GSHADE: Faster Privacy-Preserving Distance Computation and Biometric Identification

Julien Bringer
Morpho
julien.bringer@morpho.com

Hervé Chabanne
Morpho, Télécom ParisTech
chabanne@telecom-paristech.fr

Mélanie Favre
Morpho
melanie.favre@morpho.com

Alain Patey
Morpho, Télécom ParisTech
patey@telecom-paristech.fr

Thomas Schneider
Engineering Cryptographic Protocols Group
TU Darmstadt, Germany
thomas.schneider@ec-spride.de

Michael Zohner
Engineering Cryptographic Protocols Group
TU Darmstadt, Germany
michael.zohner@ec-spride.de

ABSTRACT

At WAHC'13, Bringer et al. introduced a protocol called SHADE for secure and efficient Hamming distance computation using oblivious transfer only. In this paper, we introduce a generalization of the SHADE protocol, called GSHADE, that enables privacy-preserving computation of several distance metrics, including (normalized) Hamming distance, Euclidean distance, Mahalanobis distance, and scalar product. GSHADE can be used to efficiently compute one-to-many biometric identification for several traits (iris, face, fingerprint) and benefits from recent optimizations of oblivious transfer extensions. GSHADE allows identification against a database of 1 000 Eigenfaces in 1.28 seconds and against a database of 10 000 IrisCodes in 17.2 seconds which is more than 10 times faster than previous works.

General Terms

Algorithms, Security

Keywords

Signal Processing in the Encrypted Domain; Privacy, Biometrics; Oblivious Transfer

1. INTRODUCTION

Secure Two-Party Computation (S2PC), introduced in the eighties by Yao [45] and Goldreich-Micali-Wigderson (GMW) [21] enables two parties to interactively compute a function on their private inputs without revealing any information other than what can be inferred from the function output. A natural field of application for S2PC is privacy-preserving biometric identification, *e.g.,* [3, 5, 9, 10, 18, 28, 35, 38, 41–43]. In this setting, a client $\mathcal{C}$, who holds a fresh biometric sample of a person, and a server $\mathcal{S}$, who holds a database of biometric data, want to determine whether there is a biometric reference for $\mathcal{C}$ in the database of $\mathcal{S}$. $\mathcal{C}$ wants to prevent $\mathcal{S}$ from learning the query sample, since it would allow $\mathcal{S}$ to track the person from which the sample was taken. $\mathcal{S}$, on the other hand, wants to prevent $\mathcal{C}$ from learning information about the contents of his database. We describe more detailed examples later in §1.3.

Privacy-preserving biometric identification has been researched very extensively in the last years. While the first protocols were based on (additively) homomorphic encryption schemes only (e.g., [18, 38]), it was soon demonstrated that protocols which use generic secure computation techniques such as Yao's garbled circuits protocol [45] achieve a better performance and allow the extension to a richer set of functionalities. These protocols either combine homomorphic encryption with generic secure computation protocols, e.g., [3, 5, 25, 28, 41], or exclusively use generic secure computation protocols, e.g., [10, 27, 35].

A recent development in the area of secure computation is the design of efficient protocols that are based on oblivious transfer (OT) [40]. Although efficient constructions for OT have been known for several years, in particular OT extension [30] which allows to base OT on symmetric cryptographic primitives, OT was regarded as an expensive primitive. However, efficient implementations of OT, e.g., [12, 27], have shown that OT can be performed at very low cost and have renewed the interest in protocols using OT. An example can be seen in the field of private set-intersection, where an OT-based solution was presented recently [17] that outperforms prior approaches based on homomorphic encryption [14] and generic secure computation [26]. Another example, from the field of biometric identification, is the SHADE (for Secure HAmming DistancE computation) protocol [9], which allows secure computation of the Hamming distance using OT only. In parallel to these applications of OT, even more efficient OT protocols have been developed recently that further improve the computation and communication complexity of OT extension [1, 32].

IH&MMSec'14, June 11–13, 2014, Salzburg, Austria.

http://dx.doi.org/10.1145/2600918.2600922.

1.1 Our Contributions

In this paper, we build on recent developments in the area of efficient OT and develop a generic framework for secure biometric identification, called GSHADE (Generalized SHADE). GSHADE allows the efficient computation of various distance measures such as the (normalized) Hamming distance, the Euclidean distance, the scalar product, and the Mahalanobis distance using OT only. Furthermore, GSHADE can be combined with generic secure computation techniques and hence can be used to efficiently compute a rich set of functionalities that are based on distance measures. We evaluate the efficiency of GSHADE both theoretically and experimentally and compare it to related work.[1] Overall, GSHADE allows a factor of 10 to 20 improvement in runtime compared to the best previous solutions. A brief summary of the runtimes and communication complexities of GSHADE for different biometric identification schemes (described in §4), in combination with the generic secure computation protocol of Golreich-Micali-Wigderson (GMW),[2] is depicted in Tab. 1.

1.2 Setting

In our setting, a server $\mathcal{S}$ and a client $\mathcal{C}$ want to securely compute 1-vs-N biometric identification. The actual inputs and possible outputs are given in Fig. 1. The privacy requirements of S2PC imply that one party does not get more information about the other party's inputs than what can be deduced from its own inputs and outputs. This is formally expressed using a simulation game, see [24] for more details. In this work, we focus on *semi-honest* (also called *passive* or *honest but curious*) adversaries. In this model, privacy of the inputs is ensured against parties that do not cheat but try to infer additional information from the observed messages. In particular, this model guarantees that even an insider that is able to access the communication records of the secure computation is unable to obtain additional information about the inputs. The semi-honest model is sufficient for many applications [6, 7] and allows to construct highly efficient protocols.

1.3 Example Use Cases

We emphasize the importance of our GSHADE protocol for biometric identification by giving a non-exhaustive list of use cases to which our solution can be applied.

Anonymous Biometric Access Control. Our first use case deals with biometric access control to, for instance, a company building. The employer $\mathcal{S}$ wants to ensure that only the registered (and biometrically enrolled) employees are allowed to enter the building. We can use privacy-preserving biometric identification based on our solution to prevent the employer from tracking his employees' activities.

Biometric Anonymous Credentials. In this example, we have three parties: a client $\mathcal{C}$, a service provider $\mathcal{P}$, and a biometric data server holder $\mathcal{S}$. $\mathcal{S}$ can be, for instance, a government that holds a database of all people satisfying a given criterion (*e.g.,* be over 21). To access the services of $\mathcal{P}$, $\mathcal{C}$ identifies against the database of $\mathcal{S}$. If $\mathcal{C}$ was successfully identified, $\mathcal{S}$ gives him a token to present to $\mathcal{P}$ to prove that he fulfills the requirements. In the whole process, $\mathcal{C}$ reveals his identity neither to $\mathcal{P}$ nor to $\mathcal{S}$.

Secure Biometric Database Intersection. In our third example, we consider two law enforcement agencies that want to identify the suspects they have in common or that want to determine whether a suspect is registered in a given database. For privacy and security reasons, the involved parties want to keep the data that is not in the intersection secret. Our solution can be adapted to this use case by letting the client input a list of biometric data.

Inputs:

- Client $\mathcal{C}$ inputs a biometric acquisition X
- Server $\mathcal{S}$ inputs N biometric data items $Y^1, \ldots, Y^N$

Possible Outputs (given to $\mathcal{S}$ and/or $\mathcal{C}$):

- A yes/no answer (Is X close enough to one of the Y^is?)
- The index and/or distance of the closest match
- The Y^is that are sufficiently close to X
- An identification score
- All distances between X and the Y^is
- ...

Figure 1: Secure Two-party Computation of Biometric Identification.

1.4 Outline

The remainder of this paper is organized as follows. We describe preliminaries for this work in §2, including main techniques for S2PC, the state of the art in privacy-preserving biometric identification, and the original SHADE protocol. We introduce GSHADE, our generalization of the SHADE protocol for computing several distance metrics, in §3 and describe its applications to biometric identification in §4. We give implementation results and compare the performance of GSHADE to the state of the art in §5 and conclude in §6.

2. PRELIMINARIES

In this section, we summarize the properties of and techniques used for secure two-party computation (§2.1), distance metrics for biometric identification (§2.2), the state of the art in privacy-preserving biometric identification (§2.3), and the original SHADE protocol (§2.4). More details on S2PC can for instance be found in [24], while a deeper study of its application to biometric identification can be found in [8].

2.1 Secure Two-Party Computation (S2PC)

Several techniques can be applied to realize Secure Two-Party Computation (S2PC), most prominently Yao's garbled circuits protocol [45] and the protocol of Goldreich-Micali-Wigderson (GMW) [21] that both use oblivious transfer [19, 40]; or alternatively (additively) homomorphic en-

[1] Our GSHADE implementation is available online at `http://encrypto.de/code/GSHADE`.

[2] Alternatively, Yao's garbled circuits protocol could be used.

