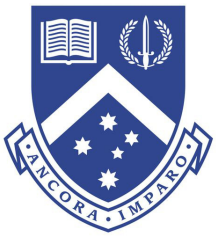
Privacy preserving authentication

Using advanced cryptographic protocols



Trung Dinh

Supervisor: Dr. Ron Steinfeld

Supervisor: Dr. Nandita Bhattacharjee

Faculty of Information Technology

Monash University

This dissertation is submitted for the degree of

Doctor of Philosophy

July 2018

I dedicate this thesis to my daughter, Laetitia. You provided the inspiration necessary for me to complete this work. I love you.

Declaration

I hereby declare that except where specific reference is made to the work of others, the contents of this dissertation are original and have not been submitted in whole or in part for consideration for any other degree or qualification in this, or any other university. This dissertation is my own work and contains nothing which is the outcome of work done in collaboration with others, except as specified in the text and Acknowledgements.

Trung Dinh

July 2018

Acknowledgements

I would like to express my profound gratitude to my main supervisor, Dr Ron Steinfeld, for his inspired guidance and advice, his infinite patience, encouragement and support during my PhD years. I am deeply indebted to him for his kindness and understanding, in both academic, professional and personal aspects.

I am beholden to my co-supervisor Dr Nandita Bhattacharjee for her priceless confidence and enlightening guidance.

I offer all my thanks to the Monash MGS and MIPRS Scholarship for their financial aid.

Last but not least, my appreciative thoughts go to my father, my mother, my wife, my daughter and my brother for their great love, support and encouragement. A very special thanks to my close friend Tang who helped me a lot during the writing of this thesis.

Abstract

In recent years, lattice-based cryptography has become one of the most active research topics in cryptography. Existing security mechanisms are currently evolving so as to provide extra usability and security features. This research focuses on user authentication issues, especially privacy-preserving biometrics authentication. Several known techniques use the homomorphic operations of some lattice-based cryptosystems to provide privacy-preserving features to the authentication protocol. However, most of them still need to rely on a trusted third party to decrypt the authentication result. Many protocols assume the so-called "Honest-But-Curious" security model for the communicating parties, while the client should rather be assumed to be "malicious" in authentication scenarios. It is therefore desirable to engineer a solution providing all necessary security features while still keeping the protocol working under practical performance conditions.

In this thesis, we propose a series of protocols with strong security guarantees and reasonable computation time and communication size, suitable for any Hamming Distance-based biometric authentication system. Our constructions rely on different state of the art cryptographic techniques designed to work with lattice-based cryptosystems.

Table of contents

List of figures

List of tables

Chapter 1

Introduction

1.1 User Authentication and Biometrics

User authentication is the process of verifying the claimed identity of a user, and is therefore a crucial component within the big picture of information and network security. Generally speaking, there are three types of identifying modes ("factors"): Something You Know (SYK), such as password; Something You Have (SYH), such as smartcard, and Something You Are (SYA), such as an iris or a fingerprint idiosyncracies. The last factor, also known as biometrics, can be considered to be the most convenient one, as it exempts system-users from carrying or remembering something for authentication purposes. Stimulated by the reduced cost of the hardware it consumes and by its unsurpassed usability level, biometric authentication has lately been deployed profusely on numerous platform types.

Various body features have been proposed and used for recognition and verification of a person's identity, the three most popular ones being fingerprints, face and iris modalities. Some commercial applications enforce other recognition techniques, based on palm-prints, hand geometry and voice. There are many other idiosyncracies proposed by researchers (such as gait, ear, keystroke dynamics, etc.), yet to attain a sufficient level of technological maturity for deployment. Regardless of the traits used, in biometrics verification systems, there are

generally two stages: enrollment and authentication. In the enrollment stage, a user registers his biometric template on a server. In the authentication stage, he produces a template afresh and submits it to the server to prove his identity. Unlike password authentication, where a user always enters the same string he chose when registering, in biometric systems, the query template of a genuine user is not exactly equal, but only very similar to the registered one, because, in the case of fingerprints, for instance, when authenticating, the user might not touch the sensor on exactly the same corner he pressed while registering . The server has thus to compute some kind of distance between the registered and the query templates, and compare the result to some threshold values, in order to decide on the authentication results. Unlike SYK and SYH, the SYA factor has two parameters, namely, False Acceptance Rate (FAR) and False Rejection Rate (FRR): FAR is the rate at which the system wrongly accepts an impersonating user as genuine, FRR is the rate at which the system rejects a genuine user. Different biometrics authentication systems use different extraction methods and distance measures to reduce FAR and FRR (this is also an active research area). In this thesis, we do not focus on template extraction techniques or on improving FAR and FRR; instead, we are interested in means providing extra privacy and security features to any methods based on Hamming Distance of biometrics represented by bitstrings.

1.2 Privacy-preserving Biometric Authentication

Many biometrics authentication systems lack protection for the data stored on a server. A server breach incident can thus result in catastrophic consequences: in 2015, nearly 6 millions plaintext fingerprint data tokens of US government employees were leaked due to a security attack [OPM]. When biometric data is revealed or stolen, the victims become vulnerable to impersonation attacks for the rest of their lives, as, unlike password or smartcards, one's fingerprints or iris are nearly impossible to change. Biometric privacy poorly protected against server exposure can therefore be considered as the main drawback of SYA authentication.

SYA proving nonetheless the most convenient authentication method from the usability point of view, a strong motivation arose to enhance it with an adequate privacy protection of biometric data. There are four main approaches to privacy-preserving biometrics authentication at present (details available in surveys of [72] and [71])

- Transforming biometric data using a key-less many-to-one transformation, also known as 'Non-Invertible' transforms (e.g. [110]): Since the function is key-less, biometric impersonation security under server exposure is vulnerable to off-line brute-force attacks, also known as FAR attacks [127, 114], which are the biometric analogue of off-line dictionary attacks on weak password hashing. Moreover, the many-to-one mapping trades off the accuracy (FRR) performance of the biometric scheme. A solution to overcome such an issue is to introduce a cryptographic secret-key authentication factor, i.e., to transform the registered biometric template stored on the server by encrypting it with a secret key stored at the client's side. With this second authentication factor, verification will require the secret key, which prevents brute-force attacks on the biometric data. Generally speaking, two-factor security can still protect the system when either the secret key is exposed (e.g., the client's device is stolen), or the encrypted biometric data stored on the server is leaked (e.g., server breach). All other approaches we consider below are based on this two-factor method.

- Transforming biometric data using a keyed 'distance-preserving' encryption scheme: Here, the distance between the biometric query template and the biometric registered one is measured by the server using encrypted versions of them. Typical techniques in this category include cancelable biometrics and biohashing ([126], [74], 2P-MCC ([33]). However, the security of such encryption schemes is unclear, as they are based on heuristic and non-standard cryptographic assumptions, several of them being reputedly insecure ([83, 81]).

- Biometric cryptosystems / Fuzzy hashing: This approach is based on techniques using error-correcting codes ([128, 96]) to extract a noiseless cryptographic key from noisy biometric data. However, depending on parameter settings, these techniques may leak significant information, and the trade-off between biometric accuracy (False Acceptance Rate and False Rejection Rate) and security remains controversial or not well understood. In practice, the implemented techniques require strong restrictions on accuracy: in particular, the underlying error-correcting codes are only capable of handling a specific range of the threshold, depending on the defined usability parameters.

- Secure Computation / Homomorphic encryption techniques ([133], [120], [64]): In this approach, the biometric data is encrypted on the server using a homomorphic encryption scheme. The operations on encrypted data are meant to measure the similarity between the query and the stored templates. This approach has the potential to overcome the heuristic security issues of the above approaches (by using an encryption scheme based on standard cryptographic assumptions), as well as the biometric accuracy issues (by implementing homomorphically the same verification check as the underlying biometric scheme). However, existing protocols following this approach suffer from other significant drawbacks. In particular, earlier approaches [120] were based on the Paillier *additive* homomorphic scheme and are granted with limited efficiency due to both the inefficiency of homomorphic operations of the Pailler scheme, and the extra protocol overhead required to handle the 'addition-only' homomorphism limitation of the this scheme. Besides, the Paillier system does not provide *long term* security (it is not quantum resistant). Its efficiency has been significantly improved by a recent protocol due to Yasuda et al. [133], based on lattice-based SomeWhat Homomorphic encryption (SWHE), which supports both homomorphic additions and multiplications. The cryptosystem used is believed to be quantum-resistant as well.

However, this protocol requires the use of a trusted-third party server to verify the final authentication result: The main server stores encrypted biometric data and computes HD homomorphically, while another other server stores the decryption key and decrypts/verifies the HD. The method is thus not secure against server exposure attacks, as keys hold by the trusted server are able to decrypt data stored on the main one.

We focus our work on the fourth approach, adding various improvements to it, the goal being to design an authentication protocol able to satisfy more complex security and usability requirements according to the current trends. These requirements include:

Privacy against server exposure. The biometric data should be securely encrypted prior to uploading it to the server, using a key known only by the biometric data owner. Ideally, the server storing the encrypted biometric information should not be able to comprehend the data. In other words, it should not possess the encryption key.

Quantum resistant privacy. Considering that quantum computing is developing fast and that biometric data will persist over the lifetime of a user, encrypted biometric data should be granted long term security against quantum computing attacks.

No reliance on trusted third parties. With the wide exposure of cloud computing to potential attacks, the protocol should not rely on any cloud-based trusted third parties to carry out the authentication process.

Security against malicious clients. An attacker attempting to impersonate the real client before the server should not be able to authenticate without genuine biometric features, even being malicious and not following the authentication protocol. Consequently, an impersonation attacker should never be assumed to be 'honest but curious'(HBC).

Two factor security. If the client is responsible for decrypting biometric-related data, the protocol should be multi-factor secure: an attacker with a compromised key should not be able to authenticate without genuine biometric features.

Practical performance. The computation time and communication size of the whole protocol should remain within a practical time frame.

Previous protocols in the literature do not meet one or more of our requirements. In particular, many previous protocols ([29], [42], [101]) assume honest-but-curious clients and are insecure in an authentication context involving malicious clients. Some protocols involving malicious clients ([120], [119]) are not quantum-resistant. Previous practical quantum-resistant protocols ([133], [90]) are not secure against malicious clients and involve a trusted third-party verification server, thus compromising the client's privacy.

1.3 Contributions of the thesis

As has been outlined, research gaps result from the combination of non-satisfied security constraints stated in Section 1.2. Either existing protocols are not safe against authentication involving malicious clients, or, when they are, they do not prove quantum-resistant. Previous quantum-resistant protocols not only involve a third party, but lack robustness with respect to malicious clients security model. The aim of our work is to overcome these security issues while providing satisfactory practical performance, in terms of computation time and communication size. We propose different protocols to support the above stated requirements. Our technique combines state-of-the-art cryptographic tools, such as Homomorphic Encryption (HE) and Zero-Knowledge-Proof (ZKP), aimed at balancing the security and the usability of the system. The techniques we put forward are all lattice-based, which appears to be one of the best framework for preserving long term security against quantum attacks. Our protocol also achieves privacy protection against server exposure, by avoiding to rely on a trusted third party. Our contributions include:

- Four different Quantum-resistant and provable-secure biometric authentication protocols that do not rely on trusted third-parties. The server stores the encrypted data

and performs homomorphic operations to compute the distance (Hamming Distance) allowing to decide on the authentication result. It does not need a third party to decrypt the encrypted HD but, instead, sends it to the client for decryption. The client uses ZKP to convince the server that the ciphertext was correctly decrypted.

- The protocols provide security under the malicious client model.This is achieved by new ZKP techniques we designed specifically for different ciphertext packing methods. It is also applicable to the proving of plaintext knowledge for the BGV Homomorphic Encryption scheme ([27]) we adapted. We apply different ZKP variants based on [125] and [116], the two main approaches for Zero Knowledge Proof.

- Due to the noise inherent to lattice-based homomorphic encryption and its correlation with the evaluated ciphertext, we observe that there can be information leakage affecting the original plaintexts used in the homomorphic computations of a two-factor attacker having exposed the client's secret key. We propose an approach to cover such leakages without significantly reducing the efficiency of the protocol: the approach is a new application of Renyi Divergence (RD)-based analysis to uncover the security of the protocol with a small 'imperfect' one-time pad. The correlation of Homomorphic Encryption noise with the original plaintext before homomorphic evaluation, has been considered to be a problem of "circuit privacy" in theoretical HE literature ([115], [69]), but the proposed solutions ([49], [102], [51]) involve 'smudging' (imperfect masking) or bootstrapping techniques. Such techniques produce an exponentially large noise (in the security parameter), which reduces efficiency. By contrast, our Renyi-based method can manage with much smaller imperfect masks, which leads to an increased efficiency of the process. To our knowledge, this is the first application of Renyi divergence techniques to circuit privacy of HE.

- We propose a new Oblivious Transfer (OT) approach to obtain inputs to Garbled Circuits under the malicious client model (the inputs' correctness need to be proved that they match the Homomorphic Encryption plaintext format). Unlike traditional OT approaches, where the receiver obtains keys as inputs to a garbled circuit, our one is compatible with the underlying homomorphic cryptosystem and smoothly integrates the encryptions of the keys to the circuit. The approach proves effective with just one level of homomorphic multiplication operations and can be deployed on various application scenarios requiring secure multi-party computations.

1.4 Organization of the thesis and related publication

The thesis is organized as follows:

In Chapter 2, we describe the general notation and recall the basics of Lattice-based cryptography, Authentication, Zero Knowledge Proof and Secure Multi-party Protocols used throughout the thesis.

In Chapter 4, we present our first variant of the protocol removing the role of trusted third parties in a privacy preserving authentication system.

In Chapter 5, we present the new technique to cover circuit privacy while still keeping the parameters of the homomorphic cryptosystems within practical thresholds.

In Chapter 6, we construct the third variant of the protocol for both computing and comparing the Hamming Distance homomorphically (Improving the privacy of the previous protocols which only compute HD on ciphertexts).

In Chapter 7, we present the last variant of the protocol, which removes the computation and communication overheads (and the extra cost they imply) during the initialization stage at the cost of an extra one-time precomputation and storage.

In the last Chapter, we conclude the thesis and state the open problems for future research. The results of Chapter 4 and 5 are contained in the following publication.

Dinh, T., Steinfeld, R., & Bhattacharjee, N. (2017, December). A Lattice-Based Approach to Privacy-Preserving Biometric Authentication Without Relying on Trusted Third Parties. In Information Security Practice and Experience - 13th International Conference, ISPEC 2017, Melbourne, VIC, Australia, December 13-15, 2017, Proceedings. Lecture Notes in Computer Science 10701, Springer 2017.

Chapter 2

Definitions and Preliminaries

2.1 Notation

We use the following notation throughout the thesis:

Vectors, Matrices We denote column vectors by standard vector notation (e.g., $\vec{x}, \vec{y}$) and matrices by bold upper-case letters (e.g., $\mathbf{A}, \mathbf{B}$). We write $\vec{b}^T$ to convert a vector $\vec{b}$ into a row vector. The length of a vector $\vec{b} = [b_0, b_1, \ldots, b_{n-1}]$ (*norms*) is denoted by $\left\|\vec{b}\right\| = \sqrt{b_0^2 + b_1^2 + \cdots + b_{n-1}^2}$ for Euclidean norm and $\|\vec{b}\|_\infty = max(|b_i|)$ for infinity norm. The i component of a vector is denoted interchangeably as b_i or $b[i]$.

Groups and Rings Sets of numbers are denoted as standard (e.g., $\mathbb{R}$ for the field of real numbers, $\mathbb{Z}$ for integers), we denote $\mathbb{Z}_q$ to be the ring of integers modulo q (in the range $(-q/2, q/2]$).$\mathbb{Z}[n]$ denotes polynomials of order n with elements in $\mathbb{Z}$, R_q denotes the ring of polynomial modulo $x^n + 1$: $R_q = \frac{\mathbb{Z}_q[x]}{x^n+1}$. We use bold lower-case to denote an element of the ring R_q (e.g., $\mathbf{a} \in R_q$).

Others We use notation $x \xleftarrow{r} D$ when x is sampled from the distribution D. If x is the output of an algorithm D, we write $x \leftarrow D$. Algorithms are $efficient$ if they run in

probabilistic polynomial time (PPT). A function is said *negligible* in n if it vanishes faster than the inverse of any polynomial.

2.2 Lattices

2.2.1 Motivation

We choose lattice-based cryptography as the main technique used in the thesis for several reasons:

Provable Security. Lattice-based cryptosystems' security can be related to known worst-case computationally hard problems such as the Shortest Vector Problem (SVP) or the Closest Vector Problem (CVP) and therefore allow to demonstrate the security property in terms of mathematical proofs. For classical systems such as RSA, there is no proof showing that breaking RSA is as hard as factoring an integer, even though in many cases the best attack we are aware of is to carry out an operation of this type.

Quantum Resistance. In 1994, Shor [121] showed that we can write a program for quantum computers to factor a large integer into prime factors. The latest D-Wave systems [80] initial deployments suggest that the idea may be feasible in near future. If this happens, it will have significant implications for the currently used public key cryptosystems such as RSA, which will be broken. Lattice based cryptography is currently one of the best candidates to resist even quantum attacks.

Homomorphic Operations. Finally, lattice-based cryptography allows us to perform operations infeasible before, namely, flexible computing on encrypted data. This special type of processing is able to protect privacy in online communication.

Research Progress. Research on Lattice-based systems has been pursued actively in the last decade and has reached a stage allowing for implementation in a near future.

Performance. Lattice-based cryptography has the potential to achieve very fast operations and is therefore a good option for high-performance requiring cryptographic applications.

We will recall some of the mathematics behind lattices (lattices are actually mathematical objects). We will look into the definitions and introduce some of the core problems underpinning security into consideration, focusing on how to use lattices to build our cryptographic schemes.

2.2.2 Lattice Preliminaries

Euclidean Lattice is an area of mathematics combining both matrices, vector algebra and integer variables. The definitions of lattices follow:

Definition 1 (Lattice)**.** *An n-dimensional (full-rank)* lattice $L(B)$ *is the set of all integer linear combinations of some* basis *set of linearly independent vectors* $\vec{b_1},\ldots,\vec{b_n} \in \mathbb{R}^n$*:*

$$L(B) = \left\{ c_1\vec{b_1} + c_2\vec{b_2} + \cdots + c_n\vec{b_n} : c_i \in \mathbb{Z}, i = 1,\ldots,n \right\}$$

The $n \times n$ matrix $B = (\vec{b_1},\ldots,\vec{b_n})$ is called a basis for $L(B)$. There are infinitely many different bases for a lattice.

Example. Consider a 2-dimensional lattice generated by the basis $(\vec{b_1},\vec{b_2}) = \left(\begin{bmatrix} 1 \\ 2 \end{bmatrix} \begin{bmatrix} 0 \\ 3 \end{bmatrix} \right)$ To generate a lattice from a basis, we compute all the integer multiples of each vector of the basis, then we add them together to generate new vectors. The result is an infinite grid of vectors, this grid is a lattice. We can illustrate graphically two or three dimensional lattices easily. For higher dimensions, it is not possible to draw them, but the intuition stands and all algebraic operations can be performed easily on the coordinates of the vectors. Lattice-based cryptography works with high dimensional lattices ($n > 100$) for security reasons. A lattice

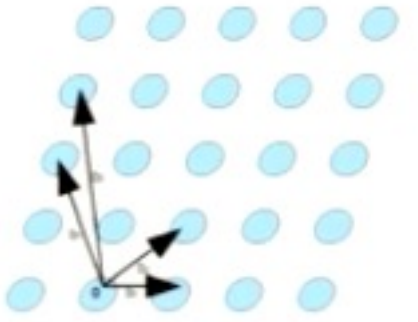


Fig. 2.1 Example of a 2-dimensional lattice

is also an infinite group under vector addition (a group is a mathematical structure, a set with a single operation on it).

Definition 2 (Fundamental Parallelepiped)**.** *For an n-dimensional lattice basis* $B = \left(\vec{b_1}, \ldots, \vec{b_n}\right) \in \mathbb{R}^{n \times n}$*, the* fundamental parallelepiped *(FP), denoted* $P(B)$*, is the set of all real valued* $[0,1)$*-linear combinations of some basis set of linearly independent vectors* $\vec{b_1}, \ldots, \vec{b_n} \in \mathbb{R}^n$*.*

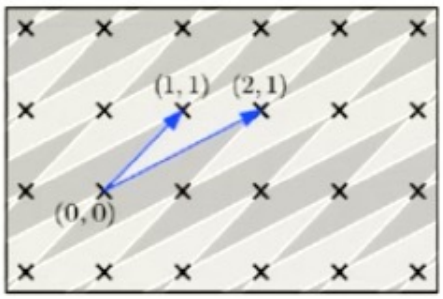


Fig. 2.2 A parallelogram of 2-dimentional lattice

In order to determine if a set of vectors is a basis of a lattice, there are geometric and algebraic approaches [93].

Lemma 1. *There is exactly one* L *point contained in P(B') (the* $\vec{0}$ *vector) if and only if B' is a basis of L.*

We let $det(U)$ denotes the determinant of the matrix U. Geometrically, the determinant computes the area (or volume) of the parallelogram of a lattice. We can associate each point

of a lattice with a parallelogram; consequently, for a large n-dimensional ball S, the number of lattice points in S is $\approx vol(S)/det(L)$.

Definition 3 (Determinant)**.** *For an n-dimensional lattice $L(B)$, the determinant of $L(B)$, denoted $det(L(B))$ is the n-dimensional volume of the FP P(B).*

Lemma 2. *B' is a basis of L(B) if and only if $B' = B \cdot U$, for some $n \times n$ integer matrix U with $det(U) = \pm 1$, such U is called unimodular matrix.*

Lemma 3. *For an n-dimensional lattice $L(B)$, $det(L(B)) = |det(B)|$.*

Cryptography involves handling of computationally complex problems. Some are related to the geometry of lattices, such as the Shortest Vector Problem (SVP) and the Closest Vector Problem (CVP).

Definition 4. *For an n-dimensional lattice L, its* minimum $\lambda(L)$ *is the length of the shortest non-zero vector of L:*

$$\lambda(L) = min(\left\|\vec{b}\right\| : \vec{b} \in L)$$

Theorem 1 (Minkowski's First Theorem)**.** *For any n-dimensional lattice L, $\lambda(L) \leq \sqrt{(n)}.det(L)^{1/n}$.*

As mentioned, the SVP problem of finding the shortest vector of a lattice becomes harder as the dimensions grows: in theoretical computer science, this problem has been shown to be NP-hard under randomized reductions [4]. In cryptography, SVP is not used as it is; instead, a variant of the problem, the definition of approximate SVP problem $\gamma - SVP$, defined below is brought into play.

Definition 5 (γ-SVP problem)**.** *Given a basis B for an n-dimensional lattice L, find $\vec{b} \in L$ with $0 \leq \left\|\vec{b}\right\| \leq \lambda(L)$.*

As γ increases, the problem becomes easier. We have good algorithms to find γ-SVP as follows:

- For $\gamma \geq 2^{O(n)}$: LLL algorithm solves in Poly(n) time.
- For $\gamma \leq O(1)$: NP-hard, it is not likely that a Poly(n) algorithm to solve the problem exists.

In cryptography, many systems depend on this γ-SVP problem, where γ is in between the two extremes $\gamma \approx O(n^c)$. Even with this factor, the best algorithm (including quantum ones) still takes $2^{O(n)}$ time to solve γ-SVP. (In classical cryptosystems for example RSA, the best algorithm to solve the integer factorization problem takes $\approx \mathscr{O}\left(2^{n^{1/3}}\right)$ time).

2.2.3 q-ary lattices and SIS problem

There are many useful subclasses of special lattices. In cryptography, one of these, termed *q-ary* lattices, which relates to the so-called Short Integer Solution - SIS, is in common use (SIS is a hard problem appearing in many cryptographic design solutions, the ZKP technique referred to below is based on a variant of this problem). The use of lattice-based cryptosystems is enforced by the hardness of the $\gamma - SVP$ problem given a basis B. It is critical to initially chose the basis B in order to make sure that solutions to the challenges it gives rise to are hard to find, as there are easy instances of $\gamma - SVP$, even for $\gamma = 1$. A way to generate *random* lattices' bases for which $\gamma - SVP$ is hard to solve 'on-average' is therefore needed. One possible answer comes from [3]: random *q-ary* lattices, for which the hardness of finding short vectors within them can be proved.

Definition 6 (Ajtai's perp lattices)**.** *Given an integer* q *and a uniformly random matrix* $A \in \mathbb{Z}_q^{n \times m}$*, the* q-ary *perp lattice* $L_q^{\perp}(A)$ *is defined by:*

$$L_q^{\perp}(A) = \left\{ \vec{v} \in \mathbb{Z}^m : A.\vec{v} = \vec{0} \mod q \right\}$$

Although it might not be clear from the aforementioned definition of lattices, the set of all vectors $\vec{v}$ actually forms a lattice, and there is a fast algorithm to compute a basis. The parameters for these lattices are A and q and we can choose a lattice to work on by selecting random values for these parameters. This particular set of *q-ary* lattices is convenient because Ajtai mathematically proved that if there is an algorithm to solve the $\gamma - SVP$ problem for these lattices, the same algorithm can be used to break $\gamma - SVP$ for any lattice. This is the *worst-case to average-case* security reduction and allows for confidence in concluding that the problem is hard on average.

We recall one fundamental tool in lattice-based cryptography: Gaussian distributions over lattices, parameterized by $\sigma > 0$, is defined by the density function

$$\forall x \in \mathbb{R}, D_\sigma(x) = 1/\sigma \cdot exp(-\pi(x/\sigma)^2)$$

The m-dimensional Gaussian distribution is defined by the density function $D_\sigma(x) = 1/\sigma^m \cdot exp(-\pi(\|\mathbf{x}\|/\sigma)^2)$. Denote $D_{\sigma,c}$ to be the m-dimensional Gaussian distribution with center $\mathbf{c} \in \mathbb{R}^m$. Let $L \subseteq \mathbb{Z}^m$ be a lattice, the discrete Gaussian distribution $D_{L,\sigma,c}$ is the m-dimensional Gaussian centered at $\mathbf{c}$, with support restricted to the lattice L. Namely, the density function of the discrete Gaussian distribution is

$$\forall x \in L : D_{L,\sigma,c}(\mathbf{x}) = \frac{D_{\sigma,\mathbf{c}(\mathbf{x})}}{\sum_{x \in L} D_{\sigma,c}}$$

Lemma 4 ([52])**.** *Given a basis* $\boldsymbol{B}$ *of an m-dimensional lattice L, a Gaussian parameter* $\sigma \geq \|\mathbf{B}\| \cdot \omega(\sqrt{\log m})$*, and an arbitrary center* $\mathbf{c} \in \mathbb{Z}^m$*, there is a probabilistic polynomial time algorithm that outputs a sample from a distribution statistically close to* $D_{L,\sigma,c}$

In the thesis, we denote D_σ for simplicity, we are interested in discrete Gaussian distribution over integers ($L = \mathbb{Z}^m$). Next, we discuss SIS, which will be extensively used throughout the thesis.

Definition 7 (Small Integer Solution (SIS) problem)**.** *$SIS_{q,m,n,\gamma}$ Given n and matrix* $\mathbf{A}$ *sampled uniformly in $\mathbb{Z}_q^{n\times m}$, find $\vec{z}\in\mathbb{Z}^{m\times n}$ such that $A\vec{v}=\vec{0}$ and $\|\vec{v}\|_\infty\leq\beta$.*

Informally, SIS is just the name given to the $\gamma-SVP$ problem for particular *q-ary* lattices. β is the approximation parameter, it defines how short the lattice vector should be. Some relations between SIS and $\gamma-SVP$ follow:

- $det(L_q^\perp(A))=q^n$
- By the Minkowski's Theorem, $\lambda(L_q^\perp(A))\leq\sqrt{m}$, for $m\geq n\log q$. It implies that, to solve SIS, vectors not much longer than $\sqrt{m}$ need to be found.
- γ of SVP can be related to β of SIS: $\gamma\approx\beta/\sqrt{m}q^{n/m}$.

The SIS problem can also be interpreted as the problem of finding a small non-trivial solution of a random homogeneous system of modular linear equations. This is formalized in [52] as *inhomogeneous* variant of SIS, also known as ISIS

Definition 8 (*$ISIS^p(n,m,q,\beta)$*)**.** *The Inhomogeneous Small Integer Solution problem (ISIS) in the l_p norm with parameters n,m,q,β denoted by $ISIS^p(n,m,q,\beta)$ is as follows: Given a uniformly random matrix $\mathbf{A}\in\mathbb{Z}_q^{n\times m}$, and a uniformly random vector $\mathbf{y}\in\mathbb{Z}_q^n$, find a vector $\mathbf{x}\in\mathbb{Z}_q^m$ such that $\|\mathbf{x}\|_p\leq\beta$ and $\mathbf{A}\cdot\mathbf{x}=\mathbf{y} \mod q$*

2.2.4 A cryptographic example

Cryptographic hash functions have many applications to provide authenticity and integrity in different security contexts. One of the main properties of cryptographic hash functions is collision resistance: it is hard to find 2 message inputs producing the same hash value output. The current constructions being used in practice are not lattice-based but built on non-linear boolean functions. We discuss how to construct a variant from lattices and show the relevant techniques to be used during security proofs and parameter choices throughout

this thesis. The security level of this function is directly related to the hardness of the SIS problem pointed out before

Definition 9 (Ajtai's Hash Function.)**.** *Pick* $A = (\mathbf{a}_{i,j}) \xleftarrow{r} \mathbb{Z}_q$ *(A is the function 'public key'). Given* $\vec{x} \in \mathbb{Z}^{mn}$ *having 'small' coordinates (*$\|\vec{x}\|_\infty \leq d$*), the hash function out put is defined as*

$$g_{q,m,n,d,A}(\vec{x}) = A.\vec{x} \mod q$$

It entails that from a long input vector, a short output vector -the hash value-, is obtained. The parameter values should be chosen so as to make sure that the collision resistance constraint is satisfied. The important question is how to determine the security of this function, or how to ensure that this function is collision resistant. To this end, we introduce the concept of security reduction, used frequently in lattice-based cryptography's security proofs. We show that if there was an algorithm to find a collision in this hash function, this algorithm could be used to solve a hard lattice problem. In our case, SIS, as analyzed by Ajtai, has shown, on solid foundations, to be hard. What we intend to show here is that, if a collision can efficiently be found in this hash function, a way to solve the SIS problem, contradicting the hardness of SIS, can equally be found. Elaborating on this contradiction, we can conclude that there must not exist any efficient algorithm to find a collision for the hash function.

Theorem 2. *Collision-Resistance of g is at least as hard as* $SIS_{q,m,n,\beta}$ *with* $\beta = 2d$.

Proof. Suppose there was an efficient collision-finder attack algorithm CF for function g: Given a random instance (A,q), CF outputs $\vec{x_1} \neq \vec{x_2}$ such that $A\vec{x_1} = A\vec{x_2}$.

CF can be used to build another algorithm S that solves the SIS problem for any instance (A,q):

- Runs CF on (A,q) to get $\vec{x_1}, \vec{x_2}$.

- S outputs solution for SIS problem: $\vec{v} = \vec{x_1} - \vec{x_2}$.

The algorithm works because $A\vec{v} = A\vec{x_1} - A\vec{x_2} = \vec{0}$, i.e., $\vec{v}$ is in the lattice $L_q^{\perp}(A)$ and $\|\vec{v}\|_\infty \leq \beta$, where $\beta = 2d$ (When vectors are added/subtracted, the result vector's length is upper bounded by the sum of the lengths of the vectors used in the operation). Moreover, S is efficient as CF is efficient (run-time $T_S \approx T_{CF}$). □

This has been the first application of lattices in cryptography and it actually triggered other modern techniques. The next important questions are:

- How to choose the parameter values to obtain a given security level?
- How does a SIS problem relate to a $\gamma - SVP$ problem?

2.2.5 Security of lattice-based cryptography

Lenstra–Lenstra–Lovász (LLL) and its various improvements are the main ways to access the security of lattice-based systems (by assessing the security of underlying problems, such as SVP or $\gamma - SVP$). They enable an appropriate choice of the size of the parameters for the problem at hand (such as dimensions), in order to guarantee security against the best known attack. We refer the reader to [99] for a detailed discussion of the algorithm.

We are generally interested in what kind of approximate SVP LLL is able to solve: If LLL is run on some basis, a reduced basis is produced as its output. The shortest vector in that reduced basis is then taken into consideration, in order to evaluate the difference between its length and the length of the shortest vector of the lattice.

Theorem 3 (Short vector from LLL)**.** . *The LLL algorithm solves in polynomial time $\gamma - SVP$ for $n - dim$ lattices, with $\gamma < 2^{(n-1)/2}$.*

In summary, the first vector of LLL output is bounded by

$$\left\|\vec{b_1^*}\right\| \leq (1/(\delta - 1/4))^{(n-1)/2} . \lambda(L)$$

It follows that the approximation factor is exponential in the dimension n of the lattice. Another way to measure the length of the output vector of LLL is the Hermite Factor (HF), which is the ratio of the algorithm's output to the n^{th} root of the determinant of the lattice. Sometimes it proves convenient to use this measure because the determinant of the lattice is easy to compute, whereas the length of the shortest vector of the lattice is usually not. In practice, HF is often used as a main measure. Again, LLL produces an HF that is exponential in the dimension n. The conclusion is that, although LLL runs in polynomial time, the approximation factor increases exponentially with respect to the dimension of the lattice. When this dimension is large, the approximation factor will grow in such a proportion that LLL will fail to break the security of underlying cryptosystems.

Theorem 4. *The LLL algorithm solves in polynomial time $\gamma - SVP$ for $n-dim$ lattices, with $\gamma \leq 2^{(n-1)/2}$. It can also be shown also be shown that Hermite Factor $\gamma_{HF} = \frac{\|\vec{b}_1\|}{det(L)^{1/n}} \leq 2^{(n-1)/4}$*

When running LLL experimentally ([98]) on a random lattice, it was shown that the algorithm performs much better than the bound showed in Theorem 4. Specifically, HF is close to 1.02^{n-1}. Although this factor is still exponential in dimension n, slow growth allows LLL to be used on large dimension lattices. In order to reduce HF further, LLL can be modified. There are several ways to do that. The basic idea is to combine LLL with an exhaustive search (enumeration algorithm). The best lattice exhaustive search algorithm (Fincke-Phost/Kannan) does it in time $2^{O(n\log(n))}$; actually, it performs even worse than exponential time growth.

There are variants of such a method, but they all take at least exponential time with respect of the dimension n (with the expense of using $2^{O(n)}$ memory). In general, all enumeration algorithms are very inefficient. Therefore, combining LLL with brute-forcing is a more practical approach: BKZ is the typical algorithm in this category. BKZ allows trades-off between run-time and approximation factor γ, LLL is modified by changing the block size

k of vectors in the swapping step. As k is increased, the lattice of dimension k is searched for by using brute force search first and then running LLL with a smaller k dimension. The 'interpolation' between the two extremes of LLL (corresponding to $k = 2, \gamma = 2^{O(n)}, T = n^c$) and the enumeration (corresponding to $k = n, \gamma = 1, T = 2^{O(n\log(n))}$) are computed. In between, BKZ can achieve $\gamma(k) \leq k^{(n-1)/(k-1)}$ with running time $n^c.k^{O(k)}$. [63]. From this trade-off setting, an attacker can then choose the optimal value of k for some particular context: A lattice-based cryptographic scheme can only be broken if the adversary can find short enough vectors. This also defines how small γ needs to be in order to resist such an attack.

In summary, given a security level λ for a system (i.e., taking time 2^{λ} for the best attack to work even with the best value choices for the BKZ parameter k), the best approximation factor γ that the algorithm can reach will be related to λ as:

$$\log(\gamma_{HF}) = \Omega(\frac{n\log^2\lambda}{\lambda})$$

where Ω represent the asymptotic "lower than" condition when the parameters are large. Overall, we can conclude the *lattice 'rule of thumb'* for $\gamma - SVP$ using BKZ is:

$$n = \Omega(\frac{\lambda}{\log^2\lambda}.\log\gamma_{HF}) \approx \lambda.\log\gamma_{HF}$$

Remark 1. *The dimension n of a lattice-based cryptosystem needs to be proportional to the product of the bit-security level λ and the* log *(base 2) of the approximation factor γ_{HF}. This factor is a reason behind the long keys of such cryptosystems.*

For concrete values of parameters, Chen and Nguyen [35] gave numerical estimates for the Hermite Factor and time for random lattices versus block size for optimized BKZ variants:

Table 2. Approximate required blocksize for high-dimensional BKZ, as predicted by the simulation

Target Hermite Factor	1.01^n	1.009^n	1.008^n	1.007^n	1.006^n	1.005^n
Approximate Blocksize	85	106	133	168	216	286

Table 3. Upper bound on the cost of the enumeration subroutine, using extreme pruning with aborted-BKZ preprocessing. Cost is given as $\log_2$(number of nodes).

Blocksize	100	110	120	130	140	150	160	170	180	190	200	250
BKZ-75-20%	41.4	47.1	53.1	59.8	66.8	75.2	84.7	94.7	105.8	117.6	129.4	204.1
Simulation of BKZ-90/100/110/120	40.8	45.3	50.3	56.3	63.3	69.4	79.9	89.1	99.1	103.3	111.1	175.2

Fig. 2.3 BKZ parameters

From table 2, for a given HF that BKZ is set to achieve, it gives the block size parameter k needed for use in the attack. It should be noticed that the HF is much smaller than the value defined for LLL (whose $HF \approx 1.02^{n-1}$ in practice). Once the block size is known, the running time of the algorithm can be estimated after the contents of table 3.

2.2.6 An example of parameter values choice

We discuss Ajtai's hash function as an example of how to choose parameter values for a given security level. We showed that if there is an algorithm that can break the collision-resistance property of the hash function, it can be use it to find a short vector of the SIS problem with length $\beta = 2d\sqrt{m}$. The question is how to choose the values of the parameters q, n, m, d for the hash function to reach a given security level λ based on the hardness of SIS. According to the BKZ state-of-the-art attack, to attain a running time of 2^λ, the block size and the corresponding Hermite Factor γ_{HF} can be derived from the information contained in table 3. This information allows to understand that the best attacker can compute a non-zero vector $\vec{v}$ in a SIS lattice $L_q(A)$ of norm $\leq l = min(q, \gamma_{HF}.det(L_q(A))^{1/m})$. The result of comparing l with β (or $2d\sqrt{m}$), if $l < \beta$, predicts the success or the failure of a possible attack.

There are often ways to optimize such attacks, as the attacker can choose parameter values allowing him to minimize effort. For example in the attack of the SIS problem mentioned in [92], the attacker can look at a subset $m' \leq m$ of columns of A, finding a short vector $\vec{v'}$ such that $A'\vec{v'} = \vec{0} \mod q$. He can choose $\vec{v''} = \vec{0}$ and set $\vec{v} = \vec{v'} + \vec{v''}$. It turns out that there are optimal values an attacker can choose for m', which minimize the length $l(m')$ for $\vec{v'}$. Specifically, $l(m') = min(q, \gamma'_{HF}.det(L_q(A))^{1/m'})$ is a function of m' (the graph of this function is illustrated in Figure 2.4). It appears that at some point m' there is

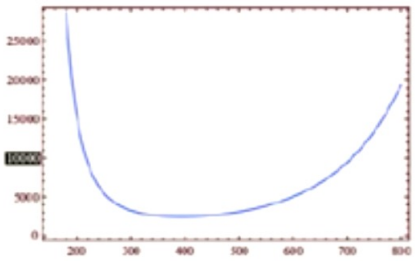


Fig. 2.4 Choosing best m' for SIS attack for parameters $\delta = 1.01, q = 4416857, n = 100$

some minimum value for the length of the vector obtained with BKZ, which is the value an attacker would want to choose. Denoting m^* to be this optimal selection, results in $l(m^*) = min(q, 2^{2.sqrtn \log q \log \delta})$. The condition for the failure of the attack being $l(m^*) > \beta$, so

$$q \geq \beta = 2d\sqrt{m} \text{ and } n \geq \frac{\log^2(\beta)}{\log q \log(\delta)}$$

We have looked at the best practical algorithms to attack lattice problems (LLL and its variants). In theory, there is also Ajtai's proof, where the hardness of solving the SIS problem for random matrices is addressed. The connection between average case and worst-case complexity of lattice problems in general is the theoretical foundation of lattice-based cryptography. We present Ajtai's average-case to worst-case connection Theorem (1996, improved by [52]).

Theorem 5 (Ajtai's hardness proof for SIS.)**.** *If there is an algorithm A that solves* $SIS_{m,n,q,\beta}$ *in poly-time, for some non-negligible fraction of input matrices* $G \in \mathbb{Z}_{||}^{\geqslant \times \Join}$*, then there is an algorithm B that solves* $\gamma - SVP$ *in polynomial time for all input lattices L of dimension n with:*

$$\gamma = O(\beta\sqrt{n}), q = \omega(\gamma\sqrt{\log n})$$

Although the theorem does not directly yield concrete information to choose parameter values securing a cryptosystem of the type discussed previously, it does provide qualitative confidence in the security level of the problem.

2.3 Learning With Errors

In the last section, we discussed how to use lattice-based cryptography to construct a hash functions. This section looks into another main tool in cryptography , that is, encryption, or how to ensure confidentiality. We need here to use a type of lattice problem slightly different from the SIS problem, much more suitable for building the encryptions based on it: the Learning With Error (LWE) problem. We will discuss how to use LWE to build a symmetric key encryption as well as how to modify it to achieve public key encryption (Regev cryptosystem). This cryptosystem is foundational, as it grounds further developments in the area and techniques used in this thesis.

In the SIS problem, given a matrix A, a short vector has to be found: $\vec{v}$ such that $A\vec{v} = \vec{0}$. The cryptographic hash function based on SIS is constructed so as to allow arbitrary inputs and produce collision-resistance outputs. Such an operation always implies a many-to-one function type in the construction: given an output, the original input must not be derivable from it. This many-to-one property is not useful in encryption contexts: ciphertexts cannot be decrypted in order to obtain the original plaintexts using such a method. In encryption

contexts, the reverse situation is usually the case : the set of possible inputs might be a lot smaller than the set of possible outputs. The function serving encryption should be at least a one-to-one function mapping a plaintext to a unique ciphertext. Or there may even be one-to-many functions mapping a plaintext to many possible ciphertexts: as long as there is no intersection between the ciphertext output sets, it is still possible to achieve a correct decryption. Such situations appear in non-deterministic cryptosystems, where the encryption function takes the message as well as a randomness factor as its inputs. Actually, almost all encryptions being used in practice are somehow randomized, the main reason being to stop the leakage of information given many ciphertexts. A deterministic cryptographic scheme can be exploited if the plaintext space is small: suppose that Alice sends deterministic encrypted messages to her agent Bob everyday instructing him to sell or keep some of her companies' shares. An attacker capturing the ciphertexts sent throughout several days can easily distinguish such decisions without having to decrypt the messages at all. This is one of the important properties that we want to include in cryptosystems: indistinguishability, also known as IND-CPA. This type of security prevents the adversary from distinguishing the ciphertexts, even if all the possible plaintexts are known.

We introduce the LWE problem [111] allowing to construct one-to-one or one-to-many functions. The main idea behind LWE is that, given a secret vector and a random lattice point with some 'small noises' added to the point, it is infeasible for an attacker to conclude anything from the output, but it is easy for the data owner possessing the secret to revert back to the original source. The description of LWE follows.

2.3.1 Definitions

Let us assume a matrix $A^{m\times n}$, initialised by defining integers q,m,n. This matrix is different from SIS regarding its dimension, just to comply with a convenient representation purpose. q is still the modulus of $a_{i,j}$, where $a_{i,j} \xleftarrow{r} \mathbb{Z}_q$. Let $\vec{s}^T = [s_1, s_2, \ldots, s_n]$ be a vector of independent

uniformly random elements of $\mathbb{Z}_q$, this is the secret element that corresponds to the secret key of the cryptosystem. Let $\vec{e}^T = [e_1, \ldots, e_n, \ldots, e_m]$ be a vector of independent 'small' integers, each one of them sampled from a probability distribution $\chi_{\alpha q}$. By 'small', we mean $\|e_i\|_\infty \leq \alpha.q$ for some parameter $0 < \alpha < 1$. Typically, $\chi_{\alpha q}$ is chosen from a Normal (Gaussian) distribution with standard deviation $\approx \alpha.q$. (From the implementation point of view, one way to achieve such sampling efficiently is to take the real samples from a normal continuous distribution and round them to the nearest integers, the mean of the distribution is set to 0 to make sure the samples are small).

We discuss two variants of the LWE problems to be used in cryptosystems: Search-LWE and Decision-LWE problems.

Definition 10 (Search-LWE Problem.)**.** *Given q, m, n, α and a matrix $A \xleftarrow{r} \mathbb{Z}_q^{m \times n}$ and $\vec{y} = A.\vec{s} + \vec{e} \mod q$ (with $\vec{e} \xleftarrow{r} \chi_{\alpha q}^m$ and $\vec{s} \xleftarrow{r} \mathbb{Z}_q^n$), find $\vec{s}$.*

Definition 11 (Decision-LWE Problem.)**.** *Given q, m, n, α and $A \xleftarrow{r} \mathbb{Z}_q^{m \times n}$, $\vec{y}$, distinguish between the following two scenarios*

- Real Scenario: $\vec{y} = A.\vec{s} + \vec{e} \mod q$ *(with $\vec{e} \xleftarrow{r} \chi_{\alpha q}^m$ and $\vec{s} \xleftarrow{r} \mathbb{Z}_q^n$)*
- Random Scenario: $\vec{y} \xleftarrow{r} \mathbb{Z}_q^m$.

2.3.2 Security of LWE

An average-case to worst-case connection similar to Theorem (5) for the SIS problem can also be established for LWE ([111])

Theorem 6. *If there is an algorithm A that solves $DLWE_{q,m,n,\alpha}$ in poly-time, with non-negligible distinguishing advantage, for $\alpha.q > 2\sqrt{n}$, then there is a quantum algorithm B that solves $\gamma - GapSVP$ in polynomial time for all input lattices L of dimension n with:*

$$\gamma = O(n/\alpha)$$

Theoretically there are strong reasons to believe that LWE is hard. The theorem even says that it links the security of LWE to the quantum security of the shortest vector problem. Using LWE as the basis for quantum resistant cryptographic techniques therefore deserves high confidence. In practice, one of the best known attack against LWE amounts to reducing the problem to the SIS problem [93]: Given a LWE instance $(A \in \mathbb{Z}_q^{m \times n}, \vec{y} \in \mathbb{Z}_q^m)$:

- Find a short non-zero vector $\vec{v}$ in the SIS lattice $L_q^{\perp}(A^T)$ with $\|v\| \leq \beta$. That is, $A^T.\vec{v} = \vec{0} \mod q$.
- Compute $e' = \vec{v}^T.\vec{y} \mod q$ and check:
  - In a 'Real DLWE Scenario', e' is small
  - In a 'Random Scenario', e' is not small

In other words, solving $DLWE_{q,m,n,\alpha}$ reduces to solving $SIS_{q,m,n,\beta=1/\alpha}$. Because of this, to secure the system based on DLWE, parameter values should be chosen in such a way that $SIS_{q,m,n,\beta}$ is hard. (The smaller the noise we choose for DLWE, the larger the β becomes, or $\gamma - SVP$ is easier, which is expected). The condition of $\alpha q > 2\sqrt{n}$ is also important, as when the noise becomes small enough, it turns out that there are efficient algebraic attacks against LWE [6].

