SNUSE: A Secure Computation Approach for Large-Scale User Re-Enrollment in Biometric Authentication Systems

Ivan De Oliveira Nunes, Karim Eldefrawy, Tancrède Lepoint

Abstract—Recent years have witnessed an increasing demand for biometrics based identification, authentication and access control (BIA) systems, which offer convenience, ease of use, and (in some cases) improved security. In contrast to other methods, such as passwords or pins, BIA systems face new unique challenges; chiefly among them is ensuring long-term confidentiality of biometric data stored in backends, as such data has to be secured for the lifetime of an individual. Cryptographic approaches such as Fuzzy Extractors (FE) and Fuzzy Vaults (FV) have been developed to address this challenge. FE/FV do not require storing any biometric data in backends, and instead generate and store helper data that enables BIA when a new biometric reading is supplied. Security of FE/FV ensures that an adversary obtaining such helper data cannot (efficiently) learn the biometric. Relying on such cryptographic approaches raises the following question: *what happens when helper data is lost or destroyed (e.g., due to a failure, or malicious activity), or when new helper data has to be generated (e.g., in response to a breach or to update the system)*? Requiring a large number of users to physically re-enroll is impractical, and the literature falls short of addressing this problem. In this paper we develop SNUSE, a secure computation based approach for non-interactive re-enrollment of a large number of users in BIA systems. We prototype SNUSE to illustrate its feasibility, and evaluate its performance and accuracy on two biometric modalities, fingerprints and iris scans. Our results show that thousands of users can be securely re-enrolled in seconds without affecting the accuracy of the system.

Index Terms—Biometrics, Authentication, Secure Computation, MPC.

1 Introduction

Current Biometrics-based Identification and Authentication (BIA) systems[1] [1], [2] typically store user's reference Biometric Templates (BTs), such as fingerprint minutiae points and/or iris codes, in the backend. If the backend is compromised or breached, sensitive biometric data of a large number of users may be leaked, enabling adversaries to impersonate users and circumvent BIA systems. For example, in 2015, the Office of Personnel Management was compromised and led to the leakage of 5.6 million fingerprints of federal workers that applied for security clearances in the United States of America [3].

In order to protect BTs, typical solutions use secure elements or encryption. Secure elements are applicable when matching is performed against a few biometrics (e.g., the Touch ID technology on iPhones [4]) but do not scale to a large number of biometrics. Encrypting BTs also brings challenges: (1) BTs have to be decrypted to match newly supplied biometric readings during authentication, (2) decryption keys associated with the encrypted templates have to be stored somewhere close by (logically and maybe even physically), and (3) when the backend is compromised or breached, the encrypted templates may be leaked. As reference BTs have to be protected for the lifetime of individuals, it is important to select algorithms and key sizes that remain secure for several decades (say, 40-50 years), which remains a challenging task.

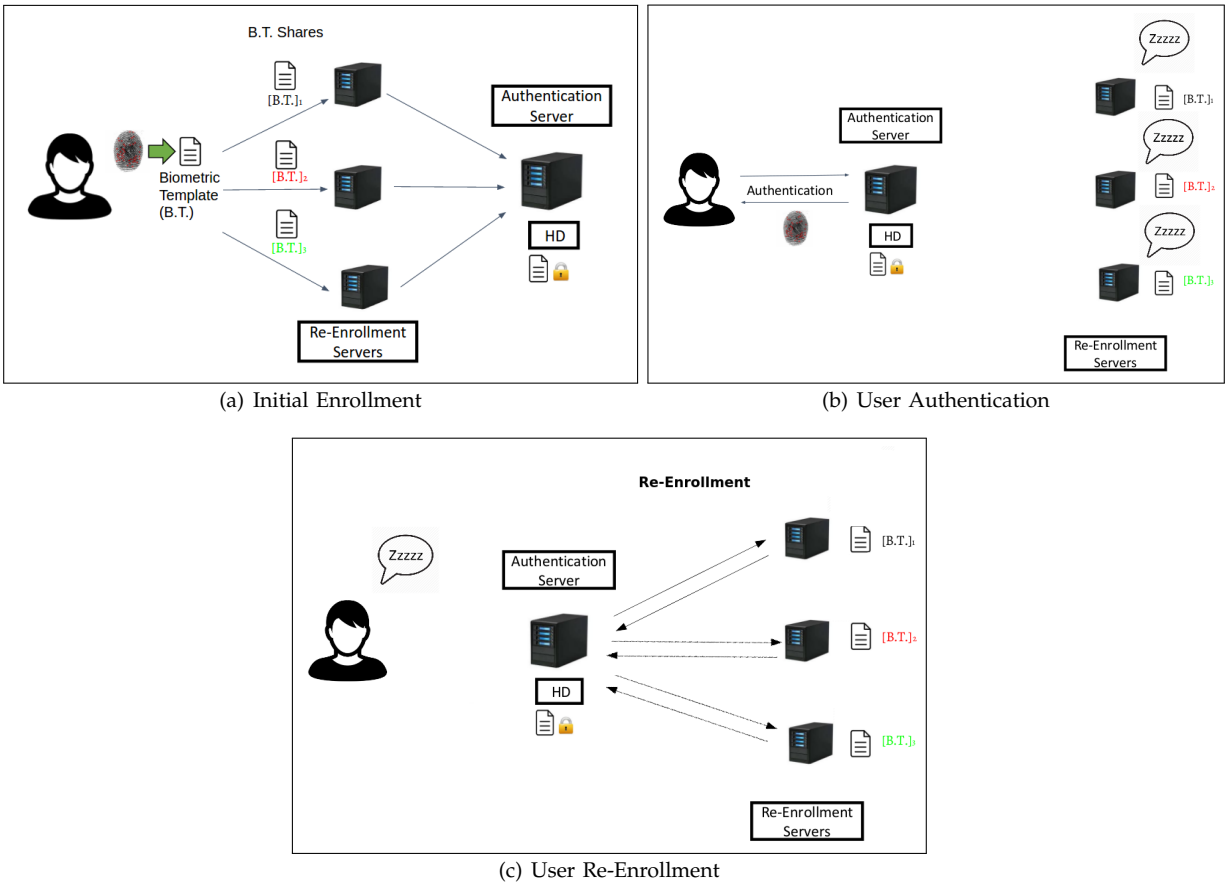
A third cryptographic approach to address the above shortcomings and to construct secure BIA systems has been proposed in the form of Fuzzy Vaults (FV) [5] and Fuzzy Extractors (FE) [6]. FE and FV alleviate the need to store BTs in the system's backend. They enable one to perform matching during the normal operation of a system by using some Helper Data (HD), extracted during the user's initial enrollment into the system. The HD securely encodes[2] a secret, or cryptographic key, that cannot be retrieved unless a biometric sample similar enough to the one used to generate this HD is provided as input. Therefore, the BIA system can determine if a new reading of a biometric corresponds to the user being authenticated. The security of the approach stems from the fact that HDs do not convey information about the underlying biometric. This guarantee can be information-theoretic/statistical or computational depending on the details of the FE/FV scheme.

To deploy this third cryptographic approach at scale, one has to address the challenge of re-enrollment: in

- *Ivan De Oliveira Nunes is with University of California Irvine.*
- *Karim Eldefrawy is with SRI International.*
- *Tancrède Lepoint is with Google Inc.*
- *E-mails: ivanoliv@uci.edu, karim.eldefrawy@sri.com, tancrede.lepoint@gmail.com*

1. We omit explicitly mentioning access control, we assume it implicitly when authenticating an individual and then granting access based on the authenticated identity.

2. We use the term "encode" loosely as helper data may not actually encode the secret, but only enable constructing it when the biometric is also present.



(a) Initial Enrollment

(b) User Authentication

(c) User Re-Enrollment

Fig. 1. Initial user enrollment, user authentication, and user re-enrollment in SNUSE. During regular authentication, the user interacts with the authentication server only, which stores the HD of all the users and enables recovering the user's secret/key when the correct biometric is supplied. When re-enrollment is required, the authentication server communicates with the re-enrollment servers and uses MPC to compute new HD, which encodes a new secret/key, from the secret-shared BT. We emphasize that regular authentication does not require involvement of the re-enrollment servers and, conversely, the re-enrollment phase does not require user involvement.

the lifetime of a cryptographically secure BIA system, it will be often necessary to re-generate HDs, because of breaches or to perform maintenance or updates. For example, a secret key may need to be revoked and replaced by a fresh one when the HD is leaked, damaged, or corrupted. A second example is when access control is enforced via encryption. In this case, changing a given user's permissions might require changing the user's cryptographic keys. In both examples, existing systems require physical presence of the user to refresh cryptographic material, which is laborious, slow, costly, and hence impractical. This problem is currently not addressed in the literature.

1.1 Contributions

In this work we introduce SNUSE (Secure Non-interactive User at Scale re-Enrollment), a new non-interactive approach for secure user re-enrollment that does not require user involvement, nor storing biometric templates (BTs) of users, in the clear, nor in encrypted form, at any single backend server. Instead, SNUSE uses secret sharing [7] to distribute the original BTs among several offline components (the exact number of components is a configurable parameter), and performs the re-enrollment in a secure distributed manner by computing the required Helper Data (HD) generation algorithms using efficient secure multi-party computation (MPC). This approach ensures that at no point during the system's operation are original reference BTs reconstructed in the clear at any backend servers. The components storing shares of BTs can remain offline during normal operation and thus are inaccessible through the network. When re-enrollment is required, the components are brought online and connected to the system for only a brief period (e.g., seconds).

We envision that SNUSE would be used in industrial, enterprise, or government settings, where several (e.g., thousands) of employees/users are required to authenticate daily. In such a setting there might be several entry points. For example, an enterprise might use biometrics to control physical access to different buildings. The users should be able to enroll once and, after that

point, authenticate from any entry point. The enterprise wants to be in charge of, and to be able to change, any cryptographic parameters and access control policies without requiring participation of users. The employees do not want their biometrics stored in clear and subject to breaches or attacks. Conversely, the enterprise wants to avoid liability of storing thousands of users' biometrics insecurely. Ideally, all these functionalities should not impact the underlying biometric matching accuracy. To achieve these goals the enterprise would be responsible for deploying one Authentication Server, for handling normal authentication requests, and a set of Re-Enrollment Servers, used to store secret shares of the users' BTs and to refresh users' cryptographic secrets and access control policies.

Figure 1 illustrates the initial enrollment, subsequent authentication, and re-enrollment phases in SNUSE. To the best of our knowledge, this is the first approach amending Biometrics-based Identification and Authentication (BIA) systems to enable performing non-interactive user re-enrollment without storing the user's biometric template in clear, or encrypted, in any single component or server. In summary, our contributions are threefold:

- **SNUSE Design:** We develop SNUSE, a set of protocols for secure biometrics-based authentication. SNUSE relies on Fuzzy Extractors/Vaults to ensure the privacy of biometric templates while using Secure Multi-Party Computation (MPC) to enable large scale non-interactive re-enrollment.
- **Protocol Optimization:** To enable SNUSE operation in large enterprise or industrial settings, we show how the multi-party computation involved in SNUSE can be optimized via pre-computation of multiplication operations in the enrollment phase. Such optimization allows fast re-enrollment of thousands of users in seconds.
- **Prototype & Evaluation:** We implement a fully functioning prototype of SNUSE, including all of its sub-protocols using the Number Theory Library (NTL) [8]. Our prototype works with fingerprints and iris scans. We evaluate our prototype under different configurations and the results demonstrate that SNUSE can achieve fast re-enrollment of a large number of users without affecting the accuracy of the underlying biometric matching algorithm.

