Available online at www.sciencedirect.com

ScienceDirect

ELSEVIER

Speech Communication 00 (2019) 1–18

Journal Logo

Privacy-Preserving PLDA Speaker Verification using Outsourced Secure Computation

Amos Treiber [a,*], Andreas Nautsch [b,c], Jascha Kolberg [b], Thomas Schneider [a], Christoph Busch [b]

[a] *ENCRYPTO – Cryptography and Privacy Engineering Group, Technische Universität Darmstadt, 64293 Darmstadt, Germany*
[b] *da/sec – Biometrics and Internet Security Research Group, Hochschule Darmstadt, 64295 Darmstadt, Germany*
[c] *Audio Security and Privacy Research Group, Digital Security Department, EURECOM, France*

Abstract

The usage of biometric recognition has become prevalent in various verification processes, ranging from unlocking mobile devices to verifying bank transactions. Automatic speaker verification (ASV) allows an individual to verify its identity towards an online service provider by comparing freshly sampled speech data to reference information stored on the service provider's server. Due to the sensitive nature of biometric data, the storage and usage thereof is subject to recent EU regulations introduced as means to protect the privacy of individuals enrolled in an automatic biometric verification system. Stored biometric data need to be *unlinkable*, *irreversible*, and *renewable* to satisfy international standards. Preserving privacy in ASV is also important because, contrary to other biometric characteristics such as fingerprints, speech data can be used to infer a lot of sensitive information about the data subject. As a result, some architectures have been proposed to enable privacy-preserving ASV in the encrypted domain. Recently, homomorphic encryption (HE) was proposed to protect both subject features and vendor models in an embedding-based ASV. This architecture improves on previous privacy-preserving ASV by using (probabilistic) embeddings (i-vectors) and by additionally protecting the vendor's model. However, the usage of HE comes with a rather heavy overhead and significantly slows down the verification process.

In this work, we align the cryptographic notion of outsourced secure two-party computation to embedding-based ASV. Our architecture protects biometric information in ASV and can also be used for any automatic biometric verification task. We show that unlinkability, irreversibility, and renewability are granted. Compared to the HE solution, our architecture results in considerably lower communication and computation overhead. Our architecture has been implemented and is experimentally evaluated on the NIST i-vector challenge 2014 using the cosine distance and log-likelihood ratio (LLR) scores from probabilistic linear discriminant analysis (PLDA) and two-covariance (2Cov) comparators. The results show that verification accuracy is retained while efficiency is improved. For instance, a PLDA verification with an embedding dimension of 200 takes about 77 milliseconds over a LAN. This is an improvement of more than 3000× over the HE-based solution and shows that privacy of subject and vendor data can be preserved in ASV while retaining practical verification times. Moreover, our system is secure against malicious client devices.

Keywords: biometric information protection, automatic speaker verification

[*] Corresponding author

Email addresses: `treiber@encrypto.cs.tu-darmstadt.de` (Amos Treiber), `nautsch@eurecom.fr` (Andreas Nautsch), `jascha.kolberg@h-da.de` (Jascha Kolberg), `schneider@encrypto.cs.tu-darmstadt.de` (Thomas Schneider), `christoph.busch@h-da.de` (Christoph Busch)

1. Introduction

In the past few years, the convenience and increased accuracy of automatic biometric verification architectures resulted in widespread use of popular systems like, e.g., automated passport controls or Apple's FaceID. After an initial enrolment of the individual's biometric reference information, the identity of the data subject can be verified by comparing a freshly extracted probe vector with the reference stored on a remote authentication server. Such verification systems provide strong and usable authentication guarantees even for online services that are invoked from a variety of devices, as the information required for the verification itself does not need to be stored on any device. Due to the unchangeable nature of biometric characteristics, security against impersonation can no longer be guaranteed once biometric information is leaked [1]. Thus, biometric data embody especial sensitive personal information and need to be protected even from service providers when used in automatic verification systems. This prevents not only the leakage of biometric information in the case of data breaches through targeted attacks or negligence, but also preserves the privacy of data subjects within the system. Automatic speaker verification (ASV), the case where verification relies on speech data, is a sought-after concept for critical use cases like, e.g., a bank that wants to verify the identity of a user via telephone or VoIP without the inconvenience of requiring the user to remember a secure password or requiring a device-specific cryptographic key. For the speech data used in ASV, additional unique privacy concerns arise because speech not only uniquely identifies a speaker, but can also be used to infer a wide range of sensitive information about, among others, the subject's gender (Harb & Chen, 2005), age (García et al., 2015), or health status (Gómez-Vilda et al., 2009).

Recently, the privacy concerns of outsourced biometric data were addressed in current EU privacy regulations (European Council, 2016) and, as a result, the need to preserve privacy by properly protecting biometric information became more prominent in research as well as in industry. Biometric information protection is achieved by three properties as requested by the ISO/IEC 24745 standard (ISO/IEC JTC1 SC27 Security Techniques, 2011):

1. *Unlinkability:* Given only protected biometric information, it is not possible to say whether two protected biometric sample representations belong to the same subject. This prevents cross-comparisons of databases from different applications and ensures the privacy of the subject.

2. *Renewability:* If a protected biometric reference is leaked or lost, the reference data can be revoked and renewed from the same biometric trait without the need to re-enroll.

3. *Irreversibility:* Recovering biometric data from leaked protected biometric information is impossible without knowing the secret used to protect the biometric information. The restoration of valid biometric features or samples is prevented.

In addition, the recognition performance of unprotected systems should be preserved.

State-of-the-art speaker recognition systems rely on embeddings, namely *intermediate-sized vectors* (i-vectors) (Dehak et al., 2011), here referred to as *probabilistic embeddings*; and *x-vectors* (deep learning embeddings) (Snyder et al., 2016), here referred to as *discriminative embeddings*. Conventionally, either are compared by probabilistic linear discriminant analysis (PLDA) (Prince & Elder, 2007; Garcia-Romero & Epsy-Wilson, 2011). Seeking fast comparisons, one can employ cosine comparison, however comparisons carried out in a (lower dimensional) biometric subspace promise better biometric discrimination performance. Moreover, comparisons carried out in biometric subspaces and resulting in probabilistic comparison scores (such as log-likelihood ratios; LLRs) are coherent within decision frameworks based on information theory and decision cost risks. One of the goals of our work is to preserve these LLR score properties. When the subspace is of the same dimension as the embedding feature space, PLDA simplifies to the two-covariance model (2Cov) (Cumani & Laface, 2014). We will refer to i-vectors in our experiments, as the comparator technology is independent of whether probabilistic or discriminative embeddings are employed.

Privacy preservation has been in the focus of research for some time, though ASV itself has not received a level of attention that is equivalent to other biometrics. Approaches tackling biometric information protection can be classified

[1] This is because changing the way biometric information is extracted is not sufficient to prevent linkability. For instance, Glembek et al. (2015) show how i-vectors (probabilistic embeddings) from different speaker recognition systems can be linked. Also, research has proven that it is possible to recover biometric samples from templates for other modalities such as finger (Cappelli et al., 2007), face (Adler, 2003), or iris (Galbally et al., 2013).

into three main categories: i) cancelable biometrics (Patel et al., 2015), where samples or biometric information representations are irreversibly transformed, ii) cryptobiometrics (Campisi, 2013), where a key is either extracted from or bound to the biometric data, and iii) biometrics in the encrypted domain (Aguilar-Melchor et al., 2013), where techniques such as homomorphic encryption (HE) are applied to protect biometric data. However, the first two categories usually report some accuracy degradation (Rathgeb & Uhl, 2011) while for biometrics in the encrypted domain the operations are equivalent to plaintext ones. Only biometrics in the encrypted domain can provide biometric information protection while sustaining biometric recognition performance (without diminishing accuracy for compensations in computational speed or privacy).

An Architecture using HE for privacy-preserving embedding-based ASV with PLDA/2Cov comparisons in the encrypted domain was recently proposed in the speech community by Nautsch et al. (2018). This architecture not only protects the speaker's embedding, but also the vendor's model used during verification. However, it relies on asymmetric (public-key) cryptography, which requires a relatively large key size and is computationally expensive. Another issue is that this architecture is vulnerable (in terms of security, not in terms of privacy) to authentication by a malicious client device.

We tackle these issues by aligning secure two-party computation (STPC) protocols to embedding-based ASV in the encrypted domain. STPC is a cryptographic notion different to HE that allows participating parties to compute *any* functionality without revealing anything else about a party's input. As such, STPC seems like a promising candidate for practical biometric verification in the encrypted domain. In fact, privacy-preserving computation of biometric distances has been a popular benchmarking application in the STPC literature (Sadeghi et al., 2009; Osadchy et al., 2010; Blanton & Gasti, 2011; Blanton & Aliasgari, 2012; Bringer et al., 2014). The results show that certain biometric distances, as they are computed in various comparison subsystems, e.g., for face, fingerprint, or iris identification, can be efficiently computed in a privacy-preserving manner. However, the secure evaluation of speech data has not been a prominent subject of the cryptographic community as it usually relies on less efficient floating point arithmetic. Biometric information protection is also rarely achieved in those works: the outsourced reference data is usually known in plain to the server like in the classical STPC client-server setting. To the best of our knowledge, only the iris identification architecture by Blanton & Aliasgari (2012) and the speaker recognition architecture by Aliasgari & Blanton (2013) achieve biometric information protection solely by employing secure multi-party computation protocols between *multiple* servers. This technique will be referred to as outsourced STPC in this work and is based on the work of Kamara & Raykova (2011) who presented a transformation of any STPC protocol into an outsourced one.

In this work, we close the aforementioned gaps between the STPC and the speech communities by applying outsourced STPC to embedding-based ASV with biometric information protection for PLDA/2Cov comparators. Our main contributions are as follows:

- We propose a general architecture for protecting sensitive data in any automated biometric verification system. Our architecture is based on the outsourced STPC construction of Kamara & Raykova (2011) and is similar to the one of Blanton & Aliasgari (2012) while making no additional assumptions compared to the previous work of Nautsch et al. (2018). We show that it achieves biometric information protection according to the ISO/IEC IS 24745 standard (ISO/IEC JTC1 SC27 Security Techniques, 2011).

- Using the general architecture, we show how to protect speaker embeddings and vendor models in ASV, using probabilistic embeddings (i-vectors) as an example. Our solution relies on a mix of different STPC protocols, employing very efficient protocols for various steps of the verification. In contrast to the previous solution of Nautsch et al. (2018), it is also secure against malicious clients.

- We implement our ASV architecture using the state-of-the-art STPC framework *ABY* of Demmler et al. (2015b) and evaluate it on the NIST i-vector challenge (Bansé et al., 2014) phase III database (i.e., with labeled development data). The results show that compared to the approach of Nautsch et al. (2018) in a comparable setting, our architecture is more practical while retaining biometric performance and level of security (through equivalent security parameters). Notably, our implementation scales and truncates float inputs to improve efficiency, thereby limiting threshold precision to a reasonable extend. However, our evaluation shows that this does not degrade biometric recognition performance. Even for high embedding dimensions, PLDA/2Cov verifications are computed in about half a second. Lower dimension PLDA/2Cov verifications as well as higher dimension cosine verifications require only a few milliseconds. This is an improvement of up to 4000× over Nautsch et al. (2018).

The rest of the paper is structured as follows. We review related work in Section 2 and present the generative speech models used as the basis for our privacy-preserving ASV in Section 3. Section 4 gives a high-level overview of STPC as well as a lower-level description of the STPC protocols we employ in our architecture. We outline our general biometric as well as our specific embedding-based ASV architecture in Section 5 and thoroughly evaluate its implementation on the NIST i-vector challenge in Section 6. In Section 7, we conclude our work and give potential future work.