2.3.3 Cryptosystems from LWE

Although we mostly use public key systems in our work, we first present a typical basic symmetric cryptosystem to demonstrate the frequent used techniques of the operations, correctness and security analysis. A randomize cryptosystem based on LWE:

Key Generation - KeyGen. Fix integers q, n. Pick a secret key $\vec{s} \xleftarrow{r} \mathbb{Z}_q^n$.

Encryption - Enc. Fix integers t, l. Given a message $\vec{m} \in \mathbb{Z}_t^l$:

- Pick $A \xleftarrow{r} \mathbb{Z}_q^{l \times n}$ and 'small' noise $\vec{e} \xleftarrow{r} \chi_{\alpha q}^l$.

- Compute $\vec{c} = A.\vec{s} + \vec{e} + \lceil q/t \rfloor .\vec{m} \mod q$.
- Return ciphertext $(A, \vec{c}$.

Decryption - Dec. Given a ciphertext $(A, \vec{c})$ and a secret key $\vec{s}$:

- Compute $\vec{c'} = \vec{c} - A.\vec{s} \mod q$.
- Compute $\vec{c''}$ by rounding coordinates of $\vec{c'}$ to the neareast multiple of $\lceil q/t \rfloor$ mod q. This step exploits the fact that $\vec{e}$ is a 'small' vector rather than a random one.
- Return plaintext $\vec{m} = \frac{\vec{c''}}{\lceil q/t \rfloor}$

Correctness. Decryption is correct if the rounding step succeeds, or if all the noise coordinates e_i of $\vec{e}$ are sufficiently small:

$$e_i < \frac{1}{2}.\lceil q/t \rfloor \approx \frac{q}{2t}$$

If the noise distribution $\chi_{\alpha q}$ is a normal distribution with standard deviation αq, the probability of the noise exceeding $\frac{1}{2t\alpha}$ in magnitude is $p_e \approx 2.\left(1 - \Phi\left(\frac{1}{2t\alpha}\right)\right)$, where Φ is the cumulative distribution function of the standard normal distribution (mean 0 and standard deviation 1). Therefore, to ensure correctness, this probability should be kept sufficiently small, in order for the following *correctness condition* to hold:

$$t << \frac{1}{2\alpha}$$

We can see that the larger the noise, the smaller our message space becomes. However, when the noise is large, LWE is harder as well: security goes up when noise increases. Generally, in lattice-based cryptosystems, there is this trade-off between security and the size of the messages to encrypt.

Security. The security of LWE cryptosystems are based on the hardness of LWE related problems. The idea is to prove that, if an adversary can break the encryption scheme's security, then he can also solve the DLWE problem.

The standard security model we use in our analysis would be indistinguishability security against Chosen Plaintext Attacks (IND-CPA). The formal 'security game' is defined between a challenger *Ch* and the attacker *B* targeting the encryption scheme.

CPA Security Game. The game is defined as follows.

- The challenger *Ch* runs *Keygen* algorithm and obtains a secret key $\vec{s}$.
- The attacker *B* is given access to an 'encryption oracle': *B* can submit a plaintext $\vec{m}$ and get a ciphertext $(A,C) = Enc(\vec{,}\vec{m})$ back from *Ch*. *B* can query the oracle as many times as he wants. When B finishes the query phase, he submits a pair of 'challenge messages' $\vec{m}_0^*, \vec{m}_1^*$.
- Challenger picks a random bit $b \xleftarrow{r} U(0,1)$, computes a 'challenge ciphertext' $(A^*,C^*) = Enc(\vec{s}, \vec{m}_b^*)$ for the challenge message selected by b, and sends (A^*,C^*) to *B*.
- *B* can continue querying the 'encryption oracle'. Finally, he outputs a guess b' for the bit b chosen by the challenger. The attacker wins the game if $b' = b$.

Definition 12 (IND-CPA security.)**.** *A cryptosystem is IND-CPA secure with a security level λ if any attack algorithm B with run-time $T(B) \leq 2^\lambda$ can only win the security game with probability $\leq \frac{1}{2} + \frac{1}{2^\lambda}$. (Note that the run-time of the attacker has to be restricted as, for any cryptosystem, an attacker with unlimited time can always win the game with probability 1 using brute force search.)*

We discuss a technique to modify Symmetric-Key to Public-key Encryption (Regev's cryptosystem, 2005). This cryptosystem is the basis for a lot of other lattice-based crypto-

graphic techniques, including the ones used later in our project. Supposing a symmetric key system is used to encrypt a message $\vec{m} = \vec{0}$, it is the case that $Enc(\vec{s}, m) = Enc(\vec{s}, 0) + [\vec{0}, m] \mod q$ (recall that $[\vec{a}, \vec{as} + \vec{e} + m] = [\vec{a}, \vec{as} + \vec{e}] + [\vec{0}, m]$). This implies that a message can be encrypted by adding itself to the encryption of zero $Enc(0)$, and can therefore use that $Enc(0)$ as the public key! However, the ciphertext encrypted this way can be simply broken by just one subtraction. A further attempt can be publishing several $\vec{p}_i = Enc(\vec{s}, 0)$, which are all different as the implemented scheme is non-deterministic. During encryption, a random linear combination of such encryptions can be generated so as to have a 'fresh' $Enc(0)$ and use it for encryption.

Keygen. Fix integers q, m, n. Select a secret key $\vec{s} \xleftarrow{r} \mathbb{Z}_q^n$ and publish the public key $(A, \vec{p})$, where $A \xleftarrow{r} \mathbb{Z}_q^{m \times n}$ and $\vec{p} = A.\vec{s} + \vec{e} \mod q$ with $\vec{e} \xleftarrow{r} \chi_{\alpha q}^m$.

Encryption - Enc. Fix integers t, B_r. Given a message $m \in \mathbb{Z}_t$ and the public key $(A, \vec{p})$, select coefficients vector $\vec{r} \xleftarrow{r} \{-B_r, \ldots, B_r\}^m$, compute and return:

$$(\vec{a}^T, c) = (\vec{r}^T.A, \vec{r}^T.\vec{p} + \lceil q/t \rfloor .m \mod q)$$

Decryption - Dec. Given a ciphertext $(\vec{a}^T, c)$ and the secret key $\vec{s}$:

- Compute $c' = c - \vec{a}^T.\vec{s} \mod q$
- Compute $c'' \in \mathbb{Z}_q$ by rounding c' to the closest multiple of $\lceil q/t \rfloor \mod q$.
- Return plaintext $m = \frac{c''}{\lceil q/t \rfloor}$.

Observation. For small r_i: $r_1.Enc(\vec{s}, 0) + r_2.Enc(\vec{s}, 0) + \cdots + r_i.Enc(\vec{s}, 0) = Enc(\vec{s}, 0)$.

Correctness. The condition for correct decryption is similar to the symmetric setting where the new noise $e < \frac{1}{2}.\frac{q}{t}$. The noise used in the public key randomizing process grows at rate $(|e| > |e_1|, |e_2|, \ldots, |e_i|)$, but it is still small as long as r_i are small. Generally,

if the original noise distribution is normal with a standard deviation αq, then the distribution of the new noise $e = \vec{r}\vec{e}$ is also normally distributed with standard deviation $\alpha q \|\vec{r}\|$. Also, the expected value of $\|\vec{r}\| \approx \sqrt{B_r(B_r+1)m/3}$. At the end, to ensure the correctness of the system, the parameters should be set up to lower the probability of the noise growing big. The error probability per coordinates p_e is the probability of a standard normal random variable (mean 0, standard deviation 1) to exceed $\frac{1}{2t\alpha}$ in magnitude:

$$p_e \approx 2.\left(1-\Phi\left(\frac{1}{2t\alpha}.\sqrt{\frac{3}{B_r(B_r+1)m}}\right)\right),$$

where Φ is the cumulative distribution function of the standard normal distribution. We can rewrite the following correctness condition in terms of t:

$$t << \frac{1}{2\alpha}.\sqrt{\frac{3}{B_r(B_r+1)m}}$$

Again, we observe that if a larger noise is used, a better level of security is attained, but the message space becomes smaller. In the Public-key scheme, as compared with the symmetric-key system, another factor of $O(\sqrt{m}$ in t is lost when carrying out randomization.

Security. Similarly to the symmetric key system, the IND-CPA attack model for an attacker B and a challenger Ch needs to be defined:

- Ch runs KeyGen algorithm to obtain a secret key $\vec{s}$ and a public key $(A, \vec{p})$. The public key is given to the attacker B.
- B does not need to access an encryption oracle, it can simulate the operation itself, this is the main difference compared to the symmetric security model. B just sends a pair of 'challenge messages' $\vec{m_0^*}, \vec{m_1^*}$ to Ch .

- *Ch* selects a random bit $b \xleftarrow{r} 0,1$ and computes the 'challenge ciphertext' $(\vec{a}^*, c^*) = Enc((A,\vec{p}),\vec{m}_b^*)$ for the challenge message selected by b, and hands the result over to B.
- Attacker B outputs a guess b' for the bit b chosen by the challenger. B wins the game if $b' = b$.

Definition 13 (Regev IND-CPA security.)**.** *The public key cryptosystem is secure at* 2^λ *level if any attacker B with run-time* $T(B) \leq 2^\lambda$ *wins the game with probability* $\leq 1/2 + 1/2^\lambda$.

We introduce another technique to analyze the security of a cryptosystem. Unlike the symmetric key scenario, where a distinguisher is built using an assumed attacker and proving the security by contradiction, in this scenario, the closeness of probability distribution of the ciphertexts from the real attack and the ciphertexts from the reduction are measured. The idea is that, if the obtained distance is small enough, the attack algorithm will not be able to distinguish the ciphertexts. In cryptography, statistical distance is usually used to measure such distances.

Definition 14 (Statistical Distance.)**.** *For two probability distributions* D_1 *and* D_2 *on a discrete set S, the statistical distance* $\Delta(D_1,D_2)$ *is defined as*

$$\Delta(D_1,D_2) = \frac{1}{2}.\sum_{s\in S}|D_1(x) - D_2(x)|$$

Lemma 5. *Let* D_1, D_2 *be any two distributions, and A be any algorithm, then:*

$$|Pr_{x\xleftarrow{r}D_1}[A(x)=1] - Pr_{x\xleftarrow{r}D_2}[A(x)=1]| \leq \Delta(D_1,D_2)$$

This lemma is very useful because it states that if we make the distance small enough (negligible), then no matter what algorithm an attacker uses, he will not succeed in distinguishing the distributions by a non-negligible advantage. We will bring out that the distribution generated in the security reduction (shown to the attacker by the distinguisher) is within a negligible statistical distance to what the attacker witnesses during a real attack.

2.3.4 Efficiency of Lattice-based cryptosystems

The following properties of any cryptosystem are used to evaluate its efficiency:

- Public key size : The length of the public key is important, as this key may be stored in the local memory or sent over the network.
- Ciphertext size: It should be reduced as much as possible to save up communication size. The expansion ratio of the ciphertext should be kept small.
- Processing time: The computation time for performing encryption and decryption should be minimized to improve general efficiency.

For the Regev's Public-key encryption scheme, the public key is $pk = (A \xleftarrow{r} \mathbf{Z}_q^{m\times n}, \vec{p} = A.\vec{s}+\vec{e})$, hence, $length(pk) = m.(n+1)\log q \geq n^2 \log q$. According to the 'lattice rule of thumb', the key length is at least quadratic in the security parameter λ: $O(\lambda^2)$, this is a very large factor compared to the ones occurring in classical public key systems. Similarly, the ciphertext expansion ratio will at least be $O(\lambda)$, which is also large (in practice, a value close to 1 is ideal). In terms of encryption time, the main cost is in the matrix multiplication operation, and generally it is also at least quadratic in λ: $O(\lambda^2)$. We will discuss approaches to improve all of these aspects below.

Reducing Ciphertext Expansion. The first attempt to improve Regev's public key scheme by squeezing more messages into a ciphertext was proposed by [105] . It has been

observed that the $\vec{a^T} = \vec{r^T}.A \mod q$ part of the ciphertext is independent of the length of message m and takes lots of space. Instead of using a new random nonce for each message m, the one already used can be re-instantiated with new secret keys $\vec{s_i}$ to encrypt several integer messages, by encoding them in an integer vector. The modified scheme is presented as follows, with l being the number of secret key vectors.

- Secret-key $S = (\vec{s_1}, \ldots, \vec{s_l}) \in \mathbb{Z}_q^{n \times l} \log q$.
- Public-key $pk = (A \xleftarrow{r} \mathbb{Z}_q^{m \times n}, P = (\vec{p}_1, \ldots, \vec{p}_l))$ where $\vec{p}_i = A.\vec{s}_i + \vec{e}_i \mod q$, with $\vec{e}_i \xleftarrow{r} \chi_{\alpha q}^m$.
- Encryption - Enc $(\vec{m} \in \mathbb{Z}_t^l)$: Return the ciphertext $C = (\vec{a}^T = \vec{r}^T.A \mod q, \vec{c}^T = \vec{r}.P + \lceil q/t \rfloor.\vec{m} \mod q)$. It appears that the ciphertext's size increases from $n \log q$ to $(n+l)\log q$, the message length also increases from one integer to l integers. The expansion ratio $\frac{(n+l)\log q}{l \log t}$ can therefore be reduced. However, the IND-CPA security still holds as the scheme does not reuse $\vec{p}_i$ during encryptions. Although $\vec{p}_i$ increases the size of the public key, A's size still dominates.
- Decryption - $Dec(C = (\vec{a}^T, \vec{c}^T))$: Compute $\vec{c'}^T = \vec{c}^T - \vec{a}^T.S \mod q$ and round to the closest multiple of $\lceil q/t \rfloor \mod q$ to get $\vec{c''}$. Return plaintext $\vec{m} = \frac{\vec{c''}}{\lceil q/t \rfloor}$

Reducing Storage and Computation. The approach toward this problem has an interesting history, it evolved even before the introduction of LWE. The idea is to put some structure into the matrix A: So far, all the elements of A are chosen uniformly at random: as A is a random $m \times n$ matrix with $m \geq n$, the total number of elements in

the matrix is $m.n \geq n^2$:

$$A = \begin{bmatrix} a_{1,1} & a_{1,2} & \dots & a_{1,n} \\ a_{2,1} & a_{2,2} & \dots & a_{2,n} \\ \vdots & \vdots & \ddots & \vdots \\ a_{n,1} & a_{n,2} & \dots & a_{n,n} \\ \vdots & \vdots & \ddots & \vdots \\ a_{m,1} & a_{m,2} & \dots & a_{m,n} \end{bmatrix}$$

If, instead of choosing the elements independently from each-other, they are selected on the basis of correlations between rows or columns, n^2 elements won't need to be further specified when describing A. In other words, they can be derived from existing ones. The question is: how to do that securely? The most common approach was introduced by [66, 91], it is a special structured type of matrix called "negacyclic matrix", denoted by $rot(\vec{a})$, where $\vec{a}$ is an n-dimensional vector. Once $\vec{a}$, which is the first column of the rot() matrix, is specified, all the other columns can be derived from it: the rule is quite simple, the next column is generated by rotating the previous column by one position and changing the sign of the first element:

$$\begin{bmatrix} a_0 & -a_{n-1} & -a_{n-2} & \dots & -a_1 \\ a_1 & a_0 & -a_{n-1} & \dots & -a_2 \\ a_2 & a_1 & a_0 & \dots & -a_3 \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ a_{n-1} & a_{n-2} & a_{n-3} & \dots & a_0 \end{bmatrix}$$

The output of $rot(\vec{a})$ is a square $n \times n$ matrix. The matrix A, can thus be constructed as

$$A = \begin{bmatrix} rot(\vec{a}_1) \\ rot(\vec{a}_2) \\ \vdots \\ rot(\vec{a}_{m/n}) \end{bmatrix}$$

Therefore, the storage space for the matrix A reduces from $m \times n \log q$ to $m \log q$ as only the first column of the matrix needs to be stored. Another distinguished property of this rotational structure is its correspondence with the ring multiplication operation of the ring $R_q = \mathbb{Z}_q[x]/(x^n+1)$. In other words, the expensive operation of multiplying a matrix by a vector can be achieved by a single polynomial multiplication: For two n-dimensional vectors $\vec{a}, \vec{x}$ represented by two polynomials $a(x), s(x) \in \mathbb{Z}_q[x]$ of degree less than $n-1$, let $c(x) = a(x).s(x) \mod x^n+1$, or $c(x) = \sum_{i<n} s_i x^i a(x) \mod x^n+1$. Incidentally, $x(a_0 + a_1 x + a_2 x^2 + \cdots + a_{n-1}x^{n-1}) \mod x^n+1 = -a_{n-1} + a_0 x + a_1 x^2 + \cdots + a_{n-2}x^{n-1}$.

Hence, $rot(\vec{a}).\vec{s} \mod q = a(x).s(x) \mod x^n+1$ can be presented as:

$$\begin{bmatrix} c_0 \\ c_1 \\ \vdots \\ c_{n-1} \end{bmatrix} = \begin{bmatrix} a_0 & -a_{n-1} & -a_{n-2} & \dots & -a_1 \\ a_1 & a_0 & -a_{n-1} & \dots & -a_2 \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ a_{n-1} & a_{n-2} & a_{n-3} & \dots & a_0 \end{bmatrix} \cdot \begin{bmatrix} s_0 \\ s1 \\ \vdots \\ s_{n-1} \end{bmatrix}$$

After reinterpreting the operation this way, a fast algorithm can be put at work for polynomial multiplication, in order to speed up the computation time. Specifically, the cost of matrix-vector multiplication can go down from $O(n^3)$ to $O(n \log n)$ (Fast Fourier Transform - FFT polynomial multiplication).

Reducing Computation with FFT. There are several variants of Fourier Transform (FT). One of the classical usages of FT in engineering is to convert a time function of periodic signal into the frequency domain of that function. The version of FT we will use here is similar, but it works on discrete structures: the input to the FT is a n-dimensional vector (which can also be represented in terms of a polynomial), the output is another polynomial whose coefficients are the evaluations of the input polynomial at the n roots of unity. If the FFT is performed on $\mathbb{Z}_q$, it is called Number Theoretic Transform, short, NTT (the original FT usually works with complex numbers). It should be kept in mind that the NTT's roots of unity are ζ_i such that $\zeta_i^n = -1 \mod q$, and that in order for ζ_i to exist, the condition on q is, for $q-1$, to be divisible by $2n$. Speeding up the polynomial multiplication of $\mathbf{c}(\mathbf{x}) = \mathbf{a}(\mathbf{x}).\mathbf{s}(\mathbf{x}) \mod x^n + 1$, where $\mathbf{a}(\mathbf{x}), \mathbf{s}(\mathbf{x}) \in \mathbb{Z}_q[x]$, can be obtained as follows:

- Choose q such that $2n$ divides $q-1$, then x^n+1 has n zeros in $\mathbb{Z}_q$ of the form ζ^{2i+1} for $i = 0, \ldots, n-1$, where $\zeta \in \mathbb{Z}_q$ is a primitive $2n^{th}$ root of 1 in $\mathbb{Z}_q$.
- Evaluate $\mathbf{a}(\mathbf{x})$ and $\mathbf{s}(\mathbf{x})$ at the n points ζ^{2i+1} in $\mathbb{Z}_q$ to compute the evaluation vectors $(\mathbf{a}(\zeta), \ldots, \mathbf{a}(\zeta^{\mathbf{2n-1}}))$ and $(\mathbf{s}(\zeta), \ldots, \mathbf{s}(\zeta^{\mathbf{2n-1}}))$. This operation corresponds to multiplication by an FFT-like matrix and takes $O(n \log n)$ multiplication/addition over the ring $\mathbb{Z}_q$.
- Multiply the evaluations at each point $\mathbf{c}(\zeta^{\mathbf{2i+1}}) = \mathbf{a}(\zeta^{\mathbf{2i+1}})\mathbf{s}(\zeta^{\mathbf{2i+1}})$ for $i = 0, \ldots, n-1$.
- Interpolate (inverse NTT) $(\mathbf{c}(\zeta), \ldots, \mathbf{c}(\zeta^{\mathbf{2n-1}}))$ to reconstruct $\mathbf{c}(\mathbf{x})$. This operation again takes $O(n \log n)$ multiplications/additions over $\mathbb{Z}_q$.

The details of NTT algorithms are not discussed in this project. To all appearances, the time to compute polynomial multiplication with NTT is $O(n \log n)$ instead of $O(n^2)$ if a classical arithmetic method is enforced.

In summary, by using this structured rot matrix, not only storage space of the key from $O(n^2)$ to $O(n)$ can be saved, but also the time of the most expensive operation from $O(n^2)$ to $O(n\log n)$ can be lowered.

2.3.5 The Ring Variant Systems - RLWE

We summarize the two optimizations and derive the following *Ring* variant of Regev's encryption scheme over the ring $\mathbb{R}_q = \mathbb{Z}_q[x]/(x^n+1)$, with $m' = m/n$ and $l = n$:

KeyGen. Secret key $sk = \mathbf{s} \xleftarrow{r} \mathbb{R}_q$, public key $pk = (\mathbf{A} \xleftarrow{r} \mathbb{R}_q^{m'\times 1}, \vec{\mathbf{p}} = \mathbf{As} + \vec{\mathbf{e}} \mod q)$, with $\vec{\mathbf{e}} = [\mathbf{e_1}, \ldots, \mathbf{e_{m'}}] \xleftarrow{r} \chi_{\alpha q}^n$. The length of the public key is now $O(n\log^2 q) = O(\lambda\log^2\lambda)$ bits, or 'quasi-linear' in security λ.

Encryption. For a plaintext $\mathbf{m} \in \mathbb{R}_t$, return ciphertext $C = (\mathbf{a} = \mathbf{rA}, \mathbf{c} = \mathbf{rp} + \lceil \mathbf{q/t} \rfloor.\mathbf{m} \mod \mathbf{q})$. Note that the ciphertext expansion ratio is now $O(\log\lambda)$ and the encryption time (with NTT method) also reduces to 'quasi-linear': $O(\lambda\log^2\lambda)$.

Decryption. Given a ciphertext $C = (\mathbf{a}, \mathbf{c})$, compute $\mathbf{c}' = \mathbf{c} - \mathbf{a}.\mathbf{s}$ and round the result to the neareast multiple of $\lceil q/t \rfloor \mod q$ to get $\mathbf{c''} \in \mathbb{R}_q$. Return the plaintext $\mathbf{m} = \frac{\mathbf{c''}}{\lceil q/t \rfloor} \in \mathbb{R}_t$.

The improvements discussed above (based on the polynomial ring $\mathbb{R}_q$) can also be applied to improve the efficiency of other cryptographic schemes such as the Ajtai's hash function detailed in Definition 9. The idea is again to replace the matrix A by the structured matrix $rot(\vec{a})$.

Definition 15 (Ring variant of Ajtai's Hash Function.)**.** *Given an input* $\mathbf{x} \in \mathbb{R}^{m'}$ *having 'small' coordinates* ($\|\mathbf{x}\| \leq d$)*, select a matrix over* $\mathbb{R}_q$ $A = (\mathbf{a_1}, \ldots, \mathbf{a_{m'}})$ *uniformly random to be the hash function's public key. The output of the function is defined as*

$$g_{q,m,n,d,A}(\mathbf{x}) = A.\vec{\mathbf{x}} = \mathbf{a_1}.\mathbf{x_1} + \cdots + \mathbf{a_{m'}}.\mathbf{x_{m'}} \in \mathbb{R}_q$$

This function is meant to improve the efficiency of $O(n\log n)$ for the key A, the computation is $O(n\log^2 n)$. A practical implementation has been put forward [88] with some further optimizations for a specific set of parameters: $n = 64, m = 16, q = 257$. The hash compresses 1024-bit input to 512-bit output with the key length being 8 kilobits. The performance is competitive with respect to other hash functions, which are about 60 CPU cycles/byte.

Security Impact

We now shift to discussing how security is impacted when switching from the use of a completely random matrix A to structured polynomials, with the aim of improving the efficiency of lattice-based cryptographic schemes. To this end, the hardness of Ring-SIS or Ring-LWE should be compared to the original problems. Much work has been done on this topic. As for us, we will try to show that when choosing the parameters properly, the same level of security can be reached . The definition of the problems follows.

Definition 16 (Decision Ring Learning with Errors (Decision-RLWE))**.** *Given* $q, m, n, \alpha, A \xleftarrow{r} R_q^{m' \times n}$ *and* $\vec{y}$*, distinguish between the following two scenarios:*

- *'Real' Scenario:* $\vec{y} = A.\vec{s} + \vec{e} \mod q$ *(with* $\vec{e} \xleftarrow{r} \chi_{\alpha q}^{m'}$ *and* $\vec{s} \xleftarrow{r} \mathbb{Z}_q^n$*)*
- *'Random' Scenario:* $\vec{y} \xleftarrow{r} \mathbb{Z}_q^m$

The theoretical result is similar to LWE regarding the average-case to worst-case lattice reduction for Ring-SIS/Ring-LWE. [89]. This work proved that, if an efficient algorithm breaking Ring-LWE for random instances over the ring can be found, then it can be used to break $\gamma - SVP$ on some structured sets of lattices, called "ideal lattices". Furthermore, we also have practical evidence to believe that RLWE is hard: The best known attack on RLWE is to reduce it to RSIS, the hardness of RSIS being assessed similarly to the hardness of SIS.

It is important to mention that all the worst-case to average-case reduction results need the polynomial ring R to satisfy some conditions for the connection to hold. The choice of polynomial rings is very important for security: in former attempts, the choice $\frac{\mathbb{Z}[x]}{x^n-1}$ proved to be insecure in some systems, such as the original NTRU.

2.4 Homomorphic Cryptosystems

2.4.1 Homomorphic Encryption

Homomorphic Encryption (FHE) has been considered to be the "Swiss Army Knife" of cryptography, as it provides a powerful tool (operation on encrypted data) to many cryptographic applications: Outsourcing storage and computation, Private Information Retrieval, Multiparties Computation, etc . The idea was introduced in the late 1970s [113] and has been actively researched lately ([123], [130] , [124] [53], etc.), since the breakthrough work of Gentry [49]. This section introduces the concept and discusses some properties of it and its connections with secure computation protocols.

Definition 17 (Syntax)**.** *A homomorphic public key encryption scheme has four procedures:* $\mathbb{E} = (KeyGen, Encrypt, Decrypt, Evalutate)$

- $(sk, pk) \leftarrow KeyGen(1^\lambda, 1^\tau)$ *takes the security parameter* λ *and functionality parameter* τ *and outputs a keypair.*
- $c \leftarrow Encrypt(pk, b)$ *takes a plaintext bit and the public key, outputs a ciphertext.*
- $b \leftarrow Decrypt(sk, c)$ *takes a ciphertext and the secret key, outputs a plaintext bit.*
- $\mathbf{c}' \leftarrow Evaluate(pk, \pi, \mathbf{c})$ *takes a circuit* π*, the public key, and a vector of ciphertext* $\mathbf{c}$*, outputs another vector of ciphertext* $\mathbf{c}'$

The syntax in the definition can be extended naturally to vectors of plaintexts and ciphertexts. We refer to the ciphertext output from the *Encrypt* operation as "fresh ciphertext" and to the outputs from $Evaluate$ as "evaluated ciphertext". There are three circuit families related to the *homomorphic capacity* of $\mathbb{E}$:

Fully Homomorphic Encryption We say that $\mathbb{E}$ is fully homomorphic if it is correct for all boolean circuits. If this is the case, we can ignore the parameter τ ($\tau = 1$).

Leveled/Somewhat Homomorphic Encryption $\mathbb{E}$ is considered somewhat homomorphic if it is correct for families of circuit of depth up to τ.

Additive Homomorphic Encryption We say that $\mathbb{E}$ is additively homomorphic if it is correct for families of circuits made up of only XOR gates. Here we also ignore the parameter τ.

2.4.2 Somewhat Homomorphic Encryption

Although the idea of Fully Homomorphic Encryption (arbitrary number of operations) is feasible, its performance has not been considered practical enough. In this project, we focus on Somewhat Homomorphic Encryption (SHE) systems: the BV system by [27] and the BGV system [26], which allow for additions and some levels of multiplications on the ciphertexts. They serve our purpose in privacy preserving authentication contexts. The security of these cryptosystems is based on the hardness of the Ring-Learning With Error (RLWE) problem [89]. This section briefly discusses such systems.

SHE Scheme construction The BV cryptosystem works as follows.

Setup. Initiate (n, m, q, t, χ) to define the ciphertext space R_q, the plaintext space $R_t = \frac{\mathbb{Z}_t[x]}{x^n+1}$, and the error distribution, note that $t << q$.

KeyGen. The secret key sk can be chosen by selecting a small element $\mathbf{s} \in R_q$, $\mathbf{s} \xleftarrow{r} \chi^n$ can be done for this step. The public key pk is a pair of ring elements $(\mathbf{p_0}, \mathbf{p_1})$, where $\mathbf{p_1} \xleftarrow{r} R_q$ and $\mathbf{p_0} = -(\mathbf{p_1 s} + t\mathbf{e})$ with $\mathbf{e} \xleftarrow{r} \chi^n$.

Encryption. Given a plaintext $\mathbf{m} \in R_t$ and a public key $pk = (\mathbf{p_0}, \mathbf{p_1})$, the encryption first calculates $\mathbf{u}, \mathbf{f}, \mathbf{g} \xleftarrow{r} \chi$, then computes a fresh ciphertext by

$$Enc_{pk}(\mathbf{m}) = (\mathbf{c_0}, \mathbf{c_1}) = (\mathbf{p_0 u} + t\mathbf{g} + \mathbf{m}, \mathbf{p_1 u} + t\mathbf{f})$$

By convention, $[\![P]\!]$ denotes the encryption of a plaintext P under the BV scheme with the public key, and randomness is not taken into account. When the noise used in the encryption has to be specified, $[\![(P,e)]\!]$ is used to represent it, e being the noise.

Decryption. Although the above encryption generates ciphertexts of 2 elements only in R_q, the homomorphic operations (discussed next) will make the ciphertext longer. The decryption of the ciphertext appears as $c = (\mathbf{c_0}, \mathbf{c_1}, \ldots, \mathbf{c_L})$, with secret key $sk = (1, \mathbf{s}, \mathbf{s^2}, \ldots, \mathbf{s^L})$ as $Dec(\mathbf{c}, sk) = \Big[[\langle \mathbf{c}, \mathbf{sk} \rangle]_Q\Big]_t$.

Homomorphic Operations. Given 2 ciphertexts $c = (\mathbf{c_0}, \mathbf{c_1}, \ldots, \mathbf{c_L})$ and $c' = (\mathbf{c'_0}, \mathbf{c'_1}, \ldots, \mathbf{c'_K})$, before any operations are carried out, the ciphertexts are padded with zeros first if necessary (to make $K = L$, assuming that, initially, $K < L$). The homomorphic addition $add(c, c')$ is computed by the component wise addition $add(c, c') = (\mathbf{c_0} + \mathbf{c'_0}, \ldots, \mathbf{c_L} + \mathbf{c'_L})$. The homomorphic multiplication $mult(c, c')$ is computed by $mult(c, c') = (\hat{\mathbf{c}}_\mathbf{0}, \hat{\mathbf{c}}_\mathbf{1}, \ldots, \hat{\mathbf{c}}_{\mathbf{2L-2}})$ with $\sum_{i=0}^{2L-2} \hat{\mathbf{c}}_\mathbf{i} z^i = \sum_{i=0}^{L-1} \mathbf{c_i} z^i \times \sum_{j=0}^{L-1} \mathbf{c'_j} z^j$, where z denotes a symbolic variable.

2.4.3 Ciphertext packing

Given a bit string plaintext $m \in \{0,1\}^*$, there are several ways to encode it as a polynomial, or a ring element $\mathbf{m} \in \mathrm{R}_t$ before encryption. A recent popular approach for BV cryptosystems is called CRT packing method, and based on the Chinese Remainder Theorem [123]. The method allows Single Instruction, Multiple Data (SIMD) operations on encrypted data. However, we do not use this packing technique, as there is not yet a known efficient method to compute HD based on it. We instead apply the method of [133], which is an extension of [95], the technique allowing HD computation with just one level of multiplication. The definition follows.

Definition 18. *For* $\mathbf{T} = (t_0, \ldots, t_{n-1})$ *and* $\mathbf{Q} = (q_0, \ldots, q_{n-1})$*, two types of polynomials are defined in the ring* R_Q *of the SHE scheme:* $pm_1(\mathbf{T}) = \sum_{i=0}^{n-1} t_i x^i$ and $pm_2(\mathbf{Q}) = -\sum_{j=0}^{n-1} q_j x^{n-j}$. *The two types of packed ciphertexts are defined as* $[\![pm_1(\mathbf{T})]\!]$ and $[\![pm_2(\mathbf{Q})]\!]$

If $x^n = -1$ in the ring R_Q, then, when $pm_1(\mathbf{T})$ is multiplied by $pm_2(\mathbf{Q})$, the constant term of the result equals the inner product $\langle \mathbf{T}, \mathbf{Q} \rangle$. Homomorphic multiplication of the ciphertexts produces the ciphertext of the inner product similarly. Furthermore, these results can be used to compute HD as follows, this operation costing one level of multiplication with 3 additions and 3 multiplications on ciphertexts:

Theorem 7 ([133])**.** *Let* $C_1 = -\sum_{i=0}^{n-1} x^{n-i}$ *and* $C_2 = 2 - C_1 = \sum_{i=0}^{n-1} x^i$*. Let* $Enc(HD)$ *be a ciphertext provided by*

$$[\![pm_1(\mathbf{T})]\!] * [\![C_1]\!] + [\![pm_2(\mathbf{Q})]\!] * [\![C_2]\!] - 2 * [\![pm_1(\mathbf{T})]\!] * [\![pm_2(\mathbf{Q})]\!]$$

Then, the constant term of $Dec(Enc(HD))$ *discloses the Hamming Distance of* $\mathbf{T}$ *and* $\mathbf{Q}$*.*

2.5 Zero Knowledge Proof Systems

2.5.1 Definitions

The Zero Knowledge Proof (ZKP), first introduced by [59], is a strong cryptographic tool, a beautiful concept that goes beyond the limits of traditional proofs: In a ZKP system, a *Prover* P convinces a $Verifier$ P that some statement is true without disclosing any information other than the validity of the assertion. This section reviews some standard definitions of ZKP and a specific system (Stern-ZKP) serving as the basic tool in the later constructs of this thesis.

Traditionally, proofs are static objects that can be expressed in written form so that they can be verified. However, in non-secure environments, when the validity of a proof statement is verified, information handled by the protocol having produced the proof can sometimes be gained by the verification agent. To prevent leakages of this kind, interactive proof systems work differently. The protocol involves message exchanges between a prover and a verifier and a final "Accept" bit output issued by the verifier, if convinced that the proof statement is true (otherwise, he outputs “Reject”). We here focus on a specific category of ZKP: $\sum$-protocol, where generally 3 messages are exchanged in every round of the protocol. The prover first sends a “commitment” to the verifier, who will send back a random “challenge”. The prover then computes and sends a “response”, so that the verifier can check the validity of the statement using a function of the received message. In each round, a dishonest prover may guess the verifier’s challenge to prepare the proper response with some constant success probability. Notwithstanding, the protocol can be repeated many rounds to reduce the probability of guessing correctly all the challenges to a negligibly small value. More formally, an interactive proof system for a language $L \subset \{0,1\}^*$ is defined as follows:

Definition 19 (Interactive Proof System)**.** *An interactive proof system for a language $L \subset \{0,1\}^*$ with soundness error $s \in [0,1]$ is a two-party protocol $\langle P,V \rangle$, involving a PPT $verifier$ V and a computationally unbound prover P, which satisfies the following properties:*

- *Completeness:* $\forall x \in L$: $Pr[P(x) \leftrightarrow V(x) = 1] = 1$
- *Soundness:* $\forall x \notin L$: $Pr[P(x) \leftrightarrow V(x) = 1] \leq s$

Where $P(x) \leftrightarrow V(X)$ denotes the output of V after interacting with P on input x.

An interactive proof is said to be $zero-knowledge$ if after reading the proof, the verifier cannot acquire any information from the messages, except for the fact that $x \in L$. This intuition is formulated by the concept of *simulator*. Given $x \in L$, if there exists an efficient *simulator* capable of producing the transcript (the view) of the communication between the prover and the verifier, then it can be certified that the interaction does not provide any additional knowledge for the verifier. There are different definitions of $zero-knowledge$ depending on the security model at hand. For instance, ZK can be defined with respect to an *honest verifier model* or to a *malicious verifier model*, the ZK property can be either *perfect, statistical or computational*. In this thesis, we are interested in *statistical* ZK proofs.

Definition 20 (Statistical ZKP)**.** *An interactive proof system $\langle P,V \rangle$ for a language $L \subset \{0,1\}^*$ is considered to be statistical zero knowledge if for any PPT verifier $\hat{V}$, there exists a PPT simulator S, such that $\forall x \in L$, and the following conditions hold*

- $Pr[S^{\hat{V}}(x) = \perp] \leq 1/2$
- $View_{\hat{V}}[P(x) \leftrightarrow \hat{V}(x)] \approx S^{\hat{V}}(x)$

Where $View_V[P(x) \leftrightarrow V(x)]$ denotes the view of V on x, or all the messages sent from P to V, and $S^{V(x)}$ denotes the output distribution of S having black box access to V on input x, conditioned on $S(x) \neq \perp$. We can also say that the simulator S is required to output a transcript that is statistically blind to the real interaction with probability of at least 1/2.

Proof of Knowledge. In interactive proof systems defined above, if the statement is false then no prover can get accepted. However, if the statement is true, it is not clear that if *anyone* can give accepted proof. It is therefore desirable to have a proof such that allows a *Prover* to

convince the *Verifier* that it indeed knows why a statement is true. Such proof systems are called *Proof of Knowledge* [59, 14]. To define this notion, we first recall the definition of NP-relations.

Definition 21 (NP-relation)**.** *An NP-relation $R \subseteq \{0,1\}^* \times \{0,1\}^*$ is given by a deterministic algorithm $W(\cdot,\cdot)$ that runs in time polynomial of its first input. The relation is $R = \{(x,w) : W(x,w)$ accepts$\}$. The associated NP-language L_R is $L_R = \{x : \exists w$ s.t. $W(x,w)$ accepts$\}$. The witness set for an $x \in \{0,1\}^*$ is defined as $R(x) = \{w : W(x,w) = 1\}$.*

In order to formulate the intuition of the *Prover* knowning a satisfying witness, the concept of *knowledge extractor* is introduced: if there exists a prover $\hat{P}$ who can convince the verifier with some probability on a statement $x \in L_R$, then it is possible to extract from $\hat{P}$ a valid witness for x with a related probability

Definition 22 (Proof Of Knowledge)**.** *An interactive proof system $\langle P,V \rangle$ is a proof of knowledge for a NP-relation with knowledge error $k \in [0,1]$ if there exists a PPT knowledge extractor K, such that for any $x \in L_R$, and for any (even unbounded) $\hat{P}$, for which $p = \Pr\left[out_V[\hat{P} \leftrightarrow V(x)] = 1\right] > k$, then we have*

$$\Pr\left[K^{\hat{P}} \in R_x\right] \geq \mathsf{poly}\,(p-k)$$

Informally, we can say he probability that K ouputs a valid witness for x using the accesss to $\hat{P}$ is at least polynomially related to the probability that $\hat{P}$ convincing the verifier on x (less some knowledge error).

2.5.2 ZKPoPK and ISIS problem

There are several types of ZKP, which are the building blocks of many cryptographic protocols (anonymous credential systems, identification schemes, group signatures, etc). In this work, we focus on ZKP of knowledge (ZKPoK) ([14], [59]), where P needs to also convince V

that he knows a "witness" for the given statement. We then apply this proof to constrain the user to follow the authentication protocol transcript, and therefore claim that the protocol is secure against malicious clients. ZKPoK has been actively studied in the last 30 years ([43], [109], [94], [85]), we focus our work on techniques to implement ZKPoK for an important hard-on-average problem in lattice-based cryptography: the Inhomogeneous Small Integer Solution (ISIS) problem. The proof relation is

$$R_{ISIS_{n,m,Q,\beta}} = \{((\mathbf{A},\vec{y}),\vec{x}) \in \mathbb{Z}_Q^{n\times m} \times \mathbb{Z}_Q^n \times \mathbb{Z}^m : (\|\vec{x}\|_\infty \leq \beta) \wedge (\mathbf{A}\vec{x} = \vec{y} \mod Q)\}$$

The secret witness of P is $\vec{x}$ and the public parameters for V are $(\mathbf{A},\vec{y})$. One of the main research directions was initiated by Stern ([125]), who proposed a ZKPoPK for a simpler problem (Syndrome Decoding Problem). Ling et al. ([85]) developed a protocol to fully support ISIS proofs, i.e., the R_{ISIS} relation. The proof is a 3-move interactive protocol : P starts the protocol by computing and sending three commitments to V; V then sends P a random challenge; P reveals two of the three commitments according to the challenge. The *Prover*'s witness is the secret vector $\vec{x}$, the public inputs are $\mathbf{A}$ and $\vec{y}$. We refer readers to [85] for correctness and statistical zero-knowledge proofs. As per their results, each round of communication costs $\log\beta\tilde{O}(n\log Q)$ bits, and we denote the whole run **SternExt(A,x,y)**, which is discussed as follows.

2.5.3 SternExt Protocol

We first recall some related concepts of commitment scheme, which is used widely in zero knowledge proofs. Commitment scheme is a primitive that allows a sender to commit to some information in advance, a verifier then specifies a choice of what it wants to learn, the sender can choose to reveal the value corresponding to the verifier's choice. A Commitment scheme works in two phases: committing and revealing phase. In the first phase, the sender

that wants to commit a value s, computes $\mathbf{c} = COM(s,\rho)$, where ρ is some random factor, $\mathbf{c}$ is sent to the receiver. In the second phase, the sender reveals s and the random ρ, the verifier can compute $COM(s,\rho)$ to compare with $\mathbf{c}$ to check the validity of $\mathbf{c}$. Generally, secure commitment scheme should satisfy two requirements: statistical hiding and computational binding.

Definition 23. *A statistical hiding and computational binding string commitment scheme is a PPT algorithm $COM(s,\rho)$ satisfying:*

- $\forall s_o, s_1 \in \{0,1\}^* : COM(s_0;\cdot) \approx_s COM(s_1;\cdot)$
- *For any PPT algorithm A returning $(s_0,\rho_0);(s_1,\rho_1)$ given $s_0 \neq s_1$, we have*

$$\Pr\left[COM(s_0,\rho 0) = COM(s_1,\rho_1)\right] = \mathsf{negl}\,(n)$$

In other words, a commitment scheme is statistical hiding if any computationally unbounded cheating verifier cannot distinguish the commitments of two different strings. The scheme is computationally binding if any PPT cheating sender cannot change the committed string after the first phase. There are some constructions built from collision resistant hash functions [37, 62]. The **SternExt(A,x,y)** protocol includes 2 phases

Setup. Let COM be a statistical hiding and computational biding commitment scheme ([76] show that such scheme can be constructed based on the hardness of ISIS problem). Before the interaction, P and V create matrix $\mathbf{A}' \in \mathbb{Z}_Q^{n\times 3m}$ by extending $\mathbf{A}$ (padding $\mathbf{A}$ with $2m$ zero-columns). P also executes some extra preparation steps:

- Decomposition: Represent vector $\vec{x} = (x_0,\ldots,x_{m-1})$ by k vectors $\vec{u'}_j \in \{-1,0,1\}^m$. The algorithm first breaks each x_i into its binary representation: $x_i = b_{i,0}2^0 + b_{i,1}2^1 + \cdots + b_{i,k-1}2^{k-1}$, where $b_{i,j} \in \{-1,0,1\}$. Then $\vec{u'}_j$ is constructed by $\vec{u'}_j = (b_{0,j}, b_{1,j}, \ldots, b_{m-1,j})$. Note that $\vec{x}$ can be reconstruct by $\vec{x} = \sum_{j=0}^{k-1} 2^j \vec{u'}_j$.

- Extension: This step masks the number of -1s,1s and 0s in each $\vec{u'}_j$ by transforming each $\vec{u'}_j \in \{-1,0,1\}^m$ into another $\vec{u}_j \in \{-1,0,1\}^{3m}$. The mask is built by padding each $\vec{u}_j$ with a random vector $\vec{t} \in \{-1,0,1\}^{2m}$, such that after the masking is completed, the number of -1s, 0s, and 1s in $\vec{u}_j$ is the same and equals m. We observe that

$$\mathbf{A}' \sum_{j=0}^{k-1} 2^j \vec{u}_j = \vec{y} \mod Q \iff \mathbf{A}\vec{x} = \vec{y} \mod Q$$

The interactive proof system. The *Prover* P ad the *Verifier* V interact as follows

1. **Commitment.** P samples $\vec{r_0}, \vec{r_1}, \ldots, \vec{r_{k-1}} \xleftarrow{r} \mathbb{Z}_Q^{3m}$, $\pi_0, \ldots, \pi_{k-1} \xleftarrow{r} \mathscr{S}_{3m}$, where $\mathscr{S}_k$ denotes the symmetric group of all permutations of k elements. P sends the commitments to V:

$$\begin{cases} \mathbf{c_1} = COM(\pi_0, \ldots, \pi_{k-1}, \mathbf{A}' \sum_{j=0}^{k-1} 2^j \vec{r}_j) \\ \mathbf{c_2} = COM(\pi_0(\vec{r_0}), \ldots, \pi_{k-1}(\vec{r}_{k-1})) \\ \mathbf{c_3} = COM(\pi_0(\vec{u}_0 + \vec{r}_0), \ldots, \pi_{k-1}(\vec{u}_{k-1} + \vec{r}_{k-1})) \end{cases}$$

2. **Challenge.** After receiving the commitments $\mathbf{c_i}$, V sends a challenge $Ch \xleftarrow{r} \{1,2,3\}$ to the *Prover* P.

