Grote: Group Testing for Privacy-Preserving Face Identification

Alberto Ibarrondo, Hervé Chabanne, Vincent Despiegel, Melek Önen

GROTE: Group Testing for Privacy-Preserving Face Identification

Alberto Ibarrondo
Idemia & EURECOM
Sophia Antipolis, France

Hervé Chabanne
Idemia & Telecom Paris
Paris, France

Vincent Despiegel
Idemia
Paris, France

Melek Önen
EURECOM
Sophia Antipolis, France

ABSTRACT

This paper proposes a novel method to perform privacy-preserving face identification based on the notion of group testing, and applies it to a solution using the Cheon-Kim-Kim-Song (CKKS) homomorphic encryption scheme. Securely computing the closest reference template to a given live template requires K comparisons, as many as there are identities in a biometric database. Our solution, named GROTE, replaces element-wise testing by group testing to drastically reduce the number of such costly, non-linear operations in the encrypted domain from K to up to $2\sqrt{K}$. More specifically, we approximate the max of the coordinates of a large vector by raising to the α-th power and cumulative sum in a 2D layout, incurring a small impact in the accuracy of the system while greatly speeding up its execution. We implement GROTE and evaluate its performance.

CCS CONCEPTS

• **Security and privacy** → **Privacy-preserving protocols**; *Biometrics.*

KEYWORDS

Secure Biometrics, Face Identification, Homomorphic Encryption, Privacy Enhancing Technologies

ACM Reference Format:
Alberto Ibarrondo, Hervé Chabanne, Vincent Despiegel, and Melek Önen. 2023. GROTE: Group Testing for Privacy-Preserving Face Identification. In *Proceedings of the Thirteenth ACM Conference on Data and Application Security and Privacy (CODASPY '23), April 24–26, 2023, Charlotte, NC, USA.* ACM, New York, NY, USA, 12 pages. https://doi.org/10.1145/3577923.3583656

1 INTRODUCTION

Modern data analytics & Machine Learning (ML) have disrupted many market sectors, ranging from the entertainment industry and manufacturing (e.g., content prediction algorithms, automatic fault detection), to more sensitive areas like healthcare or the public administration (e.g., early-stage cancer detection, fraud prosecution). This undeniable potential comes with numerous risks. Data misuse and theft, specially when dealing with personal data, are ever-present concerns that can be mitigated by resorting to privacy policies and techniques (as covered in present-day data protection legislations such as GDPR [15] in Europe or HIPAA [7] for medical records in United States).

These risks are exacerbated on certain applications. Hospitals and health specialists can only use the data of their direct patients, falling short on the data volume requirements to train accurate prediction models. Banks and finance institutions are limited to their locally available data to prevent fraud and prosecute tax evasion. Biometric recognition must employ trusted entities or secure hardware to store the biometric data required for their identification models, and data manipulation is subject to strict security rules.

Focusing on biometrics, personal data acquisition (e.g., face, fingerprint, iris, ...) and processing raises severe privacy concerns. Since biometric traits cannot be modified or re-issued, its protection is deemed indispensable. Standard cryptography enables secure storage and transmission, yet it falls short for privacy-preserving processing.

Various advanced cryptographic techniques arise to the challenge. *Fully Homomorphic Encryption* (FHE)[22] is a family of encryption schemes that support certain operations between ciphertexts (typically addition and multiplication), yielding the results of these operations when decrypting. Secure Multiparty Computation (MPC) covers a series of techniques (garbled circuits[62], secret sharing[56]) that split computation of a given function across multiple distinct parties, jointly collaborating to compute the result while remaining individually ignorant of the global computation.

In client-server scenarios like biometric identification or machine-learning-as-a-service (MLaaS), FHE shines at offloading heavy computation almost exclusively to one party. Moreover, by employing well established schemes like Brakerski-Gentry-Vaikuntanathan (BGV [6]), Brakerski / Fan-Vercauteren (BFV [21]) or Cheon-Kim-Kim-Song (CKKS[9]), we obtain private computation capabilities suited for biometric operations. Standard FHE provides privacy-preserving guarantees following an *Honest - But - Curious* threat model where parties involved perform the selected protocol without deviation while attempting to obtain as much information from the private data as possible. However, FHE only offers out-of-the-shelf encrypted addition and multiplication. Non-linear operations such as comparisons must be either reformulated using ring properties as in BFV & BGV [30] or approximated with polynomials in a small interval [10, 11, 37]. In both cases, several ciphertext-to-ciphertext multiplications are required to compute an encrypted comparison, considerably increasing its computational cost and that of comparison-based operations (e.g., Rectified Linear Units (ReLU), maximum of an array). Since biometric identification systems require multiple comparisons (one per record held in the biometric database of reference), reducing the cost of this operation

CODASPY '23, April 24–26, 2023, Charlotte, NC, USA

directly improves the practicality of FHE to protect these systems in real-world deployments.

Our Contribution. We propose a novel method to perform privacy-preserving biometric identification based on the notion of group testing, and instantiate it on FHE with the CKKS scheme. Securely computing the closest reference template to a given live template requires K comparisons, as many as there are identities in a biometric database. Our solution, named Grote, replaces element-wise testing by group testing to reduce the number of such costly, non-linear operations in the encrypted domain. More specifically, we approximate the max of the coordinates of a large vector (its infinity norm) by raising to the α-th power and cumulative sum (its α norm) in a 2D layout, incurring a small impact in the accuracy of the system while greatly speeding up its execution (1.5 times faster). We implement CKKS-based Grote and show that it outperforms the straightforward alternative based on batched comparisons.

This work is arranged as follows. Section 2 introduces the CKKS encryption scheme and the layout of a standard face identification system. Section 3 details our design of an FHE-enabled privacy-preserving group testing solution, and instantiates it to the BFV scheme. Next we validate this approach with experiments on biometric data in Section 4. We conclude the paper with a review of previous works in Section 5 and some takeaways in Section 6.

2 BACKGROUND

Notation

We use regular letters to denote scalars and polynomials (e.g., N, s) and bold letters for vectors of scalars and vectors of polynomials (e.g., $\mathbf{x}$, $\mathbf{pk}$). $\mathbf{x}[i]$ denotes the ith element of vector $\mathbf{x}$. R_L expresses a polynomial ring with integer coefficients modulo L. $p[i]$ denotes the ith coefficient/element of a polynomial p.

We note $\mathcal{U}_{[S]}$ to the uniform random distribution in the set S, and write $r \sim \mathcal{U}_{[S]}$ to sampling that distribution and assigning the sample to local variable r. $\mathcal{N}(\mu, \sigma)$ denotes a univariate gaussian distribution with mean μ and standard deviation σ. Given a sampling of an individual coefficient $p[j]$ from a distribution $\mathcal{D}$ ($p[j] \sim \mathcal{D}$), we denote the sampling of a polynomial p over a ring R_L as $p \leftarrow \mathcal{D}_{[R_L]}$.

We denote $[\cdot]_q$ the reduction modulo q, and $\lfloor \cdot \rfloor$, $\lfloor \cdot \rceil$, $\lceil \cdot \rceil$ the rounding to the previous, nearest and next integer respectively. When applied to polynomials or vectors, these reductions are performed coefficient/element-wise. For a polynomial a, we write its infinity norm as $\|a\|$. We employ $(\cdot)_?$ to denote the Boolean evaluation of the expression inside the brackets, e.g., $(3 > 2)_? = 1$.

2.1 Homomorphic Encryption

A *homomorphic* encryption (HE) scheme allows certain operations over ciphertexts, yielding a ciphertext equivalent to encrypting the result of those same plaintext operations. HE allows third parties to perform computations on encrypted data without learning the inputs or the computation results. In contrast to *partially* homomorphic encryption, which supports only one arithmetic operation (e.g. only additions[48] or only multiplications[51]), *fully* homomorphic encryption allows encrypted multiplications and additions, theoretically enabling private computation of arbitrary functions. This concept was conceived by Rivest et al. in the 1970s[50], but it remained abstract until Craig Gentry's first FHE scheme in 2009[22]. Since then, FHE has gone from theoretical breakthrough to practical deployment, dropping the initial 30 minutes required to compute a multiplication between two encrypted values down to less than 20 milliseconds. Even then, FHE multiplications are still around seven orders of magnitude slower than native CPU integer multiplication instructions. Therefore, practical FHE requires that applications be specifically adapted and optimized.

A variety of efficient schemes [5, 6, 8, 12, 13, 21] target slightly different settings. The majority of modern FHE schemes are based on the Learning with Errors (LWE) hardness assumption [49] and its variants (e.g., Ring LWE) and rely on a small amount of *noise* added during encryption to guarantee security. During homomorphic operations, this noise grows negligibly for additions, and significantly for multiplications. Should the noise grow too large, correct decryption would no longer be possible. Theoretically, a computationally expensive technique known as *bootstrapping* can be used to homomorphically reset the noise in a ciphertext. Instead, schemes are often instantiated with parameters large enough to allow the computation to complete without requiring bootstrapping.

We now introduce Cheon-Kim-Kim-Song (CKKS) [9] scheme, foundational to our work, leaving out other schemes such as the Brakerski/Fan-Vercauteren (BFV) [5, 21] or TFHE [12].