1.2 Organization

Section 2 overviews the building blocks used in SNUSE. Related research efforts are overviewed in Section 3. Section 4 presents SNUSE in details. Section 5 discusses the implementation details of a prototype for SNUSE that works with fingerprints and iris scans using Fuzzy Vaults. Experimental results using the prototype are presented in Section 6. Section 7 discusses SNUSE's security.

2 Background

We here overview SNUSE's building blocks and introduce the notation that will be used in the rest of the paper.

2.1 Biometrics-Based Authentication

Biometrics-based authentication systems [1], [2] have appealing advantages when compared to password-based or token-based systems. In particular, since the biometric is already tied to the user, the user does not have to worry about memorizing or keeping track of a password/secret or carrying a physical token. During enrollment into a BIA system, a reference Biometric Template (BT), composed of features uniquely identifying the user, is sampled and stored at an Authentication Server (AS). Later, when the user attempts authentication, a biometric sample is collected and the same feature extraction process is applied to generate a second BT. This new BT is compared to the one stored at AS and, if their similarity exceeds a given threshold, the user is successfully authenticated. Otherwise, user authentication fails. A feature extraction procedure, applied to a biometric sample of a user U, results in a BT that can be represented as:

$$BT_U = \{p_1, \ldots, p_M\}, \tag{1}$$

where each p_i is a data point representing a unique detail of U's biometric. For example, fingerprint BTs include Cartesian coordinates and orientation of *minutiae*, i.e., of regions in the fingerprint image where fingerprint lines merge and/or split. Such minutiae points are encoded as:

$$p_i = (x_i, y_i, \theta_i), \tag{2}$$

where, x_i is the x coordinate, y_i is y coordinate, and θ_i the orientation of the minutiae i extracted from U's fingerprint. Similar methods can be proposed to encode other biometrics such as iris scans (see Section 5.2 for one example).

Traditional BIA systems store BTs of thousands (or even millions) of users in clear [9]. The reason for this is that each time a biometric is sampled by a sensor, it is slightly different due to noise. Standard mechanisms for secure storage of passwords (e.g., salted-hashing) cannot match two noisy readings of the same biometric because they are not exactly the same. Unfortunately, the advantages of biometrics come with a high risk; if the leaked biometric's modality is stable, leakage of a user's biometric at any point in time affects security of *all* authentication systems using this biometric for years. Consequently, protecting the confidentiality of biometric data is of utmost importance.

As discussed in Section 1, Fuzzy Extractors (FE) and Fuzzy Vaults (FV) are cryptographic schemes that use an input BT to (i) generate Helper Data (HD), which encodes a secret k; and (ii) ensure that the HD does not reveal anything about the BT or k, unless prompted with

(a noisy version of) the same biometric used to generate the HD. Note that k can be an arbitrary secret and not necessarily a symmetric key. In current approaches, the secret k stored in the HD cannot be refreshed without requiring the physical (or remote) presence of user for the re-enrollment process in which a new biometric sample is collected and new HD is computed for the new secret. This manual approach does not scale, as re-enrolling thousands (or more) users is often impractical, or at best very laborious. In this paper we propose an MPC-based solution that enables large-scale automatic re-enrollments.

2.2 Secret Sharing

In K-out-of-N secret sharing [7] a dealer distributes shares of a secret among N parties such that subsets of K or more parties can recover the secret. However, knowing up to $K-1$ shares leaks no information about it. In SNUSE, we will generate N shares of a biometric template and store them on N re-enrollment servers. Given a secret X, let $[X]_j$ denote the j-th share and denote the generation of N shares of X by:

$$\{[X]_1, \ldots, [X]_N\} \leftarrow X\,. \tag{3}$$

Denote the reconstruction of secret X from K shares by:

$$X \leftarrow \{[X]_1, \ldots, [X]_K\}\,. \tag{4}$$

In Shamir's (K, N) secret sharing scheme [7] over a finite field $\mathbb{F}$, one randomly generates the coefficients of a polynomial P of degree $d = K-1$. The independent term a_0 is then set to the secret X, and one can generate N secret shares $S = \{(i, P(i))\}_{i=1}^{N}$. Since P has degree $K-1$, K points in S are enough to interpolate P and reconstruct its coefficients, including the secret $a_0 = X$. A set of $L < K$ shares does not reveal any information about X because there are multiple polynomials of degree K that can be constructed from these points.

Summary of Assumptions and Guarantees: *Shamir's secret sharing is information theoretically secure. In a K-out-of-N scheme, it is guaranteed that less than K shares of the secret leak no information about it. Conversely, K or more shares can be used to completely recover the secret.*

2.3 Secure Multiparty Computation

Secure Multi-Party Computation (MPC) protocols enable mutually distrusting parties to jointly compute a function f of their private inputs while revealing no information (other than the output of f) about their inputs to the other parties [10].

In standard algebraic MPC protocols, each party usually generates shares of its input (using, for instance, Shamir's secret sharing scheme) and distribute one share to each party. A key observation is that if one is able to compute both addition and multiplication on the shares, such that the resulting shares can be combined into the correct result of the operations, one can implement any function f from these two basic operations. Different schemes were proposed to compute addition and multiplication over private inputs [11]–[15]. Nevertheless, most of them share the following common characteristics in the computation of these operations:

- **Addition** of secret shares can be computed locally. To that purpose each party computes addition of its own secret shares. The N local results, once combined, yield the result of an addition of the actual secret(s).
- **Multiplication** of secret shares requires communication. Even though different schemes exist, most require parties to broadcast an intermediate (blinded) result during the computation of multiplication, such that individual shares of the multiplication result can be correctly computed.

We do not specify details of the multiplication sub-protocols and we refer the reader to [12], [13] for further details. The takeaway is that multiplication requires communication between parties, and in practice the overhead to multiply is usually orders of magnitude higher than the overhead of addition. In our design we take advantage of the unique characteristics of a biometric enrollment system to reduce the number of multiplications to a minimum (sometimes even eliminating it), allowing cost-effective scalable MPC-based user re-enrollment.

Summary of Assumptions and Guarantees: *MPC assumptions and guarantees vary depending on the specific MPC scheme used. In this work we focus on the honest-but-curious (HBC) threat model with honest majority, in which corrupted parties might collaborate to learn private inputs of other parties, but they do not deviate from the protocol specification. The scheme remains secure if the number of colluding parties is smaller than half of the total number of parties.*

2.4 Fuzzy Vault Scheme

Fuzzy Vaults (FV) [5] are designed to work with BTs represented as an unordered set of points as shown in Eq. (1). Given a user's biometric template (BT_U), they hide a secret k generating a correspondent helper data HD. The secret can be, for instance, private data or a cryptographic key. The scheme consists of two algorithms: generation (FV_{GEN}) and secret reconstruction (FV_{OPEN}), which can be informally defined as follows:

- FV_{GEN}: receives as input k and BT_U. Uses BT_U and k to generate a Helper Data HD, which encodes the secret k without revealing information about neither k nor BT_U:

$$HD = FV_{GEN}(BT_U, k) \tag{5}$$

- FV_{OPEN}: receives as input HD and BT'_U. Retrieves k from HD if, and only if, BT'_U is a template extracted from the same biometric as BT_U, i.e., the

template used as FV_{GEN}'s input during HD's generation. In other words, given a distance function D defined over some metric space, and a threshold w:

$$FV_{OPEN}(BT'_U, HD) = \begin{cases} k & \text{if } D(BT_U, BT'_U) \leq w \\ \perp & \text{otherwise} \end{cases}. \quad (6)$$

The threshold w is a security parameter that allows to control the trade-off between minimizing false acceptance (revealing k to the wrong user) and false rejection (refusing to reveal k to the rightful user).

In the FV scheme of [5], FV_{GEN} algorithm starts by selecting a polynomial P of degree d defined over a field $GF(2^\tau)$ and splitting k into the $d+1$ coefficients $a_0, \ldots, a_d$ of P. The resulting polynomial is defined as:

$$P_k(x) = \sum_{i=0}^{d} a_i x^i \quad (7)$$

where the coefficients $\{a_0, \ldots, a_d\}$ are generated from k and can be used by anyone to reconstruct k. Since P is defined over $GF(2^\tau)$, each coefficient represents τ bits; this implies that the size is limited to $(d+1)\times\tau$ bits. After embedding k as the coefficients of $P_k(x)$, each of the M data points in BT_U is evaluated on the polynomial $P_k(x)$ generating a set of points in a two-dimensional space:

$$L_P = \{(p_1, P_k(p_1)), \ldots, (p_M, P_k(p_M))\}. \quad (8)$$

Note that the field must be large enough to encode a data point from BT_U as a single field element. The resulting set L_P contains only points in the plane that belong to the polynomial $P_k(x)$. In addition to L_P, a set of chaff points L_S of size $S \gg M$ is generated by randomly selecting pairs (r_x, r_y), where r_x and $r_y \in GF(2^\tau)$. It is worth noting that the r_x points should be sampled from a distribution indistinguishable from that of real data points $(p_1, \ldots, p_M)$ of the given biometric modality (e.g., fingerprints, iris scans, etc). This step results in:

$$L_S = \{(r_{x1}, r_{y1}), \ldots, (r_{xS}, r_{yS})\} \quad (9)$$

Finally, L_P and L_S are shuffled together using a random permutation π and the result is included in the HD. The HD also includes the set of public parameters $\Phi = \{\mathbb{F}, d, M, H(k)\}$, where $\mathbb{F}$ is the field in which $P_k(x)$ is defined and d is $P_k(x)$'s degree, M is the size of BT_U, i.e., the number of points in the HD that belong to $P_k(x)$, and $H(k)$ is a cryptographic hash of the secret k allowing one to verify if the correct secret was reconstructed using FV_{OPEN}.

$$HD = \{\pi(L_P \cup L_S), \Phi\} \quad (10)$$

The key idea behind security of the FV scheme is that with $d+1$ distinct points $(p_i, P_k(p_i))$, one can interpolate $P_k(x)$, retrieve its coefficients and thus recover k. However, finding which $d+1$ points to interpolate out of the $M+S$ in HD is hard if $M+S$ is sufficiently larger than d.

When attempting to reconstruct k from the HD using a new biometric reading BT'_U, the FV_{OPEN} algorithm will use a distance function (which must be defined according to the biometric type) to select, out of the $M+S$ points in the HD, the M points that are the closest matches to the points in BT'_U. If, out of the M selected points, at least $d+1$ points are points that belong to the original L_P, then the algorithm will be able to interpolate the correct polynomial and recover k. To verify that k was correctly recovered, the algorithm hashes the result and compares it to $H(k)$, which was published together with the HD. If less than $d+1$ correct points are among the M points selected via distance matching, no interpolation with combinations of $d+1$ points out of M will yield a match in the hash, because $P_k(x)$ will not be interpolated correctly. Therefore, FV_{OPEN} will reject BT'_U.