Application	SCiFI Faces [38]	IrisCodes [16]	FingerCodes [31]	Eigenfaces [44]
Distance Computation using GSHADE				
Metric	Hamming Distance	Normalized Hamming Distance	Euclidean Distance	Scalar Product + Euclidean Distance
Time in sec. (LAN/WiFi)	0.9 / 1.4	8.8 / 14.3	5.1 / 10.3	1.0 / 2.8
Communication in MB	4.3	51.3	48.9	15.1
Post-Processing using GMW				
Method	Comparison	Comparison	Closest Match	Closest Match
Time in sec. (LAN/WiFi)	0.09 / 0.27	0.3 / 1.4	1.6 / 4.1	4.0 / 13.1
Communication in MB	1.9	5.1	18.6	68.5

Table 1: Empirical performance of GSHADE for 1-vs-5 000 biometric identification schemes. Details on the choice of parameters are given in Tab. 2.

cryption, e.g., [13]. In the following, we give a short summary of each of these techniques.

Oblivious Transfer. A 1-out-of-2 oblivious transfer (OT) [19,40], denoted by OT^ℓ, is a two-party protocol where one party (the sender) inputs two ℓ-bit strings $x_0, x_1 \in \{0,1\}^\ell$ and the other party (the receiver) inputs a bit b. At the end of the protocol, the receiver obtains x_b but learns no information about x_{1-b} whereas the sender learns no information about b. OT protocols can be built from public key cryptography, e.g., [36]. For a large number of OTs, OT extension [30] can be used that extends a few base OTs to many OTs using only efficient symmetric cryptographic primitives. Recent work of [32] further improved the communication complexity of OT extension and [1] provides even more efficient protocols for the correlated OT functionality, where the sender inputs only a single value Δ together with a correlation function f s.t. at the end of the protocol, the sender obtains $x_0 \in_R \{0,1\}^\ell$ and $x_1 = f_\Delta(x_0)$ as output and the receiver obtains x_b.

Yao's Garbled Circuits Protocol. A garbled circuit [45] is an encrypted version of the binary circuit representing the function to be evaluated securely. In Yao's protocol, one party (the sender) generates the garbled circuit by building the binary circuit, choosing a pair of encryption keys for every wire of the circuit, and encrypting the output wire keys using the keys of the input wires. The sender then sends the garbled circuit and the input keys that correspond to his inputs to the second party (the receiver). The receiver obtains the keys corresponding to his inputs by engaging in an oblivious transfer with the sender. Using the obtained input keys, the receiver can then decrypt the garbled circuit to obtain the result while learning no intermediary information. See [27] for a more detailed description.

Yao's protocol relies mostly on symmetric cryptography and is best suited for functions that can efficiently be represented as binary circuits and in environments that have a high communication latency. However, Yao's protocol has a high communication complexity and requires the function and input sizes to be known in advance to allow precomputation. Yao's garbled circuits protocol has been implemented in the FastGC framework [27].

GMW Protocol. Similar to Yao's protocol, the GMW protocol [21] also uses a binary circuit representation of the function, but performs the secure evaluation on shares rather than using encrypted gates. The parties first secret-share their inputs using a XOR secret sharing scheme. To evaluate an XOR gate, the parties simply XOR the shares of the input wires. To evaluate an AND gate, the parties perform an oblivious transfer, where one party pre-computes all possible outputs of the gate and the other party obliviously obtains the output that corresponds to its input shares. To obtain the output of the circuit, the parties exchange the shares of the output wires.

As shown in [1,12,42], the GMW protocol allows the precomputation of all symmetric cryptographic operations before the function or the inputs to the function are known and requires less communication per AND gate than Yao's garbled circuits protocol. However, the GMW protocol requires a number of communication rounds that is linear in the depth of the circuit. The GMW protocol has been implemented in [12] and further optimized for the two-party case in [1,42].

Homomorphic Encryption. A public-key encryption is homomorphic if it is possible to compute over encrypted data without the knowledge of the secret key. Although fully homomorphic encryption (*i.e.,* a cryptosystem that is homomorphic for any operation) has been introduced in 2009 [20], it is not yet practical. Most implemented proposals therefore use additively homomorphic encryption schemes, such as Paillier [39] or Damgård-Geisler-Krøigaard (DGK) [15].

Homomorphic encryption is more suited for arithmetic circuits and the ciphertexts can be re-used for several instances of secure computation, which reduces the communication complexity. However, homomorphic encryption requires computationally expensive public-key operations that scale very inefficiently for larger security parameters.

2.2 Distance Metrics

In the following, we summarize some distance metrics that are used in biometric identification schemes. In §2.3 we will describe which distance is used by which biometric identification scheme and in §3 we will show that each of these distances can be computed efficiently with our generalized SHADE protocol.

Hamming Distance (HD). The Hamming distance between two ℓ-bit vectors $X = (x_1, \ldots, x_\ell)$ and $Y = (y_1, \ldots, y_\ell)$ is computed as $\text{HD}(X,Y) = \sum_{i=1}^{\ell} x_i \oplus y_i$.

Normalized Hamming Distance (NHD). The normalized Hamming distance between a ℓ-bit vector $X = (x_1, \ldots, x_\ell)$ with ℓ-bit mask $M = (m_1, \ldots, m_\ell)$ and a vector $Y = (y_1, \ldots, y_\ell)$ with mask $M' = (m'_1, \ldots, m'_\ell)$ is computed as $\text{NHD}(X, M; Y, M') = \frac{\sum_{i=1}^{\ell}(m_i m'_i (x_i \oplus y_i))}{\sum_{i=1}^{\ell}(m_i m'_i)}$.

Scalar Product (SP). The scalar product between two K-dimensional vectors $X = (X_1, \ldots, X_K)$ and $Y = (Y_1, \ldots, Y_K)$ is computed as $\text{SP}(X,Y) = \sum_{i=1}^{K} X_i Y_i$.

Squared Euclidean Distance (ED). The squared Euclidean distance between two K-dimensional vectors $X = (X_1, \ldots, X_K)$ and $Y = (Y_1, \ldots, Y_K)$ is computed as $\mathrm{ED}(X,Y) = \sum_{i=1}^{K}(X_i - Y_i)^2 = \sum_{i=1}^{K}((X_i)^2 - 2X_iY_i + (Y_i)^2)$.

Squared Mahalanobis Distance (MD). The squared Mahalanobis distance between two K-dimensional vectors $X = (X_1, \ldots, X_K)$ and $Y = (Y_1, \ldots, Y_K)$ is computed as $\mathrm{MD}(X,Y) = (X-Y)^T M (X-Y)$, where M is a positive semi-definite matrix (which might be the inverse of the covariant matrix of a sample set). The Mahalanobis distance can be used for instance for hand shape, keystroke, or signature recognition [34].

2.3 Privacy-Preserving Biometric Identification

Several different schemes for privacy-preserving biometric identification using S2PC have been proposed. Most schemes focused on face [18, 38, 41], fingerprint [3, 5, 28, 43], or iris [5, 10, 35] recognition which we summarize next. We provide more details on the underlying algorithms in §4.

Privacy-preserving face recognition. Privacy-preserving face recognition has been realized based on two different recognition algorithms: Eigenfaces used in [18,41,42] and the SCiFI algorithm used in [9, 27, 38, 42].

In protocols based on the Eigenfaces algorithm [44], the parties have to perform a projection (matrix-vector or scalar products), compute the Euclidean distance, and compare the resulting distance to a threshold. Erkin et al. [18] suggest to employ additively homomorphic encryption for the whole protocol. Sadeghi et al. [41] showed that a hybrid solution gives better performances, using additively homomorphic encryption for projection and distance computation, then garbled circuits for comparisons. Schneider et al. [42] use GMW, which allows to pre-compute all cryptographic operations and thereby achieves a fast online phase.

The SCiFI algorithm [38] is a face recognition algorithm that is based on the Hamming distance and was specifically designed to yield an efficient privacy-preserving protocol. Originally, Osadchy et al. [38] used additively homomorphic encryption and subsequently Huang et al. [27] and Schneider et al. [42] showed that using Yao's garbled circuits respectively GMW results in better performances. The SHADE protocol of Bringer et al. [9] is an even more efficient construction based on oblivious transfer (cf. §2.4 for details).

Privacy-preserving fingerprint recognition. Secure fingerprint recognition has been considered using two main solutions. The FingerCodes technique [31] relies on Euclidean distance and has been proposed in [3,5,28], which use additively homomorphic encryption for Euclidean distance and several solutions for comparison/identification operations. Use of minutiae-based fingerprint recognition [34] has been envisioned in [5,43], but we do not further discuss it in this paper as it does not fit our protocol.

Privacy-preserving iris recognition. Iris recognition using IrisCodes [16] requires secure evaluation of normalized Hamming distances and has first been considered by Blanton et al. [5] using homomorphic encryption, then by Luo et al. [35] and Bringer et al. [10] using Yao's garbled circuits.

2.4 Secure Hamming Distance Computation (SHADE)

The SHADE protocol [9] allows efficient secure Hamming distance computation using oblivious transfer. In the following we describe the original SHADE protocol and its extension to the 1-vs-N case.