Scheme 1 CKKS($n, q = [q_0 * q_1 * \cdots * q_{d+1}], w, \sigma, B$)

CKKS.keygen(sk, $\mathbf{w}$ **)** $\rightarrow$ (sk, $\mathbf{pk}$):
- Sample $s \leftarrow \mathcal{S}_{[R_q]}$
- Sample $p_1 \leftarrow \mathcal{U}(R_q)$, and $e \leftarrow \mathcal{X}_{[R_q]}$
- Set $\mathbf{pk} = (p_0, p_1) = (-sp_1 + e, \quad p_1)$
- Output (sk, $\mathbf{pk}$)

CKKS.encr($\mathbf{pk}$, m **)** $\rightarrow \mathbf{c}_m$:
- Let $\mathbf{pk} = (p_0, p_1)$ a public key
- Sample $u \leftarrow \mathcal{S}_{[R_q]}$; $e_0 \leftarrow \mathcal{X}_{[R_q]}$; $e_1 \leftarrow \mathcal{X}_{[R_q]}$
- Output $\mathbf{c}_m = (c_{m_0}, c_{m_1}) = (q_0 m + up_0 + e_0, \quad up_1 + e_1)$

CKKS.add($\mathbf{c}_a, \mathbf{c}_b$ **)** $\rightarrow \mathbf{c}_{add}$:
- Let $\mathbf{c}_a = (c_{a_0}, c_{a_1})$, $\mathbf{c}_b = (c_{b_0}, c_{b_1})$ two ciphertexts.
- Output $\mathbf{c}_{add} = ([c_{a_0} + c_{b_0}]_q, \quad [c_{a_1} + c_{b_1}]_q)$

CKKS.decr($sk, \mathbf{c}_t$ **)** $\rightarrow \mathrm{m}_{res}$:
- Let $sk = s$ a secret key, $\mathbf{c}_t = (c_{t_0}, c_{t_1})$ a ciphertext.
- Output $m_{res} = \left\lfloor \frac{1}{q_{d+1}} [c_{t_0} + sc_{t_1}]_{q_{d+1}} \right\rceil$

CKKS.encode($\mathbf{a}$ **)** $\rightarrow m$:
- Let $\mathbf{a} \in \mathbb{R}^n/2$ an input vector with n/s elements.
- Output Polynomial $m \in R$ where $m = \mathrm{InvNTT}(\mathbf{a})$.

CKKS.decode(m **)** $\rightarrow \mathbf{a}_{res}$:
- Let $m \in Z^N$ the coefficients of an encoded polynomial of degree $N - 1$ in R.
- Compute $\mathbf{a}_{dec} = \mathrm{NTT}(m)$.
- Output $\mathbf{a}_{dec}$

2.1.1 CKKS scheme. The Cheon-Kim-Kim-Song (CKKS) [9] is a ring-learning-with-errors (RLWE) [44] homomorphic encryption

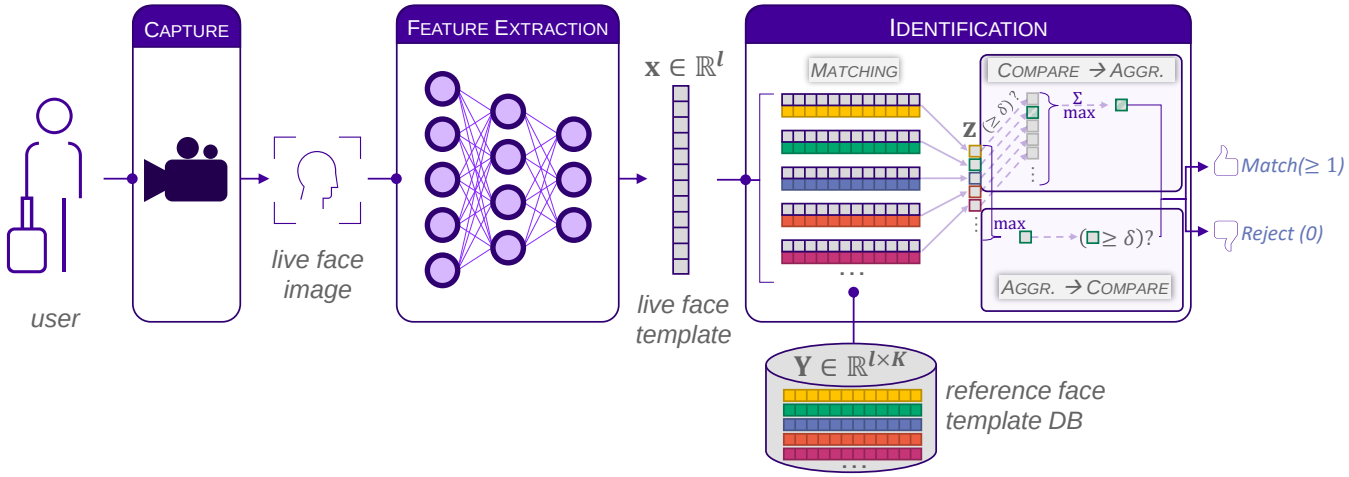

Figure 1: Face identification system

scheme, offering SIMD additions and multiplications on a vector of floating-point values. Contrary to BFV or TFHE, CKKS treats the noise introduced during encryption as part of the numerical approximation of the underneath values, thus yielding approximative results on its operations.

Messages are encoded in the plaintext space $\mathcal{R} = \mathbb{Z}[X]/(X^N + 1)$ of polynomials of degree up to $n - 1$ using an isomorphism of $\mathcal{R}$ with $\mathbb{C}^{n/2}$ and an approximated mapping, and then encrypted into the ciphertext space $R_q = \mathbb{Z}_q[X]/(X^n + 1)$ with n a power of 2 and q the product of a set of carefully chosen primes.

The CKKS scheme samples secrets from two distributions: the secret key distribution $\mathcal{S}_{[R_q]}$ with coefficients sampled from a uniform distribution $s[j] \sim \mathcal{S} \triangleq \mathcal{U}(\{-1, 0, 1\})$ so that $\mathcal{S}_{R_q} = \mathcal{Z}_{\{-1,0,1\}}[X]/(X^n + 1)$, and the error distribution $\mathcal{X}_{[R_q]}$ with coefficients $e[j] \sim \mathcal{X} \triangleq \mathcal{N}_{[-B,B]}(0, \sigma)$ sampled following a discrete Gaussian with standard deviation σ truncated into $[-B, B]$ where σ and B are two parameters of the scheme.

The security of CKKS is rooted in the hardness of the *decisional-RLWE problem*, informally stated as: given a uniformly random $a \leftarrow \mathcal{U}(R_q)$, a secret $s \leftarrow \mathcal{S}_{[R_q]}$, and an error term $e \leftarrow \mathcal{X}_{[R_q]}$, it is computationally hard for an adversary that does not know s and e to distinguish between the distribution of $(sa + e, a)$ and that of (b, a) where $b \leftarrow \mathcal{U}(R_q)$. A more formal definition can be found in Section 3.1 of [21].

While CKKS supports bootstrapping in theory [39], it is slow and thus the scheme is commonly instantiated with parameters large enough to handle the noise growth result of a limited number of multiplications (the *multiplicative depth*). Even then, a public *relinearization* should be used between multiplications to reshape the ciphertext without changing the underlying message, lowering noise growth and ciphertext size by employing a specific public key named relinearization key (rlk). A reasoning on how to select all these parameters in practice can be followed in Section 3.4 of [45].

Scheme 1 outlines the subset of algorithms conforming the CKKS scheme that are pertinent for this work.

2.2 Biometrics

Biometric systems are pattern recognition systems which establish the authenticity of a specific physiological or behavioral user's characteristic, that is, rely on "who/what you are" to identify you. These characteristics, broadly named biometric traits, are scanned and compressed into succinct representations called biometric templates. Biometric recognition systems use these templates to perform comparisons and establish & verify the identity of its users.

In essence, biometric systems have two phases. The *enrollment phase* involves registering users by collecting their biometric templates and storing them in a database. Subsequently, the *matching phase* (illustrated in Fig. 1) captures a live biometric template from a user seeking to authenticate/identify himself and compares (matches) it with the previously stored reference templates. Depending on the number of comparisons performed in the matching phase, we can have two scenarios:

- *Verification* (a.k.a. Authentication), a 1:1 similarity comparison between the live template and a single stored template. Yields a positive result if the similarity score is higher than a given threshold δ, negative otherwise. In a nutshell, it answers "Are you who you claim to be?".
- *Identification*, a 1:K comparison between the live template and a single stored template. Requiring K individual comparisons, it returns the ID of the stored template with highest similarity score or a negative result if no score is above the threshold δ. Hence, it provides the answer to "Who are you?".

Receiving its input image from a capture sensor (see Figure 1), the *feature extractor* component for face, iris and fingerprint biometrics in charge of template extraction is nowadays based on Deep Learning models applied to Vision [16, 17, 53]. Once extracted, the similarity/distance among templates is computed following a *similarity* metric. Typical metrics are the *hamming distance* ($\mathrm{HD}(\mathbf{x}, \mathbf{y}) = \sum \mathbf{x}_b[i] \oplus \mathbf{y}_b[i]$), and the *inner product (a.k.a. cosine similarity)* ($\mathrm{IP}(\mathbf{x}, \mathbf{y}) = \overline{\mathbf{x}} \cdot \overline{\mathbf{y}} = \sum \overline{\mathbf{x}}[i] \cdot \overline{\mathbf{y}}[i]$, where $\overline{\mathbf{x}} = \mathbf{x}/\|\mathbf{x}\|$ is the L2 normalized template vector). The highest score out of all the matching scores is then compared with a threshold δ to yield

"match" or "rejection" as the system's output. These thresholding and max operations can alternatively be swapped. Given a normalized input live template $\mathbf{x} \in \mathbb{R}^l$ and a database of normalized reference template with K identities $\mathbf{Y} \in \mathbb{R}^{l \times K}$, we compute identification based on cosine similarity following Eq. 1 to obtain k^*, the index of the identity yielding a positive matching:

$$\begin{gathered} \mathbf{z} \triangleq \ z[k] = \sum_{i=1}^{l} \mathbf{x}[i] \cdot \mathbf{Y}[i,k] \\ \text{Compare first: } \mathbf{z}_\delta = (\mathbf{z} \geq \delta)_? \ \rightarrow \ k^* = \arg\max_k \{\mathbf{z}_\delta[k]\} \\ \text{Max first: } k^*_{\mathbf{z}} = \arg\max_k \{\mathbf{z}[k]\} \ \rightarrow \ k^* = k^*_{\mathbf{z}}(\mathbf{z}[k^*_{\mathbf{z}}] \geq \delta)_? \end{gathered} \tag{1}$$

In the face recognition domain, modern feature extractors such as ArcFace-based Neural Networks[16] use the cosine similarity as metric. Besides, these feature extractors generate large templates ($l \in 128, 256, 512$) with a considerable floating point precision, often a requirement to yield low error rates. Contrary to other secure computation solutions, *fixed-point* encoding is not necessary to adapt the floating point template elements.

