user's biomet-

rics. As we have already said, it is hard to hide biometric traits, e.g. it is relatively easy for an attacker to collect fingerprints from a glass and use the information to create a fake finger. Obviously encryption is not the solution in this situation, but biometric readers can be designed so that they can detect fake biometrics [8].

A further possible attack can be engaged by exploiting *collision*. A biometric collision occurs when the template vector of one user, lets call it  $\mathcal{A}$ , is “close enough” to another user’s template,  $\mathcal{B}$ , so that  $\mathcal{A}$ ’s template can be used to authenticate  $\mathcal{B}$  instead. This possibility can be exploited to infiltrate the system. Any biometrics is vulnerable to this attack, because any system of this kind has FAR greater than zero. Non zero false acceptance rate means that there will be somewhere two individuals whose biometric trait are very similar and therefore they can verify against each other’s templates. In a form of this violation Eve attempts to verify her biometric traits against any one of the templates into the database, or Eve can take control of the database and choose the most similar, or she can pursue a trial-and-error attack [8]. Especially for the scenario in which the attacker takes control of the system or obtains a copy of the database, encryption can be a solution to ensure security.

Finally, habits, movements, position and also personal believes can be tracked by observing when and where the biometrics traits are used to identify someone, putting at risk the privacy of the owner. Even worse, if compromised by an adversary, biometrics data can be used to access sensitive information, and to impersonate the victim for malicious purposes, compromising the security of other systems based on the same biometrics.

To protect the system from all the threats, the security of biometric-based systems has recently become a very active research area, due to the necessity of impeding newly emerging cybercrimes like identity theft, privacy violation, unauthorized access to sensitive information and so on [9]. Privacy has to be guaranteed not only versus eavesdroppers but also between the parties to avoid any tracking and logging activity of the user presence. A dishonest party may use the knowledge acquired during the computation to gain some advantage, or worst he may sell those information to someone else. This problem has raised the necessity to process data in a privacy preserving way,

so that the client does not learn anything about the templates stored into the database except for the output, but at the same time the database owner or the server should learn nothing from the new submitted biometric template.

### 1.3 Signal processing in the encrypted domain

Processing data while they are encrypted provides an elegant solution to the problems mentioned before (Section 1.2) [10, 11]. Signal processing in encrypted domain (usually referred as s.p.e.d.) indicates the wide range of techniques allowing to process encrypted signals. As example, let us consider an identification scenario where a service provider has a list of enrolled individuals (the clients who can access a certain service or the criminals contained in a police record). The client would like to know if some biometrics are in the server's database or not, without revealing the result of the query to the database owner. Alternatively the biometric owner desires to access a service without revealing his identity. According to the s.p.e.d. paradigm, the goal mentioned above can be achieved by letting the server carry out the matching process directly on encrypted values. Even if the task seems impossible, a functionality like the one mentioned before can be implemented by resorting to Homomorphic Encryption (HE) and Secure Multi Party Computation (SMPC) techniques. SMPC techniques are a set of protocols built with the only goal to allow two or more individuals or parties to jointly compute a function without revealing each others the inputs. Virtually, any computable function or algorithm can be computed on encrypted data [11]. In a general SMPC setting one party, the client  $\mathcal{C}$ , owns a signal that must be processed in some way by the other party, hereafter referred to as the server  $\mathcal{S}$ .  $\mathcal{S}$  must process  $\mathcal{C}$ 's signal without getting any information about it, in some cases not even the result of the computation. At the same time,  $\mathcal{S}$  is interested to protect the information he uses to process the signal. Using SMPC techniques, it is possible to carry out the match between any two biometric templates and even between a biometric query and the templates stored in a database, by working only on encrypted data. Moreover, it is also possible to construct the underlying matching protocol in such a way that only the intended party knows the final result of the match without revealing any information about

the identity of the biometric owner.

SMPC tools such as oblivious transfer [12], garbled circuit [13] and homomorphic encryption [14, 15] are among the most used in literature. In a nutshell (more details can be found in Chapter 2) oblivious transfer (Section 2.2.1) is a method to exchange secrets in a secure way, garbled circuit (Section 2.2.2) is a protocol for computing encrypted functions, while homomorphic encryption schemes (Section 2.3) allow to compute some specific operations, usually addition or multiplication, on encrypted values. Recently a new set of encryption schemes that can perform both addition and multiplication on encrypted data have been developed. Those systems are indicated as fully homomorphic encryption schemes (Section 2.3). Any s.p.e.d. tool has its strong and weak points, and therefore it is better suited for some specific subprotocols. To better exploit the main strength of each technique, more than one tool is sometimes used in the a system. Each one is used to implement the specific subprotocol for which it is more efficient. Those systems are usually called hybrid (Section 2.5). The main challenge in those situations is to find an appropriate and efficient link function between the used modalities.

Any SMPC tool can be implemented in order to deal with *active* or *passive* adversaries. The first group of attackers actively try to infiltrate the system by corrupting data and cheating, the second ones passively try to gain as much information as possible only observing and participating to the protocol. Usually SMPC tools secure against active adversary require complex and time consuming implementations. Therefore the majority of works studying privacy preserving recognition protocols tools are secure in a semi-honest setting (Section 2.1). Nevertheless, protocol guaranteeing security in the semi-honest model can be always modified to be make them secure under more stringent threat models but at the cost of higher complexity [16–18].

## 1.4 State of the art

To overcome the privacy issues presented in Section 1.2, while at the same time retaining the advantages of biometric recognition systems, many works have recently focused on the privacy protection of biometric signals [10, 11] relying on SMPC techniques. A general recognition protocol is composed of



a few basic blocks, feature extraction, distance computation comparison and minimum selection. Since feature extraction involves only data provided by one party, it is usually implemented in plain domain. On the contrary distance computation, comparison and minimum selection must be implemented resorting to s.p.e.d., since these operations involve private data owned by  $\mathcal{C}$  and  $\mathcal{S}$ . To be implemented using s.p.e.d., a biometric trait should be represented through a vector of features of constant length so that a simple distance measure (e.g. Hamming or Euclidean distance) can be used to measure the degree of similarity between two vectors. Any biometrics satisfying the previous conditions can be developed using simple s.p.e.d. blocks [11, 19, 20].

Despite many recent advances and the introduction of more efficient cryptographic primitives, the complexity of s.p.e.d. based biometric recognition protocols is usually high, preventing their use in real life applications. In fact the most common approach used so far has been taking a biometric matching algorithm and transforming it into a protocol that can be implemented in the encrypted domain. For example in [2, 21–23] authors used multiplicative Homomorphic Encryption to protect recognition protocols, while in [24, 25] and [26] garbled circuits have been used. It is worth mentioning also the works by Yasuda et al. [27] and Troncoso-Pastoriza [28] for biometric recognition using fully homomorphic encryption.

In all the papers cited before authors had to deal with complexity and security. The majority of previous works on the topic are secure only against passive adversaries in order to keep complexity low. It would be better to be able to guarantee security against an active attacker trying to fool the system deviating from the protocol, such as in [29, 30].

The search for efficiency is not limited to the choice of a suitable matching algorithm, biometrics representation should be considered as well. In a s.p.e.d. protocol, the complexity of s.p.e.d. primitives depends on the number of features involved in the matching function and on the number of bits used to represent them. By decreasing the number of features, the efficiency of the implementation increases at the expense of accuracy. It is necessary to find a balance between accuracy and efficiency. For example, in [24] the authors proved that the use of a common mask for iris recognition, dramatically simplifies the implementation of the s.p.e.d. protocol [11]. This last approach

is the one we used in Chapter 5 to find a trade off between accuracy and efficiency in malicious setting. A more detailed review of the state of art can be found in Chapter 3.

For completeness we present other methods for template protection, used as alternative solutions to s.p.e.d. tools. These methods are usually categorized as biometric cryptosystems and cancellable biometrics.

Biometric cryptosystems (BC) were originally developed to secure a cryptographic key using biometric or to generate a key from a biometric template, however they can also be used for template protection [31]. The basic idea is that the biometric is used to authenticate the user and release the secret key, while a standard cryptosystem can secure the information or the communication [32]. Comparing two biometrics therefore means verifying keys validity, while the authentication output could be either a key or a failure message [31].

A solution related to the previous method but not equivalent is cancellable biometrics. In this method the original biometric signal is transformed using a one way function. The distortion can be applied to the original domain or to the feature [33]. It preserves privacy since it is computationally hard to recover the original data from the template and it has also the advantage of revocability, because a template can be re-enrolled using a different transformation [33]. Moreover cancellable biometrics does not compromise the accuracy of a matching algorithm as the statistical characteristics of features are maintained after the distortion [33].

In this thesis we focus on s.p.e.d. methods because we believe that it is possible to reduce the complexity of the resulting protocols.

## 1.5 Contributions

The aim of this thesis is to find a compromise between the accuracy of the biometrics recognition system and the complexity of the resulting privacy preserving protocol.

To design our solutions, we follow two different paths. In the first one, we use a particular cryptosystem belonging to family of *Somewhat Homomorphic Encryption schemes (SHE)* (Chapter 4). This particular set of cryptosystems

allows to operate on encrypted data obtaining the same result obtained working on the plaintexts (more details in Section 2.3). The resulting protocols are two privacy preserving biometrics identification and authentication schemes, based on iris and fingerprint in which all the computation is moved on the server side, without any interaction with the client, except for the input encryption and output decryption on the client side. Our main contribution consists in the application of the SHE scheme proposed by Pisa et al. in [34] to the biometrics recognition problem. It is worth noting that in this work we also deal with the problem of negative numbers, which was not addressed in [34] (Section 4.1). Concerning the proposed biometrics recognition protocol, we devised a solution allowing to compute the match score (Figure 3.1) between iris or fingerprint templates with the lowest possible amount of multiplications (the most complex operation) in the encrypted domain. Moreover, the distance is computed by means of parallelization in such a way to decrease the time complexity of the system (Section 4.3, Section 4.4). We also design a blinding method to obfuscate the outputs in the authentication framework and prevent  $\mathcal{C}$  to learn the exact distance from the template stored into the database (Section 4.5).

As we said, most of previous works on the topic are secure only against a passive attacker. In fact, despite the recent advances of SMPC in the malicious setting and the introduction of more efficient cryptographic primitives, the complexity of a protocol secure against an active attacker can be very high (Chapter 3). The second main contribution of the thesis (Chapter 5) exploits the strength of the SPDZ protocol, introduced by Damgård et al. in [35, 36], to provide a protocol secure in the malicious setting. Since we are looking for a trade off between complexity and accuracy, we propose SEMBA, a multimodal biometric system that combines face and iris templates. It is known that a multimodal biometric authentication protocol can reach better accuracy than algorithms based on single biometric. By using a simplified representation of the two biometric traits, the multimodal protocol can reach the same accuracy of the corresponding systems based on more accurate representation of iris or face templates, but keeping computational complexity low. It is up to system designers the decision to exploit the superior performance allowed by multimodal authentication to improve accuracy, instead,

without increase the complexity. In the resulting proposal, all the computation is carried out in a privacy preserving way and the output is revealed only if all the parties act honestly during the protocol. Our contribution in this case is mainly focused on the search for a good balance between efficiency and complexity. For this reason we analyze several possible configurations.

## 1.6 Publication list

*G. Droandi*, and R. Lazzeretti. "SHE based non interactive privacy preserving biometric authentication protocols." In *Intelligent Signal Processing (WISP), 2015 IEEE 9th International Symposium IEEE*. Siena, Italy, May 2015.

*G. Droandi*, "Non-interactive privacy preserving protocol for biometric recognition based on somewhat homomorphic encryption", in *ECCWS2015-Proceedings of the 14th European Conference on Cyber Warfare and Security 2015: ECCWS 2015*. Academic Conferences Limited, 2015, p. 355.

M. Barni, *G. Droandi*, and R. Lazzeretti, "Privacy protection in biometric-based recognition systems: A marriage between cryptography and signal processing", in *IEEE Signal Processing Magazine*, vol. 32, no. 5, pp. 66–76, 2015.

*G. Droandi*, M. Barni, R. Lazzeretti, T. Pigata. "*SEMBA: SEcure Multi-Biometric Authentication*", In ArXiv e-prints, 1803.10758, 2018.

## 1.7 Outline

The rest of this thesis is organized as follows: in Chapter 2 we introduce the general SMPC tools, we analyze thier security Section 2.1, and present a brief excursus of the most used privacy preserving techniques Section 2.2. Being directly related to our work, in Section 2.3 we describe more in detail the homomorphic encryption cryptosystems (especially the one used in Chapter 4) and in section Section 2.4 the SPDZ system of Chapter 5.

In Chapter 3, we report the state of art of secure biometric recognition

systems. We first recall the principal characteristic of biometrics in plain domain Section 3.1; then we present the state of art of privacy preserving biometric recognition, paying attention to the ones used in our research (iris Section 3.2, face Section 3.3, and fingerprint Section 3.4).

In Chapter 4 we present the privacy preserving fingerprint and iris biometric recognition systems for both authentication and identification published in [37] and [38]. In Section 4.1 we present our extension of the cryptosystem introduced in [34] to negative numbers, while Section 4.2 summarizes the general authentication and identification protocols. In sections Section 4.3, Section 4.4 and Section 4.5 we respectively describe our implementation of secure Hamming distance, Euclidean distance and the blinding method. Finally we analyze the protocol's complexity Section 4.6 and the experiments outputs Section 4.7.

In Chapter 5 we present our protocol for multimodal authentication based on SPDZ system, by using face and iris. First we analyze the stand alone iris Section 5.1 and face authentication protocols Section 5.2. Then in Section 5.3 we present SEMBA and tests results in plain Section 5.4 and encrypted Section 5.5 domain.

Finally, in Chapter 6 we sort out our conclusions.

## Chapter 2

---

# Cryptographic Tools

The basic problem in cryptography is providing secure communication over an insecure channel, whenever two parties desire to exchange a private message over a channel that can be compromised. This is usually done through encryption schemes, consisting of two main functions: *encryption* and *decryption*. Before sending a message, the sender encrypts the message obtaining the ciphertext. Once received the ciphertext, the other party decrypts the message using the decryption algorithm.

Since 1970s, following the introduction of problems like unforgeable *digital signature* and *fault-tolerant protocols*, cryptography has been redefined as everything concerned with the design of a system capable to withstand malicious attempts to abuse it, and not only the art of designing and analyzing *encryption schemes* [39].

In this thesis, we consider a *Secure Multiparty Computation* scenario, where two or more parties, for example a client and a server, are interested in jointly computing the same task without revealing each other their inputs. This scenario is also known as *secure distributed computing* [40]. The aim of secure multiparty computation (SMPC) is to enable all the parties to sort out a distributed computing task without any leak of informations.

In this chapter, we present the main characteristics of SMPC. First, in Section 2.1, we outline the general properties of SMPC and the corresponding security models. Second, in Section 2.2, we describe the most used cryptographic tools for SMPC: oblivious transfer (Section 2.2.1), garbled circuit (Section 2.2.2), secret sharing (Section 2.2.3). Then we describe a particular set of cryptographic systems known as homomorphic encryption (Section 2.3) and more in detail: somewhat homomorphic encryption (Section 2.3.1) and a new SMPC tool for secret sharing, known as SPDZ (Section 2.4) recently proposed by Damgård et al. Finally, for completeness, we briefly describe

hybrid protocols (Section 2.5).

Since in our implementations we use somewhat homomorphic encryption (SHE), in Chapter 4, and SPDZ, in Chapter 5, in this chapter we describe them in a more detailed way with respect to the other cryptographic tools.

## 2.1 Security Models

In secure multiparty computation,  $n$  participants  $P_1 \dots P_n$  want to compute jointly the same function without share their inputs. Security treats can be not only an external adversary, but also one or more of the  $n$  participants. A subset of *corrupted* players may try to gain as much information as possible about the inputs of the others, sometime also deviating from the protocol. For this reason, any SMPC protocol should respect the security properties described below [40].

- *Privacy*. Each party  $P_i$  should learn only the output.
- *Correctness*. The received output should be correct for each  $P_i$ .
- *Independence of inputs*. Each party must choose its inputs independently from the others, be them corrupted or honest.
- *Guaranteed output delivery*. Corrupted parties should not be able to prevent honest ones from receiving their output.
- *Fairness*. Corrupted parties should receive their output if and only if also honest parties receive theirs.

First of all we must define what security means regarding our scenario. In [40] Hazay and Lindell define SMPC's security as a mental experiment. Let us consider an "ideal world" where an external, trusted and incorruptible party  $\mathcal{T}$  exists.  $\mathcal{T}$ , called third party, helps the other players to carry out their computation. In this ideal world, each  $P_i$  can only send inputs and receive outputs from/to  $\mathcal{T}$ , by using perfect private channels. The third party computes the function, and gives the output back to  $P_1, \dots P_n$ . For this reason, the only action that an adversary can perform is choosing the corrupted parties' inputs [40]. In "real world", obviously, a trusted party  $\mathcal{T}$  does not exist. Hence a

real protocol is said to be *secure* if in the real execution, a corrupted party cannot do more harm than in the ideal world. The previous definition is too ideal and needs to be simplified to work in situations when it is impossible to guarantee both delivery and fairness. For this reason the definition is relaxed and the participants can also abort the computation [41–44].

We can define the adversaries models according to the actions they are allowed to take [40]:

1. *Semi-honest (passive) adversaries.* In this model, all the parties, even the corrupted ones, follow the protocol without deviating from it. However the adversary knows the status of the corrupted parties and use those information to obtain as much information as possible on the honest parties' inputs; attackers under this model are also referred as *passive* or *honest-but-curious* attackers or adversaries.
2. *Malicious (active) adversaries.* In this model, the corrupted parties may deviate from the protocol, following adversary's instructions. This kind of adversary tries actively to infiltrate the system. We will refer to this kind of adversary or attacker as *active*. To ensure security under malicious model, the computation is more complex than in the previous case.

The behavior of the adversary can also be classified according to the dynamic of the corruption. From this point of view we may distinguish the following cases [40]:

1. *Static corruption model.* A adversary has already corrupted a fixed number of parties. Honest parties remain honest and the adversary can not corrupt anyone else.
2. *Adaptive corruption model.* In this model the adversary has the ability of corrupting parties during the computation. Who and when can be decided by the adversary. Once a party is corrupted, it remains corrupted until the end of the protocol.
3. *Proactive model.* In this model, parties may be corrupted only for a limited period of time [45,46].



The majority of the secure biometric protocols proposed in literature (see Chapter 3, [11, 20] ) and those we propose in Chapter 4, are safe in a static corruption model and against a semi-honest adversary while the protocol in Chapter 5 is secure against  $n - 1$  malicious parties ( $n$  is the number of parties) also in a static model.

## 2.2 Basic SMPC Primitives

In our biometric protocols, two parties, called *server*  $\mathcal{S}$  and *client*  $\mathcal{C}$  are interested to evaluate a biometric recognition protocol without revealing information to each other. This is possible thanks to many SMPC tools proposed in the past. In the next sections, we focus on the most basic cryptographic primitives used in secure two party computation (STPC).

### 2.2.1 Oblivious transfer

*Oblivious transfer* (OT) [12, 47] is a protocol that allows the server  $\mathcal{S}$ , to send one out of  $n$  messages  $(x_1, x_2, \dots, x_n)$  to the client  $\mathcal{C}$ . On his side,  $\mathcal{C}$  chooses an index  $i$  to learn  $x_i$ . At the end of the protocol  $\mathcal{S}$  gets no information on the index  $i$  and  $\mathcal{C}$  knows nothing about the other messages  $x_j, i \neq j$ . OT is critical to implement some of the most efficient protocols for secure computation, such as *garbled circuits* [13] or *GMW* [48]. To the best of our knowledge, the best OT protocol [49] in both malicious and semi-honest case, achieves around 10000 1-out-2 OTs per second using only one thread. However, in many frameworks millions of OT protocols should be performed, making computation expensive in terms of complexity and time (for example for the privacy preserving evaluation of an AES circuit [50] or the protocols for *private set intersection* of [51] and [52]). The computational time cost can be reduced by using the OT extension method proposed by Beaver in [53, 54]. Basically, the protocol runs a small number of *base OT* that are then used to obtain many OTs, through only “cheap” symmetric cryptographic operations. OT extensions can be implemented with extraordinary efficiency in the case of passive adversaries. The protocol described in [55] after the base OT, requires only three hash functions for each OT, while the framework presented in [54] can carry out  $10^7$  OTs in less than three seconds, using four threads over a

LAN [54,56] with a bandwidth of  $km$ , that corresponds to about 256MB for  $k = 128$  bit;  $k$  is the security parameter and  $m$  the number 1-out-of-2 OT extension. In [54], the authors also observe that any other optimization that targets only runtime complexity has no effect. The OT extension protocol is so fast that this makes communication the bottleneck of the protocol. On the contrary, for malicious adversaries OT extensions are more expensive, in fact, with respect to [55], the run time of the most efficient framework is of 7.3s for  $2^{24}$  random OT extensions [57] on LAN, with communication complexity of 378MB, that is about 150% more than the corresponding passively secure protocol in [54]. To improve the efficiency, part of the computation can be moved offline and be precomputed [58].

### 2.2.2 Garbled circuits

In his revolutionary paper, Yao [13] introduced for the first time the possibility to evaluate securely any binary circuit. In Yao's *garbled circuit* protocol (GC), computation is distributed between two parties, the server  $\mathcal{S}$  and the client  $\mathcal{C}$ . In the simplest scenario,  $\mathcal{S}$  and  $\mathcal{C}$  are interested to jointly evaluate a function represented by a binary circuit. Let us consider a circuit having only one gate with two input wires  $w_1, w_2$  and one output wire  $w_3$ .  $\mathcal{S}$  knows only the input  $i_1$  to  $w_1$ , while  $\mathcal{C}$  only its input  $i_2$  to  $w_2$ . The garbled circuit allows to evaluate the circuit on the inputs so that  $\mathcal{S}$  learns nothing about  $i_1$ , while  $\mathcal{C}$  can determine nothing about  $i_2$ , except what he can deduce from the output and its own input. With  $G(i_1, i_2) \in \{0, 1\}$  we indicate the output of the gate. In practice,  $\mathcal{S}$  first encrypts (*garbles*) each input and output wire of the gate, generating two different keys  $k_j^0, k_j^1$  of  $t$ -bits each. A random permutation  $\pi$  is associated to each wire. Given  $c_j = \pi(i_j)$ ,  $k_j^{i_j} || c_j$  represents the garbled values of the wire and is completely independent from the input  $i_j$ . Then the gate is replaced by a table in which each entry contains the encryption of the corresponding garbled output of the wire; the encryption key is obtained as a combination of the input wires, and entries are sorted according the input permutation bits. Extending the procedure to a set of gates composing a binary circuit, the set of all the tables is called garbled circuit. After garbling,  $\mathcal{S}$  sends the GC and his secret inputs to  $\mathcal{C}$ , while  $\mathcal{C}$  receives the secrets associated to his inputs through OT. Finally,  $\mathcal{C}$  decrypts

the gates one by one by using the input secrets and the secrets obtained as output of previously evaluated gates, until the final output is obtained [13,59].

Considered to be impractical until few years ago, GC has recently gained more and more popularity, thanks to several efficiency improvements (summarized in [60]). The majority of the improvements are provided in the semi-honest model. However, many papers [61–66] have been focused on making the protocol secure against malicious attackers, through Zero Knowledge proof [67], cut and choose [65] and other computationally expensive techniques. These implementations can be used in biometric recognition, but the complexity cost is so high to make them un-practical in real world applications.

In a garbled circuit the complexity is distributed between the two parties. Even if great part of the computation is performed on  $\mathcal{S}$ 's side,  $\mathcal{C}$  must perform several computations. It is important to underline that circuit garbling does not depend on the actual inputs and in some particular scenarios where the functionality to be evaluated is known in advance, circuit encryption and transmission can be precomputed, so that only evaluation is carried out online by the client.

The computational complexity of GC operating in the semi-honest model depends linearly on the number of non-XOR gates composing the circuit, which in turn depends on the input bitlengths. In fact, thanks to the improvement proposed in [68], XOR gates can be evaluated with negligible computational and communicational complexity.

Given that, GCs are suited for operations such as sums and comparisons, for which the number of gates composing the circuit depends linearly on the input bitlength. On the contrary, GCs are less efficient for other operations, e.g. for the computation of products or divisions, since in this case the number of gates depends quadratically on the input bitlength.

As far as we know, the best GC protocol secure in the malicious model needs about  $\mathcal{O}(s \cdot C)$  symmetric encryptions, where  $C$  is the number of non-XOR gates and  $s$  the number of circuits necessary for the cut-and-choose protocol [65,66]. It is worth nothing that in semi-honest model  $\mathcal{O}(C)$  symmetric encryptions are required. The other malicious protocols are still impractical, for example, the one proposed in [62] is based on zero-knowledge proofs and it can be extended with a constant number of large modulus exponentiations,

but for short circuits the protocol is not yet practical and for big circuit it is not realistic at all. The LEGO protocol [61] is more efficient than [62], the number of exponentiations in this case is of  $\mathcal{O}(s \cdot |D|/\log |D|)$ , where  $s$  is a security parameter and  $D$  the size of the circuit.

### 2.2.3 Secret sharing

A secret-sharing scheme is a method allowing to distribute a secret among a group of  $n$  participants. Each party receives a *share* of the secret. Only when a certain number of participants put together their shares, the secret can be reconstructed. Therefore an individual's share alone does not reveal any information on the secret. Secret sharing schemes were introduced by Blakley [69] and Shamir [70] for secure information storage, but they have found many applications in cryptography and SMPC. In Blakley and Shamir schemes, the participants' subsets that can reconstruct the secret are all the sets whose cardinality is at least equal to a certain threshold  $t$ , systems with this property are usually referred to as  $(t, n)$ -*threshold scheme*. In [71] Ito et al. presented secret-sharing schemes for a more general access structure to the secret. Most of the known secret-sharing schemes are *linear*, i.e. the secret is an element of a finite field, and the shares are obtained by applying a linear function and several independent random elements to the secret. An interested reader can find in [72] a summary of secret sharing schemes.

In this thesis, we focus on a new secret-sharing scheme proposed by Damgård et al. in [35, 36] and usually referred to SPDZ. Since one of the privacy preserving biometric protocols we implemented uses SPDZ, we are going to describe SPDZ in more details in Section 2.4.