The SHADE Protocol. The SHADE protocol was first intended for secure computation of Hamming distances. For $\mathcal{S}$ and $\mathcal{C}$ with ℓ-bit inputs Y and X the protocol works as follows. $\mathcal{S}$ and $\mathcal{C}$ perform ℓ $\mathrm{OT}^{\lceil \log_2(\ell+1) \rceil}$ where, in the i-th OT, $\mathcal{S}$ chooses a random $r_i \in_R \mathbb{Z}_{\ell+1}$ and inputs $(r_i + y_i, r_i + (y_i \oplus 1))$ and $\mathcal{C}$ inputs y_i as choice bit and receives $t_i = r_i + (x_i \oplus y_i)$. $\mathcal{S}$ then sums up the random masks and outputs $R = \sum_{i=1}^{\ell} r_i$ and $\mathcal{C}$ sums up the received values and outputs $T = \sum_{i=1}^{\ell} t_i$. Note that we have $T - R = \sum_{i=1}^{\ell}(r_i + (x_i \oplus y_i)) - \sum_{i=1}^{\ell} r_i = \sum_{i=1}^{\ell} x_i \oplus y_i = \mathrm{HD}(X,Y)$.

SHADE for the 1-vs-N Case. SHADE was observed to be efficiently extendable to the 1-vs-N case, where $\mathcal{S}$ holds N ℓ-bit values $Y^1, \ldots, Y^N$ and $\mathcal{C}$ holds a single ℓ-bit value X. The only additional overhead for the extended protocol is longer bit strings in the oblivious transfers. More detailed, in the i-th OT, the parties perform ℓ $\mathrm{OT}^{N\lceil \log_2(\ell+1) \rceil}$ where $\mathcal{S}$ inputs $(r_i^1 + x_i^1 || \ldots || r_i^N + x_i^N, r_i^1 + \bar{x}_i^1 || \ldots || r_i^N + \bar{x}_i^N)$ and $\mathcal{C}$ inputs y_i and receives $t_i = (r_i^1 + (x_i^1 \oplus y_i) || \ldots || r_i^N + (x_i^N \oplus y_i))$. In the final step, the parties can again simply compute and output $R^1, \ldots, R^N$ and $T^1, \ldots, T^N$, where $R^b = \sum_{i=1}^{\ell} r_i^b$ and $T^b = \sum_{i=1}^{\ell} t_i^b$, for $1 \le b \le N$.

3. OUR GENERALIZED SHADE (GSHADE) PROTOCOL

In this section we describe our generalized SHADE protocol, called GSHADE, which allows to compute different distances (§3.1). We describe how to combine GSHADE with comparison or minimum protocols (§3.2), outline how to efficiently extend it to 1-vs-N matching (§3.3) and how to base it on the more efficient correlated OT functionality (§3.4). We give applications of GSHADE to biometric identification with new adaptations for IrisCodes and Eigenfaces later in §4.

3.1 The GSHADE Protocol

We observe that the original SHADE protocol extends to the family $\mathcal{F}^{GSHADE}$ of functions that can be expressed as $f(X,Y) = f_X(X) + \Sigma_{i=1}^{n} f_i(x_i, Y) + f_Y(Y)$, where $X = (x_1, \ldots, x_n) \in \{0,1\}^n$ is the input of $\mathcal{C}$ and Y is the input of $\mathcal{S}$. (The set $\mathtt{S}$ to which Y belongs does not impact the protocol.) In particular, several metrics used for biometric matching are included in this family of functions:

Hamming Distance $X = (x_1, \ldots, x_\ell)$ and $Y = (y_1, \ldots, y_\ell)$ are $n = \ell$-bit vectors. We have $f_X = f_Y = 0$ and $f_i(x_i, Y) = x_i \oplus y_i$, for $i = 1, \ldots, n$.

Scalar Product $X = (X_1, \ldots, X_K)$ with $X_i = (x_{K(i-1)+1}, \ldots, x_{K(i-1)+\ell})$ and $Y = (Y_1, \ldots, Y_K)$ with $Y_i = (y_{K(i-1)+1}, \ldots, y_{K(i-1)+\ell})$ are $n = K \times \ell$-bit-integer vectors. We have $f_X = f_Y = 0$ and $f_{K\cdot(i-1)+j}(x_{K(i-1)+j}, Y) = 2^{j-1} \cdot x_{K(i-1)+j} \cdot Y_i$, for $i = 1, \ldots, K$ and $j = 1, \ldots, \ell$.

Squared Euclidean Distance $X = (X_1, \ldots, X_K)$ with $X_i = (x_{K(i-1)+1}, \ldots, x_{K(i-1)+\ell})$ and $Y = (Y_1, \ldots, Y_K)$ with $Y_i = (y_{K(i-1)+1}, \ldots, y_{K(i-1)+\ell})$ are $n = K \times \ell$-bit-integer vectors. We have $f_X(X) = \Sigma_{i=1}^{K}(X_i)^2$, $f_Y(Y) = \Sigma_{i=1}^{K}(Y_i)^2$ and $f_{K\cdot(i-1)+j}(x_{K(i-1)+j}, Y) = -2^j \cdot x_{K(i-1)+j} \cdot Y_i$, for $i = 1, \ldots, K$ and $j = 1, \ldots, \ell$.

Squared Mahalanobis Distance We assume that $M = (M_{uv})_{u,v=1,\ldots,K}$ is known by both parties, and not a private input of either party.[3] $X = (X_1, \ldots, X_K)$ with $X_i = (x_{K(i-1)+1}, \ldots, x_{K(i-1)+\ell})$ and $Y = (Y_1, \ldots, Y_K)$ with $Y_i = (y_{K(i-1)+1}, \ldots, y_{K(i-1)+\ell})$ are $n = K \times \ell$-bit-integer vectors. We have $f_X(X) = X^T M X$, $f_Y(Y) = Y^T M Y$, $f_{K\cdot(i-1)+j}(x_{K(i-1)+j}, Y) = -2^j \cdot x_{K(i-1)+j} \cdot \sum_{v=1}^{K} M_{i,v} Y_v$, for $i = 1, \ldots, K, j = 1, \ldots, \ell$.

In Fig. 2, we describe the generalized SHADE protocol. Note that m is such that the output of $f(X, Y)$ belongs to $\mathbb{Z}_m$. For instance, if f is the Hamming distance between two ℓ-bit vectors, we set $m = \ell + 1$.

Inputs:

- $\mathcal{C}$ inputs a n-bit string $X = (x_1, \ldots, x_n)$
- $\mathcal{S}$ inputs $Y \in \mathtt{S}$

Outputs:

- $\mathcal{S}$ obtains $R \in_R \mathbb{Z}_m$
- $\mathcal{C}$ obtains $T = R + f(X, Y)$

Protocol:

1. $\mathcal{S}$ chooses n random values $r_1, \ldots, r_n \in_R \mathbb{Z}_m$.
2. For each $i = 1, \ldots, n$, $\mathcal{S}$ and $\mathcal{C}$ engage in a $\mathrm{OT}^{\lceil \log_2(m) \rceil}$ where
   - $\mathcal{S}$ acts as the sender and $\mathcal{C}$ as the receiver.
   - $\mathcal{C}$'s selection bit is x_i.
   - $\mathcal{S}$'s input is $(r_i + f_i(0, Y), r_i + f_i(1, Y))$.
   - The output obtained by $\mathcal{C}$ is consequently $t_i = r_i + f_i(x_i, Y)$.
3. $\mathcal{C}$ computes and outputs $T = \Sigma_{i=1}^{n} t_i + f_X(X)$.
4. $\mathcal{S}$ computes and outputs $R = \Sigma_{i=1}^{n} r_i - f_Y(Y)$.

Figure 2: Generalized SHADE (GSHADE) protocol.

Correctness. Since $r_1, \ldots, r_n$ are picked uniformly at random over $\mathbb{Z}_m$, then, for fixed X and Y, $R = \sum_{i=1}^{n} r_i - f_Y(Y)$ is distributed uniformly over $\mathbb{Z}_m$ and the output of $\mathcal{S}$ is correct. Moreover, we have $T - R = \sum_{i=1}^{n}(t_i - r_i) + f_X(X) + f_Y(Y) = \sum_{i=1}^{n} f_i(x_i, Y) + f_X(X) + f_Y(Y) = f(X, Y)$. Thus, $T = R + f(X, Y)$ and the output of $\mathcal{C}$ is correct. □

Security. The proof of security of GSHADE is similar to that of SHADE [9]. We give a proof sketch against static semi-honest adversaries next. Security is proven by simulation in the OT-hybrid setting, where OTs are simulated by a trusted oracle. We recall that each simulator is provided with the input and output of the corrupted party.
Case 1 – $\mathcal{S}$ is corrupted. Since $\mathcal{S}$ receives no messages beyond those in OT, its view can be perfectly simulated.
Case 2 – $\mathcal{C}$ is corrupted. Given $\mathcal{C}$'s output T and input X, $\mathcal{C}$'s view can be perfectly simulated by sending random values $t'_1, \ldots, t'_{n-1} \in_R \mathbb{Z}_m$ and $t'_n = T - \sum_{i=1}^{n-1} t'_i - f_X(X)$ to $\mathcal{C}$ in the OTs. □

[3]This assumption is reasonable, for instance, if MD is used instead of ED in the Eigenfaces protocol (see §4.4). Indeed, the matrix M would only disclose statistical information about the projection space, which is not very sensitive, whereas the Eigenfaces basis gives information about real biometric data (they can reveal "average" faces) and should be kept private, as it is the case in our protocol.