2.2.1 Towards Secure Biometrics. Following prior work [52, 61], a secure biometric system should address the following requirements:

- **Irreversibility**: Given a protected/secured template, an entity holding said template should not be able to recover the original biometric template unless it has access to the secret material (in the case of CKKS, the secret key).
- **Unlinkability**: Given a protected/secured template, an entity holding said template should not be able to link the encrypted biometric template of a user to his/her identity.
- **Cancellability**: The biometric system should be able to revoke a user's access to the system by deleting his/her biometric template from the database.
- **Accuracy preservation**: The secure version of the biometric system should not substantially degrade the accuracy of the original system.

3 OUR CONTRIBUTION

3.1 GROTE: Group Testing for Biometrics

The notion of group testing emanates from the field of statistics[19], and has been applied extensively across industries (e.g., in the healthcare sector [25], or in fault detection [41]). The essence of group testing consists of performing a check in a group of samples all at once, rather than checking on individual samples. For example, in the biometric identification domain one must compare each of the similarity scores resulting from $1:K$ matchings to a defined threshold δ, or alternatively compute the max of the vector of scores and test if this element is above the threshold. Figure 2 illustrates our proposal of a biometric matching algorithm based on group testing, that we name GROTE.

The main insight that drives our solution is that, as proposed in [23], for a sufficiently large exponent α we can approximate the max operation of Eq. 1 (also expressed as the infinity norm $\|\mathbf{z}\|_\infty$) by the α norm:

Algorithm 2 GROTE($\mathbf{z}$, h, w, δ_w,δ_h, α) $\rightarrow k^*$

Input: $\mathbf{z}$, a vector of size K holding the similarity scores of a live template with each of the K reference templates,
h, number of rows (or size of columns) of the group testing 2D matrix,
w, number of columns (or size of rows) of the group testing 2D matrix,
δ_w, a threshold for row-wise comparison,
δ_h, a threshold for column-wise comparison,
α, exponent for max approximation.

Output: Index k^* of the single score above the thresholds in the flattened score vector, set to zero if zero or several scores above the thresholds.

Assumption: With overwhelming probability there are either zero or one elements in $\mathbf{z}$ above δ.

1: $\mathbf{z} \in \mathbb{R}^K \rightarrow \mathbf{Z} \in \mathbb{R}^{h \times w}$. Reshape vector $\mathbf{z} \in \mathbb{R}^K$ as matrix $\mathbf{Z} \in \mathbb{R}^{h \times w}$. Fill empty spaces with zeros.
2: $\mathbf{Z} \rightarrow \mathbf{Z}^\alpha$. Raise each element of $\mathbf{Z}$ to the α power.

Cumulative Sum:
3: $\overrightarrow{\mathbf{w}} : \mathbf{w}[i] = \sum_{j=1}^{w} \mathbf{Z}^\alpha[i,j]$. Compute $\overrightarrow{\mathbf{w}} \in \mathbb{R}^h$, the row-wise sum of $\mathbf{Z}^\alpha$.
4: $\overrightarrow{\mathbf{h}} : \mathbf{h}[j] = \sum_{i=1}^{h} \mathbf{Z}^\alpha[i,j]$. Compute $\overrightarrow{\mathbf{h}} \in \mathbb{R}^w$, the column-wise sum of $\mathbf{Z}^\alpha$.

Comparison:
5: $\overrightarrow{\mathbf{w}}_{\delta_w} : \overrightarrow{\mathbf{w}}_{\delta_w}[i] = (\overrightarrow{\mathbf{w}}[i] \geq \delta_w)_?$. Compare elements of $\overrightarrow{\mathbf{w}}$ to threshold δ_w.
6: $\overrightarrow{\mathbf{h}}_{\delta_h} : \overrightarrow{\mathbf{h}}_{\delta_h}[j] = (\overrightarrow{\mathbf{h}}[j] \geq \delta_h)_?$. Compare elements of $\overrightarrow{\mathbf{h}}$ to threshold δ_h.

Validation.
7: Compute sums $v_w = \Sigma\overrightarrow{\mathbf{w}}_{\delta_w}$ and $v_h = \Sigma\overrightarrow{\mathbf{h}}_{\delta_h}$.
8: Compute $v_K = (v_h \cdot v_w \leq 1)_?$, check if up to one non-zero element in 2D matrix layout.

ArgMax.
9: $i^* = \Sigma_{i=1}^{w}(i \cdot \mathbf{w}_{\delta_w}[i])$. Compute i^*, row index of the above-threshold score.
10: $j^* = \Sigma_{j=1}^{h}(j \cdot \mathbf{h}_{\delta_h}[j])$. Compute j^*, column index of the above-threshold score.
11: $k^* = (wi^* + j^*)$. Compute the index of the above-threshold score.
12: **return** either k^* and v_K separately or $k^* \cdot v_K$.

$$\max(\mathbf{z}) = \|\mathbf{z}\|_\infty \approx \|\mathbf{z}\|_\alpha = \sqrt[\alpha]{\Sigma_{i=1}^{K}(\mathbf{z}[i]^\alpha)} \tag{2}$$

Moreover, we can turn it into a linear operation by removing the root and tweaking the threshold δ:

$$(\|\mathbf{z}\|_\infty \geq \delta)_? \approx (\Sigma_{i=1}^{K}(\mathbf{z}[i]^\alpha) \geq \delta^\alpha)_? \tag{3}$$

This approximation is more precise the higher the exponent α, and is rendered less precise the more elements there are in the vector. To balance out, we resort to pooling (aggregating) parts of the $\mathbf{z}$ vector inspired by *group testing*: we reshape $\mathbf{z} \in \mathbb{R}^K$ into a 2D

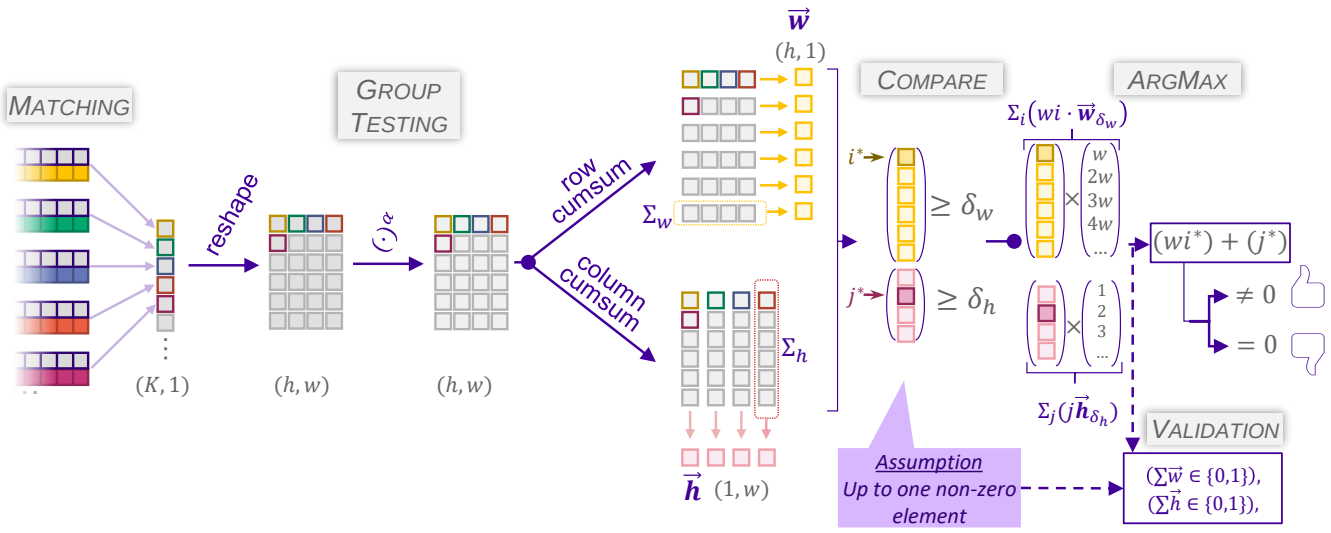

Figure 2: Group testing diagram

matrix[1] $\mathbf{Z} \in \mathbb{R}^{h \times w}$, obtaining h rows and w columns, and use Eq. 4 to approximate the maximum value row-wise and column-wise:

$$\begin{aligned} \vec{\mathbf{w}} \triangleq \vec{\mathbf{w}}[i] &= \sum_{j=1}^{w} (\mathbf{Z}[i,j])^{\alpha} \quad \forall i \\ \vec{\mathbf{h}} \triangleq \vec{\mathbf{h}}[j] &= \sum_{i=1}^{h} (\mathbf{Z}[i,j])^{\alpha} \quad \forall j \end{aligned} \tag{4}$$

We then resort to standard comparisons and ArgMax to pinpoint the element yielding a positive score, if any. The threshold δ, tied to the binary classification task arising from the biometrics scenario, must be tweaked to account for the pooling operation and the exponent α. For that purpose, we define two new group-wise thresholds δ_w and δ_h that must be set with properly adapted biometric-based experiments (see Section 4 for an example of such setting), and we use them to compare the row-wise and column-wise groupings $\vec{\mathbf{w}} \geq \delta_w$ and $\vec{\mathbf{h}} \geq \delta_h$ respectively:

$$\begin{aligned} \vec{\mathbf{w}}_{\delta_w} \triangleq \vec{\mathbf{w}}_{\delta_w}[i] &= (\vec{\mathbf{w}}[i] \geq \delta_w)_? \quad \forall i \\ \vec{\mathbf{h}}_{\delta_h} \triangleq \vec{\mathbf{h}}_{\delta_h}[j] &= (\vec{\mathbf{h}}[j] \geq \delta_h)_? \quad \forall j \\ i^* \triangleq \arg\max_i \ \vec{\mathbf{w}}_{\delta_w}[i] \\ j^* \triangleq \arg\max_j \ \vec{\mathbf{h}}_{\delta_h}[j] \end{aligned} \tag{5}$$

If the two comparisons yield a positive result at some indices i^* and j^* respectively, we conclude that the element $\mathbf{Z}[i^*, j^*]$ and the element $\mathbf{z}[w(i^*)+(j^*)]$ in the unrolled score vector contain a positive score, yielding the identity $w(i^*)+(j^*)$ as result. Otherwise, we conclude that no positive score was found, and return a negative result.