2. Related Work

Homomorphic encryption (HE) enables operations on encrypted data without requiring a decryption key, while secure two-party computation (STPC) allows two parties to interactively compute any functionality in a secure manner. In order to preserve privacy in biometric applications, the efficient and secure computation of biometric distances has been a popular use case of these advanced cryptographic mechanisms. Plenty of solutions exist for, e.g., privacy-preserving iris (Blanton & Gasti, 2011; Blanton & Aliasgari, 2012; Bringer et al., 2014), face (Erkin et al., 2009; Sadeghi et al., 2009; Osadchy et al., 2010; Bringer et al., 2014; Hu et al., 2018), and fingerprint (Blanton & Gasti, 2011; Evans et al., 2011; Hu et al., 2018) computations. For the most part, the literature is focused on identification rather than verification: the task is to identify a client based on a comparison of the freshly extracted biometric information (the probe) and a database of stored reference data without revealing information about the probe vector to the server and without revealing information about the reference database to the client. Naturally, secure computation techniques like HE and STPC can be employed to guarantee those privacy goals. However, the setting of biometric verification, where the identity of the data subject is already determined and needs to be authenticated via biometric characteristics, has received far less attention. The transformation of existing identification into verification architectures is not trivial due to the additional goal of protecting biometric information being violated by the server usually knowing the reference database in plaintext. Of the mentioned solutions, only the architectures of Blanton & Aliasgari (2012); Chun et al. (2014); Hu et al. (2018) protect stored biometric data via HE and/or STPC. STPC has also been considered to securely compute the combination of multiple biometrics for the authentication process (Toli et al., 2016; Toli, 2018).

In Bringer et al. (2013) and Barni et al. (2015), the authors provide an overview of several biometric information protection schemes based on HE and STPC. Barni et al. (2010) present a way to protect fixed-length fingercodes (Jain et al., 1997) using HE. This system was modified in Bianchi et al. (2010) to accelerate the process by reducing the size of the fingercode. However, a reduction of information also leads to a degradation of biometric recognition performance. An anonymous biometric access control (ABAC) system (Ye et al., 2009) is presented for iris recognition such that the system setup verifies only whether a subject is enrolled without revealing the identity and thus granting anonymity towards the subjects. Another ABAC protocol is proposed in Luo et al. (2009) based on a secure similarity search algorithm for anonymous authentication.

Privacy-preserving speech technology was first considered in Smaragdis & Shashanka (2007) using STPC and hidden Markov models (HMMs) in a client-server setting. *Yao's garbled circuit* STPC protocol (Yao, 1986) has been applied to Gaussian mixture models (GMMs) in Portêlo et al. (2014). While these approaches solely rely on STPC, they require that the server knows the HMM or GMM of the subject in plain, leaking a characterization of the subject's voice to the server. For ASV in the encrypted domain based on HMMs and GMMs, a series of implemented systems that also protect the subject's model on the server via HE and allow for verification via HE and STPC have been proposed (cf. Pathak (2012) and Pathak et al. (2013)). Pathak et al. (2011) outline how to perform inference and classification of encrypted HMMs using HE and STPC while Pathak & Raj (2011) and Pathak & Raj (2013) extend these techniques for GMM-based speaker verification and identification. Aliasgari & Blanton (2013) present methods of securely computing HMMs, either in a client-server setting by also homomorphically encrypting the model, or in an outsourcing scenario where three or more computational parties share and securely evaluate the model, thereby not requiring the client to keep a secret key for an encryption scheme. They provide full floating point operations and evaluate an implementation in Aliasgari et al. (2017), demonstrating that the multi-party outsourcing scenario significantly decreases workload compared to using HE. Their outsourcing setting is very similar to ours for embedding-based ASV, but we use a mix of STPC techniques between two servers that are not allowed to collude, while their technique requires three or more parties of which a majority are required to not collude. Compared to the STPC protocols that we employ, their *honest majority* multi-party protocol does not have the possibility of pre-computing input-independent parts of the verification, which significantly improves on transaction costs (cf. Section 4). The literature for privately evaluating HMMs and GMMs outlined here shows that a secure verification using these models can be efficiently performed in a matter of seconds or minutes.

In the domain of privacy-preserving ASV based on (probabilistic/discriminative) embeddings that we address in this work, efficient STPC tools still lack deployment. The recent work of Nautsch et al. (2018) uses HE to protect biometric information in an outsourced embedding-based ASV infrastructure with two non-colluding servers called $\text{DB}_{controller}$ and $\text{AS}_{operator}$. During enrolment, the reference vector is encrypted using the public key of the authentication server $\text{AS}_{operator}$ and is stored on the database server $\text{DB}_{controller}$. The encrypted vector is sent to the client device for the process of verification and is used to homomorphically compute the comparison score in the encrypted domain. This encrypted score is then sent to $\text{AS}_{operator}$, which decrypts the score and decides whether the client should be accepted by comparing the score to a threshold. The biometric data is protected if one assumes that $\text{DB}_{controller}$ and $\text{AS}_{operator}$ do not collude, i.e., $\text{DB}_{controller}$ does not share its information with $\text{AS}_{operator}$ and $\text{AS}_{operator}$ does not share its secret key with $\text{DB}_{controller}$. This assumption can be seen as realistic, given that $\text{DB}_{controller}$ could be an independent provider of privacy services that relies on a reputation of acting honestly. The usage of HE comes with a high overhead that makes this solution impractical when considering computationally limited devices like, e.g., mobile phones. Also, this method is not secure against a malicious user that can just send the encryption of an accepting score to $\text{AS}_{operator}$. Another solution by Rahulamathavan et al. (2019) does not rely on HE but uses randomization of feature vectors in combination with a privacy-preserving scalar product protocol (Lu et al., 2014). However, this architecture is completely insecure because the protocol of Lu et al. (2014) was irreparably broken in Schneider & Treiber (2019). To the best of our knowledge, embedding-based ASV for state-of-the-art PLDA/2Cov comparators solely using STPC protocols has not been addressed by the literature to date. In this work, we demonstrate that a mix of outsourced STPC protocols can significantly reduce practical verification times for device-independent embedding-based ASV.

The usage of distance-preserving hashing techniques for privacy-preserving ASV (Pathak & Raj, 2012; Pathak et al., 2012; Portêlo et al., 2013; Jiménez et al., 2015; Jiménez & Raj, 2017a,b) is an alternative approach from cancelable biometrics. Additionally, the usage of trusted execution environments like Intel SGX (McKeen et al., 2013) for privacy-preserving speech processing using, among others, i-vectors has been explored by Brasser et al. (2018). However, this requires trusting the remote attestation feature that is built into Intel CPUs. One also has to take additional measures against side-channel vulnerabilities when employing SGX (Xu et al., 2015; Costan & Devadas, 2016).

3. Speaker Verification using Cosine and PLDA Comparison

Cosine and PLDA comparisons of two embeddings $\boldsymbol{X} = \{x_1, \ldots, x_F\}, \boldsymbol{Y} = \{y_1, \ldots, y_F\}$ (biometric probe and reference of dimension F) are typically carried out on length-normalized embeddings, approximating radial Gaussianization of the biometric data before comparison (Garcia-Romero & Epsy-Wilson, 2011). In other words, embeddings are observed to scale in their length with increasing voice sample data; by length-normalization, embeddings are projected onto the unit sphere. Then, cosine comparison scores $S_{cos}(\boldsymbol{X}, \boldsymbol{Y})$ are computed as:

$$S_{cos}(\boldsymbol{X}, \boldsymbol{Y}) = \boldsymbol{X}^\top \boldsymbol{Y} = \sum_{f=1}^{F} x_f \, y_f \,. \tag{1}$$

PLDA comparison scores $S_{\text{PLDA}}(\boldsymbol{X}, \boldsymbol{Y})$ are computed in terms of log-expectations, examining to what extent reference and probe data originate from the same speaker. If so, stacked embeddings correlate in terms of within class variance $\boldsymbol{\Sigma}_{\text{within}}$, whereas if they stem from different speakers, solely the total variability $\boldsymbol{\Sigma}_{\text{total}}$ is modeled (i.e., $\boldsymbol{\Sigma}_{\text{within}} = \boldsymbol{0}$). For centered data (i.e., with zero mean $\boldsymbol{\mu} = \boldsymbol{0}$), PLDA scores are computed as (Garcia-Romero & Epsy-Wilson, 2011):

$$S_{\text{PLDA}}(\boldsymbol{X}, \boldsymbol{Y}) = \log \mathcal{N}\left(\begin{bmatrix}\boldsymbol{X}\\ \boldsymbol{Y}\end{bmatrix} \,\middle|\, \begin{bmatrix}\boldsymbol{0}\\ \boldsymbol{0}\end{bmatrix}, \begin{bmatrix}\boldsymbol{\Sigma}_{\text{total}} & \boldsymbol{\Sigma}_{\text{within}}\\ \boldsymbol{\Sigma}_{\text{within}} & \boldsymbol{\Sigma}_{\text{total}}\end{bmatrix}\right) - \log \mathcal{N}\left(\begin{bmatrix}\boldsymbol{X}\\ \boldsymbol{Y}\end{bmatrix} \,\middle|\, \begin{bmatrix}\boldsymbol{0}\\ \boldsymbol{0}\end{bmatrix}, \begin{bmatrix}\boldsymbol{\Sigma}_{\text{total}} & \boldsymbol{0}\\ \boldsymbol{0} & \boldsymbol{\Sigma}_{\text{total}}\end{bmatrix}\right),$$

$$S_{\text{PLDA}}(\boldsymbol{X},\boldsymbol{Y}) = \boldsymbol{X}^{\mathrm{T}}\boldsymbol{Q}\,\boldsymbol{X} + \boldsymbol{Y}^{\mathrm{T}}\boldsymbol{Q}\,\boldsymbol{Y} + \boldsymbol{X}^{\mathrm{T}}\boldsymbol{P}\,\boldsymbol{Y} + \boldsymbol{Y}^{\mathrm{T}}\boldsymbol{P}\,\boldsymbol{X} + const$$
$$with \quad \boldsymbol{Q} = \boldsymbol{\Sigma}_{\text{total}}^{-1} - \left(\boldsymbol{\Sigma}_{\text{total}} - \boldsymbol{\Sigma}_{\text{within}}\,\boldsymbol{\Sigma}_{\text{total}}^{-1}\,\boldsymbol{\Sigma}_{\text{within}}\right)^{-1}, \qquad \boldsymbol{P} = \boldsymbol{\Sigma}_{\text{total}}^{-1}\,\boldsymbol{\Sigma}_{\text{within}}\left(\boldsymbol{\Sigma}_{\text{total}} - \boldsymbol{\Sigma}_{\text{within}}\,\boldsymbol{\Sigma}_{\text{total}}^{-1}\,\boldsymbol{\Sigma}_{\text{within}}\right)^{-1}, \qquad (2)$$