Note that the FV scheme does not rely on the order of the elements in BT_U and BT'_U and does not require all points to be present in both templates. Instead, $d+1$ data points in BT'_U must be close enough to points in BT_U. In that sense, the polynomial degree d acts as a security parameter that allows calibration of the scheme to reduce false acceptance by increasing the required number of matching data points.

Summary of Assumptions and Guarantees: *The security of FV relies on the infeasibility of the polynomial reconstruction problem [16], and the inability to distinguish the statistical distribution of minutiae from that of chaff points. The degree d of the polynomial used to encode k determines the minimal number coincidental minutiae (in BT and BT') that are necessary to reveal k.*

3 Related Work

New attack vectors emerge continuously with the increasing ubiquity and availability of computing devices and resources, such as Internet of Things (IoT) gadgets and cloud computing technologies. Securing this new computing ecosystem poses unprecedented challenges [17]. To leverage these new services securely, one must address the problems of user [18], [19] and device [20]–[22] authentication, which are still challenging [23], [24]. Biometrics represent an appealing approach for user authentication and have been widely investigated over the past decade [25]–[28]. This includes several proposals for interesting and unconventional biometric modalities [29], [30]. Nonetheless, due to their unique challenges [31], biometrics still represent an active research topic.

A study of security and privacy challenges facing biometrics [32], especially iris scans, investigated suitability and viability of relying on them as the sole method for identification and authentication. The results of the study in terms of accuracy and entropy were both positive and encouraging. The first Fuzzy Vault (FV) scheme was developed in [33]. It was later implemented and tested with actual fingerprint biometrics in [34]. Fuzzy Extractors (FE) were formalized in [35], and further applied to

biometrics in [36]. Most FE schemes provided statistical or information-theoretic security, until the scheme of [37] was developed; this computational FE scheme relies on hardness of the Learning with Errors (LWE) problem. Re-usability of FE was first studied in [38], where a reusable FE scheme was first formalized, and then developed for specific attacks. Re-usability enables one to extract multiple helper data from the same biometric without leaking any additional information. Not every FE/FV can be reused and still ensure security as illustrated in subsequent research [39], [40] which analyzed re-usability of multiple FE schemes and demonstrated attacks against them. Finally, indistinguishability based definitions for re-usability were presented [41] and theoretical analysis demonstrated that the computational FE scheme in [37] is not even weakly reusable. In fact, from the helper data, HD1 and HD2, of two instances of the scheme, an attacker can learn the original biometric inputs w_1 and w_2 in their entirety. Fixes to the FE scheme in [37] were then developed by using common public parameters and proven secure in the weak-reusability model, and further transformed to ensure string re-usability in the random oracle model. We are not aware of any work studying the problem of non-interactive re-enrollment in biometric authentication systems based on FEs/FVs, except for the preliminary version of the present work in [42].

Secure two/multiparty computation (2PC/MPC) has been an active area of research for the past three decades [43]–[52]. Recent models and practical schemes [53] provide a trade off between security and privacy guarantees on one hand, and required computation and communication on the other. For example, the covert model [54] accounts for settings where the involved parties are less likely to cheat if they get caught with a high probability (e.g., 0.5) and the work in [55] proposes protocols in which a malicious adversary may learn a single (arbitrary) bit of additional information about the honest party's input. Generic 2/MPC protocols can be utilized as is in cryptographically secured BIA systems but may incur higher overhead. If performance of generic protocols is unsatisfying, one can design function specific secure protocols for the generate function of FE/FV; several function-specific two and multiparty protocols for pattern matching were also developed in [56], [57].

4 SNUSE APPROACH

Figure 1 illustrates the SNUSE's operation during a) initial user enrollment, b) regular user authentication, and c) non-interactive user re-enrollment. SNUSE involves three types of parties: a User (U), an Authentication Server (AS), and n Re-Enrollment Servers ($\{RES_1, \ldots, RES_n\}$).

During the initial enrollment of U into the BIA system, U's biometric template BT_U is secret shared into n shares $([BT_U]_1, \ldots, [BT_U]_n)$. Each share is distributed to one of the n Re-Enrollment Servers ($\{RES_1, \ldots, RES_n\}$). The RESs then jointly generate U's secret k_U and use an MPC protocol to compute FV's helper data, HD_{k_U}, from their shares $([BT_U]_1, \ldots, [BT_U]_n)$, thus securely using BT_U to "lock" k_U .

Regular user authentication happens between U and AS. Since AS only stores HD_{k_U} (which reveals nothing about BT_U), the FV_{OPEN} algorithm must be used to retrieve k_U. If HD_{k_U} was correctly computed using a secure FV_{GEN} algorithm, the only way to retrieve k_U from HD_{k_U} is by providing, as input to FV_{OPEN}, a second biometric template BT'_U which is close enough to the original BT_U. Therefore, $FV_{OPEN}(BT'_U, HD_{k_U})$ will successfully retrieve k_U if, and only if, $BT'_U \approx BT_U$, i.e., BT'_U is a noisy version of the same biometric used to generate HD_{k_U}. After this stage, in the case where k_U is a cryptographic key, for instance, it can be used by U to decrypt files or messages, or to get access to resources based on the recovered secret.

Whenever k_U needs to be replaced with a fresh secret k'_U (this process may happen periodically within an organization to guarantee freshness of users' cryptographic keys), RESs are brought online and connected to the system, and AS issues a request to the RESs to compute a new $HD_{k'_U}$ for U. The RESs will securely generate a new k'_U and (as in the enrollment protocol) use U's secret shares $[BT_U]_1, \ldots, [BT_U]_n$, stored during U's initial enrollment, to compute $HD_{k'_U}$. This way, users' cryptographic materials can be refreshed and brand new HDs can be constructed without requiring users' presence to re-sample their biometrics and without storing their BTs in clear.

At a first sight, a simple approach for generating a fresh k'_U would be to multiply the y-coordinates in the FV by randomness σ, which would result in a fresh random key $k' = k \cdot \sigma$ encoded in the FV. However, this approach does not work for several reasons. First, as discussed before, k might not be an independent random byte-stream, such as a symmetric key; it could, for instance, include a set of user permissions or an asymmetric private-key (sk') associated with the user's public-key. In the latter case, the generation of fresh $k' = sk'$ implies deriving the corresponding new public-key (pk'). SNUSE can handle these cases while simple multiplication by randomness can not. Second, while this approach using randomization might work for a classic FV with symmetric keys, because the key is encoded as coefficients of a polynomial, it does not necessarily apply to other FV/FE constructions; SNUSE on the other hand is a generic approach (as any computable function can be computed using MPC), that could be implemented with other FV/FEs. Third, even in the case where the scheme uses a classic FV to encode a random symmetric key, multiplying by randomness σ will update the encoded secret but will prevent the reconstruction of the secret, i.e., $k'_U = FV_{OPEN}(BT'_U, HD)$ cannot be computed; this is because FV_{OPEN} must verify the hash of each candidate recovered key with the stored $H(k)$ to decide if the correct key was reconstructed. However, $H(k')$ cannot be updated in the same way, because for any

reasonable hash function, $H(\sigma \cdot k) \neq \sigma \times H(k)$.

Throughout the rest of this section we describe the steps of SNUSE in more details. We have implemented SNUSE with fingerprints and iris scans, and evaluated SNUSE performance in terms of computation and storage requirements. Our evaluation shows that SNUSE can re-enroll thousands of users in seconds; the storage requirements for thousands of users is a few MBytes.

Remark. *All message exchanged in the following protocols are assumed to be through secure authenticated channels, such as standard TLS. The establishment of such secure channels is omitted from the protocols for the sake of clarity.*

4.1 Initial User Enrollment

The initial user enrollment, presented in Fig. 2, is the only protocol in SNUSE that involves all parties, i.e., U, AS, and RES_i $\forall i \in [1, n]$. This protocol is executed only once for each user, all interactions after the initial enrollment are performed either between U and AS (authentication), or between AS and RESs (re-enrollment).

The protocol starts with U using a trusted enrollment device (e.g., fingerprint sensor, iris reader), referred to as B.T. Reader, to measure and extract U's biometric template BT_U (Fig. 2, line 1). BT_U is then split into n secret shares, where n is the number of RESs. Each share $[BT_U]_i$ transmitted to, and stored by, the respective RES_i (Fig. 2, lines 2-4). Note that in Fig. 2 we only depict one RES_i, however, in reality each of the n RESs receives and stores its share $[BT_U]_i$.

Once each share $[BT_U]_i$ is stored on the corresponding RES_i, the RESs agree on the new authentication material k for the user, by using $RESSecretGen$ (Fig. 2, line 5). The details of RESSecretGen are application specific and discussed below, in Section 4.2.

Note that, during user enrollment protocol, BT_U is only visible in clear to the trusted sensor device that reads and then secret shares the biometric. Each RES_i only sees its own share which leaks no information about BT_U itself. AS only sees HD, which can not be used to reconstruct BT_U by the security of FV. Therefore, confidentiality of the biometric is ensured during user enrollment. In fact, there is no single server from which BT_U can be retrieved in clear. BT_U only exists in clear ephemerally at the B.T. Reader and that must happen anyway because B.T. Reader is the sensor device used to sample the user's biometric.

4.2 User Authentication

A consequence of correct computation of HD using MPC in the enrollment phase is that the user authentication protocol consists of simply using standard local computation of FV_{OPEN} with a new biometric reading BT'_U and the stored HD. The RESs do not participate in user authentication, but only in the enrollment and re-enrollment protocols.

The authentication protocol, shown in Fig. 3, starts with a user supplying its ID (U_{ID}) and biometric sample to the B.T. Reader. A biometric template BT'_U is generated from the new sample and kept locally. U_{ID} is sent to AS which fetches U's HD from its database based on the supplied U_{ID}, and sends the associated HD as a reply. BT'_U is then used to extract k from HD using the FV_{OPEN} algorithm. Note that here, and similar to user enrollment, U's BT only exists in clear in the B.T. Reader.

The exact specification of RESSecretGen() as well as the usage of the secret k after user authentication are application specific, i.e., k can be an arbitrary secret used for different purposes. We follow an approach similar to the one (e.g., [58]) utilized in Password-Authenticated Key Agreement (PAKE) and do not restrict the usage of k, leaving it up to the use-case. Nonetheless, for clarity, we illustrate two potential uses:

- k may be used to decrypt files from a file-system, i.e., used as an encryption-based access control mechanism. In this case, RESSecretGen() will consist of the such file-system choosing k and sending it (in clear or, optionally as secret shares to the RESs). After successful user authentication, k can be used to decrypt such files.
- k may be used to authenticate to a remote server. After retrieving the secret k, the user can authenticate to the server using a standard challenge-response mechanisms based on the secret k. In that case, k might be part of an asymmetric encryption scheme (sk, pk), where $k = sk$ and pk is a public key, known to the remote server (i.e., the challenging party) and associated with the secret key sk. The remote server can then authenticate the user by sending a challenge $Enc_{pk}(nonce)$, where $nonce \leftarrow \$\{0,1\}^l$. If the user is able to retrieve k from the FV, it can then use $k = sk$ to compute $nonce \leftarrow Dec_{sk}(Enc_{pk}(nonce))$ and send the result back to the remote server, proving its claimed identity.