[1]Pooling with higher dimensionality is possible and straightforward to derive from our construction. We employ 2D pooling in our solution to allow for descriptive visuals.

Since $h \cdot w \approx K$, we achieve a reduction in the number of comparisons from K (max of K values) to $h+w$ (max on each dimension). In the most balanced case where $h \approx w$, we need $2\sqrt{K}$ comparisons, $\sqrt{K}/2$ times less comparisons than in the naïve solution.

The second insight we consider is that in the biometrics domain there is a very low chance of two or more simultaneous hits in a biometric database. This arises from the fact that the feature extractors are designed to separate (with respect to the similarity metric) templates belonging to different identities, and the reference database can be designed to minimize the likelihood of this event [2].

Relying on this fact, we can obtain the index of the non-zero value (arg max) that represents the positive identity k^* (if any) by multiplying the results of the elementwise comparison $\vec{\mathbf{w}}_{\delta}$ and $\vec{\mathbf{h}}_{\delta}$ with non-overlapping indexing vectors and summing up all the elements:

$$\begin{aligned} &(HW(\vec{\mathbf{w}}_{\delta_w}) \leq 1) \ \wedge \ (HW(\vec{\mathbf{h}}_{\delta_h}) \leq 1) \Rightarrow \\ &k^* = w \cdot i^* + j^* = \sum_{i=1}^{h} w \cdot i \cdot \vec{\mathbf{w}}_{\delta_w}[i] + \sum_{j=1}^{w} j \cdot \vec{\mathbf{h}}_{\delta_h}[j] \end{aligned} \tag{6}$$

As an additional precaution, we add an optional validation check to ensure that there is indeed up to a single non-zero element in each of the results of the elementwise comparison $\vec{\mathbf{w}}_{\delta}$ and $\vec{\mathbf{h}}_{\delta}$. Since a positive hit requires to have a non-zero element in both $\vec{\mathbf{h}}_{\delta}$ and $\vec{\mathbf{w}}_{\delta}$, we can reduce the check to:

$$\begin{aligned} \text{valid}(\vec{\mathbf{w}}_{\delta_w}, \vec{\mathbf{h}}_{\delta_h}) = \\ (HW(\vec{\mathbf{w}}_{\delta_w}) \leq 1) \ \wedge \ (HW(\vec{\mathbf{h}}_{\delta_h}) \leq 1) = \\ \left(\sum \vec{\mathbf{w}}_{\delta_w}[i]\right) \cdot \left(\sum \vec{\mathbf{h}}_{\delta_h}[j]\right) \leq 1 \end{aligned} \tag{7}$$

[2]One could measure the distance among reference templates while building the database, and reject/modify new reference templates that are too close to existing ones.

We detail this step-by-step group testing in Algorithm 2.

3.2 Applying GROTE to CKKS

In this section, we detail the steps to instantiate our group testing algorithm to CKKS. To this end, there are several aspects to take into consideration:

- All encrypted operations (additions, multiplications, comparisons) happen in SIMD fashion, applied simultaneously to all the elements of the underlying encoded vector. Given that CKKS ciphertexts encoded in a ring of polynomial degree n can hold $n/2$ floating-point values, there is room to encode multiple reference templates of length l per ciphertext ($n/2l$ templates per ciphertext to be precise, where $l < n/2$). To perform multiple scalar products at once during the matching phase, we encrypt a $n/2l$ times repetition of the live template inside the input ciphertext and operate on all repetitions at once. Similarly, the comparison operation will be applied elementwise to up to $n/2$ matching scores.
- A cumulative addition of s slots inside a ciphertext can be performed by iteratively rotating and adding a ciphertext with himself $\log_2(s)$ times.
- Ciphertext to ciphertext $c \times c$ encrypted multiplications are the costliest linear operations in CKKS both in terms of noise growth and in latency. Thus, optimizing the GROTE algorithm's runtime in CKKS involves minimizing the number of such operations. Some trade-offs to consider are:
  - Use of either non-encrypted live templates (sacrificing live template privacy) or non-encrypted reference templates (sacrificing the privacy of the reference template database), employing cheaper ciphertext-plaintext $c \times p$ multiplications for the scalar products of the matching step, also saving up in relinearization.
  - Keeping a low value of the exponent α, thus incurring in $\log_2(\alpha)$ multiplications, at the expense of a less precise approximation of the max operation.
  - Using a low number of multiplications (`depth` in [37]) in the polynomial approximation of the $sgn(x)$ function, in exchange for noisier results.
- Operating between ciphertexts requires them to have the same scale. To rescale ciphertexts (by mod-switching or multiplying with a plaintext) we will rely on the low-error techniques from [33].

3.2.1 Threat Model and Security Analysis. We consider a threat model whereby both the users and the Identification Server behave Semi-honestly, that is, they perform the computations faithfully while trying to obtain as much information as possible. We assume no collusion between the users, the IS and the BP.

The Biometric Provider must be trusted in the enrollment phase, as he is in charge of building the reference DB. Therefore, we can rely on the BP to perform the CKKS key generation, encrypt the DB and hold the CKKS secret key, decrypting the results sent by the Identification Server.

While the querying users seek to preserve the privacy of their live templates from the IS and the BP, the BP seeks to preserve the

Protocol 3 CKKS.GROTE($\langle \mathbf{x} \rangle$,$\{\langle \mathbf{Y} \rangle\}$, h, w, δ_w, δ_h, α)$\rightarrow \langle \mathbf{k}^* \rangle$

Input: $\{\langle \mathbf{Y} \rangle\} = \{\langle \mathbf{Y}[1 \dots n/2l] \rangle, \dots, \langle \mathbf{Y}[\dots K] \rangle, \}$, an encrypted database of K reference templates of length l split among $\lceil 2Kl/n \rceil$ ciphertexts.
$\langle \mathbf{x} \rangle$, the encrypted live template of length l repeated $n/2l$ times.
h, number of rows (or size of columns) of the group testing 2D matrix.
w, number of columns (or size of rows) of the group testing 2D matrix.
δ_w, a threshold for row-wise comparison.
δ_h, a threshold for column-wise comparison.
α, exponent for max approximation.

Output: Encrypted Index $\langle k^* \rangle$ of the single score above the thresholds in the flattened score vector, set to zero if no match in both dimensions

Assumption: With overwhelming probability there are either zero or one elements in $\mathbf{z}$ above δ.

Similarity:
1: **for** a in $1 \dots \lceil 2Kl/n \rceil$ **do**
2: $\quad \langle \mathbf{z}' \rangle^{(a)} = \langle \mathbf{Y} \rangle^{(a)} \cdot \langle \mathbf{x} \rangle$
3: $\quad$ **for** i in $1 \dots l$ **do**
4: $\quad\quad \langle \mathbf{z}' \rangle^{(a)} += (\langle \mathbf{z}' \rangle^{(a)} \ll i)$
5: $\quad$ **end for**
6: **end for**

Score Merge:
7: **for** t in $1 \dots \lceil 2K/n \rceil (= T)$ **do**
8: $\quad$ **for** i in $1 \dots l$ **do**
9: $\quad\quad \langle \mathbf{z} \rangle^{(t)} = \langle \mathbf{z}' \rangle^{(t \cdot 2Kl/n+i)} \cdot \mathbf{mask}_{n/2}(2l/n + i)$
10: $\quad$ **end for**
11: **end for**

Group Testing:
12: **for** t in $1 \dots T$ **do**
13: $\quad$ **for** _ in $1 \dots \log_2 \alpha$ **do**
14: $\quad\quad \langle \mathbf{z} \rangle^{(t)} = \langle \mathbf{z} \rangle^{(t)} \cdot \langle \mathbf{z} \rangle^{(t)}$
15: $\quad$ **end for**
16: **end for** $\langle \overrightarrow{\mathbf{w}} \rangle = -\delta_w$; $\left\langle \overrightarrow{\mathbf{h}} \right\rangle = -\delta_h$
17: **for** t in $1 \dots T$ **do**
18: $\quad \langle \overrightarrow{\mathbf{w}} \rangle += \sum_{i=1}^{n/2w} ((\langle \mathbf{z} \rangle^{(t)} \cdot \mathbf{mask}_{n/2}(wi)) \ll wi)$.
19: $\quad \left\langle \overrightarrow{\mathbf{h}} \right\rangle += \sum_{h=1}^{n/2h} ((\langle \mathbf{z} \rangle^{(t)} \cdot \mathbf{mask}_{n/2}(j)) \ll j)$.
20: **end for**
21: $\langle \mathbf{z}_{Grote} \rangle = concat(\langle \overrightarrow{\mathbf{w}} \rangle, \left\langle \overrightarrow{\mathbf{h}} \right\rangle)$

Comparison:
22: $\langle \mathbf{z}_\delta \rangle = \text{OptMinimaxComp}(\langle \mathbf{z}_{Grote} \rangle, 0, [\dots])$

ArgMax.
23: $\langle k^* \rangle = \sum_{h+w} \langle \mathbf{z}_\delta \rangle \cdot \{w, 2w, \dots, hw, 1, 2, \dots, w-1\}$.

Validation (optional):
24: $\langle v \rangle = \sum_w \langle \mathbf{z}_\delta \rangle \cdot ((\sum_h \langle \mathbf{z}_\delta \rangle) \ll w)$,
25: **return** either $(\langle k^* \rangle, \langle v \rangle)$ or $\langle \mathbf{z}_\delta \rangle$.

privacy of the reference DB from the IS and the querying users. Privacy of the inputs and intermediate computation results is assured by the use of CKKS thanks to the hardness of the LWE problem.

However, the decrypted output does reveal some information about both the live template and the reference template database. This input leakage has been studied in the literature before for inner product based privacy-preserving solutions [24, 28, 40]. To reduce this leakage it is advisable to output the minimal possible amount of information. In our case, the argmax would contain the least information possible, worsened very slightly by outputting also the validation result. Even if the full comparison results were decrypted, performing encrypted comparison instead of outputting the similarity scores does hinder input templates extraction attacks considerably. In this line, we argue GROTE to be inherently resilient against input leakage.

All in all, our secure biometric solution directly inherits the security guarantees of CKKS, guaranteeing unlinkability and irreversibility out of the shelf based on the hardness of the LWE problem: it ensures that encrypted templates cannot be decrypted to reveal the original template without the secret key, nor do they yield information about the identity of the user.