where a constant term summarizes the normalization terms of the log-likelihood functions. In PLDA, the $\boldsymbol{\Sigma}_{\text{within}}, \boldsymbol{\Sigma}_{\text{total}}$ parameters are derived from the underlying subspace model. If the subspace dimension is equal to the dimension of the observed feature/embedding space, PLDA simplifies to the 2Cov model (differing in the estimation of model parameters). During score computation, PLDA and 2Cov carry out the same matrix operations. By employing the *Frobenius inner product* (denoting the operation $\boldsymbol{x}^{\mathrm{T}}\boldsymbol{A}\,\boldsymbol{y} = \text{vec}(\boldsymbol{A})^{\mathrm{T}}\,\text{vec}(\boldsymbol{x}\,\boldsymbol{y}^{\mathrm{T}})$ with the operator vec(·) stacking matrices into vectors), the PLDA/2Cov score is expressed in terms of a dot product (Cumani et al., 2013) with within and between covariances $\boldsymbol{W}^{-1}$ and $\boldsymbol{B}^{-1}$ and the mean value $\boldsymbol{\mu}$ not necessarily being equal to $\boldsymbol{0}$ (features/embeddings are not centered):

$$S_{2Cov}(\boldsymbol{X},\boldsymbol{Y}) = \boldsymbol{X}^{\mathrm{T}}\boldsymbol{\Lambda}\,\boldsymbol{Y} + \boldsymbol{Y}^{\mathrm{T}}\boldsymbol{\Lambda}\,\boldsymbol{X} + \boldsymbol{X}^{\mathrm{T}}\boldsymbol{\Gamma}\,\boldsymbol{X} + \boldsymbol{Y}^{\mathrm{T}}\boldsymbol{\Gamma}\,\boldsymbol{Y} + \boldsymbol{c}^{\mathrm{T}}\,(\boldsymbol{X}+\boldsymbol{Y}) + k,$$
$$\boldsymbol{\Lambda} = \frac{1}{2}\boldsymbol{W}^{\mathrm{T}}\tilde{\boldsymbol{\Lambda}}\,\boldsymbol{W}, \qquad \boldsymbol{\Gamma} = \frac{1}{2}\boldsymbol{W}^{\mathrm{T}}\left(\tilde{\boldsymbol{\Lambda}} - \tilde{\boldsymbol{\Gamma}}\right)\boldsymbol{W}, \qquad \boldsymbol{c} = \boldsymbol{W}^{\mathrm{T}}\left(\tilde{\boldsymbol{\Lambda}} - \tilde{\boldsymbol{\Gamma}}\right)\boldsymbol{B}\,\boldsymbol{\mu}, \qquad k = \tilde{k} + \frac{1}{2}\left((\boldsymbol{B}\,\boldsymbol{\mu})^{\mathrm{T}}\left(\tilde{\boldsymbol{\Lambda}} - 2\,\tilde{\boldsymbol{\Gamma}}\right)\boldsymbol{B}\,\boldsymbol{\mu}\right),$$
$$\tilde{\boldsymbol{\Lambda}} = (\boldsymbol{B} + 2\,\boldsymbol{W})^{-1}, \quad \tilde{\boldsymbol{\Gamma}} = (\boldsymbol{B} + \boldsymbol{W})^{-1}, \qquad \tilde{k} = 2\,\log|\tilde{\boldsymbol{\Gamma}}| - \log|\tilde{\boldsymbol{\Lambda}}| - \log|\boldsymbol{B}| + \boldsymbol{\mu}^{\mathrm{T}}\,\boldsymbol{B}\,\boldsymbol{\mu}. \qquad (3)$$

Utilizing the Frobenius inner product, PLDA/2Cov scores are computed with a segregated comparison model and biometric data terms (Cumani et al., 2013):

$$S_{2Cov}(\boldsymbol{X},\boldsymbol{Y}) = \begin{bmatrix} \text{vec}(\boldsymbol{\Gamma}) \\ \text{vec}(\boldsymbol{\Lambda}) \\ \boldsymbol{c} \\ k \end{bmatrix}^{\mathrm{T}} \begin{bmatrix} \text{vec}(\boldsymbol{X}\,\boldsymbol{X}^{\mathrm{T}} + \boldsymbol{Y}\,\boldsymbol{Y}^{\mathrm{T}}) \\ \text{vec}(\boldsymbol{X}\,\boldsymbol{Y}^{\mathrm{T}} + \boldsymbol{Y}\,\boldsymbol{X}^{\mathrm{T}}) \\ \boldsymbol{X} + \boldsymbol{Y} \\ 1 \end{bmatrix}. \qquad (4)$$

For the sake of readability concerning the secure computation of these scores, we will refer to cosine scores as S_{cos}, and to PLDA/2Cov scores simply as S_{PLDA}, since 2Cov is a PLDA special case and both are computationally identical during comparison. The notation $S^{\boldsymbol{0}}_{PLDA}$ indicates centered data before score computation and $S^{\boldsymbol{\mu}}_{PLDA}$ indicates non-centered data.

In the Bayesian decision framework (Brümmer, 2008; Meuwly et al., 2017), log-likelihood ratio (LLR) scores are compared to a Bayesian threshold η. Bayesian thresholds are denoted in a formal way based on beliefs in prior class probability and cost parameters, setting the required weight of evidence a LLR needs to support to decision making. PLDA/2Cov scores directly result in LLRs, whereas cosine comparison scores need to be transformed to satisfy LLR properties (Brümmer & de Villiers, 2010; Brümmer et al., 2014). In biometric systems for identity verification, sustaining LLR score properties in privacy-preserving computations is paramount, especially when performance assessment aims at increasing information gains and lowering decision cost risks. We will demonstrate how to employ secure two-party computation to sustain LLR score properties in the encrypted domain.

4. Secure Two-Party Computation

Secure two-party computation (STPC) allows two parties P_0 and P_1 with respective inputs x and y to securely evaluate a function f without revealing any information except $f(x, y)$ to any party. The many practical implementations (Hastings et al., 2019) of various STPC protocols show that, nowadays, STPC is far from being just a theoretical concept. However, widespread deployment by non-experts still faces some hurdles, most importantly because the efficiency of different protocols highly depends on the data type, network properties, and the function that is securely evaluated. In this section, we give a brief overview of how STPC protocols work in general before providing more details on the protocols we use in our ASV architecture.

At a high level, such protocols do not operate on plain or on encrypted data, but instead on data that has been *secretly shared* between the parties. Secret sharing is a cryptographic notion that allows to split a value x into multiple values called shares. The shares can be distributed among parties and can be used to reconstruct the original secret x under the constraint that a party obtained a sufficient number shares. Since we will only consider the case of secure computation between two parties in this work, we will hereafter denote a secret sharing of x by the notation $\langle x \rangle_0$

and $\langle x\rangle_1$ (representing the first and second secret share of an input x). P_0 will be in possession of $\langle x\rangle_0$ while P_1 will hold $\langle x\rangle_1$. One share alone reveals no information about x, as both $\langle x\rangle_0$ and $\langle x\rangle_1$ are necessary to reveal the original secret x using a publicly known function *Reconstruct*. Thus, a STPC party could only reconstruct an input if it somehow obtains the share of the other party. In STPC, the function f is usually represented as a circuit, where basic functionalities called gates (e.g., a Boolean AND gate) compute outgoing values based on incoming values. Every gate in the circuit representation of f is computed by a party only on the secret shares it possesses, possibly involving some communication among both parties. The computation reveals nothing about the other party's input. At the end of the computation, the overall result can be revealed by sending the resulting output shares to the party that should obtain the output.

Compared to HE, STPC mostly uses symmetric cryptography such as AES. Thus, computation requirements are usually lower than for HE, though some rounds of communication may be required. Another advantage of STPC is the property of dividing the secure computation of a function into a setup and an online phase. In the setup phase, input-independent pre-computation is performed that is later on used for the input-dependent online phase. This method greatly reduces the time needed for the actual time-critical online phase, considering that the setup phase for potential future evaluations could be performed during idle times when computation and communication are possible, e.g., over night or during weekends.

In the following, we will use the STPC notation of the *ABY* framework by Demmler et al. (2015b) and summarize the relevant protocols described and implemented in that framework. The authors distinguish three common STPC protocols denoted by their *sharing type* and identified by the first letter of the type: *arithmetic* (A), *Boolean* (B), and *Yao* (Y, commonly known as *Yao's garbled circuits*). A sharing type determines not only how a secret share is created, distributed, and reconstructed, but also which STPC protocol is used to securely compute a function on the shares. In the notation employed here, $\langle x\rangle_i^T$ refers to the sharing of type $T \in \{A, B, Y\}$ of x held by party $P_{i\in\{0,1\}}$. We rely on arithmetic as well as Yao sharing in our architectures because Boolean sharing would result in a higher round complexity. We also assume honest-but-curious (also known as semi-honest) parties P_0 and P_1 (i.e., parties, while not trusted, are not allowed to deviate from the protocol description).

4.1. Arithmetic Sharing

In arithmetic sharing, all operations occur in the algebraic Ring $\mathbb{Z}_{2^l}$ for l-bit inputs. The protocol was first proposed by Ben-Or et al. (1988) in the setting of multiple parties. Here, we present the case for two parties using additive arithmetic secret sharing in the style of the protocol for Boolean sharing (Goldreich et al., 1987), as described by Demmler et al. (2015b). Party $P_{i\in\{0,1\}}$ can share its input x by choosing $r \in_R \mathbb{Z}_{2^l}$ uniformly at random, and setting $\langle x\rangle_i^A = x - r \mod 2^l$ and $\langle x\rangle_{1-i}^A = r$. P_i's arithmetic share of x is $\langle x\rangle_i^A$, while $\langle x\rangle_{1-i}^A$ is sent to P_{1-i} and serves as its arithmetic share. For reconstructing the secret x, one simply adds $\langle x\rangle_i^A$ and $\langle x\rangle_{1-i}^A$, as $x = \langle x\rangle_i^A + \langle x\rangle_{1-i}^A \mod 2^l$. One secret share alone does not leak anything about the input because r is sampled uniformly at random; the secret can only be revealed if both parties exchange their shares. This step occurs at the end of the protocol, in order to reveal just the result of the computation to one or to both parties.

After the initial sharing, each party is in possession of its shares $\langle x\rangle_i^A$ and $\langle y\rangle_i^A$ of both inputs. With this type of sharing, each P_i only has to perform a local addition of its shares $\langle z\rangle_i^A = \langle x\rangle_i^A + \langle y\rangle_i^A \mod 2^l$ in order to compute a sum $z = x + y \mod 2^l$. $\langle z\rangle_i^A$ is a valid share of the sum

$$\begin{aligned}\langle z\rangle_0^A + \langle z\rangle_1^A &= \langle x\rangle_0^A + \langle y\rangle_0^A + \langle x\rangle_1^A + \langle y\rangle_1^A \mod 2^l \\ &= x - r_x + r_y + r_x + y - r_y = x + y \mod 2^l, \end{aligned} \tag{5}$$

which will only be revealed if sent to the other party. Addition is very efficient here, as the parties only have to perform one local addition in $\mathbb{Z}_{2^l}$ without any communication.

Since every arithmetic circuit can be displayed with just addition and multiplication gates, the parties also need to be able to securely compute multiplications using just their shares. In contrast to addition, multiplication gates require interaction between the parties. They rely on *multiplication triples* (Beaver, 1991) of the form $\langle c\rangle_0^A + \langle c\rangle_1^A = (\langle a\rangle_0^A + \langle a\rangle_1^A) \cdot (\langle b\rangle_0^A + \langle b\rangle_i^A) \mod 2^l$, which can be pre-computed independent of the input prior to protocol execution. The evaluation of a multiplication gate $z = x \cdot y \mod 2^l$ works as follows:

1. Party P_i sets $\langle e\rangle_i^A = \langle x\rangle_i^A - \langle a\rangle_i^A \mod 2^l$ and $\langle f\rangle_i^A = \langle y\rangle_i^A - \langle b\rangle_i^A \mod 2^l$.