4.3 Non-Interactive User Re-Enrollment

Non-interactive user re-enrollment works by having the RESs compute a new HD based on a fresh secret k'. Since the shares of the biometric template are stored during the initial enrollment, this step does not require user involvement, even though the biometric template does not exist in clear neither at RESs nor AS.

In this protocol, AS sends a request for re-enrollment for each RES_i, containing the user ID(s) for which re-enrollment(s) should occur. The RESs will then jointly generate a new secret k' for each user (a different k' for each user) and encode it as a polynomial of degree d. Each RES_i then uses U_{ID} to fetch the secret share $[BT_U]_i$ associated with U_{ID} and uses FV^{MPC}_{GEN} to compute $P_{k'}(x)$ on the secret share $[BT_U]_i$. The result is a secret share $[HD]_i$ for a brand new HD which encodes the freshly generated k' under the same biometric template BT_U. This process is depicted on Fig. 4. Finally, AS receives all N shares $[HD]_i$ and compute the new HD for each user U, which can, from this point on, be used for authenticating user with the protocol in Fig. 3. Notice that, during

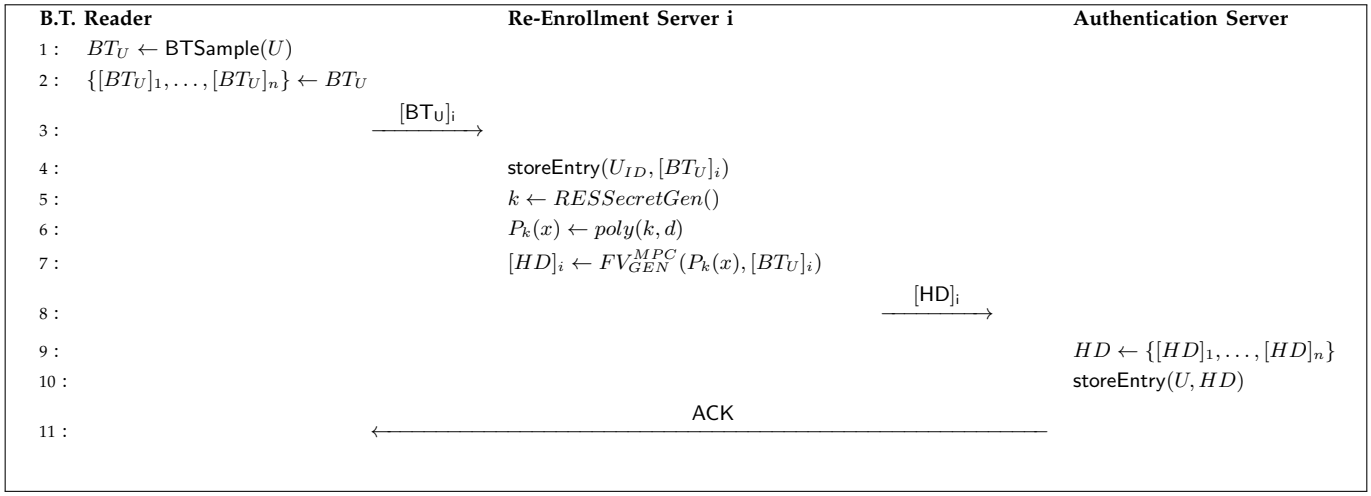


Fig. 2. User enrollment protocol in SNUSE. See detailed description of steps in Section 4.1.

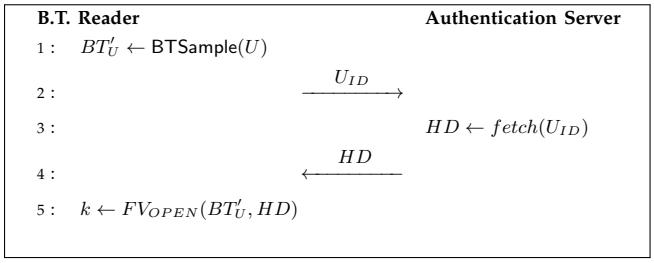


Fig. 3. User authentication protocol in SNUSE. See detailed description of steps in Section 4.2.

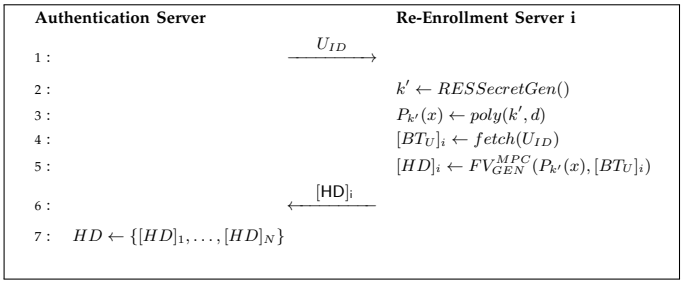


Fig. 4. Re-Enrollment protocol in SNUSE. See detailed description of steps in Section 4.3.

the execution of the re-enrollment protocol, BT_U is not reconstructed in clear at any point. This is only possible because of the computation of HD using MPC over the secret shares. Otherwise, this process would require either i) user involvement to collect another biometric reading or ii) storing the biometric template in clear in the backend servers.

4.4 Using MPC to generate the HD

The fundamental part of SNUSE that allows non-interactive user re-enrollment (without storing BT_U in clear) is the ability to compute the HD from the secret shares $\{[BT_U]_1, \ldots, [BT_U]_N\}$ of BT_U. In the protocols of Fig. 2 and Fig. 4, this is represented by the computation of the function $FV_{GEN}^{MPC}(P_k(x), [BT_U]_i)$, resulting in a secret share $[HD]_i$ that can be interpolated to reconstruct the actual HD.

In this section, we discuss how FV_{GEN} is computed from the secret shares. We start by outlining the basic operations needed to compute FV_{GEN}, and then describe details of how each is performed using secret shared data. The standard local computation of FV_{GEN} algorithm, involves three basic types of operations:

1) Evaluation of the polynomial $P_k(x)$, that encodes k, on each of the M data points ($\{p_1, \ldots, p_M\}$) that compose BT_U to generate the list of points in the polynomial P_k:

$$P = \{(p_1, P_k(p_1)), \ldots, (p_i, P_k(p_i)), \ldots, (p_M, P_k(p_M))\} \quad (11)$$

2) Generation of a list S composed of s random chaff points (r_x, r_y), to be shuffled together with the polynomial points:

$$S = \{(r_{x,1}, r_{y,1}), \ldots, (r_{x,i}, r_{y,i}), \ldots, (r_{x,s}, r_{y,s})\} \quad (12)$$

3) Random permutation π to shuffle the elements of lists S and P together generating the HD:

$$HD = \pi(P \cup S) \quad (13)$$

Steps 2 and 3 are relatively easy to compute when compared step 1. For the random chaff point generation (Step 2), each RES_j computes a random share $[(r_{x,i}, r_{y,i})]_j$. When the randomly generated shares are merged together they will result in random chaff points.

For Step 3, all M RESs agree on a single random permutation π, and all of them permute their secret shares according to this same randomly chosen permutation. Note that the permutation π is kept secret from AS, because knowing π would allow an adversary who takes control of AS to separate chaff points from the points in the polynomial by computing π, allowing reconstruction

of BT_U. Nevertheless, even though AS does not know which permutation was used, because each RES use the same permutation to compute $\pi(P \cup S)$ on their shares, each share will be matched to its correct set of shares during the reconstruction of the HD at AS.

Since we are able to generate random chaff points and compute a permutation on the secret shares (which results in a permutation on the reconstructed HD), the remaining task for FV_{GEN} is to compute the polynomial P_k using MPC on each of the secret shares. We discuss the classic approach to compute P_k using MPC and then we introduce our optimized version that takes advantage of pre-computation of secret exponents before the secret sharing phase.

In the classical approach, assuming that BT_U is composed of M data points, each secret share $[BT_U]_j$ corresponding to RES_j would be a vector in the form:

$$[BT_U]_j = \{[p_1]_j, \ldots, [p_M]_j\} \tag{14}$$

The polynomial P_k can be generically defined as:

$$P_k(x) = \sum_{i=0}^{d} a_i x^i \tag{15}$$

where:

$$\{a_0, \ldots, a_d\} \leftarrow k \tag{16}$$

denoting that the coefficients of the polynomial encode k. Therefore, computing $P_k(x)$ implies computing exponentiation up to degree d on the secret shared variables, multiplication of the resulting values by the respective constants $a_0, \ldots, a_d$, and addition on the resulting terms $a_i x^i$ for all $i \in [1, \ldots, d]$.

As discussed in Section 2, the bulk of the overhead on MPC comes from multiplication of secret shares, because addition (and consequently multiplication by a constant) can be computed locally by adding the secret shares. Multiplication, on the other hand, requires communication, since the parties must publish intermediate results, i.e., broadcast them to all other parties involved in the MPC protocol. This is usually the major source of overhead in the MPC evaluation, because each multiplication involves several rounds of communication between all parties, and network delays.

The computation of x^d, with no optimization, requires d multiplication operations, i.e., computing $\prod^d x$. In such approach, computing all terms in Eq. (15) would take $\sum_{i=0}^{d} i$. Therefore, the number of communication rounds to compute the HD shares would be:

$$T = M * \sum_{i=0}^{d} i \tag{17}$$

In terms of asymptotic complexity this *naive* approach yields $\Theta(d \times log(d))$ multiplications, where d is the polynomial degree. Such number of multiplications can be trivially reduced to d if we take into account that $x^n = x \cdot x^{n-1}$. This implies that the result of a lower order polynomial term can be used as an intermediate result for the computation of the subsequent term, reducing the number of communication rounds to compute the HD to:

$$T = M * d \tag{18}$$

resulting in linear asymptotic complexity of $\Theta(d)$ for the number of required multiplications.

To make the process more efficient, we take a different approach. We bring the number of necessary multiplications to zero by pre-computing the powers of x before distributing the shares to the RESs. From the standard BT_U, which is a vector of M data points in the format $\{p_1, \ldots, p_M\}$, we pre-compute the x^i exponents for all $i \in [1, d]$ and secret share each of the pre-computed exponents for each data point. Therefore, a secret share $[BT_U]_j$ of a biometric template with pre-computed exponents becomes a $d \times M$ matrix in the format:

$$[BT_U]_j = \begin{pmatrix} [(p_1)^1]_j & \cdots & [(p_M)^1]_j \\ \vdots & & \vdots \\ [(p_1)^i]_j & & [(p_M)^i]_j \\ \vdots & & \vdots \\ [(p_1)^d]_j & \cdots & [(p_M)^d]_j \end{pmatrix} \tag{19}$$

Each column in $[BT_U]_j$ represents a data point p_i and each line an exponentiation of such data point. For example, line 3, column 4, would contain a secret share of the fourth data point in BT_U raised to the cubic power: $[(p_4)^3]_j$.

In this approach, a secret share is included for each of the exponents of each data point. Thus, the evaluation of the polynomial requires no multiplications of secret shares because all exponents are now individual secret shares in the matrix $[BT_U]_j$. Therefore, the computation of $[HD]_j$ to be done locally. Specifically, let $[BT_U]_j(x, y)$ denote the element in line x and column y of the matrix[3]. Then $[y]_j = P_k([p_i]_j)$ can be computed locally for every i as:

$$P_k([p_i]_j) = a_0 + \sum_{k=1}^{d} a_k \times [BT_U]_j(k, i) \tag{20}$$

where $\{a_0, \ldots, a_d\}$ are the polynomial coefficients defined in Eq. (15).