3.2 Adding Comparison or Minimum

In some use-cases of privacy-preserving biometric identification, it is required that the parties learn whether the distance is lower than a certain threshold (comparison) or the index of the closest match (minimum). For these protocols, we require a secure comparison or minimum operation after the distance calculation, which keeps the resulting distance secret. Using GSHADE does not improve comparison or minimum operations. However, if one runs GSHADE, the masked results can easily be used as input to a secure comparison or minimum protocol. Several protocols are possible, depending on the actual desired output. We refer the reader to the papers mentioned in §2.3 for an overview. Note that for Yao's garbled circuits protocol and the GMW protocol, we have to build a circuit which first reconstructs $f(X, Y) = T - R$ and subsequently computes the desired functionality. In our experiments in §4 we use the GMW protocol for the comparison or minimum operations.

3.3 Adaptation to the 1-vs-N case

Analogue to the original SHADE protocol, the GSHADE protocol can be efficiently extended to the 1-vs-N biometric identification, where the client has one input X and the server has N inputs $Y^1, \ldots, Y^N$ and they want to compute all the distances $f(X, Y^b)$, for $b = 1, \ldots, N$. The protocol is modified in the following way

1. $\mathcal{S}$ generates $n \cdot N$ random values $r_{b,i} \in_R \mathbb{Z}_m$, for $b = 1, \ldots, N$.
2. For each $i = 1, \ldots, n$, $\mathcal{S}$ and $\mathcal{C}$ engage in a $\mathrm{OT}^{N\lceil \log_2(m) \rceil}$ where
   - $\mathcal{S}$ acts as the sender and $\mathcal{C}$ as the receiver.
   - $\mathcal{C}$'s selection bit is x_i.
   - $\mathcal{S}$'s input is $(r_{1,i} + f_i(0, Y^1) || \ldots || r_{N,i} + f_i(0, Y^N)$, $r_{1,i} + f_i(1, Y^1) || \ldots || r_{N,i} + f_i(1, Y^N))$.
   - The output obtained by $\mathcal{C}$ is $(t_{1,i} || \ldots || t_{N,i}) = (r_{1,i} + f_i(x_i, Y^1) || \ldots || r_{N,i} + f_i(x_i, Y^N))$.
3. $\mathcal{C}$ computes and outputs $T^1 = \Sigma_{i=1}^{\ell} t_{1,i} + f_X(X), \ldots, T^N = \Sigma_{i=1}^{\ell} t_{N,i} + f_X(X)$
4. $\mathcal{S}$ computes and outputs $R^1 = \Sigma_{i=1}^{n} r_{1,i} - f_Y(Y^1), \ldots, R^N = \Sigma_{i=1}^{n} r_{N,i} - f_Y(Y^N)$

Note that the number n of OTs remains unchanged compared to the 1-vs-1 case and only the length of the inputs grows linearly with the number of database entries N. As the protocol is essentially a parallel execution of the basic GSHADE protocol, but using OTs with longer strings, correctness and security carry over from GSHADE (cf. §3.1).

3.4 Using Correlated OTs

As described in §2.1, the *correlated* OT (C-OT) extension protocol of [1] has an even lower communication complexity than generic OT extension. Here, the sender obtains one randomly chosen value as output and inputs a correlation that determines the second value. This functionality was initially used for Yao's protocol with the free-XOR technique [33] where for each wire w one key k_w^0 is chosen randomly and the other key is correlated with $k_w^1 = k_w^0 \oplus \Delta$, where Δ is a fixed offset. In the following we show how GSHADE can be based on C-OT instead of OT. Here we assume that m is a power of 2 (we discuss the case where m is not a power of 2 in Appendix A). The GSHADE protocol can be rewritten as follows:

1. For each $i = 1, \ldots, n$, $\mathcal{S}$ and $\mathcal{C}$ engage in a C-OT$^{\lceil \log_2(m) \rceil}$ where
   - $\mathcal{C}$ acts as the receiver with selection bit x_i.
   - $\mathcal{S}$ acts as the sender with input $\Delta_i = f_i(1, Y) - f_i(0, Y)$.
   - The correlation function is $f_{\Delta_i}(\cdot) = \cdot + \Delta_i$.
   - The output obtained by $\mathcal{S}$ is $\rho_i \in_R \mathbb{Z}_m$.
   - The output obtained by $\mathcal{C}$ is $\tau_i = \rho_i - f_i(0, Y) + f_i(x_i, Y)$.
2. $\mathcal{C}$ computes and outputs $T = \sum_{i=1}^{n} \tau_i + f_X(X)$.
3. $\mathcal{S}$ computes and outputs $R = \sum_{i=1}^{n} (\rho_i - f_i(0, Y)) - f_Y(Y)$.

Correctness. During the i^{th} C-OT, $\mathcal{C}$ obtains $\rho_i = \rho_i - f_i(0, Y) + f_i(0, Y)$ if $x_i = 0$, or $\rho_i + \Delta_i = \rho_i + f_i(1, Y) - f_i(0, Y)$ if $x_i = 1$. Thus, $\mathcal{C}$ always obtains $\tau_i = \rho_i + f_i(x_i, Y) - f_i(0, Y)$. Regarding final outputs, $T - R = \sum_{i=1}^{n} (\tau_i - \rho_i + f_i(0, Y)) + f_X(X) + f_Y(Y) = \sum_{i=1}^{n} (\rho_i + f_i(x_i, Y) - f_i(0, Y) - \rho_i + f_i(0, Y)) + f_X(X) + f_Y(Y) = \sum_{i=1}^{n} f_i(x_i, Y) + f_X(X) + f_Y(Y) = f(X, Y)$. Thus, $R \in_R \mathbb{Z}_m$ and $T = R + f(X, Y)$ and the protocol is correct. □

Security. Security is proven in a similar way as for the OT-based GSHADE protocol described in §3.1. We still give a proof sketch against static semi-honest adversaries. Here, we assume that C-OTs are simulated by a trusted oracle.

Case 1 – $\mathcal{S}$ is corrupted. Given $\mathcal{S}$'s output R and input Y, $\mathcal{S}$'s view can be perfectly simulated by sending random values $\rho'_1, \ldots, \rho'_{n-1} \in_R \mathbb{Z}_m$ and $\rho'_n = R - \sum_{i=1}^{n-1} \rho'_i + \sum_{i=1}^{n} f_i(0, Y) + f_Y(Y)$ to $\mathcal{C}$ in the C-OTs.

Case 2 – $\mathcal{C}$ is corrupted. Given $\mathcal{C}$'s output T and input X, $\mathcal{C}$'s view can be perfectly simulated by sending random values $\tau'_1, \ldots, \tau'_{n-1} \in_R \mathbb{Z}_m$ and $\tau'_n = T - \sum_{i=1}^{n-1} \tau'_i - f_X(X)$ to $\mathcal{C}$ in the C-OTs. □

This adaptation is also compatible with the 1-vs-N version of GSHADE described in §3.3. Using the C-OT extension protocol of [1] allows to reduce the asymptotic communication complexity by a factor of two compared to using the original OT extension of [30], cf. [1].

4. APPLICATIONS

We demonstrate several applications where the GSHADE protocol can be used for secure and efficient distance computations: the SCiFI (§4.1) and Eigenfaces (§4.4) protocol for face recognition, the IrisCodes (§4.2) protocol, and the FingerCodes (§4.3) protocol. In Tab. 2 we summarize the parameters for the distances used, the number of OTs n, and the length of the OTs' outputs. Note that the parameters in Tab. 2 include optimizations proposed in previous works.

4.1 SCiFI

In the setting of face recognition using SCiFI [38], biometric vectors are $\ell = 900$-bit binary vectors that are compared using Hamming distance. One can simply apply the original SHADE protocol of [9] which is a special case of our GSHADE protocol. The authors of [38] point out that the Hamming distances in the SCiFI protocol never exceeded 180. Thus, we decrease the range of Hamming distances from $\mathbb{Z}_{901}$ to $\mathbb{Z}_{181}$. Therefore, in case of SCiFI, we have to perform $n = \ell = 900$ OTs on $\lceil \log_2 181 \rceil N = 8N$-bit strings.

4.2 IrisCodes

IrisCodes [16] are 512-byte representations of iris images made of a template and a mask of $\ell = 2048$-bit each. The mask signifies reliable bits of the iris template, *i.e.,* a mask bit set to 1 indicates that the corresponding template bit is reliable, while a mask bit set to 0 indicates an erasure (due to eyelids, eyelashes, blurs,...). IrisCodes can be compared using normalized Hamming distance (NHD).

One can see that normalized Hamming distance does not exactly match the family $\mathcal{F}^{GSHADE}$. However, if we adopt the convention that a template bit which is associated to a 0 in the mask is also set to 0, which is not restrictive, numerator (*num*) and denominator (*den*) of normalized Hamming distance both match the family. We integrate both template and mask to the input vector. Let $n = 2\ell$, $f^{num}(X, Y) = \sum_{i=1}^{\ell} x_{\ell+i} \cdot y_{\ell+i} \cdot (x_i \oplus y_i)$ and $f^{den}(X, Y) = \sum_{i=1}^{\ell} x_{\ell+i} \cdot y_{\ell+i}$. For $i = 1, \ldots, \ell$, let $f_i^{den}(x_i, Y) = 0$, $f_{i+\ell}^{den}(x_{i+\ell}, Y) = x_{i+\ell} \cdot y_{i+\ell}$ and let f_i^{num} and $f_{i+\ell}^{num}$ be defined as in Tab. 3. Our convention enforces that $(x_{i+\ell} = 0 \implies x_i = 0)$ (and same for Y), for each $i = 1, \ldots, \ell$. If both inputs X and Y respect this convention, then one can easily check that $f^{num}(X, Y) = \sum_{i=1}^{n} f_i^{num}(x_i, Y)$ and $f^{den}(X, Y) = \sum_{i=1}^{n} f_i^{den}(x_i, Y)$.

y_i	0		1	
$y_{i+\ell}$	0	1	0	1
$f_i^{num}(0, Y)$	0	0	–	1
$f_i^{num}(1, Y)$	0	1	–	0
$f_{i+\ell}^{num}(0, Y)$	0	0	–	−1
$f_{i+\ell}^{num}(1, Y)$	0	0	–	0

Table 3: Definition of f_i^{num} and $f_{i+\ell}^{num}$, for $i \in [1, \ell]$.