2. Party P_i sends $\langle e \rangle_i^A$ and $\langle f \rangle_i^A$ to party P_{1-i} and, upon receiving the shares of the other party, reconstructs e and f by adding the respective shares.

3. To compute the output of the multiplication gate, party P_i sets $\langle z \rangle_i^A = i \cdot e \cdot f + f \cdot \langle a \rangle_i^A + e \cdot \langle b \rangle_i^A + \langle c \rangle_i^A$.

Hence, the resulting share is a valid share of the product

$$\begin{aligned}\langle z \rangle_0^A + \langle z \rangle_1^A &= f \cdot \langle a \rangle_0^A + e \cdot \langle b \rangle_0^A + \langle c \rangle_0^A + e \cdot f + f \cdot \langle a \rangle_1^A + e \cdot \langle b \rangle_1^A + \langle c \rangle_1^A \mod 2^l \\ &= f \cdot a + e \cdot b + c + e \cdot f = x \cdot y \mod 2^l. \end{aligned} \tag{6}$$

Generation of the multiplication triples in the setup phase can be done with an efficient implementation of oblivious transfer (OT) (Asharov et al., 2013). These cryptographic operations are performed in the setup phase while the online phase only requires sending two shares and locally performing seven additions and four multiplications in $\mathbb{Z}_{2^l}$ per party. With the steps to compute additions and multiplications outlined above, two parties can compute any arithmetic functionality by securely evaluating every addition or multiplication gate in the circuit, and reconstructing the secret using the resulting share of the output. It follows that the protocol requires a number of rounds linear in the multiplicative depth of the circuit.

4.2. Yao's Garbled Circuits (GC)

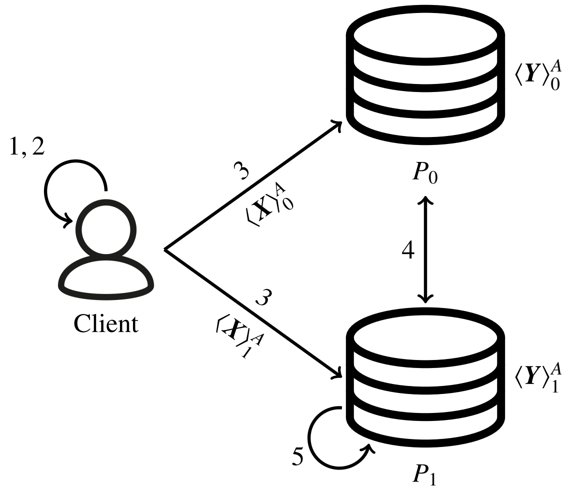
Probably the most widely known STPC protocol is Yao's garbled circuits (GC) protocol (Yao, 1986; Lindell & Pinkas, 2009). It enables the secure evaluation of f by evaluating its *garbled* Boolean circuit representation. The first party, called the garbler, creates random labels $k_0^w, k_1^w \in_R \{0,1\}^\kappa$ for every wire w in the circuit, where κ is the security parameter (e.g., $\kappa = 128$ for today's recommended security level of 128 bits). Then, it garbles every gate in the circuit by encrypting the labels corresponding to the plaintext value of the output wire using the labels corresponding to the values of the input wires. Suppose that a gate g has two input wires w_0 and w_1 with bit values a and b, and one output wire w_2 with the resulting bit value $g(a,b)$, then, for each possible input value combination, the entry $enc_{k_a^{w_0}}(enc_{k_b^{w_1}}(k_{g(a,b)}^{w_2}))$ is added to the garbled gate $\tilde{g}$, using a symmetric encryption function enc (e.g., based on AES). Finally, the garbler also randomly permutes the entries in $\tilde{g}$. Given $\tilde{g}$, $k_a^{w_0}$, and $k_b^{w_1}$, one can evaluate the garbled gate by trying to decrypt every entry and eventually obtaining $k_{g(a,b)}^{w_2}$, the label corresponding to the plaintext gate output corresponding to the evaluated inputs. [2] In Yao's protocol, the garbled circuit of f is transmitted to the second party, which is called the evaluator. The labels are transferred using oblivious transfer so that the evaluator does not obtain any knowledge about the label-value assignments and that the garbler does not obtain any knowledge about the evaluator's input. Using the garbled circuit and the labels, the evaluator can securely evaluate the entire circuit and obtain the labels corresponding to the circuit output when run on both parties' inputs. To reveal the output, the garbler communicates the assignments of the output labels to the evaluator. Thus, Yao's protocol requires a constant number of communication rounds.

After a fruitful line of research into protocol optimizations (Kolesnikov & Schneider, 2008; Zahur et al., 2015) and OT improvements (Ishai et al., 2003; Asharov et al., 2013), Yao's GC requires no communication per XOR gate (Kolesnikov & Schneider, 2008) and just 2κ bits of communication per AND gate (Zahur et al., 2015). In the notation of Demmler et al. (2015b), an execution of Yao's GC can also be seen as operating on secret shares. Basically, Yao sharing of a secret bit a depends on the corresponding labels, with $\langle a \rangle_0^Y = (k_0, k_1)$ and $\langle a \rangle_1^Y = k_a$. The sharing procedure itself is based on OT, and the original bit a can be efficiently reconstructed from $\langle a \rangle_0^Y$ and $\langle a \rangle_1^Y$.

4.3. Mixed Protocols

The different STPC protocols vary in efficiency for different types of data and applications. For example, GC only has a constant number of rounds while computation in arithmetic sharing has rounds linear in the multiplicative depth of the circuit, making GC a better choice in settings with high latency. Also, comparisons usually are more efficient in Yao sharing. On the other hand, computing arithmetic operations in arithmetic sharing results in a relatively low

[2] The encryption scheme needs to have the special property to check for correct decryption or additional techniques like *point-and-permute* (Beaver et al., 1990) are employed.



1. Client extracts probe $\boldsymbol{X}$.
2. Client shares $\boldsymbol{X} = \langle \boldsymbol{X}\rangle_0^A + \langle \boldsymbol{X}\rangle_1^A$.
3. Client sends $\langle \boldsymbol{X}\rangle_0^A$ to P_0 and $\langle \boldsymbol{X}\rangle_1^A$ to P_1.
4. Servers P_0 and P_1 securely compute:
    - 4.1. $\langle S_{cos}(\boldsymbol{X}, \boldsymbol{Y})\rangle^A$
    - 4.2. $\langle S_{cos}(\boldsymbol{X}, \boldsymbol{Y})\rangle^Y = A2Y(\langle S_{cos}(\boldsymbol{X}, \boldsymbol{Y})\rangle^A)$
    - 4.3. $\langle \textbf{yes}/\textbf{no}\rangle^Y = \tilde{>}(\langle S_{cos}(\boldsymbol{X}, \boldsymbol{Y})\rangle^Y, \eta)$
5. P_1 performs $Reconstruct(\langle \textbf{yes}/\textbf{no}\rangle^Y) \rightarrow$**yes/no**.

Figure 1. Our architecture of ASV between a client device and two servers P_0 and P_1 based on cosine similarity for length-normalized features. The data subject is already enrolled with reference embedding share $\langle \boldsymbol{Y}\rangle_0^A$ stored on P_0 and reference embedding share $\langle \boldsymbol{Y}\rangle_1^A$ stored on P_1.

communication overhead. Therefore, it is a good idea to use multiple primitives during the secure evaluation of a function, harnessing the optimal protocol for different parts of the computation. For this, the ABY framework (Demmler et al., 2015b) provides functionalities to convert a sharing of one type to a valid sharing of another type. For example, the $A2Y$ function converts an arithmetic sharing to a Yao sharing: $\langle x\rangle^Y = A2Y(\langle x\rangle^A)$. Afterwards, the rest of the function can be evaluated as a GC. We will make use of $A2Y$ in our architecture by first computing the score in arithmetic sharing and then converting it to Yao sharing to perform the threshold comparison. For an l-bit x and security parameter κ, $A2Y$ incurs costs of $13l$ symmetric encryption operations and $5l\kappa$ bits of communication in two messages.

5. Our STPC Architecture for Automatic Biometric Verification

Our goal is to achieve biometric information protection (BIP) in ASV using STPC. Although STPC lets us securely compute any kind of biometric distance, the adaption of STPC to BIP is not straightforward. The issue is that in the STPC setting, the inputs x and y are assumed to be available in plain to the corresponding party. While the input is hidden from the other party, we additionally require that the reference data $\boldsymbol{Y}$ is protected on the server and is thus not available as an input to an STPC protocol. Our proposed architecture operates in the same setting as the previous work of Nautsch et al. (2018) by assuming that two servers P_0 and P_1 perform the verification without colluding. In Nautsch et al. (2018), the server DB$_{controller}$ is used as a storage service for the encrypted embedding while the server AS$_{operator}$ is in possession of the secret key used to decrypt the final score. Conversely, in our solution, P_0 and P_1 both participate in a STPC protocol to perform the verification, with an authenticating server P_1 obtaining the verification decision at the end of the computation. We consider this to be a realistic setting, given that P_0 could be seen as an independent provider of privacy-preserving computation services that is employed to assist with the verification procedure of P_1. As such, preserving trust in the provided service is of importance to P_0, resulting in an economic incentive to not collude with P_1.

Our architecture is based on the outsourced STPC construction of Kamara & Raykova (2011) and aligns STPC to biometric verification by *pre-sharing* $\boldsymbol{X}$ and $\boldsymbol{Y}$ between P_0 and P_1. Similar to Blanton & Aliasgari (2012) in the context of iris identification using Boolean sharing, the client device secretly shares the reference $\langle \boldsymbol{Y}\rangle_i = \{\langle y_1\rangle_i, \ldots, \langle y_F\rangle_i\}$ during enrolment, and sends $\langle \boldsymbol{Y}\rangle_0$ to P_0 and $\langle \boldsymbol{Y}\rangle_1$ to P_1. This setup satisfies the three ISO/IEC IS 24745 requirements of unlinkability, irreversibility, and renewability. The data is secretly shared and, hence, it is impossible to retrieve any information about the original input, granting irreversibility. For the same reason, the shares of subjects' samples do not reveal any information about the subject, making it impossible to detect whether the samples belong to the same subject, guaranteeing unlinkability. Since adding a fresh sharing of the neutral element to an existing sharing

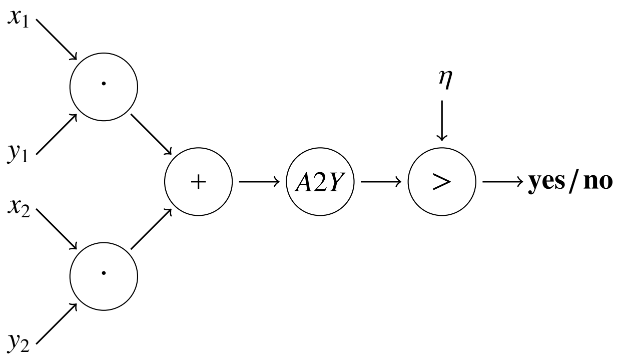


Figure 2. Example of a length-normalized cosine verification circuit for an embedding dimension of $F = 2$.

yields a new valid share in arithmetic or Boolean sharing, our setup is easily renewable without the need to re-enroll. This concept allows information protection for any biometric verification system, as the verification itself can then be computed by invoking STPC protocols on the shares. Of course, communication between the client and the servers needs to be encrypted and authenticated, e.g., by using a secure channel via TLS. Otherwise, an adversary with access to the network could just reconstruct the biometric reference data by observing both shares sent to the different servers. In the following, we present a mixed-protocol STPC verification system optimized for embedding-based ASV.