3.2.2 The end-to-end identification protocol. Building upon the GROTE algorithm, we design a protocol for privacy-preserving biometric identification based on CKKS, depicted in Figure 3. As *setup* for our scenario, the Biometric Provider (BP) acts as trusted entity and collects the reference templates, generating a pair of public and secret keys, and encrypting the reference template database with SIMD for compression (with $n/2l$ ref. templates per ciphertext). This encrypted database is deployed to an Identification Server (IS), in charge of the full encrypted computation, while the public key is then distributed to the users.

A user wishing to identify himself extracts his live template, encrypt it with $n/2l$ repetitions into a single ciphertext $\langle \mathbf{x} \rangle$, and then queries the IS with it. The server performs the following steps:

(1) *Similarity*: Compute the scalar product between the live template ciphertext $\langle \mathbf{x} \rangle$ and every ciphertext in the encrypted database of reference templates $\langle \{\mathbf{Y}[1], \ldots, \mathbf{Y}[n/2l]\} \rangle, \ldots, \langle \{\ldots, \mathbf{Y}[K]\} \rangle$. Making use of SIMD multiplications followed by cumulative additions ($\log_2(l)$ rotations and additions per DB ciphertext), the server obtains K similarity scores (one per record) distributed evenly among $\lceil 2Kl/n \rceil$ ciphertexts $\langle \mathbf{z'} \rangle_1, \ldots, \langle \mathbf{z'} \rangle_{\lceil 2Kl/n \rceil}$.
(2) *Score merge*: merge of the score ciphertexts by multiplying with masking plaintexts (vectors with ones in the slots containing scores, zeros elsewhere), and then adding all the masked scores into $T = \lceil 2K/n \rceil$ ciphertexts $\langle \mathbf{z} \rangle_1, \ldots, \langle \mathbf{z} \rangle_T$.
(3) *Group testing*: as described in Algorithm 2, to approximate the *max* by a sum of α-powered values in a 2D matrix layout. This involves, per each of the T score ciphertexts, $\log_2(\alpha)$ $c \times c$ multiplications, two cumulative additions for the "row-wise" and "column-wise" vectors (using $\log_2(h)$ and $\log_2(w)$ rotations & additions respectively), the subtraction of their respective thresholds δ_w and δ_h and their merging (as in step 2) into a single ciphertext.
(4) *Comparison*: with zero carried out following the procedure described in [36]. This involves $\log_2(\texttt{depth})$ multiplications and additions.
(5) *Argmax*: by multiplying with constant index vectors and a cumulative sum to obtain the identity (if any) of the live template's provider.
(6) *Validation*: Since comparisons are too expensive to justify one for validation, we are left with two alternatives:
   - Dropping the Argmax step entirely and decrypting the comparison result directly.
   - Performing the cumulative sum of all the elements resulting from the comparison in each vector ($\log_2(max(h, w))$ rotations and additions), multiplying the two results together and outputting it alongside the Argmax result.

A naïve solution would require K similarity computations and K comparisons (plus the Argmax and validation steps), whereas adding the *group testing* step reduces the number of comparisons to $h + w$ in exchange for $log_2(\alpha)$ multiplications and a cumulative addition. As we discuss in Section 3.2.3, the approximated comparison from [37] requires far more multiplications ($\geq$ 11) than *group testing* (for $\alpha \leq 32$). Crucially, due to the SIMD feature of CKKS, the GROTE save-up kicks in for $K > n/2$, since otherwise a single comparison would suffice for the identification and the group testing step would be redundant.

We detail the full CKKS-based computation in Algorithm 3.

3.2.3 Choosing Parameters. The GROTE related parameters h and w will be set based on performance experiments in Section 4, whereas α will be tested for values $\alpha \in \{2, 4, 8, 16, 32\}$.

The biometric template size l is set by the architecture of biometric feature extractors. In the face biometrics domain, they often amount to $l \in \{128, 256, 512\}$ to speed up non-encrypted similarity score calculations (e.g. using AVX instructions for the multiplication). We will use the smaller $l = 128$ to maximize the number of reference templates per ciphertext, and thus the number of comparisons per ciphertext.

The CKKS scheme parameters n (polynomial ring degree) and q (modulus of the polynomial coefficients) are tied to each other and linked to the sought-out security parameter. To obtain an equivalent security of 128 bits, and according to [1], $n = 16384$ allows $\log_2 q \leq 438$ bits and $n = 32768$ allows $\log_2 q \leq 881$ bits[3]. Setting q is directly related to the number of multiplications (depth) of the full arithmetic circuit d, and standard strategies to set it [3, 37–39] consist of composing a chain of primes $\{q_i\}$ $\forall i \in \{1, \ldots, d\}$ such that $q = \prod_{i=1}^{d+1} q_i$, with $q_1 = q_{d+1} \approx 2^{60}$ to ensure high precision in encoding/decoding and $q_2, \ldots, q_d$ chosen to be close to the CKKS encoding scale Δ to reduce rounding errors when performing rescaling/mod-switching. Smaller values of Δ yield less precise approximations of the $sgn(x)$ function [37], but also reduce required total size of q to the point where it might permit the use of lower n (reducing the ciphertext sizes and thus speeding up their operations). As such, we find a good trade-off in setting $\Delta = 2^{30}$, which drives us to set $q \approx 2^{120} \cdot 2^{30d}$ (permitting $d \leq 10$ for $n = 16384$ and $d \leq 25$ for $n = 32768$).

To define d, we need to count the total amount of multiplications. We require one multiplication for the *similarity* computation and one for the merging, $\log_2(\alpha)$ multiplications for the *group testing*

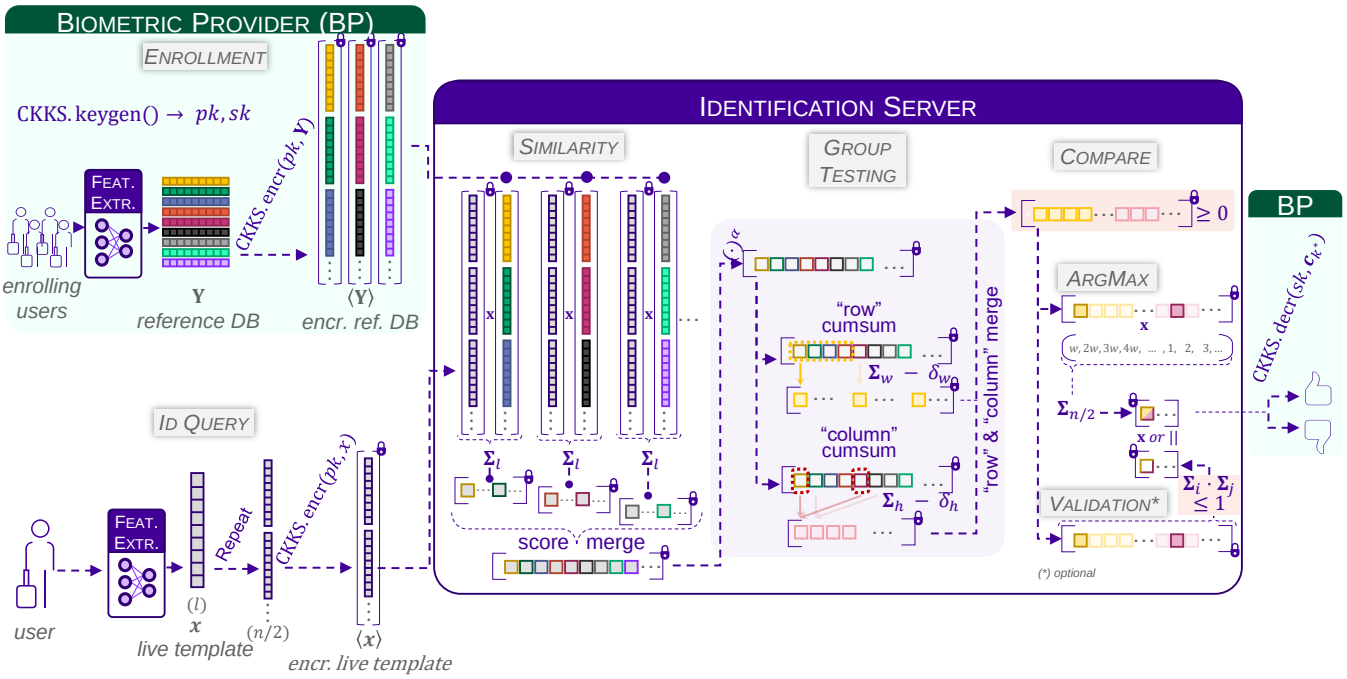

Figure 3: CKKS group testing

step plus one for the extra merging. Based on the accuracy of the approximation of the $sgn(x)$ function in Table V of [37] we employ $\texttt{depth} = 11$ to get a wide margin $\eta = 2^{-26}$, thus requiring 11 multiplications per comparison. One last multiplication is required for the argmax operation (and optionally one more for the validation step). We thus set $d = 1 + 1 + \log_2(\alpha) + 1 + 11 + 1 = 15 + \log_2(\alpha)$, leading us to confidently set $n = 32768$ for $\alpha \leq 32$.

3.2.4 On (not) applying GROTE *to BFV or TFHE.* While other FHE schemes could, on the surface, benefit from the GROTE approach, we argue that the CKKS scheme is the most suitable for this application.

The BGV/BFV schemes [6, 21] operate on integers, thus computing a value z^α requires a lot of space in the ciphertext to avoid the modulo kicking in, forcing costly parameter selection in detriment of speed. Besides, the state of the art encrypted comparison techniques [30] are not suited for full SIMD computation as they require multi-slot encoding, thus rendering the scalar product more expensive and complex than in CKKS.

The TFHE scheme [12] deals with bit-level homomorphic operations, thus needing a lot of ciphertexts to encode elements with high precision from the biometric templates, a requirement to maintain high accuracy in the system. This makes big integer operations very costly, thus rendering a z^α raising operation prohibitively expensive. Besides, the TFHE scheme is not optimized for SIMD computation, thus a TFHE-based biometric identification solution would require a sizeable horizontal scaling in the hardware to achieve the same performance as CKKS.