This eliminates the need for network communication making the scheme much faster. This optimization is feasible because in practice $d \approx 10$ and because, during enrollment, one single entity (B.T. Reader) possesses all data points. We take advantage of that to improve efficiency of SNUSE by pre-computing the exponents and including them in the secret shares of the biometric templates. As detailed in Section 6, by pre-computing the exponentiations, SNUSE achieves high performance in terms of processing time with reasonable storage

3. In our notation the first row/column of a matrix is indexed by 1 and not 0.

requirements that are comfortably within the capacity of modern computers.

Algorithm 1: FV^{MPC}_{GEN} computation on RES_j

inputs: Secret share matrix $[BT_U]_j$ (Eq. (19)); fresh secret k; random permutation π; number of chaff points s; and polynomial degree d.
output: $[HD]_j$

1 $\{a_0, \ldots, a_d\} \leftarrow$ EncodeAsPolynomialCoeffs(k, d);
2 $L_p \leftarrow$ emptyList()
3 **forall the** $i \in [1, 2, \ldots, M]$ **do**
4 $\quad pi_x \leftarrow [BT_U]_j(1, i)$
5 $\quad pi_y \leftarrow a_0 + \sum_{k=1}^{d} a_k \times [BT_U]_j(k, i)$ /* MPC */
6 $\quad L_p.append([pi_x, pi_y])$
7 **end**
8 $L_s \leftarrow$ emptyList()
9 **forall the** $i \in [1, 2, \ldots, s]$ **do**
10 $\quad r_x \leftarrow rand_{GF(2^\tau)}()$
11 $\quad r_y \leftarrow rand_{GF(2^\tau)}()$
12 $\quad L_s.append([r_x, r_y])$
13 **end**
14 $L \leftarrow concat(L_p, L_s)$
15 $[HD]_j \leftarrow permute(L, \pi)$
16 **return** $[HD]_j$;

Algorithm 1 synthesizes what is discussed in this section with an algorithmic description of the method to compute a share of a HD using MPC on the matrix $[BT_U]_j$ of Eq. (19). AS receives all shares $[HD]_j$ $\forall j \in [1, n]$ and uses them to reconstruct $HD \leftarrow \{[HD]_1, \ldots, [HD]_n\}$. Note that the MPC evaluation in line 5 of Algorithm 1 only involves additions and multiplication by constants. Therefore, Algorithm 1 can be computed locally at RES_j not requiring communication.

4.5 Secret (k) Confidentiality Discussion

SNUSE is designed to provide non-interactive re-enrollment, allowing one to refresh the stored secret k without interaction and without compromising the confidentiality of the BT. One may argue that the attack surface for an attacker interested in stealing the user's secret k, instead of the BT, will increase because now all RESs need to ephemerally store k at some point in time to enable computation of the new HD. This restriction can be addressed by generating k in a single separate server and using MPC with k as a secret share as well. We consider this optional in SNUSE design, as our focus is to protect the BT itself. Having such feature would add one communication round to the re-enrollment process, because the polynomial coefficients that encode k need to be multiplied by BT shares.

5 SNUSE'S IMPLEMENTATION

To demonstrate the practicality of SNUSE, we implemented a fully functioning prototype working with both fingerprints and iris scans. This section discusses the prototype's implementation details, including its software stack and choices of security parameters.

We implemented SNUSE in C++, using the Number Theory Library (NTL) [8]. The polynomials used in these schemes were defined over the Galois Field $GF(2^{24})$, which is large enough to securely encode cryptographic keys while keeping the scheme computationally efficient. The FV's polynomial degree is a security parameter that allows calibrating authentication accuracy; our results show that the ideal polynomial degree that yields the best accuracy results depends on the type of biometric used. The number of data points and chaff points and the distance functions used in FV_{OPEN} are also specific to the type of biometric used in the scheme. In Section 5.3 we present accuracy results as a function of the polynomial degree of the FV for both fingerprints and iris scans.

A SHA-256 hash function is used to compute the hash of k, which is included in the HD to allow verification of the reconstructed key in FV_{OPEN}. Secret sharing and MPC were also implemented using polynomials defined over $GF(2^{24})$. Therefore, a BT secret share is defined as a collection sets of shares for each data point in such BT. The number of RESs is configurable and determines the number of BT shares generated during the enrollment phase. BT shares are delivered to RESs through TCP sockets. During re-enrollment, AS sends a request to the RESs. Each RES computes its respective share of HD and sends it back to AS. AS uses all received shares to reconstruct and store HD. Details about BT extraction algorithms are biometric-specific and discussed in Sections 5.1 and 5.2. Section 5.3 presents authentication accuracy results for each biometric mode. We emphasize that SNUSE does not affect the accuracy of the underlying BT matching (and therefore extraction) procedure and improving the accuracy is an orthogonal issue. Nonetheless, we report on accuracy to justify our choices for the FV polynomial degrees considered in SNUSE's workload evaluation in Section 6.

5.1 Template Extraction with Fingerprints

The first step in SNUSE is to extract the BT from the sampled biometric. In the case of fingerprints, the biometric reading is an image of the fingerprint and, as discussed in Section 2, each data point p_i in BT is the position and orientation (x_i, y_i, θ) of a fingerprint *minutiae*. In our fingerprint BT extraction we use NIST's Biometric Image Software (NBIS) [59]. From a biometric sample, NBIS returns a set of identified *minutiae* sorted by the confidence in the quality of such *minutiae*. From NBIS output we select the 20 points with the highest confidence and encode them as elements in $GF(2^{24})$. Then we can use them as described in SNUSE's design (Section 4). In our prototype, an HD is composed of 20 real data points mixed with 200 chaff points.

During user authentication (using the FV_{OPEN}) candidate *minutiae* points are selected from the HD based on

(a) Fingerprint biometric template (b) Iris biometric template

Fig. 5. **a)** Fingerprint pre-processing and feature extraction. On the top, two impressions of the same fingerprint. At the bottom the correspondent feature extraction after pre-alignment. **b)** Iris feature extraction. On the top, raw iris scans of two different users and at the bottom the resulting iris templates as bitmaps.

their distance to the minutiae points detected in the new template BT', which is sampled from the user during authentication. We use a distance function similar to the one introduced in [60], defined as:

$$D(p_i, p_j) = \sqrt{(x_i - x_j)^2 + (y_i - y_j)^2} + \beta \times \Delta(\theta_i, \theta_j) \quad (21)$$

where $\Delta(\theta_i, \theta_j) = min(|\theta_i - \theta_j|, 360 - |\theta_i - \theta_j|)$. The parameter β allows controlling how much importance is given to the *minutiae* orientation in the overall distance relative to the euclidean distance between the points. A data point p_i is selected from the helper data if $D(p_i, p_j) < w$ for some point in BT'. As described in [60], the parameters β and w must empirically calibrated to yield the best accuracy. The fingerprint accuracy results, reported in Section 5.3, were conducted with $\beta = 0.2$ and $w = 20$.

To improve accuracy for noisy fingerprint readings, before extracting the template, during the biometric sampling, we compute the fingerprint pre-alignment algorithm proposed in [61]. This algorithm works by translating the fingerprint core-point to the center of a coordinate system and then rotating the image according fingerprint orientation patterns. This way, two misaligned fingerprints can still be accurately matched in FV_{OPEN}, thus increasing the authentication's accuracy. Fig. 5(a) illustrates the result of the template extraction for two misaligned impressions of the same fingerprint before and after pre-alignment. White squares show the minutiae detected in these fingerprints.

5.2 Template Extraction with Iris Scans

In the implementation of the iris version of our prototype, we used the Osiris open-source library [62] for iris templates extraction. Osiris receives as input an iris scan image and outputs an iris template. The resulting templates for two iris scans are depicted in Fig. 5(b). OSIRIS starts by pre-processing the eye image, using contour detection and masks to remove the eye contours, eye lashes, and the pupil from the scan, leaving only the iris. Next, the ring-shaped image, containing only the iris itself is stretched into rectangle, which is referred to as iris texture. Finally, different filters can be applied to the iris texture, each of which resulting in a different iris code, i.e., bitmaps formed by black and white pixels. The actual template is a bitmap composed by the combination of multiple iris codes generated using different filter parameters. For instance, the templates shown in Fig. 5(b) are formed by three iris codes vertically concatenated.

Standard iris based authentication works by xor-ing the bitmaps of the BT generated during enrollment with the new BT provided by the user during authentication. However, such methods are not applicable to fuzzy vaults (recall that the Fuzzy Vault requires a biometric template formed by a set of points in a field). Differently from fingerprints, the iris BT is not a set of data points that can be directly used to generate an FV. Therefore, the bitmap output must be encoded into multiple data points in such a way that these points can be matched together, by using a given distance function, during FV_{OPEN} computation.

To address this issue we develop a simple encoding mechanism. We divide the biometric template into a grid of 36 equally sized squares. Let $S[x, y]$ denote the square in line x column y of the template grid and $N(S[i, j])$ the number of black pixels in such square. The set of data points that compose the biometric template in this encoding scheme are defined as:

$$BT_U = \bigcup_{i=1}^{6} \bigcup_{i=1}^{3} [N(S[i, 7 - j]), N(S[i, j])] \quad (22)$$

The resulting BT contains 18 data points in the format $p = [x, y]$ where x and y are the number of black pixels in a pair of squares in the template. In our implementation we define the template resolution such that x and y can be represented in 12 bits each and, therefore, each data point can be encoded as an element of $GF(2^{24})$. These 18 points are mixed with 200 chaff points to generate the HD. Finally, to implement FV_{OPEN} we use the distance function:

$$D(p_i, p_j) = |x_i - x_j| + |y_i - y_j| \quad (23)$$

which measures the absolute difference in the number of black pixels in both squares that compose each data point. A point is considered a chaff point if such distance is larger than a threshold w. In our iris accuracy experiments in Section 5.3 we set $w = 100$.

The scheme above is one of multiple possibilities for encoding schemes and distance functions. As our focus in this work is not on biometric template extraction, but on non-interactive user re-enrollment using MPC, we use straight-forward encoding mechanism and distance functions. We refer the interested reader to [63] for more sophisticated methods to encode iris scans into templates

that are suitable for usage with FVs. While our simple iris encoding mechanism has modest accuracy results of $\approx 80\%$ genuine acceptant with $\approx 5\%$ false acceptance, these numbers could reach $>$ 99% GAR with nearly 0 FAR if an encoding scheme such as the one in [63] is used. It is worth emphasizing that the accuracy relates to the biometric template extraction which is orthogonal to SNUSE and thus not the focus of the present work.

5.3 Authentication Accuracy

As described above, our prototype supports fingerprints and iris scans. We evaluate the accuracy of using these two biometric modalities with respect to the two following metrics:

- **Genuine Acceptance Rate (GAR):** Percentage of rightful users that are successfully authenticated after providing the correct biometric sample.
- **False Acceptance Rate (FAR):** Percentage of users authenticated when providing unauthorized biometric sample.