5.1. Cosine Verification

Our architecture as well as the verification process is illustrated in Figure 1 for the special case of ASV using a cosine comparison score on length-normalized embeddings. Prior to verification, the data subject enrolls its length-normalized embedding $\langle \boldsymbol{Y} \rangle_0^A$ to P_0 and $\langle \boldsymbol{Y} \rangle_1^A$ to P_1. Then, during verification, the client device shares the probe vector $\langle \boldsymbol{X} \rangle_i^A = \{\langle x_1 \rangle_i^A, \ldots, \langle x_F \rangle_i^A\}$ (step 2), and sends $\langle \boldsymbol{X} \rangle_0^A$ to P_0 and $\langle \boldsymbol{X} \rangle_1^A$ to P_1 (step 3). In step 4, P_0 and P_1 engage in a mixed STPC protocol to compute the verification on the already shared inputs. Even though P_0 and P_1 did not share the inputs themselves, they can still securely compute the verification function as long as their shares are valid.

More specifically, P_0 and P_1 first securely compute $S_{cos}(\boldsymbol{X}, \boldsymbol{Y})$ in step 4.1 by using $\langle \boldsymbol{X} \rangle_i^A$ and $\langle \boldsymbol{Y} \rangle_i^A$. Subsequently, in order to perform the comparison, the parties switch from arithmetic to Yao sharing and compute $\langle S_{cos} \rangle_i^Y = A2Y(\langle S_{cos}(\boldsymbol{X}, \boldsymbol{Y}) \rangle_i^A)$ in step 4.2. In step 4.3, the threshold comparison is computed. A greater than gate $>$ is garbled into $\tilde{>}$ and the servers evaluate $\tilde{>}(S_{cos}, \eta)$ on the Yao-shared S_{cos} and a previously-shared or just publicly known threshold value η. Finally, P_0 sends its share of the decision to P_1, which is then able to recover the decision value by combining both shares using *Reconstruct* in step 5. Based on this, the data subject is either accepted or denied. The resulting circuit that is securely evaluated during the evaluation is shown in Figure 2 for a dimension of $F = 2$.

A complexity analysis of the HE solution of Nautsch et al. (2018) and our cosine architectures is presented in Table 1. At the cost of just two additional rounds stemming from the $A2Y$ conversion, the outsourced STPC solution achieves significantly smaller bandwidth and storage costs while replacing the expensive exponentiations of homomorphic cryptography with simple products and symmetric cryptography. For contemporary CPUs, symmetric AES operations are much more efficient because AES is computed via native CPU instructions. Also, the entirety of computation and communication takes place between two servers, which we assume to possess hardware with reasonable performance and a low-latency network connection. This is not the case for the HE solution, which necessitates that the client, which may even be a mobile device with power and bandwidth restrictions, does most of the computation. Apart from efficiency improvements, we designed our architecture to additionally protect the score and threshold values as well as to grant security against a malicious client that would deviate from the protocol specification. The last advantage stems from the fact that in the HE solution (Nautsch et al., 2018), a malicious client could just encrypt an accepting score S'_{cos} and AS$_{operator}$ would accept it. The reason for this security risk is that the designed protocol has no way to prove whether the score was computed in the specified way. In contrast, in our outsourced STPC-based solution all secret shared inputs, and even maliciously chosen inputs, correspond to a valid speech input of the client and therefore our protocol provides security against a malicious client. The score computation takes place between both semi-honest servers without involving the potentially malicious client. Contrary to the servers, the client has no

		Cosine		PLDA/2Cov	
		(Nautsch et al., 2018)	Ours	(Nautsch et al., 2018)	Ours
Computation	#asymmetric crypto	1	0	$3F^2 + 1$	0
	#exponentiations	F	0	$4F^2$	0
	#symmetric crypto	0	$15l$	0	$15l$
	#products	$F - 1$	$8F$	$5F^2 - 1$	$8(4F^2 + 5F)$
	#additions	0	$16F - 2$	0	$64F^2 + 74F$
Communication	Between client & servers	$2\lambda(F + 1)$	$2lF$	$2\lambda(5F^2 + F + 1)$	$2lF$
	Between servers	0	$l(4F + 5\kappa)$	0	$l(16F^2 + 20F + 5\kappa)$
	Total	≈ 193KB	≈ 17.1KB	≈ 240.2MB	≈ 8MB
	#rounds	1	3	2	4
Storage	Protected template size	$2\lambda F$ ≈ 192KB	lF ≈ 2KB	$2\lambda(F^2 + F)$ ≈ 48.2 MB	lF ≈ 2 KB
	Protected model size			$4\lambda F^2$ ≈ 96 MB	$l(2F^2 + F + 1)$ ≈ 1 MB

Table 1. Complexity analysis for the cosine and PLDA-style comparators during online verification, assuming $F = 250$ dimensional features, feature size $l = 64$ bits and long-term (recommended use until 2030 and beyond) secure key sizes as recommended by Barker (2016) (symmetric security parameter $\kappa = 128$, and public-key security parameter $\lambda = 3072$).

incentive to behave honestly and is not under public scrutiny. Thus, in addition to preserving the subject's privacy, we consider security against malicious clients to be another important property in automatic biometric verification.

5.2. PLDA Verification

PLDA comparators examine the similarity and dissimilarity of a reference – probe pair in a latent biometric subspace (in terms of a log-likelihood ratio score), considering within and between variabilities of the biometric data. That is, not in the observed data domain but in the inferred biometric domain, where the actual *biometric* comparison is carried out. By contrast, cosine comparators solely examine the data correlation: extracted features need to be representative regarding the biometric verification task in order to sustain a biometric performance competitive to PLDA. More accurate embedding verification systems rely on PLDA/2Cov comparators. The HE solution of Nautsch et al. (2018) was also extended to securely evaluate the PLDA/2Cov score. Compared to S_{cos}, those scores rely on trained models provided by a *vendor*. Since these models usually encompass intellectual property that is very valuable to the vendor, the protection of those models is considered an important goal and has been realized by the previous solutions. However, adapting the basic version for securely computing cosine S_{cos} to PLDA/2Cov scores S_{PLDA} comes with significant changes in the HE-based architecture: Nautsch et al. (2018) introduce two additional servers AS_{vendor} and DB_{vendor}, requiring an additional asymmetric key pair. We use the advantages of STPC and can easily adapt our cosine solution by just securely evaluating a circuit that performs the S_{PLDA} instead of the S_{cos} computation, with an additional model input coming from the vendor. This vendor data can be protected in the same way as the subject embedding by pre-sharing it between P_0 and P_1. Thus, PLDA/2Cov scores for centered data $\langle S^{\mathbf{0}}_{PLDA}\rangle^A$ can be easily computed on pre-shared $\langle \boldsymbol{X}\rangle^A$, $\langle \boldsymbol{Y}\rangle^A$, $\langle \boldsymbol{P}\rangle^A$, $\langle \boldsymbol{Q}\rangle^A$, and $\langle const\rangle^A$ as specified in Section 3. Analogously, PLDA/2Cov scores for non-centered data $\langle S^{\boldsymbol{\mu}}_{PLDA}\rangle^A$ can be easily computed on $\langle \boldsymbol{X}\rangle^A$, $\langle \boldsymbol{Y}\rangle^A$, $\langle \boldsymbol{\Lambda}\rangle^A$, $\langle \boldsymbol{\Gamma}\rangle^A$, $\langle \boldsymbol{c}\rangle^A$, and $\langle k\rangle^A$. The threshold comparison in Yao-sharing then stays the same as in the cosine solution.

Table 1 also provides an overview of the complexities of the PLDA-style solution of Nautsch et al. (2018) and ours. Compared to theirs, our outsourced STPC approach greatly reduces bandwidth and storage requirements. In the case of $F = 250$ and standard key sizes, our bandwidth of 8 MB is more practical than the 240 MB of the HE solution. Also, our computational costs are much smaller, requiring no exponentiations. Our solution requires two additional rounds but these costs are insignificant since we assume a low latency network between the two servers, which is often not the case in the setting of the other architecture, where the client is directly involved in the secure score computation.

6. Evaluation

We evaluate our proposed ASV architecture on the NIST i-vector machine learning challenge (National Institute of Standards and Technology (NIST), 2014). The data comprises 1 306 reference identities (each with five reference samples) and 9 634 probes, forming a progress and an evaluations set. Comparison models are trained on a segregated development set (with 64-bit float values). We report on the evaluation set with 7 542 271 comparisons. The i-vectors (probabilistic embeddings) are distributed with 5-digit precision; our computational comparisons assume pre-processed embeddings, e.g., after linear discriminant analysis (LDA), whitening, and length-normalization. Our C++ implementation of the privacy-preserving cosine and PLDA/2Cov verification procedures is based on the ABY framework (Demmler et al., 2015b) which provides a state-of-the-art implementation of the arithmetic, Boolean, and Yao STPC protocols. Our implementation is available online as open source at `https://encrypto.de/code/PrivateASV`. Using our prototype, we evaluate the real-world runtime and biometric performance of our architecture and compare the results to the HE-based solution of Nautsch et al. (2018), which is set in the same context of two-server device-independent embedding-based ASV.

6.1. Implementation Details

For the secure computation of a functionality, ABY requires a specification of the circuit of the functionality to be computed securely. Our program provides circuit descriptions of the cosine and PLDA/2Cov score computations. For Boolean and Yao sharing, floating point inputs and operations are provided by ABY (Demmler et al., 2015a), whereas arithmetic sharing operates on integer inputs. While we will also report results for a full floating point solution in Boolean sharing, we focus on the more efficient arithmetic sharing in $\mathbb{Z}_{2^{64}}$ by scaling the floating point inputs into the integer space in order to obtain practical runtimes. In this scaled solution, we multiply the floating point values by 10^5 and we will show in Section 6.2 that this does not represent a loss in verification accuracy.

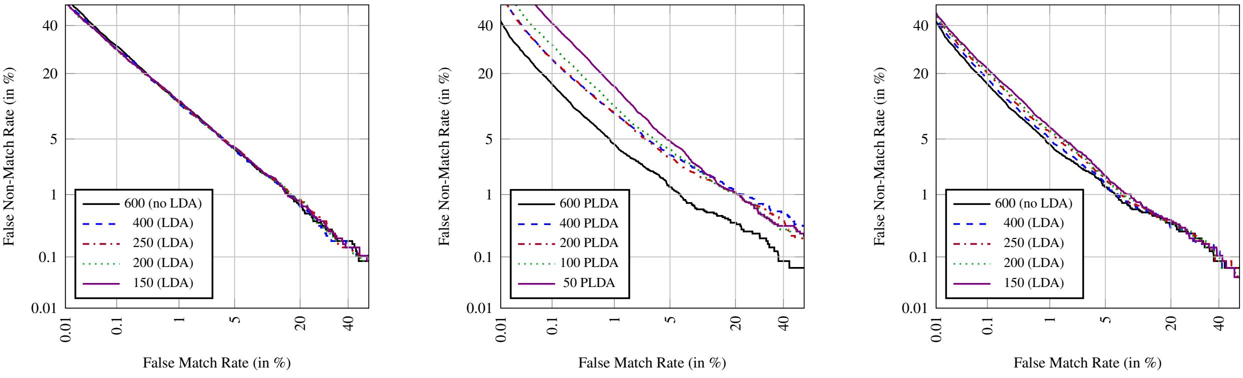
Additionally, for our scaled solution, we need to represent negative values in $\mathbb{Z}_{2^l}$. Using the two's complement, we are able to represent negative integers in $\mathbb{Z}_{2^l}$ without adapting the secure computation of multiplication or addition gates. However, we had to modify the greater-than comparator in Yao sharing invoked during the threshold comparison after the $A2Y$ operation is performed on the computed score. Adopting the optimizations for three-input functionalities of Pinkas et al. (2009); Schneider (2012) yields a two's complement greater-than comparator circuit requiring just one additional AND gate compared to the comparison circuit for unsigned integers (Kolesnikov et al., 2009).

