Research Article

Partially homomorphic framework for secure privacy-preserving ID creation

Integrated Computer-Aided Engineering
2025, Vol. 32(4) 379–396

DOI: 10.1177/10692509251342680
journals.sagepub.com/home/ico

Nikola Hristov-Kalamov[1], Raúl Fernández-Ruiz[1], Cristina Conde[1], Agustín Álvarez-Marquina[2], Francisco Domínguez-Mateos[1], Pedro Gómez-Vilda[1,2] and Daniel Palacios-Alonso[1,2]

Abstract

Homomorphic encryption has seen limited application in real-world settings due to its high computational costs, which often impede practical use in latency-sensitive scenarios. Addressing this challenge, this research work presents an efficient, partially homomorphic encryption framework for privacy-preserving ID creation tailored to biometric applications, such as border control or access to secure restricted facilities. The proposed solution leverages the additive homomorphic properties of Paillier and Elliptic Curve ElGamal encryption schemes to encapsulate biometric data within a secure cryptographic layer, enabling rapid verification while minimizing computation and storage demands. Paillier encryption ensures robust security with maximal accuracy, while Elliptic Curve ElGamal optimizes for minimal ciphertext sizes, both meeting rigorous ISO biometric security standards. Experimental results demonstrate that the framework achieves high accuracy with reduced memory and bandwidth requirements, with encrypted IDs as compact as 4 KiB, making it suitable for scalable deployment. This research work represents a key advancement in homomorphic encryption applications, balancing privacy and efficiency without the usual overhead, and making it feasible for real-time biometric processing. In summary, this framework offers a pioneering solution in secure biometric verification, setting a new standard for privacy-preserving applications, positioning it as a promising model for future secure identification systems.

Keywords

Homomorphic encryption, biometric comparison, Paillier, elliptic curves, ElGamal

Received: 1 November 2024; accepted: 25 April 2025

1 Introduction

Biometric systems are increasingly being used for authentication, verification, and identification. These systems rely on unique biological characteristics to verify an individual's identity. The uniqueness, convenience, and permanence of biometric traits have driven this increased adoption in personal verification systems. Among the most widely used biometric attributes are speech,[1–3] facial features,[4–7] fingerprints,[8,9] palm-prints,[10–12] iris patterns,[13,14] body detection[15–17] and other bio-signal sampling.[18] The fusion of multiple biometric traits has proven to be an effective strategy for enhancing the accuracy and robustness of identification systems, significantly surpassing the reliability of token-based methods. Biometric data is utilized in a wide range of scenarios, including single-identity verification, device unlocking, access control systems, and biometric database searches.

However, the storage and transmission of biometric data raise significant privacy and security concerns. Biometric data, if compromised, can be exploited for identity theft or other malicious purposes such as the disclosure of confidential medical information in healthcare settings, and fraudulent access to bank accounts in payment systems. Therefore, biometric information is classified as sensitive personal data under the European Union General Data Protection Regulation (GDPR)[19] and requires robust security measures to prevent unlawful intrusion. In this regard, the encryption of biometric data has become an active field in

[1]Escuela Técnica Superior de Ingeniería Informática, Universidad Rey Juan Carlos, Madrid, Spain
[2]Center for Biomedical Technology, Universidad Politécnica de Madrid, Madrid, Spain

Corresponding author:
Daniel Palacios-Alonso, Universidad Rey Juan Carlos, Móstoles Campus, Madrid, Spain.
Email: daniel.palacios@urjc.es

recent years due to the increasing number of works and applications.[20,21]

In most biometric applications, it is necessary to perform a comparison between different samples. These samples are typically extracted using Convolutional Neural Networks (CNNs) and represented as embeddings. The comparison between these embeddings relies on measuring the distance between them. However, this representation has an inherent vulnerability: invertibility. That is, an adversary could use deep learning algorithms to reverse the embedding extraction process and obtain an approximation of the original image, thereby compromising data confidentiality.[22–24] Moreover, the ISO/IEC IS 24745:2022[25] imposes the following properties for the security and privacy of biometric embeddings:

- Renewability: the ability to revoke obsolete templates and generate new ones from the same biometric source.
- Unlinkability: the prevention of linking two or more encrypted embeddings to the same source.
- Irreversibility: the use of processes that make the embedding irreversible to its original state.
- Recognition Performance Preservation: ensuring that the system's performance is not significantly affected by the encryption.

To address the aforementioned issue, homomorphic encryption (HE) is proposed as a solution.[26] This encryption scheme allows an untrusted party to perform computations on encrypted data without compromising its confidentiality. A significant advantage of this scheme lies in its ability to ensure the continuous protection of a user's personal data. Even after the contractual or service relationship with the untrusted party has ended, it ensures that shared information cannot be misused. However, HE presents inherent limitations, such as the maximum number of operations that can be performed without data loss and the high computational cost associated with these operations. There are three variants of HE: Partially Homomorphic Encryption (PHE), which allows an unlimited number of a single type of operation (either additions or multiplications); Somewhat Homomorphic Encryption (SHE), which enables a limited number of both additions and multiplications; and Fully Homomorphic Encryption (FHE), which supports an unlimited number of additions and multiplications.[27]

The authors' work aims to leverage HE algorithms to provide a framework for creating an **encrypted ID** that can be used in secure privacy-preserving biometric verifications. A novel approach is presented that combines the biometric features extracted from convolutional neural networks with state-of-the-art cryptographic algorithms. Specifically, PHE is leveraged to drastically improve execution times and minimize ciphertext sizes. This research work is centered primarily on citizens' IDs, like passports, utilized in border customs checking. However, the setup can be extrapolated to any system where any biometric information is needed for access, such as fingerprint payment systems, biometric access areas or border control points. The latter are places where a person's voice, face, fingerprints, palm-prints or retina scans are utilized to open a door to a secure facility. Furthermore, there are use cases in law enforcement, where a suspect's image can be matched against an encrypted ID (or database of IDs) without accessing or revealing the underlying biometric data.

1.1 Related work

There is extensive research on the risks associated with handling private or sensitive information, not only in the field of biometrics[28,29] but also in related areas such as biomedical data.[30,31] Commonly proposed solutions include strict access control, data anonymization, secure data sharing frameworks, and the application of standard cryptographic techniques. However, a great amount of publications does not sufficiently explore more advanced privacy-preserving methods, such as HE, which can limit effective risk mitigation and data protection. This study seeks to close this gap by exploring how HE can enhance privacy in the field of biometrics.

When exploring cryptographic HE centered solutions, most of the literature focuses on FHE schemes due to their theoretical capability to support an unlimited number of operations. This flexibility has driven extensive research aimed at addressing its primary limitations, such as high computational time and intensive resource consumption. Efforts to enhance FHE are pursued both by optimizing the underlying mathematical functions and by accelerating performance through the use of GPU units.[27] The increasing adoption of cloud services has further fueled interest in this area. One example of an application is the storage of multiple biometric embeddings in the cloud, where one-to-many (1 vs. n) comparisons must be performed with new embeddings submitted by users. To ensure the confidentiality of biometric data throughout the entire process, it is essential to employ HE schemes.[20,21]

Focusing on PHE schemes, in Shafagh et al.,[32] the authors propose Pilatus, a platform designed to facilitate the secure exchange of data in the cloud using a PHE scheme. This proposal aims to ensure that the data remains agnostic to the server. For the scheme's implementation, the Elliptic Curve ElGamal (ECEG) method is combined with the Chinese Remainder Theorem (CRT), enhancing computational efficiency while maintaining low resource consumption. Mobile applications integrated with Pilatus achieve a decryption overhead of less than one second, ensuring that the encryption process does not negatively impact user experience.

In Liu et al.,[33] a distributed learning model is implemented using PHE, where the weights of various neural networks trained in trust zones are shared to create a combined model in the cloud. The decision not to establish a federated server within a secure zone stems from the high resource consumption and prolonged time required to set up such private services. Consequently, for the aggregation of hyperparameters on an untrusted server, the ECEG PHE scheme is employed, as a FHE approach would render the service impractical in real-world applications due to the significant computational overhead. The study also explores a wide range of additional methods to achieve further optimizations, resulting in processes that are far more efficient than simply encrypting the federated database.

Gómez-Barrero et al. present a novel system for fusing encrypted biometric embeddings using the Paillier homomorphic encryption scheme.[34] The authors investigate various fusion strategies, with a particular focus on feature-level fusion. This approach involves combining the biometric information of individual features into a single template before encryption. A key finding of the study is that computational costs are suitable for real-time applications, with an estimated comparison time per verification as low as 0.5 milliseconds using 4 processor cores.

In Gomez-Barrero et al.[35] a privacy-preserving method for comparing user hand signatures using PHE is presented, specifically utilizing the Paillier cryptosystem. As stated by the authors, PHE is chosen for its reduced computation overhead. This research work achieves very short identification times by mixing a fixed-length comparison approach of pre-extracted general hand signature attributes with a sub-sampling of a variable-length Dynamic Time Warping (DTW) algorithm. Specifically, for a known database of 400 user hand signatures, an identification takes 2.4 seconds, using 4 processor cores, greatly outperforming similar works while almost identical precision and retaining user data security.

Mahesh Kumar et al. describe a novel iris-based biometric verification system, prioritizing user privacy.[36] For this purpose, images of both irises (left and right) were acquired to generate a more robust biometric template. Subsequently, a dimensionality reduction technique based on a modification of the local random projection algorithm was applied. The resulting embeddings were encrypted using the Paillier homomorphic encryption scheme, ensuring the confidentiality of biometric data throughout the entire verification process.

Analyzing some prominent FHE implementations, Bauspieß et al.[37] is of note. The authors propose a novel coefficient packing scheme to compute multiple distances in biometric templates, optimizing the process so that the computational cost is equivalent to that of a single comparison. With this approach, they achieve a 1.6% performance improvement compared to conventional FHE systems. In the first stage, a principal component analysis (PCA) is applied to reduce the dimensionality of the embeddings from 512 to 64 dimensions. This reduction allows for a more compact representation of the data without losing relevant information. Next, k biometric templates are combined into a single plaintext before encryption. This innovative technique enables multiple comparisons to be performed simultaneously, maintaining the computational cost associated with encrypting a single ciphertext, resulting in a quadratic reduction in time complexity. It is also worth noting that shifting operations on ciphertexts are computationally expensive, and the dimensionality reduction significantly mitigates this impact.