3. **Response.** P replies as follows:

   - If $Ch = 1$, P reveals $\mathbf{c_2}$ and $\mathbf{c_3}$: For each j, let $v_j = \pi_j(\vec{u}_j)$, and $w_j = \pi_j(\vec{r}_j)$ and sends back $RSP = (v_0, \ldots, v_{k-1}, w_0, \ldots, w_{k-1})$.
   - If $Ch = 2$, P reveals $\mathbf{c_1}$ and $\mathbf{c_3}$: For each j, let $\phi_j = \pi_j$, $\vec{z}_j = \vec{u}_j + \vec{r}_j$ and sends back $RSP = (\phi_0, \ldots, \phi_{k-1}, \vec{z}_0, \ldots, \vec{z}_{k-1})$.
   - If $Ch = 3$, P reveals $\mathbf{c_1}$ and $\mathbf{c_2}$: For each j, let $\psi_i = \pi_j$, $\vec{s}_j = \vec{r}_j$ and sends back $RSP = (\psi_0, \ldots, \psi_{k-1}, \vec{s}_0, \ldots, \vec{s}_{k-1})$.

4. **Verification.** Receiving the response *RSP*, V engages in the following protocol:

   - If $Ch = 1$, check that $\vec{v}_j \in \{-1,0,1\}^{3m}$ and

$$\begin{cases} \mathbf{c_2} = COM(w_0, \ldots, w_{k-1}) \\ \mathbf{c_3} = COM(v_0 + w_0, \ldots, v_{k-1} + w_{k-1}) \end{cases}$$

   - If $Ch = 2$, check that

$$\begin{cases} \mathbf{c_1} = COM(\phi_0, \ldots, \phi_{k-1}, \mathbf{A}' \sum_{j=0}^{k-1} 2^j \vec{z}_j - y) \\ \mathbf{c_3} = COM(\phi_0(\vec{z}_0, \ldots, \phi_{k-1}(\vec{z}_{k-1})) \end{cases}$$

   - If $Ch = 3$, Check that

$$\begin{cases} \mathbf{c_1} = COM(\psi_0, \ldots, \psi_{k-1}, \mathbf{A}' \sum_{j=0}^{k-1} 2^j \vec{s}_j) \\ \mathbf{c_2} = COM(\psi_0(\vec{s}_0), \ldots, \psi_{k-1}(\vec{s}_{k-1})) \end{cases}$$

   In each case, V outputs *Accept* if and only if all the conditions hold. Otherwise, V outputs *Reject*. Also note that all the computations are done in $\mathbb{Z}_Q$.

Theorem 8 ([85]). *The **SternExt** proof system is a statistical ZKPoK for the relation R_{ISIS}, with perfect completeness and knowledge error 2/3.*

It was proved in [85] that:

- If the honest prover follows the protocol, it is always accepted.
- If COM is statistical hiding, then a simulator (with black box access to a cheating verifier) can be constructed that produces a protocol transcript with accept probability 2/3 and is statistically close to the transcript produced by the real communication.

- If COM is computationaly binding, and if there exists a cheating prover that can pass the proof with probability higher than 2/3 then one can use it to construct a knowledge extractor that outputs $\mathbf{x}'$ such that $((\mathbf{A},\mathbf{y}),\mathbf{x}') \in R_{ISIS}$

In summary, throughout this chapter we have reviewed basic definitions and constructions in lattice-based cryptography, as well as some notions of secure multi-parties protocols that are used in the thesis. In the next chapter, we consider the definitions of the models and security of a generic biometric authentication scheme. We then present different variants of privacy preserving authentication protocol according to the models.

Chapter 3

Security model

3.1 Introduction

In this chapter, we first describe the protocol and its security model generically. We then go into details and further use it as a framework for our specific proposals in later chapters. In order to precisely grasp the definition of security, we will describe the protocols' security models in terms of **attack games** played between two parties: a **challenger** and an **adversary**, adapting the standard game based security model approach commonly used in cryptography [25] . Generally, the challenger $\mathscr{C}$ follows a simple, fixed protocol and the adversary $\mathscr{A}$ may follow an arbitrary and efficient protocol. The two players send messages back and forth to each other as specified in their protocols, and, at the end of the games, $\mathscr{A}$ outputs some value. We use *"simulation based"* security models, where the attack game defines a probability space based on the output value , and it defines the adversary's *advantage*, which normally measures the difference between the probabilities of two events in the given probability space: One is the event that the adversary wins the *real* game; the other one is the event that a related adversary wins an ideal game with one (or some) of the system's components replaced by an idealized version of such component(s). By demonstrating that the advantage between the real and ideal games is negligible, we

can conclude that the security of the "real" protocol is essentially the same as the "ideal" protocol. The security proofs of protocols will also be organized as sequences of games. This technique appears in the literature in different flavours with different degrees of formalisation. We also use an incremental game based proof approach [122] as a helpful tool to reduce the complexity of security proofs that might otherwise become messy and complicated. We note that the technique is only a tool to organize the models and the proofs, the actual ideas for cryptographic constructions and security analysis come from elsewhere, specifically the problems discussed in the previous chapter.

3.1.1 Desirable properties

According to [72], a secure template solution to biometric authentication should satisfy the following properties:

Non-invertibility: It is computationally hard to rebuild an original template from an encrypted one.

Non-linkability (Revocability): It should be possible to revoke and to re-issue new encrypted templates using a new key when the database is compromised.

Discriminability: The secure scheme should not degrade the accuracy of the biometric authentication system.

3.1.2 The threats

There are common security threats to many authentication systems such as Trojan horse, replay, man-in-the-middle (MITM) attack. Biometrics authentication systems are also vulnerable to such attacks. We can borrow ideas from secure password-based schemes to address such issues. However, there are two categories of vulnerabilities that are specific to biometric systems. The first one is impersonation, or spoofing. The attack happens at the

client's side as the adversary tries to cheat the system with counterfeit or invalid inputs. The other issue is at the server's side and yields privacy concerns.

We will present formal models that capture all the known threats of biometric authentication:

- Confidential data leakage due to attacks on the template database: Server breaches of biometric data always have catastrophic consequences ([OPM]), as we cannot change our fingerprints as easily as changing our passwords.

- Hill-Climbing attack from the server's side: This is a privacy threat where the server tries to compute good inputs X from the distance information between X and Y ([127], [64]). Note that Hill-Climbing attack by the client is very limited due to the limitation of the number of false authentication attempts in almost every biometric authentication system.

- Impersonation by a malicious client when the secret key of the user is known (replay attack): This happens when an attacker has access to the user's device but not to his biometric data, as, for example, in stolen device scenarios ([134]).

- Impersonation by a malicious client when the biometric template of the user is known (spoofing attack): This happens when an attacker collects biometric data and tries to reconstruct the template for authentication, as, for example, by rebuilding fingerprints from captured photos ([134],[44]).

- Cross matching of biometric data among databases: This threat uses information of the same user from different compromised databases to reconstruct the biometric template.

3.2 The generic model

Entities: There can be 2 or 3 typical entities involved in a secure biometric authentication system. The user $\mathscr{U}$, an authentication server $\mathscr{S}$, and a decryptor, who is a third party trusted by both the user and the server. The decryptor appears in some systems ([90], [65], [64]), with the assumption that there is no collusion between this entity and $\mathscr{U}$ or $\mathscr{S}$. In our work, we avoid the assumption of a trusted decryptor party, so that only two parties are left: $\mathscr{U}$ and $\mathscr{S}$.

Biometrics Features in non-private setting In biometric authentication systems (e.g., fingerprint authentication system), a user $\mathscr{U}$ first registers his fingerprint template X on the server $\mathscr{S}$. $\mathscr{U}$ later authenticates to $\mathscr{S}$ using the same finger with a template Y, $\mathscr{S}$ uses an algorithm $Verify(X,Y)$ to obtain the result of the authentication: **Accept** or **Reject**. Different fingerprint systems might use different features of fingers such as minutia or fingercode [45, 73] to compute the distance Δ between X and Y within the algorithm $Verify$. The distance Δ is compared to some predefined threshold value τ to determine the result of the authentication. We refer the reader to [70] for biometric feature extraction and comparison techniques.

Unlike password based systems, where $\mathscr{U}$ always uses one and the same query for many authentication trials, all biometric systems have the concept of False Acceptance Rate (FAR) (for the system **Accept**ing an incorrect template), and False Rejection Rate (FRR), (for the system **Reject**ing a genuine one). As balancing these 2 rates while keeping good performances is one of the main challenges that fingerprint verification algorithms [FVC] need to face, we also reflect these two rates in our models.

Algorithms and Procedures in privacy-preserving settings: We describe the high-level syntax of the privacy-preserving biometric authentication protocol consisting of the following procedures:

Enrol. This procedure inserts records into the server's database.

- Client's input: identity k, a registered template X_k
- Server's input: Parameters of the cryptographic tools used.
- Output: A public-private key pair (sk_k, pk_k) for the user $\mathscr{U}_k$. The server stores the protected biometric template T_k (which is typically some encryption of X_k).

Auth. This procedure allows a user to authenticate to the system.

- Client's Input: identity k, a query template Y_k and the secret key sk_k
- Server's Input: record (k, T_k, pk_k)
- Output: The server computes the authentication result $res = \{$**Accept,Reject**$\}$. The transcript of the protocol can also be obtained: We denote V_c and V_s to be the *view* of the client and the server received from the protocol execution.

Correctness Requirement: A genuine user $\mathscr{U}_k$ executes $(sk_k, T_k) \leftarrow$ **Enroll**(k, X_k) using a $X_k \xleftarrow{r} Supp(D_k)$ and later uses his biometric template $Y_k \xleftarrow{r} D_k$ to perform

$$res \leftarrow \textbf{Auth}_{\mathscr{U},\mathscr{S}}((k, Y_k, sk_k), (k, T_k, pk_k))$$

Where $\mathscr{U}, \mathscr{S}$ are the honest client and server parties. The privacy-preserving protocol works correctly if the FRR under this system is exactly equal to the FRR of the non-privacy preserving system:

$$Pr[res = verify(X_k, Y_k)] = 1$$

3.2.1 The privacy preserving model

Our security model intends to capture both threats for client and server's sides as discussed previously. There is only one variant which do not capture a threat related to Hamming

Distance leakage to server, this imperfect privacy was introduced with the assumption that given such a value of HD of 2 bit strings, it is not feasible to infer information from the original bit strings.

Privacy against an Honest But Curious server: The security model is defined in terms of the following security games.

The real game $\mathbf{Real}_{\mathscr{A}}(D_k, X_k)$: This is the game for a privacy attack against the privacy-preserving protocol for the underlying biometric system, between an attacker $\mathscr{A}$ and a challenger $\mathscr{C}$. The input of the game is an attacked $\mathscr{U}_k$ biometric distribution $D_k \in D_{bio}$ and a user template $X_k \in Supp(D_k)$.

1. $\mathscr{C}$ runs $(T_k, sk_k) \leftarrow \mathbf{Enrol}(k, X_k)$ and sends T_k to $\mathscr{A}$.
2. For $i = 1 \ldots q$:
   - $\mathscr{C}$ samples $Y_i \xleftarrow{r} D_k$
   - $\mathscr{C}$ simulates the **Auth** protocol, playing the roles of both the client and the server:
   $$res \leftarrow \mathbf{Auth_i}((k, Y_i, sk_k), (k, T_k, pk_k))$$
   - $\mathscr{C}$ sends the view V_s^i to $\mathscr{A}$.
3. $\mathscr{A}$ outputs a bit β, representing some information that $\mathscr{A}$ has discovered about (D_k, X_k). The game output is $\mathbf{Real}_{\mathscr{A}}(D_k, X_k) = \beta$.

The ideal game $\mathbf{Ideal}_{\mathscr{A}'}(D_k, X_k)$: This is the game for a privacy attack against an ideal privacy scenario for the underlying biometric authentication system, where an attacker $\mathscr{A}'$ interacts with a challenger $\mathscr{C}'$. The input of the game is an attacked $\mathscr{U}_k$ biometric distribution $D_k \in D_{bio}$ and a user template $X_k \in Supp(D_k)$. In this ideal game, the information $\mathscr{A}'$ can obtain about (D_k, X_k) is the value of HD_{X_k, Y_k} and the bit $Verify(X_k, Y_i)$.

1. For $i = 1 \ldots q$:
   - $\mathscr{A}'$ chooses query template $Y_i \in \{0,1\}^n$ and sends it to $\mathscr{C}'$.
   - $\mathscr{C}'$ sends HD_{X,Y_i} to $\mathscr{A}'$.
2. $\mathscr{A}'$ output a bit β', representing some information that $\mathscr{A}'$ has gathered about (D_k, X_k). The game output is $\mathbf{Ideal}_{\mathscr{A}'}(D_k, X_k) = \beta'$.

Let W_r be the event that $\mathscr{A}$ outputs $\beta = 1$ in the real game and let W_i be the event that $\mathscr{A}'$ output $\beta' = 1$ in the ideal game. We define $\mathscr{A}$'s **passive server security advantage** with respect to protocol $\mathscr{P}$ as

$$\mathsf{Adv}^{\mathrm{ps}}_{\mathscr{A},\mathscr{P}}(n) = |Pr[W_r] - Pr[W_i]|$$

Definition 24 (Privacy Security against Server)**.** *We say that a biometric authentication protocol is q-private in the sense of biometric template privacy against an honest but curious server $\mathscr{S}$ if for every efficient real-game attacker $\mathscr{A}$, there exists an efficient ideal-game attacker $\mathscr{A}'$ such that, for all (D_k, X_k), with $X_k \in Supp(D_k)$, the following stands:*

$$\mathsf{Adv}^{\mathrm{ps}}_{\mathscr{A},\mathscr{P}}(n) \leq negl(n).$$

Figure 3.1 and 3.2 show schematic diagrams of the Privacy Security against Server games.

At several points of this thesis, we will refer to the terms "efficient" and "negligible" (which will be defined formally in a section below). Intuitively, *negligible* means so small as to be "zero for practical purposes". For example, consider an event happening with probability 2^{-100}: we would not worry about such an event occurring more than we would worry about a similar event occurring with probability 0. Additionally, an *efficient adversary* is granted to run in a reasonable amount of time. Sometimes we also use two other related terms: A value N is called *super-poly* if $1/N$ negligible; and a *poly-bounded* value is supposed

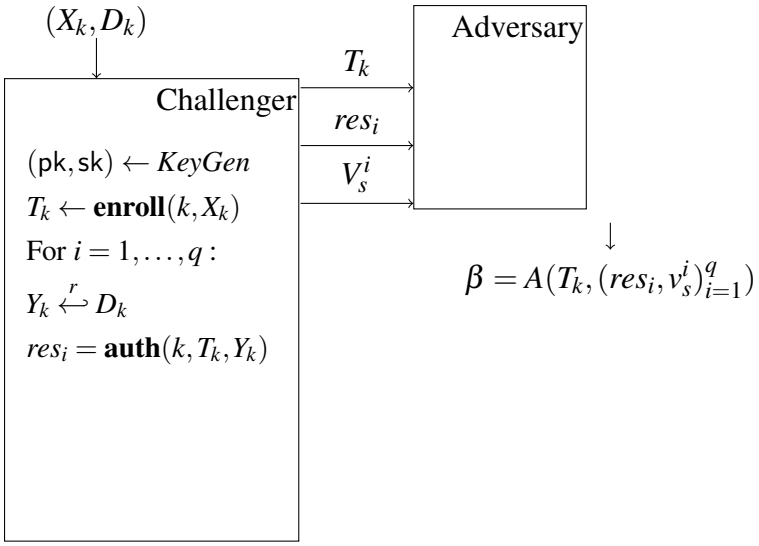


Fig. 3.1 Passive Server Attack Game (Real Game)

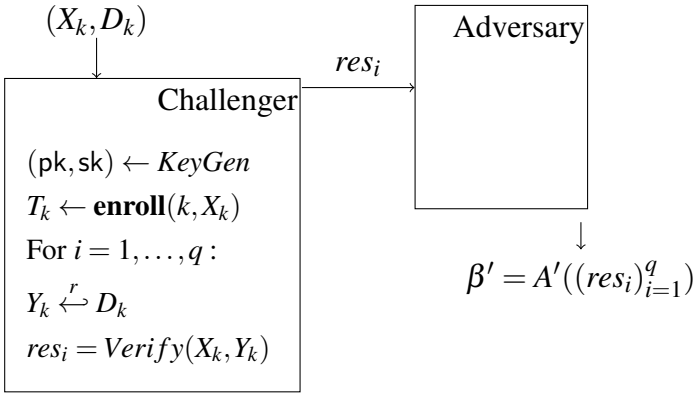


Fig. 3.2 Passive Server Attack Game (Ideal Game)

to be a reasonably sized number (we can say that the running time of any *efficient* adversary is a *poly-bounded* value). We will use the results of the following lemma in the security proofs:

Lemma 6. *If ε and ε' are negligible values, and Q and Q' are poly-bounded values, then*

- *$\varepsilon + \varepsilon'$ is a negligible value*
- *$Q + Q'$ and $Q \cdot Q'$ are poly-bounded values*

- $Q \cdot \varepsilon$ *is a negligible value*

Security against a malicious client: In this work, we aim at security against an active client, where an attacker $\mathscr{A}$ is assumed not to follow the protocol transcript.

Biometric Impersonation Attack Game. FAR is the usual biometric impersonation probability, it is inherent to biometrics itself without any cryptographic protocols. We first discuss this security game (which will be referred to as *biometric impersonation*), we then elaborate the discussion to account for models with privacy-preserving requirements.

Setup: $\mathscr{C}$ samples $X_k \xleftarrow{r} D_k$ from a random $\mathscr{U}_k$ ($D_k \xleftarrow{r} D_{bio}$).

Query: $\mathscr{A}$ is given access to the authentication oracle q times $verify(X_k, Y_i)$, which returns the authentication result of $\mathscr{U}_k$ with a query template Y_i. $\mathscr{A}$ has q attempts to make queries, in each attempt, $\mathscr{A}$ chooses a Y_i by himself and executes $verify(X_k, Y_i)$.

Guess: $\mathscr{A}$ outputs $Y_{q'}$ such that $verify(X_k, Y_{q'}) =$ **Accept**.

Figure 3.3 show the biometric impersonation (non private) attack game. Let W_0 be the event that there exists some i such that $\mathscr{A}$ output Y_i such that $verify(X_k, Y_i) =$ **Accept** and let $p_0 = \Pr[W_0]$.

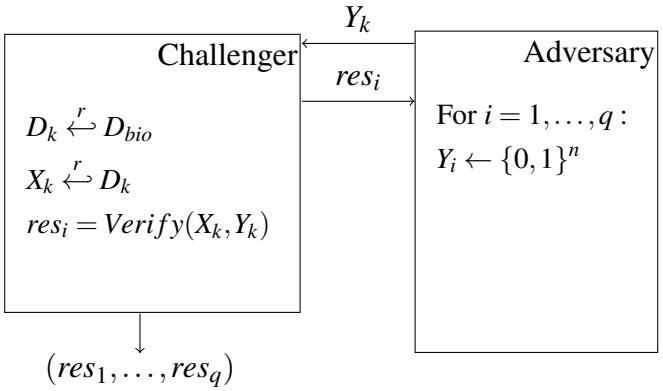


Fig. 3.3 Malicious Client Attack Game in Non-private setting

Impersonation Attack Game. This game captures the basic impersonation attack from a client

Setup: The setup phase includes 2 steps:

- $\mathscr{C}$ samples $X_k \xleftarrow{r} D_k$ from a random $\mathscr{U}_k$ ($D_k \xleftarrow{r} D_{bio}$).
- $\mathscr{C}$ runs **Enroll**(k, X_k), which returns (sk_k, pk_k, T_k).

Query: In the query phase:

- $\mathscr{A}$ is given access to the authentication oracle **Auth**, the input to **Auth** supplied by $\mathscr{A}$ are the client's messages in the **Auth** protocol, returning the authentication result of $\mathscr{U}_k$ with a query template Y.
- $\mathscr{C}$ and $\mathscr{A}$ run **Auth**() q times, For the i^{th} run, $\mathscr{A}$ plays the client's role, $\mathscr{C}$ plays the server's role, replying with $res_i =$ **Auth**.

Guess: $\mathscr{A}$ wins the game if there exists i such that $res_i =$ **Accepted**.

Figure 3.4 show the impersonation attack game. Let W_1 be the event that there exists i such that $res_i =$ **Accepted** and let $p_1 = \Pr[W_1]$, recall that P_0 is the advantage of $\mathscr{A}$ in the biometric impersonation attack game. We define $\mathscr{A}$'s **active impersonation advantage** with respect to the protocol $\mathscr{P}$ as

$$\mathsf{Adv}^{\mathsf{ai}}_{\mathscr{A},\mathscr{P}}(n) = |p_1 - p_0|$$

Definition 25 (Active Client Impersonation.)**.** *A privacy-preserving biometrics authentication protocol $\mathscr{P}$ is c-secure against active client impersonation if for all efficient adversary $\mathscr{A}$, the value $\mathsf{Adv}^{\mathsf{ai}}_{\mathscr{A},\mathscr{P}}(n)$ is not larger than $c \times$ FAR of the non-private protocol.*

Multifactor Attack Game. This model extends the above protocol and captures the client side attacks mentioned in section 3.1.2.

Setup: The setup phase includes 2 steps:

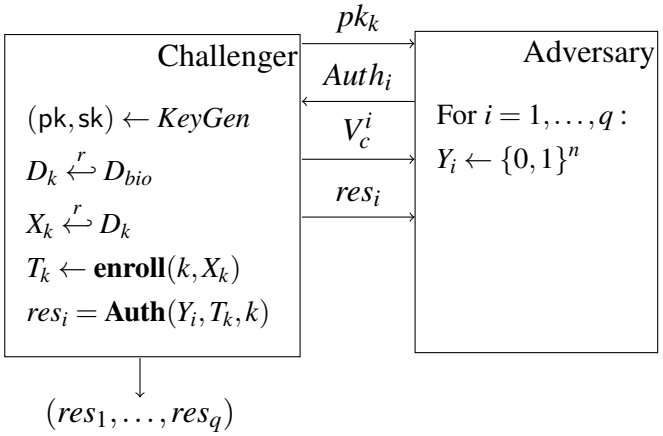


Fig. 3.4 Impersonation Attack Game

- $\mathscr{C}$ samples $X_k \xleftarrow{r} D_k$ from a random $\mathscr{U}_k$ ($D_k \xleftarrow{r} D_{bio}$).
- $\mathscr{C}$ runs **Enroll**(k, X_k), which returns (sk_k, T_k).

Query: In the query phase:

- $\mathscr{A}$ chooses the attack type $t \in \{I, II\}$ which specifies the scenario of key exposure or template exposure.
- $\mathscr{C}$ passes sk_k if $t = I$ or T_k if $t = II$ to $\mathscr{A}$. Note that this model reflects the 2-factors authentication (the secret key and the biometric template), and, thus, if $\mathscr{A}$ requests both factors, he loses the game.
- $\mathscr{C}$ and $\mathscr{A}$ run **Auth**() q times, For the i^{th} run, $\mathscr{A}$ plays the client's role, choosing and sending Y_i to $\mathscr{C}$, $\mathscr{C}$ plays the server's role, replying with $res_i = \mathbf{Auth}(Y_i)$.

Guess: $\mathscr{A}$ wins the game if it outputs Y such that $\mathbf{Auth}(Y) = \mathbf{Accepted}$.

Figure 3.5 show the multi-factors attack game. Let W_2 be the event that there exists i such that $res_i = \mathbf{Accepted}$ and let $p_2 = \Pr[W_2]$. Let p_0 define the winning probability of the $\mathscr{A}$ in the non-privacy biometric authentication game. We define

$\mathscr{A}$'s **active multifactor advantage** with respect to the protocol $\mathscr{P}$ as

$$\mathsf{Adv}^{\mathrm{am}}_{\mathscr{A},\mathscr{P}}(n) = |p_2 - p_0|$$

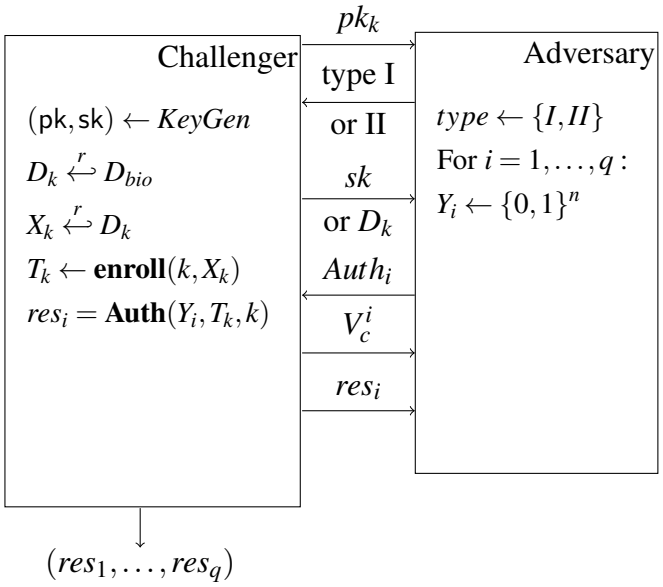


Fig. 3.5 Multi-factors Attack Game

Similar to the previous impersonation attack game, We would want this advantage value not to be too large compared to the non-privacy-preserving biometric impersonation model's advantage, which was bounded by $q \times FAR$.

Definition 26 (Active Multi-factors.)**.** *A privacy-preserving biometrics authentication protocol $\mathscr{P}$ is c-secure against active multi-factors attack if for all efficient adversary $\mathscr{A}$, the value $\mathsf{Adv}^{\mathrm{am}}_{\mathscr{A},\mathscr{P}}(n)$ is not larger than $c \times FAR$ of the non-private protocol.*

3.3 Security Proof Technique

In the security proofs of the protocols appearing in the next chapters, we will typically model the security games following the generic framework defined in this chapter, with the **Enroll**

and **Auth** modules replaced by concrete lattice-based constructions (or combination of them). Each attack game is modeled by way of a probability space, as both *adversary* and *challenger* are probabilistic processes communicating with each other. The definition of security is tied to some particular event S as seen in previous sections (where security means that for all *efficient* adversary, the probability that event S happens is *close to* some target probability, normally *negligibly* small, 1/2, or FAR in one of our contexts). In formal definitions, there is a security parameter related to the terms *efficient* or *negligible*. For example, *efficient* means time bounded by a polynomial in the security parameter, *close to* means the difference is smaller than inverse of any polynomial in the security parameter. For simplicity, we assume that all algorithms, adversaries, challengers, etc., take this parameter value as an implicit input.

The approach used in the security proofs is called *sequence-of-games*, the process is as follows. In each proof, a sequence of games is set up: Game 0, Game 1, ..., Game n, where Game 0 is normally the original attack game with respect to a concrete protocol and a given adversary. Let S_0 be the event S. For i=1,...,n, the construction defines an event S_i for each Game i, these events are normally related to S in a natural way. Then, the proof shows that $\Pr[S_i]$ is *close to* $\Pr[S_{i+1}]$ for all $i = 0,\ldots,n-1$. Finally, if $\Pr[S_n]$ can be shown to be *close to* the "target probability", provided that n is a constant, we can infer that $\Pr[S]$ is negligibly close to the "target probability", and security is proved. We have ensured that the changes from one game i to the next game $i+1$ are kept as small as to avoid major complications while analyzing them.

Generally, there are two types of changes in the proofs:

Change based on indistinguishability A change is made such that its detection by the adversary implies a method of distinguishing two distributions that are statistically or computationally indistinguishable. For instance, if P_1 and P_2 are two indistinguishable distributions, to prove that $|\Pr[S_i] - \Pr[S_{i+1}]|$ is negligible, it should be shown that

there exists an algorithm D such that, when inputting an element drawn from P_1, outputs 1 with probability $\Pr[S_i]$; and when inputting an element drawn from P_2, outputs 1 with probability $\Pr[S_{i+1}]$. It follows that $|\Pr[S_i] - \Pr[S_{i+1}]|$ is negligible, provided the indistinguishability assumption of P_1 and P_2.

Change based on failure events This type of change is based on the fact that, given A, B, F as events in some distribution, and supposing that $A \wedge \neg F \iff B \wedge \neg F$, then $|\Pr[A] - \Pr[B]| \leq \Pr[F]$, to say that two games proceed identically unless F occurs, means that $S_i \wedge \neg F$ and $S_{i+1} \wedge \neg F$ are the same. So, in order to prove that $\Pr[S_i]$ is negligibly close to $\Pr[S_{i+1}]$, it is enough to prove that $\Pr[F]$ is negligible. This can be done using the security assumption or an information-theoretic argument, as, for instance, that when F occurs, the adversary could find a collision in a cryptographic hash function.

3.3.1 Related work

Sequence of transitions based on indistinguishability (only) have been used extensively in cryptography for many years. For example, in 1984, [56] illustrated the "hybrid arguments" technique while constructing pseudo-random functions. [15] was an early example of a proof that is structured as a sequence of games including transitions based on both failure events and indistinguishability. The first formal approach to sequences of games was initiated by Kilian and Rogaway's paper [77]. The authors subsequently applied the technique in numerous papers, detailed introduction to the methodology and references of usage can be found in [16]. In Bellare and Rogaway's approach, games are treated as syntactic objects subject to formal manipulation, while [122] views games as probability spaces with random variables defined over them. This proof style is also illustrated nicely by some public key cryptography examples in an introductory manuscript [107]. The technique has been used widely ([2], [23], [28], etc.). While some of the proofs can be structured differently, sequence-of-games is a technique able of accomplishing the task in a clear and convincing way.

3.4 Conclusion

This chapter lays out the framework to be used as the security models and hints at the way security proofs will be organized. The technique (sequence-of-games) will be used subsequently in the next chapters within each of the protocol variants to illustrate to illustrate correspondingly security properties.

Chapter 4

The First Protocol - Computing HD Homomorphically

4.1 Introduction

Many biometrics privacy preserving authentication protocols rely on a trusted third party to keep the client's secret key used to decrypt the authentication result [90, 65, 129]. Our first attempt to improve the security of such schemes has been to remove the role of such a party. As it proves inconvenient, due to privacy issues, to allow a third party server to keep the client's secret key, the distance plaintext decryption should be performed by the client, which, in turn, raises the following challenges:

Multifactor security A malicious client with access to the compromised secret key of the honest client can derive information about the distance between the registered template and the extracted template while he decrypting the homomorphic ciphertext result sent by the server.

Malicious Client Model Impact It is important for the protocol to be secure against the *active client* model, by which an adversary does not simply follow the protocol

transcript but also tries to draw information from it (this model is also known as *Semi Honest*, *Honest-But-Curious*, or *Passive* Client Model).

The first issue has been solved in previous work of [90] forcing the server to mask the distance between templates by means of some random value before sending it to the client. In this chapter, we discuss the solution to the second issue, where a new Zero-Knowledge-Proof (ZKP) technique for lattice-based cryptosystems is introduced so as to compel the client to prove that the protocol is followed honestly. The technique not only provides a proof of plaintext knowledge, it has another important advantage, it also proves the binary format of the query template in a way that is compatible with the existing efficient plaintext packing technique. We can use the ZKP at two stages: When the client first sends the query in order to start authenticating, and at the final step, proving that the Hamming Distance is correctly decrypted.

4.2 Related Work

In a biometric authentication system, a user $\mathscr{U}$ first registers his biometric template X on the server $\mathscr{S}$. $\mathscr{U}$ later authenticates to $\mathscr{S}$ using the same finger with a template Y, $\mathscr{S}$ uses an algorithm $Verify(X,Y)$ to obtain the result of the authentication: **Accept** or **Reject**. Different biometric systems might use different methods to compute the distance Δ between X and Y within the algorithm $verify$. The distance Δ is compared to some predefined threshold value τ to determine the result of the authentication trial. We refer the reader to [70] for biometric feature extraction and comparison techniques. Unlike password based systems, where $\mathscr{U}$ always uses one and the same query for many authentication trials, all biometric systems implement the concept of False Acceptance Rate (FAR), corresponding to the system's **Accept** answer while receiving an incorrect template; and False Rejection Rate (FRR), where the system's **Reject** answer follows reception of a genuine one. Balancing these 2 rates

while keeping good performances is one of the main challenges that fingerprint verification algorithms [FVC] try to solve. In this work, we assume that biometric data are represented as binary codes and that HD is used to measure the similarity between two of them. We refer the readers to [38] or [Fuj] for examples of 2048-bit iris code generation and HD comparison.

There are three main approaches for privacy-preserving biometric authentication ([72], [13], [71]). In the *Feature transformation approach* (cancelable biometrics or biohashing, such as [126], [33]), the template data are encrypted using a client's key. Being a single factor protection, it has a low security level as the key might be leaked. The *Biometric cryptosystem approach* (fuzzy vault and fuzzy commitment, [128], [96]) is based on error correcting codes and the trade-off between biometric accuracy and security seems to be insufficiently understood . We focus our work on the last approach, *Homomorphic Encryption*, which appears as the best candidate to provide all of the system design requirements mentioned.

The idea was first proposed in 2006 ([118]) using the Paillier addictive homomorphic system ([103]). In 2010, Osadchy et al. ([101]) provided a privacy-preserving feature by combining Paillier system with an oblivious transfer protocol. SCiFI uses 900-bit vector to represent face image data and Hamming Distance (HD) to compare two vectors. [20] developed a similar system for iris and fingerprints but using the DGK cryptosystem [36] and garble circuit technique instead. They represented biometric data as 2048-bit vectors and also used HD for threshold comparison. [133] proposed an approach based on Somewhat Homomorphic Encryption (SHE) [27]. They introduced a ciphertext packing technique to speed up the HD computation operation. There have been variations and improvements over time ([120], [90], etc.). However, most of the protocols are only secure against a semi-honest client, many of them relying on one or more trusted third parties with the client's secret key being used to decrypt the HD.

Table 4.1 Comparison with previous works

	Client Impersonation Security			**Server Privacy Security**		
	Malicious client model	Stolen device, secret key	Leaked Template	Template protection against server	Trusted party required	Hill climbing protection
[39]	No	Yes	Yes	No	No	No
[90]	No	No	Yes	No	Yes	No
[120]	No	Yes	Yes	No	No	Yes
[64]	No	Yes	Yes	Yes	Yes	Yes
[119]	Yes	No	Yes	Yes	Yes	No
This paper	Yes	Yes	Yes	Yes	No	Yes

4.3 The BV Cryptosystem and Yasuda Packing Method

Although the idea of Fully Homomorphic Encryption (arbitrary number of operations) is feasible, its performance has not been considered practical enough. We discussed some generic LWE based cryptosystems and their properties in Section 2.3 of this Chapter. In this Section, we look specifically at BV cryptosystems, which belong to the Somewhat Homomorphic Encryption type, and allow addition and some levels of multiplication ([27]). In later chapters of the thesis, an extension of this cryptosystem (namely, BGV [26]) and its variants is used frequently. We here discuss the basic operations within a BV system, which works as follows:

Setup. Initiate (n, m, q, t, χ) to define the ciphertext space $R_q = \frac{\mathbb{Z}_q[x]}{x^n+1}$, the plaintext space $R_t = \frac{\mathbb{Z}_t[x]}{x^n+1}$, and the error distribution χ on $\mathbb{Z}_q$, for $t << q$.

KeyGen. In this step, the secret key *sk* is chosen by selecting a small element $\mathbf{s} \in R_q$, $\mathbf{s} \xleftarrow{r} \chi^n$. The public key *pk* is a pair of ring elements $(\mathbf{p}_0, \mathbf{p}_1)$, where $\mathbf{p}_1 \xleftarrow{r} R_q$ and $\mathbf{p}_0 = -(\mathbf{p}_1\mathbf{s} + t\mathbf{e})$ with $\mathbf{e} \xleftarrow{r} \chi^n$.

Encryption. Given a plaintext $\mathbf{m} \in R_t$ and a public key $pk = (\mathbf{p}_0, \mathbf{p}_1)$, the encryption first calculates $\mathbf{u}, \mathbf{f}, \mathbf{g} \xleftarrow{r} \chi$, then computes a fresh ciphertext by

$$Enc_{pk}(\mathbf{m}) = (\mathbf{c}_0, \mathbf{c}_1) = (\mathbf{p}_0\mathbf{u} + t\mathbf{g} + \mathbf{m}, \mathbf{p}_1\mathbf{u} + t\mathbf{f})$$

By convention, $[\![P]\!]$ denotes the encryption of a plaintext P under the BV scheme with the public key, and randomness is not taken into account. When the noise used in the encryption has to be specified, $[\![(P,e)]\!]$ is used to represent it, e being the noise.

Decryption. Although the above encryption generates ciphertexts of 2 elements only in R_q, the homomorphic operations (discussed next) will make the ciphertext longer. The decryption of the ciphertext appears as $c = (\mathbf{c}_0, \mathbf{c}_1, \ldots, \mathbf{c}_L)$, with secret key $sk = (1, \mathbf{s}, \mathbf{s}^2, \ldots, \mathbf{s}^L)$ as $Dec(\mathbf{c}, sk) = \Big[[\langle \mathbf{c}, \mathbf{sk} \rangle]_Q\Big]_t$.

Homomorphic Operations. Given 2 ciphertexts $c = (\mathbf{c}_0, \mathbf{c}_1, \ldots, \mathbf{c}_L)$ and $c' = (\mathbf{c}'_0, \mathbf{c}'_1, \ldots, \mathbf{c}'_K)$, before any operations are carried out, the ciphertexts are padded with zeros first if necessary (to make $K = L$, assuming that, initially, $K < L$). The homomorphic addition $add(c, c')$ is computed by the component-wise addition $add(c, c') = (\mathbf{c}_0 + \mathbf{c}'_0, \ldots, \mathbf{c}_L + \mathbf{c}'_L)$. The homomorphic multiplication $mult(c, c')$ is computed by $mult(c, c') = (\hat{\mathbf{c}_0}, \hat{\mathbf{c}_1}, \ldots, \hat{\mathbf{c}_{2L-2}})$ with $\sum_{i=0}^{2L-2} \hat{\mathbf{c}}_\mathbf{i} z^i = \sum_{i=0}^{L-1} \mathbf{c}_\mathbf{i} z^i \times \sum_{j=0}^{L-1} \mathbf{c}'_\mathbf{j} z^j$, where z denotes a symbolic variable.

4.3.1 Ciphertext packing

Given a bit string plaintext $m \in \{0,1\}^*$, there are several ways to encode it as a polynomial, or a ring element $\mathbf{m} \in \mathrm{R}_t$ before encryption. A recent popular approach for BV cryptosystems is the so-called CRT packing method, based on the Chinese Remainder Theorem [123]. The method allows Single Instruction, Multiple Data (SIMD) operations on encrypted data. However, we do not use this packing technique in this chapter as it involves some complications unrelated to the main point of this chapter (later, in chapter 7, we propose

a different approach to packing methods). This chapter focuses on the method of [133], which is an extension of [95], the technique allowing HD computation with just one level of multiplication.

Biometrics are used in authentication as the human features they record systematically differ from one person to another. There are many algorithms that extract the data used in the process ([FVC]) (we refer to this data as "biometric templates"). Many works (such as [38] and [Fuj]) use Hamming Distance (HD) as the metrics to measure the similarity of the stored and query templates, and we equally do so to evaluate our hypothesis (3). In the work of [133], the authors propose an innovative to compute the HD of two n-bits templates X and Y in the ciphertext domain. Specifically, the method packs all the bits of a template into a single ciphertext, and a linear combination of homomorphic operations would purveys an HD result. We sketch their method below.

Definition 27 ([133]). *For* $\mathbf{T} = (t_0, \ldots, t_{n-1}$ *and* $\mathbf{Q} = (q_0, \ldots, q_{n-1})$*, we define two types of polynomials in the ring* R_q *of the SHE scheme:*

$$pm_1(\mathbf{T}) = \sum_{i=0}^{n-1} t_i x^i \text{ and } pm_2(\mathbf{Q}) = -\sum_{j=0}^{n-1} q_j x^{n-j}$$

The two types of packed ciphertexts are defined as

$$ct_1(\mathbf{T}) = Enc(pm_1(\mathbf{T})) \text{ and } ct_2(\mathbf{Q}) = Enc(pm_2(\mathbf{Q}))$$

The main idea of [133] is that, if in the ring R_q where $x^n = -1$, then, when multiplying $pm_1(\mathbf{T})$ by $pm_2(\mathbf{Q})$, the constant term of the result will be the inner product $\langle \mathbf{T}, \mathbf{Q} \rangle$. If homomorphic multiplication is applied to the ciphertexts, the ciphertext of their inner product will similarly show as a result of it. Furthermore, this result can be used to compute HD as shown below, the operation costs being one level of multiplication with 3 additions and 3 multiplications on ciphertexts.

Theorem 9. *Let* $C_1 = -\sum_{i=0}^{n-1} x^{n-i}$ *and* $C_2 = 2 - C_1 = \sum_{i=0}^{n-1} x^i$. *Let* $Enc(HD)$ *be a ciphertext given by*

$$ct_1(\mathbf{T}) * Enc(C_1) + ct_2(\mathbf{Q}) * Enc(C_2) - 2 * ct_1(\mathbf{T}) * ct_2(\mathbf{Q})$$

Then, the constant term of $Dec(Enc(HD))$ *gives the Hamming Distance of* $\mathbf{T}$ *and* $\mathbf{Q}$.

Algorithm 1 HD Computation Homomorphically

1: **procedure** EVALDISTANCE($\mathbf{T}, \mathbf{Q}$)
2: $C_1 \leftarrow -\sum_{i=0}^{n-1} x^{n-i}$
3: $C_2 \leftarrow 2 - C_1$
4: $C_{HD} \leftarrow ct_1(\mathbf{T}) * Enc(C_1) + ct_2(\mathbf{Q}) * Enc(C_2) - 2 * ct_1(\mathbf{T}) * ct_2(\mathbf{Q})$
5: **return** C_{HD}

4.4 Zero-knowledge proof system

Suppose that *Paul* has found a solution to a hard problem and he wants to convince *Vicky* that the problem has been solved by him. *Paul* can simply write out the solution and give it to *Vicky*. However, *Vicky* could be as dishonest as to show the solution to others and claim that it was her who solved the problem. The Zero Knowledge Proof (ZKP) is a beautiful cryptographic concept allowing *Paul* to produce the proof in such a way that *Vicky* does not learn any information besides the fact that the problem has been solved. Since introduced in the 80s [59], ZKP has been extensively studied and is currently an important building block of many cryptographic protocols: identification schemes ([46, 43, 61, 117, 125, 76],...), group signatures([32, 9, 22, 24, 60]...), anonymous credential systems ([30], [31], [12], [34],...), interactive encryption protocols([48], [55], [58], [75],...), and many more.

In this chapter, we discuss zero-knowledge proofs of plaintext knowledge for LWE-based cryptosystems. Given a system with a public key pk, a proof of plaintext knowledge (PoPK) allows a *Prover* P to convince a *Verifier* V that it knows the plaintext M of some ciphertext $c = Enc(\mathsf{pk}, M)$. The witness used in the proof normally consists of the plaintext,

the secret key, and the randomness factor implemented during encrypting c. A PoPK system is zero-knowledge in the sense that it does not allow V to acquire any information about M. This research area has received attention in recent years. A ZKPoPK for a variant of the Ajtai-Dwork system [5] was engineered by [57], who adapted the protocol from [94]. Xagawa and Tanaka [131] used the Stern-KTX proof [125, 76] to obtain PoPK for NTRU cryptosystems [66]. Many projects targeted Regev's LWE-based cryptosystem ([17, 18, 8], but they all derived from a secure Multi-Party Computation protocol, specifically the IKOS transformation [68], and therefore had a relatively high communication cost.

4.4.1 Stern-based ZKP

Recall that we need to construct a proof for the ISIS relation:

$$R_{ISIS_{n,mq,\beta}} = \{((\mathbf{A},\vec{y}),\vec{x}) \in \mathbb{Z}_q^{n\times m} \times \mathbb{Z}_q^n \times \mathbb{Z}^m : (\|\vec{x}\|_\infty \leq \beta) \wedge (\mathbf{A}\vec{x} = \vec{y} \mod q)\}$$

There are several approaches to construct such a proof (e.g. [87], [94], [125]). We here discuss an approach based on [125]. We refer our readers to the original paper for the detailed steps of the protocol and denote the whole proof protocol $\mathbf{Stern}_{A,x,y}$. It can be applied to implement a Zero Knowledge Proof of Plaintext Knowledge (ZKPoPK) for latticed-based cryptosystems. For example, an extension of the work of [85] (denoted by **SternExt**), allows to compute plaintext knowledge by proving the encryption relation of Regev's cryptosystem [112]:

$$R_{Regev}^{q,m,n,t,\chi} = \{((p_0,p_1),(c_0,c_1),\vec{r}||M) \in (\mathbb{Z}_q^{m\times n} \times \mathbb{Z}_q^m) \times (\mathbb{Z}_q^n \times \mathbb{Z}_q) \times \mathbb{Z}_q^{n+1} : (c_0 = p_0\vec{r}) \wedge (c_1 = p_1\vec{r} + M)\}$$

The proof works by letting $\mathbf{A}' = \begin{bmatrix} p_1, 1 \\ p0, 0 \end{bmatrix}$ and $\mathbf{y} = \begin{bmatrix} c_1 \\ c_0 \end{bmatrix}$ be the public parameters and by letting $\mathbf{x} = \begin{bmatrix} \vec{r} \\ M \end{bmatrix}$ be the *Prover*'s witness. We observe that $\mathbf{A}'\mathbf{x} = \mathbf{y} \mod q$, that is, $\mathbf{x}$, is a solution to the ISIS problem, provided that $\|\vec{r}\|_\infty \approx |M|$ AND the *Prover* within a symmetric key setting. In many other contexts, the client does not know $\vec{r}$, as encryption is ensured by other parties using public keys. For such situations, we can look at the decryption equation:

$$\begin{aligned} c_1 - c_0\vec{s} &= p_1\vec{r} + M - p_0\vec{r}\vec{s} \\ &= \mathbf{A}\vec{s}\vec{r} + t\vec{e}\vec{r} + M - \mathbf{A}\vec{r}\vec{s} \\ &= t\tilde{e} + M \end{aligned}$$

We denote S_1, S_2, S_m to be subsets of $\mathbb{Z}_q$ corresponding to small coefficients of $\mathbf{s}, \mathbf{e}, \mathbf{m}$ Therefore, we can write out the decryption relation as:

$$R^{q,m,n,t\chi}_{Regev,dec} = \{((p_0,p_1),(c_0,c_1),\vec{s},\vec{e},\tilde{e},M) \in (\mathbb{Z}_q^{m\times n} \times \mathbb{Z}_q^m) \times (\mathbb{Z}_q^n \times \mathbb{Z}_q) \times S_1^n \times S_2^n \times S_m \times \mathbb{Z}_Q : (p_1 = p_0\vec{s} + t\vec{e}) \wedge (c_1 = c_0\vec{s} + t\tilde{e} + M)\}$$

Similarly, we can let $\mathbf{A}_{Stern} = \begin{bmatrix} c_0, t, 0, 1 \\ p_0, 0, t, 0 \end{bmatrix}$ (the public parameters) and let $\mathbf{X}_{Stern} = [\vec{s}, \tilde{e}, \vec{e}, M]^T$ and try applying **SternExt** to obtain the ZKPoPK. However, this will not work because the original solution proves that $\|\mathbf{X}_{Stern}\|_\infty < \beta$. In such cases, we want to prove the bound of separate components differently: in our above example, $\|\tilde{e}\|_\infty > \|\vec{e}\|_\infty$. In a Regev cryptosystem, the problem might not be clear enough, as the norms of each vector in $\mathbf{X}_{Stern}$ are quite close to each other. In other latticed-based system, the difference can be big, as for example,

the BV system's decryption relation:

$$R_{BV}^{q,n,t,\chi} = \{((\mathbf{c}_0,\mathbf{c}_1),(\mathbf{p}_0,\mathbf{p}_1),\mathbf{s},\mathbf{e}',\mathbf{e},\mathbf{m} \in (R_Q \times R_Q) \times (R_Q \times R_Q) \times S_1^n \times S_2^n \times S_2^n \times S_m : (\mathbf{p}_1\mathbf{s} + t\mathbf{e} = -\mathbf{p}_0) \wedge (\mathbf{c}_1\mathbf{s} - t\mathbf{e}' - \mathbf{m} = -\mathbf{c}_0)\}$$

Our *Prover*'s witness in this situation is $\mathbf{X}_{Stern} = [\mathbf{s},\mathbf{e}',\mathbf{e},\mathbf{m}]^T$ with $\|s\|_\infty \approx \|e\|_\infty > \|e'\|_\infty >> \|m\|_\infty$. Therefore, instead of proving $\|\mathbf{X}_{Stern}\|_\infty < \beta$, we need a proof with different constraints on the witness' components. In particular in our application, we need to prove that the message has binary coefficients (for correctness of the Hamming Distance homomorphic computation), whereas the errors have non-binary coefficients. We present a solution for this problem.