Furthermore, we use *single instruction multiple data* (SIMD) operations, enabling the parallel execution of one operation on multiple data items like, e.g., a multiplication to realize a vector product. This can greatly speed up the evaluation time of the verification circuit.

6.2. Biometric Performance

Figures 3 and 4 depict detection error trade-off plots (Martin et al., 1997) (implementation based on Nautsch et al. (2017)), comparing the biometric verification performance on different feature dimensions of cosine, PLDA, and 2Cov comparators as well as on the plaintext and the proposed scaled STPC implementations (the full floating point implementation has no degradation compared to the plaintext execution). Though our STPC implementation of PLDA/2Cov for centered and non-centered data is the same for both PLDA and 2Cov comparators (only the estimation of model parameters given as inputs vary), the biometric performances differ. While cosine comparisons are solely slightly affected by the feature space dimension, PLDA comparisons degrade the fewer information is provided, outperforming cosine on higher feature space dimension. The full-subspace PLDA, i.e., the 2Cov comparator, outperforms cosine and other PLDA comparators. Our proposed STPC implementations for cosine, PLDA, and 2Cov comparators yield the same error rate trade-off as their plaintext *Sidekit* implementation (as of Larcher et al. (2016)). In contrast to Larcher et al. (2016), we assume already pre-computed comparison model parameters, and fully processed embeddings. Score properties and distributions are preserved within our proposed STPC computation (to a reasonable extent). For all comparators, the error rate trade-off characteristics are the same when using the proposed scaled privacy-preserving method (regardless of the feature dimensions) and thus biometric recognition performance is sustained while privacy is preserved.

Notably, we observed minor differences in the score value of the Sidekit and the scaled ABY implementations, which we attribute to differences in the underlying Python and C++ libraries as well as to numerical artifacts. In particular, computations are carried out on integer rather than float values (derived by multiplication with 10^5 and



(a) Cosine

(b) PLDA

(c) 2Cov

Figure 3. Biometric performance of (from left to right) cosine, PLDA, and 2Cov comparators with different feature dimensions (without and after LDA dimension reduction): 600 (black, solid), 400 (blue, dashed), 250 (red, dash-dotted), 200 (green, dotted), 150 (violet, solid). For PLDA, 600 dimensional features are depicted, where the colors indicate subspace dimension: 600 (black, solid), 400 (blue, dashed), 200 (red, dash-dotted), 100 (green, dotted), 50 (violet, solid). In this set-up, the full-subspace 600-dimensional PLDA equals the 2Cov on 600 feature dimensions.

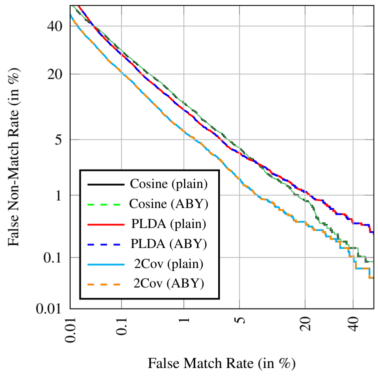


Figure 4. Biometric performance of plaintext and ABY (solid, dashed) implementations of cosine (black / green), PLDA (red / blue) with 100-dimensional subspace, and 2Cov (cyan / orange) comparators on 200-dimensional embeddings; no score normalization employed.

truncation). This multiplicative domain representation of a float value leads to different spaces before and after multiplications are securely evaluated in our implementation. The integer values are preserved within the same domain, when sustaining the same multiplication factor e.g., for the computation of 2Cov scores, the inner Frobenius terms utilizing the $\mathbf{\Gamma}$ and $\mathbf{\Lambda}$ hyperparameters are in the 10^{15} domain (due to two dot products), the term utilizing $\boldsymbol{c}$ is in the 10^{10} domain (due to one dot product), and the scalar term remains in the 10^{5} domain. Partial score terms are added, each of them being in the 10^{15} domain. As a result of these approximations in our scaled implementation, the LLR score precision is limited to 3 digits (5-digit input, up to two multiplications in the summation of score subterms). By consequence, LLR thresholds are also limited to 3-digit precision, whereas the HE-based implementation provides full precision to verification service providers when denoting LLR thresholds. [3] This is a result of the scaling used to speed up transaction times and is not inherent to STPC. However, we assume that even 2-digit precision on LLR

[3] By limiting LLR threshold precision, formally denoted decision policies are grouped. Decision policies resemble in (LLR) thresholds, which are formally outlined by prior probability and decision cost beliefs (Brümmer, 2010). In this sense, LLR thresholds are continuous as they are capable of summarizing different parameterizations of prior and cost beliefs. If we limit the precision of these LLR thresholds, summaries of prior and cost beliefs are grouped; the scale of possible decisions using LLR scores is limited. If one evaluates the information-theoretic performance of binary classifiers (see Brümmer & du Preez (2008); Ramos & Gonzalez-Rodrigues (2008)), logarithmic (strictly) proper scoring rules are employed that penalize, e.g., higher non-mated comparison scores much more than lower non-mated comparison scores. By limiting the precision, scores are effectively rounded. We assume the difference in information performance resulting from third precision rounding to be marginal for LLRs in [−5, +5], which we assume to be reasonable LLR ranges for current biometric voice recognition services.

	Cosine			PLDA/2Cov (non-centered Data)				PLDA/2Cov (Centered Data)
Dimension	HE	Ours	Improvement	HE (Plain Model)	HE (Protected Model)	Ours	Improvement	Ours
$F = 50$	22 ms	3.7 ms	6×	58 ms	15.9 s	11.2 ms	1 421×	10.5 ms
$F = 100$	44 ms	3.7 ms	12×	96 ms	61.7 s	26.6 ms	2 318×	19.9 ms
$F = 150$	62 ms	3.8 ms	16×	122 ms	135.7 s	45.1 ms	3 009×	39.8 ms
$F = 200$	84 ms	3.9 ms	22×	169 ms	241.9 s	77.4 ms	3 126×	52.8 ms
$F = 250$	106 ms	4.0 ms	27×	205 ms	379.6 s	113.7 ms	3 338×	84.2 ms
$F = 400$	144 ms	4.0 ms	36×	320 ms	973.1 s	246.7 ms	3 945×	205.6 ms
$F = 600$	203 ms	4.2 ms	47×	423 ms	2 171.6 s	529.8 ms	4 099×	411.4 ms

Table 2. Experimental online runtimes of our outsourced STPC verification architecture and the HE verification architecture of Nautsch et al. (2018) for cosine and PLDA/2Cov scores with either centered or non-centered data. For PLDA/2Cov scores, computational time is saved by centering the data before comparisons, however, for accompanying dataset shifts, one might want to employ another data mean value. Our STPC verification is implemented in C++ using ABY (Demmler et al., 2015b) while the HE solution is implemented in Python using Python-pallier (Thorne, 2017). The experiments were run on $l = 64$ bit scaled integer inputs, with long-term (recommended use until 2030 and beyond) secure key sizes as recommended by Barker (2016) (symmetric security parameter $\kappa = 128$ and public-key security parameter $\lambda = 3072$).

thresholds satisfies current technology demands and, indeed, no loss in the biometric recognition performance is observed in this evaluation. Thus, biometric verification performance is retained in our solution. From Figures 3 and 4, we conclude that not only comparative but identical performance results are sustained.

6.3. Workload Evaluation

We tested both our scaled and our full floating point implementation on a two-server infrastructure using two physically separated machines with Intel Core i9-7960X CPUs and 128 GB of RAM. Since we want to emulate well-connected service providers, the network connection between the servers is configured with a bandwidth of 1 Gbit and 1 ms round trip time.

6.3.1. Scaled Solution

Table 2 provides an overview of the measured transaction times of our scaled solution. We consider only the time of the verification itself without the client starting the process. The pre-computation time is not relevant to the verification time itself and is therefore not included. A comparison to the HE-based verification of Nautsch et al. (2018) can also be found in Table 2, including both the solution that assumes knowledge of the model in plain and the solution that additionally hides the model. Since our solution also hides the model, which is valuable intellectual property, the runtimes have to be compared to the latter. Their implementation is written in Python using Python-paillier (Thorne, 2017) that uses efficient C-coded multiple-precision arithmetic and can operate in different ways regarding the limitation of precision. Either a fixed value for all exponents can be agreed upon, which is equivalent to our scaled solution, or the exponent terms of floating point values are communicated in plain, which leads to some information leakage about the magnitude of the encrypted value (Thorne, 2017). The implementation of Nautsch et al. (2018) uses the latter option. Note that in the setting of Nautsch et al. (2018), parts of the computation have to be performed on the client device, though we benchmark the whole computation on one server to yield a useful comparison between the architectures.

The results show that the usage of outsourced STPC for embedding-based ASV significantly improves verification duration. Depending on the feature dimension F, a verification can take as little as 3.7 ms (cosine), 10.5 ms (centered data input to PLDA/2Cov), or 11.2 ms (non-centered data input to PLDA/2Cov). Even with the unaltered embeddings of $F = 600$, our PLDA/2Cov verifications require roughly half a second to complete. For the cosine distance, increasing the feature dimension only slightly increases the verification time while an increased dimension results in a higher verification time for PLDA/2Cov. Compared to the HE-based solution, our verification time is significantly improved. The HE cosine verification takes between 22 ms ($F = 50$) and 203 ms ($F = 600$) whereas ours takes between 3.7 ms and 4.2 ms, respectively. Looking at the factor of improvement over increasing values of F, one can observe that our solution also scales better for larger F. For having non-centered data inputs to PLDA/2Cov comparisons, our solution yields a similar result: the factor of improvement goes as high as 4099× ($F = 600$) and is increasing for increasing F. Though we only compare the online times here, we note that the total execution time (consisting of the setup and the online time) of our architecture is practical as well. For instance, the total evaluation time varies from 191.8 ms ($F = 50$) to 19.4 s ($F = 600$) for non-centered data, the latter still being an improvement of 112× over the HE-based solution. Even compared to the HE solution that leaks the vendor's model to allow for much more efficient runtimes compared to the HE solution that protects this information, our solution can still achieve considerably lower runtimes (for the most part) while nonetheless leaking no information about the model. Overall, this evaluation shows that our solution can also be viable for higher feature dimensions, effectively reducing the HE verification time for $F = 600$ from about 36 minutes to about half a second.