## 2.3 Homomorphic encryption

A cryptosystem is called *homomorphic* [73] if it enables the computation over encrypted data. In other words, given an operation  $\oplus$  on the plain texts, an operation  $\otimes$  on ciphertexts exists such that the application of  $\otimes$  to any pair of ciphertexts corresponds to the application of  $\oplus$  to the corresponding plain texts, that is:

$$\text{Dec}(\llbracket m_1 \oplus m_2 \rrbracket) = \text{Dec}(\llbracket m_1 \rrbracket \otimes \llbracket m_2 \rrbracket). \quad (2.1)$$

for any pairs of plain messages  $m_1$  and  $m_2$ . In Equation 2.1,  $\llbracket a \rrbracket$  indicates the encryption of the message  $a$  and  $Dec()$  the decryption operation.

Basically, a homomorphic encryption system (HE) allows to carry out some operations on the encrypted data. Once decrypted the output, the result of the operations should be the same as if all the corresponding operations were performed on the plain texts. The most common homomorphic cryptosystems, such as Paillier's cryptosystem [14], are additive and allow to evaluate the sum between encrypted values or the product between an encrypted value and a plain value. Such schemes are based on asymmetric cryptosystems, wherein a party, said  $\mathcal{C}$ , owns the public (pk) and private (sk) encryption keys while the other party, said  $\mathcal{S}$ , knows only the public key.

Given one or more inputs  $x_1 \dots x_n$  represented as integers,  $\mathcal{C}$  encrypts them with the public key obtaining the corresponding ciphertexts  $\llbracket x_1 \rrbracket, \dots \llbracket x_n \rrbracket$  and sends them to  $\mathcal{S}$  that carries out the computation by also using his part of the inputs. The final result is provided to  $\mathcal{C}$  that discovers it after decryption. Thanks to additive property of the cryptosystem, linear operations can be evaluated by  $\mathcal{S}$  without interacting with  $\mathcal{C}$ . On the contrary, non-linear operations, such as products between encrypted values or comparisons, are more complex and require interaction between the parties.

The computational complexity of HE-based protocols can be measured by counting the number of the most expensive operations or by measuring the runtime for the two parties involved in the protocol. Communication complexity is measured in terms of number of communication rounds and bandwidth. The bandwidth can be estimated by counting the number of ciphertexts that must be transmitted. Multiplicative homomorphic cryptosystems exist as well, allowing the evaluation of products between encrypted values, but they have a lower practical utility with respect to additive HE.

Fully homomorphic encryption schemes (FHE), allowing both the evaluation of additions and products between ciphertexts, are very useful, even if really expensive. FHE schemes are a cryptographic recent discovery. The first one has been presented by Gentry [74] in 2009. From this breakthrough many others have followed such those described in [15, 75–77]. We analyze FHE more in depth in the following section.

### 2.3.1 Somewhat and fully homomorphic encryption

An encryption scheme is defined fully homomorphic if it allows to perform both plain-text addition and multiplication manipulating only ciphertexts. As we said in the previous section, the first construction of a secure *somewhat homomorphic encryption* (SHE) and *fully homomorphic encryption* (FHE) schemes has been provided by Gentry [74]. SHE allows the evaluation of a limited number of operations, especially multiplications, while FHE extends SHE to overcome such a restriction but at the cost of incrementing even further the computational complexity.

A FHE scheme is the result of three steps. The first step consists in constructing a *somewhat homomorphic encryption* scheme. This scheme can perform only a limited number of operations before incurring into decryption errors due to the noise that grows after each operation. The noise is a random element that the encryption function adds to the message and which ensures that the ciphertext cannot be decrypted without knowing the secret key. To decrypt correctly, noise must be under a certain threshold, depending on the scheme and the parameters. The second step is the so called *squash*, that consists in modifying the decryption function in such a way that can be expressed as a low degree polynomial in the encrypted data and secret key bits. After each operation, noise grows and, when it is too large, the decryption becomes impossible. The innovative Gentry's idea was to introduce a third step, *bootstrapping*, which consists in the homomorphic evaluation of such a polynomial decryption on ciphertexts and additional encrypted key bits. In simple terms, thank to the squashed decryption function, a ciphertext can be decrypted and re-encrypted with a new secret key and no additional noise. So a new (refreshed) ciphertext of the same bitlength, but with reduced noise, is produced. Consequently the new ciphertext can undergo more operations before incurring again into errors. By repeating the bootstrapping stage before the noise becomes too large, the number of possible operations becomes virtually unlimited, thus allowing the construction of a fully homomorphic scheme.

Unfortunately, using FHE in real world applications is still impractical even if many improvements have been made. Starting from Gentry's scheme, five main families of FHE schemes have been proposed:

- schemes based on integers [76, 78–80];
- schemes based on ideal lattices [81–84];
- schemes based on Learning With Error (LWE) and Ring Learning With Error (RLWE) [75, 77, 85, 86];
- schemes based on the approximated eigenvector method [87, 88];
- the NTRU based schemes, that simplify the previous RLWE method because they can obtain ciphertexts of only one ring element [89, 90].

With regard to biometric recognition, the number of required operations is usually known in advance. Therefore the full strength of FHE is not necessary, and a SHE scheme is enough to guarantee security [27, 28, 91, 92]. This is the solution we adopted in [37, 38] and that we will describe in Chapter 4.

In the next section, we are describing the SHE schemes used in our applications to biometrics.

### 2.3.2 Somewhat homomorphic encryption on integers

We now present the somewhat homomorphic scheme over the integer, proposed by Pisa et al. [34] that we use in our protocols for iris and finger recognition, which will be presented in Chapter 4. The scheme we use is an extension of the DGHV cryptosystem proposed in [76]. The advantage of Pisa’s SHE scheme, is the possibility to encrypt integer values in ciphertexts having size in the order of the kilobits and compute any function expressed as a combination of sums and multiplications.

We first introduce some useful notation. Given two integer numbers  $x$  and  $p$ , we indicate with  $[x]_p$  or  $(x \bmod p)$  the reduction of  $x$  modulo  $p$ . The notation  $[x]_p$  indicates the remainder in the interval  $[0, p)$ , while  $x \bmod p$  refers to the integer in the interval  $(-p/2, p/2]$ .

Let  $\lambda$  be an integer referred to as the security parameter;  $\eta$  be the bit length of the secret key  $p$ ;  $\tau$  be the number of elements composing the public key, each of which has bit length  $\gamma$ ; finally, let  $\rho$  and  $\rho'$  be respectively the bit length of the noise in the public key and in a fresh ciphertext (Table 2.1).

For a given  $\eta$ -bit odd integer  $p$ , we use the following distribution over  $\gamma$ -bit integers:

$$\mathcal{D}_{\gamma,\rho} = \{\text{choose } q \in \mathbb{Z} \cap [0, 2^\gamma/p), r \in \mathbb{Z} \cap (-2^\rho, 2^\rho) : \text{output } x = q \cdot p + r\}$$

To explain how the scheme proposed in [34] works, we first present a symmetric scheme and then we modify it to obtain an asymmetric scheme. In DGHV scheme the message is encrypted by hiding it with additional noise of a multiple of an odd integer  $p$ . A message  $m \in \{0, 1\}$  can be hidden in a integer  $c$  such that its residue modulo  $p$  has exactly the same parity of  $m$ . In other words, the ciphertext  $c$  can be represented as  $c = m + p \cdot q + 2 \cdot r$  for some  $q, r \in \mathbb{Z}$ . Namely a ciphertext is a *near multiple* of  $p$  and the message is hidden into the noise  $m + 2r$ . If  $r$  is not too large and  $m + 2r$  does not exceed  $p/2$ , the message can be recovered with two modulus operations.

Starting from this symmetric scheme, Pisa et al. adapted the encryption function as  $c = m + p \cdot q + b \cdot r$ , with  $m \in [0, b)$ , where the integer  $p$  must be equivalent to 1 mod  $b$  and  $q$  must be odd, so that  $[[c]_p]_b = m$ . The above symmetric scheme can be modified into an asymmetric one where the secret key is still the integer  $p$ , while the public key is a set of  $\tau$  near multiples of  $p$ , namely for  $i = 0, \dots, \tau$ ,  $x_i = p \cdot q_i + r_i$  where  $r_i, q_i \in \mathbb{Z}$  are randomly chosen.

We now describe in details the scheme presented in [34] by Pisa et al.:

**KeyGen**( $\lambda$ ). The secret key is an integer  $p \in [b^{\eta-1}, b^\eta) \cap \mathbb{Z}$  such that  $p$  is not divisible by  $b$ . The public key is a set of  $\tau$  elements obtained as  $x_i = q_i \cdot p + r_i$  for all  $0 \leq i \leq \tau$ , with  $r_i \in (-b^\rho, b^\rho)$  randomly chosen noise and  $q_i \in [0, b^\gamma/p)$ . The element  $x_0$  should be greater than every other  $x_i$ , it should be odd and the noise should be even. The public key is  $\mathbf{pk} = \{x_0, x_1, \dots, x_\tau\}$ .

**Encrypt**( $\mathbf{pk}, m$ ). Given an integer message  $m \in [0, b)$ , the encryption function chooses a random integer  $r \in (-b^{\rho'}, b^{\rho'})$  and a sparse subset  $S$  of indexes in  $\{0, 1, \dots, \tau\}$ . The ciphertext  $c$  is

$$c = \left[ m + b \cdot r + b \cdot \sum_{i \in S} x_i \right]_{x_0}. \quad (2.2)$$



**Decrypt**( $p, c$ ). Given a ciphertext  $c$ , the decryption function is  $m = [c]_p \bmod b$ .

**evaluate**( $\mathbf{pk}, C, c_1, \dots, c_t$ ). To perform addition and multiplication between two given ciphertexts  $c_1, c_2$ , just sum or multiply the ciphertext modulus  $x_0$ .

It is worth noting that this scheme can also encode negative numbers, it is our small contribution to the cryptosystem (Section 4.1). Clearly, if the base  $b$  consists of  $k$ -bit, we can encrypt integers in the interval  $(-b/2, b/2]$ . In this case the decryption function is performed as  $([c]_p \bmod b)$  and returns a positive number if  $[[c]_p]_b < b/2$ , negative otherwise  $([[c]_p]_b - b)$ . In the case of negative numbers, the *base* should be twice the maximum integer that needs to be computed.

Table 2.1: Parameters recap

Symbol	Definition	
$b$	base	
$\lambda$	security	
$\eta$	bit length of sk	$\eta = \lambda^2$
$\rho$	bit length of the noise in pk	$\rho = \lambda$
$\rho'$	bit length of the noise in ciphertext	$\rho' = 2 \cdot \lambda$
$\gamma$	ciphertext's length	$\gamma = \lambda^5$
$\tau$	number of pk's elements	$\tau = \gamma + \lambda + \log_2(b)$

**Parameters** In order to preserve the semantic security of the reduction to approximate-GCD (great common divisor) problem as in [76], the parameters depend on  $\lambda$  polynomially and, according to [34], they are defined as:  $\rho = \lambda$ ,  $\rho' = 2\lambda$  respectively for the noise of each element of public key and ciphertext;  $\eta = \lambda^2$  for the length of the private key;  $\gamma = \lambda^5$  for the length of the ciphertext;  $\tau = \gamma + \lambda + \log_2(b)$  is the number of elements of the public key.

Lemma 1 proves the correctness of the decryption function under the chosen parameters set.

**Lemma 1.** *Chosen a base  $b = 2^k$ , let  $(sk, \mathbf{pk})$ , keys output from  $KeyGen(\lambda)$  as in the system shown in Section 2.3.2. Let  $c = \mathbf{encrypt}(\mathbf{pk}, m)$  for  $m \in$*

$[0, b - 1]$ , with  $b \in \mathbb{Z}$ . Then,  $c = m + a \cdot b + p \cdot q$  for some integers  $a, q$  and  $|a \cdot b + m| < p/2$ . The decryption function is correct.

The scheme is somewhat homomorphic since only a limited number of operations can be performed. This number depends on the magnitude of the noise after every operation. While after a multiplication we have a significant noise increase, the addition is less problematic because it produces only a slight noise increase. In order to decrypt the correct message, the total noise should not grow more than  $p/2$ , respecting the boundary<sup>1</sup>.

$$b^{2\lambda+1} + \lambda^5 \cdot b^\lambda \cdot (b^2 + b) < \frac{p}{2}. \quad (2.3)$$

Finding the maximum number of possible multiplications means finding the largest  $\mu$  that satisfies  $R^\mu < \frac{p}{2}$ , where  $R = b^{2\lambda+1} + \lambda^5 \cdot b^\lambda \cdot (b^2 + b)$  is the maximum allowed noise as in Equation 2.3, therefore, the bound is  $(b^{2\lambda+1} + \lambda^5 \cdot b^\lambda \cdot (b^2 + b))^{\mu+1} < \frac{p}{2}$ . As shown in [34], the number of multiplications seems to depend mainly on the security parameter  $\lambda$  while they are quite independent from the base. On the contrary the limit of additions is so high that, for certain set of parameters, is negligible.

**Security Basis of the SHE Scheme** Among the attacks against an asymmetric encryption scheme, the most important are those who recover the private key from the public key and those who uncover information from ciphertext. In this section, we briefly recall the security basis of the extended scheme presented in [34]. The security of the scheme is based on the problem known as approximate-GCD and it has been analyzed for two numbers in [93].

Indeed the public key consists of a set of elements  $x = p \cdot q + r$ . Therefore, to recover the private key it is necessary to calculate the GCD between these approximated terms. Finding the private key requires an exhaustive search in the noise space; hence the research can be made more difficult by increasing noise.

Basically, the procedure requires to find all the possible common divisors for each pair of public key elements, taking into account all possible values with the aim of defining a set. Repeating the procedure for all possible pairs of

---

<sup>1</sup>We refer to [34] for further details.

public key elements, the possible private keys can be found in the intersection of all the sets and therefore must be repeated until it is a singleton. In the DGHV scheme in [76], Van Dijk et al. adopt a maximum noise equal to  $\rho = 2\lambda$ , which requires an exhaustive search of  $2^{4\lambda}$  for each pair of public keys. In the extended scheme [34], even if the base varies, the problem and the resolution method remain essentially the same, which implies that the complexity of the scheme does not decrease. Moreover Pisa et al. claim that since the noise increases of  $\log_2 b$  (where  $b$  is the adopted base), the complexity actually increases.

A cryptographic scheme is said to be *semantically secure* if two plaintexts are completely indistinguishable after encryption. Pisa et al. claim that their system is semantically secure by relying on the approximate-GCD problem, and they provide only a sketch of the security proof. The authors consider a game with a challenger and an attacker. First of all the attacker receives the public key, then he chooses two messages of the same sizes and sends them to the challenger. After that, the challenger chooses one of the messages, encrypts it and sends it back to the attacker. The attacker wins the game if he can reveal the origin of the message. Van Dijk et al. prove in [76], that an attacker A with advantage  $\varepsilon$  can be converted into an algorithm where by B can solve the GCD problem with success probability  $\varepsilon/2$ . In [34], the procedure is essentially the same with minor differences. First the size of the parameters  $\rho, \gamma, \eta$  is multiplied by a factor  $\log_2 b$  to take into account the larger base. Second  $\tau$  must also be increased of the same factor in order to preserve the statistical indistinguishability of cipher distribution, since the message space is larger. In the extension, thanks to the larger noise, the approximate-GCD problem is more complex and therefore the system is secure as the DGHV scheme. Concluding their analysis, Pisa et al. claim that the security level is the same as the DGHV scheme because the basic principles underlying both schemes are the same.

## 2.4 SPDZ

In this section we describe the secret sharing protocol known as SPDZ and proposed by Damgård et al. in [35, 36]. SPDZ system provides security

against an active adversary corrupting up to  $n - 1$  of the  $n$  players.

We assume the computation is performed over a fixed finite field  $\mathbb{F}_p$  of characteristic  $p$ ; where  $p$  is a prime number. Each player  $P_i$  has an uniform share  $\alpha_i \in \mathbb{F}_p$  of a secret key  $\alpha$  such that  $\alpha = \sum_{i=1}^n \alpha_i \bmod p$  (in the following we omit the indication of the modulus operation for simplicity).

An item  $a \in \mathbb{F}_p$  is  $\langle \cdot \rangle$ -shared if player  $P_i$  holds a tuple  $\langle a_i, \gamma(a)_i \rangle$  such that  $a = \sum_{i=1}^n a_i$  and  $\gamma(a) = \sum_{i=1}^n \gamma(a)_i$ . In other words,  $a_i$  and  $\gamma(a)_i$  are additive secret shares of  $a$  and  $\gamma(a)$ . The value  $\gamma(a)$  represents the Message Authentication Code (MAC) of  $a$ . Any operation involving some variables is also performed on their MAC, so that, at the end of the protocol, the MAC is checked before revealing the outcome. If it is not consistent with the final output, the procedure aborts and nobody gets the output.

During the description of the protocol, we say that a  $\langle \cdot \rangle$  - shared value is *partially opened* if each party reveals to the other the value  $a_i$  but not the associated  $\gamma(a)_i$ . In this thesis we focus on secure two-party computation protocols, then  $n = 2$  and  $\alpha = \alpha_1 + \alpha_2$  and  $\langle a, \gamma(a) \rangle = \langle a_1, \gamma(a)_1 \rangle + \langle a_2, \gamma(a)_2 \rangle$ .

The protocol can be divided into two phases. The first one, referred to as *pre-computation*, where the system is set and the second, online, where the actual computation is performed. The pre-processing phase in realty needs some interaction between parties, but it's sometimes called offline only as opposed to the following online phase. In the original paper,[36] the pre-processing part is implemented by relying on SHE, with a shared private key. During the offline phase, parties generate a public key and a shared secret key for the SHE scheme (see [36] for details). They use an homomorphic scheme of the LWE family (learning with error Section 2.3), mostly based on [94] and [95,96]. Then, relying on the homomorphic properties of the SHE, the pre-processing protocol generates  $\alpha$  and  $\alpha$ 's shares, input shares, shares of tuples for multiplications and squares, and the random share values necessary to evaluate the comparison [36]. Multiplication and square tuples are sets of two or three elements needed for interactive operations, as detailed in the following:

- A multiplication tuple is the set  $\{\langle a \rangle, \langle b \rangle, \langle c \rangle\}$  for some  $a, b, c \in \mathbb{F}_p$  such that  $c = a \cdot b$  randomly chosen.
- A squared tuple is  $\{\langle a \rangle, \langle b \rangle\}$  for some  $a, b \in \mathbb{F}_p$  such that  $b = a^2$  ran-

domly chosen.

Each party can see only his set of shares, generated using the homomorphic properties of the cryptosystem. Details can be found in [36].

All the tuples are produced twice more than needed, because half of them are “sacrificed” in the sub-phase called tuple-checking. During this last sub-phase the protocol checks if the output of the previous sub-phase has been produced correctly along with the corresponding MACs. In fact, an adversary could introduce errors during MAC and decryption key generation. MAC checking is possible without revealing the MAC key  $\alpha$ . If  $a$  is a partially opened value all the parties have  $a = \sum_{i=1}^n a_i$  but not the associated  $\gamma(a)$ .  $\gamma(a)$  is additively shared between parties. To check if  $a$  is correct, we must verify that  $\gamma(a) = \alpha \cdot a$ . Since  $\gamma(a) - \alpha \cdot a$  is a linear function of  $a, \alpha, \gamma$ , the parties can compute the function locally and then check if  $\gamma(a) - \alpha \cdot a = 0$  without revealing  $\alpha$ . We will refer to this procedure as *MACCheck* (Protocol 2.1). Finally each party decrypts his set of pre-processed data by using his secret key share. In the implementation proposed in Chapter 5, we assume that the generation of tuples and inputs have been already done in the encrypted domain before the protocol starts and we focus our efforts on the analysis of the online part of the system.

Table 2.2: Linear operation in SPDZ.

<b>Operation</b>	<b>Server</b>	<b>Client</b>
$\langle a \rangle + \langle b \rangle$	$\langle a \rangle_1 + \langle b \rangle_1$	$\langle a \rangle_2 + \langle b \rangle_2$
$\langle a \rangle - \langle b \rangle$	$\langle a \rangle_1 - \langle b \rangle_1$	$\langle a \rangle_2 - \langle b \rangle_2$
$\alpha \cdot \langle a \rangle$	$\alpha \cdot \langle a \rangle_1$	$\alpha \cdot \langle a \rangle_2$
$c + \langle a \rangle$	$c + \langle a \rangle_1$	$\langle a \rangle_2$

Linear operations, such as additions and scalar multiplications (see Table 2.2), can be performed on the  $\langle \cdot \rangle$ -shared without interaction; while multiplications and comparisons need data transmission and proper sub-protocols. A summary of operations complexities is presented in Table 2.3.

**Protocol 2.1** MACCheck protocol

Sketch of the MACCheck protocol of the values  $a_1 \dots a_k$  for  $n$  parties  $\mathcal{P}_1 \dots \mathcal{P}_n$ . All players have a public set of opened values  $\{a_1, a_2 \dots a_k\}$ , meaning that in a previous sub-protocol each  $\mathcal{P}_i$  sent its share of  $a_t$  ( $t = 1 \dots k$ ) to all the other  $\mathcal{P}_i$ ,  $j = 1 \dots n$ ,  $j \neq i$ . For  $t = 1 \dots k$ ,  $(\gamma(a)_t)_i$  indicates the share of  $\gamma(a_j)$  of the party  $i$ . More details can be found in [36].

**Input:**  $\{a_1, a_2 \dots a_k\}$ , partially open values, each player has input  $\alpha_i$  and  $\gamma(a)_t)_i$ , for all  $t = 1 \dots k$ .

- 1: Each player set a random seed  $s_i$  and sends it to the other.
- 2: Each player compute  $s = s_1 \oplus s_2 \oplus \dots \oplus s_n$ .
- 3: Each player sample a random vector  $\mathbf{r}$  from an uniform pseudorandom generator  $\mathcal{U}$  initialized with the seed  $s$ . (Every one obtain the same vector because they agreed on the seed).
- 4: Each player computes  $a = \sum_{j=1}^n r_j \cdot a_j$ .
- 5: Each  $\mathcal{P}_i$  computes  $\gamma_i = \sum_{j=1}^k r_j \cdot \gamma(a_j)_i$  and  $\delta_i = \gamma_i - \alpha_i \cdot a$ .
- 6: Each  $\mathcal{P}_j$  sends  $\delta_i$  to all other parties.
- 7: Compute  $\delta = \delta_1 + \dots + \delta_n$ .

**Output:** : If  $\delta \neq 0$  abort.

**Multiplication and square** We show how to securely evaluate the product between two ciphertexts and the square of a ciphertext [36]. During the offline phase, several *multiplication triples* are produced. In the online phase to multiply two  $\langle \cdot \rangle$ -shared  $\langle x \rangle$  and  $\langle y \rangle$ , we take a multiplication triple  $\{\langle a \rangle, \langle b \rangle, \langle c \rangle\}$  and we partially open  $\langle x \rangle - \langle a \rangle$ , obtaining  $\varepsilon$ , and  $\langle y \rangle - \langle b \rangle$ , obtaining  $\delta$ . Now the share of  $z = x \cdot y$  is computed as  $\langle z \rangle = \langle c \rangle + \varepsilon \cdot \langle b \rangle + \delta \cdot \langle a \rangle + \varepsilon \cdot \delta$ . The multiplication protocol (Protocol 2.2) requires two transmissions to partially open  $\varepsilon$  and  $\delta$ .

In order to compute more efficiently the square of a sharing value  $x$  using only one transmission, as in [36], during the offline phase we prepare a list of pairs of  $\langle \cdot \rangle$ -shared values  $\{\langle d \rangle, \langle e \rangle\}$  such that  $e = d^2$  ( $d, e \in \mathbb{F}_p$ ). To square the shared value  $\langle x \rangle$ , we take a pair  $\{\langle d \rangle, \langle e \rangle\}$  and partially open the result of  $\langle x \rangle - \langle d \rangle$  obtaining  $\varepsilon$ . Then the share of  $z = x^2$  can be computed from  $\langle z \rangle = \langle e \rangle + 2 \cdot \varepsilon \cdot \langle d \rangle - \varepsilon^2$ . The complete protocol is described in Protocol 2.3.

Multiplications and squares, between shares, require respectively two and one transmissions. Therefore those are the most complex operations and we

choose to evaluate all the following complexities in terms of multiplication (or square) number and transmissions.

---

**Protocol 2.2** Multiplication.

Multiplication protocol for  $n$  parties  $\mathcal{P}_1 \dots \mathcal{P}_n$ .  $x_i$  indicates the share of  $x$  of the party  $i$ .

**Input:**  $\langle x \rangle, \langle y \rangle$

- 1: Take a multiplication triple  $\{\langle a \rangle, \langle b \rangle, \langle c \rangle\}$ .
- 2: Compute  $\langle x \rangle - \langle a \rangle = \langle \varepsilon \rangle$ .
- 3: Compute  $\langle y \rangle - \langle b \rangle = \langle \rho \rangle$
- 4:  $\forall i = 1 \dots n$ ,  $\mathcal{P}_i$  sends  $\varepsilon_i$  and  $\rho_i$  to all  $\mathcal{P}_j$ , with  $j \neq i$ .
- 5: Compute  $\varepsilon = \sum_{i=1}^n \varepsilon_i$  and  $\delta = \sum_{i=1}^n \rho_i$
- 6: Compute  $\langle z \rangle = \langle c \rangle + \varepsilon \cdot \langle b \rangle + \delta \cdot \langle a \rangle + \varepsilon \cdot \delta$ .

**Output:**  $\langle z \rangle$ .

---



---

**Protocol 2.3** Square.

Protocol for square for  $n$  parties  $\mathcal{P}_1 \dots \mathcal{P}_n$ .  $x_i$  indicates the share of  $x$  of the party  $i$ .

**Input:**  $\langle x \rangle$ .

- 1: Take a square pair  $(\langle a \rangle, \langle b \rangle)$ .
- 2: Compute  $\langle x \rangle - \langle a \rangle = \langle \varepsilon \rangle$ .
- 3:  $\forall i = 1 \dots n$ ,  $\mathcal{P}_i$  sends  $\varepsilon_i$  to all  $\mathcal{P}_j$ , with  $j \neq i$ .
- 4: Compute  $\varepsilon = \sum_{i=1}^n \varepsilon_i$
- 5: Compute  $\langle z \rangle = \langle b \rangle + 2 \cdot \varepsilon \cdot \langle a \rangle - \varepsilon^2$ .