Our construction

The idea. is that, instead of proving $\|x_i\|_\infty < \beta_i$, or proving that all the coefficients of x_i are in the range $\{-\beta_i,\ldots,\beta_i\}$, it can also be proved that $x_i + \beta_i.f(x)$ is in the range $\{0,\ldots,2\beta_i\}$, where $f(x) = 1 + x + x^2 + \cdots + x^{n-1}$. Secondly, if we decompose $x_i + \beta_i.f(x)$ to its binary representation and apply the Stern's variant of [76] to prove the relation

$$R_{KTX} = \{((\mathbf{A},\mathbf{y}),\mathbf{x}) \in \mathscr{Z}_q^{n\times m} \times \mathscr{Z}_q^n \times \{0,1\}^m : wt(\mathbf{x}) \wedge \mathbf{A}.\mathbf{x} = \mathbf{y} \mod q\},$$

the proof for the original relation $R_{BV}^{q,n,t,\chi}$ can be obtained. Note that, at this point the *Prover*'s witness is a binary vector, that is, if it needs to be to proved that some part of the message is binary, such a proof can be obtained at this point as well. It is important to use such a proof for latticed-based cryptosystems where the message space is $\mathbf{R}_2$: a proof for the ISIS relation would not be suitable in such situations, as it only asserts that the infinity norm of the whole witness is less than some β.

Protocol description. The protocol **SternBV(A,y,x)** works as follows: Let A be a matrix of $m \times l$ ring element ($A \in R_q^{m \times l}$), $\mathbf{x}$ be a vector of l ring elements $\mathbf{x} = \{\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_l\}$ and similarly $\mathbf{y} = \{\mathbf{y}_1, \ldots, \mathbf{y}_m\}$. The protocol includes the following steps:

Step 1. Normalizing the bound of each component x_i of $\mathbf{x}$ from $\{-\beta_i, \ldots, \beta_i\}$ to $\{0, \ldots, 2^{l_i}\}$, where l_i is the smallest integer satisfying $2^{l_i} > (2\beta_i - 1)$. This step is done by one ring multiplication for each $\mathbf{x}_i$, let $\mathbf{x}'_i = \mathbf{x}_i + \beta_i.\mathbf{f}(x)$, where $\mathbf{f}(x) = 1 + x + x^2 + \cdots + x^{n-1}$. After this normalization step, instead of proving the relation $\sum_j \mathbf{a}_{i,j}\mathbf{x}_j = \mathbf{y}_i$ with $\|\mathbf{x}_i\|_\infty \in \{-\beta_i, \ldots, \beta_i\}$, we prove $\sum_j \mathbf{a}_{i,j}\mathbf{x}'_j = \mathbf{y}'_i$ with $\|\mathbf{x}'_i\|_\infty \in \{0, \ldots, 2^{l_i}\}$, where $\mathbf{y}'_i = \sum_j \mathbf{a}_{i,j}\beta_j\mathbf{f}(x)$.

Step 2. Decompose $\mathbf{x}'_i = \mathbf{x}_i + \beta_i$ into their binary representation

$$\mathbf{x}''_i = \sum_{j=0}^{l_i-1} 2_j b_j$$

Let $\mathbf{x}''$ be the result ring element that concatenates all $\mathbf{x}''_i$ and has $L = \sum l_i$ coefficients. In this step we need to hide the Hamming Weight of the secret vector $\mathbf{x}'_i$. This hiding task is achieved by padding:

1. Let ζ_0 and ζ_1 be the number of coefficients of $\mathbf{x}''$ equal to 0 or 1, respectively.
2. Sample a random vector $\zeta \in \{0,1\}^L$ that has $(L - \zeta_0)$ coefficients 0 and $(L - \zeta_1)$ coefficients 1.
3. Output $\mathbf{x}_{\mathbf{Stern}} = \mathbf{x}''||\zeta$.

The resulting binary vector $\mathbf{x}_{\mathbf{Stern}}$ has length $2L$ and the total number of 0s and 1s in the $\mathbf{x}_{\mathbf{Stern}}$ are the same.

Step 3. We denote $rot(\mathbf{c}) \in \mathbb{Z}_Q^{n \times n}$ to be an anti-circulant square matrix, whose first column is $\mathbf{c}$, the remaining columns of it being the cyclic rotations of $\mathbf{c}$ with the cycled entries

negated

$$rot(\mathbf{c}) = \begin{bmatrix} c_0 & -c_{n-1} & -c_{n-2} & \dots \\ c_1 & c_0 & -c_{n-1} & \dots \\ \dots & \dots & \dots & \dots \\ c_{n-1} & c_{n-2} & c_{n-3} & \dots \end{bmatrix}$$

The matrix A can be then reconstructed as *rot* matrices:

$$\forall \mathbf{a}_{i,j} \in \mathbf{A} : \mathbf{a}'_{i,j} = rot(\mathbf{a}_{i,j})$$

The result expanded matrix is denoted $\mathbf{A}'$. We also need to pad the resulting matrix with a corresponding number of 0s to make sure that $\mathbf{A}'$ complies with all x'_i. Let $\mathbf{A_{Stern}}$ be the padded result.

Step 4. Modify $\mathbf{y}_i$: Let $\mathbf{y}'_i = \sum_j \mathbf{a}_{i,j}\beta_j\mathbf{f}(x)$ and let $\mathbf{y_{Stern}}$ be the concatenation of all $\mathbf{y}'_i$.

Step 5. Run the Stern protocol (Section 2.5.3) for the proof of $\mathbf{A_{Stern}x_{Stern}} = \mathbf{y_{Stern}}$.

Result. Our protocol has the following properties:

- The knowledge extractor produces different x_i with $\|x_i\|_\infty \leq \beta_i$. Inheriting from the original Stern protocol, the extraction gap is $\gamma = 1$. In other words, if the prover P has a valid witness for the relation and it follows the protocol, then it is always accepted by the verifier V, the proof system has perfect completeness
- The communication cost is $2(n \log q)\sum l_i + commitmentSize$ for each round.
- The Statistical Zero-Knowledge property can be proved by adapting the technique from [125, 76] to construct a simulator S which has black-box access to a cheating verifier $\hat{V}$: on inputs $\mathbf{A_{stern}}, \mathbf{y_{stern}}$, outputs an accepted transcript with probability $2/3$. Additionally, the *view* of $\hat{V}$ in the simulation should be statistically close to the view of the verifier in the real situation.

Algorithm 2 ZKPoPK Improved for BV

procedure ZKPBV$((\mathbf{c}, pk), (\mathbf{m}, \mathbf{s}, \mathbf{e}, \mathbf{e}'))$
  $rot_{c_1} \leftarrow rot(\mathbf{c}_1)$
  $rot_{p_1} \leftarrow rot(\mathbf{pk}_1)$
  let I be the $n \times n$ identity matrix
  let Z be the $n \times n$ zero matrix
  $\mathbf{A} \leftarrow \begin{bmatrix} rot_c, -tI, Z, -I \\ rot_{p_1}, Z, tI, Z \end{bmatrix}$
  $\mathbf{x} \leftarrow (\mathbf{s}, \mathbf{e}', \mathbf{e}, \mathbf{m})$
  $\mathbf{y} \leftarrow (-\mathbf{c_0}, -\mathbf{pk_0})$
  Return SternExt$(\mathbf{A}, \mathbf{x}, \mathbf{y})$

4.5 Our Protocol

4.5.1 The protocol specification

We denote $\mathscr{U}$ to be the client and $\mathscr{S}$ to be the server. There are 3 main submodules in the protocol: Setup, Enrol, and Authenticate. Figure 4.1 illustrates the protocol transcript.

Setup. $\mathscr{U}$ and $\mathscr{S}$ initialize the parameters, taking into account the following categories:

Biometric Authentication System Parameters. These parameters are standard ones used by non privacy preserving biometric authentication systems:

- False Acceptance Rate (FAR) and False Rejection Rate (FRR)
- a: The maximum number of incorrect authentication attempts allowed by $\mathscr{S}$.
- τ: Threshold to compare the Hamming Distance to decide the authentication result.
- n': The bit-length of the encoded biometric data.

Ring-LWE based techniques parameters. These parameters are used in the lattice-based cryptosystem which provides client privacy against long term quantum attacks.

- λ: General security parameter of the cryptosystem (the adversary's winning chance in the CPA security game is $1/2^{\lambda}$)
- n: Integer n defining the plaintext and ciphertext rings. This will be referred to as the degree of the polynomial objects or dimension of the underlying lattice during correctness and security proofs.
- t: Integer t defining the plaintext space ring $R_t = \mathbb{Z}_t[x]/x^n + 1$.
- q: Integer q defining the ciphertext space ring $R_q = \mathbb{Z}_q[x]/x^n + 1$
- $\chi_{\alpha q}$: A distribution used to sample noises for LWE-based techniques. Typically, χ is a discrete Gaussian distribution with standard deviation αq (Section 2.2.3).

Keygen. Keys are generated for $\mathscr{U}$:

- Secret key: $\mathbf{s} \xleftarrow{r} \chi_{\alpha q}^n$, $sk = (1, \mathbf{s}, \mathbf{s^2}, ...)$
- Public key: $pk = (\mathbf{p}_0, \mathbf{p}_1)$, with $\mathbf{p}_1 \xleftarrow{r} R_q$ and $\mathbf{p}_0 = -\mathbf{p}_1\mathbf{s} - t\mathbf{e}$, with $\mathbf{e} \xleftarrow{r} \chi_{\alpha q}^n$.

Enrolment. $\mathscr{U}$ extracts the biometric template $\mathbf{x}$, the bit string $\mathbf{x}$ being represented as a ring element of R_t. The encryption is done by $[\![\mathbf{x}]\!] = (\mathbf{c}_0, \mathbf{c}_1)$ and sent to $\mathscr{S}$.

Authentication. The following steps are required:

1. $\mathscr{U}$ extracts his biometric features again $\mathbf{y}$ to use them as the query. $\mathscr{U}$ sends $[\![\mathbf{y}]\!] = (\mathbf{c}'_0, \mathbf{c}'_1)$ to $\mathscr{S}$.
2. ZKP for the first relation: $\mathscr{U}$ has to prove that $[\![\mathbf{y}]\!]$ is a valid encryption, that is, it encrypts a bit string under the BV cryptosystem using the corresponding secret key. This is done by module **SternBV(A, y, x)** described in Sect. 4.4.1, where $\mathbf{A} = \begin{bmatrix} c_{y0}, t, 0, 1 \\ p_{y0}, 0, t, 0 \end{bmatrix}$, $\mathbf{x} = [\vec{s}, \vec{e}_y, \vec{e}_0, y]^T$ and $\mathbf{y} = [\mathbf{c_{y1}}, \mathbf{p_{y1}}]^T$.

3. HD Computation: $\mathscr{S}$ computes $[\![HD_{\mathbf{x},\mathbf{y}}]\!]$ using procedure **EvalDistance** 4.3.1 and mask HD by adding some random value r homomorphically to $[\![HD_{\mathbf{x},\mathbf{y}}]\!]$. The result $[\![\mathbf{HD}']\!] = [\![HD_{x,y}]\!] + [\![r]\!]$ is then sent to $\mathscr{U}$.
4. $\mathscr{U}$ decrypts $[\![\mathbf{HD}']\!]$ and derives the actual value HD' from the first coefficient of the plaintext: $dec\,[\![\mathbf{HD}']\!] = HD' + r_1 + r_2 + \cdots + r_{n-1}$. $\mathscr{U}$ sends $HD' \in \mathbb{Z}$ to $\mathscr{S}$.
5. $\mathscr{U}$ proves that it does the decryption honestly, this is done similarly to step 2.
6. $\mathscr{S}$ unmasks HD' (by computing $HD'' = HD' - r$) and outputs the authentication result $HD \overset{?}{<} \tau$

4.5.2 Correctness and Security

Correctness

In the enrolment step, as mentioned before, in order to encrypt $\mathbf{x}$, $\mathscr{U}$ samples $\mathbf{u}, \mathbf{f}, \mathbf{g} \xleftarrow{r} \chi^n_{\alpha q}$ and does $\mathbf{c}_0 = \mathbf{p}_0\mathbf{u} + t\mathbf{f} + \mathbf{x}$ and $\mathbf{c}_1 = \mathbf{p}_1\mathbf{u} + t\mathbf{g}$. The condition for decryption correctness is $[\langle \mathbf{c}_0 + \mathbf{c}_1\mathbf{s} \rangle]_q < q/2$, or, $-t\mathbf{e}\mathbf{u} + t\mathbf{f} + t\mathbf{g}\mathbf{s} < q/2$.

In the authentication step, $[\![HD]\!]$ is decryptable if $\|\langle [\![HD]\!], \mathbf{s} \rangle\|_\infty < q/2$, if we let U to be the upper bound of $\|\langle [\![HD]\!], \mathbf{s} \rangle\|_\infty$, and considering that $\|a+b\|_\infty \leq \|a\|_\infty + \|b\|_\infty$ and $\|a.b\|_\infty \leq n.\|a\|_\infty.\|b\|_\infty$. From theorem 7, we can derive $\|\langle [\![HD]\!], \mathbf{s} \rangle\|_\infty \leq 2nU + 2nU^2$. As in the work of [95], we can take U to be $2t\sigma^2\sqrt{n}$, which is an experimental estimation. Therefore, the final correctness condition for authentication is $16n^2t^2\sigma^4 < q$.

Lemma 7 (Condition for Correct Decryption of HD)**.** *For the BV encrypted Hamming Distance $[\![HD]\!]$, the decryption recovers the correct result if $\langle [\![HD]\!], \mathbf{s} \rangle$ does not wrap around mod q, namely, if $16n^2t^2\sigma^4 < q$.*

Security

The proposed scheme satisfies the security notions defined in Section 3.1

THE FIRST PROTOCOL

Client		**Server**
..... Enrolment		
Extract $\mathbf{x} \xleftarrow{r} D_k$		
$[\![\mathbf{x}]\!] = Enc(\mathbf{x}, \mathsf{pk})$	$\xrightarrow{[\![\mathbf{x}]\!]}$	Persist $[\![\mathbf{x}]\!]$
..... Authentication		
Extract $\mathbf{y} \xleftarrow{r} D_k$		
$[\![\mathbf{y}]\!] = Enc(\mathbf{y}, \mathsf{pk})$	$\xrightarrow{[\![\mathbf{y}]\!]}$	
	$\xrightarrow{\textbf{ZKPoPK1}}$	$[\![\mathbf{HD}]\!] = EvalDistance([\![\mathbf{x}]\!], [\![\mathbf{y}]\!])$
		$\mathbf{r} \xleftarrow{r} \mathbb{Z}_{HD}$
		$[\![\mathbf{HD}']\!] = \mathbf{Add}([\![\mathbf{HD}]\!], [\![\mathbf{r}]\!])$
	$\xleftarrow{[\![\mathbf{HD}']\!]}$	
$HD' + r_1 + \cdots + r_{n-1} = Dec([\![\mathbf{HD}']\!], \mathsf{sk})$		
	$\xrightarrow{HD'}$	
	$\xrightarrow{\textbf{ZKPoPK2}}$	$HD = HD' - r$
	$\xleftarrow{\textbf{Accept}\|\textbf{Reject}}$	$HD \overset{?}{<} \tau$

Fig. 4.1 The First Protocol

Theorem 10 (Server side security)**.** *Under the IND-CPA security of BV cryptosystems, and the zero-knowledge property of the Stern protocol, the proposed scheme satisfies (Honest But Curious) Server Privacy Security.*

The proof of Theorem 10 can be obtained using a sequence of games between a challenger $\mathscr{C}$ and an adversary $\mathscr{A}$. We present a sequence of games below. The idea is to remove sk_k and X_k from being used to compute the view of $\mathscr{A}$, except for $Verify(X_k, Y_k^j)$ queries, as in the ideal game, relying on the correctness of the protocol and the IND-CPA security of the BV encryption scheme and the zero-knowledge property of the ZKP simulator.

Game 0 (Figure 4.2). Game 0 is the original real privacy game, in which $[\![\mathbf{X}_k]\!] = Enc(\mathbf{X}_k, \mathsf{pk})$ is given by $\mathscr{C}$ to $\mathscr{A}$ at the beginning of the game. Then, for $j = 1\ldots, q$, the attacker sends $Y_i \in \{0,1\}^n$ to $\mathscr{C}$, and the latter simulates a run of the authentication protocol between an honest client with input (k, Y_i, sk_k) and an honest server with input $(k, [\![\mathbf{X}_k]\!])$, returning to $\mathscr{A}$ the protocol's view $V_S^{(i)}$ of the server. Finally, $\mathscr{A}$ outputs a bit β. In the following Game j, we let S_j denote the event that $\beta = 1$.

Game 1 (Figure 4.3). The computation of the authentication result bit res_i sent by the server to the client from the decryption of the ciphertext $[\![\mathbf{HD}'_i]\!]$ (its value in Game 0) is changed into the result returned by $Verify(X_k, Y_i)$, the change is highlighted belows. By the correctness constraint of the protocol, this does not change the value of res_i, so $\Pr[S_1] = \Pr[S_0]$.

Game 2 (Figure 4.4). The computation of the zero-knowledge protocol transcripts is modified: Instead of computing the transcripts using the secret, we generate them using the statistical zero-knowledge simulator algorithms for the zero-knowledge proofs. By the zero-knowledge property, whose simulator has a $2^{-\lambda}$ statistical distance from the real distribution, the result of it being $\Pr[S_2] - \Pr[S_1] \leq 2^{-\lambda}$.

Game 3 (Figure 4.5). We change the computation of the ciphertexts $[\![\mathbf{X}_k]\!]$ to encrypt zero messages, instead of encrypting the secret-related messages resulting from the previous game.

$\mathsf{Game}_0(n)$

1 : $D_k \xleftarrow{r} D_{bio}$
2 : $(\mathsf{sk}, \mathsf{pk}) \leftarrow KeyGen$
3 : $\mathbf{X}_k \xleftarrow{r} D_k$
4 : $[\![\mathbf{X_k}]\!] \leftarrow Enc(\mathbf{X}_k, \mathsf{pk})$
5 : For $i = 1, \ldots, q$:
6 : $\mathbf{Y}_i \leftarrow \{0,1\}^n$
7 : $[\![\mathbf{Y}_i]\!] \leftarrow Enc(\mathbf{Y}_i, \mathsf{pk})$
8 : $verifierView1 \leftarrow \mathbf{ZKPoPK1}((\mathsf{sk}, \mathbf{Y}_i); (\mathsf{pk}, [\![\mathbf{Y}_i]\!]))$
9 : $[\![\mathbf{HD}_i]\!] = EvalDistance([\![\mathbf{X}_k]\!], [\![\mathbf{Y}_i]\!])$
10 : $r_i \xleftarrow{r} \mathbb{Z}_{HD}, [\![\mathbf{HD'}_i]\!] = Add([\![\mathbf{HD}_i]\!], [\![r_i]\!])$
11 : $HD'_i \leftarrow Dec([\![\mathbf{HD'}_i]\!], \mathsf{sk})$
12 : $verifierView2 \leftarrow \mathbf{ZKPoPK2}((\mathsf{sk}, \mathbf{HD'}_i); (\mathsf{pk}, [\![\mathbf{HD'}_i]\!]))$
13 : $HD_i \leftarrow HD'_i - r_i$
14 : $res_i \leftarrow compare(HD_i, \tau)$
15 : $\beta \leftarrow \mathscr{A}(verifierView1, verifierView2, [\![\mathbf{X}_k]\!], [\![\mathbf{Y}_i]\!], res_i, r_i, HD_i, [\![\mathbf{HD'}_i]\!], [\![\mathbf{HD}_i]\!])$

Fig. 4.2 Game 0 - Server privacy game

$\mathsf{Game}_1(n)$

1 : $D_k \xleftarrow{r} D_{bio}$
2 : $(\mathsf{sk}, \mathsf{pk}) \leftarrow KeyGen$
3 : $\mathbf{X}_k \xleftarrow{r} D_k$
4 : $[\![\mathbf{X}_k]\!] \leftarrow Enc(\mathbf{X}_k, \mathsf{pk})$
5 : For $i = 1, \ldots, q$:
6 : $\mathbf{Y}_i \leftarrow \{0,1\}^n$
7 : $[\![\mathbf{Y}_i]\!] \leftarrow Enc(\mathbf{Y}_i, \mathsf{pk})$
8 : $verifierView1 \leftarrow \mathbf{ZKPoPK1}((\mathsf{sk}, \mathbf{Y}_i); (\mathsf{pk}, [\![\mathbf{Y}_i]\!]))$
9 : $[\![\mathbf{HD}_i]\!] = EvalDistance([\![\mathbf{X}_k]\!], [\![\mathbf{Y}_i]\!])$
10 : $r_i \xleftarrow{r} \mathbb{Z}_t, [\![\mathbf{HD'}_i]\!] = Add([\![\mathbf{HD}_i]\!], [\![r_i]\!])$
11 : $HD'_i \leftarrow Dec([\![\mathbf{HD'}_i]\!], \mathsf{sk})$
12 : $verifierView2 \leftarrow \mathbf{ZKPoPK2}((\mathsf{sk}, \mathbf{HD'}_i); (\mathsf{pk}, [\![\mathbf{HD'}_i]\!]))$
13 : $HD_i \leftarrow HD'_i - r_i$
14 : $res_i \leftarrow Verify(X_k, Y_i)$
15 : $\beta \leftarrow \mathscr{A}(verifierView1, verifierView2, [\![\mathbf{X}_k]\!], [\![\mathbf{Y}_i]\!], res_i, r_i, HD_i, [\![\mathbf{HD'}_i]\!], [\![\mathbf{HD}_i]\!])$

Fig. 4.3 Game 1 - Server privacy game

$\mathsf{Game}_2(n)$

1 : $D_k \xleftarrow{r} D_{bio}$
2 : $(\mathsf{sk}, \mathsf{pk}) \leftarrow KeyGen$
3 : $\mathbf{X}_k \xleftarrow{r} D_k$
4 : $[\![\mathbf{X_k}]\!] \leftarrow Enc(\mathbf{X}_k, \mathsf{pk})$
5 : For $i = 1, \ldots, q$:
6 : $\mathbf{Y}_i \leftarrow \{0,1\}^n$
7 : $[\![\mathbf{Y}_i]\!] \leftarrow Enc(\mathbf{Y}_i, \mathsf{pk})$
8 : $verifierView1 \leftarrow$ **ZKPSimulator**
9 : $[\![\mathbf{HD}_i]\!] = EvalDistance([\![\mathbf{X}_k]\!], [\![\mathbf{Y}_i]\!])$
10 : $r_i \xleftarrow{r} \mathbb{Z}_{HD}, [\![\mathbf{HD'}_i]\!] = Add([\![\mathbf{HD}_i]\!], [\![r_i]\!])$
11 : $HD'_i \leftarrow Dec([\![\mathbf{HD'}_i]\!], \mathsf{sk})$
12 : $verifierView2 \leftarrow$ **ZKPSimulator**
13 : $HD_i \leftarrow HD'_i - r_i$
14 : $res_i \leftarrow Verify(X_k, Y_i)$
15 : $\beta \leftarrow \mathscr{A}(verifierView1, verifierView2, [\![\mathbf{X}_k]\!], [\![\mathbf{Y}_i]\!], res_i, r_i, HD_i, [\![\mathbf{HD'}_i]\!], [\![\mathbf{HD}_i]\!])$

Fig. 4.4 Game 2 - Server privacy game

We also change the computation of HD_i to the direct operation $HammingDistance(\mathbf{X}_k, \mathbf{Y}_i)$. Since sk_k is thus not used any longer for generating the view of $\mathscr{A}$, it follows by a hybrid argument that $|\Pr[S_3] - \Pr[S_2] \cdot (q+1) \cdot \varepsilon_{BV}$, where ε_{BV} denotes the maximal advantage of an attacker against IND-CPA of the BV scheme against attacks with run-time $T + \text{poly}(n, \log Q)$, where T is the run-time of $\mathscr{A}$. In this game, since the only information on X_k comes via the $Verify(X_k, Y_k^j)$ and $HammingDistance$ queries, the challenger together with $\mathscr{A}$ constitute an efficient attacker against the ideal privacy game, which outputs 1 with a degree of probability different by at most $(l+1) \cdot q + 1) \cdot \varepsilon_{BV}$ from the probability of outputting 1 in the real privacy game, as required.

Theorem 11 (Client side security)**.** *Under the IND-CPA security of BV cryptosystem and the soundness property of the underlying Stern protocol, the proposed scheme satisfies Impersonation and Multifactor Security.*

$\mathsf{Game}_3(n)$

1 : $D_k \xleftarrow{r} D_{bio}$
2 : $(\mathsf{sk}, \mathsf{pk}) \leftarrow KeyGen$
3 : $\mathbf{X}_k \xleftarrow{r} D_k$
4 : $[\![\mathbf{X_k}]\!] \leftarrow Enc(0, \mathsf{pk})$
5 : For $i = 1, \ldots, q$:
6 : $\mathbf{Y}_i \leftarrow \{0,1\}^n$
7 : $[\![\mathbf{Y}_i]\!] \leftarrow Enc(0, \mathsf{pk})$
8 : $verifierView1 \leftarrow$ **ZKPSimulator**
9 : $[\![\mathbf{HD}_i]\!] = EvalDistance([\![\mathbf{X}_k]\!], [\![\mathbf{Y}_i]\!])$
10 : $r_i \xleftarrow{r} \mathbb{Z}_{HD}, [\![\mathbf{HD'}_i]\!] = Add([\![\mathbf{HD}_i]\!], [\![r_i]\!])$
11 : $HD'_i \leftarrow Dec([\![\mathbf{HD'}_i]\!], \mathsf{sk})$
12 : $verifierView2 \leftarrow$ **ZKPSimulator**
13 : $HD_i \leftarrow HammingDistance(\mathbf{X}_k, \mathbf{Y}_i)$
14 : $res_i \leftarrow Verify(X_k, Y_i)$
15 : $\beta \leftarrow \mathscr{A}(verifierView1, verifierView2, [\![\mathbf{X}_k]\!], [\![\mathbf{Y}_i]\!], res_i, r_i, HD_i, [\![\mathbf{HD'}_i]\!], [\![\mathbf{HD}_i]\!])$

Fig. 4.5 Game 3 - Server privacy game

The proof of theorem 11 can be obtained using a sequence of games between the challenger $\mathscr{C}$ and the adversary $\mathscr{A}$. We here present such a sequence, as well as the relations among the games, in order to demonstrate the type I security model proof. In the next chapter, we will present the type II security proof, where the ciphertext of the biometric data instead of the key is compromised. The steps of the proofs can be applied in both the first and the second variants of the protocols.

Game 0 (Figure 4.6). Game 0 is the original impersonation game for type I attack.

Setup. $\mathscr{C}$ intiates $D_k \xleftarrow{r} D_{bio}$ and $X_k \xleftarrow{r} D_k$. $\mathscr{C}$ sets up (sk_k, pk_k) and launches $Enrol(k, X_k)$ to get $(sk_k, [\![\mathbf{X}_k]\!] = (pk_k, \mathbf{X}_k))$. $\mathscr{A}$ submits a type 1 attack query and receives sk_k from $\mathscr{C}$.

Query. $\mathscr{A}$ runs q authentication sessions. In each session $j = 1, \ldots, q$, $\mathscr{A}$ can choose a ciphertext. $\mathscr{A}$ and $\mathscr{C}$ run **ZKPoPK1**. $\mathscr{C}$ evaluates $[\![\mathbf{HD}_k]\!]$ then computes $[\![\mathbf{HD'}_k]\!]$. The

resulting ciphertext is sent to $\mathscr{A}$, $\mathscr{A}$ decrypts $[\![\mathbf{HD'}_k]\!]$ and sends back the plaintext value of HD'_i. $\mathscr{A}$ and $\mathscr{C}$ run **ZKPoPK2**(). $\mathscr{C}$ computes $HD = HD' - r$, compares the result with τ sends back to $\mathscr{A}$ the authentication result **Accept** or **Reject**.

Guess. At the end of the game, $\mathscr{A}$ outputs $\mathbf{Y}'$.

$\mathsf{Game}_0(n)$

1 : $D_k \xleftarrow{r} D_{bio}$
2 : $(\mathsf{sk}, \mathsf{pk}) \leftarrow KeyGen$
3 : $\mathbf{X}_k \xleftarrow{r} D_k$
4 : $[\![\mathbf{X_k}]\!] \leftarrow Enc(\mathbf{X}_k, \mathsf{pk})$
5 : For $i = 1, \ldots, q$:
6 : $\mathbf{Y}_i \leftarrow \{0,1\}^n$
7 : $[\![\mathbf{Y}_i]\!] \leftarrow Enc(\mathbf{Y}_i, \mathsf{pk})$
8 : $proverView1 \leftarrow$ **ZKPoPK1**$((\mathsf{sk}, \mathbf{Y}_i); (\mathsf{pk}, [\![\mathbf{Y}_i]\!]))$
9 : $[\![\mathbf{HD}_i]\!] = EvalDistance([\![\mathbf{X}_k]\!], [\![\mathbf{Y}_i]\!])$
10 : $r_i \xleftarrow{r} \mathbb{Z}_{HD}$,$[\![\mathbf{HD'}_i]\!] = Add([\![\mathbf{HD}_i]\!], [\![r_i]\!])$
11 : $HD'_i \leftarrow Dec([\![\mathbf{HD'}_i]\!], \mathsf{sk})$
12 : $proverView2 \leftarrow$ **ZKPoPK2**$((\mathsf{sk}, \mathbf{HD'}_i); (\mathsf{pk}, [\![\mathbf{HD'}_i]\!]))$
13 : $HD_i \leftarrow HD'_i - r_i$
14 : $res_i \leftarrow compare(HD_i, \tau)$
15 : $\beta \leftarrow \mathscr{A}(proverView1, proverView2, sk, \mathbf{Y}_i, [\![\mathbf{Y}_i]\!], res_i, HD'_i, [\![\mathbf{HD'}_i]\!])$

Fig. 4.6 Game 0 - Client Impersonation game

Next, we discuss following games, the plan being to proceed towards the final game, where everything related to X_k that $\mathscr{A}$ receives can be simulated without any knowledge about X_k (the only known information about X_k is the function $Verify(X_k, Y_i)$). Let $res_i = Verify(X_k, Y_i)$ $(i = 1, \ldots, q)$ and S_j be the event in the game j such that $res_i = Accept$ for some $Y_i \in 0, 1^n$.

Game 1. In this game, we abort **ZKPoPK1** if $[\![\mathbf{Y}_i]\!]$ is not a valid encryption of the query, but

the *Prover* manages to pass the proof. Let $(*)_i$ be this event.

$$(*_1)res_i = \begin{cases} \text{Reject if } \textbf{ZKPoPK1} \text{ fails} \\ \text{res else} \end{cases}$$

Let bad_0 be the event in game 0 and $\exists j \leq q \; s.t \; (*_1)$ be the case. We want to show that when we modify *game 0*, the probability of a successful forgery S_1 in *game 1* is not much lower than before. Due to the modification, we observe that $Pr[S_1] \geq Pr[S_0] - Pr[bad_0]$. For any j in the q authentication attemps, by the $\varepsilon - soundness$ property of **ZKPoPK1** as a proof of membership in (*), $Pr[(*_1) \; occurs \; for \; some \; j] \leq \varepsilon_{ZK1}$ holds. That is, the probability of bad_0 would be the union of these events, bounded by $Pr[bad_0] \leq q\varepsilon_{ZK1}$. In other words, the advantage of $\mathscr{A}$ in *game 1* is

$$Pr[S_1] \geq Pr[S_0] - Pr[bad_0] \geq \varepsilon_{imp} - q\varepsilon_{ZK1}$$

Game 2. In this game, we abort **ZKPoPK2** if in one of the i^{th} runs, the $Verifier$ accepts but the ciphertext $[\![\mathbf{HD}']\!]$ does not satisfy the relation. Let bad_1 be this event. By the same type of argument, we can derive $Pr[bad_1] \leq q\varepsilon_{ZK2}$ and, therefore,

$$Pr[S_2] \geq Pr[S_1] - Pr[bad_1] \geq \varepsilon_{imp} - q(\varepsilon_{ZK1} + \varepsilon_{ZK2})$$

By the end of *Game 2*, if in any authentication attempt i, none of the 2 games have been aborted, then, by the correctness property of ZKP, the server's output is equal to the output of $Verify(X_k, Y_i)$. In other words, we have shown what we could get by just querying the oracle $Verify()$. Next, we want to simulate everything related to X_k available to an attacker using

that oracle.

$\mathsf{Game}_2(n)$

1 : $D_k \xleftarrow{r} D_{bio}$
2 : $(\mathsf{sk}, \mathsf{pk}) \leftarrow KeyGen$
3 : $\mathbf{X}_k \xleftarrow{r} D_k$
4 : $[\![\mathbf{X_k}]\!] \leftarrow Enc(\mathbf{X}_k, \mathsf{pk})$
5 : For $i = 1, \ldots, q$:
6 : $\mathbf{Y}_i \leftarrow \{0,1\}^n$
7 : $[\![\mathbf{Y}_i]\!] \leftarrow Enc(\mathbf{Y}_i, \mathsf{pk})$
8 : $proverView1 \leftarrow$ **ZKPSimulator**()
9 : $[\![\mathbf{HD}_i]\!] = EvalDistance([\![\mathbf{X}_k]\!], [\![\mathbf{Y}_i]\!])$
10 : $r_i \xleftarrow{r} \mathbb{Z}_{HD}$,$[\![\mathbf{HD'}_i]\!] = Add([\![\mathbf{HD}_i]\!], [\![r_i]\!])$
11 : $HD'_i \leftarrow Dec([\![\mathbf{HD'}_i]\!], \mathsf{sk})$
12 : $proverView2 \leftarrow$ **ZKPSimulator**()
13 : $HD_i \leftarrow HD'_i - r_i$
14 : $res_i \leftarrow compare(HD_i, \tau)$
15 : $\beta \leftarrow \mathscr{A}(proverView1, proverView2, sk, \mathbf{Y}_i, [\![\mathbf{Y}_i]\!], res_i, HD'_i, [\![\mathbf{HD'}_i]\!])$

Fig. 4.7 Game 1 & 2 - Client Impersonation game

Game 3. At the end of *Game 2*, $\mathscr{C}$ has the bit $b = Verify(Y^{(i)}, X_k)$. In this game, we can change res_i from $compare(HD_i, \tau)$ to $Verify(X_k, Y_i)$. By doing this change, $\mathscr{A}$ still sees the same result, so the probability of winning of $\mathscr{A}$ in this game is the same as in *Game 2*: $\Pr[S_3] = \Pr[S_2]$

Game 4. In this game, we change the way $\mathscr{C}$ computes $[\![\mathbf{HD'}_i]\!]$: In the orignal game 0, r was added to mask the value of HD. We want to prevent this r from being used anywhere in the game, so we replace $[\![r]\!]$ by $[\![0]\!]$. This change does not affect $\mathscr{A}$'s success probability: r only affects the plaintext inside $[\![\mathbf{HD'}]\!]$, since we do not use this plaintext anymore (it has been replaced in *Game 3*), this change therefore does not affect what the attacker sees. Again,

$\mathsf{Game}_3(n)$

1 : $D_k \xleftarrow{r} D_{bio}$
2 : $(\mathsf{sk}, \mathsf{pk}) \leftarrow KeyGen$
3 : $\mathbf{X}_k \xleftarrow{r} D_k$
4 : $[\![\mathbf{X_k}]\!] \leftarrow Enc(\mathbf{X}_k, \mathsf{pk})$
5 : For $i = 1, \ldots, q$:
6 : $\mathbf{Y}_i \leftarrow \{0,1\}^n$
7 : $[\![\mathbf{Y}_i]\!] \leftarrow Enc(\mathbf{Y}_i, \mathsf{pk})$
8 : $proverView1 \leftarrow \textbf{ZKPSimulator}()$
9 : $[\![\mathbf{HD}_i]\!] = EvalDistance([\![\mathbf{X}_k]\!], [\![\mathbf{Y}_i]\!])$
10 : $r_i \xleftarrow{r} \mathbb{Z}_{HD}, [\![\mathbf{HD'}_i]\!] = Add([\![\mathbf{HD}_i]\!], [\![r_i]\!])$
11 : $HD'_i \leftarrow Dec([\![\mathbf{HD'}_i]\!], \mathsf{sk})$
12 : $proverView2 \leftarrow \textbf{ZKPSimulator}()$
13 : $HD_i \leftarrow HD'_i - r_i$
14 : $res_i \leftarrow Verify(X_k, Y_i)$
15 : $\beta \leftarrow \mathscr{A}(proverView1, proverView2, sk, \mathbf{Y}_i, [\![\mathbf{Y}_i]\!], res_i, HD'_i, [\![\mathbf{HD'}_i]\!])$

Fig. 4.8 Game 3 - Client Impersonation game

$Pr[S_4] = Pr[S_3] = Pr[S_2]$.

After *Game 4* is finished, we can see that all the messages available to the attacker can be simulated using only the verified bit $b = Verify(Y^{(i)}, X_k)$. We now have an attacker A' against the biometric impersonation with advantage:

$$\varepsilon_{bio} = Adv(A') = Pr[S_4] \geq (\varepsilon_{imp} - q(\varepsilon_{ZK1} + \varepsilon_{ZK2})),$$

which results in the claimed bound.

$\mathsf{Game}_4(n)$

1 : $D_k \xleftarrow{r} D_{bio}$
2 : $(\mathsf{sk}, \mathsf{pk}) \leftarrow KeyGen$
3 : $\mathbf{X}_k \xleftarrow{r} D_k$
4 : $[\![\mathbf{X_k}]\!] \leftarrow Enc(\mathbf{X}_k, \mathsf{pk})$
5 : For $i = 1, \ldots, q$:
6 : $\mathbf{Y}_i \leftarrow \{0,1\}^n$
7 : $[\![\mathbf{Y}_i]\!] \leftarrow Enc(\mathbf{Y}_i, \mathsf{pk})$
8 : $proverView1 \leftarrow \textbf{ZKPSimulator}()$
9 : $[\![\mathbf{HD}_i]\!] = EvalDistance([\![\mathbf{X}_k]\!], [\![\mathbf{Y}_i]\!])$
10 : $r_i \leftarrow 0, enc\mathbf{HD'}_i = Add([\![\mathbf{HD}_i]\!], [\![r_i]\!])$
11 : $HD'_i \leftarrow Dec([\![\mathbf{HD'}_i]\!], \mathsf{sk})$
12 : $proverView2 \leftarrow \textbf{ZKPSimulator}()$
13 : $HD_i \leftarrow HD'_i - r_i$
14 : $res_i \leftarrow Verify(X_k, Y_i)$
15 : $\beta \leftarrow \mathscr{A}(proverView1, proverView2, sk, \mathbf{Y}_i, [\![\mathbf{Y}_i]\!], res_i, HD'_i, [\![\mathbf{HD'}_i]\!])$

Fig. 4.9 Game 4 - Client Impersonation game

4.6 Result Evaluation

4.6.1 Parameters

We now consider how to set concrete parameters. From Lemma 7 and the analysis from [95], we can take $\sigma = 8$ to make the protocol secure against the lattice attacks ([92]), note that this setting does not cover circuit privacy, which will be discussed in the next chapter. We can take $t = 2048$ to define the plaintext space ring R_t, this is enough to cover Hamming Distance of bit strings up to 2048 bits. We need $n \geq 2048$ for the packing ciphertext. With these parameters, we need q to be approximately 60 bits to satisfy $16n^2t^2\sigma^4 < q$. The proposed scheme works with only 1 level of homomorphic multiplication for the proposed set of parameters $(n = 2048, q \approx 2^{60}, t = 2048, \sigma = 8)$, the proof of concept implementation can be found in [github].

Number of rounds to repeat for each Zero Knowledge Proof: We are aiming at matching the security of biometrics systems with False Acceptance Rate of 10^{-3}, hence, with the current ZKP technique whose soundness error is $2/3$, we need to repeat the proof 17 times to achieve such level of security. Following the the communication size computed from our result $(2(n \log q) \sum l_i)$, where $\sum l_i \approx \sigma$ and the other parameters chosen as above, the communication size of the two Zero Knowledge Proofs required by the protocol is approximately 8MB to provide security against a malicious client model.

4.6.2 Limitations and open problems

We expose the HD to the server in the last step of the protocol and let the server do the threshold comparison operation in the plaintext domain. We assume that, given such a value, the server is not able to acquire any information about the original bit strings template (assuming the provable CPA-secure ciphertexts of the underlying cryptosystem). This assumption depends on the feature extraction function and would hold if the HD between a registered user's biometric template and another sample of that user's template is statistically independent of the registered template. Comparing the HD with threshold homomorphhically proves much less efficient, and removing this assumption is thus left as a topic of next chapters.

Also, we assume an honest but curious server, which is reasonable against passive exposure attacks. Active attacks are harder to carry out undetected and also slower, provided the server can be audited regularly. We emphasize that previous quantum resistant protocols also made this assumption, and were not even able to resist passive honest but curious trusted parties. Defending privacy against a malicious server is left as an open problem.

Finally, the communication size of the Stern-based ZKP protocol is large, due to the round soundness error $2/3$ (many communication rounds will be needed for security). We show how to reduce this overhead in chapter 6.

Chapter 5

The Second Protocol - Covering Circuit Privacy

5.1 Introduction

This chapter discusses Circuit Privacy, which is an important aspect of Homomorphic Encryption, and how it affects the security of our first variant of the protocol. We also describe a new technique being used to improve the security proof of the protocol, to make sure that we can choose parameters small enough for the system for practical reasons, but still preserve the security level. In particular, we show how to use the infinity-order Renyi Divergence (RD) instead of traditional Statistical Distance in the proof at security against a malicious client with exposed secret key, in order to significantly lower the initial noise bound parameter, which will result in a smaller moduli q for the ring R_q used by the cryptosystem.

Circuit Privacy means that the ciphertext generated by a homomorphic operation does not reveal anything about the circuit it evaluates, except its output value. This property applies even to the party having generated the keys.

Consider a ciphertext product result in a BV cryptosystem (section 2.4.2).

$$mult(c,c') = (\mathbf{c_0c'_0}, \mathbf{c_0c'_1} + \mathbf{c_1c'_0}, \mathbf{c_1c'_1})$$

The noise term of this result ciphertext correlates to both $\mathbf{m}$ and $\mathbf{m}'$. For example, given $c = (\mathbf{br}_1 + \mathbf{m}, -\mathbf{ar}_1)$ and $c' = (\mathbf{br}_2 + \mathbf{m}', -\mathbf{ar}_2)$, the decryption result of $mult(c,c')$, which is its inner product with $(1, \mathbf{s}, \mathbf{s}')$, would include the noise term:

$$t^2\mathbf{e}^2\mathbf{r}_1\mathbf{r}_2 + t\mathbf{em}_2\mathbf{r}_1 + t\mathbf{em}_1\mathbf{r}_2 + \mathbf{m}_1\mathbf{m}_2$$

In many contexts, a leakage of this kind might be inconsequential for a genuine user with a secret key, because s/he is supposed to know the key and decrypt the message. However, in many other contexts, especially in multi-factor authentication scenarios, this is not the case. For instance, in our system, we do not want the client to know the unmasked Hamming Distance value, so that an attacker with a stolen device and a secret key cannot derive information about the data stored in the server from the enrolment stage. We propose to mask this data by homomorphically adding the ciphertext $Enc(HD, e)$ to $Enc(r, e')$ so as to refresh the noise terms while masking the Hamming Distance at the same time (e and e' are the noises in the resulting ciphertext). The question is how far from the original noise distribution we can move away without compromising correctness, if this new security measure is applied.

Let e denotes the noise in the original ciphertext $Enc(HD, e)$, for some fixed e, let D_1 be the probability distribution of the new shifted noise $e + e'$ of $Enc(HD + r, e + e')$ with respect to the choice of the mask noise e' from Gaussian distribution with standard deviation σ. Let D_2 be the probability distribution of the unshifted noise e'. We observe that the 2 Gaussian distributions D_1 and D_2 are identical with the same standard deviation σ and different means. Let r_0 be the shift in the means of D_1 and D_2. Our goal is to set up parameter r_0, such that D_1 and D_2 are computationally indistinguishable while keeping other parameters of the

cryptosystem within practical performance thresholds. Statistical Distance (SD) is normally used to measure the difference of distributions and can be shown to be

$$SD(D_1, D_2) = \frac{1}{2}\sum_{x \in X} |D_1(x) - D_2(x)| \leq K \times \frac{r_0}{\sigma}$$

where K is a constant. We quickly see that in order for SD to be indistinguishable, ($SD < \varepsilon \approx \frac{1}{2^\lambda}$, where λ is the security parameter), the standard deviation of the mask noise needs to be larger than r_0 by an exponential factor in λ, that is $\sigma \geq K r_0 2^\lambda$.