[3]For a secret key ternary distribution, with coefficients sampled from $\{-1, 0, 1\}$. Other distributions offer similar limitations.

3.2.5 On cancellability of GROTE*-CKKS.* In order to provide cancellability, that is, the ability to invalidate compromised records from the reference DB, the keys used for protecting these records should be easily changed. We can achieve this in our current solution by employing CKKS key-switching: the trusted setup may also generate a set of key-switching keys that correspond to different pairs of public-secret keys. In the event of a leaked secret key, the reference DB can be key-switched to a new key, thus invalidating the compromised records. In the case of a compromised record, the trusted setup can deliver a new freshly-encrypted (with a new key pair) reference DB without the compromised record to the BP, with the guarantee that the two DBs are unlinkable (as per the hardness of the LWE problem) and thus the BP cannot tell which record was removed. Note that public keys, used to encrypt fresh templates, can be publicly without compromising the system as per the standard CKKS security model.

An alternative worth studying in future work is the use of multi-key FHE [42], where each user is assigned its own public key, and thus a change on his key pair would only invalidate his records. This would require a more complex key management system, but would allow for a more fine-grained cancellability.

4 EXPERIMENTS

4.1 Setup

We implement our solution using the Pyfhel [29] Python library, with the SEAL [55] C++ library acting as backend. We use an ArcFace based[16] feature extractor[4]with templates of size $l = 128$.

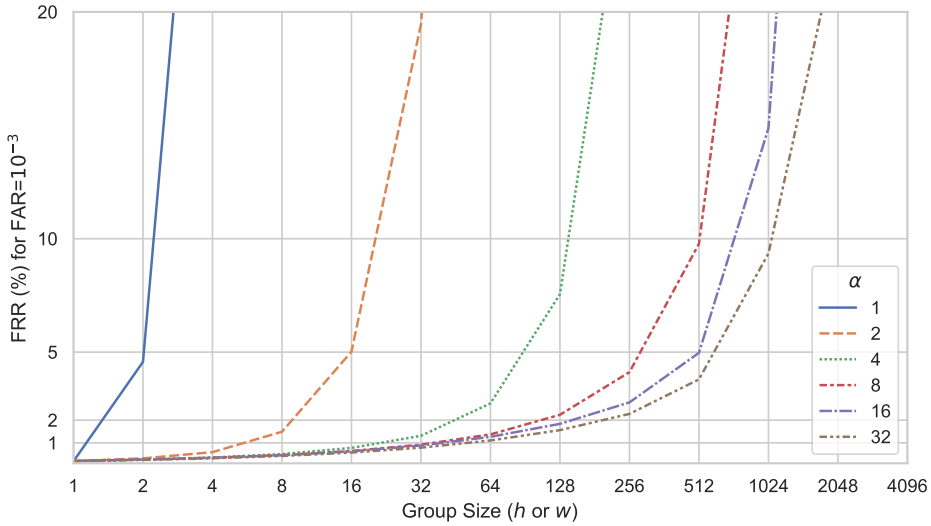

Figure 4: Face identification precision (False Rejection Rate at a fixed False Acceptance Rate) of 1D group testing based on the group/pool size and the exponent α.

The experiments were run in an Intel(R) Core(TM) i7-7800X CPU and averaged over at least 10 runs.

We measure the biometric precision with face identification benchmarks using the Labeled Faces in the Wild (LFW) dataset[27] consisting of 13233 112x112px real face images of famous people. We employ the widespread False Acceptance Rate (FAR) and False Rejection Rate (FRR) as metrics[31]. Typically, robust identification systems enforce $FAR \leq 10^{-3}$, obtaining a corresponding higher FRR.

Following the same procedure as for the threshold δ in standard biometric solutions, we set the thresholds δ_w and δ_h by calibrating a binary classifier with the outputs of the aggregated groups/pools used to identify random samplings of the LFW dataset, as well as their corresponding ground truth values. We train it with 8M negative samples (a live template with no hit in the DB) and 500k positive samples (a live template with a single hit in the DB). To benefit from convenient data alignment, and given the fact that both $n/$ (the number of slots) and l (the number of elements per template) are powers of 2, we test pools of the form $w, h \in 2^{\beta} \;\; \forall \beta \in \{1, \ldots, 11\}$. The biometric precision for a given pool size is applicable both vertically and horizontally. To estimate the combined error rate it suffices to combine the errors for selected w and h.

Parameter selection. Following the analysis from 3.2.3, we pick $n = 2^{15}$ to allow for a high enough number of multiplications. We set the modulus chain $q \approx 2^{60} * 2^{30*d} * 2^{60}$ with the maximum number of multiplications d required for the entire face identification algorithm, yielding smaller q and thus faster operations for lower circuit depth. We use templates with $l = 128$ coming out of the unmodified feature extractor.

[4]ArcFace-based[16] feature extractors with comparable latency and precision can be obtained from https://github.com/deepinsight/insightface/wiki/Model-Zoo

To select K we highlight that, in order to make the Grote-based face identification solution more performant than the naïve solution, we need a reference database size $K \geq n/2 + 1$ so that a naïve face identification algorithm requires at least $T = \lceil 2K/n \rceil \geq 2$ ciphertexts to hold all the score results. By employing a synthetic augmentation of the LFW dataset [5]we are able to set $T = 2$.

4.2 Results

We first analyze the impact of group testing in the biometric **precision** of the system by first analyzing the 1D layout case, setting $w = 1$ and playing with h or vice-versa. We run face identification experiments employing Algorithm 2 and record the FRR (probability of a registered user to not match with the database) for a fixed $FAR = 10^{-3}$ (probability of a non-registered user to match with the DB). Figure 4 displays our results, with the baseline without group testing represented in the *group size*= 1 intercept. As seen in this figure, and in line with our expectations, higher α yields a better max approximation, and with it lower errors. We observe that $\alpha \geq 16$ yields $FRR < 5\%$ for group sizes of up to 512, a small impact in the error that allows us to conclude that the Grote approach has a small impact in the system performance in that range.

We extend the biometric precision analysis to the full 2D layout with the same approach in Figure 5. The use of a 2D layout can be seen as a composition of two independent 1D layouts (one per axis) and thus the 2D layout accuracy is symmetric with respect to $w = h$. E.g., the error for $(w = 128, h = 512)$ amounts to the accumulated error in two independent 1D layouts with group sizes of 128 and 512, the same error that a $(w = 512, h = 128)$ yields. Once again, higher

[5]We generate 3 randomly perturbed templates per identity that are statistically close to the original reference templates of such identity, and ensure they follow the same distribution of matching probabilities from the original LFW. This bumps the number of identities from 5749 to $K = 22996$, yielding $T = 2$.

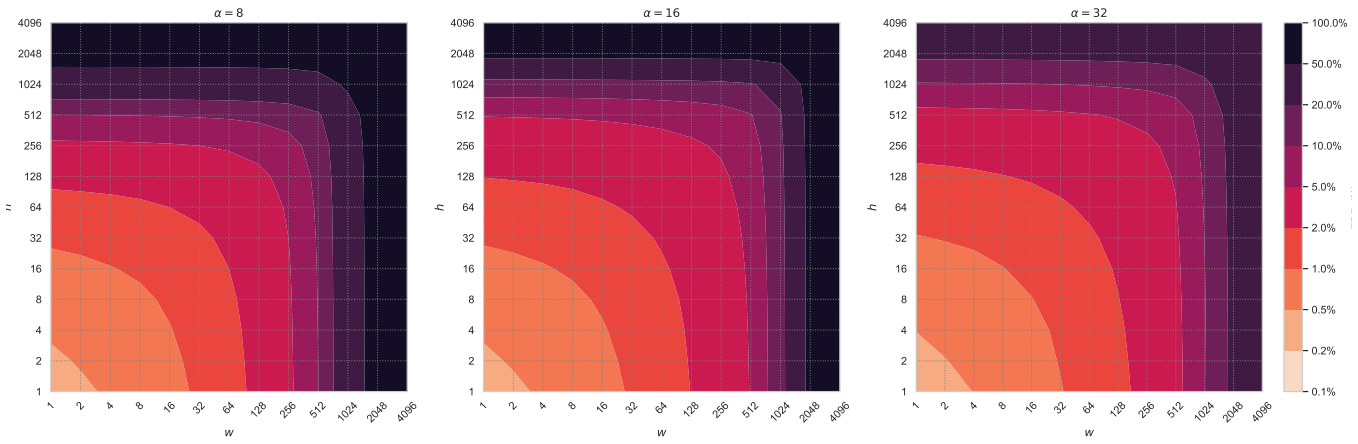

Figure 5: Face identification precision of GROTE based on 2D score matrix dimensions h and w, as well as the exponent α.

α values (center and right figure) yield lower errors than lower α (left figure) for the same group sizes. In this figure we can spot balanced configurations where $w \approx h$ that preserve the biometric accuracy of the system with $FRR < 5\%$ (e.g., $w = 128, h = 256$, allowing us to test at once $K = 32768$ identities), and this allows us to confirm that the GROTE preserves the biometric accuracy of the system.

To assess the **latency** gains of GROTE, we also record the time required to compute each operation in the encrypted domain, as well as the runtime of the entire system, comparing the runtime of the naïve solution with that of a GROTE-based solution. The results are shown in Table 1. We observe first-hand how the lower multiplication depth d requirements of GROTE allow for noticeably faster CKKS operations. This, linked to the drop in latency thanks to the reduction in the number of comparisons (from $T = 2$ to $\lceil 2(h+w)/n \rceil = 1$), yields a significant speedup in the entire system while yielding low error rates for selected $w = 128$, $h = 256$ and $\alpha = 16$ ($FRR \leq 5\%$ as per Figure 5, center). We also observe that the latency of the argmax & validation is more than a third of the latency of the entire system, thus a performant solution should sacrifice it at the expense of some loss of practical privacy (due to the increased input leakage). Overall, GROTE is able to reduce the latency of the face identification system by at least 33% (a factor of 1.5), a reduction that would only become more significant for $T > 2$, from K to $2\sqrt{K}$ elementwise comparisons.

5 PREVIOUS WORKS

The core idea of secure face biometrics has been extensively studied before with security guarantees stemming from various privacy-preserving techniques.