6.3.2. Full Floating Point Solution

Though we showed in Section 6.2 that our scaled implementation does not alter verification accuracy, we also evaluated full floating point operations (with 32-bit and 64-bit precision). Compared to our scaled solution, this displays a significant drawback in efficiency. For 32-bit precision, our implementation of PLDA/2Cov (non-centered data) takes between 11.0 s ($F = 50$), 76.1 s ($F = 150$), and 232.5 s ($F = 250$), while the runtimes for 64-bit precision are between 21.3 s ($F = 50$) and 202.5 s ($F = 150$). Notably, the circuits for native floating point operations require a much higher gate count and therefore a high amount of allocated memory for one verification. As a result, 32-bit operations become infeasible in our setting for $F = 400$ and greater (32-bit) and for $F = 200$ and greater (64-bit). Because of this, the main communication bottleneck is the high depth of the circuit introduced by the floating point operations and therefore the cosine verification runtimes take considerably longer as well (between 3.2 s and 38.0 s for 64-bit precision). Though the full floating point PLDA/2Cov transaction times are still lower than the HE-based solution of Nautsch et al. (2018) for 32-bit precision, the 64-bit precision times are not suitable for ASV. While feature extraction may operate in 64-bit, we assume that the biometric comparison can be executed in 32-bit, because just a few more additions and multiplications occur with each data value to result in LLR scores. Nonetheless, we propose to use scaling for practical privacy-preserving ASV, since our scaled solution is significantly faster without displaying a decrease in verification accuracy.

7. Conclusion

In our work we propose a privacy-preserving automatic biometric verification architecture based on outsourced STPC. We showed that the outsourced biometric data is unlinkable, irreversible, and renewable according to the ISO/IEC 24745 standard. As an important use case, we demonstrate how to build an embedding-based privacy-preserving ASV by employing mixed-protocol outsourced STPC for cosine and PLDA/2Cov score computations and threshold comparisons. A theoretical as well as a practical evaluation on the NIST i-vector challenge shows that our architecture retains biometric recognition performance while achieving practical transaction times. Our architecture is over 4000× faster than previous solutions and, in addition, is secure against a malicious client. Since we also do not require a device-specific key, we argue that the usage of outsourced STPC is a demonstrable and practical tool to satisfy recent legal restrictions for automatic biometric verification. To avoid limiting threshold precision through scaling floating point values to integers while still achieving highly efficient transaction times, we consider more efficient secure floating point operations as important future work.

Acknowledgements

We want to thank the anonymous reviewers for their helpful feedback and comments. This work was supported by: the DFG as part of projects A.1 within the RTG 2050 "Privacy and Trust for Mobile Users" and E4 within the CRC 1119 CROSSING; the German Federal Ministry of Education and Research (BMBF) and the Hessen State Ministry for Higher Education, Research and the Arts (HMWK) within CRISP; within the BioMobile II project (no. 518/16-30); the German Federal Office for Information Security within the research project DIRECT-PAD; Omilia – Conversational Intelligence, (Declarations of interest: none.)

References

Adler, A. (2003). Sample images can be independently restored from face recognition templates. In *Proc. Can. Conf. on Electrical and Computer Engineering (CCECE)* (pp. 1163–1166). IEEE.

Aguilar-Melchor, C., Fau, S., Fontaine, C., Gogniat, G., & Sirdey, R. (2013). Recent advances in homomorphic encryption: A possible future for signal processing in the encrypted domain. *IEEE Signal Proc. Mag.*, *30*, 108–117.

Aliasgari, M., & Blanton, M. (2013). Secure computation of hidden Markov models. In *Proc. Intl. Conf. on Security and Cryptography (SECRYPT)* (pp. 1–12). IEEE.

Aliasgari, M., Blanton, M., & Bayatbabolghani, F. (2017). Secure computation of hidden Markov models and secure floating-point arithmetic in the malicious model. *Intl. J. of Information Security*, *16*, 577–601.

Asharov, G., Lindell, Y., Schneider, T., & Zohner, M. (2013). More efficient oblivious transfer and extensions for faster secure computation. In *Proc. ACM SIGSAC Conf. on Computer & Communications Security (CCS)* (pp. 535–548). ACM.

Bansé, D., Doddington, G. R., Garcia-Romero, D., Godfrey, J. J., Greenberg, C. S. et al. (2014). Summary and initial results of the 2013-2014 speaker recognition i-vector machine learning challenge. In *Proc. Annual Conf. of the Intl. Speech Communication Association (INTERSPEECH)* (pp. 368–372). ISCA.

Barker, E. (2016). NIST special publication 800–57 part 1, revision 4.

Barni, M., Bianchi, T., Catalano, D., Di Raimondo, M., Labati, R. D. et al. (2010). A privacy-compliant fingerprint recognition system based on homomorphic encryption and fingercode templates. In *Proc. Intl. Conf. on Biometrics: Theory Applications and Systems (BTAS)* (pp. 1–7). IEEE.

Barni, M., Droandi, G., & Lazzeretti, R. (2015). Privacy protection in biometric-based recognition systems: A marriage between cryptography and signal processing. *IEEE Signal Proc. Mag.*, *32*, 66–76.

Beaver, D. (1991). Efficient multiparty protocols using circuit randomization. In *Proc. Annual Cryptology Conf. (CRYPTO)* (pp. 420–432). Springer.

Beaver, D., Micali, S., & Rogaway, P. (1990). The round complexity of secure protocols. In *Proc. ACM Symposium on Theory of Computing (STOC)* (pp. 503–513). ACM.

Ben-Or, M., Goldwasser, S., & Wigderson, A. (1988). Completeness theorems for non-cryptographic fault-tolerant distributed computation. In *Proc. Symposium on Theory of Computing (STOC)* (pp. 1–10). ACM.

Bianchi, T., Turchi, S., Piva, A., Labati, R. D., Piuri, V., & Scotti, F. (2010). Implementing fingercode-based identity matching in the encrypted domain. In *Proc. IEEE Workshop on Biometric Measurements and Systems for Security and Medical Applications (BIOMS)* (pp. 15–21). IEEE.

Blanton, M., & Aliasgari, M. (2012). Secure outsourced computation of iris matching. *J. of Computer Security (JoCS)*, (pp. 259–305).

Blanton, M., & Gasti, P. (2011). Secure and efficient protocols for iris and fingerprint identification. In *Proc. European Symposium on Research in Computer Security (ESORICS)* (pp. 190–209). Springer.

Brasser, F., Frassetto, T., Riedhammer, K., Sadeghi, A.-R., Schneider, T., & Weinert, C. (2018). VoiceGuard: Secure and private speech processing. In *Proc. Annual Conf. of the Intl. Speech Communication Association (INTERSPEECH)* (pp. 1303–1307). ISCA.

Bringer, J., Chabanne, H., Favre, M., Patey, A., Schneider, T., & Zohner, M. (2014). GSHADE: Faster privacy-preserving distance computation and biometric identification. In *Proc. ACM Workshop on Information Hiding and Multimedia Security (IH&MMSec)* (pp. 187–198). ACM.

Bringer, J., Chabanne, H., & Patey, A. (2013). Privacy-preserving biometric identification using secure multiparty computation: An overview and recent trends. *IEEE Signal Proc. Mag.*, *30*, 42–52.

Brümmer, N. (2008). Optimization of the accuracy and calibration of binary and multiclass pattern recognizers, for wide ranges of applications. http://arantxa.ii.uam.es/~jms/seminarios_doctorado/abstracts2007-2008/20070226NBrummer.html. Last accessed: 2018-12-18.

Brümmer, N. (2010). *Measuring, refining and calibrating speaker and language information extracted from speech*. Ph.D. thesis University of Stellenbosch.

Brümmer, N., van Leeuwen, D., & Swart, A. (2014). A comparison of linear and non-linear calibrations for speaker recognition. In *Proc. The Speaker and Language Recognition Workshop (Odyssey)* (pp. 14–18). ISCA.

Brümmer, N., & du Preez, J. (2008). Application-independent evaluation of speaker detection. *Computer Speech and Language*, *20*, 230–275.

Brümmer, N., & de Villiers, E. (2010). The speaker partitioning problem. In *Proc. The Speaker and Language Recognition Workshop (Odyssey)* (pp. 194–201). ISCA.

Campisi, P. (2013). *Security and Privacy in Biometrics*. Springer.

Cappelli, R., Maio, D., Lumini, A., & Maltoni, D. (2007). Fingerprint image reconstruction from standard templates. *IEEE Trans. Pattern Anal. Mach. Intell. (TPAMI)*, *29*, 1489 – 1503.

Chun, H., Elmehdwi, Y., Li, F., Bhattacharya, P., & Jiang, W. (2014). Outsourceable two-party privacy-preserving biometric authentication. In *Proc. ACM ASIA Conf. on Computer and Communications Security (ASIACCS)* (pp. 401–412). ACM.

Costan, V., & Devadas, S. (2016). Intel SGX explained. *IACR Cryptology ePrint Archive*, *2016*, 086.

Cumani, S., Brümmer, N., Burget, L., Laface, P., Plchot, O., & Vasilakakis, V. (2013). Pairwise discriminative speaker verification in the i-vector space. *IEEE/ACM Trans. Audio, Speech, Language Process. (TASLP)*, *21*, 1217–1227.

Cumani, S., & Laface, P. (2014). Generative pairwise models for speaker recognition. In *Proc. The Speaker and Language Recognition Workshop (Odyssey)* (pp. 273–279). ISCA.

Dehak, N., Kenny, P. J., Dehak, R., Dumouchel, P., & Ouellet, P. (2011). Front-end factor analysis for speaker verification. *IEEE/ACM Trans. Audio, Speech, Language Process. (TASLP)*, *19*, 788–798.

Demmler, D., Dessouky, G., Koushanfar, F., Sadeghi, A.-R., Schneider, T., & Zeitouni, S. (2015a). Automated synthesis of optimized circuits for secure computation. In *Proc. ACM SIGSAC Conf. on Computer and Communications Security (CCS)* (pp. 1504–1517). ACM.

Demmler, D., Schneider, T., & Zohner, M. (2015b). ABY-A framework for efficient mixed-protocol secure two-party computation. In *Proc. Network and Distributed System Security Symposium (NDSS)*. The Internet Society.

Erkin, Z., Franz, M., Guajardo, J., Katzenbeisser, S., Lagendijk, I., & Toft, T. (2009). Privacy-preserving face recognition. In *Proc. Intl. Symposium on Privacy Enhancing Technologies (PETS)* (pp. 235–253). Springer.

European Council (2016). Directive 2016/680 of the European Parliament and of the Council on the protection of individuals with regard to the processing of personal data by competent authorities for the purposes of the prevention, investigation, detection or prosecution of criminal offences or the execution of criminal penalties, and on the free movement of such data, and repealing Council Framework Decision 2008/977/JHA.

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

Galbally, J., Ross, A., Gomez-Barrero, M., Fierrez, J., & Ortega-Garcia, J. (2013). Iris image reconstruction from binary templates: An efficient probabilistic approach based on genetic algorithms. *Comput. Vis. Image Underst. (CVIU)*, *117*, 1512–1525.

García, J. A. G., Moro-Velázquez, L., Godino-Llorente, J. I., & Castellanos-Domínguez, G. (2015). Automatic age detection in normal and pathological voice. In *Proc. Annual Conf. of the Intl. Speech Communication Association (INTERSPEECH)* (pp. 3739–3743). ISCA.

Garcia-Romero, D., & Epsy-Wilson, C. (2011). Analysis of i-vector length normalization in speaker recognition systems. In *Proc. Annual Conf. of the Intl. Speech Communication Association (INTERSPEECH)* (pp. 249–252). ISCA.

Glembek, O., Matejka, P., Plchot, O., Pesan, J., Burget, L., & Schwarz, P. (2015). Migrating i-vectors between speaker recognition systems using regression neural networks. In *Proc. Annual Conf. of the Intl. Speech Communication Association (INTERSPEECH)* (pp. 2327–2331). ISCA.