In the work of [10], the authors proposed Renyi Divergence (RD) as an alternative to measure distributions' closeness and its applications to security proofs. $R_a(D_1 \| D_2)$ of order a between D_1 and D_2 is defined as the expected value of $(D_1(x)/D_2(x))^{a-1}$ over the randomness of x, sampled from D_1.

$$R_a(D_1 \| D_2) = \left(\sum_{x \in D_1} \frac{D_1(x)^a}{D_2(x)^{a-1}} \right)^{\frac{1}{a-1}}$$

Similar to SD, RD is useful in our context with its *Probability Preservation* property (we refer readers to [10] for detailed formal descriptions): Given D_1 and D_2 as described, for any event E, for instance, E may be the winning probability of an attacker in the distinguishing game, the probability of the event with respect to D_2 being bound by

$$D_2(E) \geq D_1(E)^{\frac{a}{a-1}} / RD_a(D_1 \| D_2) \tag{5.1}$$

Particularly, if we look at the second order ($a = 2$) of RD as done in previous works, we would have $D_2(E) \geq D_1(E)^2 / RD_2(D1 \| D_2)$. Provided that the distribution functions on lattices of D_1 and D_2 are discrete Gaussian, which is of the form $\rho_\sigma(x) = \frac{1}{\sigma} e^{-\pi \frac{x^2}{\sigma^2}}$, $RD_2(D_1 \| D_2) = e^{2\pi \frac{r_0^2}{\sigma^2}} \approx e^{2\pi}$, when r_0 is much smaller than σ. This means that when switching from D_1 to D_2, the success probability of E will be at least the old probability to

the power of 2 divided by some constant. This squaring factor brings about a big trade-off: for example, in our protocol, we would need to use $FAR = 2^{-20}$ in the non-privacy biometric settings to get $FAR = 2^{-10}$ into our scheme.

We aim at a solution to reduce this concrete security loss factor of the security proof. The idea is to look at RD_∞ instead of at RD_2: from equation (5.1), we can infer that when a is large, $\frac{a}{a-1}$ approaches 1. However, for usual Gaussian distributions, the value of RD_∞ is also infinity (not a constant $e^{2\pi}$ like in RD_2 when $a = 2$). This is due to the ratios $\frac{D_1(x)}{D_2(x)}$, which become large when sample is in the extreme tails of the distributions. Our idea is to truncate the distribution when doing noise sampling: if we get a noise value that is too close to the tail, we reject it and sample again. As a result, the RD_∞ can be bounded by a finite amount. The truncated distribution also scale up slightly (the small noises show higher probability when sampling), but this does not have a high impact on security.

5.2 Context and Related Work

5.2.1 Definitions

Definition 28 (Renyi Divergence)**.** *Let $Supp(D) = \{x : D(x) \neq 0\}$ denote the support for a probability distribution D. For any two discrete probability distributions P and Q such that $Supp(P) \subseteq Supp(Q)$ and $a \in (1, +\infty)$, the Renyi divergence of order a is defined by*

$$RD_a = \left(\sum_{x \in Supp(p)} \frac{P(x)^a}{Q(x)^{a-1}} \right)^{\frac{1}{a-1}}$$

When $a = 2$, the subscript is normally omitted, the Renyi divergence of order $+\infty$ is defined by $RD_\infty = max_{x \in Supp(P)} \frac{P(x)}{Q(x)}$. The definitions are extended in the natural way to continuous distributions. The following properties are considered analogues of those of Statistical Distance. Proofs can be found in [82].

Lemma 8. *Let $a \in [1,+\infty]$. Let P and Q denote distributions with $Supp(P) \subseteq Supp(Q)$. Then the following properties hold:*

Log. Positivity. $RD_a(P||Q) \geq RD_a(P||P) = 1$.

Data Processing Inequality *$RD_a(P^f||Q^f) \leq RD_a(P||Q)$ for any function f, where P^f and Q^f denote the distribution of $f(y)$ induced by sampling $y \xleftarrow{r} P$ and $y \xleftarrow{r} Q$*

Multiplicative *Assume P and Q are two distributions of a pair of random variables (Y_1, Y_2). For $i \in \{1,2\}$, let P_i and Q_i denote the marginal distribution of Y_i under P and Q, and let $P_{2|1}(\cdot|y_1)$ and $Q_{2|1}(\cdot|y_1)$ denote the conditional distribution of Y_2 given that $Y_1 = y_1$. Consequently:*

- *$RD_a(P||Q) = RD_a(P_1||Q_1) \cdot RD_a(P_2||Q_2))$ if Y_1 and Y_2 are independent for $a \in [1,\infty]$.*
- *$RD_a(P||Q) \leq RD_\infty(P_1||Q_1) \cdot max_{y_1 \in X} RD_a(P_{2|1}(\cdot|y_1)||Q_{2|1}(\cdot|y_1))$*

Probability Preservation *Let $E \subseteq Supp(Q)$ be an arbitrary event. If $a \in (1,+\infty)$, then $Q(E) \geq P(E)^{\frac{a}{a-1}}/RD_a(P||Q)$ and $Q(E) \geq P(E)/RD_\infty(P||Q)$*

Weak Triangle Inequality *Let P_1, P_2, P_3 be three distributions with $Supp(P_1) \subseteq Supp(P_2) \subseteq Supp(P_3)$. Consequently:*

$$RD_a(P_1||P_3) \leq \begin{cases} RD_a(P_1||P_2) \cdot R_\infty(P_2||P_3), \\ RD_\infty(P_1||P_2)^{\frac{a}{a-1}} \cdot RD_a(P_2||P_3) \text{if } a \in (1,+\infty) \end{cases}$$

5.2.2 Related Work

Ling et al. [86] used RD as the framework for distinguishing problems in the k-LWE context, which is a variant of LWE in which the adversary is given extra information in the distinguishing attack game. The work of [108] used RD order 1 (Kullback-Leibler divergence) to improve the communication size requirement of BLISS [40]. [11] showed

how generic RD can be used as an alternative to the statistical distance in proofs for lattice-based cryptography. Bogdanov et al. [21] adapted the work of [11] to the Learning With Rounding problem. RD was used in [84] in the context of dynamic group signatures and also in [7] to replace LWE noise distribution by an easier to sample distribution.

5.3 Renyi Divergence Analysis technique

For the following analysis, let D_1 be a discrete Gaussian on $\mathbb{Z}$ with standard deviation parameter σ shifted by the constant $r_0 \in \mathbb{Z}$, while D_2 is a discrete Gaussian on $\mathbb{Z}$ with deviation parameter σ centered on zero, i.e. $D_1 = D_{\mathbb{Z},\sigma} + r_0$ and $D_2 = D_{\mathbb{Z},\sigma}$, where

$$D_{\mathbb{Z},\sigma}(x) = \frac{e^{-\pi \cdot x^2/\sigma^2}}{\sum_{z\in\mathbb{Z}} e^{-\pi \cdot z^2/\sigma^2}}, \text{ for } x \in \mathbb{Z}$$

To allow us to use RD_∞ we put tail-cut variants $D_1^{(cut)}$ and $D_2^{(cut)}$ of D_1 and D_2 into play, respectively with parameter k. The parameter k defines where D_1 and D_2 are cut, for example, we can set $k = 3$ to cut the distributions at 3 deviation parameters from the mean. So, we let $D_{\mathbb{Z},\sigma}^{(cut)}$ denote distribution $D_{\mathbb{Z},\sigma}$ tail-cut to the interval $[-k\cdot\sigma, k\cdot\sigma]$ by rejection sampling.

We let $D_1^{(cut)} = D_{\mathbb{Z},\sigma}^{(cut)} + r_0$ and $D_2^{(cut)} = D_{\mathbb{Z},\sigma}^{(cut)}$. Notice that the supports of $D_1^{(cut)}$ and $D_2^{(cut)}$ are different, namely $Supp(D_1^{(cut)}) = [-k\sigma + r_0, k\sigma + r_0]$ while $Supp(D_2^{(cut)}) = [-k\sigma, k\sigma]$. We assume, without loss of generality, that $r_0 > 0$. We would like to switch from distribution $D_1^{(cut)}$ to $D_2^{(cut)}$, but unfortunately $R_\infty(\overline{D}_1^{(cut)} \| D_2^{(cut)})$ is not finite, since $Supp(D_1^{(cut)})$ is not a subset of $Supp(D_2^{(cut)})$. To satisfy the latter condition, we first switch from $D_1^{(cut)}$ to $\overline{D}_1^{(cut)}$ by further cutting (by rejection sampling) the positive tail of $D_1^{(cut)}$ to ensure it does not go beyond the $k\sigma$ upper bound on tail of $D_2^{(cut)}$, and use a (mild condition) statistical distance step to lower bound $\overline{D}_1^{(cut)}(E)$. Then, in a second step using $Supp(\overline{D}_1^{(cut)}) = [-k\sigma + r_0, k\sigma] \subseteq [-k\sigma, k\sigma] = Supp(D_2^{(cut)})$, we derive a finite upper bound on $R_\infty(\overline{D}_1^{(cut)} \| D_2^{(cut)})$ to lower bound $D_2^{(cut)}(E)$. Details follow.

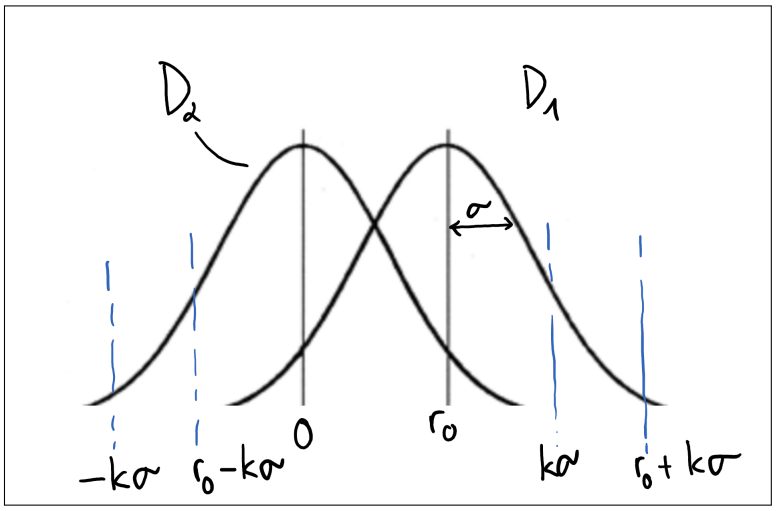


Fig. 5.1 Renyi Divergence tail cut

First SD Step. Since $Supp(D_1^{(cut)})$ is transformed into $\overline{D}_1^{(cut)}$ by rejection and resampling if a sample of $Supp(D_1^{(cut)})$ falls in $(k\sigma, k\sigma + r_0]$, we have

$$SD(D_1^{(cut)}, \overline{D}_1^{(cut)}) \leq D_1^{(cut)}((k\sigma, k\sigma + r_0]) = D_2^{(cut)}((k\sigma - r_0, k\sigma]) = D_{\mathbb{Z},\sigma}((k\sigma - r_0, k\sigma])/C_2$$

, where $C_2 = D_{\mathbb{Z},\sigma}([-k\sigma, k\sigma])$. Consequently,

$$\Delta \stackrel{\text{def}}{=} \frac{D_{\mathbb{Z},\sigma}((k\sigma - r_0, k\sigma])}{C_2} = \frac{\sum_{z \in (k\sigma - r_0, k\sigma])} \mathrm{e}^{-\pi z^2/\sigma^2}}{\sum_{z \in [-k\sigma, k\sigma])} \mathrm{e}^{-\pi z^2/\sigma^2}}.$$

For the numerator, we have the upper bound

$$\begin{aligned} \sum_{z \in (k\sigma - r_0, k\sigma])} \mathrm{e}^{-\pi z^2/\sigma^2} &\leq \int_{k\sigma - r_0}^{\infty} \mathrm{e}^{-\pi z^2/\sigma^2} dz \\ &\leq \sigma \cdot \mathrm{e}^{-\pi (k\sigma - r_0)^2/\sigma^2} \end{aligned}$$

The second inequality in the above bound is obtained by using the standard normal distribution tail upper bound

$$\int_{\gamma}^{\infty} \frac{1}{\sqrt{2\pi}\sigma} \cdot \mathrm{e}^{-z^2/\sigma^2} dz \leq \mathrm{e}^{-\gamma^2/(2\sigma^2)}, \text{for } \gamma \geq 0$$

For the denominator, we have the lower bound

$$\begin{aligned}\sum_{z\in[-k\sigma,k\sigma]} \mathrm{e}^{-\pi z^2/\sigma^2} &\geq 2\cdot \sum_{z\in[0,k\sigma])} \mathrm{e}^{-\pi z^2/\sigma^2} \\ &\geq 2\cdot (\int_0^{\infty} \mathrm{e}^{-\pi z^2/\sigma^2} dz - \int_{k\sigma}^{\infty} \mathrm{e}^{-\pi z^2/\sigma^2} dz) \\ &\geq \sigma\cdot (1-2\cdot \mathrm{e}^{-\pi(k\sigma-r_0)^2/\sigma^2})\end{aligned}$$

Therefore, $SD(D_1^{(cut)},\overline{D}_1^{(cut)}) \leq \Delta \leq \delta'/(1-2\delta') \leq 2\delta'$ if $\delta' \leq 1/4$, where $\delta' = \mathrm{e}^{-\pi(k\sigma-r_0)^2/\sigma^2}$. Defining $\delta = 2\delta'$, makes $\Delta \leq \delta$ if $\delta \leq 1/2$ and the conditions $r_0 \leq \sigma$ and $k \geq 1+\sqrt{1/\pi\cdot\ln(2/\delta)}$ hold. Therefore, by the probability preservation property, for any event E, $\overline{D}_1^{(cut)}(E) \geq D_1^{(cut)}(E) - \delta$ holds.

Second RD step. The desired RD of order ∞ is defined by

$$R_\infty(\overline{D}_1^{(cut)} \| D_2^{(cut)})) = \max_{x\in[-k\sigma+r_0,k\sigma]} \frac{\overline{D}_1^{(cut)}}{D_2^{(cut)}(x)}.$$

Observe that, for each $x \in [-k\sigma+r_0, k\sigma]$, we have $\overline{D}_1^{(cut)}(x) = C\cdot D_1^{(cut)}(x)$, where the normalization constant

$$\begin{aligned}C &= \frac{1}{1-D_1^{(cut)}((k\sigma,k\sigma+r_0])} \\ &= \frac{1}{1-D_2^{(cut)}((k\sigma-r_0,k\sigma])} \\ &= \frac{1}{1-\Delta}\end{aligned}$$

Where Δ is defined and upper bounded by δ above, under the assumed conditions on k and δ. Since the $D_1^{(cut)}(x)$ and $D_2^{(cut)}(x)$ are shifts of each other, they have the same rejection sampling normalization constant with respect to D_1 (resp. D_2). Therefore, $D_1^{(cut)}(x)/D_2^{(cut)}(x) = D_1(x)/D_2(x)$ for each x in the support of both $D_1^{(cut)}$ and $D_2^{(cut)}$, and

thus

$$\begin{aligned} R_\infty(\overline{D}_1^{(cut)} \| D_2^{(cut)}) &\leq \frac{1}{1-\delta} \cdot \max_{x \in [-k\sigma + r_0, k\sigma]} \frac{D_1(x)}{D_2(x)} \\ &= \max_{x \in [-k\sigma, k\sigma]} \frac{e^{\frac{-\pi(x-r_0)^2}{\sigma^2}}}{e^{-\pi\frac{x^2}{\sigma^2}}} \\ &= e^{\pi \cdot r_0^2/\sigma^2} \cdot \max_{x \in [-k\sigma + r_0, k\sigma]} e^{\frac{2\pi r_0}{\sigma^2} x} \end{aligned}$$

This is an exponential function yielding its max value at $x = k\sigma$:

$$R_\infty(\overline{D}_1^{(cut)} \| D_2^{(cut)}) = e^{1/(1-\delta)} \cdot e^{\pi \cdot r_0^2/\sigma^2 + 2\pi k \cdot r_0/\sigma}.$$

Since $0 < \delta \leq 1/2$, the first factor above is $\leq 1 + 2\delta \leq e^{2\delta}$. Also, a simple computation shows that the second factor is less than e if the condition $\sigma/r_0 \geq 4\pi \cdot k$ is satisfied using $k \geq 1$. We conclude, under the assumed parameter conditions, that $R_\infty(\overline{D}_1^{(cut)} \| D_2^{(cut)})) \leq e^{1+2\delta} = c'(\delta)$ is constant for a constant $\delta > 0$, so that, by the RD probability preservation property,

$$\begin{aligned} D_2^{(cut)}(E) &\geq \frac{1}{c(\delta)} \cdot \overline{D}_1^{(cut)}(E) \\ &\geq \frac{1}{c(\delta)} \cdot (D_1^{(cut)}(E) - \delta) \end{aligned}$$

Note that if $D_1^{(cut)}(E) = \varepsilon$, then, by choosing $\delta = \varepsilon/2$, we get $D_2^{(cut)}(E) \geq \frac{1}{2c(\delta)} \cdot \varepsilon$, and we only need $k \geq 1 + \sqrt{1/\pi \cdot \ln(2/\delta)}$ and σ/r_0 logarithmic in $1/\delta$, much smaller than σ/r_0 linear in $1/\delta$, which we would need if we were to use the 'SD only' analysis approach.

The above discussion immediately generalizes from the one-dimensional case of discrete Gaussian samples over $\mathbb{Z}$ to the m-dimensional case of discrete Gaussian samples over $\mathbb{Z}^m$, due to the independence of the m coordinates. The only change to the above argument is that the statistical distance in the 'SD step' can multiply by at most a factor m, whereas the RD in the 'RD step' above gets raised to the m'th power, where we replace r_0 by $\|\vec{r}_0\|_\infty$. We

compensate it by replacing the bound δ on Δ in the above analysis by the bound δ/m. We have therefore proved the following result used in our impersonation security proof, which improves upon the R_2-based analogue result for shifted Gaussians stated in [82].

Lemma 9. *For integer $m \geq 1$, real $\sigma > 0$,$k \geq 1$, real $0 < \delta \leq 1/2$ and vector $\vec{r}_0 \in \mathbb{Z}^m$, let $D_1^{(cut)} = D_{\mathbb{Z}^m,\sigma}^{(cut)} + \vec{r}_0$ and $D_2^{(cut)} = D_{\mathbb{Z}^m,\sigma}^{(cut)}$ be relatively shifted tail-cut discrete Gaussian distributions, where $D_{\mathbb{Z}^m,\sigma}^{(cut)}$ is the discrete Gaussian $D_{\mathbb{Z}^m,\sigma}$ with its tails cut to the support $[-k\sigma, k\sigma]^m$ by rejection sampling. If the conditions $k \geq 1 + \sqrt{1/\pi \cdot \ln(2m/\delta)}$ and $\sigma/\|\vec{r}_0\|_\infty \geq 4\pi \cdot k \cdot m$ hold, then, for any event E defined over the support of $D_1^{(cut)}$, it is the case that*

$$D_2^{(cut)}(E) \geq \frac{1}{2e^{1+2\delta}} \cdot \left(D_1^{(cut)}(E) - \delta \right).$$

5.4 The authentication protocol

The application to our protocol mainly happens in the step of masking the value of HD. In addition to masking HD with some value r, we also need to mask the correlated noise used in the encryption process. The rest of the protocol steps are identical to Section 4.5.1.

Setup. $\mathscr{U}$ and $\mathscr{S}$ initialize the parameters, taking into account the following categories:

Biometric Authentication System Parameters. These parameters are standard ones used by non privacy preserving biometric authentication systems:

- False Acceptance Rate (FAR) and False Rejection Rate (FRR)
- a: The maximum number of incorrect authentication attempts allowed by $\mathscr{S}$.
- τ: Threshold to compare the Hamming Distance to decide the authentication result.
- n': The bit-length of the encoded biometric data.

Ring-LWE based techniques parameters. These parameters are used in the lattice-based cryptosystem which provides client privacy against long term quantum attacks.

- λ: General security parameter of the cryptosystem (the adversary's winning chance in the CPA security game is $1/2^{\lambda}$)
- n: Integer n defining the plaintext and ciphertext rings. This will be refered to as the degree of the polynomial objects or dimension of the underlying lattice during correctness and security proofs.
- t: Integer t defining the plaintext space ring $R_t = \mathbb{Z}_t[x]/x^n + 1$.
- q: Integer q defining the ciphertext space ring $R_q = \mathbb{Z}_q[x]/x^n + 1$
- $\chi_{\alpha q}$: A distribution used to sample noises for LWE-based techniques. Typically, χ is a Gaussian distribution with standard deviation αq.
- δ: Renyi Divergence parameter for the security of noise masking.

Keygen. Keys are generated for $\mathscr{U}$:

- Secret key: $\mathbf{s} \xleftarrow{r} \chi^n_{\alpha q}$, $sk = (1, \mathbf{s}, \mathbf{s^2}, ...)$
- Public key: $pk = (\mathbf{p_0}, \mathbf{p_1})$, with $\mathbf{p_1} \xleftarrow{r} R_q$ and $\mathbf{p_0} = -\mathbf{p_1 s} - t\mathbf{e}$, with $\mathbf{e} \xleftarrow{r} \chi^n_{\alpha q}$.

Enrolment. $\mathscr{U}$ extracts the biometric template $\mathbf{x}$, the bit string $\mathbf{x}$ being represented as a ring element of R_t. The encryption is done by $[\![\mathbf{x}]\!] = (\mathbf{c_0}, \mathbf{c_1})$ and sent to $\mathscr{S}$.

Authentication. The following steps are required:

1. $\mathscr{U}$ extracts his biometric features again $\mathbf{y}$ to use them as the query. $\mathscr{U}$ sends $[\![\mathbf{y}]\!] = (\mathbf{c'_0}, \mathbf{c'_1})$ to $\mathscr{S}$.
2. ZKP for the first relation: $\mathscr{U}$ has to prove that $[\![\mathbf{y}]\!]$ is a valid encryption, that is, it encrypts a bit string under the BV cryptosystem using the corresponding secret

key. This is done by module **SternBV(A, y, x)** described in Sect. 4.4.1, where $\mathbf{A} = \begin{bmatrix} c_{y0}, t, 0, 1 \\ p_{y0}, 0, t, 0 \end{bmatrix}$, $\mathbf{X} = [\vec{s}, \tilde{e_y}, \vec{e_0}, y]^T$ and $\mathbf{Y} = [\mathbf{c_{y1}}, \mathbf{p_{y1}}]^T$.

3. HD Computation: $\mathscr{S}$ computes $[\![HD_{\mathbf{x},\mathbf{y}}]\!]$ using procedure 4.3.1. We note that the noise term $\mathbf{e_{HD}}$ of $[\![\mathbf{HD}, \mathbf{e_{HD}}]\!]$ can leak information about $\mathbf{x}$ when **HD** is decrypted. Therefore, an extra step is needed to secure the operation.
   - Sample $\mathbf{e_r} \xleftarrow{r} \chi_r^n$ such that $\|\mathbf{e_r}\|_\infty$ is big enough compared to $\|\mathbf{e_{HD}}\|_\infty$.(Section 5.3)
   - Compute $[\![\mathbf{r}, \mathbf{e_r}]\!]$ and carry out an homomorphic addition operation to mask both the values of **HD** and the noise $\mathbf{e_{HD}}$: $[\![\mathbf{HD'}, \mathbf{e'_{HD}}]\!] = [\![\mathbf{HD}, \mathbf{e_{HD}}]\!] + [\![\mathbf{r}, \mathbf{e_r}]\!]$

   The result $[\![\mathbf{HD'}]\!]$ is then sent to $\mathscr{U}$.
4. $\mathscr{U}$ decrypts $[\![\mathbf{HD'}]\!]$ and derives the actual value HD' from the first coefficient of the plaintext: $dec\,[\![\mathbf{HD'}]\!] = HD' + r_1 + r_2 + \cdots + r_{n-1}$. $\mathscr{U}$ sends HD' to $\mathscr{S}$.
5. $\mathscr{U}$ proves that it does the decryption honestly, this is done similarly to step 2.
6. $\mathscr{S}$ unmasks HD' and outputs the authentication result $HD \overset{?}{<} \tau$

5.4.1 Security Analysis

Theorem 12 (Server side security)**.** *Under the IND-CPA security of BV cryptosystems, and the zero-knowledge property of the Stern protocol, the proposed scheme satisfies (Honest But Curious) Server Privacy Security.*

Theorem 13 (Client side security)**.** *Under the IND-CPA security of BV cryptosystem and the soundness property of the underlying Stern protocol, the proposed scheme satisfies Impersonation and Multifactor Security. Concretely, for $\delta > 0$, the protocol is (q, c)-secure against impersonation with $c \leq c(\delta) + 3 \cdot c_1$, assuming the underlying non-private biometric protocol has impersonation probability ε_{bio} and the underlying Stern ZK protocols have*

knowledge error $\varepsilon_{ZK1}, \varepsilon_{ZK2}$ *such that* $q(\varepsilon_{ZK1} + \varepsilon_{ZK2}) + \delta \leq c_1 \cdot \varepsilon_{bio}$, $c(\delta) = 2e^{1+2\delta}$, *and the condition* $\sigma / r_0 \geq 4\pi knq$ *holds, for* $k = 1 + \sqrt{1/\pi \ln(2nq/\delta)}$ *and* r_0 *as an upper bound on the size of the noise in* $[\![\mathbf{HD}]\!]$.

The proof for Theorem 12 can be obtained as described in Section 4.5.2. For Theorem 13, there is a minor game change in the client impersonation model after game 4. We briefly recall the game sequence; the detailed diagrams of the changes in consecutive games can be found in Section 4.5.2.

Game 0 (Figure 4.6). Game 0 is the original impersonation game for type I attack.

Setup. $\mathscr{C}$ intiates $D_k \xleftarrow{r} D_{bio}$ and $X_k \xleftarrow{r} D_k$. $\mathscr{C}$ sets up (sk_k, pk_k) and launches $Enrol(k, X_k)$ to get $(sk_k, [\![\mathbf{X}_k]\!] = (pk_k, \mathbf{X}_k))$. $\mathscr{A}$ submits a type 1 attack query and receives sk_k from $\mathscr{C}$.

Query. $\mathscr{A}$ runs q authentication sessions. In each session $j = 1, \ldots, q$, $\mathscr{A}$ can choose $\mathbf{Y}_i \leftarrow \{0,1\}^n$ and sends $[\![\mathbf{Y}_i]\!]$. $\mathscr{A}$ and $\mathscr{C}$ run **ZKPoPK1**. $\mathscr{C}$ evaluates $[\![\mathbf{HD}_k]\!]$ then computes $[\![\mathbf{HD}'_k]\!]$. The resulting ciphertext is sent to $\mathscr{A}$, $\mathscr{A}$ decrypts $[\![\mathbf{HD}'_k]\!]$ and sends back the plaintext value of HD'_i. $\mathscr{A}$ and $\mathscr{C}$ run **ZKPoPK2**(). $\mathscr{C}$ computes $HD = HD' - r$, compares the result to τ, and sends back the authentication result **Accept** or **Reject** to $\mathscr{A}$.

Guess. At the end of the game, $\mathscr{A}$ outputs $\mathbf{Y}'$.

Next, we discuss following games, the plan being to proceed towards the final game, where everything related to X_k that $\mathscr{A}$ receives can be simulated without any knowledge about X_k (the only known information about X_k is the function $Verify(X_k, Y)$). Let $res_i = Verify(X_k, Y_i)$ and S_j be the event in the game j such that $res_j = Accept$.

In the *Game 1* and *Game 2*, the real ZKPs are replaced by the **ZKPSimulator**, by the end of *Game 2*, if in any authentication attempt i, none of the 2 games have been aborted, then, by

the correctness property of ZKP, the server's output is equal to the output of $Verify(X_k, Y_i)$. In other words, we have shown what we could get by just querying the oracle $Verify()$. Next, we want to simulate everything related to X_k the attacker can see using just that oracle.

Game 3. At the end of *Game 2*, $\mathscr{C}$ has the bit $b = Verify(Y^{(i)}, X_k)$. In this game, we can change res_i from $compare(HD_i, \tau)$ to $Verify(X_k, Y_i)$. By doing this change, $\mathscr{A}$ still sees the same result, so the probability of winning of $\mathscr{A}$ in this game is the same as in *Game 2*: $\Pr[S_3] = \Pr[S_2]$

Game 4. In this game, we change the way $\mathscr{C}$ computes $[\![\mathbf{HD'}_i]\!]$: In the orignal game 0, r was added to mask the value of HD. We want to prevent this r from being used anywhere in the game, so we replace $[\![r]\!]$ by $[\![0]\!]$. This change does not affect $\mathscr{A}$'s success probability: r only affects the plaintext inside $[\![\mathbf{HD'}]\!]$, since we do not use this plaintext anymore (it has been replaced in *Game 3*), this change therefore does not affect what the attacker sees. Again, $Pr[S_4] = Pr[S_3] = Pr[S_2]$.

Game 5. We modify the way $[\![\mathbf{HD}]\!]$ is computed in this game. Instead of calculating $[\![\mathbf{HD'}_i]\!] = [\![\mathbf{HD}_i]\!] + [\![\mathbf{r}_i]\!]$, the challenger chooses a random $HD' \xleftarrow{r} \mathbb{Z}_p$ and encrypts it with the noise used before. In this game, the plaintext has changed from being $r + HD$ to a uniform $HD' \in \mathbb{Z}_p$. Since r is also uniform in $\mathbb{Z}_p$, the attacker is confronted with a uniform plaintext in both cases. Therefore, $Pr[S_5] = Pr[S_4]$.

Game 6. Finally, we set $[\![\mathbf{HD'}_i]\!] = Enc(HD'_i, e_{HD'_i})$ for $e_{HD'} \xleftarrow{r} \chi_{mask}$ instead of $e_{HD} + e_r$. We replace the sum of the Gaussian noise with a random noise. In Section 5.3, we showed that $Pr[S_6] \geq \frac{1}{c(\delta)}(Pr[S_5] - q \cdot \delta)$, where $c(\delta) = RD(e_{HD} + e_r, e_{HD}) \leq 2 \cdot e^{1+2\delta}$ by Lemma 9 and by our assumption on the parameter's values. After we finish *Game 6*, we notice

that all the messages available to the attacker can be simulated with only the verified bit $b = Verify(Y^{(i)}, X_k)$. We now have an attacker A' against the biometric impersonation with advantage:

$$\varepsilon_{bio} = Adv(A') = Pr[S_6] \geq \frac{1}{c(\delta)}(\varepsilon_{imp} - q(\varepsilon_{ZK1} + \varepsilon_{ZK2}+) + \delta),$$

which gives the claimed bound $\varepsilon_{imp} \leq c(\delta) \cdot \varepsilon_{bio} + q \cdot (\varepsilon_{ZK1} + \varepsilon_{ZK2}) + \delta \leq (c(\delta) + c_1) \cdot \varepsilon_{bio}$ if $q(\varepsilon_{ZK1} + \varepsilon_{ZK2}) + \delta \leq c_1 \cdot \varepsilon_{bio}$.

5.4.2 Security Proof for Type II Impersionation attack

This proof can also be provided using a sequence of games between the challenger $\mathscr{C}$ and the adversary $\mathscr{A}$. We present a sequence of games as well as the relations among them to demonstrate the type II security model proof. The idea is that in this type of attack, sk_k is not used to compute the view of $\mathscr{A}$. On the other hand, the soundess of the zero-knowledge proof of knowledge **ZKPoPK** implies the existence of an efficient witness extractor algorithm, capable of extracting the witness (i.e. the secret key sk_k) from a cheating prover which succeds with probability non-negligibly higher than the knowledge error of the zero-knowledge proof, and thus contradicts the IND-CPA security of the BV encryption scheme.

Game 0. Game 0 is the original impersonation game for a type II attack, i.e., the same as Game 0 in the proof of security against Type I attacks, except that $[\![\mathbf{X_k}]\!]$ is given by $\mathscr{C}$ to $\mathscr{A}$ at the beginning of the game, rather than sk_k. For $j \in \{1, \ldots, q\}$, let res^j denote the result of the jth authentication protocol run between $\mathscr{A}$ and $\mathscr{C}$, and S_0 be the event in the game 0 such that $res^j = Accept$ for some $j = 1, \ldots, q$. We have $\Pr[S_0] = \varepsilon_{imp,II}$ as the type II success probability of $\mathscr{A}$.

Game 1. From the definition of event S_0 in Game 0, it follows that there exists some $j^* \in \{1,\ldots,q\}$ such that $\Pr[res^{j^*} = Accept] \geq \varepsilon_{imp,II}/q$. Furthermore, by an averaging argument, there must exist a set G of (D_k, X_k, pk_k) such that $\Pr[(D_k, X_k, pk_k) \in G] \geq \varepsilon_{imp,II}/(2 \cdot q)$, and for each $(D'_k, X'_k, pk'_k) \in G$, we have $\Pr[res^{j^*} = Accept | (D_k, X_k, pk_k) = (D'_k, X'_k, pk'_k)] \geq \varepsilon_{imp,II}/(2 \cdot q)$. By ε_{ZK1}-soundness of the zero-knowledge proof of knowledge **ZKPoPK1** [55], there exists a witness extractor algorithm that runs in expected time $T' = O(\text{poly}(n \log Q) \cdot T/(\varepsilon_{imp,II}/(2 * q) - \varepsilon_{ZK1}))$, where T denotes the run-time of $\mathscr{A}$ and outputs a witness containing sk_k for **ZKPoPK1**. Therefore, we obtain a (secret key recovery) attack algorithm against the IND-CPA security of the BV encryption scheme, with expected run-time T' and advantage $\varepsilon' \geq \varepsilon_{imp,II}/(2 \cdot q)$. Hence, if $\varepsilon_{imp,II}/(2 * q) - \varepsilon_{ZK1} > \varepsilon_{imp,II}/(4 * q)$, or equivalently, if $\varepsilon_{imp,II} > 2 \cdot q\varepsilon_{ZK1}$, we obtain a contradiction with the assumption the BV encryption scheme with parameters Q, n, σ is IND-CPA against attacks with expected time $O(\text{poly}(n \log Q) \cdot T/\varepsilon_{ZK1})$ and advantage $\geq \varepsilon_{ZK1}$. It follows under the latter assumption that $\varepsilon_{imp,II} \leq 2q\varepsilon_{ZK1} \leq 2c_1 \cdot \varepsilon_{bio}$, under the assumption that $q(\varepsilon_{ZK1} + \varepsilon_{ZK2} + \delta) \leq c_1 \cdot \varepsilon$.

5.5 Results

In the previous chapter, the following parameters were used for the cryptosystem: $n = 2048, q \approx 2^{70}, t = 2048, \sigma = 8$. This configuration does not cover circuit privacy as the error noise leaks information about the registered template. If the leakage is to be covered by the added noise with a magnitude computed from the Statistical Distance approach, we would need the noise added to be as large as $\mathscr{O}\left(2^{\lambda}\right)$, which results in increasing σ to $K.2^{\lambda}$. With that new noise added, the parameter q of the system needs to be adjusted to ensure correctness. According to the correctness requirement (lemma 7) , q needs to be approximately 280 bits.

With this new moduli, the overheads introduced to both computation and communication are significant. For example, in the ZKP protocol, the communication size of each round would be about 3MB and could not be considered practical for the current infrastructure

bandwidth, as we need to repeat each proof at least 17 rounds to achieve a security level corresponding to FAR $\approx 10^{-3}$ (and we have 2 proofs in our protocol, which will result in a total communication weight of approximately 100MB)

However, by applying the approach in this chapter and adding the noise according to Lemma 9, the new noise becomes $\sigma = 50$; with the proposed parameter values, the new moduli is approximately 70 bits and the communication overhead increases from 8MB to 16MB for the whole protocol. This is by far lower than the consumption of the Statistical Distance approach and can be considered practical in some of the current deployed networks (such as 100 Mbps cable internet or NBN).

In conclusion, by using Renyi Divergence as an alternative to measure the closeness of distributions instead of Statistical Distance in security proofs of lattice-based cryptography, one can efficiently reduce the parameters' sizes. This chapter introduced how a specific order of RD (infinity order) can positively affect the parameters' choices for the cryptosystem. In the next chapter, we start improving the security model by hiding the value of the distance from the server. All the results of this chapter will be kept for the purpose of covering circuit privacy as necessary in any steps concerning the server's side, to provide security against a malicious client model.

Chapter 6

The third protocol - Computing and Comparing HD Homomorphically

6.1 Introduction

In the previous chapter, we assumed that, given the Hamming Distance (HD) between the registered and queried binary bitstring templates, it is infeasible for an attacker to infer or acquire any information about the registered templates stored on the server. This assumption is plausible as long as the distributions of the HD between registered and the queried templates is not in any way correlated to the templates themselves (analyzing these distributions is out of scope of the project). However, in this chapter, we introduce our first attempt to hide this information from the server as well. Our results not only cause the protocol to work regardless of the aforementioned dependencies, it is also the first step we take to secure the authentication protocol against an *active* server security model. Although the communication size of this variant might not be suitable in practice for current network infrastructures, one can still find some generic techniques used within it to be helpful while applying them to balance the computation time and the communication size of lattice-based cryptographic protocols. The contributions of this variant include:

- A technique to compare Hamming Distance (HD) homomorphically by computing the Most Significant Bit (MSB) of a specific ciphertext.
- A packing methods conversion technique to transform a ciphertext encrypting a binary-encoded to a unary-encoded plaintext.
- Some extensions of the Zero Knowledge Proof (ZKP) technique we used in previous chapter to check the format of a message, in addition to the information carried by the plaintext itself.
- The combination of ZKP protocols to balance the communication size trade-offs when used with lattice-based cryptosystems.

6.2 Proposed Scheme

We first propose a secure fingerprint authentication scheme that combines a Somewhat Homomorphic Encryption (SHE) with a Zero-Knowledge Proof (ZKP) to provide privacy features with low FAR overhead. The scheme is secure under a hybrid model that assumes an active client and an honest but curious (HBC) server. The server $\mathscr{S}$ is considered to be HBC by assuming that it can be audited regularly. This scheme uses Hamming Distance (HD) as the main measure unit to determine the difference between two fingerprint templates. The detailed descriptions of each protocol step are discussed in later sections of the chapter. Together with notations according to previous chapter, we use extra notations regarding encryptions as follows:

- $Enc^{(1)}(m)$: The BV encryption of m as in previous chapter.
- $Enc^{(2)}(m)$: The BV encryption of x^m: $Enc^{(2)}(m) = Enc^{(1)}(x^m)$
- $Enc^{(3)}(m)$: The LWE encryption of m over $\mathbb{Z}_Q$ defined in section 2.3.3.

6.2.1 The protocol

Setup The server and the user run the setup process as follows:

- $\mathscr{S}$ invokes $params_{BV} \leftarrow SGen_{BV}(1^{\lambda})$
- We are currently using a BV cryptosystem ([27]). We refer the reader to section 2.4.2 for details of $params_{BV}$, the public key pk, the private key sk, as well as the operations of the cryptosystem.
- A user $\mathscr{U}_k$ invokes $(pk_k, sk_k) \leftarrow UGen_{BGV}(params_{BGV})$ and makes pk_k publicly available to $\mathscr{S}$.

Enrolment The enrolment process runs as follows.

- $\mathscr{U}$ uses a specific sensor and algorithm to extract his biometric template X. $\mathscr{U}$ encrypts X with his public key pk_k to get $T_k = Enc^{(1)}(X)$.
- $\mathscr{U}$ sends (k, T_k) to $\mathscr{S}$. Noted that k is used as an identity index for $\mathscr{U}$.
- $\mathscr{S}$ stores a tuple (k, T_k) as a record.

Authentication The authentication process for a user $\mathscr{U}_k$ is as follows.

1. $\mathscr{U}_k$ extracts his query template Y. He encrypts Y with his public key pk_k to obtain $Q_k = Enc^{(1)}(Y)$.
2. $\mathscr{U}_k$ sends (k, Q_k) to $\mathscr{S}$ and runs **ZKPValidEnc**$((\mathbf{Q_k}, \mathbf{pk_k}), (\mathbf{Y}, \mathbf{sk_k}))$ to prove the validity of Y. was detailed in Chapter 4.
3. $\mathscr{S}$ locates the record (k, T_k) and computes the encrypted Hamming Distance $C_{HD} = Enc^{(1)}(HD)$ of X and Y, using the homomorphic operation discussed in Section 4.3.

$$C_{HD} \leftarrow \textbf{EvalDistance}(T_k, Q_k).$$

4. $\mathscr{S}$ masks the C_{HD} by sampling $r \xleftarrow{r} \mathscr{P}$ and performs one homomorphic addition to get $C_{HD'} = Enc^{(1)}(HD+r) \leftarrow Enc^{(1)}(HD) + Enc^{(1)}(r)$. The result ciphertext is sent to $\mathscr{U}_k$.

5. $\mathscr{U}_k$ uses his private key sk_k to decrypt $C_{HD'}$ and sends the re-encryption $C_{HD'_0} = Enc^{(1)}(HD', 0, 0, \ldots, 0)$ back to $\mathscr{S}$. $\mathscr{U}_k$ also decomposes the plaintext result $HD' = HD + r$ into its binary representation:

$$HD' = b_0 + b_1 2^1 + \cdots + b_l 2^{l-1}$$

and sends $C_i = Enc^{(1)}(b_i)$ to $\mathscr{S}$ for $i = 0, \ldots, l-1$.

6. $\mathscr{U}_k$ and $\mathscr{S}$ run the **ZKPUnpack**$(C_{HD}, C_{HD'_0})$ and **ZKPBinDecomp**$(C_{HD'_0}, C_i)$ protocols to convince the server that $\mathscr{U}_k$ did follow the protocol transcript correctly. This is detailed in section 6.4.4.

7. $\mathscr{S}$ computes $C''_{HD} = Enc^{(2)}(2^l + t - HD) \leftarrow Enc^{(2)}(2^l + t) - Enc^{(2)}(HD + r) + Enc^{(2)}(r)$, where $Enc^{(2)}(HD + r)$ is computed by **ToUnary**(C_i). We note that $Enc^{(2)}$ is the "unary" mode of encryption with the message encoded in the exponent of the polynomial, which allows the Most significant bit (MSB) extraction on the ciphertext (this is detailed section 6.3.2).

8. $\mathscr{S}$ does $Enc^{(3)}(res) \leftarrow$ **MSBExtract**(C''_{HD}), this is the ciphertext of the authentication result, which is sent to $\mathscr{U}_k$. This is described in section 6.3.1.

9. $\mathscr{U}_k$ decrypts the result *res* and sends it to $\mathscr{S}$ (it will be either **Accepted** or **Rejected** by $\mathscr{S}$). $\mathscr{U}_k$ also runs another proof **ZKPCorrectDec**(res) to convince $\mathscr{S}$ that he did follow the protocol honestly (Section 6.4.4).

The proposed scheme satisfies the security notions defined in Sect. 3.1.

Theorem 14. *Under the IND-CPA security of a BV cryptosystem, and the zero-knowledge property of the Stern protocol, the proposed scheme satisfies an (Honest But Curious) Server Privacy's Security.*

Theorem 15. *Under the IND-CPA security of a BV cryptosystem and the soundness property of the underlying Stern protocol, the proposed scheme satisfies Impersonation Security. Concretely, for $\delta > 0$, the protocol is (q,c)-secure against impersonation with $c \leq c(\delta) + 3 \cdot c_1$, assuming the underlying non-private biometric protocol has impersonation probability ε_{bio} and the underlying Stern ZK protocols have knowledge errors $\varepsilon_{ZK1}, \ldots, \varepsilon_{ZK4}$ such that $q(\varepsilon_{ZK1} + \varepsilon_{ZK2} + \varepsilon_{ZK3} + \varepsilon_{ZK4}) + \delta \leq c_1 \cdot \varepsilon_{bio}$, $c(\delta) = 2e^{1+2\delta}$, and the condition $\sigma / r_0 \geq 4\pi knq$ holds, with $k = 1 + \sqrt{1/\pi \ln(2nq/\delta)}$ and r_0 being an upper bound on the size of the noise in C_{HD}.*

6.3 The Homomorphic tools

6.3.1 Extracting the Most Significant Bit homomorphically

We observe that $HD < \tau \iff MSB(2^l + \tau - HD) = 1$, where l is the bit-length of HD and MSB denotes the Most Significant Bit. This is our attempt to compare Hamming Distance homomorphically, the idea is to let the server compute homomorphically the ciphertext of $MSB(2^l + \tau - HD)$, then having the client to decrypt it and to send back the authentication result with a zero knowledge proof. This section discusses a variant of the technique from [41] to efficiently compute the MSB of the plaintext M given $Enc(M)$, we adapt this technique to work with BGV encryption rather than GSW encryption used in [41]. The main idea comes from the way we encode the message M in "unary" form before the encryption.