Sadeghi et. al. [54] combined homomorphic encryption with garbled circuits for a 2PC privacy-preserving face identification solution using *eigenfaces*[60]. SCiFI[46] employed additively homomorphic encryption and Oblivious Transfer to protect a semi-deterministic region-based face identification system. More recently, Osorio et. al. [47] employed a two-stage face identification consisting of a product quantization-based hashing stage to shortlist some candidates and a reduced homomorphic matching stage based on BFV. Face authentication/verification has also been extensively studied, yielding results fast enough to be used in practice based on the BFV scheme [4] and on other homomorphic encryption schemes[35]. Secure biometric identification has also been proposed for other types of biometric data such as iris recognition [34].

In other line of works, FHE has been widely studied as a technique for privacy-preserving biometrics, from the HE-based biometric access control system of [43], to the packing technique of [63], or [58] showing a clever encoding using packing to perform a biometric matching with one single homomorphic multiplication. [2] used Homomorphic Encryption for fingerprint biometrics, whereas [18] employed both CKKS and BFV for face identification, and [26] proposed the protection of a multi-biometric system. There are other previous works studying secure biometrics, covered in the MPC-based survey from [20] and a collection of FHE-based solutions surveyed in [52].

With the goal of providing security guarantees specifically tailored to biometrics, a wide range of works cover cancellable biometrics [32, 59, 61], where they apply user-specific geometric transformations to the template space instead of relying on cryptographic primitives.

Finally, the idea of group testing has touched the field of cryptography before, from pure combinatoric studies [14] to digital fingerprinting and key distribution patterns [57].

6 CONCLUSIONS

This paper proposed a new algorithm to perform privacy-preserving face identification based on the notion of group testing, and applied it to a solution using the Cheon-Kim-Kim-Song (CKKS) homomorphic encryption scheme. Securely computing the closest reference template to a given live template requires K comparisons, as many as there are identities in a biometric database. Our solution, named

Table 1: Latency for single-core execution of the listed algorithms. We set $K = n = 16384$ so that the naïve face identif. algorithm requires $T = \lceil 2K/n \rceil = 2$ comparisons, with template elements of $l = 128$ bits, group testing with dimensions $w = 256$ and $h = 128$ ($FRR \leq 5\%$ as per Figure 5, center) and exponent $\alpha = 16$, and $\texttt{depth} = 11$ for the comparison polynomial approximation.

Algorithm	Composed of	Grote Latency (ms)	Naïve Latency (ms)	Total #multiplications (d)
CKKS.encrypt	-	99	122	-
CKKS.add	-	1.6	2	-
CKKS.add_plain	-	0.7	0.9	-
CKKS.mult	-	12.4	23	1
CKKS.mult_plain	-	5.7	15.5	1
CKKS.rotate	-	290	438	-
CKKS.relinearize	-	288	443	-
CKKS.mod_switch		9	11	-
Matching ($l = 128$)	$(mult + relin + modswitch) + \log_2(l) * (rotate + add)$	2351	3557	1
Grote (α=16, w=128, h=256)	$\log_2(\alpha) * (mult + relin + modswitch) + \log_2(\max(w,h)) * (rotate + add)$	3570	-	4
optMinimaxComp ($\texttt{depth} = 11$)	$\texttt{depth} * (mult + relin + modswitch + add\_plain)$	3433	5202	11
ArgMax	$(mult\_plain + relin + modswitch) + \log_2(\max(w,h)) * (rotate + add)$	2636	3990	1
Validation	$\log_2(\max(w,h)) * (rotate + add) + (mult + relin + modswitch)$	2642	3997	1
Face Identif. (no Argmax)	matching + grote$_?$ + optMinimaxComp$*T_?$	9354	13961	naïve: 23; Grote: 16
Face Identif. (Argmax & valid.)	Face Identif. + argmax + valid	14632	21948	naïve: 25; Grote: 18

Grote, replaces element-wise testing for the K elements of a database by group testing to notably reduce the number of non-linear operations in the encrypted domain from K to up to $2\sqrt{K}$. More specifically, we approximate the max of the coordinates of a large vector by its α-norm (raising to the α-th power and cumulative sum) in a 2D layout, incurring a small impact in the accuracy of the system while greatly speeding up its execution. We implemented Grote and showed it to be at least 30% more performant than its naïve equivalent for sufficiently large databases $K > 8192$.

For future works, we will study the performance of Grote employing alternative feature extractors to vary both the elementwise template size and the number of elements per template. We will also extend Grote to larger datasets to evaluate its scalability.

7 ACKNOWLEDGEMENTS

This work has been partially supported by the 3IA Côte d'Azur program (ANR19-P3IA-0002).

REFERENCES

[1] Multiple authors. 2018. *Homomorphic Encryption Security Standard.* Technical Report. HomomorphicEncryption.org, Toronto, Canada.

[2] Mauro Barni, Tiziano Bianchi, Dario Catalano, Mario Di Raimondo, Ruggero Donida Labati, Pierluigi Failla, Dario Fiore, Riccardo Lazzeretti, Vincenzo Piuri, Alessandro Piva, et al. 2010. A privacy-compliant fingerprint recognition system based on homomorphic encryption and fingercode templates. In *2010 Fourth IEEE International Conference on Biometrics: Theory, Applications and Systems (BTAS)*. IEEE, 1–7.

[3] Marcelo Blatt, Alexander Gusev, Yuriy Polyakov, and Shafi Goldwasser. 2020. Secure large-scale genome-wide association studies using homomorphic encryption. *Proceedings of the National Academy of Sciences of the United States of America* 117, 21 (26 May 2020), 11608–11613. https://doi.org/10.1073/pnas.1918257117

[4] Vishnu Naresh Boddeti. 2018. Secure face matching using fully homomorphic encryption. In *2018 IEEE 9th International Conference on Biometrics Theory, Applications and Systems (BTAS)*. IEEE, 1–10.

[5] Zvika Brakerski. 2012. Fully Homomorphic Encryption without Modulus Switching from Classical GapSVP. In *Advances in Cryptology – CRYPTO 2012*. Springer Berlin Heidelberg, 868–886. https://doi.org/10.1007/978-3-642-32009-5_50

[6] Zvika Brakerski, Craig Gentry, and Vinod Vaikuntanathan. 2012. (Leveled) fully homomorphic encryption without bootstrapping. In *Proceedings of the 3rd Innovations in Theoretical Computer Science* (Cambridge, Massachusetts) *(ITCS '12)*. ACM, New York, NY, USA, 309–325. https://doi.org/10.1145/2090236.2090262

[7] Centers for Medicare & Medicaid. 1996. The Health Insurance Portability and Accountability Act of 1996 (HIPAA). Online at http://www.cms.hhs.gov/hipaa/.

[8] Hao Chen, Kim Laine, and Peter Rindal. 2017. Fast Private Set Intersection from Homomorphic Encryption. In *Proceedings of the 2017 ACM SIGSAC Conference on Computer and Communications Security* (Dallas, Texas, USA) *(CCS '17)*. Association for Computing Machinery, New York, NY, USA, 1243–1255. https://doi.org/10.1145/3133956.3134061

[9] Jung Hee Cheon, Andrey Kim, Miran Kim, and Yongsoo Song. 2017. Homomorphic Encryption for Arithmetic of Approximate Numbers. In *Advances in Cryptology – ASIACRYPT 2017*. Springer International Publishing, 409–437. https://doi.org/10.1007/978-3-319-70694-8_15

[10] Jung Hee Cheon, Dongwoo Kim, and Duhyeong Kim. 2020. Efficient homomorphic comparison methods with optimal complexity. In *International Conference on the Theory and Application of Cryptology and Information Security*. Springer, 221–256.

[11] Jung Hee Cheon, Dongwoo Kim, Duhyeong Kim, Hun Hee Lee, and Keewoo Lee. 2019. Numerical method for comparison on homomorphically encrypted numbers. In *International Conference on the Theory and Application of Cryptology and Information Security*. Springer, 415–445.

[12] Ilaria Chillotti, Nicolas Gama, Mariya Georgieva, and Malika Izabachène. 2020. TFHE: Fast Fully Homomorphic Encryption Over the Torus. *Journal of Cryptology. The Journal of the International Association for Cryptologic Research* 33, 1 (1 Jan. 2020), 34–91. https://doi.org/10.1007/s00145-019-09319-x

[13] Ilaria Chillotti, Marc Joye, and Pascal Paillier. 2021. Programmable Bootstrapping Enables Efficient Homomorphic Inference of Deep Neural Networks. Cryptology ePrint Archive, Report 2021/091. https://eprint.iacr.org/2021/091

[14] Charles J Colbourn. 1999. Group testing for consecutive positives. *Annals of Combinatorics* 3, 1 (1999), 37–41.

[15] European Commission. [n. d.]. 2018 reform of EU data protection rules. https://gdpr-info.eu/

[16] Jiankang Deng, Jia Guo, Xue Niannan, and Stefanos Zafeiriou. 2019. ArcFace: Additive Angular Margin Loss for Deep Face Recognition. In *CVPR*.

[17] Jiankang Deng, Jia Guo, Zhou Yuxiang, Jinke Yu, Irene Kotsia, and Stefanos Zafeiriou. 2019. RetinaFace: Single-stage Dense Face Localisation in the Wild. In *arxiv*.

[18] Pawel Drozdowski, Nicolas Buchmann, Christian Rathgeb, Marian Margraf, and Christoph Busch. 2019. On the application of homomorphic encryption to face identification. In *2019 International Conference of the Biometrics Special Interest Group (BIOSIG)*. IEEE, 1–5.

[19] Dingzhu Du, Frank K Hwang, and Frank Hwang. 2000. *Combinatorial group testing and its applications*. Vol. 12. World Scientific.

[20] Diana-Elena Fălămaș, Kinga Marton, and Alin Suciu. 2021. Assessment of Two Privacy Preserving Authentication Methods Using Secure Multiparty Computation Based on Secret Sharing. *Symmetry* 13, 5 (2021), 894.

[21] J Fan and F Vercauteren. 2012. Somewhat Practical Fully Homomorphic Encryption. *IACR Cryptology ePrint Archive* (2012). https://eprint.iacr.org/2012/144

[22] Craig Gentry et al. 2009. *A fully homomorphic encryption scheme*. Vol. 20. Stanford.