Kolberg et al. propose a system for verification and identification based on iris codes that leverages homomorphisms in the N-th degree truncated polynomial ring units (NTRU) framework.[38] The authors achieve an improvement both in the accuracy rate and in the algorithm performance. Before applying the homomorphism techniques, the bit blocks of the iris code are reordered, placing those that contain more relevant information at the beginning of the sequence. To compare the embeddings, a block-by-block comparison process is carried out. If the distance between two blocks is below a predefined threshold, it is considered that both embeddings come from the same source due to their high similarity, and the comparison terminates. On the other hand, if the distance is greater than a second threshold, it is concluded that the embeddings correspond to different sources. If the distance falls between the two thresholds, the comparison of the remaining blocks continues.

The current research work achieves similar computation times to other PHE implementations, specifically the verifications in Gomez-Barrero et al.[34,35] and Morampudi et al.[36] These works make use of additional algorithm optimizations as well as a certain degree of parallelization. The distinguishing factor in this work is the small ciphertext sizes. With the right parameter choice, these require 256 bits per embedding value (before compression), resulting in an 8 times reduction compared to Paillier ciphertexts. Moreover, analyzing FHE algorithms, these are far more suitable for 1-n identification due to batching techniques. However, in the 1-1 case, our PHE implementation achieves promising results compared to FHE alternatives and leaves room for more aggressive optimizations, as discussed in the 6 section.

Taking the previous related work into consideration as well as a general overview of the field, some objectives are established for the encrypted homomorphic ID. These are founded on the Privacy by Design framework[39,40] and based on the aforementioned ISO/IEC IS 24745:2022 standard.[25] Firstly, the proposed solution must offer **full and efficient functionality**, meaning the system should perform its intended tasks at a level comparable to conventional solutions, without introducing excessive computational complexity. Secondly, the scheme must exhibit **static functionality**, allowing it to work without requiring any

computation from the user. This design ensures that conventional ID formats, such as paper-based documents or passports, can still be utilized, while remaining compatible with computational devices, such as smartphones, which are also capable of storing data. Additionally, the system must ensure **irrecoverable information**, meaning that any sensitive or biometric data embedded within a document or device should only be recoverable by official entities, such as the user's government or the issuing company. This provides two layers of protection. On the one hand, if the ID is stolen or lost, unauthorized individuals cannot access the biometric data. On the other hand, entities performing biometric verification, such as border control, have limited access to the sensitive data while still being able to complete the verification process. **Oblivious verification** is another crucial requirement, wherein all parties involved in the authentication process, including the user, the verifying entity, and the issuing entity (if necessary), should remain unaware of the specific underlying embedded data to the greatest extent possible. Finally, biometric data must never be stored in centralized databases, even in encrypted form, as such storage presents a clear target for hackers. This necessitates the **distributed storage** of biometric information to enhance security and minimize vulnerabilities.

Last, this paper is organized as follows: 2 describes an overview of the conceptual foundation needed for this solution. 3 summarizes the algorithm architecture and its building blocks as well as the vulnerability assessment under the semi-honest and malicious settings. 5 states key experimental outcomes obtained from profiling the two different implementations. 6 examines said outcomes, presents use cases for the proposed scheme, compares the results with other methods and addresses advantages and limitations. Finally, 7 presents some closing thoughts and future lines of research.

2 Theoretical framework

2.1 Biometric embedding comparison

Biometric information comparison is the process of measuring how similar or different pieces of data (such as images of faces, fingerprints, voice recordings, among others) are based on their content. A common and popular technique for this is to use neural networks, especially CNNs, to capture patterns and features from a piece of data. These networks convert the inputs into vectors of multiple elements called embeddings. Each embedding reflects the distinctive attributes of its corresponding input. By comparing these embeddings in a high-dimensional space, similarities can be accurately calculated.[5,41,42]

2.2 Homomorphic encryption

When describing Homomorphic Encryption (HE), a distinction is made between Fully-Homomorphic Encryption (FHE), Somewhat-Homomorphic Encryption (SHE) and Partially-Homomorphic Encryption (PHE). On the one hand, FHE, conceptualized first in Rivest and Dertouzos,[26] allows arbitrary computation on encrypted data that yields the same encrypted results as if it were performed on plaintexts. Typically, this unlimited computation is reduced to addition and multiplication. For messages m_A, m_B and encryption function $E()$:

$$E(m_A) + E(m_B) = E(m_A + m_B) \tag{1}$$

$$E(m_A) \cdot E(m_B) = E(m_A \cdot m_B) \tag{2}$$

A similar functionality is offered by SHE but with a limited number of operations before data degradation occurs. SHE implementations, although more limiting, remain more efficient in practical scenarios. Most FHE implementations rely on some form of SHE as their base coupled with a bootstrapping technique. Some of the most relevant are explained in: Gentry[43]; van Dijk et al.[44]; Fan and Vercauteren[45]; Brakerski et al.[46]; Brakerski and Vaikuntanathan.[47,48] For a detailed guide to FHE, see[49] and for modern surveys on the field, see Yousuf et al.[50] and Marcolla et al.[51]

On the other hand, when using PHE only certain specific operations are possible (usually addition or multiplication). Therefore, this approach is more limiting but can be orders of magnitude faster to compute. Some of the classical partially homomorphic algorithms are outlined here: Rivest et al.[52]; Elgamal[53] and Paillier.[54] For a modern survey of current implementations, see Ryu et al.[55]

2.3 Paillier encryption

Paillier[54] is a well-known probabilistic public-key partially homomorphic cryptosystem. It works as follows:

- Key generation. Two large primes q and p of approximately equal length are chosen. The value $n = p \cdot q$ is computed along with its totient $\phi(n) = (p-1)\cdot(q-1)$. Furthermore, the value $\delta = lcm(p-1, q-1)$ is obtained. Additionally, a random integer $g \in \mathbb{Z}^*_{n^2}$ is chosen (a simplified approach sets $g = n+1$). Last, the value μ is calculated with the following formula:

$$\mu = \left(\frac{(g^{\delta} \mod n^2) - 1}{n} \right)^{-1} \mod n \tag{3}$$

  where the fraction indicates the quotient between numerator and denominator. If the modular multiplicative inverse does not exist, g is invalid. The public key is $p_k = (n, g)$ and the secret key is $s_k = (\delta, \mu)$.

- Encryption. A message $m \in \mathbb{Z}_n$ is encrypted by first choosing a random number $r \in \mathbb{Z}_n^*$. This provides the probabilistic nature to the algorithm. The ciphertext is then obtained:

$$c = g^m \cdot r^n \mod n^2 \tag{4}$$

- Decryption. The original message m can be retrieved like so:

$$m = \left(\frac{(c^\delta \mod n^2) - 1}{n} \right) \cdot \mu \mod n \tag{5}$$

where the fraction represents the quotient between numerator and denominator.

The Paillier encryption system is additively homomorphic modulo n. For messages m_A and m_b, their ciphertexts can be combined in a way that reflects the addition of the plaintexts:

$$\begin{aligned} & E(m_A) \cdot E(m_B) \mod n^2 \\ & = c_{A1} \cdot c_{B1} \mod n^2 \\ & = g^{m_A} \cdot {r_A}^n \cdot g^{m_B} \cdot {r_B}^n \mod n^2 \\ & = g^{m_A + m_B} \cdot (r_A \cdot r_B)^n \mod n^2 \\ & = E(m_A + m_B) \end{aligned} \tag{6}$$

The product of two ciphertexts will correspond to the sum of the original messages once decrypted. It should be noted that, even if $m_A + m_B \geq n$, since the decryption operation is done modulo n (as denoted in Equation 5) the result will be bound to the modular structure of the system.

The security of the Paillier encryption scheme is founded on the computational hardness of the Decisional Composite Residuosity (DCR) problem.[54] The assumption posits that, given a composite number $n = p \cdot q$ (product of two primes p and q) and a number $z \in \mathbb{Z}_{n^2}$, it is hard to decide whether z is an n-residue modulo n^2, i.e., whether there exists an $x \in \mathbb{Z}_{n^2}$ such that $z = x^n \mod n^2$.

Paillier provides indistinguishability under chosen plaintext attack (IND-CPA) security. To prove this, an adversary is given two messages, m_0 and m_1, and asks for the encryption of one of them. For random bit $b \in \{0, 1\}$, the encryption is:

$$c = g^{m_b} \cdot r^n \mod n^2 \tag{7}$$

The adversary has to guess whether c is the cipher of m_0 or m_1. As a first step, the adversary tries to remove g^{m_b} by performing $c \cdot g^{-m_0} \mod n^2$ and $c \cdot g^{-m_1} \mod n^2$. One of the obtained values is an n-residue modulo n^2, that is to say, one of them is the result of $r^n \mod n^2$; however, due to the DCR assumption, it is computationally unfeasible to check this property. Therefore, the ciphertext gives no information about the underlying message and the adversary cannot distinguish between the encryption of m_0 and m_1 with a probability better than random guessing.

2.4 Elliptic curve cryptography

Elliptic Curve Cryptography (ECC)[56–58] is a framework for constructing public key cryptosystems that exploits the mathematical properties of elliptic curves over finite fields. An elliptic curve E defined over a finite field F_p (where p is a prime) is described by the following equation:

$$E : y^2 = x^3 + a \cdot x + b \mod p \tag{8}$$

where $a, b \in F_p$, and the discriminant $\Delta = 4 \cdot a^3 + 27 \cdot b^2 \neq 0$ ensures the curve is non-singular.