THE THIRD PROTOCOL

Client		**Server**
..... Enrolment		
Extract $\mathbf{x} \xleftarrow{r} D_k$		
$T_k = Enc^{(1)}(\mathbf{x}, \mathsf{pk})$	$\xrightarrow{(k, T_k)}$	Persist (k, T_k)
..... Authentication		
Extract $\mathbf{y} \xleftarrow{r} D_k$		
$Q_k = Enc^{(1)}(\mathbf{y}, \mathsf{pk})$	$\xrightarrow{(k, Q_k)}$	
	$\xrightarrow{\textbf{ZKPValidEnc}}$	$C_{HD} \leftarrow \textbf{EvalDistance}(T_k, Q_k)$
	$\xleftarrow{C_{HD'}}$	$C_{HD'} \leftarrow Enc^{(1)}(HD, e_0) + Enc^{(1)}(r, e_r)$
$HD' \leftarrow Dec(C_{HD'}, \mathsf{sk})$		
$C_{HD'_0} \leftarrow Enc^{(1)}(HD', 0, \ldots, 0)$	$\xrightarrow{C_{HD'_0}}$	
$HD' = b_0 + b_1 2 + \cdots + b_l 2^{l-1}$		
$C_i \leftarrow Enc^{(1)}(b_i)$	$\xrightarrow{C_i}$	
	$\xrightarrow{\textbf{ZKPUnpack}(C_{HD}, C_{HD'_0})}$	
	$\xrightarrow{\textbf{ZKPBinDecomp}(C_i, C_{HD'_0})}$	$C''_{HD} \leftarrow Enc^{(2)}(2^t + \tau - HD)$
	$\xleftarrow{C_{res}}$	$C_{res} = Enc^{(3)}(res) \leftarrow \textbf{MSBExtract}(C''_{HD})$
$res = Dec(C_{res})$	$\xrightarrow{\textbf{ZKPCorrectDec}(res)}$	

Fig. 6.1 The Third Protocol

Given $M \in \mathbb{Z}_{2n}$, we observe that

$$\begin{cases} MSB(M) = 0 \iff M \in [0,n) \\ MSB(M) = 1 \iff M \in [n,2n) \end{cases}$$

Assume that we encrypt M using the SHE scheme (section 2.4.2). With the message space R_t and assuming that $2n < t < q$, we can encode M as a ring element $m \in R_t$ as follows

$$m(x) = 0x^0 + 0x^1 + \cdots + 1x^M + \cdots + 0x^{n-1} \text{ if } M \in [0,n)$$

or, due to $x^n = -1$ in R_q, we can also encode M as

$$m(x) = 0x^0 + 0x^1 + \cdots - 1x^M + \cdots + 0x^{n-1} \text{ if } M \in [n,2n)$$

The ciphertext $Enc^{(2)}(M)$ has this form $(\mathbf{c} = \mathbf{p_0}\mathbf{u} + t\mathbf{g} + \mathbf{m}, \mathbf{c}' = \mathbf{p_1}\mathbf{u} + t\mathbf{f})$, where $(\mathbf{p_0}, \mathbf{p_1})$ is the public key and $\mathbf{u}, \mathbf{f}, \mathbf{g} \xleftarrow{r} \chi_{\alpha q}$. We denote $rot(\mathbf{c}) \in \mathbb{Z}_q^{n \times n}$ as being an anti-circulant square matrix, whose first column is $\mathbf{c}$, the other columns being the cyclic rotations of $\mathbf{c}$ with the cycled entries negated

$$rot(\mathbf{c}) = \begin{bmatrix} c_0 & -c_{n-1} & -c_{n-2} & \dots \\ c_1 & c_0 & -c_{n-1} & \dots \\ \dots & \dots & \dots & \dots \\ c_{n-1} & c_{n-2} & c_{n-3} & \dots \end{bmatrix}$$

It's easy to see that $rot(\mathbf{cu}) = rot(\mathbf{c})\vec{u}$. Let $rot(\mathbf{c})[0]$ to be the first column of $rot(\mathbf{c})$.

Lemma 10. *Given $M \in Z_{2n}$ and $(\mathbf{c}, \mathbf{c}') = Enc^{(2)}(M)$ as the first level ciphertext of M from the BV scheme, let $\vec{1} = \{1, 1, \dots, 1\} \in \mathbb{Z}^n$. The transformed ciphertext $(l, l') \leftarrow (\vec{1}rot(\mathbf{c})[0], \vec{1}rot(\mathbf{c}'))) \in (\mathbb{Z}_Q, \mathbb{Z}_Q^n)$ encrypts the MSB information of M under $Enc^{(3)}$.*

Algorithm 3 Most Significant bit extraction

Input: $\mathbf{c} = Enc^{(2)}(M)$
Output: $EncLWE((-1)^{MSB(M)})$

1: **procedure** MSBEXTRACT(**c**)
2: $allOne = \{1,\ldots,1\}$
3: $rot_{c_0} \leftarrow rot(c_0)$
4: $lwe_0 \leftarrow allOne \cdot rot_{c_0}[0]$
5: $rot_{c_1} \leftarrow rot(c_1)$
6: $lwe_1 \leftarrow allOne \cdot rot_{c_1}$
7: **return** (lwe_0, lwe_1)

Proof. The following is the case:

$$\begin{aligned}
\vec{1} \cdot rot(\mathbf{c})[0] \in \mathbb{Z}_Q &= \vec{1}\overrightarrow{\mathbf{p_0 u} + t\mathbf{g} + \mathbf{m}} \\
&= \vec{1} \cdot \overrightarrow{p_0 u} + t\vec{1} \cdot \vec{g} + \vec{1} \cdot \vec{m} \\
&= -\vec{1}(rot(p_1 s)\vec{u} + t.rot(e)\vec{u}) + t\vec{1} \cdot \vec{g} + (-1)^{MSB(M)} \\
&= -\vec{1} rot(p_1)\vec{u}\vec{s} - t.rot(e)\vec{u} + t\vec{1} \cdot \vec{g} + (-1)^{MSB(M)}
\end{aligned}$$

Provided that $\mathbf{c}' = \mathbf{p_1 u} + t\mathbf{f}$, the decryption $\langle (l, l'), (1, \mathbf{s}) \rangle \mod t$ is ± 1. This implies the MSB information of M as discussed. We use here the relation $\vec{1} \cdot \vec{m} = (-1)^{MSB(M)}$ thanks to the "unary" encoding of M discussed previously. □

6.3.2 Converting from binary-encoded to unary-encoded plaintext

This section discusses a linear transformation to map a message $b \in \{0,1\}$ onto a message x^{jb} for $j \geq 1$. Let $T : cx + d = y$ be the linear transformation. We want T to map $0 \rightarrow x^{j0}$ and

$1 \rightarrow x^j$, or

$$\begin{cases} c.0 + d = 1 \\ c + d = x^j \end{cases} \Leftrightarrow \begin{cases} c = x^j - 1 \\ d = 1 \end{cases}$$

Due to the homomorphism property of BV cryptosystems , we can apply T in the ciphertext domain to obtain $Enc(x^{jb})$ given $Enc(b)$, for $b \in \{0,1\}$:

$$\begin{aligned} Enc^{(2)}(j.b) = Enc^{(1)}(x^{jb}) &= Enc^{(1)}(c)Enc(b) + Enc^{(1)}(d) \\ &= Enc^{(1)}(x^j - 1)Enc^{(1)}(b) + Enc^{(1)}(1) \end{aligned} \tag{6.1}$$

From the implementation point of view, this operation can be done faster by ensuring $\mathbf{u}, \mathbf{f}, \mathbf{g} \leftarrow 0$ instead of sampling them from χ during $Enc(x^j - 1)$ and $Enc(1)$, which results in $Enc^{(1)}(x^j - 1) = (x^j - 1, 0)$ and $Enc^{(1)}(1) = (1,0)$. These are still valid encryptions and will not affect the correctness of Eq. (6.1).

This submodule is used in our protocol at authentication step 7. Recall that the server needs to compute $Enc^{(2)}(HD')$, which is the encryption of HD' in the "unary" mode of encoding (the message is encoded in the exponent instead of in the coefficient of the polynomial). Given the encryptions $Enc(b_i)$ of the bits b_i of $HD' = \sum_{j=0}^{l-1} b_j 2^j$, the server can convert $Enc(b_j)$ to $Enc(2^i b_i)$ $Enc(x^{jb})$ and perform

$$Enc^{(2)}(HD') = Enc^{(1)}(x^{HD'}) = \prod_{i=0}^{l-1} Enc^{(1)}(x^{b_i 2^i})$$

Note that this operation involves $log(l)$ levels of homomorphic multiplication, where l is the bit length of the Hamming Distance.

Algorithm 4 Binary to Unary ciphertext

Input: $\mathbf{c_i} = Enc(b_i)$
Output: HD

1: **procedure** TOUNARY($\mathbf{c_i}$)
2: **for** $i = 0, \ldots, l-1$ **do**
3: let $j = 2^i$
4: let $\mathbf{hd_i} \leftarrow (x^j - 1, 0) \times \mathbf{c_i} + (1, 0)$
5: let $\mathbf{hd} \leftarrow \mathbf{hd}_0 \times \mathbf{hd}_1 \times \cdots \times \mathbf{hd}_{l-1}$
6: **return hd**

6.4 The Zero Knowledge Tools

6.4.1 ZKPoPK of Regev Cryptosystem

In this section, we first review an application of the [85] technique to apply ZKPoPK on Regev Cryptosystems, then we elaborate on the variants to be used in the protocol. Given parameters q, m, n, t, χ of a typical LWE-based cryptosystem, we can describe a variant of Regev's system as follows:

Kengen. A secret key $\vec{s}$ can be chosen from χ^n. The public key is then generated as $pk = (p_0, p_1) = (\mathbf{A}, \mathbf{A}\vec{s} + t\vec{e})$. Where $\mathbf{A} \xleftarrow{r} \mathbb{Z}_q^{m \times n}$ and $\vec{e} \xleftarrow{r} \chi^n$

Encrypt. Given a message $M \in \mathbb{Z}_t$, the ciphertext C is computed by first sampling a random vector $\vec{r} \in \chi^n$ and setting $C = (c_0, c_1) = (p_0\vec{r}, p_1\vec{r} + M)$

Decrypt. Given a ciphertext $C = (c_0, c_1)$, the message M can be recovered by computing $M = c_1 - c_0\vec{s} \mod t$

In [85], the ZKPoPK was attained by proving the encryption relation

$$R_{Regev}^{q,m,n,t,\chi} = \{((p_0, p_1), (c_0, c_1), \vec{r} || M) \in (\mathbb{Z}_q^{m \times n} \times \mathbb{Z}_q^m) \times (\mathbb{Z}_q^n \times \mathbb{Z}_q) \times \mathbb{Z}_q^{n+1} : (c_0 = p_0\vec{r}) \wedge (c_1 = p_1\vec{r} + M)\}$$

Let $\mathbf{A}' = \begin{bmatrix} p_1, 1 \\ p_0, 0 \end{bmatrix}$, and $\mathbf{y} = \begin{bmatrix} c_1 \\ c_0 \end{bmatrix}$ be public parameters in the proof and let $\mathbf{x} = \begin{bmatrix} \vec{r} \\ M \end{bmatrix}$ be the *Prover*'s witness. We observe that $\mathbf{A}'\mathbf{x} = \mathbf{y} \mod q$, that is, $\mathbf{x}$ is a solution to the ISIS problem defined by $(\mathbf{A}', \mathbf{y})$ and that we can use the **SternExt** protocol to obtain an efficient ZKPoPK. This works in a symmetric key setting, where the *Prover* knows the random $\vec{r}$ to use as his witness. In our context, the client does not know that $\vec{r}$ as encryption was set by the server. Following the decryption equation:

$$\begin{aligned} c_1 - c_0\vec{s} &= p_1\vec{r} + M - p_0\vec{r}\vec{s} \\ &= \mathbf{A}\vec{s}\vec{r} + t\vec{e}\vec{r} + M - \mathbf{A}\vec{r}\vec{s} \\ &= t\tilde{e} + M, \end{aligned}$$

we can write out the decryption relation as

$$R^{q,m,n,t,\chi}_{Regev,dec} = \{((p_0,p_1),(c_0,c_1),\vec{s},\vec{e},\tilde{e},M) \in (\mathbb{Z}_q^{m\times n} \times \mathbb{Z}_q^m) \times (\mathbb{Z}_q^n \times \mathbb{Z}_q) \times \chi^n \times \chi^n \times \chi \times \mathbb{Z}_Q : \tag{6.2}$$

$$(p_1 = p_0\vec{s} + t\vec{e}) \wedge (c_1 = c_0\vec{s} + t\tilde{e} + M)\}$$

In this situation, we can let $\mathbf{A}' = \begin{bmatrix} c_0, t, 0, 1 \\ p_0, 0, t, 0 \end{bmatrix}$ and $\mathbf{y} = \begin{bmatrix} c_1 \\ p_1 \end{bmatrix}$ be the public parameters and let $\mathbf{x} = \begin{bmatrix} \vec{s} \\ \tilde{e} \\ \vec{e} \\ M \end{bmatrix}$, further applying the **SternExt** to obtain the ZKPoPK . We note that the two separate rows of $\mathbf{A}'$ prove the two separate relations in (6.2): The *Prover* needs to prove that he knows a secret $\vec{s}$ that can decrypt (c_0, c_1); he also needs to prove that the secret key $\vec{s}$ is also the one corresponding to the public key (p_0, p_1).

6.4.2 ZKPoPK of BV cryptosystem

In this section, we discuss ZKPoPK for a ring variant of the Regev scheme discussed above, which is the BV system that we use in our application context (section 2.4.2). We will refer to this proof as **ZKPValidEnc**. Recall that our public key is a pair of ring elements $pk = (\mathbf{p_0}, \mathbf{p_1})$, where $\mathbf{p_0}, \mathbf{p_1} \in R_Q$ and $\mathbf{p_1 s} + t\mathbf{e} = -\mathbf{p_0}$. This is one of the relation that the client will need to prove later on. Next, given a ciphertext $c = (\mathbf{c_0}, \mathbf{c_1})$, the original plaintext $\mathbf{m} \in R_t$ can be recovered by $\mathbf{m} = \mathbf{c_0} + \mathbf{c_1 s} \mod t$ (spelled out as $\mathbf{c_1 s} + \mathbf{c_0} = \mathbf{m} + t\mathbf{e}'$, or $\mathbf{c_1 s} - t\mathbf{e}' - \mathbf{m} = -\mathbf{c_0}$). In summary, the relation required for the ZKPoPK is:

$$R_{BV}^{Q,n,t,\chi} = \{((\mathbf{c_0}, \mathbf{c_1}), (\mathbf{p_0}, \mathbf{p_1}), \mathbf{s}, \mathbf{e}', \mathbf{e}, \mathbf{m} \in (R_Q \times R_Q) \times (R_Q \times R_Q) \times \chi^n \times \chi^n \times \chi^n \times R_t : \tag{6.3}$$

$$(\mathbf{p_1 s} + t\mathbf{e} = -\mathbf{p_0}) \wedge (\mathbf{c_1 s} - t\mathbf{e}' - \mathbf{m} = -\mathbf{c_0})\}$$

We can proceed in a way similar to what appears in section 6.4.1, where we tried to derive the ISIS relation ($\mathbf{Ax} = \mathbf{y} \mod q$) from the above relation and obtain the ZKP accordingly. The matrix $\mathbf{A}$ should be derived from $\mathbf{T} = \begin{bmatrix} \mathbf{c_1}, -t, 0, -1 \\ \mathbf{p_1}, 0, t, 0 \end{bmatrix}$ in order to obtain (6.3). Note that with $\mathbf{c_1}, \mathbf{p_1} \in R_Q$, we can construct $\mathbf{A}$ by replacing from $\mathbf{T}$: $\mathbf{c_1}$ and $\mathbf{p_1}$ are replaced by $rot(\mathbf{c_1})$ and $rot(\mathbf{p_1})$, constants are replaced by the product of the themselves by the identity matrix $\mathbf{I}$. Recall that $rot(\mathbf{c}) \in \mathbb{Z}_q^{n \times n}$ is defined to be an anti-circulant square matrix, whose first column is $\mathbf{c}$, the other columns being the cyclic rotations of $\mathbf{c}$ with the cycled entries negated

$$rot(\mathbf{c}) = \begin{bmatrix} c_0 & -c_{n-1} & -c_{n-2} & \dots \\ c_1 & c_0 & -c_{n-1} & \dots \\ \dots & \dots & \dots & \dots \\ c_{n-1} & c_{n-2} & c_{n-3} & \dots \end{bmatrix}$$

So, the matrix $\mathbf{A}$ is of the following form:

$$\begin{bmatrix} rot(\mathbf{c_1}) & -t\mathbf{I} & \mathbf{0} & -\mathbf{I} \\ rot(\mathbf{p_1}) & \mathbf{0} & t\mathbf{I} & \mathbf{0} \end{bmatrix} \tag{6.4}$$

By constructing the matrix $\mathbf{A}$ this way, we can let $\mathbf{x} = \begin{bmatrix} \mathbf{s} \\ \mathbf{e'} \\ \mathbf{e} \\ \mathbf{m} \end{bmatrix}$, $\mathbf{y} = \begin{bmatrix} -\mathbf{c_0} \\ -\mathbf{p_0} \end{bmatrix}$ and come up with the original ISIS relation $\mathbf{Ax} = \mathbf{y} \mod Q$. Again, we can use the **SternExt** protocol (section 2.5.3) to obtain the ZKPoPK with $\mathbf{x}$ being the *Prover*'s witness and $\mathbf{A}, \mathbf{y}$ being the public parameters (Algorithm 5).

Algorithm 5 ZKPoPK for BV

1: **procedure** ZKPBV$((\mathbf{c}, pk), (\mathbf{m}, \mathbf{s}, \mathbf{e}, \mathbf{e'}))$
2: $rot_{c_1} \leftarrow rot(\mathbf{c_1})$
3: $rot_{p_1} \leftarrow rot(\mathbf{pk_1})$
4: let I be the $n \times n$ identity matrix
5: let Z be the $n \times n$ zero matrix
6: $\mathbf{A} \leftarrow ((rot_{c_1}, -tI, Z, -I), (rot_{p_1}, Z, tI, Z))$
7: $\mathbf{x} \leftarrow (\mathbf{s}, \mathbf{e'}, \mathbf{e}, \mathbf{m})$
8: $\mathbf{y} \leftarrow (-\mathbf{c_0}, -\mathbf{pk_0})$
9: **Return SternExt**$(\mathbf{A}, \mathbf{x}, \mathbf{y})$

6.4.3 ZKP of plaintext with zero coefficients

This section extends the previous proof with several submodules. Firstly, we need a proof to convince the server that our ciphertext encrypts the message of the form $\mathbf{m}(\mathbf{x}) = 0 + m_1 x^1 + m_2 x^2 + \cdots + m_{n-1} x^{n-1}$. Next, we need another proof to convince about the knowledge of encryption of $\mathbf{m}(\mathbf{x}) = m_0 + 0x^1 + 0x^2 + \cdots + 0x^{n-1}$. We also use some other proofs for

messages containing only 1s or 0s. In other words, we want the *Prover* to convince the $Verifier$ about the format correctness of the plaintext, in addition to asserting knowledge of the plaintext itself.

Proving $\mathbf{m} = \sum_{i=0}^{l}(m_i x^i) \wedge m_0 = 0$ (ZKPExt1). If and only if the *Prover* P has the message **m** in this format, he can compute:

$$rot(\mathbf{c_0})\mathbf{s} - t\mathbf{Ie}' - \begin{bmatrix} 1 \\ 0 \\ \cdots \\ 0 \end{bmatrix} \mathbf{m_0} = -\mathbf{c_0}$$

$$\iff rot(\mathbf{c_0})\mathbf{s} - t\mathbf{Ie}' - \begin{bmatrix} m_0 \\ 0 \\ \cdots \\ 0 \end{bmatrix} = -\mathbf{c_0}$$

In order to set up this proof, we can proceed similarly to what was proposed in section 6.4.2, just a minor modification is needed in the matrix represented in (6.4): We completely remove all the last $(n-1)$ columns of the last n columns during the construction of the matrix **A** (the new **A** will thus have dimension $n \times (3n+1)$ instead of $n \times 4n$). The specification is summarized in Algorithm (6).

Proving $\mathbf{m} = \sum_{i=0}^{l}(m_i x^i) \wedge m_j = 0\ for\ j = 1,2 \ldots l-1$. (ZKPExt2) This proof can be engineered similarly to **ZKPExt1**, following the same method. Except that in this one, instead of removing $(n-1)$ columns, we remove only the first column of the last n columns of **A**. The result is the matrix **A**, with dimension $n \times (4n-1)$. The specification is summarized in Algorithm (7).

Algorithm 6 ZKP for zero constant coefficient

procedure ZKPEXT1($(\mathbf{c}, pk), (\mathbf{m}, \mathbf{s}, \mathbf{e}, \mathbf{e}')$))
  $rot_{c_1} \leftarrow rot(\mathbf{c_1})$
  $rot_{p_1} \leftarrow rot(\mathbf{pk_1})$
  let I be the $n \times n$ identity matrix
  let I' be 1 column matrix with all 1s.
  let Z be the $n \times n$ zero matrix
  let Z' be 1 column matrix with all 0s.
  $\mathbf{A} \leftarrow ((rot_{c_1}, -tI, Z, -I'), (rot_{p_1}, Z, tI, Z'))$
  $\mathbf{x} \leftarrow (\mathbf{s}, \mathbf{e}', \mathbf{e}, \mathbf{m})$
  $\mathbf{y} \leftarrow (-\mathbf{c_0}, -\mathbf{pk_0})$
  Return SternExt$(\mathbf{A}, \mathbf{x}, \mathbf{y})$

Algorithm 7 ZKP for only constant non-zero coefficient

procedure ZKPEXT2($(\mathbf{c}, pk), (\mathbf{m}, \mathbf{s}, \mathbf{e}, \mathbf{e}')$))
  $rot_{c_1} \leftarrow rot(\mathbf{c_1})$
  $rot_{p_1} \leftarrow rot(\mathbf{pk_1})$
  let I be the $n \times n$ identity matrix
  let I' be I with the first column removed.
  let Z be the $n \times n$ zero matrix
  let Z' be Z with the first column removed.
  $\mathbf{A} \leftarrow ((rot_{c_1}, -tI, Z, -I'), (rot_{p_1}, Z, tI, Z'))$
  $\mathbf{x} \leftarrow (\mathbf{s}, \mathbf{e}', \mathbf{e}, \mathbf{m})$
  $\mathbf{y} \leftarrow (-\mathbf{c_0}, -\mathbf{pk_0})$
  Return SternExt$(\mathbf{A}, \mathbf{x}, \mathbf{y})$

Proving $\mathbf{m} = \sum_{i=0}^{l}(m_i x^i) \wedge m_j = 0 \vee m_j = 1 \; for \; j = 0, 1 \ldots l-1$ (ZKPExt3 and ZKPExt4). Following the previous extensions, proving that a message contains only zero or one coefficients amounts to a trivial modification of the matrix **A** and an according set up of the bound of **SternExt**.

6.4.4 ZKP of re-encryption correctness

This section first discusses the module **ZKPUnpack(c, c')**, which proves the correctness of the re-encryption of a single slot unpacked plaintext $\mathbf{m}' = Dec(\mathbf{c}')$ from the coefficients-packed ciphertext $\mathbf{m} = Dec(\mathbf{c})$. We then provide details about the **ZKPBinDecomp**$(\mathbf{c}, \mathbf{c_i})$ module, aimed at proving the correctness of re-encryption of a binary-encoded plaintext as a unary-encoded one.

ZKPUnpack(c,c',pk, s, e, e'). The relation of the proof is:

$$R_{\mathbf{ZKPUnpack}} = \{\mathbf{c}, \mathbf{c}' \in R_q \times R_q, \mathbf{pk} \in R_q^2; (\mathbf{s}, \mathbf{e}, \mathbf{e}') \in \chi^n : \\ \mathbf{m} = Dec(\mathbf{c}) \wedge \mathbf{m}' = Dec(\mathbf{c}') \wedge m_0 = m_0' \wedge (m_i' = 0 \; \forall i \neq 0)\}$$

Given $m_0 = m_0'$, we observe $\mathbf{m}(\mathbf{x}) - \mathbf{m}'(\mathbf{x}) = 0 + m_1'' x + \cdots + m_{n-1}'' x^{n-1}$. Therefore, **ZKPUnpack** can be carried out as specified in Algorithm (8)

Algorithm 8 ZKP of coefficients transform

1: **procedure** ZKPUNPACK($\mathbf{c}, \mathbf{c}', \mathbf{pk}, \mathbf{s}, \mathbf{e}, \mathbf{e}'$)
2: Let $b_1 \leftarrow$ **ZKPExt1**$((\mathbf{c} - \mathbf{c}', \mathbf{pk}), (\mathbf{Dec}(\mathbf{c} - \mathbf{c}'), \mathbf{s}, \mathbf{e}, \mathbf{e}'))$
3: Let $b_2 \leftarrow$ **ZKPExt2**$((\mathbf{c}', \mathbf{pk}), (\mathbf{Dec}(\mathbf{c}'), \mathbf{s}, \mathbf{e}, \mathbf{e}'))$
4: **Return** $b_1 \wedge b_2$

ZKPBinDecomp$(\mathbf{c}, \mathbf{c_i})$. The relation of the proof is:

$$R_{\mathbf{ZKPBinDec}} = \{\mathbf{c}, \mathbf{c}_i|_{i=0}^{l-1} \in R_q \times R_q^l, \mathbf{pk} \in R_q^2; (\mathbf{s}, \mathbf{e}, \mathbf{e}') \in \chi^n : \\ \mathbf{m} = Dec(\mathbf{c}) \wedge \mathbf{m_0} = \sum_{i=0}^{l-1} b_i 2^i \wedge \mathbf{c_i} = Enc(b_i) \wedge b_i \in 0,1\}$$

The specification of the proof is detailed in Algorithm (9).

Algorithm 9 ZKP of encoding transform

1: **procedure** ZKPBINDECOMP$(\mathbf{c}, \mathbf{c_i})$
2: Let $\mathbf{c}' \leftarrow \sum_{i=0}^{l-1} \mathbf{c_i} 2^i$
3: Let $\mathbf{c}'' \leftarrow c - c'$
4: **Return ZKPExt1**$((\mathbf{c}'', \mathbf{pk}), (\mathbf{Dec}(\mathbf{c}''), \mathbf{s}, \mathbf{e}, \mathbf{e}'))$

Applications in our protocol. In authentication step 5, after receiving the ciphertext $C_{HD'}$, which encrypts a plaintext of the form $(HD', g_1, \ldots, g_{n-1})$, $\mathscr{U}_k$ removes the noise terms $g_1, \ldots, g_{n-1}$ and sends the re-encryption $C_{HD'_0} = Enc(HD', 0, 0, \ldots, 0)$ back to $\mathscr{S}$. The client needs to prove that he performs this step correctly, this is done by **ZKPUnpack**$(\mathbf{C_{HD}}, \mathbf{C_{HD'_0}})$. In this step, the server also receives $\mathbf{c_i} = Enc(b_i)$ to compute $Enc(x^{HD'})$, as discussed in section 6.3.2. Before doing this operation, the client needs to convince the server that the bits sent are actually the ones decomposed from HD'. This proof is done by **ZKPBinDecomp**$(\mathbf{C_{HD'_0}}, \mathbf{c_i})$.

Besides, In authentication step 1, $\mathscr{U}_k$ uses **ZKPBV** to convince $\mathscr{S}$ that he is authenticating with a valid template Y. Moreover, in step 9, $\mathscr{U}_k$ can use either **ZKPExt3** or **ZKPExt4** to convince $\mathscr{S}$ about the authentication result.

6.5 Security Proofs

6.5.1 Security Proof for Theorem 15: Type I Impersonation attack

The proof of theorem 14 and 15 can be provided using a sequence of games between the challenger $\mathscr{C}$ and the adversary $\mathscr{A}$. We present a sequence of games as well as the relations among them to demonstrate the type I security model proof.

Game 0 Game 0 is the original impersonation game for type I attack.

Setup. $\mathscr{C}$ initiates $D_k \xleftarrow{r} D_{bio}$ and $X_k \xleftarrow{r} D_k$. $\mathscr{C}$ sets up (sk_k, pk_k) and executes $Enrol(k, X_k)$ to get $(sk_k, T_k = (pk_k, C_k = Enc^{(1)}_{pk_k}(X_k)))$. $\mathscr{A}$ submits the attack query type I and receives sk_k from $\mathscr{C}$.

Query. $\mathscr{A}$ runs q authentication sessions. In each session $j = 1, \ldots, q$, $\mathscr{A}$ sends $(Q_k^{(j)} = Enc^{(1)}(Y^{(j)}))$. $\mathscr{A}$ and $\mathscr{C}$ runs **ZKPValidEnc**$(Q_k^{(j)}, Y^{(j)})$. $\mathscr{C}$ evaluates $C_{HD} =$ **EvalDistance**$_{pk_k}(C_k, Q_k^{(j)})$, then computes $C_{HD'} = (-1)C_{HD} + Enc^{(1)}_{pk_k}(r_{HD'}, e_{HD'})$ for $r_{HD'} \xleftarrow{r} \mathscr{P}$ and $e_{HD'} \xleftarrow{r} \chi_{HD}$. The result ciphertext is sent to $\mathscr{A}$, $\mathscr{A}$ decrypts $C_{HD'}$ and decomposes it into bits and further sends back the ciphertexts $C_0^{(j)}, \ldots, C_{l-1}^{(j)} (= Enc^{(1)}(b_i), i = 0, \ldots, l-1)$, as well as the re-encryption $C_{HD'_0}$. $\mathscr{A}$ and $\mathscr{C}$ engage **ZKPUnpack**$(C_{HD}, C_{HD'_0})$ and **ZKPBinDecomp**$(C_{HD'_0}, C_i^{(j)})$. $\mathscr{C}$ evaluates $C''_{HD} = Enc^{(2)}(2^l + t) +$ **ToUnary**$(C_i^{(j)}) + Enc^{(2)}(-r)$ to get $Enc^{(2)}(2^l + t - HD)$. $\mathscr{C}$ computes $Enc^{(3)}(res) \leftarrow$ **MSBExtract**(C''_{HD}) and sends the result to $\mathscr{A}$. $\mathscr{A}$ decrypts it and sends the authentication result bit back. $\mathscr{C}$ and $\mathscr{A}$ then fire **ZKPCorrectDec**(res) to trigger the server's acceptance of the authentication result. At the end, the server outputs **Accept** if all the proofs pass and $res =$ **Accept**.

Next, we discuss the games that follow, the plan being to proceed towards the final game where everything $\mathscr{A}$ receives relating to X_k can be simulated without any knowledge about X_k, except that X_k is the function $Verify(X_k, Y)$. Let $res_s = Verify(X_k, Y^{(j)})$ and S_i be the

event in the game i such that $res_s = Accept$.

Game 1. In this game, we abort **ZKPValidEnc** if $Q_k^{(j)}$ is not a valid encryption of the query, but the *Prover* manages to pass the proof. Let $(*)_1$ be this event.

$$(*_1)res_s = \begin{cases} \text{Reject if } \textbf{ZKPValidEnc} \text{ fails} \\ \text{res else} \end{cases}$$

Let bad_0 be the event in game $0 : \exists j \leq q \; s.t \; (*_1)$ is thus the case. We want to show that when we modify *game 0*, the probability of a successful forgery S_1 in this *game 1* is not much lower than what it was. Due to the modification, we observe that $Pr[S_1] \geq Pr[S_0] - Pr[bad_0]$ .For any j in the q authentication attempts, by the $\varepsilon - soundness$ property of **ZKPValidEnc** as a proof of membership in (*), we have $Pr[(*_1) \; occurring \; for \; some \; j] \leq \varepsilon_{ZK1}$. So, the probability of bad_0 would be the union of these events, which is bounded by $Pr[bad_0] \leq q\varepsilon_{ZK1}$. In other words, the advantage of $\mathscr{A}$ in *game 1* is

$$Pr[S_1] \geq Pr[S_0] - Pr[bad_0] \geq \varepsilon_{imp} - q\varepsilon_{ZK1}$$

Game 2. In this game, we abort **ZKPUnpack** if, in one of the j^{th} runs, the $Verifier$ accepts but the ciphertext does not satisfy the relation. Let bad_1 be this event, by the same type of argument, we can derive $Pr[bad_1] \leq q\varepsilon_{ZK2}$ and therefore

$$Pr[S_2] \geq Pr[S_1] - Pr[bad_1] \geq \varepsilon_{imp} - q(\varepsilon_{ZK1} + \varepsilon_{ZK2})$$

Game 3 and Game 4. Similarly, we abort **ZKPBinDecomp** and **ZKPCorrectDec** if, in any j^{th} runs, the $Verifier$ accepts even when the correctness of the ZKP is not satisfied. We have

$$Pr[S_3] \geq \varepsilon_{imp} - q(\varepsilon_{ZK1} + \varepsilon_{ZK2} + \varepsilon_{ZK3})$$

and

$$Pr[S_4] \geq \varepsilon_{imp} - q(\varepsilon_{ZK1} + \varepsilon_{ZK2} + \varepsilon_{ZK3} + \varepsilon_{ZK4})$$

By the end of *Game 4*, if in any authentication attempt j, there is no abort while running any of the 4 games, then, by the correctness property of ZKP, the server output is equal to the output of $Verify(X_k, Y^{(j)})$. In other words, we have shown what we could get by just querying the oracle $Verify()$. Next, we want to simulate everything related to X_k the attacker can see using just that oracle.

Game 5. At the end of *Game 4*, $\mathscr{C}$ possesses the bit $b = Verify(Y^{(i)}, X_k)$. In this game, we can change $C_{res}^{(j)}$ from $\textbf{MSBExtract}(C''_{HD}) = Enc(b; e_{res})$ to $Enc(verify(Y^{(i)}, X_k); e_{res})$, given that the challenger can use the secret key to extract e_{res}. Despite this change, the same ciphertext is still accessible to $\mathscr{A}$, so its probability of winning of is the same as in *Game 4*.

Game 6. In this game, we change the way $\mathscr{C}$ computes C''_{HD}: In the original game 0, $Enc(-r)$ was added to remove the mask. We now want to remove this r from being used anywhere in the game, so we replace this with $Enc(0)$. This change does not affect $\mathscr{A}$'s success probability: r only affects the plaintext inside C''_{HD}, since we do not use this plaintext anymore (as it has been replaced in *Game 5*), so this change does not affect the information available to the attacker. Again, $Pr[S_6] = Pr[S_5] = Pr[S_4]$.

Game 7. We modify the way C'_{HD} is computed in this game. Instead of calculating $C'_{HD} \leftarrow (-1)C_{HD} + Enc(r; e_{HD'})$, the challenger chooses a random $HD' \xleftarrow{r} \mathbb{Z}_t$ and encrypts it with the noise used before: $C'_{HD} \leftarrow Enc(HD'; -e_{HD} + e_{HD'})$. In this game, the plaintext has changed from being $r + HD$ to a uniform $HD' \in \mathbb{Z}_t$. Since r is also uniform in $\mathbb{Z}_t$, the attacker is confronted with a uniform plaintext in both cases. Therefore, $Pr[S_7] = Pr[S_6]$.

Game 8. Finally, we set $C_{HD'} = Enc(HD', e_{HD'})$ for $e_{HD'} \xleftarrow{r} \chi_{mask}$ instead of $-e_{HD} + e_{HD'}$. We replace the sum of the Gaussian noise with a random noise. In the previous chapter, we showed that $Pr[S_8] \geq \frac{1}{c(\delta)}(Pr[S_7] - q \cdot \delta)$, where $c(\delta) = RD(-e_{HD} + e_{HD'}, e_{HD}) \leq 2 \cdot e^{1+2\delta}$ by Lemma 9 and by our assumption on the parameter's values. After we finish *Game 8*, we

notice that all the messages available to the attacker can be simulated with only the verified bit $b = Verify(Y^{(i)}, X_k)$. We now have an attacker A' against the biometric impersonation with advantage:

$$\varepsilon_{bio} = Adv(A') = Pr[S_8] \geq \frac{1}{c(\delta)}(\varepsilon_{imp} - q(\varepsilon_{ZK1} + \varepsilon_{ZK2} + \varepsilon_{ZK3} + \varepsilon_{ZK4}) + \delta),$$

which gives the claimed bound $\varepsilon_{imp} \leq c(\delta) \cdot \varepsilon_{bio} + q \cdot (\varepsilon_{ZK1} + \varepsilon_{ZK2} + \varepsilon_{ZK3} + \varepsilon_{ZK4}) + \delta \leq (c(\delta) + c_1) \cdot \varepsilon_{bio}$ if $q(\varepsilon_{ZK1} + \varepsilon_{ZK2} + \varepsilon_{ZK3} + \varepsilon_{ZK4}) + \delta \leq c_1 \cdot \varepsilon_{bio}$.

6.5.2 Security Proof for Theorem 15: Type II Impersonation attack

The proof of Theorem 15 can be provided using a sequence of games between the challenger $\mathscr{C}$ and the adversary $\mathscr{A}$. We present a sequence of games as well as the relations among them to demonstrate the type II security model proof. The idea is that in this type of attack, sk_k is not used to compute the view of $\mathscr{A}$. On the other hand, the soundness of the zero-knowledge proof of knowledge **ZKPValidEnc** implies the existence of an efficient witness extractor algorithm, that can be used to extract the witness (i.e. the secret key sk_k) from a cheating prover succeeding with probability non-negligibly higher than the knowledge error of the zero-knowledge proof, thus contradicting the IND-CPA security of the BV encryption scheme.

Game 0. Game 0 is the original impersonation game for a type II attack, i.e., the same as Game 0 in the proof of security against Type I attacks, except that $(T_k = (pk_k, C_k = Enc^{(1)}_{pk_k}(X_k)))$ is given by $\mathscr{C}$ to $\mathscr{A}$ at the beginning of the game, rather than sk_k. For $j \in \{1, \ldots, q\}$, let res^j denote the result of the jth authentication protocol run between $\mathscr{A}$ and $\mathscr{C}$, and S_0 be the event in the game 0 such that $res^j = Accept$ for some $j = 1, \ldots, q$. We have $\Pr[S_0] = \varepsilon_{imp,II}$ as the type II success probability of $\mathscr{A}$.

Game 1. From the definition of event S_0 in Game 0, it follows that there exists some $j^* \in \{1,\ldots,q\}$ such that $\Pr[res^{j^*} = Accept] \geq \varepsilon_{imp,II}/q$. Furthermore, by an averaging argument, there must exist a set G of (D_k, X_k, pk_k) such that $\Pr[(D_k, X_k, pk_k) \in G] \geq \varepsilon_{imp,II}/(2 \cdot q)$, and for each $(D'_k, X'_k, pk'_k) \in G$, we have $\Pr[res^{j^*} = Accept | (D_k, X_k, pk_k) = (D'_k, X'_k, pk'_k)] \geq \varepsilon_{imp,II}/(2 \cdot q)$. By ε_{ZK1}-soundness of the zero-knowledge proof of knowledge **ZKPValidEnc** [55], there exists a witness extractor algorithm that runs in expected time $T' = O(\text{poly}(n \log Q) \cdot T/(\varepsilon_{imp,II}/(2 * q) - \varepsilon_{ZK1}))$, where T denotes the run-time of $\mathscr{A}$ and outputs a witness containing sk_k for $ZKPValidEnc$. Therefore, we obtain a (secret key recovery) attack algorithm against the IND-CPA security of the BV encryption scheme, with expected run-time T' and advantage $\varepsilon' \geq \varepsilon_{imp,II}/(2 \cdot q)$. Hence, if $\varepsilon_{imp,II}/(2 * q) - \varepsilon_{ZK1} > \varepsilon_{imp,II}/(4 * q)$, or equivalently, if $\varepsilon_{imp,II} > 4 \cdot q\varepsilon_{ZK1}$, we obtain a contradiction with the assumption the BV encryption scheme with parameters Q, n, σ is IND-CPA against attacks with expected time $O(\text{poly}(n \log Q) \cdot T/\varepsilon_{ZK1})$ and advantage $\geq \varepsilon_{ZK1}$. It follows under the latter assumption that $\varepsilon_{imp,II} \leq 2q\varepsilon_{ZK1} \leq 4c_1 \cdot \varepsilon_{bio}$, under the assumption that $q(\varepsilon_{ZK1} + \varepsilon_{ZK2} + \varepsilon_{ZK3} + \varepsilon_{ZK4} + \delta) \leq c_1 \cdot \varepsilon_{bio}$.

6.5.3 Security Proof for Theorem 14: Privacy against Server

The proof of Theorem 15 can be done using a sequence of games between the challenger $\mathscr{C}$ and the adversary $\mathscr{A}$. We present a sequence of games. The idea is to proceed to remove sk_k and X_k from being used to compute the view of $\mathscr{A}$, except for $Verify(X_k, Y_k^j)$ queries, as in the ideal game, relying on the correctness of the protocol and the IND-CPA security of the BV encryption scheme.

Game 0. Game 0 is the original real privacy game, in which $(T_k = (pk_k, C_k = Enc^{(1)}_{pk_k}(X_k)))$ is given by $\mathscr{C}$ to $\mathscr{A}$ at the beginning of the game. Then, for $j = 1\ldots,q$, the attacker sends

$Y_k^{(j)} \in \{0,1\}^n$ to $\mathscr{C}$, and the latter simulates a run of the authentication protocol between an honest client with input $(k, Y_k^{(j)}, sk_k)$ and an honest server with input (k, T_k), returning to $\mathscr{A}$ the protocol view $V_S^{(j)}$ of the server. Finally, $\mathscr{A}$ outputs a bit β. In the following Game i, we let S_i denote the event of $\beta = 1$.

Game 1. We change the computation of the authentication result bit $res^{(j)}$ sent by client to server from the decryption of the ciphertext **MSBExtract**(C''_{HD} (its value in Game 0) to the result returned by $Verify(X_k, Y^{(j)})$. By correctness of the protocol, this does not change the value of $res^{(j)}$, so $\Pr[S_1] = \Pr[S_0]$.

Game 2. We change the computation of the zero-knowledge protocol transcripts. Instead of computing those transcripts using the secret witnesses, we simulate them using the statistical zero-knowledge simulator algorithms for the zero-knowledge proofs. By the zero-knowledge property, this is a perfect simulation, yielding $\Pr[S_2] = \Pr[S_1]$.

Game 3. We change the computation of the ciphertexts $C_i^{(j)}$ for $i = 0, \ldots, l-1$ and $Q_k^{(j)}$ for $j = 1, \ldots, q$ and C_k to encrypt zero messages, instead of encrypting the secret-related messages as in the previous game. Since now sk_k is not used anywhere in generating the view of $\mathscr{A}$, it follows by a hybrid argument that $|\Pr[S_3] - \Pr[S_2]|((l+1) \cdot q + 1) \cdot \varepsilon_{BV}$, where ε_{BV} denotes the maximal advantage of an attacker against IND-CPA of BV scheme against attacks with run-time $T + \text{poly}(n, \log Q)$, where T is the run-time of $\mathscr{A}$. In this game, since the only information on X_k comes via the $Verify(X_k, Y_k^j)$ queries, the challenger together with $\mathscr{A}$ constitute an efficient attacker against the ideal privacy game, which outputs 1 with probability different by at most $(l+1) \cdot q + 1) \cdot \varepsilon_{BV}$ relatively to the probability of outputting 1 in the real privacy game, as required.

6.6 Conclusion

We quickly notice that the main bottle neck of this protocol is the communication size: 4 different Zero-Knowledge Proofs are needed for one authentication. With the parameters set used previously ($n = 2048, q \approxeq 2^{70}, t = 2048, \sigma = 8$), the communication size for each authentication ($\approx 84MB$) will not satisfy the practical usage requirement. However, the contributions of this chapter maybe of independent interest, as building blocks for other secure computation protocols are based on the BV SWHE scheme:

- A ZKP technique to prove the correctness of re-encryption of a binary-encoded plaintext (useful for efficient homomorphic addition/multiplication arithmetic operations) as a unary-encoded plaintext (useful for efficient homomorphic comparison operations).
- Correctness of re-encryption of a coefficient-packed plaintext slot vector into an unpacked single slot plaintext.

In the next chapter, we discuss different techniques used to remove this communication overhead: we replace the Stern-based ZKP technique, which requires many rounds of proof for security by the Schnoor-based technique (requiring only 1 round). Other cryptographic tools including Garbled Circuit and Oblivious Transfer will be used in the HD comparison step to avoid supplementary proofs during authentication.

Chapter 7

The fourth protocol - Removing The Overhead

7.1 Introduction

In the last chapter, we developed a two-parties authentication protocol that proves secure against a malicious client and does not leak any information to the server (the main difference with the second protocol being that the server is not able to acquire the value of the Hamming Distance between the registered and queried templates). This final variant aims to reduce both the communication size and the computation time, while providing all the recommended security features.

Reducing Communication Size We observe that the main bottle neck of the communication is the size of the Stern-based ZKPs. We propose the following approach to replace old ZKPs:

- A Schnorr-based approach to replace the Stern-based one, to ensure that the noise in the ciphertext is small. This variant reduces the total communication

rounds from $log_{3/2}(FAR)^{-1}$ to just 1. Given a typical value of $FAR = 10^{-4}$, such improvement reduces the total number of communication rounds from ≈ 22 to 1.

- A challenge-response protocol to prove the binary format of the plaintext.

Reducing Computation Time. By applying a new ciphertext packing method, we can effectively change the way of computing the Hamming Distance by summing up the bits of the XOR result of 2 bitstrings instead of computing their inner product. Therefore, enforcing this approach, we lower the computation time by performing additions only (without any homomorphic multiplication). The cost of this change is a set of switching keys for homomorphic rotation operations, which are generated during the enrollment process. Additionally, we introduce the application of state-of-the-art *Secure-Multiparty-Protocol* techniques, such as *Garbled Circuit* and *Oblivious Transfer* into the lattice-based context, to further improve both the computation time and the communication size overhead implied by previous ZKP protocols.

This approach is practical because it moves a major part of the overhead to the one-time set up phase of the protocol (the enrollment stage). Our results show that the protocol can work efficiently during authentication stage. The technique maybe generalized to similar multi-stage protocols.

7.2 Reducing the communication overhead

7.2.1 The Schnorr-based ZKP

In the previous variants of the protocol, the critical module required to provide active security against a malicious client is the binary message format prover. Specifically, Alice sends a ciphertext C of a bitstring to Bob. Bob is supposed to use a homomorphic encryption technique to process C. Before doing so, Bob might want to make sure that C is an encryption

of a bitstring (and not something else). In the last chapter, we used the Stern-based Zero Knowledge Proof technique for this step, and it was shown that the communication size for this is large due to the soundness error $2/3$ for each round of the ZKP (the repetition of many proof-rounds being required to obtain a specific security level). This chapter investigates another alternative to ZKP, exhibiting a better soundness error of $1/n$ within one round of proof only, where n is typically the required parameter: the Schnorr-based ZKP technique. We first recall some notation and concepts to be used in the chapter.

We denote a language $\mathscr{L} \subseteq \{0,1\}^*$ having a witness relationship $R \subseteq \{0,1\}^* \times \{0,1\}^*$ if $x \in \mathscr{L} \iff \exists (x,w) \in R$, where w is a witness for $x \in \mathscr{L}$.We discuss a definition of the sigma protocol from [19], which can be adapted to suit the techniques we use in our project.