Our accuracy experiments were conducted using two fingerprint and one iris datasets:

- **FVC2000-DB1:** Fingerprint database containing 10 fingers with 8 impressions each. Samples collected using a low-cost optical sensor[4].
- **FVC2000-DB2:** Fingerprint database containing 10 fingers with 8 impressions each. Samples collected using a low-cost capacitive sensor[5].
- **IIT Delhi Iris Database (Version 1.0):** Iris scans from 200 eyes with 10 impressions each. Samples collected using JPC1000 digital CMOS camera[6].

For each database we cross-check every possible pair of biometric samples using one of them to compute generate the HD by computing FV_{GEN} and the other to unlock the HD by computing FV_{OPEN}. Therefore, GAR is computed as the fraction of different impressions for the same biometric that are successfully authenticated during FV_{OPEN}. Conversely, FAR is the fraction of biometric samples that succeed in computing FV_{OPEN} in an HD generated using a biometric sample of some other user.

Recall that the polynomial degree in the fuzzy vault algorithm determines the number of data points correctly retrieved from the HD that are necessary to reconstruct the secret. Therefore, a low polynomial degree tends to increase GAR and FAR and a high polynomial degree tends to decrease both. We present the accuracy results for each of the aforementioned databases in Fig. 6, as a function of the polynomial degree. The degree can be calibrated, depending on the application case to yield appropriate accuracy. For example, in the fingerprint databases, a degree of 7 yields a GAR of ≈90% with a FAR of ≈3%. Degree 8 yields ≈80% GAR with 0 FAR. In the iris scan implementation a degree of 5 results in ≈80% GAR with ≈5% FAR.

4. Available at: http://bias.csr.unibo.it/fvc2000/
5. Available at: http://bias.csr.unibo.it/fvc2000/
6. Available at: http://www4.comp.polyu.edu.hk/~csajaykr/IITD/Database_Iris.htm

6 SNUSE Evaluation

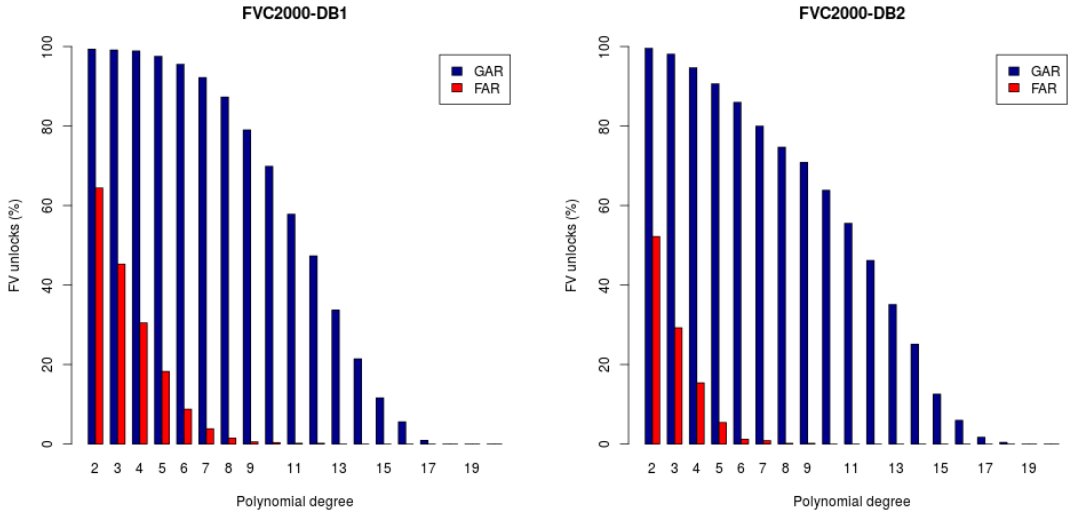
In this section we evaluate additional workload and storage requirements imposed by SNUSE. The choice for FV polynomial degrees and other parameters used in this evaluation was based on the results in Section 5.3. Such accuracy results show, for example, that for fingerprints, a polynomial with degree 7 yields a GAR of ≈90% with a FAR of ≈3%. A polynomial with degree 8 yields ≈80% GAR with 0 FAR. In our iris scans implementation, a polynomial with degree 5 results in ≈80% GAR with ∼5% FAR. In the following we proceed with the evaluation of computational workload in terms of time and storage required by SNUSE.

6.1 Computational Workload

In this section we evaluate the computational workload of SNUSE. We measure the time required to compute SNUSE protocols with special emphasis on the large scale user re-enrollment over a local enterprise network. This simulates the use case in which a company/enterprise wants to re-enroll all its employees/users refreshing their secrets/cryptographic keys. The experiments presented throughout this section were executed in an Intel Core i7-3770 octacore CPU @3.40GHz, with 16GB of RAM, running Linux (Ubuntu 14.04LTS).

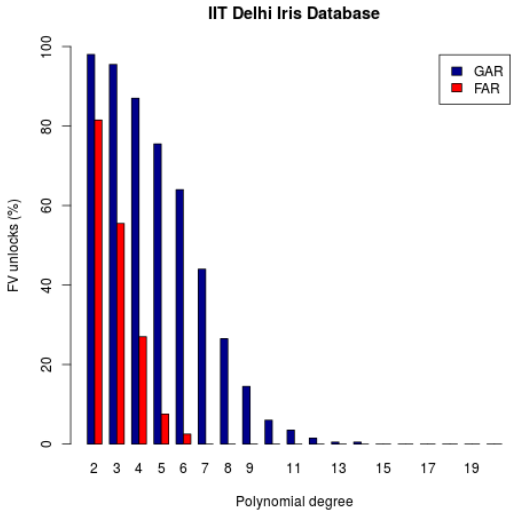
AS and the RESs were implemented as independent processes communicating though TCP sockets. An artificial delay of 10 milliseconds is introduced in order to simulate a typical communication delay for a local area network. Unless stated otherwise, the FV polynomial degree is set to 7 (which has shown to be a reasonable value according to the accuracy results reported in Section 5.3) and the number of RESs is set to 5. We use a (K, N) Shamir secret sharing scheme with $K = N$. Therefore, all secret shares are required to reconstruct the shared secret and, consequently, all parties are required to participate in the MPC protocol.

We start the evaluation by measuring the execution times of each of SNUSE's protocols: Enrollment, Authentication, and Re-Enrollment. Table 1 presents the average times and standard deviations (out of 100 independent executions) of each protocol for a single user. The results show that enrollment and authentication phases are considerably more time demanding (in the order of a second) than re-enrollment. This cost is a consequence of the BT extraction process, required in both phases. Since re-enrollment is performed based on the secret shares of the BT, it does not require BT extraction. Moreover, due to the exponentiation pre-computation optimization described in Section 4.4, it only requires one communication round between the AS and the RESs. As a consequence re-enrollment for one user is performed in 13 milliseconds, on average. In our next experiment, we



(a) Low-cost Optical Sensor

(b) Low-cost Capacitive Sensor



(c) Iris Scans

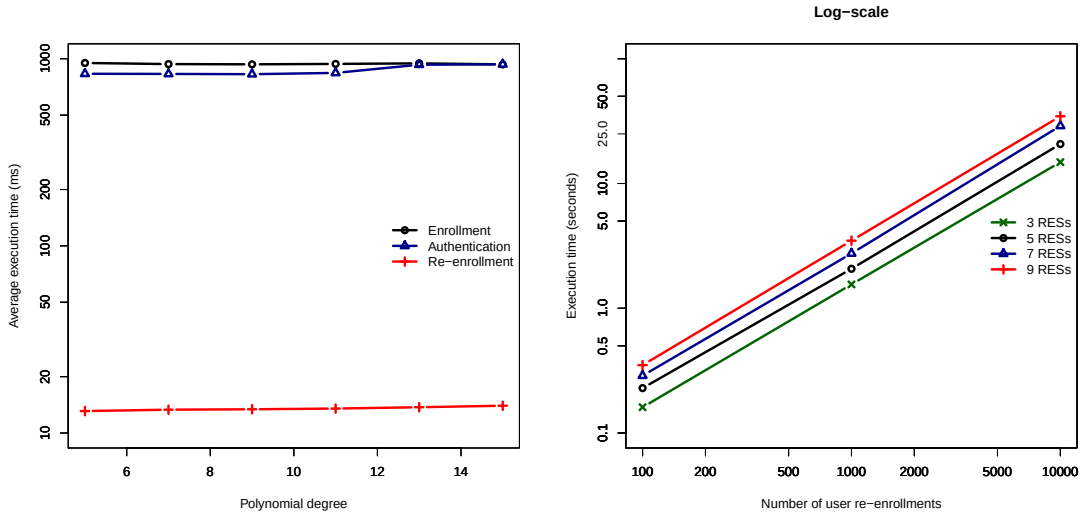
Fig. 6. SNUSE accuracy: GAR and FAR using fingerprint low-cost optical sensor, fingerprint low-cost capacitive sensor, and iris scans

Protocol	Avg. Time	Std. Dev.
Enrollment	945.9 ms	24.1 ms
Authentication	848.7 ms	26.2 ms
Re-Enrollment	13.2 ms	0.2 ms

TABLE 1
Execution times for SNUSE protocols

evaluate the effect that increasing the FV's polynomial degree has on the computation time of each algorithm. We consider reasonable polynomial degrees from 5 to 15 and compute the overall required time for each case. The results displayed on Fig. 7(a) show that the variation of polynomial degree has negligible impact on the execution time of the protocols. The only perceptible change is in the authentication protocol, that has a slightly higher execution time with high degrees. FV_{OPEN}, which is computed during authentication, executes multiple polynomial interpolations with the candidate data points selected by comparing the provided biometric template data points with the points in the HD. In this process, subsets of size $d+1$ (where d is the polynomial degree) are selected from the 20 candidate points, interpolated and the resulting secret is hashed and compared to the hash in the HD. Increasing d also increases the number of possible combinations, consequently increasing the number of polynomial interpolations. Therefore, the average computation time for authentication also increases by a total of 12% with a polynomial degree of 15, when compared to a polynomial degree of 5.

Finally, we evaluate how re-enrollment performs at



(a) (b)

Fig. 7. **a.** Execution time as a function of the FV's polynomial degree **b.** Processing time for massive user re-enrollments with varying number of RESs

large scale, in the order of thousands of simultaneous user re-enrollments. We run the large scale experiments using 3, 5, 7, and 9 RESs. The results for the large scale experiments are depicted in Fig. 7(b). We can see that the re-enrollment processing time scales linearly with the number of simultaneous re-enrollment requests. The re-enrollment time for SNUSE running with more RESs is also slightly higher than with smaller number of RESs, which is expected due to the higher amount of computation required. Nevertheless, our results indicate that the re-enrollment of massive number of users is affordable. For example, re-enrolling 100 thousand users would using 9 RESs would take less than five minutes and re-enrolling 1 million users would be possible in less than an hour.