The set of points $P = (x, y)$ that satisfy this equation, together with a special point called the "point at infinity" $\mathcal{O}$, forms an abelian group under a defined addition operation. The group law on elliptic curves consists of:

1. Point Addition. Given two points $P = (x_1, y_1)$ and $Q = (x_2, y_2)$, the sum $P + Q = (x_3, y_3)$ is computed as such:

$$\begin{aligned} \text{If } P \neq Q, \quad \delta &= \frac{y_2 - y_1}{x_2 - x_1} \mod p \\ \text{If } P = Q, \quad \delta &= \frac{3 \cdot x_1^2 + a}{2 \cdot y_1} \mod p \\ x_3 &= \delta^2 - x_1 - x_2 \mod p \\ y_3 &= \delta \cdot (x_1 - x_3) - y_1 \mod p \end{aligned} \tag{9}$$

2. Scalar Multiplication. Given point $P = (x, y)$, it can be added to itself k times thus computing $k \cdot P \mod p$.

Encryption typically involves the generation of ciphertexts using carefully chosen elliptic curve points. The security of this process fundamentally relies on the Elliptic Curve Discrete Logarithm Problem (ECDLP),[56,58] which asserts that given a generator point G on an elliptic curve, a random scalar k and another point $P = k \cdot G$, it is computationally infeasible to determine the scalar k within a reasonable time frame. This property allows for the creation of a highly secure and efficient alternative to traditional asymmetric cryptographic methods, enabling strong encryption with smaller key sizes, reduced computational overhead, and enhanced performance in a variety of modern applications.[59–61]

2.5 Elliptic curve elGaman encryption

ECEG is a probabilistic public-key partially homomorphic cryptosystem based on ECC and the ElGamal scheme.[53] It works as follows:

- Key generation. The user selects an appropriate elliptic curve E over a finite field F_p and chooses a base point $G \in E(F_p)$ that generates a large cyclic subgroup of points on the curve with order n. This point G functions as the generator. Since verifying the security of a curve is highly computationally intensive, several organizations and international bodies, such as the National Institute of Standards and Technology (NIST), provide a list of verified and recommended curves.[62] Once a valid curve is chosen, a random integer is assigned as a secret key $s_k \in \{1, 2, \dots n-1\}$. The public key is then computed, $P_k = s_k \cdot G$.
- Encryption. The message m is mapped to a point $M \in E(F_p)$. An ephemeral key $k \in \{1, 2, \dots n-1\}$ is chosen. The ciphertext comprises two values $C = (C_1, C_2)$, computed as follows:

$$\begin{aligned} C_1 &= k \cdot G \\ C_2 &= M + k \cdot P_k \end{aligned} \tag{10}$$

- Decryption. Using the private key s_k, the mapped message is $M = C_2 - s_k \cdot C_1$. The original message m can be obtained by inverting the mapping.

The original ElGamal is multiplicatively homomorphic modulo p, whereas ECEG is additively homomorphic. For messages m_A, m_b and mappings M_A, M_B, their ciphertexts can be combined in a way that reflects the addition of the plaintexts:

$$\begin{aligned} E(M(m_A)) = E(M_A) = C_A &= (C_{A1}, C_{A2}) \\ E(M(m_B)) = E(M_B) = C_B &= (C_{B1}, C_{B2}) \\ E(M_A) + E(M_B) &= (C_{A1} + C_{B1},\ C_{A2} + C_{B2}) \\ &= E(M_A + M_B) \end{aligned} \tag{11}$$

Here, $M()$ denotes the mapping function onto the elliptic curve E. The sum of the two ciphertexts corresponds to the sum of their plaintexts upon decryption. Two important aspects warrant attention. First, the additive homomorphism applies specifically to the mapped messages M_A and M_B; for most mapping functions, the original messages do not inherently exhibit this property. Second, the additive homomorphism in this context operates over elliptic curve point addition (as detailed in the 2.4 subsection), rather than conventional scalar addition. This distinction is essential, as it reflects the unique algebraic structure underlying elliptic curves.

The security of ECEG encryption relies on the difficulty of solving two problems. For both, an elliptic curve E defined over a finite field F_p, a generator point $G \in E(F_p)$ of a large cyclic subgroup of points on the curve, and some random values a and b are needed. The first is the Elliptic Curve Discrete Logarithm Problem (ECDLP),[56,58] which posits that, given a public point $P = a \cdot G$, it is computationally infeasible to find a from G and P. The second is the Elliptic Curve Computational Diffie-Hellman (ECCDH) Problem,[56,58] which asserts that, given $a \cdot G$ and $b \cdot P$, it is challenging to compute $a \cdot b \cdot G$ without prior knowledge of a or b.

The ECEG encryption scheme is secure under indistinguishability against chosen plaintext attacks (IND-CPA). To demonstrate this, an adversary is provided two messages m_0, m_1, which are mapped to points M_0, M_1, and requests the encryption of one of these points. For a randomly chosen bit $b \in \{0, 1\}$, the resulting encryption is:

$$C = (k \cdot G, M_b + k \cdot P_k) \tag{12}$$

The adversary's task is to determine whether C is the ciphertext of M_0 or M_1. The first component of C provides no information, as the random selection of k makes it computationally infeasible to deduce due to the ECDLP. Similarly, the second component, $M + k \cdot P_k$, also reveals no useful information. Since $k \cdot P_k = k \cdot s_k \cdot G$, under the ECCDH assumption, it is impossible to derive $k \cdot P_k$ without prior knowledge of k or s_k. Furthermore, as the addition of points in this second component of C occurs within an abelian group defined by the elliptic curve, an adversary cannot infer information about either of the two addends. Consequently, the adversary cannot distinguish between the encryption of M_0 and M_1 with a probability greater than random guessing. Similar to the conventional ElGamal cryptosystem, certain known attacks on ECEG include:

- Brute force attack. An adversary attempts to directly guess the private key; however, this approach is computationally infeasible for large values of p.
- Baby-Step Giant-Step algorithm.[63,64] This is a time-space trade-off algorithm that allows an adversary to solve the ECDLP in $O(\sqrt{p})$ time and memory usage.
- Pollard's Rho algorithm.[65,66] An integer factorization approach that also achieves $O(\sqrt{p})$ time complexity but with significantly reduced memory usage.
- Quantum attacks. If realized, quantum computers could potentially break the ECDLP by employing Shor's algorithm.[67–69]

3 Algorithm design

In this section, a general overview of the algorithm is presented. It is visualized in Figure 1 and the pseudo-code is portrayed in Algorithms 1 and 2.

The scheme is divided in two sections. Firstly, the ID is created (or initialized). During this step, a trusted entity **S** generates a key pair for an additive homomorphic cryptosystem. The public key is saved in the final document or device. Additionally, the user's biometric data is captured

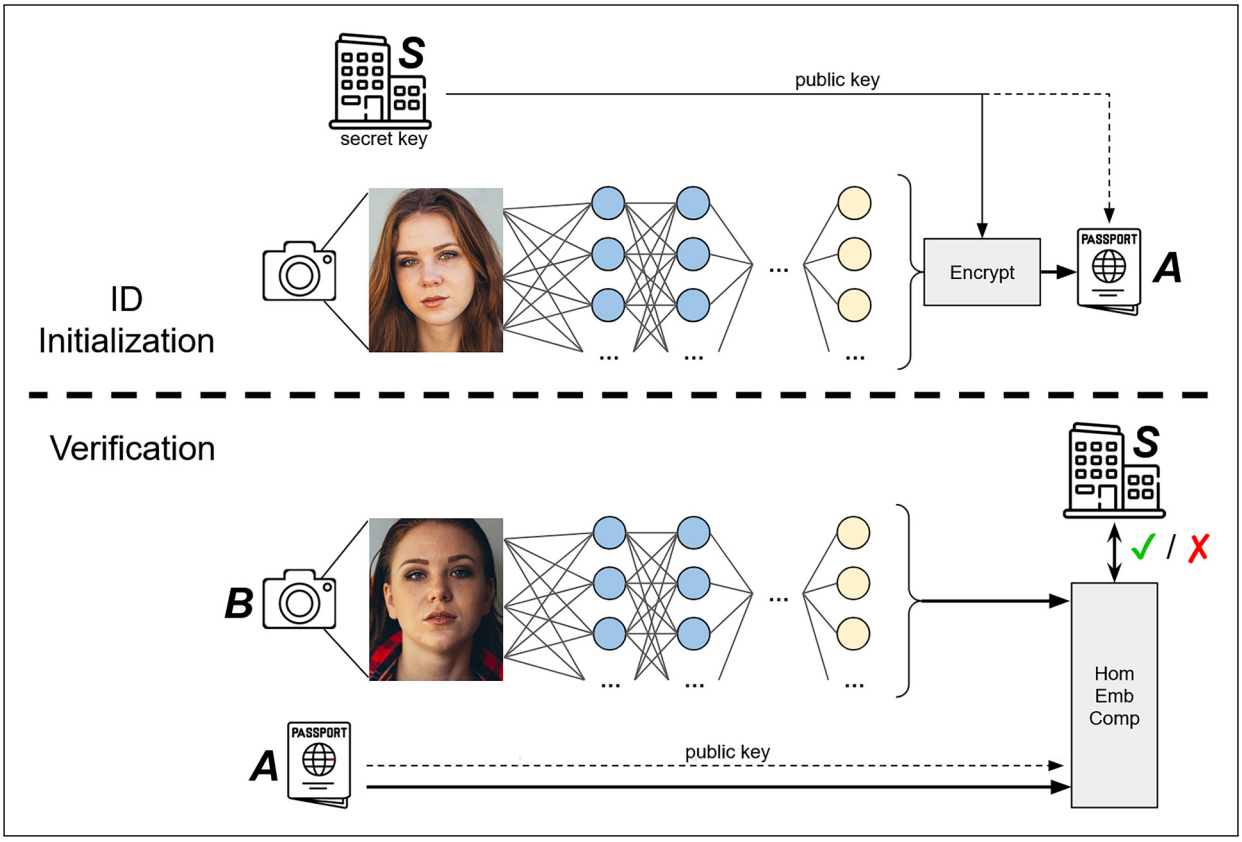

Figure 1. Schematic of ID initialization and verification. Note, both images are obtained from a publicly available digital image repository that provides full permission for academic and commercial use.

(in this example, a frontal picture of their face is taken), an embedding is extracted (typically through a neural network), the embedding is homomorphically encrypted, and subsequently stored within the same document or device. It should be noted that the algorithm used for obtaining embeddings must be publicly known and highly standardized, as it will also be employed during the verification process. Nonetheless, this work does not impose any specific properties or implementations for procuring embeddings; in other words, it is completely **agnostic to embedding extraction**. Furthermore, as the homomorphic operations are performed exclusively on the embeddings vector, no computational overhead is introduced during the extraction phase. The specific homomorphic system and its necessary properties will be described in a subsequent subsection. Moreover, to prevent unauthorized IDs, all stored data on a document or device is digitally signed by **S**. Last, the secret key is stored in a secure database owned by **S** in accordance with best practices for key management in cryptographic systems. This ensures no central point of failure, as an attacker would need to both breach this database and steal the document or device containing the ID.

In the second step, the user's document is compared to an additional sample of biometric data. An arrival at an airport can be seen as a practical example, where a person must proceed through border control. The user presents their ID, and a frontal face picture is taken; these are then compared to verify a match. This is a standard case of biometric authentication. Within the context of the presented scheme, once the biometric data is obtained, an embedding vector is extracted using the same methodology as the initial step. Utilizing the public key saved on the document, the user's stored encrypted embeddings are homomorphically compared to the extracted vector using a distance function (typically cosine or Euclidean distance). The result is sent to the entity responsible for the ID issuance **S**. The entity then decrypts the resulting ciphertext using the corresponding secret key and compares the outcome to a predefined threshold. A single bit is returned, indicating either a "match" or "not match". It is important to note that a decryption request must be made to **S**, necessitating an appropriate communication channel. This can be achieved through a secure http (https) request for a single verification or via a pre-established two-way simultaneous communication channel for multiple verifications. The continuous need for a connection introduces some overhead and poses a limitation that will be thoroughly examined in the 6 section.