**Output:**  $\langle z \rangle$ .

---

**Comparison** In this paragraph, we show how to compute the outcome of a secure comparison  $x < y$ , for any two elements  $x, y \in \mathbb{F}_p$ , according to the protocol proposed in [97]. The comparison computation is based on the observation that  $\langle x < y \rangle$  is determined by the truth values of  $\langle x < \frac{p}{2} \rangle$ ,  $\langle y < \frac{p}{2} \rangle$ , and  $\langle (x - y) \bmod p < \frac{p}{2} \rangle$ , where  $\langle x < y \rangle$  indicates the share values of the outcome of  $x < y$ . We can avoid to compute also  $\langle x < \frac{p}{2} \rangle$  and  $\langle y < \frac{p}{2} \rangle$  since, as in [97], we choose  $p$  large enough that both inputs are lower than  $\frac{p}{2}$ . If  $z$  is equal to  $x - y$ , given  $\langle z < \frac{p}{2} \rangle$ , then  $\langle x < y \rangle$  can easily computed as  $1 - \langle z < \frac{p}{2} \rangle$ .

We observe that if  $z > \frac{p}{2}$  then  $2z > p$ . Since we work on  $\mathbb{F}_p$ , we have that  $2z \bmod p = 2z - p$  and it is odd; else if  $z < \frac{p}{2}$  then  $2z < p$  and it will be even because we do not need any modular operation. Therefore, to establish if  $z$  is larger or smaller than  $\frac{p}{2}$  we need to determine only the last significant bit of  $2z$ . To compute the last significant bit of  $2z$ , we use a value  $\langle r \rangle$  shared by parties both as integer and as a bit array. The value  $r$  along with its bit decomposition are pre-computed offline. We indicate by  $r_0 r_1 \dots r_\ell$  the bits of  $r$  and with  $\langle r_i \rangle$  their shared values.

First of all we compute  $\langle s \rangle = \langle 2z + r \rangle$ , than  $s$  is partially opened. Now if  $s < p$  then the last significant bit of  $2z$  is equal to  $s_0 \oplus r_0$  else it is equal to  $1 - (s_0 \oplus r_0)$ .

Since we work in the field  $\mathbb{F}_p$ ,  $s < p$  if and only if  $s < r$ . For this reason we must only determine whether  $s < r$  or not. We remind that  $s$  is known to both parties therefore we can easily obtain a  $\langle \cdot \rangle$ -share of  $\delta$ , the truth value of  $\langle s < r \rangle$  (i.e.  $\langle \delta \rangle = \langle s < r \rangle$ ) working on the bits of  $s$  and on the shared bits of  $r$ . In order to do that we use the following procedure to calculate  $\langle \delta \rangle = \langle s < r \rangle$ :

1. If  $s_0 = 0$  then  $\langle \delta \rangle = r_0$  else  $\langle \delta \rangle = 0$ .
2. For all  $i < \ell - 1$   
     if  $s_i = 0$  then  $\langle \delta \rangle = \langle r_i \rangle + \langle \delta \rangle \cdot \langle 1 - r_i \rangle$   
     else  $\langle \delta \rangle = \langle r_i \rangle \cdot \langle \delta \rangle$ .

Now  $\langle z < \frac{p}{2} \rangle$  can be easily calculated as

$$\langle \delta \oplus s_0 \oplus r_0 \rangle = \langle \delta \rangle - \langle s_0 \oplus r_0 \rangle - \langle \delta \rangle \cdot \langle s_0 \oplus r_0 \rangle \quad (2.4)$$

Since  $s_0$  is known, in our implementation, Equation 2.4 can be simplified: if  $s_0 = 0$

$$\langle \delta \rangle + \langle r_0 \rangle - 2 \cdot \langle \delta \rangle \cdot \langle r_0 \rangle$$

else

$$1 + 2\langle \delta \rangle \cdot \langle r_0 \rangle - \langle r_0 \rangle - \langle \delta \rangle.$$

For more details see [97, 98].

The entire protocol is summarized in Protocol 2.4. This protocol requires one multiplication for each iteration plus one for the last step. Therefore the



complexity depends on the bit length of  $r$  and is equal to  $\ell$  multiplications and  $2\ell$  transmissions.

---

**Protocol 2.4** Comparison
 

---

Protocol for comparison between two share  $\langle x \rangle, \langle y \rangle$  for  $n$  parties  $\mathcal{P}_1 \dots \mathcal{P}_n$ .  
 $\langle x \rangle_i$  indicates the share of  $x$  of the party  $i$ .

**Input:**  $\langle x \rangle, \langle y \rangle$ .

- 1:  $\langle z \rangle = \langle x \rangle - \langle y \rangle$ .
  - 2:  $\langle \varepsilon \rangle = 2 \cdot \langle z \rangle$ .
  - 3: Take a random shared value  $\langle r \rangle$  and its shared bits  $\langle r_\ell \rangle \langle r_{\ell-1} \rangle \dots \langle r_0 \rangle$ .
  - 4: Compute  $\langle s \rangle = \langle \varepsilon \rangle + \langle r \rangle$ .
  - 5:  $\forall i = 1 \dots n$ ,  $\mathcal{P}_i$  sends his  $\langle s \rangle_i$  to all  $\mathcal{P}_j$ , with  $j \neq i$ .
  - 6: compute  $\varepsilon = \sum_{i=1}^n \langle s \rangle_i$
  - 7: **if**  $s_0 = 0$  **then**
  - 8:    $\langle \delta \rangle = r_0$ .
  - 9: **else**
  - 10:    $\langle \delta \rangle = 0$ .
  - 11: **end if**
  - 12: **for all**  $i < \ell - 1$  **do**
  - 13:   **if**  $s_i = 0$  **then**
  - 14:     **return**  $\langle \delta \rangle = \langle r_i \rangle + \langle \delta \rangle \cdot \langle 1 - r_i \rangle$
  - 15:   **else**
  - 16:      $\langle \delta \rangle = \langle r_i \rangle \cdot \langle \delta \rangle$ .
  - 17:   **end if**
  - 18: **end for**
  - 19: **if**  $s_0 = 0$  **then**
  - 20:    $\langle z < \frac{p}{2} \rangle = \langle \delta \rangle + \langle r_0 \rangle - 2 \cdot \langle \delta \rangle \cdot \langle r_0 \rangle$
  - 21: **else**
  - 22:    $\langle z < \frac{p}{2} \rangle = 1 + 2\langle \delta \rangle \cdot \langle r_0 \rangle - \langle r_0 \rangle \langle -\delta \rangle$ .
  - 23: **end if**
- Output:**  $\langle z < \frac{p}{2} \rangle$ .
- 

**Security** In [36] it is demonstrated that the whole system is secure against at most  $n - 1$  corrupted parties. The tuple checking phase prevents the adversary to introduce errors because MAC is checked without revealing  $\alpha$ .

Table 2.3: Computational complexity for SPDZ operations.  $\langle \cdot \rangle$  indicates shares,  $a, b, c \in \mathbb{F}_p$ ,  $\ell$  is the bit length of  $p$ .

Operation	Round	Bandwidth
$\langle a \rangle + b$	0	0
$\langle a \rangle - b$	0	0
$b - \langle a \rangle$	0	0
$b \cdot \langle a \rangle$	0	0
$\langle a \rangle \cdot \langle c \rangle$	2	$2\ell$
$\langle a \rangle^2$	1	$\ell$
$\langle a \rangle < \langle c \rangle$	$2 \cdot \ell$	$2 \cdot \ell^2$

Moreover they construct a simulator of the offline and online phases and demonstrate that the adversary is able to cheat the MACCheck procedure with probability  $2/p$ . Thus the cheating probability is negligible since  $p$  can be chosen ad hoc.

## 2.5 Hybrid protocols

At the end of our excursus we mention hybrid protocols. Complex protocols can be divided into subprotocols and different tools can be used for their implementation, in order to take the best from each approach. Usually hybrid protocols work with HE and GC (as in [99]), but can be adapted to different tools. The main challenge of hybrid protocols is the adoption of proper interface protocols to link subparts implemented relying on different techniques. For example, it may happen that a HE protocol outputs an intermediate  $x$  and the same  $x$  must be used as an input in a GC subroutine, or vice versa. For this reason the two parts of the protocol must be connected in such a way that the security of the whole system is guaranteed. At the same time, the representation of the variable  $x$  must be adapted to the new subprotocol requirements. Such an interface can be obtained through additive blinding: a random value  $R$  is added to  $x$  by the server at the end of a subprotocol and then removed at the beginning of the next one, so that only the masked value of  $x$  is disclosed to the client.



## Chapter 3

---

# Privacy Preserving Biometrics Matching

Biometric based recognition refers to the automatic recognition of individuals based on their unique anatomical traits (face, iris, fingerprints, etc.) or their characteristic behavior (such as signature, gait) [100]. In the last years it has become a useful tool for people identification since identity verification is an integral part of our society, especially in the digital cyber world, thanks to the universality, distinctiveness, ease collectability, permanence and performances of biometric traits. While passwords can be stolen or lost, biometrics traits are always with us. For this reason, creating a reliable automatic recognition systems is crucial for many business sectors. On the other hand, since biometric traits cannot be changed, it is also becoming crucial to find a way to protect them from forgery and spoofing.

Reliable and secure biometric recognition protocols can be developed using Secure Multi-Party Computation techniques, as shown in [11]. In this chapter, we present the most common biometric recognition methods, i.e. iris, fingerprints, and face, in (Section 3.1) and their implementation in the encrypted domain (Section 3.2, Section 3.3, Section 3.4). These three biometrics are strictly related to our research, as it will be evident from Chapter 4 and Chapter 5. Then in section Section 3.5 we describe the principal characteristic of a multi-biometric system. Finally in Section 3.6 we review the state of art of multi-biometric works, secure in the malicious setting.

### 3.1 Biometric recognition in plain domain

A biological trait must have some requirements to be qualified as a biometrics. Some of the main characteristics are:

**universality** each person should have the characteristic;

**distinctiveness** the same characteristic in two people should be enough different to avoid false matches;

**permanence** the characteristic should not change over a period of time;

**collectability** that characteristic can be measured quantitatively;

**acceptability** this feature refers to extent people are willing to accept the use of a particular biometric characteristic in their daily life.

**strength** how difficult is to gain a positive response using fraudulent methods.

**reducibility** the captured trait should be reduced to a form easy to handle.

However to be practical a biometric system should be at the same time accurate, fast, accepted by the intended population, and enough robust to resist to various attacks to the recognition system itself [5].

Many biometric characteristic are used in several applications: commercial (computer network login, electronic data security, e-commerce, etc.), governmental (national ID card, driver's license etc.), and forensic (terrorist identification, criminal investigation, etc.). DNA, iris, face, fingerprint, ear shape, voice, gait, and palmprint are only some examples of biometric trait that can be used in recognition systems. Each one has its strong and weak points and none is optimal for all the possible applications [5]. The match between a specific biometric trait and an application is determined depending on the operational modalities and the properties of the biometric trait.

Any biometric recognition system consists of four main stages (Figure 3.1).

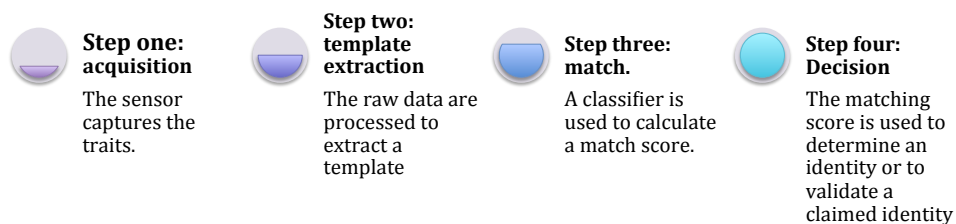


Figure 3.1: Biometric recognition system main steps.

First some kind of sensor captures the traits of an individual as a raw biometric data. Second, raw data are processed to extract a compact repre-

sensation of the physical trait, usually called *features*. Features of the same person can be grouped in a set of  $n$  elements and are usually called *template*. The third stage is the matching step during which the system employs a classifier  $D(\cdot)$  to compare the extracted features with the templates stored in a database. In the final step, the decision step, the matching score is used to determine an identity or to validate a claimed identity.

In a generic recognition system, a server  $\mathcal{S}$  owns a database of  $n$  enrolled biometric feature vectors  $(\{y_i\}, i = 1 \dots n)$ . The client  $\mathcal{C}$  owns a biometric vector  $x$ . Depending on the application context, the biometric system can operate in two ways: *identification* (Figure 3.3 ) or *verification* (Figure 3.2). The first mode, *identification*, answers the question *who does this biometric*

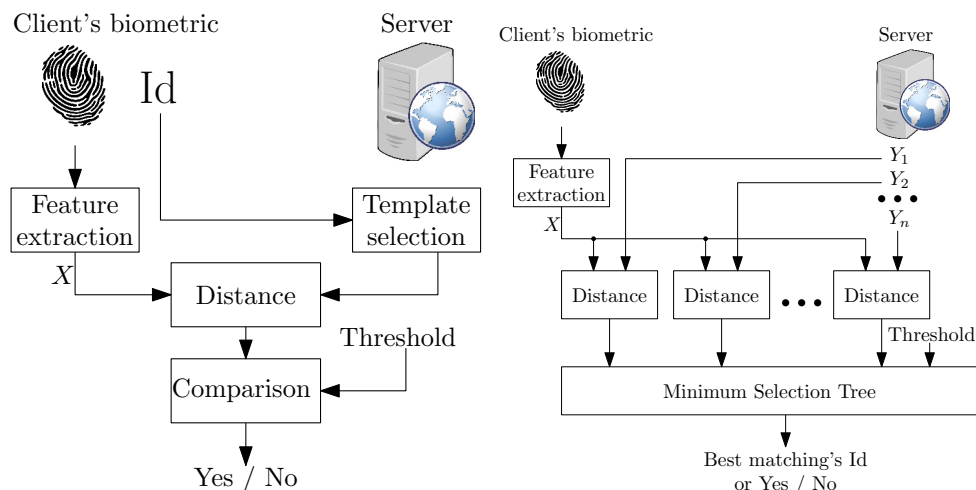


Figure 3.2: Authentication protocol

Figure 3.3: Identification protocol

*belong to?* and aims to establish if the identity is known by the system, without that the subject claims who he/she is. The identification system matches the new template with all those into the database, searching for the most similar one. Given a threshold  $t$ , the server must verify if an  $i$  exists such that  $D(x, y_i) < t$ . In the second mode, also called *authentication*, the system validates the identity of a user to answer the following question: *is he who claims to be?* The person claims to be someone and the system needs to confirm or deny the claim, comparing the new template with the one stored

into the database [5]. Given the identification tag  $\text{id}$ , the server checks if the distance between the query  $\mathbf{x}$  and the probe in the database  $\mathbf{y}_{\text{id}}$  is lower than the threshold, i.e.  $D(\mathbf{x}, \mathbf{y}_{\text{id}}) < t$ . We will use the generic term *recognition* when there is no need to make a distinction between the two modalities.

Biometric recognition poses some risks (Section 1.2). Since a physical traits, like face, voice or fingerprint, are impossible to hide from others, an impostor may attempt to use a copy or an imitation of a legitimate user's biometric trait in order to access the system. Those attacks, known as *spoof attacks*, are particularly relevant for voice and signature, but also other physical traits can be subject to spoof attacks, for examples it has been demonstrated in [101] that it is possible to construct fake fingerprints in order to bypass a fingerprint recognition system. It is also possible that an impostor may exploit biometric collision. As said in Section 1.2, since no biometrics has false acceptance rate equal to zero, it is possible that the biometrics of the attacker is similar to one of the many stored into the server database. Therefore he may target a specific template or engage a trial-and-error attack [8]. Even worse the attacker may take control of the database and use the informations to infiltrate other systems.

Since biometric traits cannot be changed, unlike passwords, in the last years the necessity to protect biometric templates has arisen. Both  $\mathcal{S}$  and  $\mathcal{C}$  can be interested in preserving the privacy of their data. Many privacy protection protocols for biometric recognition has been proposed in the past [11, 20]. The main part of those protocols use SMPC tools that guarantee privacy against a semi-honest adversary (Section 2.1), this choice is due to the superior efficiency of semi-honest cryptographic solutions with respect to malicious ones. In the following sections we present the state of art in privacy preserving biometric protocols, while our implementations are described in Chapter 4 and Chapter 5.

### 3.2 Iris recognition in the encrypted domain

Iris can be easily represented as a  $N$  elements bit vector [102, 103], and two irises can be compared using their Hamming distance (HD). Usually  $N = 2048$  [104] but it can also vary depending on the radial  $r$  and angular  $\theta$  resolution

used in the extraction process [103]. The angular resolution  $\theta$ , is defined by radial lines passing through the iris region, the radial resolution  $r$  instead is the number of points selected on each radial line. At the end of the encoding phase data from each sector, defined by  $r$  and  $\theta$ , is represented with two bits, therefore  $N$  is equal to  $2 \cdot r \cdot \theta$ . The simplicity of representation and comparison of iris recognition is attracting for privacy preserving protocols and many implementations have been presented in the past.

Luo et al. in [21] implemented an HE-based privacy preserving iris identification protocol based on Daugman iriscode [102] and tested the system on CASIA Iris Database [105], containing the images of 100 individuals. They used the Matlab code from [106] to extract the iris and mask code of length  $N = 9600$  bits, where the mask code is a vector, extracted from the eye image together with the iris template and representing the region altered by noise such as reflection, eyelashes and eyelids. On a Linux machine with AMD Athlon 64, 2.4 GHz and 2GB of memory, a single identification query requires 27.1 minutes on average (about 16.2s per record). This large time complexity is justified by the length of the iris code (9600), which is encrypted bit by bit with the corresponding 2048-bit ciphertext using Pailler cryptosystem [21]. For the secure comparison the protocol [21] relies on DGK scheme [107], but the client has to collaborate with the database owner to output the comparison result.

In [1] Blanton et al., instead, present a different approach for iris and finger identification, based on a hybrid HE and GC protocol. They represent iriscode as a 2048 bit vector. For the HE part of the protocol they use DGK cryptographic scheme [107, 108], it has reduced functionalities but has faster encryption and decryption times than Pailler ones, and it has a 1024-bit ciphertext (Pailler ciphertext is 2048-bit long). GC is used to perform comparison because it has minor time complexity than HE [109]. To better illustrate the advantages of their implementation of iris identification protocol, the authors test the protocol using both DGK cryptographic scheme [107] and Pailler's one [14]. The framework based on DGK's scheme is about 25% faster than the Pailler-based one, because they precompute most of the operations, and they optimize the multiplication protocol. Blanton et al. work is not exactly comparable with Luo et al. one because even using the Pailler



implementation, they consider a smaller iricode representation. The protocol has been implemented in C and runs on a 2.13-GHz dual-core processor. Offline and online computation times are summarized in Table 3.1. Online communication and computation complexity is indicated per record (using the notation "/rec") because it is influenced by database size. It is worth nothing that the comparison between two 2048-bit iricodes requires only 0.15ms.

Table 3.1: Offline and online performance of iricode, fingercode and minutiae based fingerprint identification [1]. Some of the overheads depend on the server’s database size, for this reason online complexity are indicated per record (“/rec”) and so they refer to a single match.

	Server Runtime		Client Runtime		Bandwidth	
	Precomp. ms	Online ms/rec	Precomp. ms	Online ms/rec	Precomp. KB	Online KB/rec
<b>Iriscode</b>	89	149.25	0	22.61	0.5	19.9
<b>Fingercode</b>	0.22	1.42	4.7	1.08	2.12	0.86
<b>Minutiae</b>	6	339	25	1876	16	294

Two different full-GC implementations of identification protocols are presented in [25] and [24]. In [25], authors use a secure filtering method to avoid an exhaustive search into the database. They also implement rotation, and consider 7 rotation for each iris. They used the ICE database [110, 111] it contains images of 243 eyes, for which the number of images for each eye is variable for a total of 2953 images. The authors tested the protocol on datasets of different size  $n$ . The resulting bandwidth is  $475n + 0.08n^2$  kilobytes and runtime of about 2.4 seconds for each match. They used two different machines for server and client. The client is equipped with a 2.66GHz quad core processor connected to a server with a 2GHz processor through a local area network. Given  $k$  the number of candidates used in the filtering phase, the resulting system requires approximately  $0.2n + 16.9k$  seconds instead of the  $16.8n$  seconds would need a system that employs a 1:1 exhaustive search.

Similar results have been obtained in the protocol presented in [24] by Luo et al. [24]. Luo et al. confront the test on two sets of iricodes, in the first set the length  $N$  of the iricode is 2048 and it is based on Daugman work [102] and the second based on the open source iris recognition system

in [103, 106], where  $N = 9600$ . The matching between two 2048-bits iriscodes needs 563ms and the transmission of 571KB, while the matching of two 9600-bit templates needs 2563ms and 2655KB of transmission. In the results is included the precomputation for OT because it needs to be executed every time the protocol is implemented. The OT offline time is independent from the database size but it depends on iriscode length. On the contrary circuit construction and circuit transmission is not counted because it has to be run only one time. Luo et al. tested the protocol also using a common average mask for all the iris. In this setting the performance is comparable with the one in [1], but using a full GC implementation. Tests have been implemented in Java and run on a Intel Core2 Duo CPU 3.00GHz processor with 8GB RAM on 64 windows 7 Professional, over the CASIA Iris Database (CASIA-IrisV3-Lamp) [105] that includes 3763 samples from 292 individuals. The efficiency of the protocol relies on the offline computation of the circuit garbling phase and circuit transmission.

A completely different approach has been proposed by Bringer et al. in [112]. The new identification framework, called GSHADE (*Generalized Secure Hamming Distance Computation*) relies on a hybrid use of OT and GMW[48]. GMW is a MPC primitive similar to GC, it also represents functions as binary circuit, but securely evaluates shares rather than gates. GSHADE has been implemented in C++ and tested using two machines equipped with 3.2-GHz processor. For a database of 320 iriscodes of 2048 bits each, this new protocol has a computational complexity three time larger than the hybrid protocol presented in [1] but it is 35 times faster.

After the widespread of Gentry's SHE, Yasuda et al. in [27] presented a general authentication protocol based on Gentry and Halevi SHE scheme [15]. This framework can be applied to any biometric represented with a 2048 bit vector and match score calculated with Hamming distance. The authors have introduced a packed representation of the biometric template that allows to compute the HD with only three products. They performed tests on a Intel Xeon at 3.07-GHz processor and have shown that the protocol in [27] needs only 18.10 milliseconds for computing HD. It is faster than the implementation in [28]. It is important to point out that Yasuda et al. assume the presence of three parties: server  $\mathcal{S}$ , client  $\mathcal{C}$  and a trusted third party  $\mathcal{A}$ .  $\mathcal{A}$  generates

passwords and gives public passwords to  $\mathcal{S}$  and  $\mathcal{C}$ , keeping the secret one for itself.  $\mathcal{C}$  generates the encrypted template vector while  $\mathcal{S}$  computes Hamming distance. Finally  $\mathcal{A}$  decrypts the output distance  $d$ , compares it against  $t$  and returns *yes* if  $d < t$ .

### 3.3 Face recognition in the encrypted domain

Erkin et al. identification protocol, presented in [2], is based on the eigenface method [113, 114]. A detailed description of eigenface protocol is out of the scope of this work. Hence we only give a simple sketch and we invite interested readers to read the original papers [2, 26, 113, 114] for further details. Each face image can be seen as a two dimensional array. The aim of the eigenface framework is to define  $k$  eigenfaces that can be used to describe all the faces into the database, as eigenvectors in linear algebra. For this reason a face template is a vector of  $k$  real numbers. For identification, the server looks for the index  $i$  of the images into the database such that the Euclidean distance between templates is lower than a threshold  $t$ . Unlike many following privacy preserving protocols, feature extraction [2] is carried out in the encrypted domain, relying on the homomorphic properties of Pailler cryptosystem ([14] and Section 2.3). Using the same properties, squared Euclidean distance is also implemented in the encrypted domain, while for the comparison step the authors rely on DGK scheme [107]. The recognition protocol has been evaluated on a computer with 2.4-GHz dual-core processor, using the “ORL Databases of Faces” [115]. The resulting runtime is about 40 seconds for matching against 320 images, that could be reduced to 18 seconds if precomputation is applied. Computational and communication complexity can be reduced by assuming that the eigenfaces for feature extraction are public, as shown in Table 3.2. This last assumption has been used in almost all the subsequently works on this topic.

Sadeghi et al. [26] improved Erkin et al. identification protocol, by proposing two different protocols: a full-GC, and an hybrid scheme (Section 2.5) where HE is used to compute distance and GC for comparison. Tests have run on a PC mounting a 2.6-GHz processor and the resulting protocol is 30% faster than that by Erkin et al. Moreover, the round complexity of the hybrid

Table 3.2: Computational and communication complexity of privacy-preserving face recognition [2].

DB size $n$	Complexity			Communication (KB)	
	Full Query	Computational (sec)		Full Query	Public Eigenfaces
		With precomputation	Public Eigenfaces		
10	24	8.5	1.6	2725	149
200	34.2	14.5	11.4	5497	2921
320	40	18	18.2	7249	4674

protocol is reduced from logarithmic in the size of the database of the protocol proposed in [2] to 6 moves while the full-GC protocol requires only two rounds. The main drawback of GC protocol is the huge offline complexity of hundreds of megabits, compared to few kilobytes of hybrid solution. Results for on line computation are summarized in Figure 3.4.

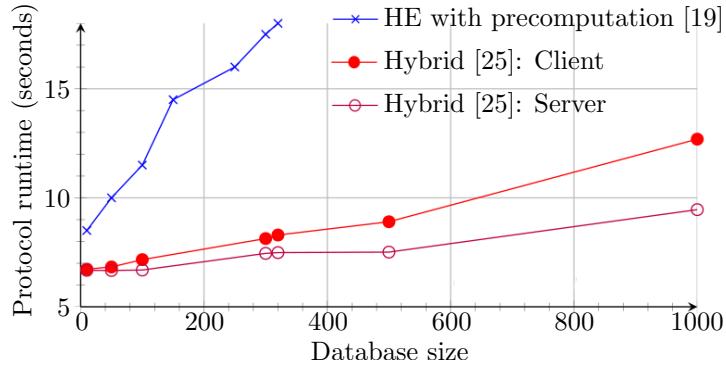


Figure 3.4: Runtime comparison of HE [2] and hybrid [26] implementations of the Eigenface protocol. Time complexity for full-GC implementation is not shown in the figure because authors were not able to compile the circuit due to memory restriction of the compiler. They estimate the GC protocol to be slower than the hybrid one.

