Oblivious Extractors and Improved Security in Biometric-based Authentication Systems

Ivan De Oliveira Nunes
Rochester Institute of Technology
Rochester, NY, USA

Peter Rindal
Visa Research
Palo Alto, CA, USA

Maliheh Shirvanian
Visa Research
Palo Alto, CA, USA

ABSTRACT

We study the problem of biometric-based authentication with template privacy. Typical schemes addressing this problem, such as Fuzzy Vaults (FV) and Fuzzy Extractors (FE), allow a server, *aka* Authenticator, to store "random looking" Helper Data (HD) instead of biometric templates in clear. HD hides information about the corresponding biometric while still enabling secure biometric-based authentication. Even though these schemes reduce the risk of storing biometric data, their correspondent authentication procedures typically require sending the HD (stored by the Authenticator) to a client who claims a given identity. The premise here is that only the identity owner – i.e., the person whose biometric was sampled to originally generate the HD– is able to provide the same biometric to reconstruct the proper cryptographic key from HD. As a side effect, the ability to freely retrieve HD, by simply claiming a given identity, allows invested adversaries to perform offline statistical attacks (a biometric analog for dictionary attacks on hashed passwords) or re-usability attacks (if the FE scheme is not reusable) on the HD to eventually recover the user's biometric.

In this work we develop Oblivious Extractors: a new construction that allows an Authenticator to authenticate a user without requiring neither the user to send a biometric to the Authenticator, nor the server to send the HD to the client. Oblivious Extractors provide concrete security advantages for biometric-based authentication systems. From the perspective of secure storage, an oblivious extractor is (provably) as secure as its non-oblivious fuzzy extractor counterpart. In addition, it enhances security against aforementioned statistical and re-usability attacks. To demonstrate the construction's practicality, we implement and evaluate a biometric-based authentication prototype using Oblivious Extractors.

KEYWORDS

Biometrics, Authentication, Fuzzy Extractors, Fuzzy Vault

1 INTRODUCTION

Biometric-based authentication systems have grown in popularity especially due to their ease of use and potential for increased security. In contrast with other traditional modes/factors of authentication, such as passwords/PINs ("something you know") and physical authentication tokens ("something you have"), biometrics do not require additional burden (e.g., to memorize a password or carry an authentication token around) on the users. Biometrics are a reasonably unique part of the user ("something you are") and therefore their usage for authentication is convenient.

Despite its tangible advantages, the use of biometrics for authentication also introduces unique security challenges. The storage of stable biometrics (stable refers to not changing much through the life-span of an individual – e.g., fingerprints, iris scans) also represent a privacy and security risk. In contrast with passwords/PINs or authentication tokens, stable biometrics cannot be changed. Therefore, leakage of biometric templates is a serious threat which unfortunately has already happened in large scale [1, 2]. In addition, typical measures used to protect the confidentiality of passwords/PINs, such as salted hashing, are not applicable to biometrics. This is because biometric samples are always slightly different from each other, due to noise and imperfections in the biometric sensor hardware and sensing process. Consequently, even with small noises, cryptographic hashes applied to biometrics result is completely different digests, making the matching of hashed templates infeasible.

Fuzzy Extractors (FE) [3] are cryptographic constructions that allow provably secure biometric storage and matching of noisy samples, thus enabling secure biometric-based authentication with biometric template confidentiality (we overview a concrete construction for a Fuzzy Extractor in Section 2.3). In a nutshell, an FE embeds a reference Biometric Template (BT) and a cryptographic key ($\mathcal{K}$) into random looking helper data (HD). Given that BT has sufficient entropy, then computation of $\mathcal{K}$ from HD is intractable. However, during authentication, if one is able to provide BT′ such that BT′ is "close enough" (within some configurable distance threshold) to BT, BT′ can be used in conjunction with HD to reconstruct $\mathcal{K}$, i.e., the same cryptographic key chosen during HD's generation. This property, in turn, allows a client and a server to agree upon a common secret if and only if the client is able to provide the same biometric registered to the server during user enrollment. Figure 1 depicts a typical user authentication procedure utilizing FE-generated HD.

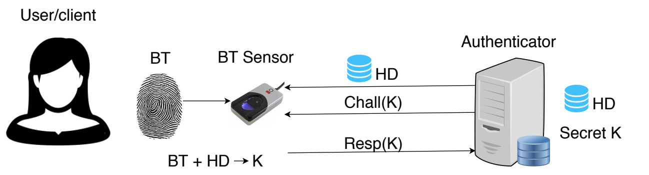

Figure 1: Typical authentication using FE

As shown in Figure 1, a user who wishes to authenticate starts by claiming an identity (e.g., a user ID). Authenticator then sends back to the user the HD corresponding to this identity. To reconstruct the authentication key ($\mathcal{K}$) from HD, the user provides a new sample of its own biometric. If the matching succeeds, $\mathcal{K}$ is reconstructed and can be used in a standard challenge-response authentication protocol. By authenticating in this way, the user's BT is never visible to the Authenticator.

We argue that this authentication approach has an intrinsic problem: anyone is able to retrieve HD by simply claiming an identity. The ability to retrieve HD allows invested attackers to perform offline attacks on HD to recover $\mathcal{K}$ and/or BT, e.g., [4, 5]. These attacks are analogous to a password-server sending the hashed password to a client which in turn would allow them to mount an offline dictionary attack.