Algorithm 1. ID generation pseudo-code.

Input S: enc_method ▷ S does all computation
Input A: $picture_A$
if $enc\_method == Paillier$ **then**
  $k_s, k_p = GenKeys_{Paillier}(\lambda_P)$
else if $enc\_method == ECEG$ **then**
  $k_s, k_p = GenKeys_{ECEG}(\lambda_{EC})$
end if
$\overline{x}_{10} = Exctract\_Normalized\_Embeddings(picture_A)$
for $i = 0; i < l; i{+}{+}$ **do**
  $x_i = int(x_{10i} \cdot 2^{bp})$ ▷ Scalar conversion
  if $enc\_method == Paillier$ **then**
    $c_i = E_{Paillier}(x_i, k_p)$
  else if $enc\_method == ECEG$ **then**
    $X_i = M(x_i)$
    $C1_i, C2_i = E_{ECEG}(X_i, k_p)$
    $c_i = C1_i.x || C2_i.x$ ▷ Only x coordinate is taken
  end if
end for
$v = Sign(\overline{c}||k_p)$
Output A: $\overline{c}$, k_p and v ▷ All stored in the ID

The functions $GenKeys_{Paillier}()$, $GenKeys_{ECEG}()$, $E_{Paillier}()$ and $E_{ECEG}()$ are detailed in the Paillier Encryption and Elliptic Curve ElGaman Encryption subsections. The $M()$ function performs the ECEG point mapping. $Sign()$ represents any conventional digital signature procedure. Last, $||$ represents concatenation.

Algorithm 2. ID cosine comparison pseudo-code.

Input B: $picture_B$ ▷ B and S do all computation
Input A: $\overline{c}$, k_p and v
$\overline{y}_{10} = Exctract\_Normalized\_Embeddings(picture_B)$
$Check\_Validity(v)$ ▷ End process if signature is invalid
for $i = 0; i < l; i{+}{+}$ **do**
  $y_i = int(y_{10i} \cdot 2^{bp})$ ▷ Scalar conversion
  $c_{res} += c_i * y_i$
end for

Data sent to S: c_{res}
if $enc\_method == Paillier$ **then**
  $m_{res} = {E^{-1}}_{Paillier}(c_{res}, k_s)$
else if $enc\_method == ECEG$ **then**
  $C1_i.x, C2_i.x = Separate(c_i)$ ▷ Undo concatenation
  $C1_i = Decompress(C1_i.x)$
  $C2_i = Decompress(C2_i.x)$
  $M_{res} = {E^{-1}}_{ECEG}(C1_i, C2_i, k_s)$
  $m_{res} = M^{-1}(M_{res})$
end if
$m_{10res} = m_{res}/2^{2 \cdot bp}$ ▷ Decimal conversion
$match = m_{10res} < treshold$
Output S: $match$ ▷ Result is returned to B

The functions ${E^{-1}}_{Paillier}()$ and ${E^{-1}}_{ECEG}()$ are detailed in the Paillier Encryption and Elliptic Curve ElGaman Encryption subsections. Functions $Decompress()$, $M()$ and $M^{-1}()$ are described in the EC ElGamal Embedding Comparison section. Last, the $Check\_Validity()$ function, once more, utilizes any conventional digital signature verification procedure.

3.1 Homomorphic distance function

As aforementioned in the 2.2 subsection, fully or somewhat homomorphic systems that support both addition and multiplication of ciphertexts are orders of magnitude slower than partially homomorphic algorithms and restrict the number of operations before data degradation occurs. Therefore, to achieve efficient functionality (as described in the 1.1 section), this work aims to employ PHE.

The chosen distance functions, namely cosine and Euclidean distance, can be computed using only an additive homomorphism. It should be noted that additive PHE also supports multiplication by a known scalar, as this operation can be achieved by adding a ciphertext to itself iteratively. A similar technique was followed in prior work on garbled embedding comparison.[70]

- **Normalized cosine distance**.

$$s = \frac{\bar{x} \cdot \bar{y}}{|x| \cdot |y|} = \sum \left(\frac{x_i}{|x|} \cdot \frac{y_i}{|y|} \right) = \sum (x_i \cdot y_i) \tag{13}$$

As it can be seen, the conventional cosine distance can be broken down into a sum of products. If the values are normalized, the fraction denominator can be removed. For two parties, **A** and **B**, one party (e.g., **A**) can encrypt its respective values x_i and transmit the resulting ciphertexts to the other party. The receiver can then multiply these ciphertexts by its own known values y_i and subsequently sum the results.

- **Normalized euclidean distance**.

$$s = \sqrt{2 \cdot (1 - \text{similarity}_{\cos})}$$
$$s^2 = 2 - 2 \cdot \text{similarity}_{\cos} \tag{14}$$

Only in the case of normalized vectors does the Euclidean distance become almost identical to the cosine distance. No additional cryptographic functionality is required to compute this similarity; instead, the cosine distance is calculated and the desired output is obtained from it. The squared value of the distance is adopted to avoid computationally expensive evaluation of square root operations.

An essential consideration must be addressed. Embedding values are represented as decimal numbers in floating-point notation, rendering them incompatible with most HE schemes without prior transformation. As in Hristov-Kalamov et al.,[70] the scheme utilized here follows a similar approach. To simplify the process, the embedding vectors will be restricted to positive values and, as a first step, will be normalized. Subsequently, all floating-point inputs are converted to a fixed-precision decimal system and treated as scalars; i.e., they are "shifted" left enough times to eliminate the decimal part. For instance, using 8 bits of precision ($bp = 8$), the number 0.7894561237_{10} is approximated to the nearest fixed representation, 0.11001010_2 which is equivalent to 0.7890625_{10}:

$$\frac{1}{2^1}+\frac{1}{2^2}+\frac{0}{2^3}+\frac{0}{2^4}+\frac{1}{2^5}+\frac{0}{2^6}+\frac{1}{2^7}+\frac{0}{2^8} \tag{15}$$

The value is then treated as a scalar, yielding $11001010_2 = 202_{10}$. To revert this transformation, the value is divided by 2^{bp}. It is crucial to note that, to maintain the decimal behavior of arithmetic operations, special adjustments are made for multiplications. For two transformed numbers a, b with bit precision bp_a, bp_b respectively, and resulting product $c = a \cdot b$, the value c must be divided by $2^{bp_a+bp_b}$ to restore it to decimal form. This adaptation simulates shifting of the "point" during decimal multiplication. In this particular case, it is necessary to account for one product with inputs of equal length bp. Thus, the resulting scalar will be divided by $2^{2 \cdot bp}$. Finally, it is worth noting that the choice of bit precision bp will depend on the desired accuracy and the constraints of the chosen additive homomorphic encryption scheme.

3.2 Paillier embedding comparison

The Paillier cryptosystem is one of the two chosen schemes for this research work. As explained in the 2.3 section, it is an additive PHE algorithm. The implementation for the ID generation is relatively straightforward. First, for a given security parameter λ, a suitable Paillier composite length is chosen λ_P. A key pair (s_k, p_k) is generated by the trusted entity **S** responsible for creating the ID. After capturing the biometric data, embeddings are extracted, normalized, transformed into scalars, and subsequently encrypted using the public key p_k. Every single value e_{Ai} is encrypted separately. Therefore, for an embedding vector of length l, l ciphertexts $E(e_{Ai})$ are obtained, each with a size of $2 \cdot \lambda_P$ bits. All ciphertexts, with the public key p_k, are stored in the document.

The homomorphic embedding comparison is also straightforward. Once again, l normalized scalar embeddings e_{Bi} are captured. The cosine distance is computed as follows: $r = \sum(E(e_{Ai}) \cdot e_{Bi})$. Note that the public key p_k is required for these operations. The output r is then sent to the trusted entity **S**, where it is decrypted using the secret key s_k to obtain the distance $d = E^{-1}(r)$. If the Euclidean distance is requested, the value is computed as $d = 2 - 2 \cdot E^{-1}(r)$, as indicated in the Homomorphic Distance Function formula. The resulting value is compared to a predefined threshold, returning "match" or "not match".

3.3 EC ElGamal embedding comparison

The EC ElGamal cryptosystem is the second scheme chosen for this research work. As outlined in the 2.5 subsection, it is an additive PHE algorithm; however, the additive homomorphism works on elliptic curve points rather than on integers. Consequently, a custom mapping function is utilized to adapt this algorithm. In this context, mapping refers to transferring a message from a specific domain to another, specifically from an integer to an elliptic curve point. Most mappings are unsuitable for this scenario, as they do not preserve the linear additive behavior of integers. A somewhat one-way **scalar multiplication mapping** is chosen. For a curve of order n, generator G, and a message $m \in \{0, 1, \ldots n-1\}$, the corresponding point M is computed as $M = m \cdot G$. It should be noted that, for a secure curve with properly chosen parameters, this mapping function is effectively one-way; recovering m from M would require solving the ECDLP. Nevertheless, in this case, the message range is known and determined by the bit precision bp. For sufficiently small ranges, the Baby-Step Giant-Step algorithm can be used to retrieve m. Thus, the complexity of reversing the mapping is $O(\sqrt{2^{bp}})$, which is exponentially dependent on bp. This aspect will be measured and examined in the 5 and 6 sections. As demonstrated, for message m_1, m_2 and corresponding points M_1, M_2, this technique preserves the required additive property:

$$\begin{aligned} M_3 &= M_1 + M_2 = m_1 \cdot G + m_2 \cdot G \\ &= (m_1 + m_2) \cdot G \end{aligned} \tag{16}$$

An additional key concept is **point compression**. Any ECEG ciphertext consists of two points $C1$ and $C2$; furthermore, each point is defined by two coordinates, x and y. To reduce ciphertext size, a point can be compressed by storing only the x coordinate along with a single bit indicating the sign of y. To decompress this representation, the elliptic curve equation must be solved ($y^2 = x^3 + a \cdot x + b \mod p$), which introduces computational overhead due to the need for calculating a modular square root. In the 5 section, this overhead will be quantified. Once implemented, this compression scheme approximately halves the size of a ciphertext.