In [22] the authors propose a new technique for feature extraction, called SCiFI, designed to improve privacy preserving identification protocols efficiency. In this case a face, is mapped into a bit vector of fixed length that represents the characteristics that almost every face has. Each element vector

represents one of this components and in the template it is equal to 1 if the face contains that components. In other words in [22] Osadchy et al. think of a face like a collage of known elements [22]. In SCiFI, the distance score between faces is calculated by using Hamming distance. In a first and offline, phase the server  $\mathcal{S}$  prepares the representation of the images in his database and also initialize the cryptographic algorithm. The online phase starts when the client  $\mathcal{C}$  acquire an image, then  $\mathcal{C}$  extracts the template representation and finally  $\mathcal{S}$  and  $\mathcal{C}$  run the secure matching protocol between client's probe and server's database in order to output a decision. For tests, on the new face recognition algorithm authors use FERET [116] (1196 images of 194 subjects) and PIE [117](2856 probes) databases of images. In the privacy preserving implementation, match score is evaluated by using Pailler Cryptosystem and then comparison is implemented by using a 1-out-of- $d$  OT, where  $d$  is the maximum value that the distance can assume. The experiments have been run on two computers equipped with a 2.6-GHz processor and a 2.8-GHz dual-core processor, representing respectively the client and the server. Offline precomputation took 213 second to client, almost half of this time is spent on preparing and sending homomorphic encryptions. The online time complexity is about 0.30 seconds for a single match and about 31s for an identification task on 100 images. In [118] and [119], performances had improved by expressing the SCiFI framework as a binary circuit and evaluated through GC [118] and GMW [119] protocols.

The GSHADE protocol (Section 3.2, [112]) has been tested against face templates generated by eigenfaces and SCiFI algorithms for identification tasks on two machines equipped with a 3.2-GHz processor. The speedup improvement factor ranges from 66 to 100 and communication complexity is comparable with previous implementations on the eigenfaces case, while in the SCiFI protocol the runtime decreases by a factor 4 – 5 and communication reduces by 14 for  $5 \cdot 10^4$  elements with respect to the implementation of [119].

One of the firsts non-interactive biometric authentication protocol is presented by Troncoso-Pastoriza et al. in [28] and is based on the SHE scheme presented in [15]. All the computation is moved on the server's side, the client has only to encrypt inputs, decrypt outputs (represented as a soft score of a

SVM) and compare the output against the threshold. In fact comparison is not carried out in the encryption domain. A C++ implementation of the system has been run on a machine equipped with a 3.30GHz processor. As databases he uses XM2VTS [120] and LFW [121] databases Troncoso-Pastoriza et al. implementation runs in 59 seconds, improving the results of 420 seconds with respect to the equivalent Pailler implementation [2]. On the contrary, the communication complexity is much higher (393 MB in [28] and 16.4 MB in the Pailler-based system) due to the larger expansion factor of a lattice based cryptosystem [15].

### 3.4 Fingerprint recognition in the encrypted domain

In order to guarantee the efficiency of the privacy preserving protocols, most of the proposed schemes for fingerprint recognition rely on the fingerprint representation [122], since the resulting template is a fix-length integer vector. In the original setting [122], the extraction process produces a template (*fingerprint*) of 640 features, each feature can be quantized into 256 values and requires 1 byte of storage.

Barni et al. in [23], used the fingerprint to implement a Pailler based privacy preserving identification protocol. In [23] authors also study the quantization method of [122] in order to find a trade off between feature bit length and accuracy. They demonstrate that the representation of a fingerprint through a vector of 192 features of 4 bits each guarantees an equal error rate of 6.7%. According to the results reported in [23], by representing a fingerprint with a vector of 96 features of 2 bits each, the equal error rate increases only of 7.6%, this last configuration is more appealing to be used in privacy preserving protocols. As a conclusion, when tested on a database of 64 identities the protocol in [23] runs in about 16 seconds on a machine equipped with a 2.4-GHz dual-core processor.

In [1] the authors use both fingerprint and minutiae [123] for identification on a database of 320 elements. Given a fingerprint  $X$ , a minutiae  $X_i$  is a planar point  $(a_i, b_i, \alpha_i)$ , where  $a_i, b_i$  are coordinates and  $\alpha_i$  is a direction on the plane. Therefore  $X$  can be represented as the set  $X = \{(a_1, b_1, \alpha_1), \dots, (a_{m_x}, b_{m_x}, \alpha_{m_x})\}$ ,

Table 3.3: Online performance of the fingercode identification system in [3].

Database size	Running time(sec)	Bandwidth (KB)
128	2.22	966.84
256	4.33	1927.71
512	9.12	3849.48
1024	18.11	7692.98

where  $m_x$  is the number of minutiae for the fingerprint  $X$ . Two minutiae are matching if the Euclidean distance between points is less than a threshold  $d_0$  and the distance between the two directions is less than a certain limit  $\alpha_0$ . Some tolerance values are necessary to compensate errors introduced by skin distortion or algorithms errors. Comparison of two fingerprints means finding the best match between two sets of points so that the number of matching minutiae is maximized. The protocol presented in [1] for both minutiae and fingercode representations, is an Hybrid HE and GC protocol and is similar to the iris one (Section 3.2). When compared with [23], using the same fingercode's length and a database of the same size, the protocol is 35 times faster. In fact, client's run time is 0.35s while server's one is 0.45s. In their minutiae implementation Blanton et al. use a fix number  $m$  of minutiae for all fingerprints, because the computational cost increases with  $m^2$  but nevertheless, the protocol, adapted to operate on minutiae, is less efficient. In fact according to [1] runtime increases significantly. Table 3.1 summarizes runtimes when 32 minutiae are used to representing a fingerprint.

In [3], Evans et al. present an hybrid implementation for fingercode-based identification. Authors assume that fingercode extraction is performed by client in plain domain. Each fingercode is a vector of length  $n$  and each element is a 8-bit integer and sever owns a database of  $m$  elements. The database is organized in a  $m \times n$  matrix in which each row is a vector representing some fingerprint. Euclidean distance protocol builds on the one proposed by Erkin et al in [2] and adopts the packing technique proposed by Sadeghi et al in [26]. In distance protocol, they move as much computation as possible to a preprocessing phase that can be done by server alone; moreover, in contrast

with previous works, most computation is done on client side and data are encrypted with the server public key of a homomorphic encryption scheme. On server side database elements are packed by columns and then encrypted. This solution allows to reduce the number of homomorphic operation because a single one performs arithmetic over several packed scalars and at the same time it uses less bandwidth than previous implementations. The matching protocol is carried out by using a garbled circuit formed by several subprotocols. Online runtimes of a java implementation, running on a two machines equipped with 2.0GHz processor and connected by LAN, are summarized in Table 3.3. They use a random generated 640 elements vector as benchmark and they confront it with a random database's element.

### 3.5 Multimodal Biometrics

Thanks to the use of SMPC techniques, biometric matching can be carried out in such a way that the parties involved in the computation cannot access the information or the result owned by the counterpart. In the works presented above, all the authors have found a trade off between accuracy and algorithmic simplification in order to make them suited for a SMPC implementation. It is evident, as pointed out in [11], that the main question is not *if* a certain computation can be carried out in the encrypted domain but *how efficiently* a certain solution can be implemented. The need for efficiency often leads to use tools and SMPC techniques secure in the semi-honest model, introducing in this way an imperfect security and therefore the possibility for malicious adversary to infiltrate the protocol. We believe that a solution can be found working at the same time on both the signal processing and cryptographic aspects of the problem, While regard to accuracy, a solution can be found employing more than a biometric trait.

Most real word applications for biometric recognition use only one source of information i.e. only one biometric probe is captured and processed. This systems have some limitations and weaknesses that can lead to lower accuracy and efficiency. A recognition process based on a single biometric can fail for a multitude of reasons. Data can be affected by *noise*, such as scars on fingerprint images or voice altered by a cold. Moreover two individuals can



have similar traits, as Golfarelli et al. state in [124], the number of differences between faces are only in the order of  $10^3$ , therefore inter-class similarities are quite common. Combining more than one source of information makes it possible to improve accuracy, and at the same time it allows to reduce template complexity. This is the solution we exploit in Chapter 5.

Such systems, usually known as *multimodal biometric systems*, are more reliable because they can access a certain number of different and independent pieces of information. In addition, this can be also a way to make spoof attacks more difficult to carry on. In fact forging more than one biometric trait can be challenging [125]. For this reason, novel devices are often equipped with more sensors.

Here we review some possibilities to take the best advantage from the availability of multiple sources of evidence [7, 125] (a summary can be found in Figure 3.5).

- *Multi-sensor*. Different sensors are used to capture the same biometric trait, in order to extract more information. For example in [132], Chang et al. use both 2D and 3D images of a face and combine them at the data level.
- *Multi-sample*. A Multi-sample system collects and processes multiple images of the same biometric trait, to anticipate variations that can occur or to have a better representation of the biometrics. The cell-phone's fingerprint sensor is an everyday example. Usually the enrollment or training phase of the system requires several images of the same fingerprint to have a complete set of images of the finger on multiple positions.
- *Multi-Instance*. Similarly to the previous case, a multi-instance system collects and processes images of several distinct instances of the same biometrics trait. This system does not need new sensors nor a new feature extraction and matching system. Multi-instance examples are systems that use both irises, or the Automated Fingerprint Identification System (AFIS) that obtains information from ten prints of the same subject.

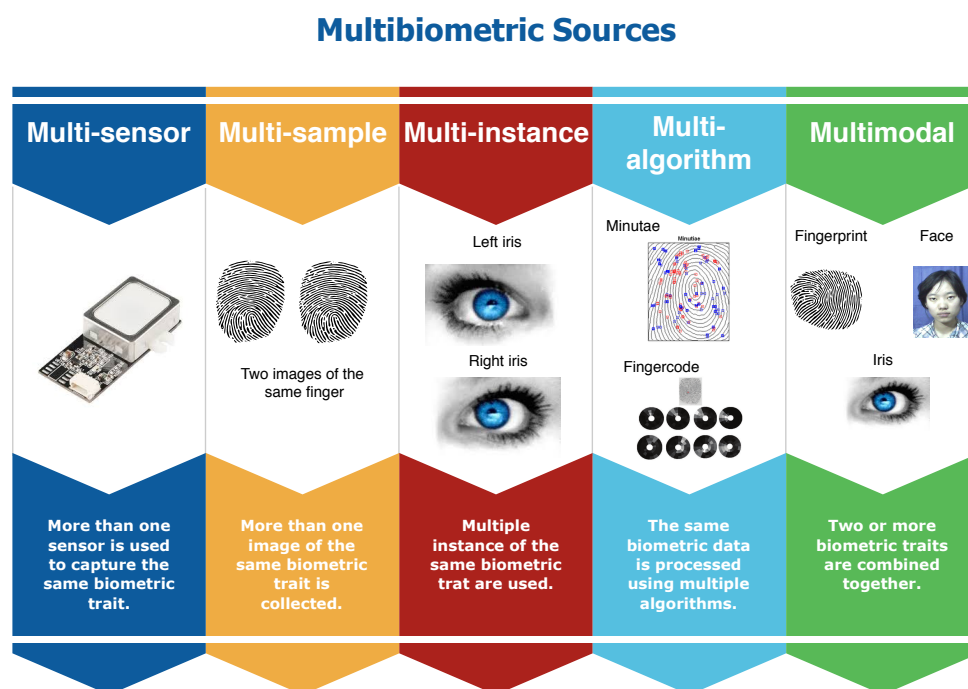


Figure 3.5: The possible sources of information in a multibiometric system: multi-sensor, multi-sample, multi-instance, multi-algorithm, multimodal. In the last scenario (multimodal) more than one biometric trait is used to achieve evidence. In all the others a single biometric is used. Images sources [126–131]

- *Multi-algorithm*. In these systems the same biometric feature is matched using more than one algorithm. For example Ross et al. [133] combine the matching score of a minutiae based fingerprint matcher to the score of a texture-based matcher to improve performance.
- *Multimodal*. In this last set of systems, information from two or more different biometric traits are combined together during the recognition process. Many combinations have been explored in the literature: face and iris [134] fingerprints, face, and iris [135], hand and face [136], etc.

In multimodal systems, the acquisition sequence indicates the way in which each source of evidence is acquired. Usually, each sample is acquired independently in a short time interval, even if in some cases probes can be obtained

at same time, i. e. face and iris can be collected almost simultaneously using two different cameras [7]. In order to output a decision, information can be processed *serially*, or in *parallel*. In the first modality the first source, for example fingerprint, is processed; then if the result is inconclusive the second source is elaborated, for example the face. This procedure can reduce computation time if a decision is made before going into all the biometric probes. On the contrary in parallel mode all the various evidences are processed independently at the same time and all the partial results are combined together, using an appropriate fusion scheme [7]. In our work (Chapter 5), we use a multimodal system, involving face and iris, to improve efficiency in a privacy protection authentication protocol. We process information derived from the two biometric traits in parallel.

In a multimodal biometric system, information can be merged in any of the steps, summarized in Figure 3.1, [7, 125, 137].

- *Signal Level Fusion.* Multiple samples may be combined together to extract a better template. Theoretically signal level fusion is preferable to the others, since fusion is carried out before any information is discarded, however performing fusion at the signal level is a difficult task because various modalities may not be easily combined and the access to raw data is not always possible.
- *Feature Level Fusion.* After feature extraction, this kind of systems combine all the features into a single biometric signature.
- *Score Level Fusion.* The match scores are combined to obtain a final result. It is the most used approach because accessing and combining the scores obtained by relying on different modalities is relatively easy.
- *Decision Level Fusion.* The outputs (usually accept or reject) of multiple classifiers are fused using various techniques as, for example, majority voting. Despite its simplicity, decision level fusion is highly suboptimal, since a great deal of possibly useful information has already been discarded in the previous steps.

To the best of our knowledge, Gomez-Barrero et al. [138] have proposed the only previous work on multibiometric privacy protection. In their work,

the authors present a general framework for multi-biometric template protection based on Pailler cryptosystem, in which only encrypted data is handled. In Gomez-Barrero et al. the client computes the encrypted distance while the server owns both secret and public keys. They assume that all involved parties act correctly, according to the honest-but-curious model, moreover they assume that an adversary may take control of the server, but  $\mathcal{C}$  and  $\mathcal{S}$  can not collude. The authors examine the outcome of the fusion of online signature and fingerprints, at three different levels of fusion: feature, score and decision level. The system presented by Gomez-Barrero et al. has a low computational cost (only one decryption on the server side and no encryptions at verification time), moreover it ensure a good accuracy (EER = 0.12%), with a required time for a single comparison of about  $5 \cdot 10^{-4}$ s. A drawback with the system described in [138], is that comparison is carried out on plain data by the server, thus introducing a breach into the security of the system.

### 3.6 Biometric recognition in malicious setting

Almost all the schemes presented in the previous sections have been designed to provide security in the semi-honest model because the security in the malicious setting can be achieved only at the cost of high complexity. Nevertheless there are some works on privacy preserving biometric authentication that are secure against a malicious attacker. For example, in [29] the author presents a general privacy protection authentication protocol and proves its security in the malicious model. In his work, Abidin combines verifiable computation with homomorphic encryption in order to guarantee security against a malicious adversary colluding with the server. Abidin assumes to have a semantically secure HE cryptographic scheme, that allows to homomorphically compute the distance between two encrypted inputs. He outlines a generic authentication framework, but does not provide results about the practical implementation of the protocol on the contrary he proves security against malicious adversaries.

In [30], Pathak and Raj present two speech-based authentication protocols. The first one is an interactive protocol based on Pailler cryptosystem which is secure against a semi-honest adversary, the second one is a non-interactive

protocol based on BNG [139] cryptosystem, which allows to perform an arbitrary number of additions and one multiplication between ciphertexts, and is secure against malicious attacks. During the protocols, the user can only observe masked data, moreover he cannot obtain any information on the other party's speech. Pathak and Raj also state that the privacy preserving protocols have the same accuracy of a non-private one using the same matching framework. In both cases the output is a probability value and the client checks if such a value is equal to zero or not in the plain domain. From the tests and the analysis reported in [30], it is clear that the interactive protocol is way more efficient than the malicious one. A single verification task, in fact, takes about 5 minute using the interactive protocol and about 5 hours using the non-interactive one, using encryption key of the same size. The big difference in time complexity is mainly due to the execution of the private inner product, that is a very expensive operation.

Recently, Gasti et al. [140] proposed a biometric authentication protocol based on simple garbled circuits and secure against malicious adversaries by relying on an untrusted third party (the cloud). The goal of the protocol is to minimize the amount of computation performed by the biometric owner's device (a smartphone), while also reducing the protocol execution time and without the necessity to rely on cut-and-choose techniques. In the protocol, the biometric owner acts as circuit constructor, the Cloud as circuit evaluator, while the server verifies the correctness of the circuit. The approach is secure against colluding biometric owner and Cloud, but not against a colluding server and Cloud.

## Chapter 4

---

# Biometrics Recognition Based on Somewhat Homomorphic Encryption

In this chapter we present two non interactive privacy preserving biometric recognition protocols. The two protocols have been published in the following papers:

- *G. Droandi*, and R. Lazzeretti. "SHE based non interactive privacy preserving biometric authentication protocols." In *IEEE 9th International Workshop on Intelligent Signal Processing (WISP), 2015* IEEE. Siena, Italy, May 2015.
- *G. Droandi*, "Non-interactive privacy preserving protocol for biometric recognition based on somewhat homomorphic encryption", in *Proceedings of the 14th European Conference on Cyber Warfare and Security 2015: ECCWS 2015*. Academic Conferences Limited, 2015, p. 355-362.

In the first one we describe a secure authentication protocol, while in the second we propose a privacy preserving identification scheme.

As introduced in Section 3.1, any biometric recognition system is composed by four fundamental steps (Figure 4.1). In the protocols proposed in this chapter the latter two are computed in the encrypted domain. In other words, by assuming that a client  $\mathcal{C}$  has already acquired the traits and extracted the biometric template in the plain domain,  $\mathcal{C}$  and  $\mathcal{S}$  compute the matching score and compare it against the threshold in a secure way. Since comparison is a complex operation in the encrypted domain, we perform it in clear on client side, paying attention to obfuscate the values (more details in Section 4.5) involved in the computation.

The security of both schemes relies on the somewhat homomorphic scheme proposed by Pisa et al. [34] and described in Section 2.3.2.

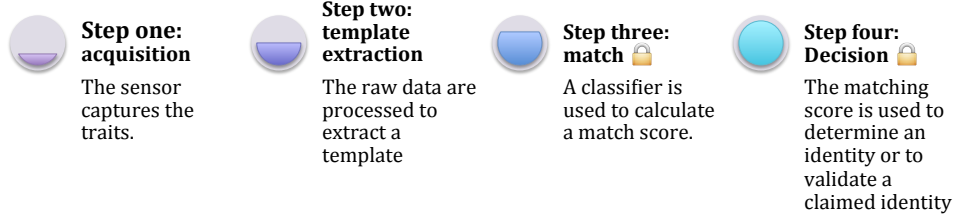


Figure 4.1: Biometric recognition system: main steps. In the presented privacy preserving protocols, the last two steps are carried out in the encrypted domain.

The two protocols have been applied to two different biometrics: iris and fingerprint. For the iris we used Daugman’s [102] iriscode, assuming that the feature vector is composed by 2048 bits. In Daugman’s work the match score is obtained by using a weighted Hamming distance (WHD) that considers also bit-masks, in order to include only significant bits, as described by Equation 4.1:

$$WHD(\mathbf{t}_1, \mathbf{t}_2) = \frac{\|(\mathbf{t}_1 \oplus \mathbf{t}_2) \wedge \mathbf{m}_1 \wedge \mathbf{m}_2\|}{\|\mathbf{m}_1 \wedge \mathbf{m}_2\|} \quad (4.1)$$

where  $\mathbf{t}_1 = (t_{1,1} \cdots t_{1,n})$  and  $\mathbf{t}_2 = (t_{2,1} \cdots t_{2,n})$  are the template vectors, while  $\mathbf{m}_1, \mathbf{m}_2$  are the mask vectors.

In our protocols, we discard the bitmasks and simply compare two templates by using the Hamming distance (HD, Equation 4.2) to keep the computational complexity low.

$$HD(\mathbf{t}_1, \mathbf{t}_2) = \|\mathbf{t}_1 \oplus \mathbf{t}_2\| = \sum_{i=1}^n t_{1,i} \oplus t_{2,i}. \quad (4.2)$$

Considering that the maximum value the distance can assume is 2048, it requires 11 bits to be represented, therefore the base of the cryptosystem must be at least  $b = 2^{11}$ .

For fingerprint matching, we use fingercode [122] template. In the original setting [122] the extraction process produces 640 components for each fingercode with a EER equal to 6.5%. To reduce the magnitude of SHE pa-

rameters, we decided to represent a fingerprint with a vector of 96 features of 2 bits each, because, according to the results reported in [23, 141], in this way the equal error rate is only 7.6% (Section 3.4). To match two fingerprint vectors it is necessary to compute the Euclidean distance. If  $\mathbf{t}_1$  and  $\mathbf{t}_2$  are  $n$ -element vectors then

$$\text{ED}(\mathbf{t}_1, \mathbf{t}_2) = \sqrt{\sum_{i=1}^n (t_{1,i} - t_{2,i})^2}. \quad (4.3)$$

The maximum distance between two features is 3 (assuming for example that of  $t_{2,i}$  is equal to zero, and  $t_{1,i}$  is equal to the maximum number that can be presented with two bits, i.e. 3), then the maximum value that  $(t_{1,i} - t_{2,i})^2$  can assume is 9 and, since a fingerprint template has 96 elements,  $\sum_{i=1}^n (t_{1,i} - t_{2,i})^2$  is equals  $9 \cdot 96 = 864$  in the worst case. This last value can be represented with 10 bits.

In the following sections, we describe the SHE-based implementation. Given a message  $m$ , we indicate with  $\llbracket m \rrbracket$  its encryption with the public key. If  $\mathbf{t}$  is a vector of  $n$  messages then  $\llbracket \mathbf{t} \rrbracket$  is the element-wise encryption ( $\llbracket t_1 \rrbracket, \llbracket t_2 \rrbracket, \dots, \llbracket t_n \rrbracket$ ).

Given the characteristic of the chosen biometric, we represent each template as a vector of  $n$  elements in a fixed interval:  $[0, 1]$  for iris codes and  $[0, 2^\ell]$ , with  $\ell \in \mathbb{N}$  for fingerprints. Therefore iris code template is a vector  $(i_1, i_2, \dots, i_{2048})$  with  $i_j \in \{0, 1\}$  and fingerprint one is  $(f_1, f_2, \dots, f_{96})$  with  $f_j \in \{0, 1, 2, 3\}$ .

Since the focus of this part of the research was the implementation of the somewhat homomorphic scheme and its applications, in the tests we used vectors of random generated features. We do not report the EER because it has no meaning for these tests.

## 4.1 Representing negative numbers in SHE

A contribution of this thesis is the extension of Pisa et al. cryptographic scheme [34] to negative numbers, which is necessary in distance computation. The extension is based on the observation that usually there are two ways



to compute the reduction modulo  $n$ . As we have seen in Section 2.3.2, we use two different notations to indicate the reduction of  $x$  modulo  $p$ . The notation  $[x]_p$  indicates the remainder in the interval  $[0, p)$ , while  $x \bmod p$  refers to the integer in the interval  $(-p/2, p/2]$ . In algebra two integers  $a, b$  are congruent modulus  $p$  (in symbols  $a \equiv b \pmod{p}$ ) if their difference  $a - b$  is an integer  $r$  multiple of  $p$ . This rule holds also for negative numbers, for example  $-8 \equiv 7 \pmod{5}$  because  $-8 - 7 = -15$  that is a multiple of 5. We exploit this property to encode also negative numbers. Given a base  $b = 2^k$ , we can encrypt integers in the interval  $(-b/2, b/2]$ . In this case the decryption function is performed as  $[c]_p \bmod b$ , for any ciphertext  $c$ . Obviously, to represent negative numbers, the base should be twice the maximum integer that can be obtained during computation.

## 4.2 General Protocol

In this section we describe the general privacy preserving biometric recognition protocols. The protocols involve two parties: the client  $\mathcal{C}$  and the server  $\mathcal{S}$ .

All the computations are performed on the server side, without interaction between the parties.  $\mathcal{C}$  only encrypts inputs and decrypts output. During the computation,  $\mathcal{S}$  never accesses the client features, he sees only encrypted values.

### 4.2.1 Authentication protocol

In this case the client must prove to be who he claims. During the enrollment phase,  $\mathcal{C}$  generates a feature vector from his biometric and encrypts it with his public key. Then he sends the encrypted template to the server for future use.  $\mathcal{S}$  gives back an identification tag.  $\mathcal{C}$  sends his public key along with the template, because the server needs both to operate. Even if  $\mathcal{S}$  maintains the database of enrolled individuals, it can not access to those because the templates are encrypted with client's public keys. The authentication protocol is divided into three steps (Figure 4.2):

1. *Client encryption.* The client generates an encrypted probe  $\llbracket \mathbf{q} \rrbracket$  of his biometric and sends it to the server along with an identification tag.

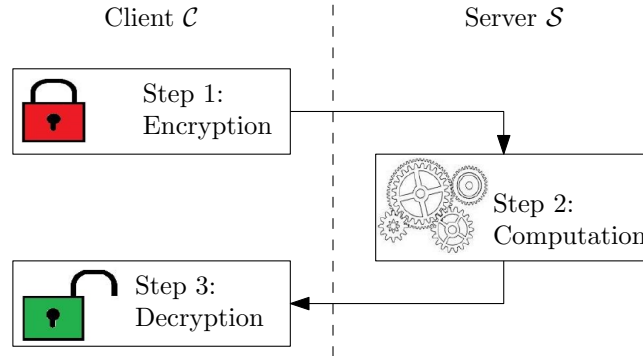


Figure 4.2: Three steps of the general recognition protocol.