Definition 29. *Let (P,V) be a two-party protocol, where V is PPT, and let $\mathscr{L},\mathscr{L}' \subseteq \{0,1\}^*$ be languages with witness relations $\mathscr{R},\mathscr{R}'$ such that $\mathscr{R} \subseteq \mathscr{R}'$. (P,V) is called a $\sum'-$protocol for $\mathscr{L},\mathscr{L}'$ with completeness error α, challenge set $\mathscr{C}$, public input x and private input w, if and only if it satisfies the following conditions:*

- *Three-move form: The protocol is of the following form: The prover P, on input (x,w), computes a commitment t and sends it to V. The verifier V, on input x, then draws a challenge $c \xleftarrow{r} \mathscr{C}$ and sends it to P. The prover sends a response s to the verifier. Depending on the protocol transcript (t,c,s), V accepts or rejects the proof.*

- *Completeness: Whenever $(x,w) \in \mathscr{R}$, the verifier V accepts with probability at least $1-\alpha$.*

- *Special Soundness: There exists a PPT algorithm E, also known as the knowledge extractor, that takes two accepted transcript $(t,c',s'),(t,c'',s'')$ satisfying $c' \neq c''$ as inputs and output w' such that $(x,w') \in \mathscr{R}'$.*

- *Special honest-verifier zero-knowledge (HVZK): There exists a simulator S, which is a PPT algorithm taking $x \in \mathscr{L}$ and $c \in \mathscr{C}$ as inputs and outputs (t,s) so that the triple (t,c,s) is indistinguishable from a valid protocol transcript.*

This definition is different from the original sigma protocol in 2 ways. First, it allows a completeness error α instead of perfect completeness ($\alpha = 0$ in the original protocol). Second, a second language $\mathscr{L}$ is introduced, with witness relation $\mathscr{R} \subseteq \mathscr{R}'$: A prover knows a witness in $\mathscr{R}$ is guaranteed privacy, but the verifier is only ensured that the P only knows a witness for $\mathscr{R}'$. We refer to this as *soundness gap*, if the gap is small enough, it implies security guarantee for higher level applications ($\mathscr{R} = \mathscr{R}'$ in the original protocol).

The original Schnorr ZKP protocol [116] can be used to prove knowledge of a discrete logarithm (DL). It is described informally as in Figure 7.1

Schnorr Protocol for DL

Prover		**Verifier**
$(x = \log_g h)$		
$u \xleftarrow{r} \mathbb{Z}_n$		
$t \leftarrow g^u$	$\xrightarrow{\quad t \quad}$	
	$\xleftarrow{\quad c \quad}$	$c \xleftarrow{r} \mathbb{Z}_n$
$s \leftarrow u + cx$	$\xrightarrow{\quad s \quad}$	
		$g^s \stackrel{?}{=} ah^c$

Fig. 7.1 Schnorr Protocol

The soundness property of Schnorr's protocol can be argued by assuming that if a prover is able to answer correctly at least two challenges c and c' (with $c \neq c'$) after committing to an announcement t, then he would be able to infer a solution for the hard problem of discrete log $x = \log_g h$. Hence, intuitively, if the prover can answer at most one challenge correctly, the probability of success is bounded by $1/n$, which can be negligibly small. The

Zero Knowledge property (against a semi-honest Verifier) was proved by showing that the real conversation $\{(t,c,s) : u,c \xleftarrow{r} \mathbb{Z}_n; t \leftarrow g^u; s \leftarrow u+cx\}$ and the simulated conversation $\{(t,c,s) : c,s \xleftarrow{r} \mathbb{Z}_n; t \leftarrow g^s h^{-c}\}$ are identical (each valid conversation $(t;c;s)$ occurs with probability $1/n^2$ in both distributions).

In the work of [19], the authors proposed a variant of such technique to prove knowledge of small RLWE secret $\mathbf{s}$ and noise $\mathbf{e}$ (in short, we mean that $\|\mathbf{s}\|, \|\mathbf{e}\| \leq \mathscr{O}(n\alpha)$ for noise parameter α), where $(\mathbf{a}, \mathbf{y} = \mathbf{as}+\mathbf{e})$ is the verifier's input and $(\mathbf{s},\mathbf{e})$ is the prover's witness input (see Figure 7.2)

Benhamouda Protocol for RLWE

Prover		**Verifier**
$\mathbf{r}_s, \mathbf{r}_e \xleftarrow{r} D_{\mathscr{O}(\sqrt{n}\alpha)}$		
$t = a\mathbf{r}_s + \mathbf{r}_e$		
$(c_{aux}, d_{aux}) = aCommit(t)$	$\xrightarrow{c_{aux}}$	
	$\xleftarrow{c}$	$c \xleftarrow{r} \{0,\ldots,2n-1\}$
$\mathbf{s}_s = \mathbf{r}_s + X^c s$		
$\mathbf{s}_e = \mathbf{r}_e + X^c e$		
	$\xrightarrow{t, d_{aux}, (\mathbf{s}_s, \mathbf{s}_e)}$	$X^c y + t \stackrel{?}{=} a\mathbf{s}_s + \mathbf{s}_e$
		$aCOpen(t, c_{aux}, d_{aux}) \stackrel{?}{=}$ accept
		$\|\mathbf{s}_s\|, \|\mathbf{s}_e\| \leq \mathscr{O}(n\alpha)$

Fig. 7.2 Proof of knowledge of RLWE secrets

Where $y = as+e$, and $s,e \xleftarrow{r} D_\alpha$. In the protocol, the prover is able to convince a verifier that it knows short secrets for twice the public input (in short, we mean that $\|s\|, \|e\| \leq \mathscr{O}(n\alpha)$). We present an adaption of this ZKP protocol to prove the plaintext knowledge of BV cryptosystems. Recall that given $(\mathbf{sk},\mathbf{pk}) = (\mathbf{s},(\mathbf{p_0},\mathbf{p_1}))$ where $\mathbf{s} \in \chi^n_{\beta_0}$, $\mathbf{p_0},\mathbf{p_1} \in R_q$, a message $\mathbf{m} \in R_t$ is encrypted by $[\![\mathbf{m}]\!] := (\mathbf{c_0},\mathbf{c_1}) = (\mathbf{p_0u}+t\mathbf{f}+\mathbf{m}, \mathbf{p_1u}+t\mathbf{g})$. The protocol (Figure 7.3) is an AND composition of 2 ZKPs that prove 2 relations: the secret key $\mathbf{s}$

corresponds to the public key $(\mathbf{p_0}, \mathbf{p_1})$, i.e., $\mathbf{p_0} = -\mathbf{p_1 s} - t\mathbf{e_0}$, and the ciphertext $(\mathbf{c_0}, \mathbf{c_1})$ is well-formed and encrypts the message $\mathbf{m}$, that is, $\mathbf{c_0} + \mathbf{c_1 s} = \mathbf{m} + t\mathbf{e}$, with $\|\mathbf{e}\|_\infty \leq \beta$. Note that the protocol guarantees by V that P knows the plaintext encrypted in $2\mathbf{c_0}$

Schnorr-based ZKP for BV

Prover		**Verifier**
$\mathbf{r_s}, \mathbf{r_{e_0}} \xleftarrow{r} D_{\beta_0}$		
$\mathbf{r_e} \xleftarrow{r} D_{\beta}$		
$\mathbf{r_m} \xleftarrow{r} D_t$		
$\mathbf{t_c} = -\mathbf{c_1 r_s} + t\mathbf{r_e} + \mathbf{r_m}$		
$\mathbf{t_k} = -\mathbf{p_1 r_s} - t\mathbf{r_{e_0}}$		
$(c_{aux}, d_{aux}) = aCommit(\mathbf{t_c}, \mathbf{t_k})$	$\xrightarrow{c_{aux}}$	
		$c \xleftarrow{r} \mathscr{C} = \{0, \ldots, 2n-1\}$
	$\xleftarrow{c}$	
$\mathbf{s_s} = \mathbf{r_s} + \mathbf{x^c s}$		
$\mathbf{s_e} = \mathbf{r_e} + \mathbf{x^c e}$		
$\mathbf{s_m} = \mathbf{r_m} + \mathbf{x^c m}$		
$\mathbf{s_{e_0}} = \mathbf{r_{e_0}} + \mathbf{x^c e_0}$		
	$\xrightarrow{d_{aux}, \mathbf{t_c}, \mathbf{t_k}, \mathbf{s_s}, \mathbf{s_e}, \mathbf{s_m}, \mathbf{s_{e_0}}}$	
		$\mathbf{x^c c_0} + \mathbf{t_c} \stackrel{?}{=} -\mathbf{c_1 s_s} + t\mathbf{s_e} + \mathbf{s_m}$
		$\mathbf{x^c p_0} + \mathbf{t_k} \stackrel{?}{=} -\mathbf{p_1 s_s} - t\mathbf{s_{e_0}}$
		$aCOpen(\mathbf{t_c}, \mathbf{t_k}, c_{aux}, d_{aux}) \stackrel{?}{=} accept$
		$\|s_s\|, \|s_{e_0}\| \leq 2\beta_0$
		$\|s_e\| \leq 2\beta$
		$\|s_m\| \leq 2t$

Fig. 7.3 ZKP for BV cryptosystem relation

Theorem 16. *Figure (7.3) is a $\sum'$-Protocol for the following relations:*

$$\begin{aligned}\mathscr{R} = \{(\mathbf{c_0}, \mathbf{c_1}, \mathbf{p_0}, \mathbf{p1}, t), (\mathbf{m}, \mathbf{s}, \mathbf{e}, \mathbf{e_0}) : \mathbf{c_0} = -\mathbf{c_1 s} + t\mathbf{e} + \mathbf{m} \wedge \mathbf{p_0} = -\mathbf{p_1 s} - t\mathbf{e_0} \\ \wedge \|\mathbf{m}\|_\infty \leq t \wedge \|\mathbf{e}\| \leq \beta \wedge \|\mathbf{s}\|, \|\mathbf{e_0}\| \leq \beta_0\}\end{aligned} \tag{7.1}$$

$$\mathscr{R}' = \{(\mathbf{c_0}, \mathbf{c_1}, \mathbf{p_0}, \mathbf{p1}, t), (\mathbf{m}, \mathbf{s}, \mathbf{e}, \mathbf{e_0}) : \mathbf{2c_0} = -\mathbf{2c_1 s} + 2t\mathbf{e} + 2\mathbf{m} \wedge \mathbf{2p_0} = -\mathbf{2p_1 s} - 2t\mathbf{e_0} \\ \wedge \|\mathbf{2m}\|_\infty \leq 2t \wedge \|\mathbf{2e}\| \leq 2\beta \wedge \|\mathbf{2s}\|, \|\mathbf{2e_0}\| \leq 2\beta_0\} \tag{7.2}$$

The protocol has a knowledge error of $\frac{1}{2n}$ *and a completeness error of* $1 - \frac{1}{M}$.

Proof. We prove the properties specified in Definition 29 following the proof from [19].

Completeness. If the Prover P sends a response, the following becomes the case:

$$\begin{aligned} -\mathbf{c_1 s_s} + t\mathbf{s_e} + \mathbf{s_m} &= -\mathbf{c_1}(\mathbf{r_s} + \mathbf{x^c s}) + t(\mathbf{r_e} + \mathbf{x^c e}) + \mathbf{r_m} + \mathbf{x^c m} \\ &= \mathbf{x^c}(-\mathbf{c_1 s} + t\mathbf{e} + \mathbf{m}) - \mathbf{c_1 r_s} + t\mathbf{r_e} + \mathbf{r_m} \\ &= \mathbf{x^c c_0} + \mathbf{t_c} \end{aligned}$$

Similarly

$$\begin{aligned} -\mathbf{p_1 s_s} - t\mathbf{s_{e_0}} &= -\mathbf{p_1}(\mathbf{r_s} + \mathbf{x^c s}) - \mathbf{t}(\mathbf{r_{e_0}} + \mathbf{x^c e_0}) \\ &= \mathbf{x^c}(-\mathbf{p_1 s} - t\mathbf{e_0}) - \mathbf{p_1 r_s} - t\mathbf{r_{e_0}} \\ &= \mathbf{x^c p_0} + \mathbf{t_k} \end{aligned}$$

Regarding norms, we have $\|\mathbf{s_s}\| = \|\mathbf{r_s} + \mathbf{x^c s}\| \leq \|\mathbf{r_s}\| + \|\mathbf{s}\| \leq 2\beta_0$, and the same for $\|\mathbf{s_e}\|, \|\mathbf{s_m}\|, \left\|\mathbf{s_{e_0}}\right\|$.

Special Soundness. Before proving this property, we describe the following technical lemma

Lemma 11. *Let n be a power of 2 and let $0 < i, j < 2n - 1$. Then, $2(x^i - x^j)^{-1}$ mod $(x^n + 1)$ only has coefficients in $\{-1, 0, 1\}$.*

Assume that a dishonest prover does not know 'small' $2\mathbf{s}, 2\mathbf{e}, 2\mathbf{m}$ and $2\mathbf{e_0}$ and can output different accepted transcripts of a same commitment: $(c_{aux}, c', (d'_{aux}, \mathbf{t'_c}, \mathbf{t'_k}, \mathbf{s'_s}, \mathbf{s'_e}, \mathbf{s'_m}, \mathbf{s'_{e_0}}))$ and $(c_{aux}, c'', (d''_{aux}, \mathbf{t''_c}, \mathbf{t''_k}, \mathbf{s''_s}, \mathbf{s''_e}, \mathbf{s''_m}, \mathbf{s''_{e_0}}))$. From the binding property of the auxiliary commitment scheme [104], we have $\mathbf{t'_c} = \mathbf{t''_c} := \mathbf{t_c}$ and $\mathbf{t'_k} = \mathbf{t''_k} := \mathbf{t_k}$. As the transcripts pass the checks by the verifier, it appears that:

$$\begin{cases} \mathbf{x}^{\mathbf{c}'}\mathbf{c_0} + \mathbf{t_c} = -\mathbf{c_1}\mathbf{s'_s} + t\mathbf{s'_e} + \mathbf{s'_m} \\ \mathbf{x}^{\mathbf{c}'}\mathbf{p_0} + \mathbf{t_k} = -\mathbf{p_1}\mathbf{s'_s} - t\mathbf{s'_{e_0}} \end{cases} \quad \& \quad \begin{cases} \mathbf{x}^{\mathbf{c}''}\mathbf{c_0} + \mathbf{t_c} = -\mathbf{c_1}\mathbf{s''_s} + t\mathbf{s''_e} + \mathbf{s''_m} \\ \mathbf{x}^{\mathbf{c}''}\mathbf{p_0} + \mathbf{t_k} = -\mathbf{p_1}\mathbf{s''_s} - t\mathbf{s''_{e_0}} \end{cases}$$

By subtracting the equations we get:

$$\begin{cases} \mathbf{c_0}(\mathbf{x}^{\mathbf{c}'} - \mathbf{x}^{\mathbf{c}''}) = -\mathbf{c_1}(\mathbf{s'_s} - \mathbf{s''_s}) + t(\mathbf{s'_e} - \mathbf{s''_e}) + (\mathbf{s'_m} - \mathbf{s''_m}) \\ \mathbf{p_0}(\mathbf{x}^{\mathbf{c}'} - \mathbf{x}^{\mathbf{c}''}) = -\mathbf{p_1}(\mathbf{s'_s} - \mathbf{s''s}) - t(\mathbf{s'_{e_0}} - \mathbf{s''_{e_0}}) \end{cases}$$

Multiplying by $2(\mathbf{x}^{\mathbf{c}'} - \mathbf{x}^{\mathbf{c}''})^{-1}$:

$$\begin{cases} \mathbf{2c_0} = -\mathbf{c_1}2(\mathbf{s'_s} - \mathbf{s''_s})(\mathbf{x}^{\mathbf{c}'} - \mathbf{x}^{\mathbf{c}''})^{-1} + t2(\mathbf{s'_e} - \mathbf{s''_e})(\mathbf{x}^{\mathbf{c}'} - \mathbf{x}^{\mathbf{c}''})^{-1} + 2(\mathbf{s'_m} - \mathbf{s''_m})(\mathbf{x}^{\mathbf{c}'} - \mathbf{x}^{\mathbf{c}''})^{-1} \\ \mathbf{p_0} = -\mathbf{p_1}\mathbf{2}(\mathbf{s'_s} - \mathbf{s''_s})(\mathbf{x}^{\mathbf{c}'} - \mathbf{x}^{\mathbf{c}''})^{-1} - t2(\mathbf{s'_{e_0}} - \mathbf{s''_{e_0}})(\mathbf{x}^{\mathbf{c}'} - \mathbf{x}^{\mathbf{c}''})^{-1} \end{cases}$$

Let $2\hat{\mathbf{s}} = 2(\mathbf{s'_s} - \mathbf{s''_s})(\mathbf{x}^{\mathbf{c}'} - \mathbf{x}^{\mathbf{c}''})^{-1}$, $2\hat{\mathbf{e}} = 2(\mathbf{s'_e} - \mathbf{s''_e})(\mathbf{x}^{\mathbf{c}'} - \mathbf{x}^{\mathbf{c}''})^{-\mathbf{1}}$, $2\hat{\mathbf{e_0}} = 2(\mathbf{s'_{e_0}} - \mathbf{s''_{e_0}})(\mathbf{x}^{\mathbf{c}'} - \mathbf{x}^{\mathbf{c}''})^{-\mathbf{1}}$ and $2\hat{\mathbf{m}} = 2(\mathbf{s'_m} - \mathbf{s''_m})(\mathbf{x}^{\mathbf{c}'} - \mathbf{x}^{\mathbf{c}''})^{-\mathbf{1}}$. Applying the result from Lemma 11, we see that P knows the 'small' secrets $\left\|2\hat{\mathbf{s}}\right\| < 2\beta_0$, $\left\|2\hat{\mathbf{e}}\right\| < 2\beta_0$, $\left\|2\hat{\mathbf{e_0}}\right\| < 2\beta$ and $\left\|2\hat{\mathbf{m}}\right\| < 2t$ that can pass the verifier's checks. This contradicts our assumption. In other words, the prover has to know the secret to pass the proof with probability $\frac{1}{2n}$.

Honest Verifier Zero-Knowledge. The verifier can use a simulator S to reproduce the protocol transcript as follows:

- V samples $c \xleftarrow{r} \mathscr{C}$

- With probability 1/M, S chooses $\mathbf{s_s}, \mathbf{s_{e_0}} \xleftarrow{r} D_{\beta_0}$, $\mathbf{s_e} \xleftarrow{r} D_\beta$ and $\mathbf{s_m} \xleftarrow{r} D_t$.
- S computes $\mathbf{t_c} = -\mathbf{c_1 s_s} + t\mathbf{s_e} + \mathbf{s_m} - \mathbf{x^c c_0}$ and $\mathbf{t_k} = -\mathbf{p_1 s_s} - t\mathbf{s_{e_0}} - \mathbf{x^c p_0}$ and $(c_{aux}, d_{aux}) \leftarrow aCommit(\mathbf{t_c}, \mathbf{t_k})$.
- S outputs $(c_{aux}, c, ((\mathbf{t_c}, \mathbf{t_k}), d_{aux}, (\mathbf{s_s}, \mathbf{s_{e_0}}, \mathbf{s_e}, \mathbf{s_m})))$

The distribution of $(\mathbf{s_s}, \mathbf{s_{e_0}}, \mathbf{s_e}, \mathbf{s_m})$ does not depend on real secrets. Hence, the simulated transcript becomes insdistinguishable from the real one.

□

7.2.2 The challenge response operation

It has been shown in the previous section that we can replace the Stern-based ZKP technique with a Schnorr-based one and consequently save many rounds of communication, due to the soundness error decreasing from $2/3$ to $1/n$. However, the new proof technique would not be capable of proving that the encrypted message is in binary format: The proof only shows that the noise level is small ($\mathbf{s}_e \leq 2\beta$) and that the message is small ($\mathbf{s}_m \leq 2t$) as well. To solve this problem, we include one homomorphic operation, to be performed after the *Client* has passed the proof. The operation works as follows.

1. The server samples $r_1, r_2 \xleftarrow{r} R_q$ and computes $ch = enc(r_1 * Q * (1-Q) + r_2) + enc(0, r)$ and sends ch to the client.
2. The client decrypts the result and sends back rsp.
3. The server accepts the response if and only if $rsp = r_2$

We remark that the first operation needs to be computed bit-wise, that means the packing technique we used in the previous protocol cannot be applied. In the next section, we discuss how CRT-packing method [123] can be adapted to compute the Hamming Distance of the registered and queried template.

7.3 The BGV Cryptosystem and CRT packing method

We used the BV cryptosystem [27] in the previous variants of the protocol, as only 2 levels of multiplication on ciphertexts were needed. In this variant, we do not use one inner product operation for HD computation, but, instead, we add the bits of the bitstring together. Hence, more operations on ciphertexts during HD computation will be needed. The BGV cryptosystem [26] has a mechanism to control the noise after operations on ciphertexts have been performed. We hereby discuss a specific version of this cryptosystem, where parameters are set to comply with the rings to be used in the protocol. The basic cryptosystem works as follows.

E.Setup Let λ be the security parameter that represents 2^λ security level against known attacks. Choose a modulus q and a noise distribution χ satisfying correctness and security requirements (discussed later). Let the "dimension" n be a power of 2 and let $R = \mathbb{Z}[x]/(x^n+1)$, let $params = (q, n, \chi)$

E.SecretKeyGen Draw $\mathbf{s}' \xleftarrow{r} \chi^n$. Set $sk = (1, \mathbf{s}') \in R_q^2$

E.PublicKeyGen Takes as its input a secret key sk and $params$. Draw $\mathbf{a}' \in R_q$ and a vector $\mathbf{e} \in \chi^n$ and set $\mathbf{b} \leftarrow \mathbf{a}'\mathbf{s}' + 2\mathbf{e} \in R_q$. Then we set $\mathbf{A} = (\mathbf{b}, -\mathbf{a}') \in R_q^2$. Let the public key $pk = \mathbf{A}$. Observe that $\mathbf{A} \cdot \mathbf{s} = 2\mathbf{e}$

E.Enc($params, pk, m$) To encrypt a message $\mathbf{m} \in R_2$, sample $\mathbf{r} \xleftarrow{r} R_2$ and output the ciphertext $\mathbf{c} \leftarrow \mathbf{r}\mathbf{A} + (\mathbf{m}, \mathbf{0}) \in R_q^2$

E.Dec($params, sk, \mathbf{c}$) Output $\mathbf{m} \leftarrow \Big[[\langle \mathbf{c}, \mathbf{s} \rangle]_q\Big]_2$

We have seen that a plaintext can be packed in a specific way to support homomorphic operations efficiently. The approach of [133] can compute the inner product operations to perform really fast; however, it does not support the Single Instruction Multiple Data (SIMD) operation, which is required by our challenge-response protocol. This section

discusses a different approach for plaintext packing and specifies the operations it supports: we are referring to the CRT packing method [123]. In this method, a message $\mathbf{m} \in R_t$ is packed by representing it in the NTT domain. In NTT domains, a single operation (addition or multiplication) of a pair of ciphertexts implicitly compute the entire plaintext vectors component-wise. Besides SIMD support, we can also *rotate* or *permute* the plaintext slots ([50]), we only need a *rotate* operation to this end.

Number Theoretic Transform In our application context, the most costly low-level computation operation is the ring multiplication (polynomial multiplication). This section discusses the Number Theoretic Transform (NTT), a technique providing efficient algorithms for cyclic and nega-cyclic convolutions that can be applied to compute polynomial multiplication efficiently. Moreover, in our research context, when the message is represented in the NTT domain, the operations are computed component-wise simultaneously. This important property helps our challenge-response protocol to save most of the communication size, compared with the ZKPs overhead of the previous protocols.

Notation Given $a(x) = a_0 + a_1x + \cdots + a_{n-1}x^{n-1}$ and $b(x) = b_0 + b_1x + \cdots + b_{n-1}x^{n-1}$, we denote $\cdot$ to be the convolution multiplication and $\triangle$ to be the component-wise multiplication of the two polynomials.

Introduction One of the most important applications of the Fast Fourier Transform (FFT) is the multiplication of large integers and polynomial multiplication (which is the cyclic convolution of two integer sequences). The last operation can be computed by applying the FFT to the sequences, then multiplying the result component-wise and applying the inverse FFT to the product:

$$a(x) \cdot b(x) = FFT^{-1}(FFT(a(x)) \triangle FFT(b(x)))$$

When the coefficients are elements of a finite field, the FFT is called the Number Theoretic Transform (NTT). We recall the definition of NTT: Given the parameters of our context, where n is a power of 2 and t is a prime with $t = 1 \mod 2n$, let $\mathbf{a} = [a_0, a_1, \ldots, a_{n-1}] \in R_t$, let ω be a primitive n^{th} root of unity in $\mathbb{Z}_t$, that is, $\omega^n = 1 \mod t$. The forward transform $NTT(\mathbf{a}) = \tilde{\mathbf{a}}$ is defined as $\tilde{\mathbf{a}}_i = \sum_{j=0}^{n-1} \mathbf{a}_j \omega^{ij} \mod t$ for $i = 0, \ldots, n-1$. Informally, the coefficients of $\tilde{\mathbf{a}}$ are the evaluations of the polynomial $a(x)$ at the n roots of unity. The inverse transformation is given by $\mathbf{b} = NTT^{-1}(\tilde{\mathbf{a}})$, where $\mathbf{b}_i = n^{-1} \sum_{j=0}^{n-1} \tilde{\mathbf{a}}_j \omega^{-ij} \mod t$ for $i = 0, \ldots, n-1$. For these settings, $NTT^{-1}(NTT(\mathbf{a})) = \mathbf{a}$. The above convolution operation computes a polynomial $c(x) = a(x) \cdot b(x)$ with order at most $2n-1$.

Therefore, in normal polynomial multiplication scenarios, $a(x)$ and $b(x)$ are padded with coefficients 0 before applying NTT. If the original $a(x)$ and $b(x)$ are not padded with 0s, the result product corresponds to the actual product modulo $x^n - 1$.

$$a(x) \cdot b(x) \mod (x^n - 1) = NTT^{-1}(NTT(a(x)) \triangle NTT(b(x)))$$

However, in our contexts, we want the operations to be computed modulo $x^n + 1$. The original ring elements can still be padded with 0s and the product modulo $x^n + 1$ computed, this will double the length of NTT inputs and also require an extra step of explicitly reducing $x^n + 1$. The other way to avoid this issue is by exploiting the *negative wrapped convolution* [88]. Let ψ be a primitive $2n^{th}$ root of unity in $\mathbb{Z}_t$ such that $\psi^2 = \omega$ and denote $NTT_+(\mathbf{a})$ to be the special NTT transform such that $NTT_+(\mathbf{a}) = NTT(\mathbf{a} \triangle (1, \psi, \psi^2, \ldots, \psi^{n-1}))$ and $NTT_+^{-1}(\mathbf{a}) = NTT^{-1}(\mathbf{a}) \triangle (1, \psi^{-1}, \psi^{-2}, \ldots, \psi^{-(n-1)})$. It can be proved that $NTT_+(\mathbf{a})$ is the evaluation of $a(x)$ at the odd roots of the $2n^{th}$ root of unity: $NTT_+(\mathbf{a}) = (a(\omega_0), a(\omega_1), \ldots, a(\omega_{n-1}))$,

where $\omega_i = \psi^{2i+1}$. It turns out that

$$a(x) \cdot b(x) \mod (x^n+1) = NTT_+^{-1}(NTT_+(a(x)) \triangle NTT_+(b(x)))$$

Since most of our work is based on $R_q = \frac{\mathbb{Z}_q}{x^n+1}$, when we mention NTT throughout the text, we imply the operation is done with NTT_+ instead of plain NTT. In the implementation, in order to find ψ, we need to find the group element such that $\psi^2 = \omega \mod t$, where ω is the n^{th} root of unity. There are heuristic approaches to find the square root of an element; in our context, we can find it easily (the problem is finding an element with order of $2n$). Given n,t such that $2n|(t-1)$ (these parameters are set up at the initialization stage of the protocol), if we can find a generator g of order $t-1$ in $\mathbb{Z}_t^{*J}$, then $\psi = g^{\frac{t-1}{2n}}$:

$$order(g^{\frac{t-1}{2n}}) = \frac{t-1}{gcd(\frac{t-1}{2n}, t-1)} = \frac{t-1}{(\frac{t-1}{2n})} = 2n$$

7.3.1 The rotate and permute function

Slot Rotation Given a polynomial $a(x) = (a_0, a_1, \ldots, a_{n-1}) \in \mathbb{Z}_t^n$, let $\mathbf{a}$ be the coefficient representation of $a(x)$ and $\hat{\mathbf{a}}$ be the NTT representation of $a(x)$. The *rotate* operation is a mapping K such as $a(x) \overset{K_j}{\mapsto} a(x^j)$, where j is the number of positions being rotated and $gcd(j,n) = 1$. This operation can be done on either the coefficient or the NTT representation of $a(x)$.

Coefficients domain In order to map $a(x) \overset{K_j}{\mapsto} a(x^j)$, or

$$\mathbf{a} = (a_0, a_1, \ldots, a_{n-1}) \overset{K_j}{\mapsto} a(x^j) = a_0 + a_1(x^j)^1 + a_2(x^j)^2 + \cdots + a_{n-1}(x^j)^{n-1}$$

We can look at the relation between the coefficients before and after the mapping: Each one of the result coefficients $a_i(x^{j(n-i)})$ will be mapped onto one of the original

positions a_l for $l = 0, \ldots, n-1$ when we execute modulo $x^n + 1$, or $x^n = -1$. In other words, the rotate operation of $\mathbf{a}$ is just a permutation of the original coefficients with some sign changes on some of them. However, we are more interested in the rotation operation in the NTT domain, as our message was packed with the CRT packing method.

NTT domain Given $\hat{\mathbf{a}}$, which is the evaluation of $a(x)$ at the n primitive $2n^{th}$ roots of a unity modulo t (Section 7.3): The rotate operation maps $\hat{\mathbf{a}}$ to $\hat{\mathbf{b}} = (\hat{b}_0, \hat{b}_1, \ldots, \hat{b}_{n-1})$, where $\hat{b}_i = a(x^j)|_{x=\omega_i} = a(\omega_i^j) = a(\psi^{(2i+1)j})$. We can see that the rotation on the NTT domain is also a permutation of the original coefficients, the difference compared to the operation on the coefficient domain being that it does not have the sign change effect on the original coefficients. However, this permutation is not the one we need for the HD computation algorithm (Algorithm 10): the i^{th} CRT component of $\hat{\mathbf{a}}$ is the i'^{th} CRT component of $\hat{\mathbf{b}}$, where i' satisfies $2i' + 1 \equiv (2i+1)j \mod 2n$.

We define $\bar{\mathbf{a}}$ to be also the evaluation of $a(x)$ at the n primitive $2n^{th}$ roots of unity modulo t, but in a different order, such that when applying the rotation operation, we will get back exactly the order we need for the HD computation algorithm. The idea is, with n a power of 2, those 2n-th roots of unity $\psi_{\mathbf{i}}$ for $i = 0, \ldots, n-1$ form a group that is isomorphic to the multiplicative group of integers modulo n (the set of congruence classes relatively prime to the modulo n), denoted by $\mathbb{Z}_{2n}^*$. Note that this group $\mathbb{Z}_{2n}^*$ is not a cyclic group, but that it can be generated by sets of "generators". For n is a power of 2, the group $\mathbb{Z}_{2n}^*$ can be generated by $(3)^x(-1)^y$ for $x \in \mathbb{Z}_{n/2}$ and $y \in \mathbb{Z}_2$. The polynomial $\bar{\mathbf{a}}$ can be represented by

$$\bar{\mathbf{a}} = (a(\psi^{3^0(-1)^0}), a(\psi^{3^1(-1)^0}), \ldots, a(\psi^{3^{n/2-1}(-1)^0}),$$
$$a(\psi^{3^0(-1)^1}), a(\psi^{3^1(-1)^1}), \ldots, a(\psi^{3^{n/2-1}(-1)^1}))$$

Each coefficient of $\bar{\mathbf{a}}$ can be indexed and denoted as follows:

$$\bar{a}_{i,i'} = a(\psi^{3^i(-1)^{i'}}) \text{ for } i \in \mathbb{Z}_{n/2} \text{ and } i' \in \mathbb{Z}_2$$

Considering the mapping $a(x) \overset{K_j}{\mapsto} a(x^j)$ again under this new representation yields the findings below. Denoting $b(x) = a(x^j)$, where $j \in \mathbb{Z}_{2n}^*$, every value of j can also be represented by $j = 3^{x_j}(-1)^{y_j}, (x_j \in \mathbb{Z}_{n/2} \text{ and } y_j \in \mathbb{Z}_2)$. Thus,

$$\bar{\mathbf{b}} = (b(\psi^{3^0(-1)^0}), b(\psi^{3^1(-1)^0}), \ldots, b(\psi^{3^{n/2-1}(-1)^0}),$$
$$b(\psi^{3^0(-1)^1}), b(\psi^{3^1(-1)^1}), \ldots, b(\psi^{3^{n/2-1}(-1)^1}))$$
$$= (a((\psi^{3^0(-1)^0})^{3^{x_j}(-1)^{y_j}}), a((\psi^{3^1(-1)^0})^{3^{x_j}(-1)^{y_j}}), \ldots, a((\psi^{3^{n/2-1}(-1)^0})^{3^{x_j}(-1)^{y_j}}),$$
$$a((\psi^{3^0(-1)^1})^{3^{x_j}(-1)^{y_j}}), a((\psi^{3^1(-1)^1})^{3^{x_j}(-1)^{y_j}}), \ldots, a((\psi^{3^{n/2-1}(-1)^1})^{3^{x_j}(-1)^{y_j}}))$$
$$= (a(\psi^{3^{0+x_j}(-1)^{0+y_j}}), a(\psi^{3^{1+x_j}(-1)^{0+y_j}}), \ldots, a(\psi^{3^{n/2-1+x_j}(-1)^{0+y_j}}),$$
$$a(\psi^{3^{0+x_j}(-1)^{1+y_j}}), a(\psi^{3^{1+x_j}(-1)^{1+y_j}}), \ldots, a(\psi^{3^{n/2-1+x_j}(-1)^{1+y_j}}))$$

This representation shows that the $\bar{\mathbf{b}}$'s coefficients $\bar{b}_{i,i'}$ are the $\bar{\mathbf{a}}$'s coefficients $\bar{a}_{i+x_j \mod n/2, i'+y_j \mod 2}$. Actually, there is a rotation here! It is not exactly a normal rotation, but it is a double rotation on the 2 halves, each one of them rotating by x_j position, then swapping if $y_j = 1$. In our context, we can take $x_j = 2^0, 2^1, \ldots, 2^{\log n/2}$ and $y_j = 0$ to add up the bits of each half, then we swap the halves by setting $x_j = 0$ and $y_j = 1$ to add up all the bits and get the final value of the Hamming Distance.

For ciphertexts that result from operations on polynomials, rotations can be mapped one by one and then added or multiplied:

$$a(x).b(x) \mapsto a(x^j).b(x^j)$$
$$a(x)+b(x) \mapsto a(x^j)+b(x^j)$$

Therefore, a ciphertext of the form $c = (\mathbf{as}+t\mathbf{e}+\mathbf{m}, -\mathbf{a})$ can be rotated by mapping each of its components $\mathbf{a}, \mathbf{s}, \mathbf{e}$ and $\mathbf{m}$ before adding up the results.

7.3.2 HD Homomorphic Computation and Key Switching

Given two bistring $\mathbf{a}$ and $\mathbf{b}$ of length n (for n being a power of 2), and the *rotate* function discussed above, the Hamming Distance $HD_{(\mathbf{a},\mathbf{b})}$ can be computed by first rotating and adding the bits of $\mathbf{a}+\mathbf{b}-2\mathbf{ab}$ (Algorithm 10). Provided that the BGV cryptosystem in use supports SIMD operations, the algorithm can run with homomorphic addition and multiplication. ($\triangle$ denotes component-wise homomorphic multiplication).

Algorithm 10 HD computation

```
1: procedure HDBGV(a, b)
2:     c ← a + b − 2a△b
3:     let n ← length(a)
4:     for i = 0, ..., n − 1 do
5:         let t ← rotate(c, 2^i)
6:         let c ← add(c, t)
7:     let c_swap ← swapHalves(c)
8:     let c ← add(c, c_swap)
9:     return c[0]
```

Another useful tool to control the noise during homomorphic operations is *key switching*, it keeps the multiplied ciphertexts to always be 2 elements of the ring R_q. We briefly discuss this operation.

Key Switching This operation allows a ciphertext encrypted with a key **s**' to be transformed into another ciphertext encrypted with a key **s**" (the dimension of **s**' and **s**" can be different). The idea is to first prepare a set of switching key pairs that can be considered the encryptions of the power of 2 of the bits of $\mathbf{s}'$, putting 2^i in front of them:

$$\begin{cases} \mathbf{a}_i \xleftarrow{r} R_q, \mathbf{e}_i \xleftarrow{r} \chi^n \\ \mathbf{b}_i \leftarrow \mathbf{a}_i\mathbf{s}'' + 2^i\mathbf{s}' + 2\mathbf{e}_i \end{cases}$$

Then, given a ciphertext encrypted with a key **s**' having the form $(\mathbf{a}', \mathbf{c}' = \mathbf{a}'\mathbf{s}' + 2\mathbf{e}' + \mathbf{m}')$, we want to switch the keys $(\mathbf{a}', \mathbf{c}')$ from $\mathbf{s}'$ to $\mathbf{s}''$. The process is accomplished through several steps.

1. Decompose $\mathbf{a}' \in R_q$ into its bit representation $\mathbf{a}' = \sum_0^{\log q - 1} 2^i \mathbf{a}'_i$.
2. Linear combination computation $\sum_0^{\log q - 1} \mathbf{a}'_i \cdot (\mathbf{a}_i, \mathbf{b}_i)$, with result

$$(\mathbf{a} = \sum_0^{\log q - 1} \mathbf{a}'_i\mathbf{a}_i, \mathbf{b} = \sum_0^{\log q - 1} \mathbf{a}'_i(\mathbf{a}_i\mathbf{s}'' + 2^i\mathbf{s}' + 2\mathbf{e}_i))$$

   or

$$(\mathbf{a}, \mathbf{a}\mathbf{s}'' + \mathbf{a}'\mathbf{s}' + \tilde{\mathbf{e}}')$$

3. Let $\mathbf{b}'' = \mathbf{c}' - \mathbf{b} = 2\mathbf{e}' - \mathbf{m}' - \mathbf{a}\mathbf{s}'' - \tilde{\mathbf{e}}$.

In other words, $(-\mathbf{a}, \mathbf{b}'')$ is the new ciphertext encrypted with $\mathbf{s}''$. With the discussed techniques, we are able to compute the Hamming Distance homomorphically, the next step being to compare the outcomes with the threshold in order to output the authentication result. In the third variant of our protocol, we have noticed that applying the ZKP technique only proves not practical enough, due to the large communication size. Next, we propose a new approach, which is based on different cryptographic techniques: Garbled Circuit and Oblivious Transfer.

7.4 Garbled Circuit and Oblivious Transfer

This section describes how to mix other cryptographic techniques with Homomorphic Encryption (HE) to improve efficiency while still achieving malicious client security. We first review the Garbled Circuit (GC) tool on how can it provide privacy preserving feature to multi-party protocols and how the circuit is built in our context. Next, we show a novel Oblivious Transfer (OT) technique to be used to retrieve the inputs for the circuit. The novel OT is based on BGV homomorphic operation and naturally becoming an interface between HE and GC.

7.4.1 Garbled Circuit

Context The "garbled circuit" concept was originally proposed in [132], to allow two parties to securely evaluate any functions (represented by a logic circuit). It is secure in the sense that the communicating parties do not acquire any information about each others' inputs, as they only have access to the output of the function. The idea is that, given a circuit (composed of gates connected by wires), the server "garbles" the circuit by randomly assigning two encryption keys $\omega_{j,0}$ and $\omega_{j,1}$ to each wire ω_j. The pair of keys represent respectively values 0 and 1, which are possible values of the logic gates' wires. The server then encrypts a truth table corresponding to each gate using nested encryption. Figure 7.4 illustrates how this can be done: Computing the output key of a gate requires knowing two of its input's keys.

After garbling the gates, the server sends the ciphertext tables and the keys corresponding to the server's input values to the client. The client uses an *Oblivious Transfer* technique (discussed below) to obtain the keys corresponding to the client's input values. After having the keys for each wire, the client can, in turn, decrypt the gates until discovering the final output keys (which can be encoded initially in a special way to represent value 0 or 1). The wires' keys are also referred to as *labels* in literature.

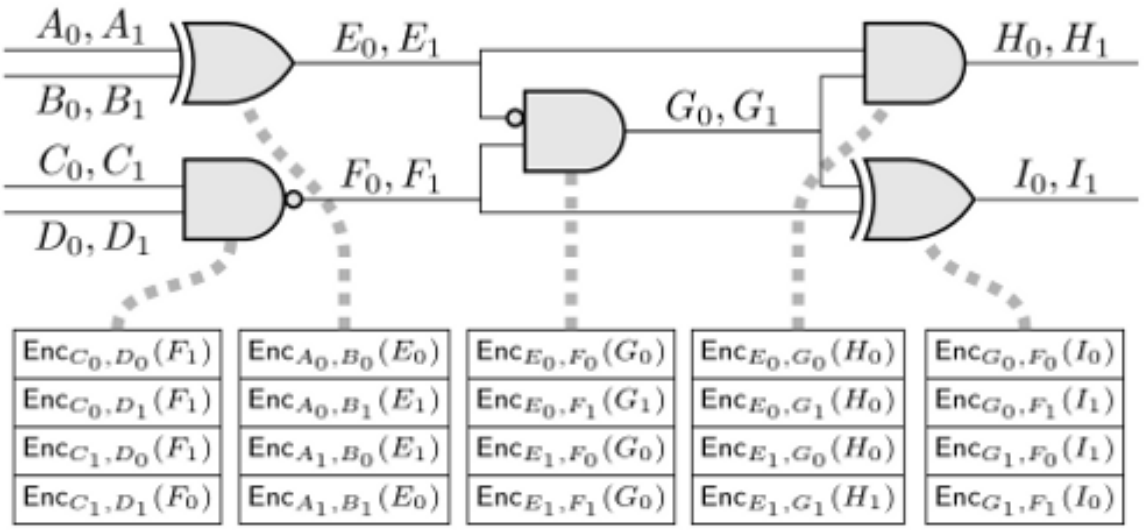


Fig. 7.4 Garbled Circuit example

Recent work provides optimizations to improve the computation and communication overhead associated with encrypt/decrypt operations and ciphertext tables' sizes. In [79], the authors proposed a modification to allow XOR gates to be evaluated *for free*: the labels of XOR gates are not chosen independently but by $\omega_{i,0} = \omega_{i,1} \oplus r$, for some random value of r. Pinkas et al. [106] introduced a way to reduce the communication size of binary gates by 25%: each gate can be specified by three ciphertexts instead of all four. Finally, [78] improve some commonly used circuits such as addition, comparision, etc., by reducing the number of non-XOR gates.

7.4.2 Using GC with Homomorphic Encryption in the protocol

We use a standard subtractor circuit for the operation $HD = HD' - r$: Given a 2 n-bit bitstring, the circuit is built from 1 half-subtractor and $n-1$ full-subtractors as in Figure 7.5, where a half-subtractor is built from 1 XOR gate and 1 AND gate, a full-subtractor is built from 2 half-subtractors and 1 OR gate (Figure 7.6)

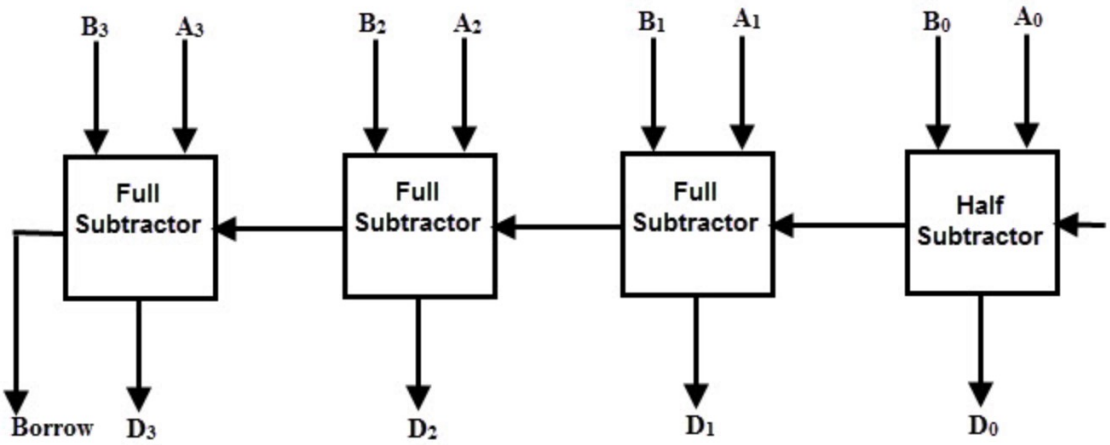


Fig. 7.5 Subtractor Circuit

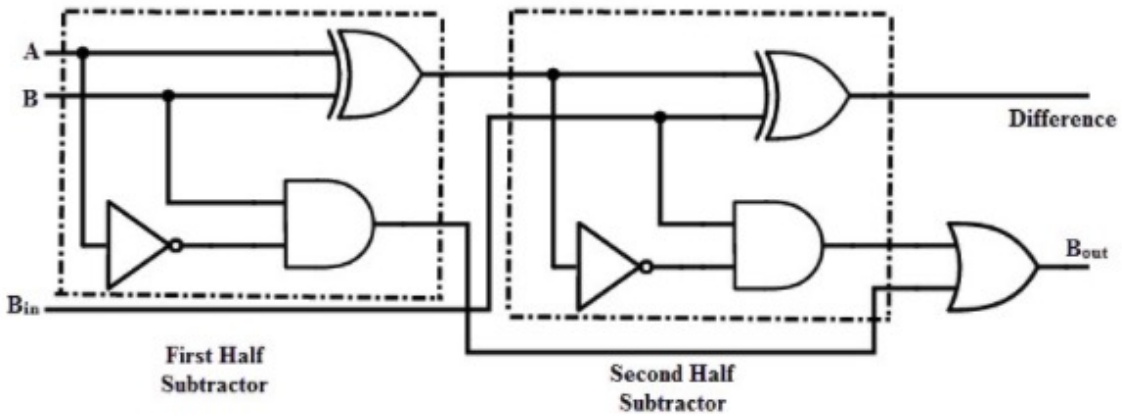


Fig. 7.6 Half-subtractor and Full-subtractor

In our context, let l be the bit length of the HD' and r (l is typically 10-12 bits for 1024-2048 bits biometrics data), which makes the number of subtractor logic gates equal to $5l$. The comparision circuit no longer needs to output 3 results (as a standard circuits would: Either A > B, A < B or A = B). Our aim is just to check whether $HD' - r < \tau$, so we simply need to have l NAND gates plus $l - 1$ XOR gates at hand for the comparison circuit. Figure 7.7 shows an example of such a configuration

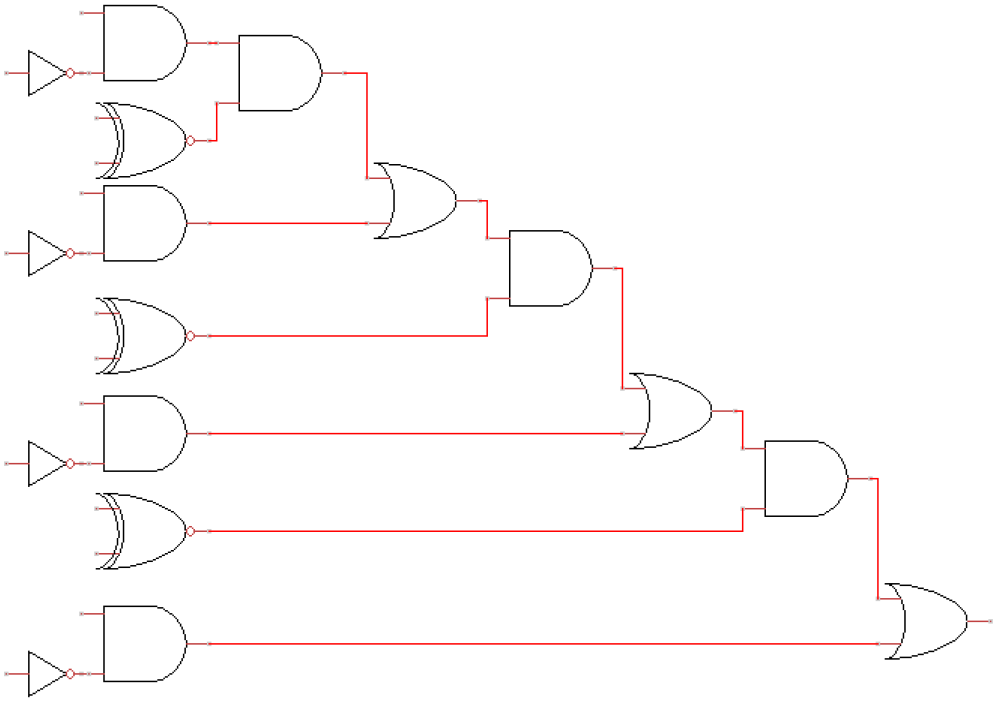


Fig. 7.7 Comparator Circuit

Garbled Circuit preparation We denote GC to be the garbled circuit prepared by the server. The inputs to the circuit include the server's GC_r and the client's GC_h, which are the cryptographic key *labels* corresponding to the server's input r and to the client's input HD' to the function *CompareHD*. The main function of the circuit is, first, to subtract $HD' - r = HD$ then compare $HD \overset{?}{<} \tau$, where τ is the constant value of the threshold required to decide upon the authentication result, which can be built into the circuit itself. We note that either the

server or the client can play the role of setting up the garbled circuit; in our work, we let the server take this role (Figure 7.10 - Step 5, 6, 7, 8): as we have been working on a *semi-honest* server model and a *malicious* client, this will save us one proof to assess that the circuit is generated correctly.