During ID generation, given a security parameter λ, a suitable elliptic curve is chosen, usually with prime length $\lambda_{EC} = 2 \cdot \lambda$. A key pair (s_k, P_k) is generated by the trusted entity **S**, responsible for creating the ID. Biometric data is then captured, embeddings are extracted, normalized, converted into scalar values, mapped to points on the elliptic

curve, and encrypted using the public key P_k. Every single value e_{Ai} is encrypted separately. Therefore, for an embedding vector of length l, l ciphertexts $E(e_{Ai})$ are obtained, each with a size of $2 \cdot \lambda_{EC}$ bits. All ciphertexts are stored in the document.

For the homomorphic embedding comparison, l normalized scalar embeddings e_{Bi} are captured. The cosine distance is computed as follows: $R = \sum(E(e_{Ai}) \cdot e_{Bi})$. The output R is sent to the trusted entity **S**, where it is decrypted using the secret key s_k and subsequently unmapped as $d = M^{-1}(E^{-1}(R))$. If the Euclidean distance is requested, it is computed as $d = 2 - 2 \cdot M^{-1}(E^{-1}(R))$, as denoted in the Homomorphic Distance Function formula. Last, the threshold comparison is performed, yielding a "match" or "no match" result. It should be emphasized that the public key p_k is not strictly necessary for the homomorphic operations and therefore it does not need to be stored within the ID.

4 Security analysis

In this section, the presented homomorphic ID scheme is analyzed within both semi-honest and malicious settings, considering the perspectives of all participants. The involved parties are as follows: the user **A**, the verifier **B** (e.g., the customs agency), and the ID generator **S**. Furthermore, the entity **S** is divided into two sub-entities: one responsible for creating the ID, $\mathbf{S_1}$, and another tasked with decrypting the final result, $\mathbf{S_2}$. In this expanded configuration, $\mathbf{S_2}$ obtains the private key from $\mathbf{S_1}$. Since $\mathbf{S_1}$ and $\mathbf{S_2}$ perform distinct tasks, this division allows for the implementation of differentiated security standards for each sub-entity.

The only assumption made is that $\mathbf{S_1}$ is **completely honest**. This entity is assumed to generate a valid key pair, correctly encrypt and store the biometric data on the document, and ensure that this data is neither leaked nor stored elsewhere. Without this assumption, the scheme would be rendered invalid. It should be noted that this requirement is realistic, as special precautions can be implemented during the ID creation process to prevent information leakage or involvement from untrusted parties.

4.1 Security of ID generation

During the generation stage, only two parties are involved: **A** and $\mathbf{S_1}$. As aforementioned, $\mathbf{S_1}$ is assumed to be fully trustworthy. Concerning **A**, in the semi-honest setting, the security of the scheme depends on the irreversibility of the underlying PHE. As demonstrated in the 2.3 and 2.5 subsections, both Paillier and ECEG schemes ensure privacy against a computationally bounded adversary. In the malicious setting, **A** remains passive, meaning it does not perform any computation and simply receives the final ID. Consequently, there is no possibility for malicious interference.

4.2 Security of biometric comparison

In this step, three parties are involved: **A**, **B**, and $\mathbf{S_2}$. The private key has been securely transferred to $\mathbf{S_2}$ by $\mathbf{S_1}$. In the **semi-honest setting**, security against a compromised party **A** is, once again, based on the irreversibility of the PHE cryptosystem. The use of Paillier and ECEG provides this property, as described in the 2.3 and 2.5 subsections.

For **B**, the situation is different. During the initial homomorphic comparison, as with the previous scenarios, no information leakage occurs provided that the PHE systems in use are secure. However, once the result is returned from $\mathbf{S_2}$, a single bit of information is inevitably revealed. Nevertheless, it should be noted that any verification process definitionally reveals at least one bit indicating success or failure, regardless of the method employed. Given this understanding, the scheme remains secure against a semi-honest adversary in the role of **B**.

Party $\mathbf{S_2}$ receives only a single ciphertext value, which it decrypts using the private key p_k. Therefore, there is no difference whether it is honest or compromised. $\mathbf{S_2}$ does not possess the original user embedding and only obtains a random distance value. Without additional context about what is being compared, it is virtually impossible to recover any sensitive information.

In the **malicious setting**, where parties may deviate from the established protocols, certain security aspects are undermined. Party **A** has no alternative but to present valid information. All data included in its document is signed by the honest and trusted $\mathbf{S_1}$. Therefore, if incorrect data is provided, **B** can immediately terminate the protocol. Party **A** performs no additional actions and cannot interfere in any other way. Therefore, the protocol remains secure against a malicious **A**.

Since parties **B** and $\mathbf{S_2}$ are responsible for performing computations, they can send or return invalid results. The protocol is not secure against malicious **B** or $\mathbf{S_2}$. Additionally, security is not guaranteed in cases where **A** with **B**, **A** with $\mathbf{S_2}$, or **B** with $\mathbf{S_2}$ are colluding.

To sum up, the proposed solution is fully secure in the semi-honest setting and partially secure in the malicious setting. This partial security is sufficient for real world applications since the entities responsible for ID generation (e.g. government agencies) and ID verification (e.g. customs agencies) are trustworthy.

5 Implementation and results

A standard security parameter of $\lambda = 128$ bits is selected. For the Paillier implementation, to meet the desired security level, $\lambda_P = 2048$ is used, meaning each chosen prime will have an approximate length of 1024 bits. The Intel Paillier Cryptosystem Library[71] is employed, which supports hardware acceleration by leveraging specific 512-bit registers to optimize arithmetic operations. For the ECEG

Table 1. Accuracy % based on bits of precision and embedding lengths. Starting from a standard accuracy of 100% computed in the clear with floating-point precision, this table measures the fixed decimal precision error percentage.

Emb Lengths	$bp = 8$ (%)	$bp = 10$ (%)	$bp = 12$ (%)
128	99.82	99.95	99.99
256	99.83	99.95	99.99
512	99.82	99.96	99.99
4096	99.82	99.95	99.99

Table 2. Paillier computation (C) and decryption (E^{-1}) times in ms.

Emb Lengths	Operation Type	$bp = 8$ (ms)	$bp = 10$ (ms)	$bp = 12$ (ms)
128	C	6.168	7.122	8.132
	E^{-1}	2.128	2.129	2.129
256	C	11.863	13.769	15.795
	E^{-1}	2.131	2.131	2.133
512	C	22.799	26.513	30.469
	E^{-1}	2.139	2.137	2.136
4096	C	162.595	176.207	189.552
	E^{-1}	2.139	2.113	2.137

implementation, achieving 128 bits of security requires operating on well-known curves like secp256r1 (NIST P-256), which utilizes a prime number p with a length of 256 bits ($\lambda_{EC} = 256$). Arithmetic operations between curve points are performed using the OpenSSL library.

For testing purposes, four different embedding lengths are chosen, based on popular neural network algorithms for face verification. Specifically, the vector sizes are 128, 256, 512, and 4096. In addition, three increasing levels of bit precision are tested: 8, 10, and 12 bits.

All the experimental results are gathered on an AWS EC2 r7iz instance, powered by a 4th Gen Intel Xeon Scalable (Sapphire Rapids) processor operating on a **single core**. This architecture enables the Intel Paillier Cryptosystem Library to fully leverage its hardware optimizations. The full code is available at Hristov-Kalamov,[72] and the profiling results are shown in Tables 1 to 5.

6 Discussion

Before analyzing specific results, it is important to discuss the chosen embedding lengths. The values correspond to widely recognized face comparison models and libraries such as FaceNet (128 vector dimensions), DeepFace (4096 vector dimensions), VGGFace (4096 vector dimensions), VGGFace2 (512 vector dimensions), ArcFace (512 vector dimensions), CosFace (512 vector dimensions), SphereFace (512 vector dimensions), Dlib (128 vector dimensions), OpenFace (128 vector dimensions) and the 256-dimensional variant of ArcFace (256 vector dimensions).

Table 3. ECEG decompression (D), computation (C), decryption (E^{-1}) and unmapping (M^{-1}) times in milliseconds.

Emb Lengths	Operation Type	$bp = 8$ (ms)	$bp = 10$ (ms)	$bp = 12$ (ms)
128	D	2.704	2.703	2.701
	C	3.411	3.409	3.410
	E^{-1}	0.043	0.043	0.043
	M^{-1}	8.858	35.104	141.492
256	D	5.403	5.4	5.395
	C	6.763	6.762	6.77
	E^{-1}	0.043	0.043	0.043
	M^{-1}	8.844	35.464	140.347
512	D	10.793	10.799	10.795
	C	13.492	13.470	13.472
	E^{-1}	0.044	0.044	0.044
	M^{-1}	8.811	35.271	139.584
4096	D	86.146	86.134	86.100
	C	111.788	111.459	111.106
	E^{-1}	0.050	0.049	0.050
	M^{-1}	9.205	35.301	140.999

Table 4. Size of ciphertexts in MiB.

Emb Lengths	Paillier (MiB)	ECEG (MiB)
128	0.063	0.008
256	0.125	0.016
512	0.25	0.031
4096	2	0.25

Table 5. Additional memory needed for computation in KiB.

Precision Bits	Paillier (KiB)	ECEG (KiB)
8	0	3
10	0	12
12	0	48

It should be noted that most modern face comparison schemes are trained using angular loss, triplet loss, or contrastive loss functions, which typically work with L2 normalized inputs and outputs. Therefore, the authors' choice to compute distances between normalized vectors is completely valid and additionally offers greater computational efficiency.

In Table 1, the cosine distance was first computed in the clear, using standard floating-point operations and subsequently calculated using our fixed-precision approach with 8, 10 and 12 bits of precision. The latter results were then compared to the former. Two aspects can be observed. On the one hand, accuracy is independent of embedding length. On the other hand, accuracy increases slightly with

a higher fixed-bit count. While this is intuitively expected, the observed improvement remains minimal.

In Table 2, homomorphic arithmetic and decryption times were recorded as a function of embedding size and bit precision. Upon examining the computation metrics, a slight linear increase in time is observed with higher bit precision, while a significantly larger linear increase is seen with longer embedding lengths. Last, decryption time remains constant across all cases, as only a single final value requires decryption in each scenario. This behavior is consistent with expectations.

In Table 3, the times for EC point decompression, homomorphic computation, decryption and unmapping were measured as a function of embedding size and bit precision. Point decompression and PHE computation remain constant in regard to bit precision but increase linearly as embedding length grows. Decryption is both fully constant and incredibly fast, owing to the fact that, irrespective of setup, deciphering the result requires only a single point multiplication and addition. Finally, the time to unmap a point to a valid message is exponentially dependent on bit precision, which can be attributed to the $O(\sqrt{2^{bp}})$ time complexity of the Baby-Step Giant-Step algorithm.[63,64]

Table 4 provides a straightforward measurement of the ciphertext sizes for both schemes. It is crucial to note that these values affect the storage requirements for the protected ID as well as the networking considerations when sending a single encrypted result to the trusted entity S.