2. *Server computation.* The server retrieves the template  $\llbracket \mathbf{s} \rrbracket$  stored into the database by using the identification tag and then runs the privacy preserving protocol to calculate the Hamming or the Euclidean distance  $\llbracket D(\mathbf{q}, \mathbf{s}) \rrbracket$  between  $\llbracket \mathbf{q} \rrbracket$  and  $\llbracket \mathbf{s} \rrbracket$ . Then  $\mathcal{S}$  should compare the output with the given threshold  $\varepsilon$ . Since comparison is an operation that cannot be performed on encrypted values without interaction, the server calculates  $d = \llbracket D(\mathbf{q}, \mathbf{s}) - \varepsilon \rrbracket$ . The client is authenticated only whether  $d < 0$ . However the magnitude of this value must be obfuscated because it could reveal information to malicious attackers, therefore  $\mathcal{S}$  opportunely blinds  $d$ , obtaining  $d_b$ , without changing its sign (we describe the blinding technique in Section 4.5). Finally  $\mathcal{S}$  sends  $d_b$  to the client (Figure 4.3).
3. *Client decryption.* The client decrypts the output. If the number is lower than zero, then his identity is confirmed.

### 4.2.2 Identification protocol

In this scenario the server owns a database containing the templates  $\mathbf{x}_1, \mathbf{x}_2 \dots \mathbf{x}_{N_d}$  of  $N_d$  individuals and can access each template  $\mathbf{x}_i$  in plain. Client on the other hand wants to know if a biometric probe is one of the  $N_d$  entries of  $\mathcal{S}$ , but he desires to protect his template. For this reason  $\mathcal{C}$  encrypts his probe, and we assume that the server has already the client's public key.

The identification protocol is divided into three steps (Figure 4.2):

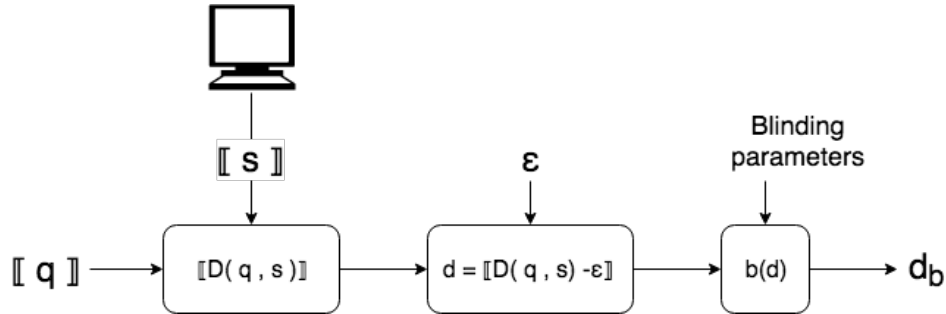


Figure 4.3: Authentication protocol: step 2 processing at the server. With the notation  $b(d)$  we indicate the blinding operation (see Section 4.5).

1. *Client encryption.* The client encrypts his biometric template  $\llbracket \mathbf{q} \rrbracket = (\llbracket q_1 \rrbracket, \llbracket q_2 \rrbracket, \dots, \llbracket q_n \rrbracket)$  with his public key and sends it to the server.
2. *Server computation.* The server loads the biometric templates and encrypts the values needed in the computation.  $\mathcal{S}$  evaluates the distance from the client's probe using the privacy preserving distance calculation described in Section 4.3 and Section 4.4, it repeats the process for each biometric into the database. As before, he subtracts the acceptance threshold  $\epsilon$  from each result. The vector containing all the outputs is permuted before being sent back to the client, to avoid revealing to the client the position of its biometrics in the database (Figure 4.4).
3. *Client decryption.* The client decrypts all the outputs. If one number is minor than zero then its biometrics is into the database.

We remark that in both protocols computation is performed only by the server, while client encrypts the input and decrypts the outputs. Moreover, in both protocols the final decision is carried out in plain by the client.

### 4.3 Computation of the Hamming distance under SHE

To compute the Hamming distance in the encrypted domain, we need to evaluate the XOR between ciphertexts. However, with the SHE encryption

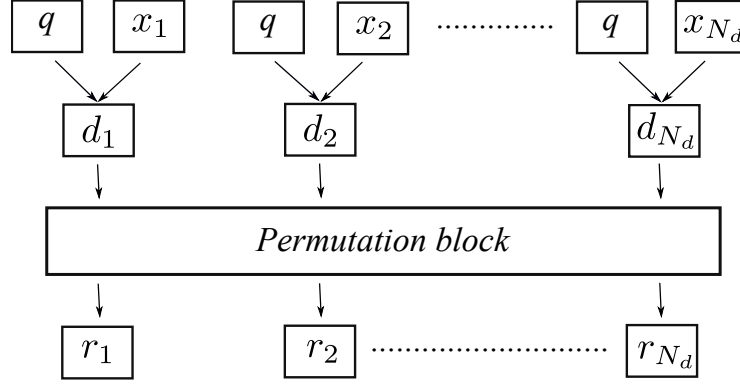


Figure 4.4: Identification protocol, step 2, server computation.

scheme we can perform additions modulus  $b$ , hence XOR between the probe element  $q_j$  and the database's element  $x_{i,j}$ , must be evaluated as

$$q_j \oplus x_{i,j} = q_j + x_{i,j} - 2 \cdot q_j \cdot x_{i,j}. \quad (4.4)$$

Once all database entries are encrypted with the client's public key, XOR can be computed as

$$\begin{aligned} \llbracket (q_j \oplus x_{i,j}) \rrbracket &= \llbracket q_i \cdot (1 - 2 \cdot x_{i,j}) + x_{i,j} \rrbracket \\ &= (\llbracket q_i \rrbracket \cdot \llbracket 1 - 2 \cdot x_{i,j} \rrbracket + \llbracket x_{i,j} \rrbracket) \end{aligned} \quad (4.5)$$

thanks to the homomorphic properties of the cryptosystem.

In the authentication protocol, the term  $\llbracket 1 - 2 \cdot x_{i,j} \rrbracket$  could be either sent to the server in the enrollment phase or calculated in advance by the server as  $\llbracket 1 \rrbracket + \llbracket -1 \rrbracket \cdot \llbracket x_{i,j} \rrbracket$ .

This solution is adopted in the authentication protocol where it is imperative to protect the privacy of the client and to prevent any unauthorized use of his biometric.

In the identification protocol, the server has access to the bits of the biometry  $\mathbf{x}_i$ , hence given the encrypted bits of the probe  $\llbracket q_j \rrbracket$ , it can compute

the XOR as

$$\llbracket q_j \oplus x_{i,j} \rrbracket = \begin{cases} \llbracket q_j \rrbracket & \text{if } x_{i,j} = 0 \\ \llbracket 1 - q_j \rrbracket = \llbracket 1 \rrbracket + \llbracket -1 \rrbracket \cdot \llbracket q_j \rrbracket & \text{if } x_{i,j} = 1. \end{cases} \quad (4.6)$$

Since server owns the database, he does not need to encrypt all the features of all the biometric in the database, but only the encryption of 1 and  $-1$  with the user's public key are necessary, thus avoiding several encryption operations and lowering memory requirements. This solution is used in the identification protocol in order to improve efficiency, to reduce memory, and time complexity. We remind that the representation of negative numbers is possible, as described in Section 4.1.

Finally, given a XOR implementation, an encrypted probe  $\llbracket \mathbf{q} \rrbracket$  and a database entry  $\llbracket \mathbf{x}_i \rrbracket$ , the  $\mathcal{S}$  computes the Hamming distance as:

$$\llbracket \text{HD}(\mathbf{q}, \mathbf{x}_i) \rrbracket = \left\llbracket \sum_{j=1}^n (q_j \oplus x_{i,j}) \right\rrbracket = \sum_{i=1}^n \llbracket q_j \oplus x_{i,j} \rrbracket. \quad (4.7)$$

#### 4.4 Computation of the Squared Euclidean distance under SHE

Comparing two fingerprint codes requires Euclidean distance (ED). Square root is an operation that the cryptosystem cannot evaluate, for this reason we compute the square Euclidean distance (SED). Given two template vectors  $\mathbf{t}_1, \mathbf{t}_2$ , then

$$\text{SED}(\mathbf{t}_1, \mathbf{t}_2) = \sum_{i=1}^n (t_{1,i} - t_{2,i})^2 \quad (4.8)$$

The cryptosystem allows operation modulus  $b$ , so, given a new probe  $\mathbf{q}$  and a database entry  $\mathbf{x}_i$ , server simply computes

$$\llbracket \text{SED}(q, x_i) \rrbracket = \left\llbracket \sum_{j=1}^n (q_j - x_{i,j})^2 \right\rrbracket = \sum_{i=1}^n (\llbracket q_j \rrbracket + \llbracket -x_{i,j} \rrbracket)^2. \quad (4.9)$$

In the authentication protocol, in order to compute the lowest possible

number of multiplication,  $\mathcal{S}$  stores in his database  $\llbracket -\mathbf{x}_i \rrbracket = (\llbracket -x_{i,1} \rrbracket, \dots, \llbracket -x_{i,n} \rrbracket)$  for  $i = 1 \dots D$ .  $\llbracket -\mathbf{x}_i \rrbracket$  could either been sent in the enrollment phase or it could has been calculated as  $\llbracket -1 \rrbracket \cdot \llbracket x_{i,j} \rrbracket$  for  $i = 1 \dots D$  and  $j = 1 \dots n$ .

On the contrary, in the identification protocol, where  $\mathcal{S}$  has access to the features in plain, he stores only the encryption of  $b - 1$  numbers, avoiding keeping all database encrypted with client public key (Equation 4.10).

$$\llbracket (q_j - x_{i_j})^2 \rrbracket = \begin{cases} (\llbracket q_j \rrbracket)^2 & \text{if } x_{i_j} = 0 \\ (\llbracket q_j \rrbracket + \llbracket -1 \rrbracket)^2 & \text{if } x_{i_j} = 1 \\ (\llbracket q_j \rrbracket + \llbracket -2 \rrbracket)^2 & \text{if } x_{i_j} = 2 \\ \vdots & \vdots \\ (\llbracket q_j \rrbracket + \llbracket -(b-1) \rrbracket)^2 & \text{if } x_{i_j} = b-1 \end{cases} \quad (4.10)$$

## 4.5 Blinding and comparison

In both the protocols, the distance  $D(q, x_i)$  must be compared against the threshold  $\varepsilon$ , but comparison evaluation is expensive in the SHE cryptosystem and it requires an interactive protocol. This problem can be overcome by computing the difference between the distance and the threshold as  $d_i = D(q, x_i) - \varepsilon$  under encryption i.e.  $\llbracket d_i \rrbracket = \llbracket D(q, x_i) - \varepsilon \rrbracket = \llbracket D(q, x_i) \rrbracket + \llbracket -\varepsilon \rrbracket$ . Two biometrics match if the result is negative.

During the authentication protocol, in order to better preserve the privacy of the server, the number  $\llbracket d_i \rrbracket$  is blinded, before disclosing it to the client. Unfortunately, additive blinding cannot be used because it could change the sign of  $d_i$ . On the other hand, multiplicative blinding is known to be much less secure [142]. The best solution, among those allowed by the encryption scheme in use, is to adopt a hybrid multiplicative/additive blinding: in practice two random values  $k_{i,1}$  and  $k_{i,2}$  are chosen and the final result is obfuscated by the server by computing  $\llbracket d_{b,i} \rrbracket = \llbracket k_{i,1} \cdot (D(q, x_i) - \varepsilon) + k_{i,2} \rrbracket = \llbracket k_{i,1} \rrbracket \cdot \llbracket D(q, x_i) - \varepsilon \rrbracket + \llbracket k_{i,2} \rrbracket$ .

Given the base  $b$ , the range of possible values is  $(-b/2, b/2]$ , hence  $k_1$  (we avoid the index  $i$  for simplicity) has to be chosen so that the product modulus does not exceed  $b/2$ , limiting the possible values to the range  $\left(1, \frac{b}{2(D_{\max} - \varepsilon)}\right)$ , where  $D_{\max}$  denotes the maximum value the distance can assume (in our

protocol  $D_{\max}$  is 2048 for iriscode and 864 for fingercode). Given  $k_1, k_2$  must be chosen in such a way that the sign of the final result does not change.  $k_2$  can be positive or negative, to define its range we must address the following two worst cases:

1. if  $D(q, x_i) - \varepsilon = 1$ , then  $|k_2| < k_1$ ;
2. if  $D(q, x_i) = D_{\max}$ , then  $|k_2| < b/2 - (D_{\max} - \varepsilon)$ .

By defining  $k_{2,\text{lim}} = \min\{k_1, b/2 - (D_{\max} - \varepsilon)\}$ ,  $k_2$  is randomly chosen in the range  $(-k_{2,\text{lim}}, k_{2,\text{lim}})$ . To guarantee high security levels, a very large base would be needed, however this would result in a high complexity for the system, hence a trade-off is needed (see Section 4.7 for more details about the trade-off we reached in our system). Finally the result is decrypted by the client.

In the identification protocol, the result vector is instead permuted to protect the identification number of the possible positive match. Once the vector is received, the client decrypts the outputs and the presence of a negative number proves that the client's biometric is into the database. In this protocol the client receives the distance of his probe from all the biometric templates in the database and this could be considered a security issue. Lets imagine that an attacker acting as  $\mathcal{C}$  wants to forge a template to infiltrate the system. First he creates a random template and submits it to the server. He receives various results and chooses the one with the smallest value to try to improve his forged template and then try to submit it to the server again. The new set of outputs the adversary receives is scrambled differently than before and he cannot get useful information from that. Therefore he can not know if after the second round his probe is more or less similar to the target database one. Multiplicative blinding could be also applied but it will make the computation slower because it involves a homomorphic multiplication. Permutation instead does not need any further operation in the encrypted domain, and therefore its complexity is negligible. Moreover after the permutation step it is impossible to determine to which client each output correspond. For this reason we decided to avoid the expensive blinding operation that should be applied to each of the  $N_d$  elements of the database increasing the complexity by  $N_d$  encrypted multiplications.

## 4.6 Complexity of the proposed protocols

The main advantage of the chosen cryptosystem is the possibility to directly evaluate the Hamming or Euclidean distance on the server, working on encrypted values. In our proposed protocols we have decided to work over integers, with base  $b$ . Indeed the protocols could be computed by using the DGHV scheme with base 2, but while the XOR between the bits could be more efficiently implemented without products, square values, sums and differences of the features require circuits composed by many AND gates. For example, we can observe that in the case of the Hamming distance, a reverse tree structure to compute the sums is composed by  $\log_2 n$  layers, wherein the  $i$ -th layer is made by  $n/2^i$  adders working on  $i$ -bit long inputs (each one requiring  $i$  AND gates). Hence for each biometric in the database,  $2 \cdot (n - 1) - \log_2 n$  products are needed for the sum and, being the depth of the tree  $\log_2 n$ , it is important that at least  $\log_2 n$  multiplications are allowed before the SHE incurs in a decryption error. The Euclidean distance requires even more AND gates. Since our solution works directly on integer values, only  $n$  products are evaluated in parallel to compute the XORs ( $\llbracket q_i \cdot (1 - 2 \cdot x_{i,j}) \rrbracket$  or  $\llbracket -1 \rrbracket \cdot \llbracket x_{i,j} \rrbracket$ ). Also for the Euclidean distance,  $n$  products are sufficient to evaluate in parallel the squares of  $q_i - x_{i,j}$ .

In the authentication protocol, we need an extra product for blinding, hence for the computation of both the distances it is sufficient that the SHE scheme allows the evaluation of at least 2 products, if we consider that  $\llbracket 1 - x_{i,j} \rrbracket$  or  $\llbracket -x_{i,j} \rrbracket$  has been sent encrypted from  $\mathcal{C}$  in the enrollment phase, while if the server must compute them before executing the protocol, SHE must allow three multiplications.

In the identification protocol an average of only  $n/2$  products is required to compute the XORs because  $\mathcal{S}$  can access the templates stored in plain. Furthermore such products can be run in parallel, hence we only need that the SHE copes with the noise amplification due to one multiplication. Moreover, it is important to note that for distance computation the server does not need to encrypt all the entries in the database, but only few numbers. In particular, for the case of a Euclidean distance, by assuming that each feature is represented with  $\ell$  bits, if  $2^\ell - 1 < n \times N_d$ , it is more convenient to encrypt the  $2^\ell - 1$  values that the features can assume (the encryption of 0 is avoided)



Table 4.1: Size of the ciphertext, secret and public key as a function of the security parameter and the base.

$\lambda$	Base	Secret Key	Cipher	Public Key
10	2	10 B	0.1 KB	0.1 MB
	$2^{10}$	101B	1.2 KB	1.3 MB
	$2^{50}$	506 B	6.3 KB	6.6 MB
	$2^{100}$	1013 B	12.5 KB	13.8 MB
	$2^{150}$	1519 B	18.8 KB	21.7 MB
15	2	25Byte	7 KB	7 MB
	$2^{10}$	245 B	72 KB	70 MB
	$2^{50}$	1 KB	360 KB	352 MB
	$2^{100}$	2.4 KB	721 KB	704 MB
	$2^{150}$	3.5 KB	1081 KB	1056 MB
20	2	45 B	66 KB	216 MB
	$2^{10}$	451 B	657 MB	2.06 GB
	$2^{50}$	2.2 KB	3 MB	11 GB
	$2^{100}$	4.5 KB	6 MB	21 GB
	$2^{150}$	6.7 KB	10 MB	32 GB

and select the correct ciphertext given the real value of the feature, rather than encrypting the single features.

## 4.7 Experimental results

In this section we discuss the results of the authentication and identification protocols, obtained by a practical implementation of the proposed cryptosystem.

The SHE-based iris and fingerprint recognition protocols have been implemented and tested on a desktop equipped with a Quad-Core CPU (Intel i7 at 3,40GHz) and 16 GB RAM, mounting a 64-bit Windows OS, to measure the communication and computational complexity of the scheme in terms of bandwidth and runtime. According to Section 2.3.1, the secret key is in the range  $[b^{\eta-1}, b^\eta)$ , with  $b = 2^k$  and  $\eta = \lambda^2$  (Table 2.1), therefore it is represented with  $k\eta$  bits. Similarly the secret key and each element composing the

public key are represented with up to  $k\lambda^5$  bits. Table 4.1 shows how the size of ciphertext, public key and secret key increases as a function of the base  $b$  and the security parameter  $\lambda$ . We need at least  $b = 2^{50}$  in the authentication scenario, and  $b = 2^{12}$  in identification for which, if  $\lambda = 20$ , we obtain a public key of 11GB, and 2GB respectively. Therefore for our tests we considered  $\lambda = 10, 15$  for which the public key goes from 1.5 MB to 350MB depending on  $b$  and  $\lambda$ . More details can be found in Table 4.1 and in next sections.

We ran 150 tests and we have measured the average times (ms or s) required by single encryption, decryption, key generation and multiplication, by using a Java implementation of the SHE scheme (Table 4.2). Addition and secret key generation are not reported, because the corresponding runtimes are negligible. The runtime of all the other operations grows with the base's bit-length and most of all the security parameter. For high values of  $b$  and  $\lambda$ , public key generation is an expensive operation, but it must be executed only one time by the client and it does not affect the performance of the recognition protocols.

Table 4.2: The averaged time results needed for a single encryption, decryption, key generation and multiplication.

$\lambda$	Base	Public Key	Encryption	Decryption	Multiplication
10	2	0.01 s	0.20 ms	0.00 ms	0.00 ms
	$2^{10}$	0.04 s	0.17 ms	0.03 ms	0.00 ms
	$2^{50}$	0.58 s	0.27 ms	0.93 ms	3.00 ms
	$2^{100}$	2.03 s	0.63 ms	2.57 ms	8.07 ms
	$2^{150}$	3.59 s	0.93 ms	4.80 ms	15.13 ms
15	2	0.18 s	0.60 ms	0.00 s	0.01 s
	$2^{10}$	3.30 s	2.90 ms	0.01 s	0.10 s
	$2^{50}$	2 min 39 s	15.68 ms	0.18 s	1.11 s
	$2^{100}$	4 min 42 s	28.58 ms	0.45 s	3.01 s
	$2^{150}$	8 min 56 s	44.00 ms	0.75 s	5.66 s

### 4.7.1 Authentication protocol

We now analyze more in depth the authentication protocol. To optimize the protocol, distance computation is parallelized into 4 threads.

Table 4.3: Iriscode and fingercode templates communication complexity in the authentication protocol.

	$\lambda$	$2^{50}$	Base $2^{100}$	$2^{150}$
<b>Iris</b>	10	13 MB	25 MB	38 MB
	15	750 MB	1.4GB	2 GB
<b>Finger</b>	10	606 KB	1.2 MB	1.7 MB
	15	34 MB	68 MB	102 MB

**Iris.** We considered the algorithm proposed in [24, 102], where an iris is represented through a vector of 2048 binary features, hence the Hamming distance is used and calculated as described in Section 4.3. As said before, we need at least  $b = 2^{11}$  to encode the maximum distance value and  $b = 2^{12}$  account for also negative number's representation (Section 4.1). For  $\lambda = 10$  we ran the protocol on different bases ( $2^{50}, 2^{100}, 2^{150}$ ). As to blinding, for  $b = 2^{50}$  the choice of  $k_1$  is in the range  $(0, 2^{39})$ , for  $b = 2^{100}$   $k_1 \in (0, 2^{88})$ , while for  $b = 2^{150}$   $k_1 \in (0, 2^{110})$ . As shown in Table 4.3 with  $\lambda = 15$ , the memory occupied by a single iris template grows significantly with base increases (for a 150 bits base it takes about 2GB) and during server computation three 2048-elements vectors must be stored at the same time (that needs about 6 GB) plus the public key (about 2GB). Due to those memory problems, we performed tests only for  $b = 2^{50}$ . 2048 ciphertexts are transmitted from  $\mathcal{C}$  to  $\mathcal{S}$  and a single ciphertext from the server to the client, resulting in a total transmission of 13MB, 25MB and 38MB respectively for each base with the lowest security parameter considered, while with the biggest  $\lambda$  the total transmission is 750MB (Table 4.3). We performed tests on randomly chosen bit vectors to simulate a possible iris probe. We repeated the tests 50 times and the average results for each part of the protocol are shown in Table 4.4. As expected, independently from the parameter set, the most expensive part

is the server’s one. With  $\lambda = 10$ , the whole computation takes some seconds (from about 3 to 17 seconds), while with  $\lambda = 15$  server computation is really slow, it takes about 14 minutes (we remember that only  $b = 2^{50}$  has been tested for memory problems).

Table 4.4: average execution time of the iricode authentication protocol.

$\lambda$	Base	Step 1	Step 2	Step 3
10	$2^{50}$	1.2 s	2.0 s	0.2 ms
	$2^{100}$	1.2 s	5.3 s	4.3 ms
	$2^{150}$	1.8 s	9.7 s	5.6 ms
15	$2^{50}$	29s	14 min 33 s	0.2 s

**Fingerprint.** We considered the system described in [23, 141], and we represented each fingerprint with a vector of 96 features of 2 bits each, randomly chosen to simulate a template. As outlined at the beginning of the chapter, the maximum squared distance value is 864 and it can be represented with 10 bits. For each security parameter’s value we run the tests using different bases ( $2^{50}, 2^{100}, 2^{150}$ ). The vector length of the fingerprint is lower than the iricode, allowing us to use a bigger base in the tests (for  $\lambda = 15$  and  $b = 2^{150}$  the public key is about 1GB, a iricode template about 2GB, while a fingerprint template is only about 101MB). In this case, 96 ciphertexts are transmitted from client to server and one from server to client, with a bandwidth of about 102 MB in the biggest parameter set considered. Bandwidths for different bases and  $\lambda$  setups are summarized in Table 4.3. As for the iricode case, the most expensive part is server’s computation, which takes up to 3 minutes. In this case execution time is faster than in the iris case, due to the lower number of elements for each fingerprint representation. Results obtained for fingerprint matching are summarized in Table 4.5.

As expected, the runtimes needed by the SHE implementation of the privacy preserving iris and fingerprint matching protocols are by far larger than the execution time of protocols based on Paillier HE or GC. Nonetheless in our protocol all the computation is moved into the server side and no interaction is

Table 4.5: Fingerprint authentication protocol’s average execution time

$\lambda$	Base	Step 1	Step 2	Step 3
10	$2^{50}$	0.03 s	0.10 s	1 ms
	$2^{100}$	0.06 s	0.30 s	3 ms
	$2^{150}$	0.09 s	0.53 s	5 ms
15	$2^{50}$	1.37 s	37 s	0.18 s
	$2^{100}$	2.80 s	1min 48 s	0.45 s
	$2^{150}$	4.25 s	3 min 17 s	0.79 s

needed. Moreover, runtimes can be reduced by using powerful servers allowing for parallelization across more threads.

In [28] the author presents a fully non interactive face verification protocol with a structure similar to ours. The author uses Gentry’s homomorphic cryptosystem [15] extended to integer values, where the final match score is a 22 bits integer. In Troncoso-Pastoriza work, the server computation runs in 12.3 seconds, but a face is represented using 5200-dimensional Gabor vector with three bits elements. Our implementation of finger and iris authentication protocol is faster for  $\lambda$  equal to 10, while for  $\lambda = 15$ , only the fingerprint protocol has comparable runtimes, for a base of 50 bits. The difference in performances is given by both the distinct homomorphic scheme but also by the total bits used in integer representation.

#### 4.7.2 Identification Protocol

Regarding the identification protocol, we use the same representation of the authentication scheme (Section 4.7.1).

**Iris.** For iris matching, since in this scenario we do not need extra bits for blinding, we can choose  $b = 2^{12}$ , hence public key size is 1.5MB if  $\lambda = 10$  and 77MB if  $\lambda = 20$ . We also point out that in identification protocols, for each client  $i$ ,  $\mathcal{S}$  must store only the template  $\llbracket \mathbf{x}_i \rrbracket$ , possibly all the terms  $\llbracket 1 - 2 \cdot x_{i,j} \rrbracket$ , and the values  $-1, 1$  encrypted with  $\mathcal{C}$ ’s public key (Section 4.3). As in the previous section, we performed tests on randomly chosen bit vectors to simulate a possible iris database. The tests have been repeated 50 times.