Outputting the numerator and the denominator does not satisfy complete privacy requirements, if one wants to securely evaluate the normalized Hamming distance. Nevertheless, our goal is to apply GSHADE to biometric identification. Thus, instead of outputting NHD, we output the result of $\text{NHD}(X, Y) \stackrel{?}{<} t$, which can be rewritten as $f^{num}(X, Y) \stackrel{?}{<} t \cdot f^{den}(X, Y)$, where $0 < t < 1$ is a threshold. Thus, one runs GSHADE on both f^{num} and f^{den}. $\mathcal{C}$ obtains $T^{num} = f^{num}(X, Y) + R^{num}$ and $T^{den} = f^{den}(X, Y) + R^{den}$ while $\mathcal{S}$ holds masks R^{num} and R^{den} (see Fig. 2). If t is known by $\mathcal{C}$, then $\mathcal{C}$ includes T^{num} and $t \cdot T^{den}$ and $\mathcal{S}$ inputs R^{num} and $t \cdot R^{den}$ to a protocol that first pairwise subtracts inputs and then compares the results. If t is not known by $\mathcal{C}$,

Protocol	Operation	n	$\lceil \log_2(m) \rceil$
SCiFI [38]	Hamming Distance	900	8
IrisCodes [5]	Normalized Hamming Distance	$2\,048 + 2\,048$	$31 + 11$
FingerCodes [28]	Euclidean Distance	$640 \times 8 = 5\,120$	16
Eigenfaces (projection) [18, 25, 41, 42]	Scalar Product	$10\,304 \times 8 = 82\,432$	$12 \times 30 = 360$
Eigenfaces (distance) [18, 25, 41, 42]	Euclidean Distance	$12 \times 30 = 360$	50

Table 2: Parameters used in our experiments (n: number of OTs, $\mathbb{Z}_m$: range of OT inputs).

this secure protocol should first include a secure multiplication step or GSHADE should be run on $t \cdot f^{den}$.

When actually running this protocol, we suggest to run n OTs for both f^{num} and f^{den}, where the first ℓ OTs only concern f^{num} while the last ℓ OTs concatenate contributions to both f^{num} and f^{den}. Thus, complexity (before comparison) is $\ell \times \mathrm{OT}^{\lceil \log_2(\ell) \rceil} + \ell \times \mathrm{OT}^{2\lceil \log_2(\ell) \rceil}$ in the 1-vs-1 case or $\ell \times \mathrm{OT}^{N\lceil \log_2(\ell) \rceil} + \ell \times \mathrm{OT}^{2N\lceil \log_2(\ell) \rceil}$ in the 1-vs-N case.

4.3 FingerCodes

Fingerprint recognition via FingerCodes [31] uses a global representation (unlike the more standard minutiae-based recognition protocols that describe local features) of biometric data as integer vectors. Comparison between two biometric data samples is then simply done using Euclidean distance. As mentioned in §3.1, one can directly apply GSHADE to securely evaluate this metric. In general, without experimental optimizations, one has to perform $n = K\ell$ OTs on $N(2\ell + \lceil \log_2(K) \rceil)$-bit inputs. Two different sets of parameters for FingerCodes were suggested: [3, 5] use $K =$ 16-dimensional vectors of $\ell = 7$-bit elements and perform the comparison on 19-bit results while [28] uses $K = 640$-dimensional vectors of $\ell = 8$-bit elements and performs the comparison on 16-bit results. Consequently, for the parameters of [3, 5] we have to perform $n = K\ell = 112$ OTs on $19N$-bit strings, and for the parameters of [28] we have to perform $n = 5\,120$ OTs on $16N$-bit strings. For our experiments in Tab. 1 we choose the parameters of [28].

4.4 Eigenfaces

In the setting of face recognition using Eigenfaces [44], the client holds a face image $X = (x_1, \ldots, x_{K'})$ with ℓ-bit elements and the server holds an average face image $\Psi = (\psi_1, \ldots, \psi_{K'})$, a set of Eigenfaces $(U^1, \ldots, U^K)$, with $U^j = (u_1^j, \ldots, u_{K'}^j)$ for $j = 1, \ldots, K$, and a database of N projected faces $Y^1, \ldots, Y^N$. The identification protocol consists of three phases:

1. Projection: The average face image Ψ is subtracted from X. The result is projected on the Eigenfaces basis. Thus, one gets $\bar{X} = (\bar{x}_1, \ldots, \bar{x}_K)$, with $\bar{x}_j = \mathrm{SP}(X - \Psi, U^j)$, for $j = 1, \ldots, K$.
2. Distance: The Euclidean distances $d_j = \mathrm{ED}(\bar{X}, Y^i)$ are computed, for $i = 1, \ldots, N$.
3. Comparison: The distances d_i are compared to thresholds and an identification result is output (see §3.2).

We suggest to use our GSHADE protocol to securely compute the first two steps, in the following way:

Projection. The operation consists of a subtraction then a scalar product and thus belongs to the family of functions that can be computed using GSHADE. Since $\mathcal{C}$ uses the same input for all K projections, one can use the 1-vs-K variant described in §3.3. Thus, $\mathcal{C}$ gets $T = \bar{X} + R = (\bar{x}_1 + r_1, \ldots, \bar{x}_K + r_K)$ and $\mathcal{S}$ gets $R = (r_1, \ldots, r_K)$, where the r_is are random masks in $\mathbb{Z}_p$, where $p = 2^{2\ell + \lceil \log_2 K' \rceil}$. Note that here we compute this step as $\bar{x}_j = \mathrm{SP}(X, U^j) - \mathrm{SP}(\Psi, U^k)$, *i.e.,* $-\mathrm{SP}(\Psi, U^j)$ is computed on the server's side as part of f_Y (with notations of §3.1).

Distance. Here $\mathcal{C}$ and $\mathcal{S}$ use the GSHADE protocol a second time in the 1-vs-N variant, where the computed function is $(T; R, Y) \mapsto \mathrm{ED}(T - R, Y)$, which is included in $\mathcal{F}^{GSHADE}$.

For the parameters used in [18,25,41,42], the projection is computed on $K' = 10\,304$-dimensional vectors with $\ell = 8$-bit elements and yields a 30-bit result on a $K = 12$-dimension plane. The Euclidean distance is then computed on $K = 12$-dimensional vectors with 30-bit elements and results in a 50-bit value. Thus, we have to perform 82 432 OTs on 360-bit elements and 360 OTs on $50N$-bit elements. Notice that the cost of the projection phase is independent of the size of the database N.

We emphasize that GSHADE can be applied to other privacy-preserving biometric recognition protocols that follow the same architecture as Eigenfaces, such as Fisherfaces [4], where the difference mostly lies in the algorithm to choose the basis (in the case of Fisherfaces, Linear Discriminant Analysis instead of Principal Component Analysis for Eigenfaces) and thus does not impact the identification protocol.

5. PERFORMANCE EVALUATION

In this section we evaluate the performance of GSHADE. We discuss the use of quantized inputs in §5.1, asymptotic complexities are studied in §5.2 and our experimental results are described and compared to the state of the art in §5.3.

5.1 Quantization

Our GSHADE protocol relies on the fact that inputs are binary vectors, either originally binary or by binarizing vectors of integers. However, it is often the case that the coordinates of feature vectors used for, e.g., face recognition are real or floating point numbers. It must be validated that the same protocols can be used on integer or binary inputs without loss of accuracy. Erkin et al. [18] showed, from experiments on the AT&T [2] database, that integers can be used as inputs to the Eigenfaces protocol (by multiplying original inputs by 1 000) without reliability losses and parameters used in [18] are chosen accordingly.

The performance of GSHADE is directly related to the size of the inputs, but also to the size of the outputs ($\lceil \log_2(m) \rceil$ in Tab. 2). We ran experiments to show that the outputs' size could be further reduced without impacting accuracy. Our biometric experiments have been con-

ducted on the AT&T [2] and Multi-PIE [11, 22] (restricted to frontal images taken using camera 05_1) databases, using the Python Face Recognition Library [23, 29]. We took 30 eigenfaces (instead of 12 for [18]) and showed that the size of the projected faces could be reduced to $30 \times 13 = 390$-bit vectors, which is comparable to the parameters of [18]. However, our analysis showed that squared Euclidean distances can be reduced to 26-bit (for the AT&T database) or 24-bit integers (for the Multi-PIE database), which is about half the size used in [18, 25, 41, 42]. Using these parameters would allow to further decrease the communication complexity of GSHADE by about a factor of two without any loss in correctness, accuracy, or security. However, aiming at fairness in our comparison to other protocols, experiments described in §5.3 were run with the same parameters as in [18, 25, 41, 42].