Goldreich, O., Micali, S., & Wigderson, A. (1987). How to play any mental game. In *Proc. ACM Symposium on Theory of Computing (STOC)* (pp. 218–229). ACM.

Gómez-Vilda, P., Fernández-Baillo, R., Rodellar-Biarge, V., Lluis, V. N., Álvarez-Marquina, A., Mazaira-Fernández, L. M., Martínez-Olalla, R., & Godino-Llorente, J. I. (2009). Glottal source biometrical signature for voice pathology detection. *Speech Commun.*, *51*, 759–781.

Harb, H., & Chen, L. (2005). Voice-based gender identification in multimedia applications. *J. Intell. Inf. Systems (JIIS)*, *24*, 179–198.

Hastings, M., Hemenway, B., Noble, D., & Zdancewic, S. (2019). SoK: General-purpose compilers for secure multi-party computation. In *Proc. IEEE Symposium on Security and Privacy (S&P)*. IEEE. Full version: `https://marsella.github.io/static/mpcsok.pdf`, last accessed: 2018-12-18.

Hu, S., Li, M., Wang, Q., Chow, S. S., & Du, M. (2018). Outsourced biometric identification with privacy. *IEEE Trans. Inf. Forensics Security (TIFS)*, (pp. 2448–2463).

Ishai, Y., Kilian, J., Nissim, K., & Petrank, E. (2003). Extending oblivious transfers efficiently. In *Proc. Annual Intl. Cryptology Conf. (CRYPTO)* (pp. 145–161). Springer.

ISO/IEC JTC1 SC27 Security Techniques (2011). *ISO/IEC 24745:2011. Information Technology - Security Techniques - Biometric Information Protection*. International Organization for Standardization.

Jain, A. K., Hong, L., Pankanti, S., & Bolle, R. (1997). An identity-authentication system using fingerprints. *Proc. IEEE*, *85*, 1365–1388.

Jiménez, A., & Raj, B. (2017a). Privacy preserving distance computation using somewhat-trusted third parties. In *Proc. IEEE Intl. Conf. on Acoustics, Speech and Signal Processing (ICASSP)* (pp. 6399–6403).

Jiménez, A., & Raj, B. (2017b). A two factor transformation for speaker verification through ℓ_1 comparison. In *Proc. IEEE Intl. Workshop on Information Forensics and Security (WIFS)* (pp. 1–6).

Jiménez, A., Raj, B., Portêlo, J., & Trancoso, I. (2015). Secure modular hashing. In *Proc. IEEE Intl. Workshop on Information Forensics and Security (WIFS)* (pp. 1–6).

Kamara, S., & Raykova, M. (2011). Secure outsourced computation in a multi-tenant cloud. In *Proc. IBM Workshop on Cryptography and Security in Clouds* (pp. 15–16).

Kolesnikov, V., Sadeghi, A.-R., & Schneider, T. (2009). Improved garbled circuit building blocks and applications to auctions and computing minima. In *Proc. Intl. Conf. on Cryptology and Network Security (CANS)* (pp. 1–20). Springer.

Kolesnikov, V., & Schneider, T. (2008). Improved garbled circuit: Free XOR gates and applications. In *Proc. Intl. Colloq. on Automata, Languages, and Programming (ICALP)* (pp. 486–498). Springer.

Larcher, A., Lee, K., & Meignier, S. (2016). An extensible speaker identification SIDEKIT in Python. In *IEEE Intl. Conf. on Acoustics, Speech and Signal Processing (ICASSP)* (pp. 5095–5099). IEEE. `http://lium.univ-lemans.fr/sidekit`, last accessed: 2018-12-17.

Lindell, Y., & Pinkas, B. (2009). A proof of security of Yao's protocol for two-party computation. *J. of Cryptology (JoC)*, (pp. 161–188).

Lu, R., Zhu, H., Liu, X., Liu, J. K., & Shao, J. (2014). Toward efficient and privacy-preserving computing in big data era. *IEEE Netw.*, (pp. 46–50).

Luo, Y., Sen-ching, S. C., & Ye, S. (2009). Anonymous biometric access control based on homomorphic encryption. In *Proc. IEEE Intl. Conf. on Multimedia and Expo (ICME)* (pp. 1046–1049). IEEE.

Martin, A., Doddington, G., Kamm, T., Ordowski, M., & Przybocki, M. (1997). The DET curve in assessment of detection task performance. In *Proc. European Conf. on Speech Communication and Technology (EUROSPEECH)* (pp. 1895–1898). ISCA.

McKeen, F., Alexandrovich, I., Berenzon, A., Rozas, C. V., Shafi, H., Shanbhogue, V., & Savagaonkar, U. R. (2013). Innovative instructions and software model for isolated execution. In *Proc. Workshop on Hardware and Architectural Support for Security and Privacy (HASP)*. ACM.

Meuwly, D., Ramos, D., & Haraksim, R. (2017). A guideline for the validation of likelihood ratio methods used for forensic evidence evaluation. *Forensic Sci. Int.*, *276*, 142–153.

National Institute of Standards and Technology (NIST) (2014). *The 2013-2014 Speaker Recognition i-vector Machine Learning Challenge*. Technical Report NIST.

Nautsch, A., Isadskiy, S., Kolberg, J., Gomez-Barrero, M., & Busch, C. (2018). Homomorphic encryption for speaker recognition: Protection of biometric templates and vendor model parameters. In *Proc. The Speaker and Language Recognition Workshop (Odyssey)* (pp. 16–23). ISCA.

Nautsch, A., Meuwly, D., Ramos, D., Lindh, J., & Busch, C. (2017). Making likelihood ratios digestible for cross/application performance assessment. *IEEE Signal Proc. Let.*, *24*, 1552–1556.

Osadchy, M., Pinkas, B., Jarrous, A., & Moskovich, B. (2010). SCiFi-A system for secure face identification. In *Proc. IEEE Symposium on Security and Privacy (S&P)* (pp. 239–254). IEEE.

Patel, V. M., Ratha, N., & Chellappa, R. (2015). Cancelable biometrics: A review. *IEEE Signal Proc. Mag.*, *32*, 54–65.

Pathak, M., Portelo, J., Raj, B., & Trancoso, I. (2012). Privacy-preserving speaker authentication. In *Proc. Intl. Conf. on Information Security (ISC)* (pp. 1–22). Springer.

Pathak, M., & Raj, B. (2012). Privacy preserving speaker verification as password matching. In *Proc. IEEE Intl. Conf. on Acoustics, Speech and Signal Processing (ICASSP)*.

Pathak, M., Rane, S., Sun, W., & Raj, B. (2011). Privacy preserving probabilistic inference with hidden Markov models. In *Proc. IEEE Intl. Conf. on Acoustics, Speech and Signal Processing (ICASSP)* (pp. 5868–5871). IEEE.

Pathak, M. A. (2012). *Privacy-preserving machine learning for speech processing*. Springer Science & Business Media.

Pathak, M. A., & Raj, B. (2011). Privacy preserving speaker verification using adapted GMMs. In *Proc. Annual Conf. of the Intl. Speech Communication Association (INTERSPEECH)*. ISCA.

Pathak, M. A., & Raj, B. (2013). Privacy-preserving speaker verification and identification using Gaussian mixture models. *IEEE/ACM Trans. Audio, Speech, Language Process. (TASLP)*, *21*, 397–406.

Pathak, M. A., Raj, B., Rane, S. D., & Smaragdis, P. (2013). Privacy-preserving speech processing: cryptographic and string-matching frameworks show promise. *IEEE Signal Processing Magazine*, *30*, 62–74.

Pinkas, B., Schneider, T., Smart, N. P., & Williams, S. C. (2009). Secure two-party computation is practical. In *Proc. Intl. Conf. on the Theory and Application of Cryptology and Information Security (ASIACRYPT)* (pp. 250–267). Springer.

Portêlo, J., Raj, B., Abad, A., & Trancoso, I. (2014). Privacy-preserving speaker verification using garbled GMMs. In *Proc. European Signal Processing Conf. (EUSIPCO)* (pp. 2070–2074). IEEE.

Portêlo, J., Raj, B., Boufounos, P., Trancoso, I., & Abad, A. (2013). Speaker verification using secure binary embeddings. In *Proc. European Signal Processing Conf. (EUSIPCO)*. IEEE.

Prince, S. J. D., & Elder, J. H. (2007). Probabilistic linear discriminant analysis for inferences about identity. In *Proc. Intl. Conf. on Computer Vision (ICCV)*. CVF.

Rahulamathavan, Y., Sutharsini, K., Ray, I. G., Lu, R., & Rajarajan, M. (2019). Privacy-preserving iVector-based speaker verification. *IEEE/ACM Trans. Audio, Speech, Language Process. (TASLP)*, *27*, 496–506.

Ramos, D., & Gonzalez-Rodrigues, J. (2008). Cross-entropy analysis of the information in forensic speaker recognition. In *Proc. The Speaker and Language Recognition Workshop (Odyssey)*. ISCA.

Rathgeb, C., & Uhl, A. (2011). A survey on biometric cryptosystems and cancelable biometrics. *EURASIP J. on Information Security (JIS)*, *2011*.

Sadeghi, A.-R., Schneider, T., & Wehrenberg, I. (2009). Efficient privacy-preserving face recognition. In *Proc. Intl. Conf. on Information Security and Cryptology (ICISC)* (pp. 229–244). Springer.

Schneider, T. (2012). *Engineering Secure Two-Party Computation Protocols: Design, Optimization, and Applications of Efficient Secure Function Evaluation*. Springer.

Schneider, T., & Treiber, A. (2019). A comment on privacy-preserving scalar product protocols as proposed in ”SPOC”. *IEEE Trans. Parallel Distrib. Syst.*, . To appear.

Smaragdis, P., & Shashanka, M. (2007). A framework for secure speech recognition. *IEEE Trans. Audio, Speech, Language Process. (TASLP)*, *15*, 1404–1413.

Snyder, D., Garcia-Romero, D., Povey, D., & Khudanpur, S. (2016). Deep neural network-based speaker embeddings for end-to-end speaker verification. In *Proc. IEEE Spoken Language Technology Workshop (SLT)* (pp. 165–170). IEEE.

Thorne, B. (2017). Python Paillier. `https://github.com/n1analytics/python-paillier/`, last accessed: 2018-12-17.

Toli, C.-A. (2018). *Secure and Privacy-Preserving Biometric Systems*. Ph.D. thesis KU Leuven, ESAT, COSIC.

Toli, C.-A., Aly, A., & Preneel, B. (2016). A privacy-preserving model for biometric fusion. In *Proc. Intl. Conf. on Cryptology and Network Security (CANS)* (pp. 743–748). Springer.

Xu, Y., Cui, W., & Peinado, M. (2015). Controlled-channel attacks: Deterministic side channels for untrusted operating systems. In *Proc. IEEE Symposium on Security and Privacy (S&P)* (pp. 640–656). IEEE.

Yao, A. C.-C. (1986). How to generate and exchange secrets. In *Proc. Annual Symposium on Foundations of Computer Science (FOCS)* (pp. 162–167). IEEE.

Ye, S., Luo, Y., Zhao, J., & Cheung, S. (2009). Anonymous biometric access control. *EURASIP J. on Information Security (JIS)*, *2009*, 1–17.

Zahur, S., Rosulek, M., & Evans, D. (2015). Two halves make a whole: Reducing data transfer in garbled circuits using half gates. In *Proc. Annual Intl. Conf. on the Theory and Applications of Cryptographic Techniques (EUROCRYPT)* (pp. 220–250). Springer.