Table 4.6: Iris and fingerprint identification protocol’s average execution time

$\lambda$	Biometric	Base	Step 1	Step 2	Step 3
10	iris	$2^{12}$	0.33 s	0.12 s	0.00 s
	fingerprint	$2^{11}$	0.02 s	0.01 s	0.00 s
15	iris	$2^{12}$	7.10 s	54 s	0.01 s
	fingerprint	$2^{11}$	0.34 s	3.76 s	0.01 s

We expect long execution time due to the time required for each multiplication. Table 4.6 summarize results. We measured the average execution time in seconds of each part of the protocol with respect to a database of a single element. Hence the total time must be multiplied by a factor  $N_d$ . As shown in Table 4.6, for  $\lambda = 15$  the most expensive part in terms of execution time is server computation, which takes almost a minute, while the initial client encryption for all the 2048 features takes some seconds (and client decryption is negligible). On the other hand, for  $\lambda = 10$  the execution of the whole protocol takes less than a second, but is less secure.

A vector of 2048 ciphertexts is transmitted by the client to the server and another vector of the same size from the server to the client. The total transmission requires 6MB and 346MB for  $\lambda = 10$  and  $\lambda = 15$  respectively.

**Fingerprint.** In the identification scenario,  $\mathcal{S}$  can access the stored biometric, hence to evaluate the Euclidean distance, only the encryptions of  $-1$ ,  $-2$  and  $-3$  are needed on servers side, avoiding as before the problem of encrypting and storing many features. We remind that each fingercode templates is a 96-elements vector and each feature is represented with 2 bits (as outlined in Section 4.7.1) moreover the representation of negative numbers is possible thanks to our extension (Section 4.1). To allow negative number representation, we take  $b = 2^{11}$ . As for the iris case, to simulate the fingercode database, we used randomly chosen vectors of numbers in  $\{1, 2, 3\}$ . As we discussed at the beginning of Section 4.7 we let  $\lambda = 10$  and 15. With this set of parameters public key is respectively 1.4MB and 84MB. For  $\lambda = 10$  the protocol takes less than a second, while for the larger security parameter the most expensive phase is server’s computation, which takes about 4 seconds for each biometrics

in the database. At the beginning of the protocol, the client transmits 96 ciphertexts to the server, at the end  $\mathcal{S}$  sends to  $\mathcal{C}$  a vector of the same size of the database. For  $\lambda = 10$ , each encrypted fingerprint of 96 feature requires about 120KB, and a ciphertext about 1.3KB. Therefore, the resulting bandwidth, given the number of database elements  $N_d$ , is about  $(120 + N_d \cdot 1.3)$ KB. For  $\lambda = 15$ , bandwidth for a fingerprint template is 7.4MB and a single ciphertext is about 79.3KB. Results are summarized in Table 4.6.

For  $\lambda = 10$ , the time required by our implementation of the privacy preserving iris and fingerprint protocols has similar performances to GC [24] or Pailler [23] implementations. For  $\lambda = 15$ , the time needed by the SHE implementation is instead by far larger than the execution time of GC or Pailler protocols. The full-GC implementation for iris matching in [24] needs less than a second, but both circuit garbling and circuit transmission are pre-computed. Nonetheless, in our protocol all the computation is moved on the server side, and no interaction is needed, making it appealing for an offline database search. Running times can be lowered by using powerful servers allowing a greater number of threads for parallelization. The  $N_d$  matching operations can be also evaluated in parallel.

## Chapter 5

---

# Multi-Modal Biometrics authentication in the Malicious Setting

As detailed in Section 3.1, the great majority of works on biometric recognition protocols focus on the use of innovative SMPC cryptographic primitives. In Chapter 4 we presented two non interactive biometric recognition (an identification and an authentication) systems based on a SHE encryption scheme (Section 2.3). In this chapter, instead, we follow a less investigated approach, we worked on the signal processing side of the problem looking for a biometric recognition protocol which is better suited to be implemented in a SMPC framework. In fact, as highlighted in the introduction (Section 1.4) and in [11], this strategy may help to reduce the complexity of the resulting SMPC protocol.

The complexity of the biometric feature and therefore of the recognition system, can be reduced by using more than a single biometric trait in the process and therefore we choose a multimodal system. As we have seen in Section 3.5, a multimodal recognition system consist in the combination or *fusion* of two or more distinct biometric traits. In this way it is possible to achieve the same accuracy of a single modality protocol but using less feature and therefore the implementation requires less operations.

In this chapter, we present SEMBA, a *SEcure Multi-Biometric Authentication* protocol [143], a multimodal biometric authentication protocol which achieves a better trade-off between efficiency and accuracy with respect to the single modality subprotocols composing it. Specifically, SEMBA analysis face and iris templates and can be easily implemented by using Secure Multiparty Computation protocols.

SEMBA is published in the paper:

*G. Droandi, M. Barni, R. Lazzeretti, T. Pigata. "SEMBA: SEcure Multi-*



*Biometric Authentication*”, In ArXiv e-prints, 1803.10758, 2018.

Multimodal biometric systems are commonly used to decrease intra-class similarity in the presence of noise and other distortion [7]. We demonstrate that by using a properly simplified representation of the two biometric traits, the multimodal protocol can reach the same accuracy of a corresponding single-modality systems based on more accurate and complicated representation of a single biometric (particularly of the iris), but with a lower computational complexity. This is a crucial advantage in privacy preserving protocols. We also show that system designers could decide to exploit the superior performance allowed by multimodal recognition to improve accuracy, maintaining the same complexity of the single modality protocols. The security of SEMBA is based on SPDZ [35, 36], described in (Section 2.4).

To be used with SPDZ we need biometric recognition systems that have a fixed length template, and a simple match function that can be implemented using few operations. At the same time to allow our search for efficiency, we have the necessity to change the length of the vector in order to find the best configuration for our purpose. For iris, between the possible choices in literature, we choose to use the system presented in [103] since it matches to all our needs. For iris template extraction, we use the open source Matlab code [106] (as Luo et al. in [21]). The source code is part of the work [103], the distance and the template proposed respect the characteristics needed to be used with s.p.e.d. We consider a binary template of  $N$  features (*iriscode*), and a corresponding noise mask (Section 3.2 and Chapter 4). Since in this chapter we want to find a trade off between complexity and efficiency, we tested in plain domain many possible combinations of  $\theta$  and  $r$  in order to find a value of  $N = 2 \cdot \theta \cdot r$  (Chapter 3.2), looking for one guaranteeing a low complexity and a low EER. As result, our system obtains the same accuracy of the stand alone iris authentication protocol described in [103]. In contrast with what we did in Chapter 4, here we use a weighted Hamming distance (Equation 4.1), where the weights depend on the masks bits. In this way, only significant bits are used to calculate the distance between the two templates. In the output of the Matlab code, the value  $m_{i,j} = 1$  of the mask vector indicates that the bit is affected by noise and the corresponding bit in the iriscode template must be excluded by the computation. Keeping the same notation of Chapter 4,

Equation 4.1 can be also written as

$$\begin{aligned} WHD &= \frac{\|(\mathbf{t}_1 \oplus \mathbf{t}_2) \wedge (\overline{\mathbf{m}}_1 \wedge \overline{\mathbf{m}}_2)\|}{N - \|\mathbf{m}_1 \vee \mathbf{m}_2\|} \\ &= \frac{\sum_i^N [(t_{1,i} \oplus t_{2,i}) \wedge (\overline{m}_{1,i} \wedge \overline{m}_{2,i})]}{N - \sum_{j=1}^N m_{1,j} \vee m_{2,j}}. \end{aligned} \quad (5.1)$$

Where we indicate with  $\bar{a}$  the negation of a bit feature  $a$ , equal to  $1 - a$ .

The previous equation is equivalent to

$$WHD = \frac{\sum_{i=1}^{N-1} (t_{1,i} \oplus t_{2,i}) \vee (\overline{m}_{1,i} \vee \overline{m}_{2,i})}{\sum_{i=0}^{n-1} m_{1,i} \vee m_{2,i}}. \quad (5.2)$$

In the original work [103], iris templates are shifted  $m$  times, to bypass the error introduced by possible movements of the owner and to improve accuracy. The weighted Hamming distance is computed for each of them, and the lowest WHD is used to authenticate the iris owner. In his work, Masek recommend to perform  $m = 8$  shifts; to reduce the computational complexity, we did not consider rotation. In Section 5.4 we analyze the results of our tests on template length, and in Section 5.5 we use them to set the parameters for the tests in the encrypted domain.

For the face template, we use Tuck and Pentland eigenfaces [113]. The eigenface algorithm determines a set of vectors that can be used to generate a *face-space* and to project all the faces into the database (Section 3.3). Given a face image  $\Gamma$ ,  $\Gamma$  is projected into the *face-space* defined by eigenfaces. The resulting template is the vector  $\mathbf{\Omega} = [\omega_1 \dots \omega_k]$  where each  $\omega_i$  describes the contribution of each eigenface in representing the input image. In order to find the image that best matches  $\Gamma$ , we look for the projection vector  $\Omega_j$  among all the stored images, that minimizes the Euclidean distance

$$\|\mathbf{\Omega} - \mathbf{\Omega}_j\| = \sqrt{\sum_{i=0}^k (\Omega_i - \Omega_{j,i})^2}. \quad (5.3)$$

The general authentication protocol is the same of Chapter 4, Section 4.2.1 and described in Figure 3.2. The client  $\mathcal{C}$  is already enrolled in the database

owned by the server  $\mathcal{S}$ . Then  $\mathcal{S}$  needs to verify  $\mathcal{C}$  identity, but the client does not want to disclose his biometric features to  $\mathcal{S}$ . For this reason match score calculation and comparison are carried out in encrypted domain (Figure 4.1).

First we present the general lines of the stand alone iris and face protocols (Section 5.1 and Section 5.2). Then we describe the secure protocol for multi-biometrics recognition (Section 5.3). After we present the tests we run in plain regarding iris, face and on multi-biometrics protocols. (Section 5.4). Finally in Section 5.5 we analyze the outputs of the tests in the encrypted domain.

## 5.1 Iris authentication protocol in SPDZ

The server,  $\mathcal{S}$ , and the client,  $\mathcal{C}$ , cooperate to complete the protocol. In our implementation we assume that the offline part of the SPDZ (Section 2.4), i.e. initial setting, has already been completed. For this reason  $\mathcal{C}$  and  $\mathcal{S}$  both have shares of a new probe  $\mathbf{t}_1$ , of a template  $\mathbf{t}_2$  given in a previous enrollment phase, and of the corresponding masks  $\mathbf{m}_1, \mathbf{m}_2$ . Then they compute WHD as described in Section 5.1.1 (Equation 5.4). Finally the result of WHD is compared with a threshold as described in Section 5.1.1 (Equation 5.5, Equation 5.6 and Equation 5.7). The threshold is set during tests on plain values (5.4).

### 5.1.1 Hamming distance and comparison in SPDZ

Section 2.4 shows that SPDZ system supports operations modulus  $p$ . Each binary element of an iris feature is hence encrypted as a modulo  $p$  integer  $\langle a \rangle$ -share. For this reason, to implement the Hamming distance computation as in Equation 5.1, we need to implement the logical operations  $\oplus, \vee, \wedge$  as a combination of integer operations  $+, -, \cdot$ . The correspondences between binary and integer operations are detailed in Table 5.1.

Given the implementation of the binary operations, we can describe the SPDZ-based implementation of WHD. By indicating with  $\mathbf{t}_1 = (t_{1,1}, t_{1,2}, \dots, t_{1,N})$  and  $\mathbf{t}_2 = (t_{2,1}, t_{2,2}, \dots, t_{2,N})$  two binary iris feature templates, where  $N$  is the number of features, we denote with  $\langle \mathbf{t}_i \rangle$  the vectors containing the shares of each element, i.e. the vector  $(\langle t_{i,1} \rangle, \langle t_{i,2} \rangle, \dots, \langle t_{i,n} \rangle)$  for  $i = 1, 2$ .

Table 5.1: Correspondence table between binary and integer operations

Binary	Integer
$a \oplus b$	$a + b - 2 \cdot a \cdot b$
$a \wedge b$	$a \cdot b$
$a \vee b$	$a + b - a \cdot b$
$\bar{a}$	$1 - a$

Since  $\bar{\mathbf{m}}_1 \wedge \bar{\mathbf{m}}_2$  is equivalent to  $\overline{\mathbf{m}_1 \vee \mathbf{m}_2}$ , the Hamming distance in Equation 5.2 can be computed as:

$$\frac{\sum_{i=1}^N \{(t_{1,i} + t_{2,i} - 2 \cdot t_{1,i} \cdot t_{2,i}) \cdot [1 - (m_{1,i} + m_{2,i} - m_{1,i} \cdot m_{2,i})]\}}{N - \sum_{j=1}^N (m_{1,i} + m_{2,i} - m_{1,i} \cdot m_{2,i})} \quad (5.4)$$

Division is a very expensive operation in SPDZ, hence the denominator is multiplied by the acceptance threshold before comparison. By letting

$$num = \sum_{i=1}^N \{(t_{1,i} + t_{2,i} - 2t_{1,i}t_{2,i}) [1 - (m_{1,i} + m_{2,i} - m_{1,i}m_{2,i})]\} \quad (5.5)$$

and

$$den = N - \sum_{j=1}^N (m_{1,i} + m_{2,i} - m_{1,i} \cdot m_{2,i}), \quad (5.6)$$

the authentication check corresponds to

$$num < \tau \cdot den. \quad (5.7)$$

For the comparison we use the protocol described in Section 2.4.

### 5.1.2 Complexity

Computing  $\mathbf{t}_1 \oplus \mathbf{t}_2$  and  $\mathbf{m}_1 \wedge \mathbf{m}_2$  requires  $N$  multiplications each, one for each element of the template; moreover,  $N$  multiplications are required to compute  $(\mathbf{t}_1 \oplus \mathbf{t}_2) \wedge (\mathbf{t}_1 \wedge \mathbf{t}_2)$ . The total cost associated to the computation of  $num$  is  $3N$  multiplications. Multiplication between shares requires data transmission (see

Section 2.4), slowing down the computation. To increase efficiency, we split the multiplication protocol into two parts. First we calculate  $\langle \varepsilon_i \rangle = \langle t_{1,i} \rangle - \langle a \rangle$  and  $\langle \delta_i \rangle = \langle t_{2,i} \rangle - \langle b \rangle$ , for all  $i = 1 \dots N$ . Then, to partially open the values, both server and client exchange the shares by using a packet for all  $\varepsilon$ 's and one for all  $\delta$ 's. In this way, we need only two transmissions for  $N$  multiplications. Therefore for the numerator we need only 6 transmissions. The computation of *den* (Equation 5.6) has a negligible complexity since  $m_{1,i} + m_{2,i} - m_{1,i} \cdot m_{2,i}$  has already been calculated for all  $i$ 's in Equation 5.5. Finally, we need a multiplication between *den* and  $t$ , and  $\ell$  multiplications for the comparison, where  $\ell$  is the number of bits necessary to represent a modulo  $p$  integer (see Section 2.4). Therefore, we need  $3N + \ell + 1$  multiplications but only  $2\ell + 7$  transmissions for the iris protocol (Table 2.3).

## 5.2 Face authentication protocol in SPDZ

As in the previous section, we assume the offline part of the SPDZ (Section 2.4) initial setting has already been completed. For this reason  $\mathcal{C}$  and  $\mathcal{S}$  both have shares of a new probe  $\mathbf{\Omega}$ , and a template  $\mathbf{\Omega}_i$  given in a previous enrollment phase. Each face feature has been rounded to be represented in  $\mathbb{F}_p$  (we avoid rounding operator for simplicity). Then both parties cooperate to compute SED as described in Section 5.2.1 and finally compare the score (Equation 5.9) with a threshold set during the tests in the plain domain.

### 5.2.1 Euclidean distance in SPDZ

Given the projection  $\mathbf{\Omega}$  of the query face image, the face-based biometric authentication protocol must evaluate the Euclidean distance  $\|\mathbf{\Omega} - \mathbf{\Omega}_j\|$  (Equation 5.3) where  $\mathbf{\Omega}_j$  represents the projection of the registered image  $\Gamma_j$ . In the following equation (Equation 5.8)  $\omega_i$  indicates an element of the face  $\mathbf{\Omega}$  and  $\omega_{j,i}$  the  $i$ -th element of the projection  $\mathbf{\Omega}_j$ . Since the square root cannot be evaluated efficiently in SPDZ, we instead use the squared Euclidean distance (SED):

$$\text{SED} = \sum_{i=1}^k (\omega_i - \omega_{j,i})^2. \quad (5.8)$$

The SED score is compared (Equation 5.9) against the squared threshold as described in Section 2.4

$$\text{SED} < \tau^2. \quad (5.9)$$

### 5.2.2 Complexity

The computation of the squared Euclidean distance needs the evaluation of  $k$  squares, but, as we did for WHD, we separate the square computation into two parts (Section 5.1.2), hence only a transmission is necessary. The complexity of multiplication and square they are similar, therefore in the following we consider squaring as multiplications. For the comparison, we need  $\ell$  products (Section 2.4), as in the iris authentication protocol. The complexity of the protocol is hence given by  $k + \ell$  products (Table 2.3).

## 5.3 SEMBA: the fusion protocol in SPDZ

We now describe the SEMBA protocol, a multimodal system that merges iricode and face template at the score level (Figure 4.1) in a privacy preserving way. As before, we consider that the offline part of the protocol has already been computed, and that both  $\mathcal{S}$  and  $\mathcal{C}$  have shares of a new probe and the template stored into the database. The protocol can be divided into four parts: distance calculation, normalization, linear combination, comparison and decision (Figure 5.1).

**Distance calculation.** Given shares of a new face template  $\langle \Omega \rangle$  and of a new iricode  $\langle \mathbf{t} \rangle$ , we calculate the distance between the new templates and the ones stored into the database  $\langle \Omega_j \rangle, \langle \mathbf{t}_j \rangle$  using SED and WHD. Squared euclidean distance and Hamming distance are calculated as described in Section 5.1.1 and 5.2.1.

**Normalization.** The output of the iris protocol is a real number in  $[0, 1]$ , while the output of the face recognition protocol is a squared number in  $[0, M], M \in \mathbb{R}$ . Among the solutions proposed in the past to overcome the problems generated by the differences between scores output by different biometric recognition systems (see [7] for more details), we

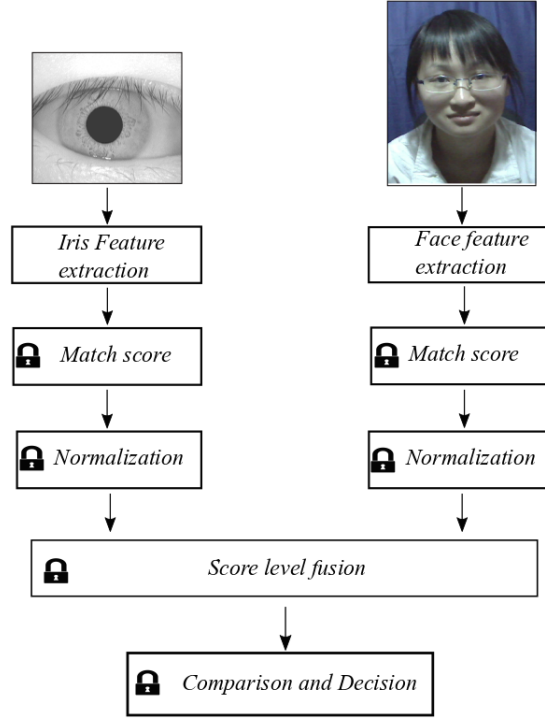


Figure 5.1: General scheme of our fusion protocol.

choose to use a min - max normalization method [7].

$$F_{\text{norm}} = \frac{\text{score} - \min_{\text{face}}}{\max_{\text{face}} - \min_{\text{face}}}, \quad (5.10)$$

where  $\min_{\text{face}}, \max_{\text{face}}$ , indicate the minimum and maximum values of the face range. Since  $\min_{\text{face}} = 0$ , Equation 5.10 can be written as

$$F_{\text{norm}} = \frac{\text{score}}{\max_{\text{face}}}. \quad (5.11)$$

**Linear combination.** Due to the characteristics of the SMPC system, to combine the matching scores, we choose a linear combination. If  $\alpha$  and  $\beta$  are proper weights in the interval  $[0, 1] \in \mathbb{R}$ ,  $\text{WHD} \in [0, 1]$  is

the Hamming distance resulting from iris match, SED the Euclidean distance between faces, and  $R$  the maximum value of the face recognition system, the fusion rule is

$$\alpha \cdot \text{WHD} + \beta \cdot \frac{\text{SED}}{R} \quad (5.12)$$

**Comparison and decision.** We compare the linear combination with the threshold. Multimodal recognition corresponds to check if the following inequality holds:

$$\alpha \cdot \text{WHD} + \beta \cdot \frac{\text{SED}}{R} < T. \quad (5.13)$$

The choice of the parameters  $\alpha, \beta, T$  determines the trade-off between equal error rate (EER) and computational complexity (Section 5.4). As in [137, 138] we choose  $\beta = 1 - \alpha$  for the tests in the plain domain. To avoid performing a division in WHD (Equation 5.4) and in normalization  $\left(\frac{\text{SED}}{R}\right)$ , instead of the Equation 5.13 we evaluate

$$\alpha \cdot \text{num} \cdot R + \beta \cdot \text{SED} \cdot \text{den} < T \cdot \text{den} \cdot R, \quad (5.14)$$

where  $\text{num}$  (Equation 5.5) and  $\text{den}$  (Equation 5.6) stand for the numerator and denominator of the Hamming distance respectively, while SED and  $R$  stand for squared Euclidean distance score and face maximum range, and  $T$  stands for threshold. The SPDZ framework does not allow the use of non-integer numbers, so  $\alpha, \beta$  and  $T$  are scaled and approximated to integers in the interval  $[0, 10]$ . We chose this interval because it is accurate enough to obtain the same results achieved in the plain domain, and the resulting bitlength is small enough to make it possible to represent  $\alpha \cdot \text{num} \cdot R + \beta \cdot \text{SED} \cdot \text{den}$  and  $T \cdot \text{den} \cdot R$  in  $\mathbb{F}_p$ .

### 5.3.1 Complexity of SEMBA

Equation 5.14 requires three multiplications and six transmissions that cannot be run in parallel and it needs  $\ell$  multiplications for the comparison. The linear combination needs  $\ell + 6$  multiplications and  $2\ell + 12$  transmissions,



plus the multiplications necessary to compute the Hamming and the squared Euclidean distances. The final complexity of the full multimodal protocol is  $3N + \ell + 6 + k$  multiplications (since squares and multiplications have similar complexity we consider all together as multiplications), while it needs only  $2\ell + 19$  transmissions (Table 2.3).

Table 5.2: Complexity Summary. We underline that transmission number depends only on  $p$ 's bitlength  $\ell$ .  $\alpha, \beta, t$  instead do not affect complexity but only accuracy.

	Multiplications	Squares	Transmissions
Iris	$3N + \ell + 1$	0	$2\ell + 7$
Face	$\ell$	$k$	$2\ell + 1$
Multimodal	$3N + \ell + 6$	$k$	$2\ell + 19$

### 5.3.2 Security

All the protocols involve a client (the biometric owner) and a server that authenticates the identity of the client. On one hand  $\mathcal{C}$  does not want to reveal his biometric templates, on the other hand  $\mathcal{S}$  does not want to dislocate its records. Both client and servers can be malicious but not at the same time. We underline that if both parties act maliciously, the protocol abort and they obtain no real information about the other. We also assume that the parties are connected through a secure channel providing security against eavesdroppers and any third party that can compromise the transmission.

The whole protocol is developed by relying on the SPDZ framework. According to [36], the SPDZ protocol involving  $n$  parties is secure up to  $n - 1$  malicious adversary. Therefore our protocol is secure in the UC model if at least one of the two party is honest. The offline phase does not depend on the functionality evaluated and its security demonstration against active adversaries in the UC model is provided in [36]. The following theorem provide the security demonstration of the protocol.

**Theorem 1.** *The online SPDZ implementation of SEMBA is computationally secure against any static adversary corrupting at most 1 party if  $p$  is exponential in the security parameter.*

*Proof.* The proof follows the security demonstration of the online SPDZ protocol in [36]. We rely on the simulator  $\mathcal{S}_{\text{ONLINE}}$  defined in [36] to work on top of the ideal multi-biometric authentication functionality  $\mathcal{F}_{\text{ONLINE}}$ . In this way the adversary cannot distinguish among the simulator using the real function  $\mathcal{F}_{\text{ONLINE}}$  and the real SPDZ-based implementation. Since input values broadcasted by both the simulator and honest players are uniform, it is not possible to distinguish among them. The only interaction with the player is performed during multiplication, squaring, and comparison where for both honest party and simulator partial opening reveals uniform values. Also MACs have similar distribution in both the protocol and the simulation. If the protocol does not abort due to a cheat detection, both the real and the simulated runs output the decision bit. In the simulation the decision bit is obtained by a correct evaluation of the multi-biometric function on the inputs provided by the player. In real world SPDZ-based implementation (i.e. not in the simulator) the adversary can cheat in the MACCheck (Algorithm 2.1) with probability  $2/p$ . Hence the probability that the adversary can distinguish the simulated environment from the real one is negligible if  $p$  is exponential. If the protocol does not abort, the adversary can observe only its inputs, the shared inputs received by the other party and the final result. The adversary is not able to obtain the inputs of the honest player, he should be able to solve the inequality in (Equation 5.14), which has  $N_i + N_f$  unknown variables for the server and  $N_i + N_f + 3$  for the client, where  $N_i$  and  $N_f$  are the number of features used to represent iris and face respectively.  $\square$

## 5.4 Preliminary tests in plain

In this section, we present the results of the tests performed on plain data. Those results are used to choose the best parameters to build an efficient protocol working in the encrypted domain. Tests have been carried out on the ‘‘CASIA-IrisV1’’ database for irises and ‘‘CASIA-FaceV5 part 1’’ database for faces, both collected by the Chinese Academy of Sciences’ Institute of Automation (CASIA) [129, 144].

The CASIA-IrisV1 database for irises [144] contains 756 grey-scale eye images with 108 unique irises (or classes) and 7 images for each of them.

Only the subset of well segmented images can be used for tests [103]. During the segmentation step, the iris region is isolated from the rest of the eye, in a digital image. The iris region can be approximated by two circles, one between iris and sclera and one for the iris pupil boundary. Segmentation is a crucial part of iriscodes template's extraction, since the data that is falsely represented as iris pattern will corrupt the biometrics template and it will alterate the match score [103]. The resulting database contains 625 eye images.