Table 5 indicates the additional auxiliary memory required for each approach. Only ECEG uses an increase in RAM usage as bit precision increases, due to the $O(\sqrt{2^{bp}})$ space complexity of the Baby-Step Giant-Step algorithm.[63,64] When precomputing the baby steps, a dictionary is necessary to store each point alongside its corresponding value. In this implementation, each point is hashed to a 64-bit integer, while the associated number is represented by a 32-bit integer. Therefore, the memory requirements can be expressed as $mem = 96 \cdot 2^{bp}$ bits. For the future work, the Pollard's Rho algorithm could replace Baby-Step Giant-Step for unmapping, significantly reducing overall space consumption.

After analyzing the data presented in the results tables, several considerations emerge. For the Paillier solution, bit precision proves practically irrelevant, allowing accuracy to be prioritized. A precision higher than those tested, such as **16 bits**, may be adopted. However, this scheme might be less suitable for ID cards or devices with limited storage capacity, as its ciphertexts occupy eight times the space of the ECEG variant. Conversely, ECEG execution times are far more dependent on fixed precision bit count. The minimal viable accuracy required by a project's specific needs should be selected to expedite point unmapping as much as possible. As shown in Table 1, **8 bits** achieves 99.8 % accuracy, which is generally sufficient for most applications. The future work will explore two alternative approaches that may mitigate this limitation:

- Identifying a point mapping function that preserves scalar homomorphism without exhibiting exponential complexity behavior.
- Retraining the face comparison model used, incorporating custom fixed-precision arithmetic in the loss function. However, this approach would render the proposed solution no longer agnostic to the underlying neural network.

As a final remark, it is worth noting that embedding length can be further reduced to 64 values, effectively halving the required PHE arithmetic. This technique is successfully leveraged in Bauspieß et al.[37] Additionally, further optimizations can be applied to the presented algorithms. Distance calculation, whether performed in the clear or homomorphically, is inherently parallelizable and can be accelerated by utilizing multiple CPU cores or GPU processing. This means that parallelization linearly improves computation speed; for instance, using eight cores would decrease execution times approximately by a factor of eight. Since this linear behavior is very common in cryptographic research, establishing a **1-core standard** with consistent hardware specifications for benchmarking is highly advisable. Such a standard would ensure consistency when comparing algorithm performance across different systems and studies.

6.1 Disadvantages and limitations

In the 3 section, it is stated that the presented solution requires a constant internet connection to execute biometric verification effectively. This dependency poses a disadvantage compared to existing systems that do not have such a requirement. The ECEG variant with point compression reduces the data exchange to a 512-bit request and 1-bit response; however, the necessity of an internet connection remains a potential limitation.

Furthermore, the stringent protection of biometric data may present practical challenges. For instance, in a customs checking scenario, this level of protection may complicate the tasks of border control agents, as a face image is not immediately available on the document. This limitation is inherent to the design of this approach and is explicitly defined as an objective in alignment with privacy by design principles. If a biometric comparison can be performed without accessing the underlying sensitive data, it must be conducted accordingly. Nevertheless, the authors acknowledge that such rigorous standards may present difficulties in practical implementation. Moreover, the proposed approach requires synchronization among multiple entities, such as various governments, border control agencies, and

other organizations involved in the process. Thus, the complexity of this system may introduce additional challenges in terms of deployment.

Finally, and perhaps most importantly, the data comparison process is entirely reliant on a neural network. In fact, it is impossible for a human to access the underlying information. Consequently, the effectiveness of the solution is highly dependent on the accuracy and reliability of the verification model.

6.2 Additional use cases

While the primary aim of this research work is the development of a protected ID for customs and border control, homomorphic biometric comparison has potential applications across various other domains. Any secure access point could utilize this scheme. For instance, a private facility might require a biometric sample, such as an iris scan, for entry. To avoid central storage of this information, an encrypted ID could be issued and employed alongside the biometric scan.

An emerging use case lies in payment systems that leverage biometric authentication for enhanced security. For instance, fingerprint-based payment methods are becoming increasingly prevalent, wherein a user's fingerprint is compared to an encrypted version of their fingerprint ID, stored either on their mobile device or in the cloud. In these scenarios, the encryption ensures that sensitive biometric data remains secure while enabling user-friendly operation. Such scenarios would follow an identical process, whereby a PHE-encrypted ID is pre-generated by a trusted entity and homomorphically compared to an extracted fingerprint during a payment transaction. These systems are typically automated, requiring no human supervision, and are often managed by private companies, which further underscores the need for robust privacy protections.

Last, the proposed ID system can be adopted in law enforcement agencies to guarantee the protection of citizens' civil rights. For example, if a police agency possesses a picture of a crime suspect and aims to match it against a broad population, homomorphic identification can be employed to conduct the comparison without accessing individuals' underlying biometric data. This approach preserves both privacy and security throughout the identification process.

6.3 Other methods comparison

A comparison with alternative approaches is performed in this subsection as well as a discussion on whether these other frameworks meet the laid-out requirements in the 1.1 subsection. First, FHE is analyzed as it is the primary candidate for an alternative. As described in the 1.1 section, many implementations utilize FHE to achieve a similar purpose. Additionally, since FHE supports all the operations and privacy features of SHE, it also meets most of the outlined requirements. Utilizing the latest literature that profiles the different state-of-the-art FHE libraries,[73–75] a direct comparison is made emulating the necessary operations for a cosine distance performed exactly as detailed in the Homomorphic Distance Function formula. To conduct this comparison, the four principal libraries most commonly referenced in the literature have been employed, namely, https://github.com/microsoft/SEALSEAL, https://gitlab.com/palisadePALISADE, https://github.com/homenc/HElib HElib and https://www.ckks.org/HEAAN. Specifically, the CKKS scheme is examined as it is the main one used in privacy-preserving deep learning, especially in the biometric or biomedical fields.

Table 6. Cost of CKKS operations depending on circuit depth (minimum - maximum) based of Gouert et al.[73]; Takeshita et al.[74]; Zhu et al.[75]

Operation	SEAL (ms)	PALISADE (ms)	HElib (ms)	HEAAN (ms)
Addition	3 - 30	4 - 40	5 - 35	3 - 25
Product	25 - 200	20 - 250	30 - 230	25 - 200
Rotation	80 - 600	75 - 800	90 - 650	80 - 600
Decryption	10	8	12	10
Decoding	8	7	10	8

To compute the cosine distance as efficiently as possible, batching is utilized allowing for all embedding elements to be encrypted into the same ciphertext. A singular ciphertext representing **A**'s data will be multiplied with an encoded vector containing **B**'s information. The result will be rotated and added to itself $\log_2(l)$ times. Last, a decryption and decoding will be performed to obtain a usable value. Therefore, a product with a known value, $\log_2(l)$ rotations, $\log_2(l)$ additions, a decryption and a decoding are needed.

The required computation is dependent on circuit depth. FHE implementations have a so-called "noise budget" that measures the number of operations that can be performed, i.e. the depth of the circuit. If this budget is spent, a valid output can no longer be decrypted. To reset the "noise budget", a special calculation called bootstrapping is carried out. This action is very performance-intensive. When setting up the FHE library, a shorter circuit depth can be chosen where each operation is faster, but bootstrapping might be required, or a longer depth can be implemented with slower operations. This aspect is showcased in Table 6 where the cost of each operation is visualized for the smallest and largest depths specific to the cosine distance. Moreover, Table 7 displays the total estimated timings for the shortest and longest depths compared to the timings obtained in this research work. The following parameters are chosen for the FHE calculations: security level $\lambda = 128$ bits and ring dimension $N = 16384$.

Analyzing the estimated timings in Table 7, some aspects are of note. On the one hand, a maximum depth circuit presents results orders of magnitude slower than the

Table 7. Cost of CKKS cosine distance depending on circuit depth (minimum - maximum) based of Gouert et al.[73]; Takeshita et al.[74]; Zhu et al.[75] and **our results (Paillier and ECEG)** detailed in Tables 2 and 3 for bit precision $bp = 8$.

Emb Lengths	SEAL (ms)	PALISADE (ms)	HElib (ms)	HEAAN (ms)	Paillier (ms)	ECEG (ms)
128	624 - 4630	588 - 6150	717 - 5050	624 - 4600	**8**	**15**
256	707 - 5260	667 - 6990	812 - 5730	707 - 5220	**14**	**21**
512	790 - 5890	746 - 7830	907 - 6420	790 - 5840	**25**	**33**
4096	1039 - 7780	983 - 10350	1192 - 8470	1039 - 7720	**165**	**207**

SHE findings obtained in this research work. On the other hand, while minimum depth values showcase somewhat shorter durations, it is important to mention that bootstrapping can take **several seconds** even on modern CPUs. Thus, the overall time also becomes orders of magnitude longer.

Additionally, the sizes of FHE elements, as profiled in Takeshita et al.,[74] are also of note. A compressed ciphertext can take approximately 30 KB of data, making it strictly better than our Paillier scheme but worse than our ECEG scheme for most practical embedding lengths. However, when uncompressed the ciphertext balloons to 4 - 8 MB of size, thus leading to much greater memory demands. Similarly, an uncompressed public key necessitates an additional 4 - 8 MB. Last, for the rotation operation to be possible, another element is required, the Galois keys. These can take up to 30 MB of space, imposing some limitations on storage (especially if the digital ID is placed in a small embedded device or digital card). All of this leads to the conclusion that, without some custom implementation or strategy, FHE is less adequate for the specific scenario outlined in this publication than the presented Paillier and ECEG schemes.

Aside from HE, a comparison can be made with other privacy-preserving protocols. These can include Garbled Circuits (GC), Function Secret Sharing (FSS), Homomorphic Secret Sharing (HSS) or direct computing with Oblivious Transfer (OT) or Oblivious Linear Evaluation (OLE). In the previous work of Hristov-Kalamov et al.,[70] a similar setup of embedding comparison was implemented and profiled. This approach was based on exploiting a simple homomorphism of a One-Time Pad coupled with the OT and OLE protocol. The proposed solution emphasized speed above all else. The measured computation times were significantly faster, requiring microseconds to complete, as shown in Tables 8 and 9. The memory and networking requisites were closer in size but also lesser, as portrayed in Table 10. However, this scheme relies on both parties performing some computation each time embeddings are compared. Therefore, as outlined in the 1.1 subsection, it breaks the **static functionality** requirement which demands that the holder of the ID (party **A**) cannot perform any computation and is thus **not suited**. The same can be said for most privacy-preserving alternatives, as each necessitates active participants. Even though there are much faster alternatives, HE remains the only suitable primitive for our specifications.