[23] Ran Gilad-Bachrach, Nathan Dowlin, Kim Laine, Kristin Lauter, Michael Naehrig, and John Wernsing. 2016. Cryptonets: Applying neural networks to encrypted data with high throughput and accuracy. In *International Conference on Machine Learning*. 201–210.

[24] Bart Goethals, Sven Laur, Helger Lipmaa, and Taneli Mielikäinen. 2004. On private scalar product computation for privacy-preserving data mining. In *International Conference on Information Security and Cryptology*. Springer, 104–120.

[25] Christian Gollier and Olivier Gossner. 2020. *Group testing against Covid-19*. Technical Report. EconPol Policy Brief.

[26] Marta Gomez-Barrero, Emanuele Maiorana, Javier Galbally, Patrizio Campisi, and Julian Fierrez. 2017. Multi-biometric template protection based on homomorphic encryption. *Pattern Recognition* 67 (2017), 149–163.

[27] Gary B. Huang, Manu Ramesh, Tamara Berg, and Erik Learned-Miller. 2007. *Labeled Faces in the Wild: A Database for Studying Face Recognition in Unconstrained Environments.* Technical Report 07-49. University of Massachusetts, Amherst.

[28] Alberto Ibarrondo, Hervé Chabanne, and Melek Önen. 2021. Practical Privacy-Preserving Face Identification based on Function-Hiding Functional Encryption. In *International Conference on Cryptology and Network Security*. Springer, 63–71.

[29] Alberto Ibarrondo and Alexander Viand. 2021. Pyfhel: Python for homomorphic encryption libraries. In *Proceedings of the 9th on Workshop on Encrypted Computing & Applied Homomorphic Cryptography*. 11–16.

[30] Ilia Iliashenko and Vincent Zucca. 2021. Faster homomorphic comparison operations for BGV and BFV. *Proceedings on Privacy Enhancing Technologies* 2021, 3 (2021), 246–264.

[31] Anil K Jain, Patrick Flynn, and Arun A Ross. 2007. *Handbook of biometrics*. Springer Science & Business Media.

[32] Harkeerat Kaur and Pritee Khanna. 2020. Privacy preserving remote multi-server biometric authentication using cancelable biometrics and secret sharing. *Future Generation Computer Systems* 102 (2020), 30–41.

[33] Andrey Kim, Antonis Papadimitriou, and Yuriy Polyakov. 2022. Approximate homomorphic encryption with reduced approximation error. In *Cryptographers' Track at the RSA Conference*. Springer, 120–144.

[34] Jascha Kolberg, Pia Bauspieß, Marta Gomez-Barrero, Christian Rathgeb, Markus Dürmuth, and Christoph Busch. 2019. Template protection based on homomorphic encryption: Computationally efficient application to iris-biometric verification and identification. In *2019 IEEE International Workshop on Information Forensics and Security (WIFS)*. IEEE, 1–6.

[35] Jascha Kolberg, Pawel Drozdowski, Marta Gomez-Barrero, Christian Rathgeb, and Christoph Busch. 2020. Efficiency analysis of post-quantum-secure face template protection schemes based on homomorphic encryption. In *2020 International Conference of the Biometrics Special Interest Group (BIOSIG)*. IEEE, 1–4.

[36] Eunsang Lee, Joon-Woo Lee, Young-Sik Kim, and Jong-Seon No. 2021. Minimax approximation of sign function by composite polynomial for homomorphic comparison. *IEEE Transactions on Dependable and Secure Computing* (2021).

[37] Eunsang Lee, Joon-Woo Lee, Young-Sik Kim, and Jong-Seon No. 2022. Optimization of homomorphic comparison algorithm on rns-ckks scheme. *IEEE Access* 10 (2022), 26163–26176.

[38] Joon-Woo Lee, HyungChul Kang, Yongwoo Lee, Woosuk Choi, Jieun Eom, Maxim Deryabin, Eunsang Lee, Junghyun Lee, Donghoon Yoo, Young-Sik Kim, et al. 2022. Privacy-preserving machine learning with fully homomorphic encryption for deep neural network. *IEEE Access* 10 (2022), 30039–30054.

[39] Joon-Woo Lee, Eunsang Lee, Yongwoo Lee, Young-Sik Kim, and Jong-Seon No. 2021. High-precision bootstrapping of RNS-CKKS homomorphic encryption using optimal minimax polynomial approximation and inverse sine function. In *Annual International Conference on the Theory and Applications of Cryptographic Techniques*. Springer, 618–647.

[40] Damien Ligier, Sergiu Carpov, Caroline Fontaine, and Renaud Sirdey. 2017. Information leakage analysis of inner-product functional encryption based data classification. In *2017 15th Annual Conference on Privacy, Security and Trust (PST)*. IEEE, 303–3035.

[41] Chun Lo, Mingyan Liu, Jerome P Lynch, and Anna C Gilbert. 2013. Efficient sensor fault detection using combinatorial group testing. In *2013 IEEE international conference on distributed computing in sensor systems*. IEEE, 199–206.

[42] Adriana López-Alt, Eran Tromer, and Vinod Vaikuntanathan. 2012. On-the-fly multiparty computation on the cloud via multikey fully homomorphic encryption. In *Proceedings of the forty-fourth annual ACM symposium on Theory of computing*. 1219–1234.

[43] Ying Luo, S Cheung Sen-ching, and Shuiming Ye. 2009. Anonymous biometric access control based on homomorphic encryption. In *2009 IEEE International Conference on Multimedia and Expo*. IEEE, 1046–1049.

[44] Vadim Lyubashevsky, Chris Peikert, and Oded Regev. 2010. On ideal lattices and learning with errors over rings. In *Annual international conference on the theory and applications of cryptographic techniques*. Springer, 1–23.

[45] Christian Mouchet, Juan Troncoso-Pastoriza, Jean-Philippe Bossuat, and Jean-Pierre Hubaux. 2020. Multiparty homomorphic encryption from ring-learning-with-errors. *Cryptology ePrint Archive* (2020).

[46] Margarita Osadchy, Benny Pinkas, Ayman Jarrous, and Boaz Moskovich. 2010. Scifi-a system for secure face identification. In *2010 IEEE Symposium on Security and Privacy*. IEEE, 239–254.

[47] Dailé Osorio-Roig, Christian Rathgeb, Pawel Drozdowski, and Christoph Busch. 2021. Stable hash generation for efficient privacy-preserving face identification. *IEEE Transactions on Biometrics, Behavior, and Identity Science* (2021).

[48] Pascal Paillier. 1999. Public-Key Cryptosystems Based on Composite Degree Residuosity Classes. In *Advances in Cryptology — EUROCRYPT '99*. Springer Berlin Heidelberg, 223–238. https://doi.org/10.1007/3-540-48910-X_16

[49] Oded Regev. 2005. On Lattices, Learning with Errors, Random Linear Codes, and Cryptography. In *In STOC*. ACM Press, 84–93.

[50] Ronald L Rivest, Len Adleman, and Michael L Dertouzos. 1978. On Data Banks and Privacy Homomorphisms. *Foundations of secure computation* 4, 11 (1978), 169–180. https://people.csail.mit.edu/rivest/RivestAdlemanDertouzos-OnDataBanksAndPrivacyHomomorphisms.pdf

[51] R L Rivest, A Shamir, and L Adleman. 1978. A method for obtaining digital signatures and public-key cryptosystems. *Commun. ACM* 21, 2 (1 Feb. 1978), 120–126. https://doi.org/10.1145/359340.359342

[52] Zhang Rui and Zheng Yan. 2018. A survey on biometric authentication: Toward secure and privacy-preserving identification. *IEEE access* 7 (2018), 5994–6009.

[53] T Sabhanayagam, V Prasanna Venkatesan, and K Senthamaraikannan. 2018. A comprehensive survey on various biometric systems. *International Journal of Applied Engineering Research* 13, 5 (2018), 2276–2297.

[54] Ahmad-Reza Sadeghi, Thomas Schneider, and Immo Wehrenberg. 2009. Efficient privacy-preserving face recognition. In *International conference on information security and cryptology*. Springer, 229–244.

[55] SEAL 2021. Microsoft SEAL (release 3.7). https://github.com/Microsoft/SEAL. Microsoft Research, Redmond, WA..

[56] Adi Shamir. 1979. How to share a secret. *Comm. of the ACM* 22, 11 (1979), 612–613.

[57] Douglas R Stinson, Tran Van Trung, and Ruizhong Wei. 2000. Secure frameproof codes, key distribution patterns, group testing algorithms and related structures. *Journal of Statistical Planning and Inference* 86, 2 (2000), 595–617.

[58] Hiroto Tamiya, Toshiyuki Isshiki, Kengo Mori, Satoshi Obana, and Tetsushi Ohki. 2021. Improved Post-quantum-secure Face Template Protection System Based on Packed Homomorphic Encryption. In *2021 International Conference of the Biometrics Special Interest Group (BIOSIG)*. IEEE, 1–5.

[59] Andrew Beng Jin Teoh and Chong Tze Yuang. 2007. Cancelable biometrics realization with multispace random projections. *IEEE Transactions on Systems, Man, and Cybernetics, Part B (Cybernetics)* 37, 5 (2007), 1096–1106.

[60] Matthew A Turk and Alex P Pentland. 1991. Face recognition using eigenfaces. In *Proceedings. 1991 IEEE computer society conference on computer vision and pattern recognition*. IEEE Computer Society, 586–587.

[61] Wencheng Yang, Song Wang, Muhammad Shahzad, and Wei Zhou. 2021. A cancelable biometric authentication system based on feature-adaptive random projection. *Journal of Information Security and Applications* 58 (2021), 102704.

[62] Andrew Chi-Chih Yao. 1986. How to generate and exchange secrets. In *27th Annual Symposium on Foundations of Computer Science (sfcs 1986)*. IEEE, 162–167.

[63] Masaya Yasuda, Takeshi Shimoyama, Jun Kogure, Kazuhiro Yokoyama, and Takeshi Koshiba. 2013. Packed homomorphic encryption based on ideal lattices and its application to biometrics. In *International Conference on Availability, Reliability, and Security*. Springer, 55–74.