6.2 Storage Requirements

SNUSE stores the HD in the AS and the biometric template secret shares in the RESs. The HD public parameters $\Phi = \{F, d, M, H(k)\}$ (recall that $H(k)$ is implemented with a SHA-256 hash function) by themselves require 39 bytes per HD. In our implementation, using $GF(2^{24})$, each element in the HD (data points and chaff points) can be encoded into 6 bytes. Considering an HD with 20 biometric data points and 200 chaff points, and HD for one user requires $220 \times 6 + 39 = 1359$ bytes. Considering that an additional user ID of 30 bytes is associated to each HD entry in the AS database, a massive number of user entries, e.g., 1 million, would require $(1359 + 30) \times 10^6 = 1.39$ GB of storage, which is well within the capacity of modern computers. Increasing the number of users or the number of chaff points used to construct the vault would increase this number linearly. The degree of the polynomial used in the fuzzy vault does not affect the storage requirements on the AS.

Each RES has to store one share of a BT per user. Therefore, the storage requirement depends on the size of such shares. In Section 4.4 we discuss three approaches that differ from each other in the number of multiplications required and consequently in the number of network communication rounds. In the first two approaches, each secret share will have the same size of the original biometric template, because no pre-computation of exponentiations takes place before the secret sharing. Therefore, considering a biometric template composed of 20 data points in $GF(2^{24})$, each secret share would require 60 bytes of storage plus 30 bytes for storing the associated user ID. In the case of pre-computing exponentiations before secret sharing, each secret share is in the format of the matrix in Eq. (19) of Section 4.4. In this latter case, the size of each secret share depends on the polynomial degree used in the fuzzy vault and on the number of data points. If $GF(2^{24})$ is used as the field for the scheme, each element in the secret share matrix can be encoded as 3 bytes. Considering a biometric template with 20 data points and a polynomial of degree 15 (which is a comfortable upper bound, based on the accuracy result we present in Section 5.3) each secret share would require $3 \cdot 20 \cdot 15 = 900$ bytes. With 1 million users this would result in a storage requirement of 900 MB per RES.

7 Security Analysis

7.1 Confidentiality of Stored Biometrics

In practice, backend servers may store thousands or even hundreds of thousands of BTs. Therefore, an adversary that compromises a backend server which stores

biometrics in clear can obtain this massive number of biometrics.

In SNUSE, compromising AS does not allow the adversary to reconstruct BT because AS only store HDs, and the infeasibility of reconstructing a BT from an HD is guaranteed by security of the underlying FV scheme, which in turn relies on the infeasibility of the polynomial reconstruction problem [16].

On the other hand, the adversary might attempt to retrieve the stored BTs by compromising the RESs. SNUSE's security, in this case, relies on the secret sharing scheme. In other words, if a (K,N) scheme is used, reconstructing BT in clear implies compromising at least K out of the N RESs. K can be made arbitrarily large according to the desired resilience in a given organization.

7.2 Confidentiality of Biometrics During Execution

The system does not store the BT at any backend server (RES or AS). In addition, the biometric is not reconstructed at the servers during the computation of enrollment, re-enrollment, or during user authentication. The biometric is only visible in clear at the B.T. Reader, during initial enrollment and regular authentication. However, there is no way to work around that, because B.T. Reader is the physical sensor device that samples the biometric. Moreover, since these sensors are simple devices (compared to backend servers), orthogonal trusted computing techniques such as remote attestation for low-end devices [64]–[66] can be efficiently used to ensure that such devices are operating as expected (i.e., not infected by Malware that could compromise BT confidentiality).

If the HD's secret (k) is visible to RESs (see Section 4.5 for further discussion), an adversary that is able to compromise one RES, during the execution of enrollment or re-enrollment protocols, obtains k. We argue that this is not as threatening as reconstructing the BT in clear. Note that, given the re-enrollment functionality provided by SNUSE, it is relatively easy to revoke and re-assign a fresh secret to a user. On the other hand, a user's biometric is usually tied to the individual through its whole life.

Regarding the biometric's confidentiality, compromising less than K RESs during the execution of the protocol does not reveal anything about BT. However, compromising at least one RES and the AS, at the same time, during enrollment/re-enrollment execution reveals the BT being computed at that time. This is due to the construction of the fuzzy vault scheme, which allows someone possessing the secret k and HD to tell apart which points in the HD are points in the polynomial encoding k and which ones are chaff points. Therefore, by having both k and HD one can reveal BT. This limitation can be addressed by using MPC to perform two things: (1) compute the HD of a FV from randomly generated secret shares $[k]_i$ of secret k chosen in a secure distributed manner, and (2) compute the permutation π. Another interesting direction is to develop an FE or FV scheme[7] that does not allow someone in possession of k and HD to reveal BT. We defer these as future work. Nevertheless, even in this case, the attack surface is much smaller, because of three reasons: (1) it requires compromising both AS plus one RES, (2) the attacker must have control of both of them during the protocol computation, and (3) even then only one BT (the one for which the FV is being computed at that time) is leaked, instead of the whole database.

7.3 Re-Usability of Fuzzy Vaults

Another important issue is the re-usability of traditional FVs and FEs. In FVs, for example, the possession of two different HDs generated from the same BT (e.g., a user authenticates with the same biometric in two different organizations) allows an adversary to tell apart chaff points from minutiae points. It would be interesting to implement MPC-based re-enrollment with a reusable FEs (e.g., [67]). We defer this direction as future work.

8 Conclusion and Future Work

We study the problem of non-interactive re-enrollment in cryptographically secured Biometrics-based Identification and Authentication (BIA) systems. We argue that addressing this issue is paramount for real-life deployments of such systems and to ensure long-term confidentiality of biometrics. We develop a new approach for Secure Non-interactive Users at Scale re-Enrollment (SNUSE) and prototype it. SNUSE does not affect the the accuracy of the underlying biometric feature extraction scheme and our experimental results, using standard computing servers, show efficient re-enrollment for thousands of users in a few seconds. Our prototype implementation achieves a high detection accuracy with over 90% Genuine Acceptance Rate (GAR) and less than 5% False Acceptance Rate (FAR).

The following are avenues for future work. First, our approach could be expanded to support other FE/FV schemes, e.g., computational and reusable ones. Second, it would be interesting to add support for other biometrics in addition to fingerprints and iris, and other types of devices, e.g., smart-phones. One challenge is that other biometrics have other notions of matching distance which may require development and optimizations of the secure computation circuits to be practical. In theory any regeneration algorithm and any notion of matching distance can be handled, but practical feasibility does not always follow from theoretical feasibility. Finally, our prototype is in the honest-but-curious model, extending it to handle fully malicious, or covert adversaries, without significant overhead is an interesting problem.

7. This property is *not* guaranteed by default by several FE/FV schemes, because it is not captured by their definitions.

REFERENCES

[1] A. Jain, R. Bolle, and S. Pankanti, *Biometrics: personal identification in networked society*. Springer Science & Business Media, 2006, vol. 479.

[2] N. K. Ratha, J. H. Connell, and R. M. Bolle, "Enhancing security and privacy in biometrics-based authentication systems," *IBM systems Journal*, vol. 40, no. 3, pp. 614–634, 2001.

[3] Wikipedia, "Office of Personnel Management data breach," https://en.wikipedia.org/wiki/Office_of_Personnel_Management_data_breach (accessed 2017-12-05).

[4] Touch ID. [Online]. Available: https://support.apple.com/en-us/HT204587

[5] A. Juels and M. Sudan, "A fuzzy vault scheme," *Designs, Codes and Cryptography*, vol. 38, no. 2, pp. 237–257, 2006.

[6] B. Fuller, X. Meng, and L. Reyzin, "Computational fuzzy extractors," in *International Conference on the Theory and Application of Cryptology and Information Security*. Springer, 2013, pp. 174–193.

[7] A. Shamir, "How to share a secret," *Communications of the ACM*, vol. 22, no. 11, pp. 612–613, 1979.

[8] V. Shoup, "Ntl: A library for doing number theory," *www.shoup.net/ntl/*, 2001.

[9] World's biggest biometric database grows in india amid doubt. [Online]. Available: https://www.bloomberg.com/news/articles/2017-12-13/world-s-biggest-biometric-database-grows-in-india-despite-doubts

[10] A. C. Yao, "Protocols for secure computations," in *Foundations of Computer Science, 1982. SFCS'08. 23rd Annual Symposium on*. IEEE, 1982, pp. 160–164.

[11] O. Goldreich, S. Micali, and A. Wigderson, "How to play any mental game," in *Proceedings of the nineteenth annual ACM symposium on Theory of computing*. ACM, 1987, pp. 218–229.

[12] T. Rabin and M. Ben-Or, "Verifiable secret sharing and multiparty protocols with honest majority," in *Proceedings of the twenty-first annual ACM symposium on Theory of computing*. ACM, 1989, pp. 73–85.

[13] D. Beaver, "Efficient multiparty protocols using circuit randomization." in *Crypto*, vol. 576. Springer, 1991, pp. 420–432.

[14] R. Cramer, I. Damgård, and U. Maurer, "General secure multi-party computation from any linear secret-sharing scheme," in *Advances in CryptologyEUROCRYPT 2000*. Springer, 2000, pp. 316–334.

[15] I. Damgård, V. Pastro, N. Smart, and S. Zakarias, "Multiparty computation from somewhat homomorphic encryption," in *Advances in Cryptology–CRYPTO 2012*. Springer, 2012, pp. 643–662.

[16] A. Kiayias and M. Yung, "Cryptographic hardness based on the decoding of reed-solomon codes," in *International Colloquium on Automata, Languages, and Programming*. Springer, 2002, pp. 232–243.

[17] C. Stergiou, K. E. Psannis, B.-G. Kim, and B. Gupta, "Secure integration of iot and cloud computing," *Future Generation Computer Systems*, vol. 78, pp. 964–975, 2018.

[18] V. M. Patel, R. Chellappa, D. Chandra, and B. Barbello, "Continuous user authentication on mobile devices: Recent progress and remaining challenges," *IEEE Signal Processing Magazine*, vol. 33, no. 4, pp. 49–61, 2016.

[19] M. S. Farash, M. Turkanović, S. Kumari, and M. Hölbl, "An efficient user authentication and key agreement scheme for heterogeneous wireless sensor network tailored for the internet of things environment," *Ad Hoc Networks*, vol. 36, pp. 152–176, 2016.

[20] M. Alhaidary, S. M. M. Rahman, M. Zakariah, M. S. Hossain, A. Alamri, M. S. M. Haque, and B. Gupta, "Vulnerability analysis for the authentication protocols in trusted computing platforms and a proposed enhancement of the offpad protocol," *IEEE Access*, vol. 6, pp. 6071–6081, 2018.

[21] A. Tewari and B. Gupta, "Cryptanalysis of a novel ultra-lightweight mutual authentication protocol for iot devices using rfid tags," *The Journal of Supercomputing*, vol. 73, no. 3, pp. 1085–1102, 2017.

[22] A. L. M. Neto, A. L. Souza, I. Cunha, M. Nogueira, I. O. Nunes, L. Cotta, N. Gentille, A. A. Loureiro, D. F. Aranha, H. K. Patil *et al.*, "Aot: Authentication and access control for the entire iot device life-cycle," in *Proceedings of the 14th ACM Conference on Embedded Network Sensor Systems CD-ROM*. ACM, 2016, pp. 1–15.

[23] B. B. Gupta, *Computer and Cyber Security: Principles, Algorithm, Applications, and Perspectives*. CRC Press, 2018.