5.2 Asymptotic Performance Comparison

In the following we compare the asymptotic performance of the GSHADE protocol when computing various distances to related work. The parameters that affect the performance of GSHADE are the number of OTs and the length of strings that are transferred obliviously. In the following, κ is the symmetric security parameter and ρ is the asymmetric security parameter (in our experiments in §5.3 we set $\kappa = 80$ and $\rho = 1\,024$).

In Tab. 4, we summarize the asymptotic computation and communication complexities when computing different distance metrics using GSHADE with those of secure distance computation protocols in related work (§2.3). For each solution, we outline the technique that is used, i.e., homomorphic encryption (HE), garbled circuits (GC), GMW, or GSHADE, the computation complexity of $\mathcal{S}$ (which dominates the one of $\mathcal{C}$) and the overall communication. Note that the complexities only include the distance computation excluding later computation steps such as minimum, greater than, and so on. However, each of the solutions can be extended by a generic secure computation protocol to obtain the desired functionality. We obtain the computation complexity for HE by counting the number of modular exponentiations and for GC, GMW, and GSHADE by counting the number of symmetric cryptographic operations. For GC we determine the communication complexity as the size of the garbled circuit (3κ bits per non-linear gate using the free XOR [33] and garbled row reduction [37] techniques) plus the number of input keys of the server (κ bits per $\mathcal{S}$'s input bit). We neglect the inputs of $\mathcal{C}$ as these do not depend on the database size N. For GMW, the communication complexity is two symmetric keys per non-linear gate (see complexity analysis in [1, 42]). The complexity of GSHADE takes into account the C-OT technique of [1]: communication complexity for $n \times \mathrm{C-OT}^{\ell}$ is $n \times (\ell + \kappa)$ bits and computation complexity is $3n + 2n\ell/o$ symmetric operations, where o is the output size of the PRF (pseudo-random function) used by the C-OT (see [1]). The HE-based protocols of [3, 5, 18, 25, 28, 38] to compute the Hamming distance, scalar product, and Euclidean distance use Paillier's additively homomorphic encryption [39], so ciphertexts have size 2ρ bits. For GSHADE and HE we include in Tab. 4 some communication that does not depend on N, but that might be dominant when N is not too large (e.g., for a few dozens or hundreds of entries).

Computation. For the Hamming distance and normalized Hamming distance, where the HE, GC, and GMW techniques have a computation complexity that is linear in $N\ell$, GSHADE achieves a computation complexity linear in $(N\ell \log_2 \ell)/o$. Considering that the output size of the PRF o is in the order of some hundred bits (e.g., 128, 192, or 256), while $\log_2 \ell$ is much smaller (e.g., about 10 for SCiFI and IrisCodes), the computation complexity of GSHADE is considerably smaller than that of the previous techniques, especially compared to HE based solutions because of the cost of asymmetric operations. For the scalar product and Euclidean distance, GMW requires $\mathcal{O}(NK\ell^2)$ symmetric cryptographic operations while GSHADE requires $\mathcal{O}(NK\ell^2/o)$ and hence GSHADE has a smaller computation complexity by around a factor of o. The HE-based schemes, on the other hand, achieve a performance of NK asymmetric cryptographic operations, compared to $4NK\ell^2/o$ symmetric cryptographic operations in GSHADE. However, asymmetric cryptographic operations, i.e., modular exponentiations, are usually several orders of magnitudes slower than symmetric cryptographic operations such as AES or SHA, especially when increasing the security parameter. We provide concrete performance numbers in our experiments in §5.3.

Communication. For all distances, we can observe that GSHADE requires transmitting some orders of magnitude less data than the generic secure computation protocols (GC or GMW). When comparing the communication complexity of GSHADE to HE-based protocols, on the other hand, the communication complexity of GSHADE depends highly on the bit length ℓ. For the Hamming distance and normalized Hamming distance, the communication complexity of GSHADE grows with $\mathcal{O}(N\ell \log_2 \ell)$ while the communication complexity of HE-based protocols grows with $2N\rho$. In biometric identification protocols that use (normalized) Hamming distance the bitlength is relatively large (e.g., $\ell \in \{900, 2\,048\}$) s.t. $\ell \log_2 \ell > 2\rho$ and the communication complexity of GSHADE is higher than that of HE-based protocols. This is also the case for the scalar product and Euclidean distance where the communication complexity of GSHADE grows with $2NK\ell^2$ which is higher than the $2N\rho$ of HE-based protocols. Still, in our experiments in §5.3 GSHADE requires less than three times more communication than HE-based protocols.

5.3 Experimental Performance Comparison

In the following, we experimentally compare the performance of GSHADE to that of related work on the Eigenfaces and IrisCodes applications.

Experimental Setup. For all schemes, we compare the overall run-time and estimated communication complexity. We measured the complexity for our GSHADE protocol and compare it to the numbers reported in related work. We do not claim that we provide a fair comparison, since the results were measured on different machines and using different programming languages. Moreover, most implementations are not publicly available. We rather intend for the experiments to support the asymptotic complexities summarized in Tab. 4. However, we argue that the orders of magnitude improvements that we obtain are too significant to be explained by the use of different programming languages, libraries, or hardware alone. Note that, other than some related works, we only give the overall complexities and do not divide them into pre-computation and online phase. We

Distance Metric	Technique	Computation of $\mathcal{S}$	Communication [bits]
Hamming Distance	HE [38]	$N\ell$ asym	$2(N+\ell)\rho$
	GC [27]	$4N\ell$ sym	$4N\ell\kappa$
	GMW [42]	$4N\ell$ sym	$2N\ell\kappa$
	GSHADE (§3)	$(2N\ell\log_2\ell)/o + 3\ell$ sym	$\ell(N\log_2\ell+\kappa)$
Normalized Hamming Distance	HE [5]	$N\ell$ asym	$(2N+\ell)\rho$
	GC [10, 35]	$16N\ell$ sym	$14N\ell\kappa$
	GSHADE (§3)	$(8N\ell\log_2\ell)/o + 6\ell$ sym	$2\ell(2N\log_2\ell+\kappa)$
Scalar Product / Euclidean Distance	HE [3, 5, 18, 25, 28]	NK asym	$2(N+K)\rho$
	GMW [42]	$8NK\ell^2$ sym	$4NK\ell^2\kappa$
	GSHADE (§3)	$(2NK\ell(2\ell+\log_2 K))/o + 3K\ell$ sym	$K\ell(2N\ell+N\log_2 K+\kappa)$

Table 4: Asymptotic complexities for different 1-vs-N distance metrics with bit length ℓ, PRF output size o, vector dimension K, symmetric security parameter κ, and asymmetric security parameter ρ.

point out that, since the only cryptographic protocol that we use is OT, we can pre-compute all required cryptographic operations and thereby achieve a very efficient online phase, as demonstrated in [12, 42]. We implemented our GSHADE protocol using the correlated OT extension from the C++ OT library of [1]. For evaluating the comparison and minimum circuits, we use the C++ GMW implementation of [12] with optimizations of [42] and the random OT extension protocol of [1]. We chose the GMW framework of [12] as it is implemented in the same programming language and is also extensively based on OT. However, as GSHADE is independent of the generic secure computation protocol, we could alternatively use Yao's garbled circuits protocol, e.g., as implemented in [27]. For the WiFi experiments in Tab. 1 we limit the bandwidth between the PCs using the `tc` command. In order to allow comparison with previous works we also use short-term security parameters, i.e., symmetric security parameter $\kappa = 80$ and asymmetric security parameter $\rho = 1\,024$. We run our experiments on two 3.2 GHz Intel i5-4570 CPUs with 8 GB RAM each running Ubuntu 12.04 that are connected via Gigabit LAN.

5.3.1 SCiFI

The SCiFI protocol for face-recognition was specifically designed to be computed in a privacy-preserving fashion and has been implemented using various secure computation protocols. While the seminal work of [38] introduced the SCiFI algorithm and used homomorphic encryption to perform face-recognition, [27] and [42] improved on its performance by expressing the SCiFI functionality as binary circuit and evaluate it using Yao's garbled circuits protocol and the GMW protocol, respectively. The original SHADE protocol [9] improved on the performance of both when securely computing the Hamming distance. In the following we compare the performance of GSHADE, which in the case of SCiFI is the same as the SHADE protocol. We depict our results in Tab. 5.

From our results we can observe that the (G)SHADE protocol outperforms previous protocols both in communication and computation. Additionally, compared to the GMW implementation of [42], the performance of (G)SHADE scales better with increasing database size, resulting in an increasing runtime advantage of factor 4-5 and a communication advantage of factor 14 for 50 000 elements.