The CASIA-FaceV5 Databases for faces part 1 [129] contains 500 face images of 100 subjects. The face images are captured using Logitech USB camera in one session. All face images are 16 bit color BMP files and the image resolution is  $640 \times 480$  pixels.

We implemented and tested the SPDZ-based iris and multi-biometric protocols on a desktop equipped with 8GB RAM processor Intel Core i3 CPU 550 @ 3.20 GHz Quad-Core running Ubuntu 14.04 LTS (64 bit) operating system. We developed the test using C++ language with GMP free library for arbitrary precision arithmetic, operating on signed integers, rational numbers, and floating-point numbers.

To implement the SPDZ protocol we chose a prime number  $p$  of 46 bits which is big enough to allow all modular operations and comparisons required by the protocol. Server and client run on the same computer, and we used a socket to simulate the transmission channel.

We performed tests on plain data, running the authentication protocol on each single biometric and then fusing eigenfaces and iriscodes.

#### 5.4.1 Iris

As we saw before (Chapter 4), iris can be represented with a bit vector of length  $N$ . The number  $N$  of features depends on the radial  $r$  and angular  $\theta$  resolution used in the extraction process (Section 3.2) and it's equal to  $2 \cdot r \cdot \theta$  [103]. For testing the iris authentication protocol, we have chosen a radial resolution  $r$  ranging from 4 to 20 and an angular resolution  $\theta$  between 100 and 200 (Table 5.3). We tested the protocol on 625 eye images and each one has been compared with all the others. To extract the features, we used the Matlab code provided by L. Masek (iris recognition source code [106]), WHD and comparison has been also been implemented in Matlab. The best accuracy

is achieved by letting the angular resolution be equal to 160 angles and radial resolution equal to 20, corresponding to an iris feature vector of length 6400 (as it can be seen from Table 5.3). To better simulate the computation in the encrypted domain, we do not perform shifts during tests.

Table 5.3: Iriscode EER (%) without shifting, as a function of different values of  $r$  and  $\theta$ .

$r$	Angular Resolution $\theta$					
	100	120	140	160	180	200
4	8.19	6.43	4.37	3.34	3.31	3.10
6	6.88	5.05	3.01	2.45	2.71	3.01
8	6.13	4.42	2.69	2.19	2.58	4.36
10	6.31	4.03	2.64	2.44	2.54	3.92
12	5.96	4.10	2.56	2.14	2.59	3.71
14	5.71	3.85	2.54	2.17	2.58	3.27
16	5.49	3.79	2.32	2.13	2.51	3.31
18	5.71	3.61	2.46	2.31	2.41	3.18
20	5.77	3.74	2.20	2.08	2.41	3.13

### 5.4.2 Face

We have implemented the eigenface protocol by using the Open Source Computer Video (openCV) library<sup>1</sup> and Matlab. Face images are  $640 \times 480$  pixel and we transformed them in 256 grey level images. The protocol has been tested on 500 images. We used the algorithm provided in the openCV library to build  $k$  eigenfaces with  $k = 1 \dots 10$ . Each image is thus represented by a projection vector with  $k$  16-bit elements. Values have been rounded to the closest integers. The squared Euclidean distance has been calculated by using Matlab. We observed that the use of more than 5 projections does not provide any significant improvement (see Table 5.4).

<sup>1</sup><http://opencv.org/about.html>

Table 5.4: Eigenface EER (%) values, as a function of the number of projections.

$k$	EER (%)	$k$	EER (%)
1	28.77	6	17.01
2	17.37	7	16.19
3	16.62	8	16.51
4	16.59	9	16.38
5	16.08	10	16.09

Table 5.5: EER of the multimodal biometric authentication protocol. The first column show iris's number of feature (N), while, the second one represents Iris authentication system's EER (%). All the others columns contain the EER's obtained by fusing an iris template of length  $N$  and a face template with  $k \in \{1 \dots 10\}$  eigenfaces. Each EER represents the minimum EER among all those obtained by changing  $\alpha$  and  $\beta$ , therefore to each EER corresponds to a different choice of the optimal parameters. We highlighted in bold the configurations that we have selected for the tests under encryption.

Iris		Number of Eigenfaces ( $k$ )									
N	EER	1	2	3	4	5	6	7	8	9	10
6400	2.08	1.17	<b>1.15</b>	1.02	1.25	1.25	1.24	1.31	1.37	1.4	1.41
5760	2.51	1.26	<b>0.98</b>	1.01	1.22	1.38	1.36	1.43	1.47	1.49	1.50
5600	2.20	1.20	1.08	1.19	1.18	1.34	1.28	1.36	1.38	1.39	1.40
4800	3.74	1.90	1.97	1.65	1.98	2.04	2.15	2.11	2.17	2.2	2.21
3840	2.14	1.84	1.52	1.50	1.45	1.63	1.74	1.76	1.77	1.78	1.78
3600	2.54	1.36	1.23	<b>0.97</b>	1.65	1.82	1.99	2.07	2.15	2.17	2.19
3360	2.56	1.51	1.32	1.38	1.31	1.56	1.61	1.58	1.63	1.67	1.69
2560	2.19	1.50	1.24	1.19	1.15	1.39	1.49	1.59	1.6	1.61	1.62
2400	3.01	2.01	1.83	1.89	2.13	2.37	2.56	2.66	2.7	2.72	2.73
2160	2.71	1.92	1.74	1.82	1.98	2.04	2.09	2.07	2.13	2.16	2.18
1920	2.45	1.47	1.42	1.43	1.57	1.66	1.95	1.92	1.95	1.96	1.97
1600	3.10	<b>2.01</b>	<b>1.87</b>	1.85	2.41	2.37	2.56	2.67	2.69	2.71	2.71
1280	3.34	2.29	1.89	2.22	2.26	2.51	2.80	2.80	2.92	2.98	3.01

Table 5.6: EER (%) dependance from  $\alpha$  variation in plain. We chose the  $\alpha$ 's value corresponding to the minimum EER, highlighted in bold. We recall that  $\beta = 1 - \alpha$ .

$\alpha$	Iris (N) and Face (k)				
	$N = 1600$ $k = 1$	$N = 1600$ $k = 2$	$N = 3600$ $k = 3$	$N = 5760$ $k = 2$	$N = 6400$ $k = 2$
0.90	2.52	3.36	2.45	1.52	2.29
0.85	2.37	2.33	2.58	2.02	1.47
0.80	<b>2.01</b>	2.13	1.68	<b>0.98</b>	<b>1.15</b>
0.75	2.57	3.69	3.51	2.02	2.51
0.70	3.00	2.11	2.15	1.92	1.24
0.65	2.33	<b>1.87</b>	1.95	1.15	1.17
0.60	3.81	3.55	2.78	3.13	2.27
0.55	4.25	2.02	<b>0.97</b>	1.63	1.15
0.50	3.85	3.37	4.75	2.57	3.18

### 5.4.3 Multimodal system

In order to evaluate the efficiency of the fusion protocol in the plain domain, the fusion of the outcomes of face and iris sub-protocols is implemented using Matlab. From Table 5.3, we have chosen some relevant iris configurations, based on the achieved EER or number of features. First of all, to better compare the plain tests with the best iris result, we chose  $r = 20$  and  $\theta = 160$  resulting in  $N = 6400$ , then for each  $\theta$ , we looked for any accuracy lower than 4%, and finally we chose those configurations with EER similar to the previous ones but less features. Moreover, we varied  $\alpha$  in the interval  $[0, 1]$  (Table 5.6) and the number of eigenfaces  $k$  from 1 to 10. Table 5.5 summarizes our results, showing the EER for each  $N$  and  $k$ . The results in Table 5.5 represent the minimum EER when we vary  $\alpha$  and  $\beta$ , therefore each EER corresponds to a different choice of the optimal parameters. In fact, as it can be seen from Table 5.6, reporting the data only for the chosen configurations, the value of  $\alpha$  (and consequently of  $\beta$ ) influences the accuracy of the multimodal system. We summarize the used set of parameters in Table 5.7.

The same accuracy of the 6400 stand alone iris protocol (2.08%) can be reached with many different multi-biometric configurations, as can be seen

from Table 5.5. For example by using  $N = 3600$  and  $k = 7$  EER is equal to 2.08% or by using only 1600 iris features and  $k = 1$  (EER=2.01%). For the tests in the encrypted domain, we chose the last one since it has lower bandwidth and computational complexity (see Table 4.3 and Table 5.8). We can also notice that keeping  $N = 1600$ , but using 2 features for face representation, we can lower both accuracy and complexity. Generally, by using two eigenfaces, the best possible accuracy is obtained with 5760 iris features, however the same performance is obtained also by the combination of 3600 iris feature and three face features. The results of the tested configurations in the encrypted domain are summarized in Table 5.7. We can also observe that the number of iris features influences the execution times. A bigger  $N$  corresponds to a higher computational time while the use of more iris features does not always result in a better accuracy after fusion. Finding a good trade-off is not easy. In the sequel we chose the parameters based on experimental results.

## 5.5 Results of tests on SEMBA

We evaluated the computational complexity of our implementation of SEMBA (Section 5.3), by using the parameters chosen in the previous section (see Table 5.3 and Table 5.7). Since  $\alpha, \beta$  and  $t$  do not affect complexity but impact on the accuracy, we chose the values associated with the lowest EER, Table 5.6 shows the variation of EER when  $N$  and  $k$ , are fixed and  $\alpha$  is varied. We recall that  $\beta = 1 - \alpha$  (Section 5.3). The number of multiplications affects heavily the execution times. As we said in Section 5.1, when possible, we performed a single transmission, by packing data. To calculate the execution time, reported in Table 5.9, we used the *clock* function, measuring the CPU time of the process. To convert the CPU time in seconds, we used the formula  $\frac{n, \text{ clock}}{\text{clock par sec}}$ . The symbol *n. clock* indicates the processor time consumed by the protocol's implementation, the value returned is expressed in clock ticks, which are units of time of a constant but system-specific length. The quantity *clock par sec*, instead, indicates the number of clock ticks per second. Therefore the previous formula returns the number of second required by the computation. In the protocol only the the bitlength  $\ell$  of the prime number

Table 5.7: Equal Error Rate of iris and multimodal biometric authentication protocols for different settings;  $\alpha$ ,  $t$  respectively stand for fusion coefficient and threshold.

Iris N	EER		Face $k$	Fusion parameters	
	Iris (%)	Fusion (%)		$\alpha$	$t$
1600	3.10	2.01	1	0.80	0.35
1600	3.10	1.87	2	0.65	0.25
3600	2.54	0.97	3	0.55	0.25
5760	2.51	0.98	2	0.80	0.35
6400	2.08	1.15	2	0.80	0.35

$p$  influences the number of transmission rounds and the feature configuration has no impact on it, as it can be seen from Table 5.2. On the contrary, the amount of data transmitted by each party also depends on the number of features used in the protocol. In fact, the iris authentication protocol has a bandwidth of  $(6N + 2\ell + 2) \cdot \ell$  bits, while the multimodal protocol bandwidth is  $\ell \cdot (6N + k + 2\ell + 12)$  bits. The complexity of the face-based authentication protocol is negligible with respect to the iris protocol ones. For this reason, the overhead introduced by the multimodal biometric authentication is of few bytes, as shown in Table 5.8. Therefore, the communication complexity remains almost constant switching from the iris to the multimodal protocol.

Our goal is to reduce complexity in order to reach the same efficiency of the iris-based protocol. These results are a first mark in this direction, because the overhead introduced by the multi-biometric protocol is low. Moreover, our analysis shows that SEMBA can also be used to lower the EER without a significant loss in terms of complexity. In the following we analyze both cases.

**Improved efficiency** The running time of the stand alone iris authentication protocol ranges from 0.03s for 1600 bits, up to 0.12s for a 6400 bit-long template in the malicious setting (see Table 5.9), while Luo et al. protocol [24]



Table 5.8: Communication complexity for the iris and multimodal protocols. It is important to notice that adding few eigenfaces increases the bandwidth by a few bytes only.

Iris $N$	bandwidth (KB)			Face $k$
	iris	multimodal	overhead	
1600	53.24 kb	53.30 kb	0.06 kb	1
3600	119.16 kb	119.23 kb	0.07 kb	3
5760	190.35 kb	190.42 kb	0.07 kb	2
6400	211.44 kb	211.51 kb	0.07 kb	2

with masks needs 2.5s for 9600 bits and 0.56s for 2048 bits in the semi-honest setting. Moreover from Table 5.7 and Table 5.9, it is evident that the multi-biometric authentication protocol can provide the same accuracy of the best stand alone iris protocol, but with lower execution time and computational complexity. As a matter of fact, the best EER for the standalone iris protocol is 2.08% for 6400 features corresponding to 19246 multiplications (Table 4.3) in 0.12s, while in SEMBA for 1600 iris features and 1 eigenface feature, we need only 8744 multiplications (Table 4.3, where we consider squares as multiplications) to obtain an EER equal to 2.01% in about 0.03 seconds. On the contrary, the number of required transmissions increases from  $2\ell + 7$  to  $2\ell + 19$  (Table 4.3), but it depends only on the bit length of  $p$ .

Table 5.9: Iris protocol time in SPDZ system.

Iris $N$	CPU time		Face $k$
	Iris (s)	multimodal (s)	
1600	0.029s	0.030	1
1600		0.030	2
3600	0.048	0.049	3
5760	0.11s	0.109	2
6400	0.12s	0.120	2

**Improved accuracy** Two biometrics instead of one can be used to achieve a higher accuracy, at the cost of a slight increase of complexity with respect to the iricode protocol. In fact, complexity depends heavily on the number of iris features, as shown in Table 4.3, however it is possible to decrease the EER rate by adding two eigenfaces, while the number of multiplications increases only from  $3N + \ell + 1$  to  $3N + \ell + 6 + k = 3N + \ell + 8$  (as usual we consider squaring to be equivalent to a multiplication). More generally, when we use the multimodal authentication, the total CPU time slightly increases with respect to the unimodal iris protocol, but the EER always decreases by adding one more eigenface ( $k = 2$ ) to the 1600 iris feature configuration considered above, we can have a better EER (1.87%) with the same time complexity (0.03 seconds). For the case of 5760 bit long iris template the EER passes from 2.1% for the unimodal authentication to 0.98% of the bimodal case with  $k = 2$  (Table 5.7). Finally, keeping 0.98% as target accuracy, we highlight that we can reduce  $N$  to 3600 at the cost of an additional feature in the face representation ( $k = 3$ ). In this case, computational complexity goes from 36926 to 19596 multiplications and time complexity decreases from 0.109s to 0.05s (Table 5.8 and Table 5.9).



The goal of this thesis has been to develop tools for the protection of biometric templates from identity theft and other cybercrimes. We started from the consideration that most of the works in this research area are secure in the semi-honest model, i.e. against a passive adversary, a listener that exploits every chance to learn private information without tampering with the protocol. We proposed two solutions, the first one based on a somewhat homomorphic encryption scheme and the second one on SPDZ, a new secure multiparty computation tool.

With regard to the first solution, we have implemented and tested a SHE scheme [34] that works on integer values. Then we used it to build two privacy-preserving biometric matching protocols. The resulting complexity is lower than that of a SHE solution working directly on bits. The prior knowledge of the number of multiplications required by the protocol allows us to use a SHE scheme instead of a FHE, avoiding in this way expensive squash and bootstrap operations. Even if the protocol is not as efficient as protocols based on other STPC techniques such as garbled circuits and homomorphic encryption, our solution has the advantage of moving all the computation to the server side, without the necessity of any interaction with the client, hence making the protocol suitable for low power client devices. We have observed, that the operation runtime and the bitsize of ciphertexts and keys are more affected by the security parameter than the magnitude of the base  $b$ . In fact, changing  $b$  from 2 to  $2^{150}$  has no consequences in time complexity. This observation allows us to choose the base that better suits our computational needs. We have implemented a small upgrade of the cryptographic system by finding a solution to the problem of encrypting negative numbers. Concerning the iris and fingerprint identification protocols, we have devised a solution to keep the number of multiplications and therefore the complexity as low

as possible, especially in the identification protocol. Moreover, we formulate blinding methods to protect output results: permutation for identification and a multiplicative blinding in the authentication protocol. We have tested the identification and authentication protocols with different security parameters and observed that a match takes from some seconds to several minutes, especially in the iris case. Due to the time required for a single match our iris recognition protocol is more appropriate for an offline search while the fingerprint one can be used also online.

In Chapter 5 we presented SEMBA a secure multimodal authentication system based on the SMPC approach SPDZ [35, 36]. The novelty of this approach is to use a multimodal system in order to find a better trade off between accuracy and efficiency, in addition using SPDZ, also the final decision is taken in the encrypted domain by the client that, interacting with the server, compares the final output with the threshold. In this way we overcome security weakness of the previous SHE implementations where the client makes the last decision after decrypting the outputs. We have shown how it is possible to improve the efficiency of the secure recognition process in terms of computational and time complexity, without any loss of accuracy, with respect to an unimodal iris recognition system. We have also seen that it is possible to improve the accuracy with a minor increase of complexity. As parallel contributions, we adapted the iris and face authentication protocols to work in the SPDZ setting and we reduced the complexity by resorting to packed transmission of the encrypted data involved in the secure multiplication protocol.

## 6.1 Highlights for future research

Secure biometrics recognition systems and Signal Processing in the Encrypted Domain are wide and young research areas, and many open issues could be addressed in the near future. Those most directly related to the research covered in this thesis are briefly highlighted in the following. First of all, new and more efficient implementations of the underlying SPED tools, and building blocks can be studied to improve the protocol efficiency. In both the protocols, multiplications between encrypted data slow down the computa-

tion; solutions can be found to speed up the computation. Consequently the match step can be improved.

In the SHE implementation, the length of the template vectors can be reduced and tested, maybe using a different feature extractor or a different biometrics. A cryptographic line of research can investigate a new way to prepare and store the public key in order to reduce the storage requirement and therefore allowing to use a bigger security parameter. Moreover the security of the identification protocol can be improved by applying the blinding devised for authentication before permutating the output vector.

With regard to SEMBA, in the future we can extend our approach to handle even more biometric traits, like fingerprints, behavioral biometrics etc. looking for a simple template to represent and a match score easy to implement but that at the same time guarantees high accuracy with low computational complexity. Maintaining the same biometrics, we can test different algorithms and alternative fusion rules. Obviously, in this case we must also consider if the complexity of each new proposal is a fair price to pay. For the multimodal system, we could also consider fusion at different levels. It could be interesting to study fusion at the feature level, and looking for a new merged template, to be used efficiently with SPED tools, like SCiFI [22] for face.

In all the implementations presented so far, floating point numbers are always rounded to integers and a difficult task consists in finding a solution that allows MPC protocols to work directly with those numbers, avoiding approximation errors.

Nowadays, cell phones use a biometric key not only to unlock the phone but also for payments. The biometrics are stored somewhere in the phone and a criminal, could retrieve and use them to access the bank account and exploit the weakness of the new payment methods. In this scenario, it could be interesting to make the system portable to be used on a mobile phone in order to protect the stored biometrics and the privacy of the owner. It is not difficult to foresee a future in which biometrics are used to identify people, as we currently do with ID cards, and to replace physical tokens, as credit cards, to access the bank account. To make this future possible for every aspect of our live, it is necessary to implement efficient and secure recognition

systems in the malicious model, maybe using one or more biometrics. It is also important to make the framework portable, and reliable enough to be used in everyday application such as home banking or to confirm a digital identity like the Italian SPID (*Sistema Pubblico di Identità Digitale*).

As a final and general note, not only related to biometrics but to s.p.e.d. tools in general, we point out that the need for privacy is promoting the use of s.p.e.d. tools in real world applications. In the last few years some attempts have been made in that direction. For example, N. Smart, one of the author of SPDZ is co-founder of a company called Unbound Tech<sup>1</sup>, that offers services for secure computation and aims to prevent the problem of server breach. Moreover Smart and his group at KU Leuven university, developed and distributed an open source version of all their SMPC systems for commercial and research purposes<sup>2</sup>. Similarly, the non-profit company Alexandra Institute implemented the FRESCO project [145], which is a Framework for Efficient and Secure COmputation, released under open source MIT license. The project aims to ease the development of both prototype applications based on secure computation and new techniques (called *protocol suits* for secure multiparty computation) by providing ready to use frameworks. Their goal is to create value, growth and welfare in society, by helping public and private organizations to develop innovative products and services based on IT research.

Initiatives such the ones mentioned above contribute to spread SMPC and *s.p.e.d.* techniques in everyday applications. To continue in this direction it will be very important to make secure recognition systems efficient and portable enough to be used on cellphones or other small devices.

---

<sup>1</sup><https://www.unboundtech.com>

<sup>2</sup><https://homes.esat.kuleuven.be/nsmart/SCALE>

---

## Bibliography

- [1] M. Blanton and P. Gasti, “Secure and efficient protocols for iris and fingerprint identification,” in *European Symposium on Research in Computer Security*. Springer, 2011, pp. 190–209.
- [2] Z. Erkin, M. Franz, J. Guajardo, S. Katzenbeisser, I. Lagendijk, and T. Toft, “Privacy-preserving face recognition,” in *Privacy Enhancing Technologies*. Springer, 2009, pp. 235–253.
- [3] D. Evans, Y. Huang, J. Katz, and L. Malka, “Efficient privacy-preserving biometric identification,” in *Proceedings of the 17th Conference on Network and Distributed System Security Symposium, NDSS*, 2011.
- [4] L. O’Gorman, “Comparing passwords, tokens, and biometrics for user authentication,” *Proceedings of the IEEE*, vol. 91, no. 12, pp. 2021–2040, 2003.
- [5] A. K. Jain, A. Ross, and S. Prabhakar, “An introduction to biometric recognition,” *IEEE Transactions on circuits and systems for video technology*, vol. 14, no. 1, pp. 4–20, 2004.
- [6] A. Ross, K. Nandakumar, and A. K. Jain, “Introduction to multibiometrics,” in *Handbook of Biometrics*. Springer, 2008, pp. 271–292.
- [7] A. A. Ross, K. Nandakumar, and A. K. Jain, *Handbook of multibiometrics*. Springer Science & Business Media, 2006, vol. 6.
- [8] S. M. Matyas Jr and J. Stapleton, “A biometric standard for information management and security,” *Computers & Security*, vol. 19, no. 5, pp. 428–441, 2000.
- [9] “Biometrics security and privacy protection,” *IEEE Signal Processing Magazine*, vol. 32, no. 5, Sept 2015.



- 
- [10] R. Legendijk, Z. Erkin, and M. Barni, “Encrypted signal processing for privacy protection: Conveying the utility of homomorphic encryption and multiparty computation,” *IEEE Signal Processing Magazine*, vol. 30, no. 1, pp. 82–105, 2013.
  - [11] M. Barni, G. Droandi, and R. Lazzeretti, “Privacy protection in biometric-based recognition systems: A marriage between cryptography and signal processing,” *IEEE Signal Processing Magazine*, vol. 32, no. 5, pp. 66–76, 2015.
  - [12] O. R. Michael, “How to exchange secrets by oblivious transfer,” Technical Report TR-81, Aiken Computation Laboratory, Harvard University, Tech. Rep., 1981.
  - [13] A. C. Yao, “How to generate and exchange secrets,” in *Proceedings of the 27th Annual IEEE Symposium on Foundations of Computer Science*, 1986, pp. 162–167.
  - [14] P. Paillier, “Public-key cryptosystems based on composite degree residuosity classes.” Springer, 1999, pp. 223–238.
  - [15] C. Gentry and S. Halevi, “Implementing gentry’s fully-homomorphic encryption scheme,” *Advances in Cryptology–EUROCRYPT 2011*, pp. 129–148, 2011.
  - [16] M. Kiraz and B. Schoenmakers, “A protocol issue for the malicious case of yao’s garbled circuit construction,” in *27th Symposium on Information Theory in the Benelux*, 2006, pp. 283–290.
  - [17] Y. Lindell and B. Pinkas, “An efficient protocol for secure two-party computation in the presence of malicious adversaries,” *Advances in Cryptology–EUROCRYPT 2007*, pp. 52–78, 2007.
  - [18] I. Haitner, “Semi-honest to malicious oblivious transfer—the black-box way,” in *Theory of Cryptography Conference*. Springer, 2008, pp. 412–426.
  - [19] P. Campisi, *Security and Privacy in Biometrics*. Springer, 2013.
  - [20] J. Bringer, H. Chabanne, and A. Patey, “Privacy-preserving biometric identification using secure multiparty computation: An overview and recent trends,” *IEEE Signal Processing Magazine*, vol. 30, no. 2, pp. 42–52, 2013.
  - [21] Y. Luo, S. C. Sen-ching, and S. Ye, “Anonymous biometric access control based on homomorphic encryption,” in *Multimedia and Expo, 2009. ICME 2009. IEEE International Conference on*. IEEE, 2009, pp. 1046–1049.
  - [22] M. Osadchy, B. Pinkas, A. Jarrous, and B. Moskovich, “Scifi—a system for secure face identification,” in *Security and Privacy (SP), 2010 IEEE Symposium on*. IEEE, 2010, pp. 239–254.