[24] B. Gupta, D. P. Agrawal, and S. Yamaguchi, *Handbook of research on modern cryptographic solutions for computer and cyber security*. IGI global, 2016.

[25] K. Delac and M. Grgic, "A survey of biometric recognition methods," in *Proceedings. Elmar-2004. 46th International Symposium on Electronics in Marine*. IEEE, 2004, pp. 184–193.

[26] C. Rathgeb and A. Uhl, "A survey on biometric cryptosystems and cancelable biometrics," *EURASIP Journal on Information Security*, vol. 2011, no. 1, p. 3, 2011.

[27] O. S. Adeoye, "A survey of emerging biometric technologies," *International Journal of Computer Applications*, vol. 9, no. 10, pp. 1–5, 2010.

[28] S. P. Banerjee and D. L. Woodard, "Biometric authentication and identification using keystroke dynamics: A survey," *Journal of Pattern Recognition Research*, vol. 7, no. 1, pp. 116–139, 2012.

[29] T. Kaczmarek, E. Ozturk, and G. Tsudik, "Assentication: User deauthentication and lunchtime attack mitigation with seated posture biometric," in *International Conference on Applied Cryptography and Network Security*. Springer, 2018, pp. 616–633.

[30] K. B. Rasmussen, M. Roeschlin, I. Martinovic, and G. Tsudik, "Authentication using pulse- response biometrics," 2014.

[31] A. K. Jain, K. Nandakumar, and A. Ross, "50 years of biometric research: Accomplishments, challenges, and opportunities," *Pattern Recognition Letters*, vol. 79, pp. 80–105, 2016.

[32] G. Itkis, V. Chandar, B. W. Fuller, J. P. Campbell, and R. K. Cunningham, "Iris biometric security challenges and possible solutions: For your eyes only?using the iris as a key," *IEEE Signal Processing Magazine*, vol. 32, no. 5, pp. 42–53, Sept 2015.

[33] A. Juels and M. Sudan, "A fuzzy vault scheme," *Des. Codes Cryptography*, vol. 38, no. 2, pp. 237–257, Feb. 2006. [Online]. Available: http://dx.doi.org/10.1007/s10623-005-6343-z

[34] K. Nandakumar, A. K. Jain, and S. Pankanti, "Fingerprint-based fuzzy vault: Implementation and performance," *IEEE Transactions on Information Forensics and Security*, vol. 2, no. 4, pp. 744–757, Dec 2007.

[35] Y. Dodis, L. Reyzin, and A. Smith, *Fuzzy Extractors: How to Generate Strong Keys from Biometrics and Other Noisy Data*. Berlin, Heidelberg: Springer Berlin Heidelberg, 2004, pp. 523–540. [Online]. Available: https://doi.org/10.1007/978-3-540-24676-3_31

[36] X. Boyen, Y. Dodis, J. Katz, R. Ostrovsky, and A. Smith, "Secure remote authentication using biometric data," in *Proceedings of the 24th Annual International Conference on Theory and Applications of Cryptographic Techniques*, ser. EUROCRYPT'05. Berlin, Heidelberg: Springer-Verlag, 2005, pp. 147–163. [Online]. Available: http://dx.doi.org/10.1007/11426639_9

[37] B. Fuller, X. Meng, and L. Reyzin, "Computational fuzzy extractors," in *Advances in Cryptology - ASIACRYPT 2013 - 19th International Conference on the Theory and Application of Cryptology and Information Security, Bengaluru, India, December 1-5, 2013, Proceedings, Part I*, 2013, pp. 174–193.

[38] X. Boyen, "Reusable cryptographic fuzzy extractors," in *Proceedings of the 11th ACM Conference on Computer and Communications Security*, ser. CCS '04. New York, NY, USA: ACM, 2004, pp. 82–91. [Online]. Available: http://doi.acm.org/10.1145/1030083.1030096

[39] M. Blanton and M. Aliasgari, "On the (non-)reusability of fuzzy sketches and extractors and security in the computational setting," in *Proceedings of the International Conference on Security and Cryptography*, July 2011, pp. 68–77.

[40] ——, "Analysis of reusability of secure sketches and fuzzy extractors," *IEEE Transactions on Information Forensics and Security*, vol. 8, no. 9, pp. 1433–1445, Sept 2013.

[41] D. Apon, C. Cho, K. Eldefrawy, and J. Katz, "Efficient, reusable fuzzy extractors from LWE," in *Cyber Security Cryptography and Machine Learning - First International Conference, CSCML 2017, Beer-Sheva, Israel, June 29-30, 2017, Proceedings*, 2017, pp. 1–18.

[42] I. D. O. Nunes, K. Eldefrawy, and T. Lepoint, "Secure non-interactive user re-enrollment in biometrics-based identification and authentication systems," in *International Symposium on Cyber Security Cryptography and Machine Learning*. Springer, 2018, pp. 162–180.

[43] A. C.-C. Yao, "How to generate and exchange secrets," in *Proceedings of the 27th Annual Symposium on Foundations of Computer Science*, ser. SFCS '86. Washington, DC, USA: IEEE Computer Society, 1986, pp. 162–167. [Online]. Available: https://doi.org/10.1109/SFCS.1986.25

[44] M. Ben-Or, S. Goldwasser, and A. Wigderson, "Completeness theorems for non-cryptographic fault-tolerant distributed computation," in *STOC '88*. ACM.

[45] D. Chaum, C. Crépeau, and I. Damgard, "Multiparty unconditionally secure protocols," in *Proceedings of the Twentieth Annual ACM Symposium on Theory of Computing*, ser. STOC '88. New York, NY, USA: ACM, 1988, pp. 11–19. [Online]. Available: http://doi.acm.org/10.1145/62212.62214

[46] D. Beaver, *Foundations of Secure Interactive Computing*. Berlin, Heidelberg: Springer Berlin Heidelberg, 1992, pp. 377–391. [Online]. Available: https://doi.org/10.1007/3-540-46766-1_31

[47] G. Asharov, Y. Lindell, and T. Rabin, *Perfectly-Secure Multiplication for Any t ¡ n/3*. Berlin, Heidelberg: Springer Berlin Heidelberg, 2011, pp. 240–258. [Online]. Available: https://doi.org/10.1007/978-3-642-22792-9_14

[48] Z. Beerliová-Trubíniová and M. Hirt, *Perfectly-Secure MPC with Linear Communication Complexity*. Berlin, Heidelberg: Springer Berlin Heidelberg, 2008, pp. 213–230. [Online]. Available: https://doi.org/10.1007/978-3-540-78524-8_13

[49] R. Canetti, U. Feige, O. Goldreich, and M. Naor, "Adaptively secure multi-party computation," in *Proceedings of the Twenty-eighth Annual ACM Symposium on Theory of Computing*, ser. STOC

'96. New York, NY, USA: ACM, 1996, pp. 639–648. [Online]. Available: http://doi.acm.org/10.1145/237814.238015

[50] R. Canetti, Y. Lindell, R. Ostrovsky, and A. Sahai, "Universally composable two-party and multi-party secure computation," in *Proceedings of the Thiry-fourth Annual ACM Symposium on Theory of Computing*, ser. STOC '02. New York, NY, USA: ACM, 2002, pp. 494–503. [Online]. Available: http://doi.acm.org/10.1145/509907.509980

[51] I. Damgård, V. Pastro, N. Smart, and S. Zakarias, "Multiparty computation from somewhat homomorphic encryption," in *CRYPTO'12*. Springer-Verlag New York, Inc., 2012.

[52] I. Damgård, M. Keller, E. Larraia, V. Pastro, P. Scholl, and N. P. Smart, *Practical Covertly Secure MPC for Dishonest Majority – Or: Breaking the SPDZ Limits*. Berlin, Heidelberg: Springer Berlin Heidelberg, 2013, pp. 1–18. [Online]. Available: https://doi.org/10.1007/978-3-642-40203-6_1

[53] D. W. Archer, D. Bogdanov, B. Pinkas, and P. Pullonen, "Maturity and performance of programmable secure computation," *IEEE S & P*, 2016.

[54] Y. Aumann and Y. Lindell, "Security against covert adversaries: Efficient protocols for realistic adversaries," *J. Cryptol.*, vol. 23, no. 2, pp. 281–343, Apr. 2010. [Online]. Available: http://dx.doi.org/10.1007/s00145-009-9040-7

[55] Y. Huang, J. Katz, and D. Evans, "Quid-pro-quo-tocols: Strengthening semi-honest protocols with dual execution," in *IEEE S & P'12*. IEEE Computer Society, 2012.

[56] C. Hazay and Y. Lindell, "Efficient protocols for set intersection and pattern matching with security against malicious and covert adversaries," in *TCC'08*. Springer-Verlag, 2008.

[57] J. Baron, K. Eldefrawy, K. Minkovich, R. Ostrovsky, and E. Tressler, "5PM: Secure pattern matching," in *SCN'12*. Springer-Verlag, 2012.

[58] S. Jarecki, H. Krawczyk, M. Shirvanian, and N. Saxena, "Device-enhanced password protocols with optimal online-offline protection," in *Proceedings of the 11th ACM on Asia Conference on Computer and Communications Security*. ACM, 2016, pp. 177–188.

[59] K. Ko, "User's guide to nist biometric image software (nbis)," *NIST Interagency/Internal Report (NISTIR)-7392*, 2007.

[60] K. Nandakumar, A. K. Jain, and S. Pankanti, "Fingerprint-based fuzzy vault: Implementation and performance," *IEEE transactions on information forensics and security*, vol. 2, no. 4, pp. 744–757, 2007.

[61] B. Tams, "Absolute fingerprint pre-alignment in minutiae-based cryptosystems," in *2013 International Conference of the BIOSIG Special Interest Group (BIOSIG)*, Sept 2013, pp. 1–12.

[62] N. Othman, B. Dorizzi, and S. Garcia-Salicetti, "Osiris: An open source iris recognition software," *Pattern Recognition Letters*, vol. 82, pp. 124–131, 2016.

[63] Y. J. Lee, K. Bae, S. J. Lee, K. R. Park, and J. Kim, "Biometric key binding: Fuzzy vault based on iris images," in *International Conference on Biometrics*. Springer, 2007, pp. 800–808.

[64] K. Eldefrawy, G. Tsudik, A. Francillon, and D. Perito, "SMART: Secure and minimal architecture for (establishing dynamic) root of trust," in *NDSS*, vol. 12, 2012, pp. 1–15.

[65] F. Brasser, B. El Mahjoub, A.-R. Sadeghi, C. Wachsmann, and P. Koeberl, "Tytan: tiny trust anchor for tiny devices," in *2015 52nd ACM/EDAC/IEEE Design Automation Conference (DAC)*. IEEE, 2015, pp. 1–6.

[66] I. De Oliveira Nunes, K. Eldefrawy, N. Rattanavipanon, M. Steiner, and G. Tsudik, "Formally verified hardware/software co-design for remote attestation," *arXiv preprint arXiv:1811.00175*, 2018.

[67] D. Apon, C. Cho, K. Eldefrawy, and J. Katz, "Efficient, reusable fuzzy extractors from lwe," in *International Conference on Cyber Security Cryptography and Machine Learning*. Springer, 2017, pp. 1–18.