5.3.2 IrisCodes

We perform the IrisCode experiments using the same parameters as [5], where the iris codes $X, Y^1, \ldots, Y^N$ and masks $M, M'^1, \ldots, M'^N$ are 2 048-bit long and $\mathcal{S}$ holds thresholds $t^1, \ldots t^N$. The protocol of [5] uses DGK [15] to compute the numerator $A_j = |M \wedge M'^i \wedge (X \oplus Y^i)|$ and the denominator $B_i = |M \wedge M'^i|$ of the normalized Hamming distance and the product $B_i t^i$ and performs the comparison $A_i \overset{?}{<} t^i B_i$ using garbled circuits. The protocol of [5] also proposes to let the server rotate his codes Y^i and masks M'^i left and right by c different offsets (thus creating $2c$ rotated vectors, in addition to the original Y^i) and perform the comparison on each rotated value. If any of the distances with a rotated version of Y^i is below the threshold, the protocol outputs a match for Y^i, which can be done by evaluating a circuit consisting of $2c$ OR-gates. For our experiments, we set the number of rotations to $c = 0$, since performing c rotations essentially adds the same performance overhead for both, the protocol of [5] and GSHADE, as increasing the database size N by factor $2c + 1$.

Performances of GSHADE applied to IrisCode identification are depicted in Tab. 6. We observe that the performance of GSHADE for biometric IrisCode identification is acceptable overall, but as shown in Tab. 1 it is the slowest among all biometric identification protocols we tested. This low performance can be explained by the high amount of single-bit operations that are required to process the data in the OTs. While other applications process the data byte-wise or, in the case of SCiFI, require only few bitwise operations, the IrisCode application makes an extensive use of bitwise operations, thereby becoming far less efficient than other applications. However, we stress that a more careful implementation of bit operations as well as pre-computing required inputs would further decrease the runtime of GSHADE. In particular, all the values $f_i(0, Y^j)$ and $f_i(1, Y^j)$ can be computed once, when a data Y^j is added to the database, and stored with the database, instead of computing them online.

As shown in Tab. 6, the communication complexity of GSHADE is around 3 times higher compared to the homomorphic encryption based protocol of [5]. However, the overall computation time is much lower for GSHADE than for [5]: GSHADE is about 35 times faster for $N = 320$ and 12 times faster for $N = 10\,000$. Note that this improvement by more than one order of magnitude is significant and does not result from using different hardware.

	$N = 100$				$N = 320$			$N = 50{,}000$	
Protocol (Techniques)	[38] (HE)	[27] (GC)	[42] (GMW)	**Ours** (GSHADE +GMW)	[27] (GC)	[42] (GMW)	**Ours** (GSHADE +GMW)	[42] (GMW)	**Ours** (GSHADE +GMW)
Programming Language	Java	Java	C++	C++	Java	C++	C++	C++	C++
Time in sec.	244	8.8	0.3	**0.2**	42.9	0.5	**0.3**	46.0	**9.9**
Communication in MB	7.3	2.6	1.7	**0.2**	8.3	5.7	**0.5**	886.5	**63.4**

Table 5: Performances of privacy-preserving SCiFI identification protocols.

	$N = 320$		$N = 10\,000$	
Protocol (Techniques)	[5] (HE+GC)	**Ours** (GSHADE+GMW)	[5] (HE+GC)	**Ours** (GSHADE+GMW)
Programming Language	C	C++	C	C++
Time in sec.	17.6	**0.5**	212.6	**17.2**
Communication in MB	1.7	**4.9**	37.6	**87.5**

Table 6: Performances of privacy-preserving IrisCode identification protocols.

5.3.3 *FingerCodes*

For privacy-preserving FingerCode identification, we compare GSHADE to [28], which uses Paillier encryption to compute the Euclidean distance and garbled circuits to find the minimum. The results of this comparison are depicted in Tab. 7. Note that we excluded the backtracking step of [28] in the evaluation, which can be added if necessary.

While our protocol only slightly improves the communication complexity, the runtime improvements are significant, i.e., do not only result from choosing a different programming language. Our protocol improves the runtime by factor 500 for $N = 128$ elements and by factor 700 for $N = 1\,024$ elements.

5.3.4 *Eigenfaces*

Secure computation of face recognition using Eigenfaces has been initially proposed based on homomorphic encryption [18] and subsequently the runtime has been improved by using a combination of homomorphic encryption and garbled circuits [25, 41] and using the GMW protocol [42]. These works use the same parameters summarized in Tab. 2 and we give a performance comparison with our protocol in Tab. 8.

We observe that GSHADE achieves a very efficient runtime and achieves a speedup of factor 20 over the GMW-based protocol of [42] and even factor 66 to 100 over the HE-based protocols of [18,25]. Note that these runtime improvements of orders of magnitude are significant and hence do not only result from using different programming languages and hardware. The communication complexity of GSHADE is several times lower than that of the GMW-based solution and even comparable to the HE-based protocols.

6. CONCLUSION

We described an efficient protocol called GSHADE to securely evaluate several distance metrics ((normalized) Hamming distance, Euclidean distance, scalar product, Mahalanobis distance) and showed that it can be used for efficient privacy-preserving biometric identification of several biometric traits (iris, face, fingerprint) and protocols (SCiFI, Eigenfaces, Fisherfaces, FingerCodes, IrisCodes). GSHADE is based on oblivious transfer and benefits from recently proposed optimizations for oblivious transfer extensions. Our performance analysis shows that, depending on the traits, GSHADE can be used for privacy-preserving identification against up to several thousand database items per second.

We believe that GSHADE can be used for several applications in signal processing, pattern recognition, or image processing that require privacy. Finding further applications of GSHADE is an interesting topic for future research.

Acknowledgements

This work was supported by the German Federal Ministry of Education and Research (BMBF) within EC SPRIDE, by the Hessian LOEWE excellence initiative within CASED, and by the European Union Seventh Framework Programme (FP7/2007-2013) under grant agreement n. 609611 (PRACTICE). This work has been partially funded by the European FP7 FIDELITY project (SEC-2011-284862). The opinions expressed in this document only represent the authors' view. They reflect neither the view of the European Commission nor the view of their employer. This work has also been partially funded by the ANR SecuLar project. The first four authors are with Identity and Security Alliance (The Morpho and Télécom ParisTech Research Center).

7. REFERENCES

[1] G. Asharov, Y. Lindell, T. Schneider, and M. Zohner. More efficient oblivious transfer and extensions for faster secure computation. In *Computer and Communications Security (CCS)*, pages 535–548. ACM, 2013. Code available at `http://encrypto.de/code/OTExtension`.

[2] AT&T Laboratories Cambridge. The database of faces. `http://www.cl.cam.ac.uk/research/dtg/attarchive/facedatabase.html`.

[3] M. Barni, T. Bianchi, D. Catalano, M. Di Raimondo, R. Donida Labati, P. Failla, D. Fiore, R. Lazzeretti, V. Piuri, F. Scotti, and A. Piva. Privacy-preserving fingercode authentication. In *ACM workshop on Multimedia and Security (MMSEC)*, pages 231–240. ACM, 2010.

[4] P. N. Belhumeur, J. P. Hespanha, and D. J. Kriegman. Eigenfaces vs. Fisherfaces: Recognition

	$N = 128$		$N = 1\,024$	
Protocol (Techniques)	[28] (HE+GC)	**Ours** (GSHADE+GMW)	[28] (HE+GC)	**Ours** (GSHADE+GMW)
Programming Language	Java	C++	Java	C++
Time in sec.	148.2	**0.3**	1114.3	**1.6**
Communication in MB	2.2	**1.8**	17.5	**13.8**

Table 7: Performances of privacy-preserving FingerCode identification protocols.

	$N = 320$				$N = 1\,000$		
Protocol (Techniques)	[18] (HE)	[25] (HE+GC)	[42] (GMW)	**Ours** (GSHADE +GMW)	[25] (HE+GC)	[42] (GMW)	**Ours** (GSHADE +GMW)
Programming Language	C++	Python	C++	C++	Python	C++	C++
Time in sec.	40	79.6	17.7	**0.6**	139.6	26.3	**1.3**
Communication in MB	7.3	9.2	291.1	**7.7**	17	446.0	**9.4**

Table 8: Performances of privacy-preserving Eigenfaces identification protocols.

using class specific linear projection. In *European Conference on Computer Vision (ECCV)*, volume 1064 of *LNCS*, pages 43–58. Springer, 1996.

[5] M. Blanton and P. Gasti. Secure and efficient protocols for iris and fingerprint identification. In *European Symposium on Research in Computer Security (ESORICS)*, volume 6879 of *LNCS*, pages 190–209. Springer, 2011.

[6] D. Bogdanov, R. Talviste, and J. Willemson. Deploying secure multi-party computation for financial data analysis. In *Financial Cryptography (FC)*, volume 7397 of *LNCS*, pages 57–64. Springer, 2012.

[7] P. Bogetoft, D. L. Christensen, I. Damgård, M. Geisler, T. P. Jakobsen, M. Krøigaard, J. D. Nielsen, J. B. Nielsen, K. Nielsen, J. Pagter, M. I. Schwartzbach, and T. Toft. Secure multiparty computation goes live. In *Financial Cryptography (FC)*, volume 5628 of *LNCS*, pages 325–343. Springer, 2009.

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

[9] J. Bringer, H. Chabanne, and A. Patey. SHADE: Secure HAmming DistancE computation from oblivious transfer. In *Workshop on Applied Homomorphic Cryptography (WAHC)*, volume 7862 of *LNCS*, pages 164–176. Springer, 2013.