As pointed out previously, the server garbles the circuit *GC* and sends the result to the client together with its *labels* of r, which are $K_{r_i}^j$. Next, the client uses our OT technique (discussed in the following section) to select the correct *labels* of HD': the first part of the OT will ensure that the client will get back the encryptions $\left[\!\left[K_{h_i}^j\right]\!\right]$. The second part of the OT makes sure that the protocol is secure against *malicious* clients: it needs to prove that it decrypted and decomposed $[\![\mathbf{HD'}]\!]$ correctly (Figure 7.10 - Step 12,13). In other words, the client wants to prove that the encryptions $[\![h_i]\!]$ it sent are actually the encryptions of the bits of HD'_i. Recall that the server still has $[\![\mathbf{HD'}]\!]$, so the proof can be a homomorphical check of $[\![\mathbf{HD'}]\!] - \sum 2^i [\![h_i]\!] = [\![0]\!]$. However, in order to allow this proof to support homomorphic operations, we need to use the same encryption scheme for the OT protocol, in other words, we need an OT protocol based on BGV. In the next section, we discuss a this novel solution to the problem of interfacing Garbled Circuit with Homomorphic Encryption.

We note that inside the garbled circuit, AES is normally used to encrypt the *labels*, this encryption is a part of the garbled circuit and not related to the OT protocol we just described, and can still be used independently to ensure the performance of circuit evaluations. Next, we discuss an approach to design an OT protocol compatible with the BGV cryptosystem

7.4.3 Oblivious Transfer

"You take the blue pill, the story ends. You wake up in your bed and believe whatever you want to believe. You take the red pill, you stay in wonderland, and I show you how deep the rabbit hole goes." - Morpheus to Neo, The Matrix.

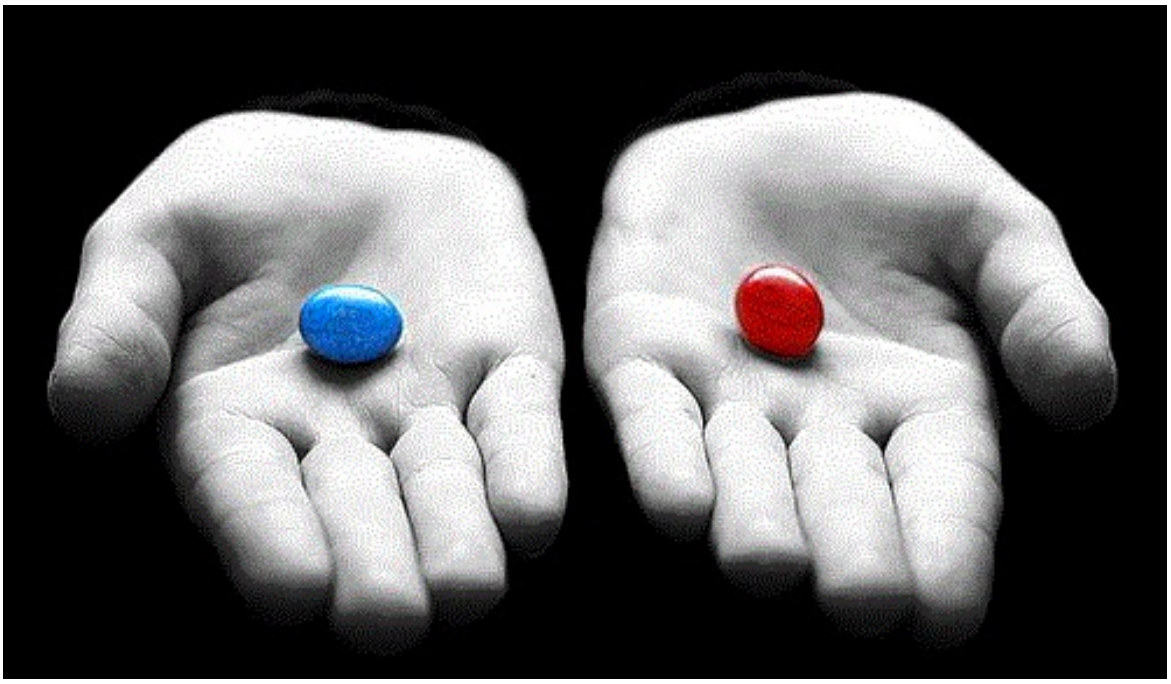


Fig. 7.8 Red Pill - Blue Pill

What if, in such situation, there were a protocol to give privacy to *Neo*: *Morpheus* should not be able to discover what option *Neo* has chosen. Moreover, the protocol can also provide privacy to *Morpheus*: *Neo* being thus unable to get information about the unchosen pill. In cryptography, Oblivious Transfer (OT) is a protocol able to support such a scenario. In its most basic form, it is a two-parties protocol between a *Sender* and a *Receiver*, denoted by $\binom{2}{1}$-OT. The *Sender* uses two private inputs x_0, x_1 and the *Receiver* uses one input bit s. At the completion of the protocol, the *Receiver* gets the bit x_s, without letting the *Sender* acquire any information about the value of s: $\binom{2}{1}$-OT$(x_0, x_1; s) = x_s$.

The general idea is that, when the receiver requests an item, the sender sends all the available items to him, unaware of which one has been requested. However, the response is encrypted in such a way that the receiver can only decrypt the one he requested. A concrete implementation example of the $\binom{2}{1}$-OT protocol based on discrete log DH is illustrated in figure 7.9 (This protocol is secure against *honest but curious* attacks). The receiver picks h_0, h_1 such that $h_0 h_1 = h$, he cannot access both $\log_g h_0$ and $\log_g h_1$. Given h_0, h_1, the sender

returns ElGamal encryptions of bits x_0, x_1 using h_0, h_1 as public keys. The receiver then decrypts one of the encryptions to recover either x_0 or x_1

DH-based Protocol

Sender		**Receiver**
$(x_0, x_1 \in \{0,1\})$		$(s \in \{0,1\})$
		$u \xleftarrow{r} \mathbb{Z}n$
		$h_s \leftarrow g^u$
	$\xleftarrow{h_0, h_1}$	$h_{1-s} \leftarrow h/g^u$
$u_0, u_1 \xleftarrow{r} \mathbb{Z}_n$		
$(A_0, B_0) \leftarrow (g^{u_0}, h_0^{u_0} g^{x_0})$		
$(A_1, B_1) \leftarrow (g^{u_1}, h_1^{u_1} g^{x_1})$	$\xrightarrow{(A_0,B_0),(A_1,B_1)}$	$x_s \leftarrow \log_g (B_s / A_s^u)$

Fig. 7.9 OT protocol based on DH

Efficient implementations of Oblivious Transfer can be found in [97]. The techniques found in [67] can reduce a large number of OT protocol executions to λ, where λ is the security parameter.

7.4.4 The BGV-based Oblivious Transfer protocol

In this thesis, we propose a new OT technique to be used with lattice-based cryptosystems. The technique can be used together with the garbled circuit protocol discussed earlier. The problem can be stated as follows: Given a BGV ciphertext of a bit selection h_i: $[\![h_i]\!]$, the server computes and returns a BGV encryption of the corresponding key selected by the bit. We observe that the following homomorphic operations will satisfy such a requirement:

$$\left[\!\left[K_i^j\right]\!\right] = \left[\!\left[h_i^j\right]\!\right] \cdot [\![K_i^1]\!] + (1 - \left[\!\left[h_i^j\right]\!\right]) \cdot [\![K_i^0]\!]$$

where K_i^0, K_i^1 are the *labels* corresponding to the wire inputs 0 or 1. The function includes one level of homomorphic multiplication, two additions and one multiplication with a constant. This approach is simple enough and it is secure against a *malicious client* model in our context because the client needs to prove that he actually encrypted $[\![h_i]\!]$ correctly already. Other applications of this OT should take into account that a malicious client can encrypt $[\![h_i]\!]$ incorrectly to obtain information about K_i^0 and/or K_i^1.

The other issue that we have to be careful about is *Circuit Privacy*: for a multi-factor attacker able to access the secret key but not the biometric template, it is possible to obtain information about the randomness of the operation. The randomness of the above operation is of the form $K_i^0 + (K_i^1 - K_i^0)r_0$, which is correlated to both keys. Again, we can mask the randomness of this leakage similarly to what we did in the second variant of the protocol. By a similar analysis, we can still show that computing the HD cannot be achieved with a significant level of probability (the privacy security requirement related to what the client sees is indistinguishable from the probability he has to compute the HD, which is not much more than random guessing).

7.5 The details

7.5.1 The protocol description

With a notation similar to the one used for previous protocols, we describe the last variant of the authentication system as follows (Figure 7.10)

1. In the enrollment stage, the client $\mathscr{U}$ extracts his template $[\![\mathbf{T}]\!]$ and sends it to the server $\mathscr{S}$

2. In the authentication stage, $\mathscr{U}$ extracts his query template $[\![\mathbf{Q}]\!]$ and sends it to the server

3. $\mathscr{U}$ uses a Schnorr-based ZKP (Section 7.2.1) to prove that the noise associated with $[\![\mathbf{Q}]\!]$ is small

4. $\mathscr{U}$ uses the Challenge-Response technique (Section 7.2.2) to prove that $[\![\mathbf{Q}]\!]$ is encrypting a binary message

5. If the proofs pass, $\mathscr{S}$ starts preparing the garbled circuit GC, it first samples $r \xleftarrow{r} \mathbb{Z}_t$ and decomposes it into a binary representation, assuming that r is k bits long.

6. $\mathscr{S}$ establishes k pair of keys $(K^0_{r_i}, K^1_{r_i})$ for the inputs of GC_r

7. $\mathscr{S}$ establishes k pair of keys $(K^0_{h_i}, K^1_{h_i})$ for the inputs of GC_h

8. $\mathscr{S}$ establishes the pair of keys for authentication result outputs (K^0_{out}, K^1_{out})

9. $\mathscr{S}$ encrypts the k keys of GC_h to obtain $\left[\!\left[K^0_{h_i}\right]\!\right], \left[\!\left[K^1_{h_i}\right]\!\right]$

10. $\mathscr{S}$ computes $[\![HD]\!] \leftarrow HDBGV([\![\mathbf{T}]\!], [\![\mathbf{Q}]\!])$

11. $\mathscr{S}$ masks the Hamming Distance $[\![\mathbf{HD}']\!] = [\![\mathbf{HD} + \mathbf{r}]\!]$ and sends the result together with the garbled circuit GC and the keys $K^j_{r_i}$ to $\mathscr{U}$, where j associates the key of GC_r corresponding to the r chosen by $\mathscr{S}$.

12. $\mathscr{U}$ decrypts $[\![\mathbf{HD}']\!]$

13. $\mathscr{U}$ decomposes the plaintext HD' into its binary representation $HD' = \sum_0^{k-1} 2^i h_i$

14. $\mathscr{U}$ re-encrypts the bits of the decomposition

15. $\mathscr{U}$ uses the ciphertext $[\![h_i]\!]$ in the Oblivious Transfer Protocol (Section 7.4.3) to retrieve the ciphertexts $\left[\!\left[K^j_{h_i}\right]\!\right]$

16. $\mathscr{U}$ decrypts the result to obtain another set of keys for the garbled circuit

17. $\mathscr{U}$ evaluates the circuit with the keys $(K^j_{h_i}, K^j_{r_i})$ and sends the result to the server

18. $\mathscr{S}$ compares this outcome with the output keys prepared in step 8 and outputs the final authentication result.

7.5.2 Implementation Result

We observe significant improvements in communication size thanks to the Challenge-Response technique (only 2 messages required). The Schnorr-based ZKP also contributes to this factor by reducing the number of rounds in proofs to 1, instead of 17 for similar settings ($n = 2048, t = 2048, \alpha q = 8, q \approx 2^{72}, FAR \approx 10^{-3}$). The total number of gates used in the garbled circuit for $n = 2048$ equals 25, which only cost ≈ 3.2 KB for the HD comparison purpose. All of these improvements come with a one time setup cost for preparing the switch keys for the HD computation operation: we need to rotate 10 times and swap 1 time to compute HD, each operation needs 1 key switch to transform the result back to the original ciphertext format (2 ring elements) before the next homomorphic operation. The total key switch size needed is $2 * n * \log q * 11$, which is approximately 389 MB in our setting. Considering the privacy features provided, we believe such one time set up cost is acceptable. We conclude the chapter with the comparison of the 4 protocols described in the thesis

Table 7.1 Protocols' features comparison

	Multi-factor	**HD Non-exposure**	**Low Init cost**	**Low Communication**
Protocol 1	No	No	Yes	Yes
Protocol 2	Yes	No	Yes	Yes
Protocol 3	Yes	Yes	Yes	No
Protocol 4	Yes	Yes	No	Yes

The Fourth Protocol

Client		**Server**
1. Registered template $\mathbf{T}$	$\llbracket \mathbf{T} \rrbracket \longrightarrow$	Store $\llbracket \mathbf{T} \rrbracket$
2. Query template $\mathbf{Q}$	$\llbracket \mathbf{Q} \rrbracket \longrightarrow$	Initiate Query Proof
	3. Prove small noise in $\mathbf{Q}$ $\longleftarrow$	
	4. Prove $\mathbf{Q}$ is binary $\longleftarrow$	
		Sample $r \xleftarrow{r} \mathbb{Z}_t$
		5. Setup garbled circuit GC, for $i = 1, \ldots, k$
		6. Setup $(K^0_{r_i}, K^1_{r_i})$ for GC_r
		7. Setup $(K^0_{h_i}, K^1_{h_i})$ for GC_h
		8. Setup (K^0_{out}, K^1_{out})
		9. Compute $\llbracket K^0_{h_i} \rrbracket, \llbracket K^1_{h_i} \rrbracket$
		10. $\llbracket \mathbf{HD} \rrbracket = HDBGV(\llbracket \mathbf{T} \rrbracket, \llbracket \mathbf{Q} \rrbracket)$
		Compute $\llbracket \mathbf{HD}' \rrbracket = \llbracket \mathbf{HD} + \mathbf{r} \rrbracket$
	$\llbracket \mathbf{HD}' \rrbracket \longleftarrow$	
	$GC \longleftarrow$	
	$K^j_{r_i}$ of $GC_r \longleftarrow$	
12. Decrypt $\mathbf{HD}' = dec(\llbracket \mathbf{HD} \rrbracket)$		
13. Decompose $\mathbf{HD}' = \sum_{i=0}^{k} 2^i h_i$		
14. Encrypt $\llbracket h_i \rrbracket$		
15. Initiate OT protocol		
	16. $\llbracket h_i \rrbracket \longrightarrow$	
	17. Prove correct decryption $\longrightarrow$	
	18. Retrieve $\left\llbracket K^j_{h_i} \right\rrbracket \longleftarrow$	
19. Decrypt $K^j_{h_i}$		
20. $K_{out} = GC(K^j_{h_i}, K^j_{r_i})$		
	$K_{out} \longrightarrow$	21. Compare K_{out} with K^0_{out}, K^1_{out}
		Output res = **Accept**|**Reject**

Fig. 7.10 The fourth Protocol

Chapter 8

Conclusion and Open Problems

Throughout this thesis, we have developed different solutions to privacy preserving biometrics authentication. Starting from a protocol with an inspiring technique of ciphertext packing but lacking of protection against a malicious client model, we constructed a Zero Knowledge Proof technique to provide a solution to the problem. The technique also causes the protocol not to rely on any trusted third party to keep the client's secret key for decrypting results from homomorphic operations. Discussing the next protocol, we noted that, to fully support multi-factor security, especially in the biometrics authentication context, the circuit privacy problem needs to be solved efficiently. We proposed to enforce the Renyi Divergence technique in the security analysis, so that the protocol can preserve its security level while still keeping the parameter settings within practical thresholds. Specifically, we showed that the noise within the results of homomorphic operations can be covered by not-too-large masks to add extra security to the ciphertext result. The third variant was our first attempt to do all the essential operations homomorphically: besides computing the Hamming Distance, we also wanted to compare the ciphertext result with some threshold value, under the malicious client mode hypothesis. Although the results were not practical enough in terms of communication size, we learned different techniques to balance the tradeoffs between communication and computation. In our last version, we proposed another alternative to

address all the weakness of the previous protocols. The solution includes the combinations of many advanced cryptographic techniques: Homomorphic Encryption, Zero Knowledge Proof, Garbled Circuit and Oblivious Transfer. However, we still notice a significant one time setup cost.

Some of the techniques developed in this thesis can be used to construct other privacy-preserving protocols. For example, the decomposition technique used to construct Stern-based ZKP can be applied to contexts where balancing communication/computation is necessary, or the Renyi Divergence security analysis technique can be applied to lattice-based cryptosystems requiring circuit privacy and keeping the parameters within practical thresholds. The new challenge-response and Oblivious Transfer submodules used in the last protocol can be used in different Homomorphic Cryptosystems in various contexts.

There are open questions with respect to the schemes we have put forward:

- We obtain security against a malicious client model and assume the server is *semi-honest*. Designing a practical scheme that is also secure against a malicious server is an interesting open challenge.

- We mainly used Hamming Distance to measure the difference between the registered and queried templates. There are many other approaches [72] yielding a better False Acceptance Rate and False Rejection Rate. The applicability of our technique to such methods is left for future work

- The operations of computing and comparing different types of distances can be applied to contexts unrelated to biometric authentication.

- Further optimizations to reduce computation time or communication size, such as the composition of the Zero Knowledge Proof (AND, OR, EQ) can be investigated to improve the practical implementation of protocols.

References

[Fuj] Fujitsu develops world's first slide-style vein authentication technology based on palm veins - fujitsu global. http://www.fujitsu.com/global/about/resources/news/press-releases/2017/0110-01.html. (Accessed on 01/23/2017).

[2] Abdalla, M., Fouque, P.-A., and Pointcheval, D. (2005). Password-based authenticated key exchange in the three-party setting. In *International Workshop on Public Key Cryptography*, pages 65–84. Springer.

[3] Ajtai, M. (1996). Generating hard instances of lattice problems. In *Proceedings of the twenty-eighth annual ACM symposium on Theory of computing*, pages 99–108. ACM.

[4] Ajtai, M. (1998). The shortest vector problem in l 2 is np-hard for randomized reductions. In *Proceedings of the thirtieth annual ACM symposium on Theory of computing*, pages 10–19. ACM.

[5] Ajtai, M. and Dwork, C. (1997). A public-key cryptosystem with worst-case/average-case equivalence. In *Proceedings of the twenty-ninth annual ACM symposium on Theory of computing*, pages 284–293. ACM.

[6] Albrecht, M., Cid, C., Faugere, J.-C., Fitzpatrick, R., and Perret, L. (2014). Algebraic algorithms for lwe problems.

[7] Alkim, E., Ducas, L., Pöppelmann, T., and Schwabe, P. (2016). Post-quantum key exchange-a new hope. In *USENIX Security Symposium*, volume 2016.

[8] Asharov, G., Jain, A., López-Alt, A., Tromer, E., Vaikuntanathan, V., and Wichs, D. (2012). Multiparty computation with low communication, computation and interaction via threshold fhe. In *Annual International Conference on the Theory and Applications of Cryptographic Techniques*, pages 483–501. Springer.

[9] Ateniese, G., Camenisch, J., Joye, M., and Tsudik, G. (2000). A practical and provably secure coalition-resistant group signature scheme. In *Annual International Cryptology Conference*, pages 255–270. Springer.

[10] Bai, S., Langlois, A., Lepoint, T., Stehlé, D., and Steinfeld, R. (2015a). Improved security proofs in lattice-based cryptography: using the rényi divergence rather than the statistical distance. In *International Conference on the Theory and Application of Cryptology and Information Security*, pages 3–24. Springer.

[11] Bai, S., Langlois, A., Lepoint, T., Stehlé, D., and Steinfeld, R. (2015b). Improved security proofs in lattice-based cryptography: using the rényi divergence rather than the statistical distance. In *International Conference on the Theory and Application of Cryptology and Information Security*, pages 3–24. Springer.

[12] Belenkiy, M., Camenisch, J., Chase, M., Kohlweiss, M., Lysyanskaya, A., and Shacham, H. (2009). Randomizable proofs and delegatable anonymous credentials. In *Advances in Cryptology-CRYPTO 2009*, pages 108–125. Springer.

[13] Belguechi, R., Alimi, V., Cherrier, E., Lacharme, P., Rosenberger, C., et al. (2011). An overview on privacy preserving biometrics. *Recent Application in Biometrics*, pages 65–84.

[14] Bellare, M. and Goldreich, O. (1992). On defining proofs of knowledge. In *Annual International Cryptology Conference*, pages 390–420. Springer.

[15] Bellare, M. and Goldwasser, S. (1989). New paradigms for digital signatures and message authentication based on non-interactive zero knowledge proofs. In *Conference on the Theory and Application of Cryptology*, pages 194–211. Springer.

[16] Bellare, M. and Rogaway, P. (2004). The game-playing technique. *International Association for Cryptographic Research (IACR) ePrint Archive: Report*, 331:2004.

[17] Bendlin, R. and Damgård, I. (2010). Threshold decryption and zero-knowledge proofs for lattice-based cryptosystems. In *Theory of Cryptography Conference*, pages 201–218. Springer.

[18] Bendlin, R., Damgård, I., Orlandi, C., and Zakarias, S. (2011). Semi-homomorphic encryption and multiparty computation. In *Annual International Conference on the Theory and Applications of Cryptographic Techniques*, pages 169–188. Springer.

[19] Benhamouda, F., Camenisch, J., Krenn, S., Lyubashevsky, V., and Neven, G. (2014). Better zero-knowledge proofs for lattice encryption and their application to group signatures. In *International Conference on the Theory and Application of Cryptology and Information Security*, pages 551–572. Springer.

[20] Blanton, M. and Gasti, P. (2011). Secure and efficient protocols for iris and fingerprint identification. In *European Symposium on Research in Computer Security*, pages 190–209. Springer.

[21] Bogdanov, A., Guo, S., Masny, D., Richelson, S., and Rosen, A. (2016). On the hardness of learning with rounding over small modulus. In *Theory of Cryptography Conference*, pages 209–224. Springer.

[22] Boneh, D. and Boyen, X. (2004). Short signatures without random oracles. In *International Conference on the Theory and Applications of Cryptographic Techniques*, pages 56–73. Springer.

[23] Boneh, D. and Katz, J. (2005). Improved efficiency for cca-secure cryptosystems built using identity-based encryption. In *Cryptographers' Track at the RSA Conference*, pages 87–103. Springer.

[24] Boneh, D. and Shacham, H. (2004). Group signatures with verifier-local revocation. In *Proceedings of the 11th ACM conference on Computer and communications security*, pages 168–177. ACM.

[25] Boneh, D. and Shoup, V. (2008). A graduate course in applied cryptography. *Version 0.1, fr om http://cryptobook. net*.

[26] Brakerski, Z., Gentry, C., and Vaikuntanathan, V. (2014). (leveled) fully homomorphic encryption without bootstrapping. *ACM Transactions on Computation Theory (TOCT)*, 6(3):13.

[27] Brakerski, Z. and Vaikuntanathan, V. (2011). Fully homomorphic encryption from ring-lwe and security for key dependent messages. In *Annual cryptology conference*, pages 505–524. Springer.

[28] Bresson, E., Chevassut, O., and Pointcheval, D. (2002). Group diffie-hellman key exchange secure against dictionary attacks. In *International Conference on the Theory and Application of Cryptology and Information Security*, pages 497–514. Springer.

[29] Bringer, J., Chabanne, H., Izabachène, M., Pointcheval, D., Tang, Q., and Zimmer, S. (2007). An application of the goldwasser-micali cryptosystem to biometric authentication. In *Information Security and Privacy*, pages 96–106. Springer.

[30] Camenisch, J. and Groß, T. (2008). Efficient attributes for anonymous credentials. In *Proceedings of the 15th ACM conference on Computer and communications security*, pages 345–356. ACM.

[31] Camenisch, J. and Lysyanskaya, A. (2001). An efficient system for non-transferable anonymous credentials with optional anonymity revocation. In *International Conference on the Theory and Applications of Cryptographic Techniques*, pages 93–118. Springer.

[32] Camenisch, J. and Stadler, M. (1997). Efficient group signature schemes for large groups. In *Advances in Cryptology—CRYPTO'97*, pages 410–424. Springer.

[33] Cappelli, R., Ferrara, M., and Maltoni, D. (2010). Minutia cylinder-code: A new representation and matching technique for fingerprint recognition. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 32(12):2128–2141.

[34] Chase, M., Kohlweiss, M., Lysyanskaya, A., and Meiklejohn, S. (2013). Malleable signatures: Complex unary transformations and delegatable anonymous credentials. *IACR Cryptology ePrint Archive*, 2013:179.

[35] Chen, Y. and Nguyen, P. (2011). Bkz 2.0: Better lattice security estimates. *Advances in Cryptology–ASIACRYPT 2011*, pages 1–20.

[36] Damgard, I., Geisler, M., and Kroigard, M. (2008). Homomorphic encryption and secure comparison. *International Journal of Applied Cryptography*, 1(1):22–31.

[37] Damgård, I. B., Pedersen, T. P., and Pfitzmann, B. (1993). On the existence of statistically hiding bit commitment schemes and fail-stop signatures. In *Annual International Cryptology Conference*, pages 250–265. Springer.

[38] Daugman, J. (2003). The importance of being random: statistical principles of iris recognition. *Pattern recognition*, 36(2):279–291.

[39] Dodis, Y., Ostrovsky, R., Reyzin, L., and Smith, A. (2008). Fuzzy extractors: How to generate strong keys from biometrics and other noisy data. *SIAM journal on computing*, 38(1):97–139.

[40] Ducas, L., Durmus, A., Lepoint, T., and Lyubashevsky, V. (2013). Lattice signatures and bimodal gaussians. In *Advances in Cryptology–CRYPTO 2013*, pages 40–56. Springer.

[41] Ducas, L. and Micciancio, D. (2015). Fhew: Bootstrapping homomorphic encryption in less than a second. In *Advances in Cryptology–EUROCRYPT 2015*, pages 617–640. Springer.

[42] Erkin, Z., Franz, M., Guajardo, J., Katzenbeisser, S., Lagendijk, I., and Toft, T. (2009). Privacy-preserving face recognition. In *International Symposium on Privacy Enhancing Technologies Symposium*, pages 235–253. Springer.

[43] Feige, U., Fiat, A., and Shamir, A. (1988). Zero-knowledge proofs of identity. *Journal of cryptology*, 1(2):77–94.

[44] Feng, J. and Jain, A. K. (2011). Fingerprint reconstruction: from minutiae to phase. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 33(2):209–223.

[45] Ferrara, M., Maltoni, D., and Cappelli, R. (2012). Noninvertible minutia cylinder-code representation. *IEEE Transactions on Information Forensics and Security*, 7(6):1727–1737.

[46] Fiat, A. and Shamir, A. (1986). How to prove yourself: Practical solutions to identification and signature problems. In *Conference on the Theory and Application of Cryptographic Techniques*, pages 186–194. Springer.

[FVC] FVC. Fvc-ongoing. https://biolab.csr.unibo.it/FVCOnGoing/UI/Form/Home.aspx. (Accessed on 12/04/2016).

[48] Galil, Z., Haber, S., and Yung, M. (1985). Symmetric public-key encryption. In *Conference on the Theory and Application of Cryptographic Techniques*, pages 128–137. Springer.

[49] Gentry, C. (2009). *A fully homomorphic encryption scheme*. PhD thesis, Stanford University. crypto.stanford.edu/craig.

[50] Gentry, C., Halevi, S., and Smart, N. P. (2012). Fully homomorphic encryption with polylog overhead. In *Annual International Conference on the Theory and Applications of Cryptographic Techniques*, pages 465–482. Springer.

[51] Gentry, C., Halevi, S., and Vaikuntanathan, V. (2010). i-hop homomorphic encryption and rerandomizable yao circuits. In *Proceedings of the 30th annual conference on Advances in cryptology*, pages 155–172. Springer-Verlag.

[52] Gentry, C., Peikert, C., and Vaikuntanathan, V. (2008). Trapdoors for hard lattices and new cryptographic constructions. In *Proceedings of the fortieth annual ACM symposium on Theory of computing*, pages 197–206. ACM.

[53] Gentry, C., Sahai, A., and Waters, B. (2013). Homomorphic encryption from learning with errors: Conceptually-simpler, asymptotically-faster, attribute-based. In *Advances in Cryptology–CRYPTO 2013*, pages 75–92. Springer.

[github] github. rimrim/rqbv. https://github.com/rimrim/rqbv. (Accessed on 03/04/2017).

[55] Goldreich, O. (2009). *Foundations of cryptography: volume 2, basic applications*. Cambridge university press.

[56] Goldreich, O., Goldwasser, S., and Micali, S. (1984). How to construct randolli functions. In *Foundations of Computer Science, 1984. 25th Annual Symposium on*, pages 464–479. IEEE.

[57] Goldreich, O., Micciancio, D., Safra, S., and Seifert, J.-P. (1999). Approximating shortest lattice vectors is not harder than approximating closest lattice vectors. *Information Processing Letters*, 71(2):55–61.

[58] Goldwasser, S. and Kharchenko, D. (2005). Proof of plaintext knowledge for the ajtai-dwork cryptosystem. In *Theory of Cryptography Conference*, pages 529–555. Springer.

[59] Goldwasser, S., Micali, S., and Rackoff, C. (1989). The knowledge complexity of interactive proof systems. *SIAM Journal on computing*, 18(1):186–208.

[60] Groth, J. (2007). Fully anonymous group signatures without random oracles. In *International Conference on the Theory and Application of Cryptology and Information Security*, pages 164–180. Springer.

[61] Guillou, L. C. and Quisquater, J.-J. (1990). A "paradoxical" identity-based signature scheme resulting from zero-knowledge. In *Proceedings on Advances in cryptology*, pages 216–231. Springer-Verlag New York, Inc.

[62] Halevi, S. and Micali, S. (1996). Practical and provably-secure commitment schemes from collision-free hashing. In *Annual International Cryptology Conference*, pages 201–215. Springer.

[63] Hanrot, G., Pujol, X., and Stehlé, D. (2011). Terminating bkz. *IACR Cryptology ePrint Archive*, 2011:198.

[64] Higo, H., Isshiki, T., Mori, K., and Obana, S. (2015). Privacy-preserving fingerprint authentication resistant to hill-climbing attacks. In *International Conference on Selected Areas in Cryptography*, pages 44–64. Springer.

[65] Hirano, T., Hattori, M., Ito, T., and Matsuda, N. (2013). Cryptographically-secure and efficient remote cancelable biometrics based on public-key homomorphic encryption. In *International Workshop on Security*, pages 183–200. Springer.

[66] Hoffstein, J., Pipher, J., and Silverman, J. (1996). Ntru: A ring-based public key cryptosystem. *Algorithmic number theory*, pages 267–288.

[67] Ishai, Y., Kilian, J., Nissim, K., and Petrank, E. (2003). Extending oblivious transfers efficiently. In *Crypto*, volume 2729, pages 145–161. Springer.

[68] Ishai, Y., Kushilevitz, E., Ostrovsky, R., and Sahai, A. (2007). Zero-knowledge from secure multiparty computation. In *Proceedings of the thirty-ninth annual ACM symposium on Theory of computing*, pages 21–30. ACM.

[69] Ishai, Y. and Paskin, A. (2007). Evaluating branching programs on encrypted data. In *TCC*, volume 4392, pages 575–594. Springer.

[70] Jain, A., Flynn, P., and Ross, A. A. (2007). *Handbook of biometrics*. Springer Science & Business Media.

[71] Jain, A. K., Nandakumar, K., and Nagar, A. (2008). Biometric template security. *EURASIP Journal on Advances in Signal Processing*, 2008:113.

[72] Jain, A. K., Nandakumar, K., and Ross, A. (2016). 50 years of biometric research: Accomplishments, challenges, and opportunities. *Pattern Recognition Letters*.

[73] Jain, A. K., Prabhakar, S., Hong, L., and Pankanti, S. (1999). Fingercode: a filterbank for fingerprint representation and matching. In *Computer Vision and Pattern Recognition, 1999. IEEE Computer Society Conference on.*, volume 2, pages 187–193. IEEE.

[74] Jin, A. T. B., Ling, D. N. C., and Goh, A. (2004). Biohashing: two factor authentication featuring fingerprint data and tokenised random number. *Pattern recognition*, 37(11):2245–2255.

[75] Katz, J. (2003). Efficient and non-malleable proofs of plaintext knowledge and applications. In *International Conference on the Theory and Applications of Cryptographic Techniques*, pages 211–228. Springer.

[76] Kawachi, A., Tanaka, K., and Xagawa, K. (2008). Concurrently secure identification schemes based on the worst-case hardness of lattice problems. In *International Conference on the Theory and Application of Cryptology and Information Security*, pages 372–389. Springer.

[77] Kilian, J. and Rogaway, P. (1996). How to protect des against exhaustive key search. In *Annual International Cryptology Conference*, pages 252–267. Springer.

[78] Kolesnikov, V., Sadeghi, A.-R., and Schneider, T. (2009). Improved garbled circuit building blocks and applications to auctions and computing minima. *Cryptology and Network Security*, pages 1–20.

[79] Kolesnikov, V. and Schneider, T. (2008). Improved garbled circuit: Free xor gates and applications. *Automata, Languages and Programming*, pages 486–498.

[80] Korenkevych, D., Xue, Y., Bian, Z., Chudak, F., Macready, W. G., Rolfe, J., and Andriyash, E. (2016). Benchmarking quantum hardware for training of fully visible boltzmann machines. *arXiv preprint arXiv:1611.04528*.

[81] Lacharme, P., Cherrier, E., and Rosenberger, C. (2013). Preimage attack on biohashing. In *Security and Cryptography (SECRYPT), 2013 International Conference on*, pages 1–8. IEEE.

[82] Langlois, A., Stehlé, D., and Steinfeld, R. (2014). Gghlite: More efficient multilinear maps from ideal lattices. In *Annual International Conference on the Theory and Applications of Cryptographic Techniques*, pages 239–256. Springer.

[83] Lee, Y., Chung, Y., and Moon, K. (2009). Inverse operation and preimage attack on biohashing. In *Computational Intelligence in Biometrics: Theory, Algorithms, and Applications, 2009. CIB 2009. IEEE Workshop on*, pages 92–97. IEEE.

[84] Libert, B., Ling, S., Mouhartem, F., Nguyen, K., and Wang, H. (2016). Signature schemes with efficient protocols and dynamic group signatures from lattice assumptions. In *International Conference on the Theory and Application of Cryptology and Information Security*, pages 373–403. Springer.

[85] Ling, S., Nguyen, K., Stehlé, D., and Wang, H. (2013). Improved zero-knowledge proofs of knowledge for the isis problem, and applications. In *Public-Key Cryptography–PKC 2013*, pages 107–124. Springer.

[86] Ling, S., Phan, D. H., Stehlé, D., and Steinfeld, R. (2017). Hardness of k-lwe and applications in traitor tracing. *Algorithmica*, 79(4):1318–1352.

[87] Lyubashevsky, V. (2008). Lattice-based identification schemes secure under active attacks. In *International Workshop on Public Key Cryptography*, pages 162–179. Springer.

[88] Lyubashevsky, V., Micciancio, D., Peikert, C., and Rosen, A. (2008). Swifft: A modest proposal for fft hashing. In *International Workshop on Fast Software Encryption*, pages 54–72. Springer.

[89] Lyubashevsky, V., Peikert, C., and Regev, O. (2010). On ideal lattices and learning with errors over rings. In *Annual International Conference on the Theory and Applications of Cryptographic Techniques*, pages 1–23. Springer.

[90] Mandal, A., Roy, A., and Yasuda, M. (2015). Comprehensive and improved secure biometric system using homomorphic encryption. In *International Workshop on Data Privacy Management*, pages 183–198. Springer.

[91] Micciancio, D. (2002). Generalized compact knapsacks, cyclic lattices, and efficient one-way functions. *Computational Complexity*, 16(4):365–411.

[92] Micciancio, D. and Regev, O. (2008). Lattice-based cryptography, book chapter in postquantum cryptography, dj bernstein and j. buchmann.

[93] Micciancio, D. and Regev, O. (2009). Lattice-based cryptography. In *Post-quantum cryptography*, pages 147–191. Springer.

[94] Micciancio, D. and Vadhan, S. P. (2003). Statistical zero-knowledge proofs with efficient provers: Lattice problems and more. In *Annual International Cryptology Conference*, pages 282–298. Springer.

[95] Naehrig, M., Lauter, K., and Vaikuntanathan, V. (2011). Can homomorphic encryption be practical? In *Proceedings of the 3rd ACM workshop on Cloud computing security workshop*, pages 113–124. ACM.

[96] Nagar, A., Nandakumar, K., and Jain, A. K. (2010). A hybrid biometric cryptosystem for securing fingerprint minutiae templates. *Pattern Recognition Letters*, 31(8):733–741.

[97] Naor, M. and Pinkas, B. (2001). Efficient oblivious transfer protocols. In *Proceedings of the twelfth annual ACM-SIAM symposium on Discrete algorithms*, pages 448–457. Society for Industrial and Applied Mathematics.

[98] Nguyen, P. Q. and Stehlé, D. (2006). Lll on the average. In *International Algorithmic Number Theory Symposium*, pages 238–256. Springer.

[99] Nguyen, P. Q. and Vallée, B. (2009). *The LLL algorithm: survey and applications*. Springer Science & Business Media.

[OPM] OPM. Opm says 5.6 million fingerprints stolen in cyberattack, five times as many as previously thought, the washington post. https://www.washingtonpost.com/news/the-switch/wp/2015/09/23/opm-now-says-more-than-five-million-fingerprints-compromised-in-breaches/. (Accessed on 10/23/2016).

[101] Osadchy, M., Pinkas, B., Jarrous, A., and Moskovich, B. (2010). Scifi-a system for secure face identification. In *Security and Privacy (SP), 2010 IEEE Symposium on*, pages 239–254. IEEE.

[102] Ostrovsky, R., Paskin-Cherniavsky, A., and Paskin-Cherniavsky, B. (2014). Maliciously circuit-private fhe. In *International Cryptology Conference*, pages 536–553. Springer.

[103] Paillier, P. et al. (1999). Public-key cryptosystems based on composite degree residuosity classes. In *Eurocrypt*, volume 99, pages 223–238. Springer.

[104] Pedersen, T. P. (1991). Non-interactive and information-theoretic secure verifiable secret sharing. In *Annual International Cryptology Conference*, pages 129–140. Springer.

[105] Peikert, C., Vaikuntanathan, V., and Waters, B. (2008). A framework for efficient and composable oblivious transfer. In *Annual International Cryptology Conference*, pages 554–571. Springer.

[106] Pinkas, B., Schneider, T., Smart, N. P., and Williams, S. C. (2009). Secure two-party computation is practical. In *Asiacrypt*, volume 9, pages 250–267. Springer.

[107] Pointcheval, D. (2005). Provable security for public key schemes. In *Contemporary cryptology*, pages 133–190. Springer.

[108] Pöppelmann, T., Ducas, L., and Güneysu, T. (2014). Enhanced lattice-based signatures on reconfigurable hardware. In *International Workshop on Cryptographic Hardware and Embedded Systems*, pages 353–370. Springer.

[109] Rackoff, C. and Simon, D. R. (1991). Non-interactive zero-knowledge proof of knowledge and chosen ciphertext attack. In *Annual International Cryptology Conference*, pages 433–444. Springer.

[110] Ratha, N. K., Chikkerur, S., Connell, J. H., and Bolle, R. M. (2007). Generating cancelable fingerprint templates. *IEEE Transactions on pattern analysis and machine intelligence*, 29(4).

[111] Regev, O. (2005). On lattices, learning with errors, random linear codes, and cryptography. *Journal of the ACM (JACM)*, 56(6):34.

[112] Regev, O. (2009). On lattices, learning with errors, random linear codes, and cryptography. *Journal of the ACM (JACM)*, 56(6):34.

[113] Rivest, R. L., Adleman, L., and Dertouzos, M. L. (1978). On data banks and privacy homomorphisms. *Foundations of secure computation*, 4(11):169–180.

[114] Roberts, C. (2007). Biometric attack vectors and defences. *Computers & Security*, 26(1):14–25.

[115] Sander, T., Young, A., and Yung, M. (1999). Non-interactive cryptocomputing for nc/sup 1. In *Foundations of Computer Science, 1999. 40th Annual Symposium on*, pages 554–566. IEEE.

[116] Schnorr, C.-P. (1989). Efficient identification and signatures for smart cards. In *Conference on the Theory and Application of Cryptology*, pages 239–252. Springer.

[117] Schnorr, C.-P. (1991). Efficient signature generation by smart cards. *Journal of cryptology*, 4(3):161–174.

[118] Schoenmakers, B. and Tuyls, P. (2006). Efficient binary conversion for paillier encrypted values. In *Annual International Conference on the Theory and Applications of Cryptographic Techniques*, pages 522–537. Springer.

[119] Šeděnka, J., Govindarajan, S., Gasti, P., and Balagani, K. S. (2015). Secure outsourced biometric authentication with performance evaluation on smartphones. *IEEE Transactions on Information Forensics and Security*, 10(2):384–396.

[120] Shahandashti, S. F., Safavi-Naini, R., and Ogunbona, P. (2012). Private fingerprint matching. In *Australasian Conference on Information Security and Privacy*, pages 426–433. Springer.

[121] Shor, P. W. (1994). Algorithms for quantum computation: Discrete logarithms and factoring. In *Foundations of Computer Science, 1994 Proceedings., 35th Annual Symposium on*, pages 124–134. Ieee.

[122] Shoup, V. (2004). Sequences of games: a tool for taming complexity in security proofs. *IACR Cryptology ePrint Archive*, 2004:332.

[123] Smart, N. P. and Vercauteren, F. (2014). Fully homomorphic simd operations. *Designs, codes and cryptography*, pages 1–25.

[124] Stehlé, D. and Steinfeld, R. (2010). Faster fully homomorphic encryption. *Advances in Cryptology-ASIACRYPT 2010*, pages 377–394.

[125] Stern, J. (1993). A new identification scheme based on syndrome decoding. In *Annual International Cryptology Conference*, pages 13–21. Springer.

[126] Teoh, A. B., Kuan, Y. W., and Lee, S. (2008). Cancellable biometrics and annotations on biohash. *Pattern recognition*, 41(6):2034–2044.

[127] Uludag, U. and Jain, A. K. (2004). Attacks on biometric systems: a case study in fingerprints. In *Proceedings of SPIE*, volume 5306, pages 622–633.

[128] Uludag, U., Pankanti, S., Prabhakar, S., and Jain, A. K. (2004). Biometric cryptosystems: issues and challenges. *Proceedings of the IEEE*, 92(6):948–960.

[129] Upmanyu, M., Namboodiri, A. M., Srinathan, K., and Jawahar, C. (2010). Blind authentication: a secure crypto-biometric verification protocol. *IEEE transactions on information forensics and security*, 5(2):255–268.

[130] Van Dijk, M., Gentry, C., Halevi, S., and Vaikuntanathan, V. (2010). Fully homomorphic encryption over the integers. In *Annual International Conference on the Theory and Applications of Cryptographic Techniques*, pages 24–43. Springer.

[131] Xagawa, K. and Tanaka, K. (2009). Zero-knowledge protocols for ntru: Application to identification and proof of plaintext knowledge. In *International Conference on Provable Security*, pages 198–213. Springer.

[132] Yao, A. C.-C. (1986). How to generate and exchange secrets. In *Foundations of Computer Science, 1986., 27th Annual Symposium on*, pages 162–167. IEEE.

[133] Yasuda, M., Shimoyama, T., Kogure, J., Yokoyama, K., and Koshiba, T. (2014). Practical packing method in somewhat homomorphic encryption. In *Data Privacy Management and Autonomous Spontaneous Security*, pages 34–50. Springer.

[134] Zhang, Y., Chen, Z., Xue, H., and Wei, T. (2015). Fingerprints on mobile devices: Abusing and eaking. In *Black Hat Conference*.