- 
- [23] M. Barni, T. Bianchi, D. Catalano, M. Di Raimondo, R. Donida Labati, P. Failla, D. Fiore, R. Lazzeretti, V. Piuri, A. Piva, and F. Scotti, "A privacy-compliant fingerprint recognition system based on homomorphic encryption and fingercode templates," in *4th IEEE International Conference on Biometrics: Theory Applications and Systems (BTAS)*. IEEE, 2010, pp. 1–7.
- [24] Y. Luo, S.-c. S. Cheung, T. Pignata, R. Lazzeretti, and M. Barni, "An efficient protocol for private iris-code matching by means of garbled circuits," in *Image Processing (ICIP), 2012 19th IEEE International Conference on*. IEEE, 2012, pp. 2653–2656.
- [25] J. Bringer, M. Favre, H. Chabanne, and A. Patey, "Faster secure computation for biometric identification using filtering," in *Biometrics (ICB), 2012 5th IAPR International Conference on*. IEEE, 2012, pp. 257–264.
- [26] A. Sadeghi, T. Schneider, and I. Wehrenberg, "Efficient privacy-preserving face recognition," in *Information, Security and Cryptology-ICISC 2009*. Springer, 2010, pp. 229–244.
- [27] M. Yasuda, T. Shimoyama, J. Kogure, K. Yokoyama, and T. Koshihara, "Packed homomorphic encryption based on ideal lattices and its application to biometrics," in *International Conference on Availability, Reliability, and Security*. Springer, 2013, pp. 55–74.
- [28] J. R. Troncoso-Pastoriza, D. Gonzalez-Jimenez, and F. Perez-Gonzalez, "Fully private noninteractive face verification," *Information Forensics and Security, IEEE Transactions on*, vol. 8, no. 7, pp. 1101–1114, July 2013.
- [29] A. Abidin, "On privacy-preserving biometric authentication." in *Inscrypt*, 2016, pp. 169–186.
- [30] M. A. Pathak and B. Raj, "Privacy-preserving speaker verification and identification using gaussian mixture models," *IEEE Transactions on Audio, Speech, and Language Processing*, vol. 21, no. 2, pp. 397–406, 2013.
- [31] A. K. Jain, K. Nandakumar, and A. Nagar, "Biometric template security," *EURASIP Journal on advances in signal processing*, vol. 2008, p. 113, 2008.
- [32] U. Uludag, S. Pankanti, S. Prabhakar, and A. K. Jain, "Biometric cryptosystems: issues and challenges," *Proceedings of the IEEE*, vol. 92, no. 6, pp. 948–960, 2004.
- [33] V. M. Patel, N. K. Ratha, and R. Chellappa, "Cancelable biometrics: A review," *IEEE Signal Processing Magazine*, vol. 32, no. 5, pp. 54–65, 2015.

- 
- [34] P. S. Pisa, M. Abdalla, and O. C. M. B. Duarte, “Somewhat homomorphic encryption scheme for arithmetic operations on large integers,” in *Global Information Infrastructure and Networking Symposium (GIIS), 2012*. IEEE, 2012, pp. 1–8.
- [35] I. Damgård, V. Pastro, N. Smart, and S. Zakarias, “Multiparty computation from somewhat homomorphic encryption,” in *Advances in Cryptology—CRYPTO 2012*. Springer, 2012, pp. 643–662.
- [36] I. Damgård, M. Keller, E. Larraia, V. Pastro, P. Scholl, and N. P. Smart, “Practical covertly secure mpc for dishonest majority—or: Breaking the spdz limits,” in *European Symposium on Research in Computer Security*. Springer, 2013, pp. 1–18.
- [37] G. Droandi and R. Lazzeretti, “She based non interactive privacy preserving biometric authentication protocols,” in *Intelligent Signal Processing (WISP), 2015 IEEE 9th International Symposium on*. IEEE, 2015, pp. 1–6.
- [38] G. Droandi, “Non-interactive privacy preserving protocol for biometric recognition based on somewhat homomorphic encryption,” in *ECCWS2015- Proceedings of the 14th European Conference on Cyber Warfare and Security 2015: ECCWS 2015*. Academic Conferences Limited, 2015, p. 355.
- [39] O. Goldreich, “Foundation of cryptography (in two volumes: Basic tools and basic applications),” 2001.
- [40] C. Hazay and Y. Lindell, *Efficient secure two-party protocols: Techniques and constructions*. Springer Science & Business Media, 2010.
- [41] D. Beaver, “Foundations of secure interactive computing.” in *Crypto*, vol. 91. Springer, 1991, pp. 377–391.
- [42] S. Goldwasser and L. Levin, “Fair computation of general functions in presence of immoral majority,” in *Conference on the Theory and Application of Cryptography*. Springer, 1990, pp. 77–93.
- [43] S. Micali and P. Rogaway, “Secure computation,” in *Annual International Cryptology Conference*. Springer, 1991, pp. 392–404.
- [44] R. Canetti, “Security and composition of multiparty cryptographic protocols,” *Journal of CRYPTOLOGY*, vol. 13, no. 1, pp. 143–202, 2000.
- [45] R. Ostrovsky and M. Yung, “How to withstand mobile virus attacks,” in *Proceedings of the tenth annual ACM symposium on Principles of distributed computing*. ACM, 1991, pp. 51–59.

- 
- [46] R. Canetti and A. Herzberg, “Maintaining security in the presence of transient faults,” in *Advances in Cryptology—CRYPTO’94*. Springer, 1994, pp. 425–438.
- [47] M. Naor and B. Pinkas, “Efficient oblivious transfer protocols,” in *Proceedings of the twelfth annual ACM-SIAM symposium on Discrete algorithms*. Society for Industrial and Applied Mathematics, 2001, pp. 448–457.
- [48] S. Goldwasser, S. Micali, and A. Wigderson, “How to play any mental game, or a completeness theorem for protocols with an honest majority,” in *Proc. of the Nineteenth Annual ACM STOC*, vol. 87, 1987, pp. 218–229.
- [49] T. Chou and C. Orlandi, “The simplest protocol for oblivious transfer,” in *International Conference on Cryptology and Information Security in Latin America*. Springer, 2015, pp. 40–58.
- [50] J. B. Nielsen, P. S. Nordholt, C. Orlandi, and S. S. Burra, “A new approach to practical active-secure two-party computation.” in *CRYPTO*, vol. 7417. Springer, 2012, pp. 681–700.
- [51] Y. Huang, D. Evans, and J. Katz, “Private set intersection: Are garbled circuits better than custom protocols?” in *NDSS*, 2012.
- [52] C. Dong, L. Chen, and Z. Wen, “When private set intersection meets big data: an efficient and scalable protocol,” in *Proceedings of the 2013 ACM SIGSAC conference on Computer & communications security*. ACM, 2013, pp. 789–800.
- [53] D. Beaver, “Correlated pseudorandomness and the complexity of private computations,” in *Proceedings of the twenty-eighth annual ACM symposium on Theory of computing*. ACM, 1996, pp. 479–488.
- [54] G. Asharov, Y. Lindell, T. Schneider, and M. Zohner, “More efficient oblivious transfer and extensions for faster secure computation,” in *Proceedings of the 2013 ACM SIGSAC conference on Computer & communications security*. ACM, 2013, pp. 535–548.
- [55] Y. Ishai, J. Kilian, K. Nissim, and E. Petrank, “Extending oblivious transfers efficiently.” in *Crypto*, vol. 2729. Springer, 2003, pp. 145–161.
- [56] G. Asharov, Y. Lindell, T. Schneider, and M. Zohner, “More efficient oblivious transfer extensions with security for malicious adversaries.” *EUROCRYPT (1)*, vol. 9056, pp. 673–701, 2015.
- [57] G. Asharov, Y. Lindell, T. Schneider, and M. Zohner, “More efficient oblivious transfer extensions,” *Journal of Cryptology*, vol. 30, no. 3, pp. 805–858, 2017.

- 
- [58] D. Beaver, “Precomputing oblivious transfer,” *Advances in Cryptology?CRYPTO?95*, pp. 97–109, 1995.
- [59] B. Pinkas, T. Schneider, N. P. Smart, and S. C. Williams, “Secure two-party computation is practical.” in *Asiacrypt*, vol. 9. Springer, 2009, pp. 250–267.
- [60] R. Lazzeretti and M. Barni, “Private computing with garbled circuits [applications corner],” *IEEE Signal Processing Magazine*, vol. 30, no. 2, pp. 123–127, 2013.
- [61] J. B. Nielsen and C. Orlandi, “Lego for two-party secure computation.” in *TCC*, vol. 5444. Springer, 2009, pp. 368–386.
- [62] S. Jarecki and V. Shmatikov, “Efficient two-party secure computation on committed inputs,” in *Annual International Conference on the Theory and Applications of Cryptographic Techniques*. Springer, 2007, pp. 97–114.
- [63] Y. Ishai, M. Prabhakaran, and A. Sahai, “Secure arithmetic computation with no honest majority,” in *Theory of Cryptography Conference*. Springer, 2009, pp. 294–314.
- [64] Y. Ishai, M. Prabhakaran, and A. Sahai, “Founding cryptography on oblivious transfer—efficiently,” *Lecture Notes in Computer Science*, vol. 5157, pp. 572–592, 2008.
- [65] Y. Lindell and B. Pinkas, “Secure two-party computation via cut-and-choose oblivious transfer,” *Journal of cryptology*, vol. 25, no. 4, pp. 680–722, 2012.
- [66] Y. Lindell, “Fast cut-and-choose-based protocols for malicious and covert adversaries,” *Journal of Cryptology*, vol. 29, no. 2, pp. 456–490, 2016.
- [67] S. Goldwasser, S. Micali, and C. Rackoff, “The knowledge complexity of interactive proof systems,” *SIAM Journal on computing*, vol. 18, no. 1, pp. 186–208, 1989.
- [68] V. Kolesnikov and T. Schneider, “Improved garbled circuit: Free xor gates and applications,” *Automata, Languages and Programming*, pp. 486–498, 2008.
- [69] G. R. Blakley *et al.*, “Safeguarding cryptographic keys,” in *Proceedings of the national computer conference*, vol. 48, 1979, pp. 313–317.
- [70] A. Shamir, “How to share a secret,” *Communications of the ACM*, vol. 22, no. 11, pp. 612–613, 1979.
- [71] M. Ito, A. Saito, and T. Nishizeki, “Secret sharing scheme realizing general access structure,” *Electronics and Communications in Japan (Part III: Fundamental Electronic Science)*, vol. 72, no. 9, pp. 56–64, 1989.

- 
- [72] A. Beigel, “Secret-sharing schemes: A survey.” *IWCC*, vol. 6639, pp. 11–46, 2011.
- [73] C. Fontaine and F. Galand, “A survey of homomorphic encryption for nonspecialists,” *EURASIP Journal on Information Security*, vol. 2007, 2007.
- [74] C. Gentry, *A fully homomorphic encryption scheme*. Stanford University, 2009.
- [75] Z. Brakerski, C. Gentry, and V. Vaikuntanathan, “(leveled) fully homomorphic encryption without bootstrapping,” in *Proceedings of the 3rd Innovations in Theoretical Computer Science Conference*. ACM, 2012, pp. 309–325.
- [76] M. Van Dijk, C. Gentry, S. Halevi, and V. Vaikuntanathan, “Fully homomorphic encryption over the integers,” *Advances in Cryptology–EUROCRYPT 2010*, pp. 24–43, 2010.
- [77] Z. Brakerski and V. Vaikuntanathan, “Fully homomorphic encryption from ring-lwe and security for key dependent messages,” in *Advances in Cryptology–CRYPTO 2011*. Springer, 2011, pp. 505–524.
- [78] J.-S. Coron, A. Mandal, D. Naccache, and M. Tibouchi, “Fully homomorphic encryption over the integers with shorter public keys,” in *Advances in Cryptology–CRYPTO 2011*. Springer, 2011, pp. 487–504.
- [79] J.-S. Coron, D. Naccache, and M. Tibouchi, “Public key compression and modulus switching for fully homomorphic encryption over the integers,” in *Advances in Cryptology–EUROCRYPT 2012*. Springer, 2012, pp. 446–464.
- [80] J. H. Cheon, J.-S. Coron, J. Kim, M. S. Lee, T. Lepoint, M. Tibouchi, and A. Yun, “Batch fully homomorphic encryption over the integers,” in *Advances in Cryptology–EUROCRYPT 2013*. Springer, 2013, pp. 315–335.
- [81] C. Gentry, “Fully homomorphic encryption using ideal lattices,” in *Proceedings of the 41st annual ACM symposium on Theory of computing*. ACM, 2009, pp. 169–178.
- [82] N. P. Smart and F. Vercauteren, “Fully homomorphic encryption with relatively small key and ciphertext sizes.” in *Public Key Cryptography*, vol. 6056. Springer, 2010, pp. 420–443.
- [83] D. Stehlé and R. Steinfeld, “Faster fully homomorphic encryption,” *Advances in Cryptology-ASIACRYPT 2010*, pp. 377–394, 2010.
- [84] J. Loftus, A. May, N. P. Smart, and F. Vercauteren, “On cca-secure somewhat homomorphic encryption.” in *Selected Areas in Cryptography*, vol. 7118. Springer, 2011, pp. 55–72.

- [85] Z. Brakerski and V. Vaikuntanathan, “Efficient fully homomorphic encryption from (standard) lwe,” in *Foundations of Computer Science (FOCS), 2011 IEEE 52nd Annual Symposium on*. IEEE, 2011, pp. 97–106.
- [86] Z. Brakerski, “Fully homomorphic encryption without modulus switching from classical gapsvp,” in *Advances in Cryptology–CRYPTO 2012*. Springer, 2012, pp. 868–886.
- [87] Z. Brakerski and V. Vaikuntanathan, “Lattice-based fhe as secure as pke,” in *Proceedings of the 5th conference on Innovations in theoretical computer science*. ACM, 2014, pp. 1–12.
- [88] C. Gentry, A. Sahai, and B. Waters, “Homomorphic encryption from learning with errors: Conceptually-simpler, asymptotically-faster, attribute-based,” in *Advances in Cryptology–CRYPTO 2013*. Springer, 2013, pp. 75–92.
- [89] A. López-Alt, E. Tromer, and V. Vaikuntanathan, “On-the-fly multiparty computation on the cloud via multikey fully homomorphic encryption,” in *Proceedings of the 44th symposium on Theory of Computing*. ACM, 2012, pp. 1219–1234.
- [90] J. W. Bos, K. E. Lauter, J. Loftus, and M. Naehrig, “Improved security for a ring-based fully homomorphic encryption scheme.” in *IMA Int. Conf.* Springer, 2013, pp. 45–64.
- [91] D. Boneh, C. Gentry, S. Halevi, F. Wang, and D. J. Wu, “Private database queries using somewhat homomorphic encryption,” in *International Conference on Applied Cryptography and Network Security*. Springer, 2013, pp. 102–118.
- [92] M. Yasuda, T. Shimoyama, J. Kogure, K. Yokoyama, and T. Koshihara, “Secure pattern matching using somewhat homomorphic encryption,” in *Proceedings of the 2013 ACM workshop on Cloud computing security workshop*. ACM, 2013, pp. 65–76.
- [93] N. Howgrave-Graham, “Approximate integer common divisors,” in *Cryptography and Lattices*. Springer, 2001, pp. 51–66.
- [94] Z. Brakerski, C. Gentry, and V. Vaikuntanathan, “(leveled) fully homomorphic encryption without bootstrapping,” *ACM Transactions on Computation Theory (TOCT)*, vol. 6, no. 3, p. 13, 2014.
- [95] C. Gentry, S. Halevi, and N. P. Smart, “Fully homomorphic encryption with polylog overhead,” in *Annual International Conference on the Theory and Applications of Cryptographic Techniques*. Springer, 2012, pp. 465–482.

- 
- [96] C. Gentry, S. Halevi, and N. P. Smart, "Homomorphic evaluation of the aes circuit," in *Advances in Cryptology-CRYPTO 2012*. Springer, 2012, pp. 850–867.
- [97] T. Veugen, R. de Haan, R. Cramer, and F. Muller, "A framework for secure computations with two non-colluding servers and multiple clients, applied to recommendations," *IEEE Transactions on Information Forensics and Security*, vol. 10, no. 3, pp. 445–457, 2015.
- [98] T. Nishide and K. Ohta, "Multiparty computation for interval, equality, and comparison without bit-decomposition protocol," in *International Workshop on Public Key Cryptography*. Springer, 2007, pp. 343–360.
- [99] V. Kolesnikov, A.-R. Sadeghi, and T. Schneider, "How to combine homomorphic encryption and garbled circuits," *Signal Processing in the Encrypted Domain*, vol. 100, p. 2009, 2009.
- [100] A. K. Jain, S. Pankanti, S. Prabhakar, L. Hong, and A. Ross, "Biometrics: a grand challenge," in *Pattern Recognition, 2004. ICPR 2004. Proceedings of the 17th International Conference on*, vol. 2. IEEE, 2004, pp. 935–942.
- [101] J. Galbally, R. Cappelli, A. Lumini, G. Gonzalez-de Rivera, D. Maltoni, J. Fierrez, J. Ortega-Garcia, and D. Maio, "An evaluation of direct attacks using fake fingers generated from iso templates," *Pattern Recognition Letters*, vol. 31, no. 8, pp. 725–732, 2010.
- [102] J. Daugman, "How iris recognition works," *IEEE Transactions on circuits and systems for video technology*, vol. 14, no. 1, pp. 21–30, 2004.
- [103] L. Masek *et al.*, "Recognition of human iris patterns for biometric identification," *The University of Western Australia*, vol. 2, 2003.
- [104] J. Daugman, "How iris recognition works," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 14, no. 1, pp. 21–30, 2004.
- [105] T. Tan and Z. Sun, "Casia-irisv3," *Chinese Academy of Sciences Institute of Automation*, <http://www.cbsr.ia.ac.cn/IrisDatabase.htm>, *Tech. Rep*, 2005.
- [106] P. K. Libor Masek, "Matlab source code for a biometric identification system based on iris patterns." The School of Computer Science and Software Engineering, The University of Western Australia., 2003.
- [107] I. Damgard, M. Geisler, and M. Kroigard, "Homomorphic encryption and secure comparison," *International Journal of Applied Cryptography*, vol. 1, no. 1, pp. 22–31, 2008.



- [108] I. Damgard, M. Geisler, and M. Kroigard, “A correction to ‘efficient and secure comparison for on-line auctions’,” *International Journal of Applied Cryptography*, vol. 1, no. 4, pp. 323–324, 2009.
- [109] V. Kolesnikov, A.-R. Sadeghi, and T. Schneider, “Improved garbled circuit building blocks and applications to auctions and computing minima,” *Cryptology and Network Security*, pp. 1–20, 2009.
- [110] X. Liu, K. W. Bowyer, and P. J. Flynn, “Iris recognition and verification experiments with improved segmentation method,” in *Fourth IEEE Workshop on Automatic Identification Advanced Technologies (AutoID)*, 2005, pp. 17–18.
- [111] “National institute of standards and technology (nist). iris challenge evaluation. <http://iris.nist.gov/ice>, 2005.”
- [112] J. Bringer, H. Chabanne, M. Favre, A. Patey, T. Schneider, and M. Zohner, “GSHADE: faster privacy-preserving distance computation and biometric identification,” in *Proceedings of the 2nd ACM workshop on Information Hiding and Multimedia Security*. ACM, 2014, pp. 187–198.
- [113] M. Turk, A. P. Pentland *et al.*, “Face recognition using eigenfaces,” in *Computer Vision and Pattern Recognition, 1991. Proceedings CVPR’91., IEEE Computer Society Conference on*. IEEE, 1991, pp. 586–591.
- [114] M. Turk and A. Pentland, “Eigenfaces for recognition,” *Journal of cognitive neuroscience*, vol. 3, no. 1, pp. 71–86, 1991.
- [115] “The database of faces, (formerly “the ORL database of faces”) AT&T Laboratories Cambridge,” <http://www.cl.cam.ac.uk/research/dtg/attarchive/facedatabase.html>.
- [116] P. J. Phillips, H. Moon, S. A. Rizvi, and P. J. Rauss, “The feret evaluation methodology for face-recognition algorithms,” *IEEE Transactions on pattern analysis and machine intelligence*, vol. 22, no. 10, pp. 1090–1104, 2000.
- [117] T. Sim, S. Baker, and M. Bsat, “The cmu pose, illumination, and expression (pie) database,” in *Automatic Face and Gesture Recognition, 2002. Proceedings. Fifth IEEE International Conference on*. IEEE, 2002, pp. 53–58.
- [118] Y. Huang, D. Evans, J. Katz, and L. Malka, “Faster secure two-party computation using garbled circuits.” in *USENIX Security Symposium*, vol. 201, no. 1, 2011.
- [119] T. Schneider and M. Zohner, “Gmw vs. yao? Efficient secure two-party computation with low depth circuits,” in *Financial Cryptography and Data Security*. Springer, 2013, pp. 275–292.

- [120] K. Messer, J. Matas, J. Kittler, J. Luettin, and G. Maitre, "XM2VTSDB: The extended M2VTS database," in *2nd International Conference on Audio and Video-based Biometric Person Authentication*, vol. 964. Citeseer, 1999, pp. 965–966.
- [121] G. B. Huang, M. Ramesh, T. Berg, and E. Learned-Miller, "Labeled faces in the wild: A database for studying face recognition in unconstrained environments," Technical Report 07-49, University of Massachusetts, Amherst, Tech. Rep., 2007.
- [122] A. K. Jain, S. Prabhakar, L. Hong, and S. Pankanti, "Fingercode: a filterbank for fingerprint representation and matching," in *Computer Vision and Pattern Recognition, 1999. IEEE Computer Society Conference on.*, vol. 2. IEEE, 1999.
- [123] D. Maltoni, D. Maio, A. Jain, and S. Prabhakar, *Handbook of fingerprint recognition*. Springer Science & Business Media, 2009.
- [124] M. Golfarelli, D. Maio, and D. Malton, "On the error-reject trade-off in biometric verification systems," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 19, no. 7, pp. 786–796, 1997.
- [125] A. Ross and A. K. Jain, "Multimodal biometrics: An overview," in *Signal Processing Conference, 2004 12th European*. IEEE, 2004, pp. 1221–1224.
- [126] "By the photographer (own work)[cc by-sa 3.0 (<https://creativecommons.org/licenses/by-sa/3.0>) or gfdl (<http://www.gnu.org/copyleft/fdl.html>)], via wikimedia commons from wikimedia commons."
- [127] "<https://it.mathworks.com/matlabcentral/fileexchange/31926-fingerprint-minutiae-extraction?requesteddomain=www.mathworks.com>."
- [128] "<http://www.publicdomainpictures.net/view-image.php?image=15619&picture=staring-eyes>."
- [129] "Casia-facev5 <http://biometrics.idealtest.org/>."
- [130] "<https://www.flickr.com/photos/sparkfun/8661420394/in/photostream/>."
- [131] S. Helfroush and H. Ghassemian, "Nonminutiae-based decision-level fusion for fingerprint verification," *EURASIP Journal on Advances in Signal Processing*, vol. 2007, no. 1, p. 060590, 2006.
- [132] K. Chang, K. Bowyer, and P. Flynn, "Face recognition using 2d and 3d facial data," in *ACM Workshop on Multimodal User Authentication*. Citeseer, 2003, pp. 25–32.

- 
- [133] A. Ross, A. Jain, and J. Reisman, "A hybrid fingerprint matcher," *Pattern Recognition*, vol. 36, no. 7, pp. 1661–1673, 2003.
- [134] V. Conti, C. Militello, F. Sorbello, and S. Vitabile, "A frequency-based approach for features fusion in fingerprint and iris multimodal biometric identification systems," *IEEE Transactions on Systems, Man, and Cybernetics, Part C (Applications and Reviews)*, vol. 40, no. 4, pp. 384–395, 2010.
- [135] T. Ko, "Multimodal biometric identification for large user population using fingerprint, face and iris recognition," in *Applied Imagery and Pattern Recognition Workshop, 2005. Proceedings. 34th.* IEEE, 2005, pp. 6–pp.
- [136] A. A. Ross and R. Govindarajan, "Feature level fusion of hand and face biometrics," in *Defense and Security.* International Society for Optics and Photonics, 2005, pp. 196–204.
- [137] R. Connaughton, K. W. Bowyer, and P. J. Flynn, "Fusion of face and iris biometrics," in *Handbook of Iris Recognition.* Springer, 2013, pp. 219–237.
- [138] M. Gomez-Barrero, E. Maiorana, J. Galbally, P. Campisi, and J. Fierrez, "Multi-biometric template protection based on homomorphic encryption," *Pattern Recognition*, vol. 67, pp. 149–163, 2017.
- [139] D. Boneh, E.-J. Goh, and K. Nissim, "Evaluating 2-DNF formulas on ciphertexts," in *IACR Theory of Cryptography Conference - TCC*, vol. 3378. Springer, 2005, pp. 325–341.
- [140] P. Gasti, J. Šeděnka, Q. Yang, G. Zhou, and K. S. Balagani, "Secure, fast, and energy-efficient outsourced authentication for smartphones," *IEEE Transactions on Information Forensics and Security*, vol. 11, no. 11, pp. 2556–2571, 2016.
- [141] M. Barni, J. Guajardo, and R. Lazzeretti, "Privacy preserving evaluation of signal quality with application to ecg analysis," in *Information Forensics and Security (WIFS), 2010 IEEE International Workshop on.* IEEE, 2010, pp. 1–6.
- [142] T. Bianchi, A. Piva, and M. Barni, "Analysis of the security of linear blinding techniques from an information theoretical point of view," in *Acoustics, Speech and Signal Processing (ICASSP), 2011 IEEE International Conference on*, May 2011, pp. 5852–5855.
- [143] G. Droandi, M. Barni, R. Lazzeretti, and T. Pignata, "SEMBA:SEcure multi-biometric authentication," *ArXiv e-prints*, Mar. 2018.
- [144] "Casia-irisv1 <http://biometrics.idealtest.org/>."

- 
- [145] “Fresco: Framework for efficient and secure computation,” The Alexandra Institute’s Security Lab. [Online]. Available: [https://alexandra.dk/uk/about\\_us/about\\_alexandra](https://alexandra.dk/uk/about_us/about_alexandra)



In this thesis we study the development of privacy preserving protocols for biometric recognition. This is a new research field for which a number of solutions have been proposed in recent years. In contrast with the majority of previous works, we look for protocols which are secure against *active* adversaries, that is adversaries that deliberately and arbitrarily deviate from the recognition protocol. Specifically, we propose two possible solutions using signal processing in the encrypted domain's tools.

First we use a cryptographic scheme belonging to the somewhat homomorphic scheme's family and we propose both an *identification* and an *authentication* non-interactive scheme. In the first protocol the biometric probe of a specific individual is compared with all the probes contained in a database looking for a positive match. In the second protocol, instead, the new probe of an enrolled individual is compared with the probe of the same individual stored during the enrollment phase.

As a second contribution, we propose SEMBA: a protocol secure against active adversary for multibiometric recognition. In this case we look for a trade-off between efficiency and accuracy by combining information from two biometric traits instead of only one. The protocol relies on SPDZ, a new framework proposed by Damgård et al. which is secure also in the presence of an active adversary.



The Ph.D. School of Information Engineering of the University of Siena is a school aiming at educating scholars in a number of fields of research in the Information Engineering area. The Ph.D. School of Information Engineering is part of the Santa Chiara High School of the University of Siena. A Scientific Committee of external experts recognized Ph.D. Schools belonging to Santa Chiara as excellent, according to their degree of internationalization, their research, and educational activities.