[10] J. Bringer, M. Favre, H. Chabanne, and A. Patey. Faster secure computation for biometric identification using filtering. In *IAPR International Conference on Biometrics (ICB)*, pages 257–264. IEEE, 2012.

[11] Carnegie Mellon University. The CMU Multi-PIE face database. `http://www.multipie.org`.

[12] S. G. Choi, K.-W. Hwang, J. Katz, T. Malkin, and D. Rubenstein. Secure multi-party computation of Boolean circuits with applications to privacy in on-line marketplaces. In *Cryptographers' Track at the RSA Conference (CT-RSA)*, volume 7178 of *LNCS*, pages 416–432. Springer, 2012. Code available at `http://www.ee.columbia.edu/~kwhwang/projects/gmw.html`.

[13] R. Cramer, I. Damgård, and J. B. Nielsen. Multiparty computation from threshold homomorphic encryption. In *EUROCRYPT*, volume 2045 of *LNCS*, pages 280–300. Springer, 2001.

[14] E. D. Cristofaro and G. Tsudik. Practical private set intersection protocols with linear complexity. In *Financial Cryptography (FC)*, volume 6052 of *LNCS*, pages 143–159. Springer, 2010.

[15] I. Damgård, M. Geisler, and M. Krøigaard. Efficient and secure comparison for on-line auctions. In *Australasian Conference on Information Security and Privacy (ACISP)*, volume 4586 of *LNCS*, pages 416–430. Springer, 2007.

[16] J. Daugman. How iris recognition works. *IEEE Transactions on Circuits and Systems for Video Technology*, 14(1):21–30, 2004.

[17] C. Dong, L. Chen, and Z. Wen. When private set intersection meets big data: An efficient and scalable protocol. In *Computer and Communications Security (CCS)*, pages 789–800. ACM, 2013.

[18] Z. Erkin, M. Franz, J. Guajardo, S. Katzenbeisser, I. Lagendijk, and T. Toft. Privacy-preserving face recognition. In *Privacy Enhancing Technologies Symposium (PETS)*, volume 5672 of *LNCS*, pages 235–253. Springer, 2009.

[19] S. Even, O. Goldreich, and A. Lempel. A randomized protocol for signing contracts. In *CRYPTO*, pages 205–210. Springer, 1982.

[20] C. Gentry. Fully homomorphic encryption using ideal lattices. In *Symposium on Theory of Computing (STOC)*, pages 169–178. ACM, 2009.

[21] O. Goldreich, S. Micali, and A. Wigderson. How to play any mental game or a completeness theorem for protocols with honest majority. In *Symposium on Theory of Computing (STOC)*, pages 218–229. ACM, 1987.

[22] R. Gross, I. Matthews, J. F. Cohn, T. Kanade, and S. Baker. Multi-PIE. *Image Vision and Computing*, 28(5):807–813, 2010.

[23] M. Günther, R. Wallace, and S. Marcel. An open source framework for standardized comparisons of face recognition algorithms. In *Benchmarking Facial Image*

Analysis Technologies (BeFIT), volume 7585 of *LNCS*, pages 547–556. Springer, 2012.

[24] C. Hazay and Y. Lindell. *Efficient Secure Two-Party Protocols - Techniques and Constructions*. Information Security and Cryptography. Springer, 2010.

[25] W. Henecka, S. Kögl, A.-R. Sadeghi, T. Schneider, and I. Wehrenberg. TASTY: Tool for Automating Secure Two-partY computations. In *Computer and Communications Security (CCS)*, pages 451–462, 2010.

[26] Y. Huang, D. Evans, and J. Katz. Private set intersection: Are garbled circuits better than custom protocols? In *Network and Distributed System Security Symposium (NDSS)*. The Internet Society, 2012.

[27] Y. Huang, D. Evans, J. Katz, and L. Malka. Faster secure two-party computation using garbled circuits. In *USENIX Security Symposium*. USENIX Association, 2011.

[28] Y. Huang, L. Malka, D. Evans, and J. Katz. Efficient privacy-preserving biometric identification. In *Network and Distributed System Security Symposium (NDSS)*. The Internet Society, 2011.

[29] Idiap Research Institute. Face recognition library. `https://pypi.python.org/pypi/facereclib`.

[30] Y. Ishai, J. Kilian, K. Nissim, and E. Petrank. Extending oblivious transfers efficiently. In *CRYPTO*, volume 2729 of *LNCS*, pages 145–161. Springer, 2003.

[31] A. K. Jain, S. Prabhakar, L. Hong, and S. Pankanti. FingerCode: A filterbank for fingerprint representation and matching. In *Computer Vision and Pattern Recognition (CVPR)*, pages 187–193. IEEE, 1999.

[32] V. Kolesnikov and R. Kumaresan. Improved OT extension for transferring short secrets. In *CRYPTO*, volume 8043 of *LNCS*, pages 54–70. Springer, 2013.

[33] V. Kolesnikov and T. Schneider. Improved garbled circuit: Free XOR gates and applications. In *International Colloquium on Automata, Languages and Programming (ICALP)*, volume 5126 of *LNCS*, pages 486–498. Springer, 2008.

[34] S. Z. Li and A. K. Jain, editors. *Encyclopedia of Biometrics*. Springer, 2009.

[35] Y. Luo, S.-C. S. Cheung, T. Pignata, R. Lazzeretti, and M. Barni. An efficient protocol for private iris-code matching by means of garbled circuits. In *International Conference on Image Processing (ICIP)*, pages 2653–2656. IEEE, 2012.

[36] M. Naor and B. Pinkas. Efficient oblivious transfer protocols. In *Symposium On Discrete Algorithms (SODA)*, pages 448–457. ACM/SIAM, 2001.

[37] M. Naor, B. Pinkas, and R. Sumner. Privacy preserving auctions and mechanism design. In *Conference on Electronic Commerce (EC)*, pages 129–139. ACM, 1999.

[38] M. Osadchy, B. Pinkas, A. Jarrous, and B. Moskovich. SCiFI - a system for secure face identification. In *IEEE Symposium on Security and Privacy (S&P)*, pages 239–254. IEEE, 2010.

[39] P. Paillier. Public-key cryptosystems based on composite degree residuosity classes. In *EUROCRYPT*, volume 1592 of *LNCS*, pages 223–238. Springer, 1999.

[40] M. O. Rabin. *How to exchange secrets with oblivious transfer*, TR-81 edition, 1981. Aiken Computation Lab, Harvard University.

[41] A.-R. Sadeghi, T. Schneider, and I. Wehrenberg. Efficient privacy-preserving face recognition. In *International Conference on Information Security and Cryptology (ICISC)*, volume 5984 of *LNCS*, pages 229–244. Springer, 2009.

[42] T. Schneider and M. Zohner. GMW vs. Yao? Efficient secure two-party computation with low depth circuits. In *Financial Cryptography (FC)*, volume 7859 of *LNCS*, pages 275–292. Springer, 2013.

[43] S. F. Shahandashti, R. Safavi-Naini, and P. Ogunbona. Private fingerprint matching. In *Australasian Conference on Information Security and Privacy (ACISP)*, volume 7372 of *LNCS*, pages 426–433. Springer, 2012.

[44] M. Turk and A. Pentland. Eigenfaces for recognition. *Journal of Cognitive Neuroscience*, 3(1):71–86, 1991.

[45] A. C.-C. Yao. How to generate and exchange secrets (extended abstract). In *Foundations of Computer Science (FOCS)*, pages 162–167. IEEE, 1986.

APPENDIX

A. CORRELATED OBLIVIOUS TRANSFER

In [1], Asharov *et al.* introduce a protocol that implements the correlated OT (C-OT) functionality. They describe it in a version that is adapted to Yao's protocol, *i.e.,* where the correlation function is an exclusive-OR. The generic version of their protocol can be used with any correlation function as long as the outputs lie in $\{0,1\}^l$ (or $\mathbb{Z}_{2^l}$), for some integer l. However, if the outputs of C-OT are integers from $\mathbb{Z}_m$, where $m \in \mathbb{N}$ is not a power of 2, the protocol cannot be applied directly. In particular, we need a hash function $H : \{0,1\}^* \to \mathbb{Z}_m$ that can be modeled as a random oracle. The protocol described in [1, Sec. 5.4] should then be modified as follows:

- $\mathcal{S}$ sends $y_j = f_{\Delta_j}(H(\mathbf{q}_j)) - H(\mathbf{q}_j \oplus \mathbf{s})$, for every $j = 1, \ldots, n$.
- For every $1 \leq j \leq n$, $\mathcal{R}$ outputs $H(\mathbf{t}_j)$ if $r_j = 0$ or $y_j + H(\mathbf{t}_j)$ if $r_j = 1$.

However, it is not practical to deal with such hash functions in actual implementations. Consequently, if the modulus m can be adapted, it is preferable to use a larger modulus $m' = 2^{\lceil \log_2(m) \rceil}$ and apply the protocol. This is the case for the biometric recognition protocols dealt with by GSHADE, since the modulus m is such that all distances lie in $\mathbb{Z}_m$ and taking a larger modulus $m' > m$ would not change correctness, because all distances obviously also lie in $\mathbb{Z}_{m'}$. Also notice that taking a modulus equal to $m' = 2^{\lceil \log_2(m) \rceil}$ does not degrade communication complexity and that the choice of the modulus does not impact security.