Table 8. Generator (**A**) computation of Hristov-Kalamov et al.[70]

Emb Length	OT_frac_mult & HOTP (μ s)	OLE_trans & HOTP (μ s)
128	1	$\leq$ 1
1024	10	1
4096	50	6

Table 9. Evaluator (**B**) computation of Hristov-Kalamov et al.[70]

Emb Length	OT_frac_mult & HOTP (μ s)	OLE_trans & HOTP (μ s)
128	1	$\leq$ 1
1024	12	1
4096	52	6

Table 10. Data exchange requirements of Hristov-Kalamov et al.[70]

Emb Length	OT_frac_mult & HOTP (kB)	OLE_trans & HOTP (kB)
128	8.5	1.5
1024	68	12
4096	272	48

This juxtaposition highlights how privacy-preserving algorithms can require drastically different amounts of resources, where the same operation (in this case, a cosine distance) can take up several seconds, milliseconds, or even microseconds and how seemingly small requirements can completely change the possible solutions.

As a final comparison, since this research work merges neural networks and cryptography, CryptoNets become a natural candidate. Recent advances in privacy-preserving machine learning have introduced architectures which are designed to perform inference over encrypted data by adapting neural networks to operate within cryptographic constraints. CryptoNets demonstrate the feasibility of learning directly on encrypted inputs through polynomial approximations of activation functions, offering an appealing direction for privacy-focused applications.[76,77]

Currently, CryptoNet-style approaches remain relatively early in their development and lack extensive empirical validation. These algorithms cannot, for now, substitute conventional cryptography and are thus excluded as potential substitutes. However, as part of the future work, the viability of CryptoNet-style architectures will be investigated in privacy-sensitive settings. Furthermore, we will consider adversarial scenarios in which malicious CryptoNet-like models may attempt to infer information from encrypted embeddings, and we will aim to design safeguards to mitigate such risks.

7 Conclusion

In this research work, a secure, privacy-preserving ID creation framework is presented. By leveraging HE, particularly PHE, the entire biometric comparison process remains encapsulated in a cryptographic layer, mitigating data theft and preventing information leakage. Two schemes are detailed, Paillier and ECEG, each offering distinct advantages in computational efficiency, memory usage, storage requirements, and network bandwidth.

Upon analyzing the objectives outlined in the 1.1 subsection, it can be shown that most of them are accomplished with only minor compromises. The algorithm is fully functional and optimized for efficiency, though it introduces some overhead for computation, memory, storage, and communication. The created document solely stores data, rendering it static. Biometric information is entirely irrecoverable without authorization from the party or entity that generated the ID. This one-way property is enforced by the IND-CPA security guarantees provided by the Paillier and ECEG cryptosystems. The biometric verification process is completely oblivious, ensuring that the party responsible for the verification process has no access to the underlying biometric data contained in the ID. Last, all encrypted information is stored locally with the user, requiring only a central database for the encryption private keys. The proposed solution is also fully compliant with the security and privacy properties outlined in the ISO standard for biometric embeddings,[25] as defined in the 1 section. Any issued ID can be renewed by decrypting the old document and generating a new one with a different private key. The original private key is discarded, ensuring that the obsolete ID cannot be reused. The data remains unlinkable and irreversible, supported by the IND-CPA security guarantees of the PHE schemes.

In conclusion, this framework highlights the potential for HE to transform sensitive applications like biometric verification. While there are performance trade-offs and deployment limitations, these are offset by robust privacy guarantees that safeguard biometric data throughout its lifecycle. The future work will focus on optimizing computational performance and exploring advanced methods, such as ciphertext packing, to further enhance efficiency while maintaining strong security.

Funding

This work was partially supported by the State Research Agency of Spain (Agencia Estatal de Investigación – AEI) under Grants PID2021-124176OB-I00 and PID2023-152984OB-I00.

Declaration of conflicting interests

The author(s) declared no potential conflicts of interest with respect to the research, authorship, and/or publication of this article.

ORCID iDs

Nikola Hristov-Kalamov https://orcid.org/0000-0002-2194-1112
Raúl Fernández-Ruiz https://orcid.org/0000-0001-8325-5372
Cristina Conde https://orcid.org/0000-0003-3548-0297
Agustín Álvarez-Marquina https://orcid.org/0000-0002-3387-6709
Francisco Domínguez-Mateos https://orcid.org/0000-0003-0909-7585
Pedro Gómez-Vilda https://orcid.org/0000-0003-3283-378X
Daniel Palacios-Alonso https://orcid.org/0000-0001-6063-4898

References

1. Bai Z and Zhang XL. Speaker recognition based on deep learning: An overview. *Neural Netw* 2021; 140: 65–99.
2. Chung JS, Nagrani A and Zisserman A. Voxceleb2: Deep speaker recognition. In: *Interspeech 2018*. 2018; pp.1086–1090. DOI: 10.21437/Interspeech.2018-1929.
3. Ravanelli M and Bengio Y. Speaker recognition from raw waveform with sincnet. In: *2018 IEEE Spoken Language Technology Workshop (SLT)*. 2018; pp.1021–1028. DOI: 10.1109/SLT.2018.8639585.
4. Adjabi I, Ouahabi A, Benzaoui A, et al. Past, present, and future of face recognition: A review. *Electronics* 2020; 9. DOI: 10.3390/electronics9081188.
5. Deng J, Guo J, Xue N, et al. Arcface: Additive angular margin loss for deep face recognition. In: *2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*. 2019; pp.4685–4694. DOI: 10.1109/CVPR.2019.00482.
6. Kortli Y, Jridi M, Al Falou A, et al. Face recognition systems: A survey. *Sensors* 2020; 20. DOI: 10.3390/s20020342.
7. Wang M and Deng W. Deep face recognition: A survey. *Neurocomputing* 2021; 429: 215–244.
8. Ali MM, Mahale VH, Yannawar P, et al. Overview of fingerprint recognition system. In: *2016 International Conference on Electrical, Electronics, and Optimization Techniques (ICEEOT)*. 2016; pp.1334–1338. DOI: 10.1109/ICEEOT.2016.7754900.

9. Arrindell KJ and Lennard GR. *Fingerprint Development Techniques: Theory and Application*. Hoboken, NJ: John Wiley & Sons, 2017.
10. Fei L, Lu G, Jia W, et al. Feature extraction methods for palmprint recognition: A survey and evaluation. *IEEE Trans Syst, Man, and Cyberne: Syst* 2019; 49: 346–363.
11. Genovese A, Piuri V, Plataniotis KN, et al. Palmnet: Gabor-pca convolutional networks for touchless palmprint recognition. *IEEE Trans Inform Forens Secur* 2019; 14: 3160–3174.
12. Zhong D, Du X and Zhong K. Decade progress of palmprint recognition: A brief survey. *Neurocomputing* 2019; 328: 16–28.
13. Nguyen K, Proença H and Alonso-Fernandez F. Deep learning for iris recognition: A survey. *ACM Comput Surv* 2024; 56. DOI: 10.1145/3651306.
14. Vera-Olmos F, Pardo E, Melero H, et al. deepeye: Deep convolutional network for pupil detection in real environments. *Integr Comput Aided Eng* 2019; 26: 85–95.
15. Chaverot M, Carré M, Jourlin M, et al. Improvement of small objects detection in thermal images. *Integr Comput Aided Eng* 2023; 30: 311–325.
16. Mokayed H, Quan TZ, Alkhaled L, et al. Real-time human detection and counting system using deep learning computer vision techniques. *Artificial Intell Applications* 2022; 1: 205–213.
17. Yang G and Chen S. Visual detection and tracking algorithms for human motion. *Multimed Tools Appl* 2023; 82: 47165–47188.
18. Muñoz-Montoro AJ, Revuelta-Sanz P, Villalón-Fernández A, et al. A system for biomedical audio signal processing based on high performance computing techniques. *Integr Comput Aided Eng* 2023; 30: 1–18.
19. European Parliament and Council of the European Union. Regulation (eu) 2016/679 of the european parliament and of the council of 27 april 2016 on the protection of natural persons with regard to the processing of personal data and on the free movement of such data (general data protection regulation). *Offic J Europ Union* 2016; L119: 1–88.
20. Sharma S, Saini A and Chaudhury S. A survey on biometric cryptosystems and their applications. *Comput Security* 2023; 134: 103458.
21. Yang W, Wang S, Cui H, et al. A review of homomorphic encryption for privacy-preserving biometrics. *Sensors* 2023; 23. DOI: 10.3390/s23073566.
22. Dosovitskiy A and Brox T. Inverting visual representations with convolutional networks. In: *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. 2016; pp.4829–4837. DOI: 10.1109/CVPR.2016.522.
23. Mai G, Cao K, Yuen PC, et al. On the reconstruction of face images from deep face templates. *IEEE Trans Pattern Anal Mach Intell* 2019; 41: 1188–1202.
24. Pittaluga F, Koppal SJ, Kang SB, et al. Revealing scenes by inverting structure from motion reconstructions. In: *2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*. 2019; pp.145–154. DOI: 10.1109/CVPR.2019.00023.
25. International Organization for Standardization. Iso/iec 27001:2022 information security, cybersecurity and privacy protection — information security management systems — requirements. https://www.iso.org/standard/75302.html (2022, accessed 19 October 2024).
26. Rivest RL and Dertouzos ML. *On Data Banks and Privacy Homomorphisms*. New York: Academic Press, 1978; pp.169–179.
27. Acar A, Aksu H, Uluagac AS, et al. A survey on homomorphic encryption schemes: Theory and implementation. *ACM Comput Surv* 2018; 51. DOI: 10.1145/3214303.
28. Adler A and Schuckers S. *Biometric Vulnerabilities, Overview*. Boston, MA: Springer US, 2009.
29. Natgunanathan I, Mehmood A, Xiang Y, et al. Protection of privacy in biometric data. *IEEE Access* 2016; 4: 880–892.
30. Berrang P, Humbert M, Zhang Y, et al. Dissecting privacy risks in biomedical data. In: *2018 IEEE European Symposium on Security and Privacy (EuroS&P)*. 2018; pp.62–76. DOI: 10.1109/EuroSP.2018.00013.
31. Islam S, Abba A, Ismail U, et al. Vulnerability prediction for secure healthcare supply chain service delivery. *Integr Comput Aided Eng* 2022; 29: 389–409.
32. Shafagh H, Hithnawi A, Burkhalter L, et al. Secure sharing of partially homomorphic encrypted iot data. In: *Proceedings of the 15th ACM Conference on Embedded Network Sensor Systems*, SenSys '17. New York, NY, USA: Association for Computing Machinery. 2017; ISBN 9781450354592. DOI: 10.1145/3131672.3131697.
33. Liu C, Chakraborty S and Verma D. *Secure Model Fusion for Distributed Learning Using Partial Homomorphic Encryption*. Cham: Springer International Publishing, pp, 2019. 154–179. DOI: 10.1007/978-3-030-17277-0_9.
34. Gomez-Barrero M, Maiorana E, Galbally J, et al. Multi-biometric template protection based on homomorphic encryption. *Pattern Recognit* 2017; 67: 149–163.
35. Gomez-Barrero M, Galbally J, Morales A, et al. Privacy-preserving comparison of variable-length data with application to biometric template protection. *IEEE Access* 2017; 5: 8606–8619.
36. Morampudi MK, Gonthina N, Bojjagani S, et al. Reliable and privacy-preserving multi-instance iris verification using paillier homomorphic encryption and one-digit checksum. *Signal Image Video Process* 2024; 18: 3723–3735.
37. Bauspieß P, Olafsson J, Kolberg J, et al. Improved homomorphically encrypted biometric identification using coefficient packing. In: *2022 International Workshop on Biometrics and Forensics (IWBF)*. 2022; pp.1–6. DOI: 10.1109/IWBF55382.2022.9794523.
38. Kolberg J, Bauspieß P, Gomez-Barrero M, et al. 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). 2019; pp.1–6. DOI: 10.1109/WIFS47025.2019.9034982.
39. Cavoukian A. Privacy by design: The 7 foundational principles - implementation and mapping of fair information practices. Technical report, Information and Privacy Commissioner of Ontario, Canada; 2009. https://www.ipc.on.ca/wp-content/uploads/resources/7foundationalprinciples.pdf.
40. Palacios-Alonso D, Cousido-González M, Domínguez-Mateos F, et al. Privacidad por diseño, clave para la buena gobernanza. *Derecom* 2021; 31: 215–223.
41. Goodfellow I, Bengio Y and Courville A. *Deep Learning*. Boston: MIT Press, 2016. http://www.deeplearningbook.org.
42. Schroff F, Kalenichenko D and Philbin J. Facenet: A unified embedding for face recognition and clustering. In: *2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. 2015; pp. 815–823. DOI: 10.1109/CVPR.2015.7298682.
43. Gentry C. Fully homomorphic encryption using ideal lattices. In: *Proceedings of the Forty-First Annual ACM Symposium on Theory of Computing*, STOC '09. New York, NY, USA: Association for Computing Machinery. ISBN 9781605585062, 2009; p. 169–178. DOI: 10.1145/1536414.1536440.
44. van Dijk M, Gentry C, Halevi S, et al. Fully homomorphic encryption over the integers. In: Gilbert H (ed.) *Advances in Cryptology – EUROCRYPT 2010*. -: Springer Berlin Heidelberg. ISBN 978-3-642-13190-5, 2010; pp. 24–43.
45. Fan J and Vercauteren F. Somewhat practical fully homomorphic encryption. *Cryptology ePrint Archive* Report 2012/144; 2012.
46. Brakerski Z, Gentry C and Vaikuntanathan V. (leveled) fully homomorphic encryption without bootstrapping. In: *Proceedings of the 3rd Innovations in Theoretical Computer Science Conference*, ITCS '12. New York, NY, USA: Association for Computing Machinery. ISBN 9781450311151, 2012; p. 309–325. DOI: 10.1145/2090236.2090262.
47. Brakerski Z and Vaikuntanathan V. Fully homomorphic encryption from ring-lwe and security for key dependent messages. In: Rogaway P (ed.) *Advances in Cryptology – CRYPTO 2011*. -: Springer Berlin Heidelberg. 2011; ISBN 978-3-642-22792-9, pp.505–524.
48. Brakerski Z and Vaikuntanathan V. Efficient fully homomorphic encryption from (standard) lwe. *SIAM J Comput* 2014; 43: 831–871.
49. Armknecht F, Boyd C, Carr C, et al. A guide to fully homomorphic encryption. *IACR Cryptol ePrint Arch* 2015; 2015: 1192.
50. Yousuf H, Lahzi M, Salloum SA, et al. *Systematic Review on Fully Homomorphic Encryption Scheme and Its Application*. Cham: Springer International Publishing, 2021.
51. Marcolla C, Sucasas V, Manzano M, et al. Survey on fully homomorphic encryption, theory, and applications. *Proceedi of the IEEE* 2022; 110: 1572–1609.
52. Rivest RL, Shamir A and Adleman L. A method for obtaining digital signatures and public-key cryptosystems. *Commun ACM* 1978; 21: 120–126.
53. Elgamal T. A public key cryptosystem and a signature scheme based on discrete logarithms. *IEEE Trans Inform Theory* 1985; 31: 469–472.
54. Paillier P. Public-key cryptosystems based on composite degree residuosity classes. In: Stern J (ed.) *Advances in Cryptology — EUROCRYPT '99*. Berlin, Heidelberg: Springer Berlin Heidelberg; 1999. ISBN 978-3-540-48910-8, pp. 223–238.
55. Ryu J, Kim K and Won D. A study on partially homomorphic encryption. In: *2023 17th International Conference on Ubiquitous Information Management and Communication (IMCOM)*. 2023; pp. 1–4. DOI: 10.1109/IMCOM56909.2023.10035630.
56. Diffie W and Hellman M. New directions in cryptography. *IEEE Trans Inform Theory* 1976; 22: 644–654.
57. Hankerson D and Menezes A. *Elliptic Curve Cryptography*. Berlin, Heidelberg: Springer Berlin Heidelberg, 2019. ISBN 978-3-642-27739-9, pp. 1–4. DOI: 10.1007/978-3-642-27739-9_245-2.
58. Seroussi G. Elliptic curve cryptography. In: *1999 Information Theory and Networking Workshop (Cat. No.99EX371)*. 1999; pp.41–. DOI: 10.1109/ITNW.1999.814351.
59. Khan MR, Upreti K, Alam MI, et al. Analysis of elliptic curve cryptography & rsa. *J ICT Standardizat* 2023; 11: 355–378.
60. Saho NJG and Ezin EC. Comparative study on the performance of elliptic curve cryptography algorithms with cryptography through rsa algorithm. 2020.
61. Ullah S, Zheng J, Din N, et al. Elliptic curve cryptography; applications, challenges, recent advances, and future trends: A comprehensive survey. *Compur Sci Rev* 2023; 47: 100530.
62. Chen L, Bassham JM, Foti JW, et al. Recommendations for discrete logarithm-based cryptography: Elliptic curve domain parameters. NIST Special Publication 800-186, National Institute of Standards and Technology (NIST). 2019. DOI: 10.6028/NIST.SP.800-186.
63. Galbraith SD, Wang P and Zhang F. Computing elliptic curve discrete logarithms with improved baby-step giant-step algorithm. *Adv Mathe Commun* 2017; 11: 453–469.
64. Shanks D. Class number, a theory of factorization, and genera. In: *Proceedings of Symposia in Pure Mathematics*, volume 20. American Mathematical Society, 1971; pp.415–440.
65. Bai S and Brent RP. On the efficiency of pollard's rho method for discrete logarithms. In: *Computing: The Australasian Theory Symposium; 2008*.
66. Jindal A, Kumar S and Jatain A. Analysis of pollard rho attacks over ecdlp. In: Verma OP, Wang L, Kumar R and Yadav A (eds.) *Machine Intelligence for Research and Innovations*. Springer Nature Singapore. ISBN 978-981-99-8129-8, 2024; pp.11–21.
67. Larasati HT and Kim H. Quantum cryptanalysis landscape of shor's algorithm for elliptic curve discrete logarithm

problem. In: Kim H (ed.) *Information Security Applications*. Cham: Springer International Publishing; 2021. ISBN 978-3-030-89432-0, pp. 91–104.

68. Roetteler M, Naehrig M, Svore KM, et al. Quantum resource estimates for computing elliptic curve discrete logarithms. In: Takagi T and Peyrin T (eds.) *Advances in Cryptology – ASIACRYPT 2017*. Cham: Springer International Publishing. 2017; ISBN 978-3-319-70697-9, pp.241–270.
69. Shor P. Algorithms for quantum computation: discrete logarithms and factoring. In: *Proceedings 35th Annual Symposium on Foundations of Computer Science*. 1994; pp. 124–134. DOI: 10.1109/SFCS.1994.365700.
70. Hristov-Kalamov N, Fernández-Ruiz R, álvarez Marquina A, et al. Comparison of an accelerated garble embedding methodology for privacy preserving in biomedical data analytics. In: *Artificial Intelligence for Neuroscience and Emotional Systems*. Springer Nature Switzerland; 2024. ISBN 978-3-031-61140-7, pp.282–299.
71. Sejun Kim JJ. (2022) Intel paillier cryptosystem library. https://github.com/intel/pailliercryptolib.
72. Hristov-Kalamov. Homomorphic encrypted id. 2024. https://github.com/niki122121/HomomorphicEncryptedID.
73. Gouert C, Mouris D and Tsoutsos NG. SoK: New insights into fully homomorphic encryption libraries via standardized benchmarks. Cryptology ePrint Archive, Paper 2022/425; 2022.
74. Takeshita J, Koirala N, McKechney C, et al. HEProfiler: An in-depth profiler of approximate homomorphic encryption libraries. Cryptology ePrint Archive, Paper 2024/1059. 2024.
75. Zhu H, Suzuki T and Yamana H. Performance comparison of homomorphic encrypted convolutional neural network inference among helib, microsoft seal and openfhe. In: *2023 IEEE Asia-Pacific Conference on Computer Science and Data Engineering (CSDE)*. 2023; pp.1–7. DOI: 10.1109/CSDE59766.2023.10487709.
76. Dong T and Huang T. Neural cryptography based on complex-valued neural network. *IEEE Trans Neural Netw Learn Syst* 2020; 31: 4999–5004.
77. Gilad-Bachrach R, Dowlin N, Laine K, et al. Cryptonets: Applying neural networks to encrypted data with high throughput and accuracy. In: *Proceedings of The 33rd International Conference on Machine Learning*, *Proceedings of Machine Learning Research*, volume 48. 2016; pp.201–210.