Another possible attack is based on the lack re-usability of several practical FE schemes [6–9]. If an FE is not reusable, anyone able to obtain two (or more) instances of the scheme, i.e, HD_1 and HD_2, generated using the same template BT, is able to reconstruct BT in clear. In this case, an attacker can simply claim the user's identity at two different service providers, $\mathsf{Authenticator}_1$ and $\mathsf{Authenticator}_2$, to learn BT.

Considering these problems, in this work we propose a construction for Oblivious Extractors (OEs). OEs enjoy the same security guarantees as typical fuzzy extractors with respect to secure storage of biometric templates. However, the HD generation algorithm is constructed such that the corresponding authentication phase does not require the Authenticator to send the HD to the client nor the client to send the BT to the Authenticator. In such a setting, offline statistical attacks are not possible and online attacks can be throttled by having Authenticator to limit the maximum number of authentication attempts per user per time interval. Conversely, re-usability attacks are only possible if two or more enterprise databases (that store two or more HD for the same user) are simultaneously breached, given that the HD is no longer revealed during authentication. As it will become clear in Section 2.3, generic secure 2-party computation techniques (e.g., garbled circuits) are too heavyweight for authentication using FE. Instead, we propose a protocol, specific to FE-based authentication, based on cheaper and widely used primitives, namely oblivious programmable PRFs [10] and polynomial secret-sharing [11]. In summary we make the following contributions:

- We define a primitive called Oblivious Extractor (OE) along with a corresponding definition for its ideal functionality $\mathcal{F}_{\mathsf{OE}}$.
- We present an OE construction and analyze it, showing that it fulfills $\mathcal{F}_{\mathsf{OE}}$. In our construction, security of the HD to be persistently stored by the Authenticator is equivalent to that of a standard (non-oblivious) fuzzy extractor construction. In addition, our OE construction does not reveal any information about HD to passively corrupt clients. Against actively malicious clients – that deviate from the OE protocol specification – we show that leakage about HD is minimal and, whenever it happens, adversarial behavior on the client's part can be detected by the Authenticator with high probability. Upon detection, Authenticator can take further measures, e.g., reporting and blacklisting the malicious client.
- We implement an OE-based biometric authentication system using human fingerprints. We evaluate our prototype considering computation and communication requirements. Furthermore, we show that our scheme does not affect the accuracy of the underlying biometric matching.

1.1 Design Principles

In addition to the principal goal of providing better security via oblivious evaluation of the authentication function, our construction and system are designed with a set of secondary goals in mind. We believe that, by attaining this goals, our construction will have better usability and deployability, in addition to increased security:

1. **Biometric agnostic:** While some FE constructions work for specific distance functions (these are used to compare the features extracted from the biometrics), our scheme can be used with any distance function. Compatibility with any distance function makes the scheme flexible and applicable to different types of biometrics, as long as the their features can be encoded into a metric space. This encoding has been demonstrated for several popular types of biometric [12–14] with high matching accuracy.
2. **No trusted hardware requirements:** Several commercial biometric-based authentication systems, especially those deployed on smart-phones (e.g. FIDO [15]), rely on secure Hardware to perform the biometric matching. In these systems, the reference BT is stored in clear by the secure Hardware module and the matching is performed also in clear during authentication. The assumption is that the secure Hardware can not be breached and that its manufacturer can be trusted not to violate the user's privacy. We emphasize that, in a setting where this assumption is acceptable, our scheme can be used seamlessly as an additional layer of security. Additionally, hardware-based approaches do not scale to settings with multiple users and multiple authentication entry points, such as enterprise settings (see below). In these settings our construction might be especially applicable.
3. **Stateless authentication terminals:** Ideally, the system should not require that users always use the same (or a restricted set of) device(s) to authenticate. Consider, for example, the setting where Authenticator is a company that uses biometric-based authentication to grant physical access to its buildings and the users are the employees. Users must be able to authenticate from different physical entry points. This requires authentication terminals (i.e, the sensor devices that sample the biometric during authentication) to be stateless. Otherwise, only the terminal storing the authentication meta-data would be able to authenticate the corresponding user.

2 PRELIMINARIES

This section overviews the building-blocks used in this paper and introduces corresponding notation.

2.1 Biometric Template Matching

A Biometric Template (BT) is composed of features identifying the individual. In biometric matching applications (e.g., biometric-based authentication), first a BT is sampled and stored. This initial process is referred to as *enrollment*. Later, when a matching is required, the same feature extraction procedure is applied to collect a second BT'. This new BT' is compared to the one stored and, if their similarity exceeds a pre-defined threshold, the matching succeeds. We represent a BT corresponding to a user as a vector:

$$\mathsf{BT} = (b_1, ..., b_m) \in \mathbb{D}^m \tag{1}$$

where $b_1, ..., b_m \in \mathbb{D}$ are data points in some set $\mathbb{D}$ representing details of U's biometric. For instance, in fingerprints, each $b_i \in \mathsf{BT}$ typically represents the location and orientation of one of the fingerprint's *minutiae*. *Minutiae*, in turn, are regions in the fingerprint image in which fingerprint lines merge and/or split. In turn, each *minutiae* point is encoded as:

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

where, $\mathbb{D} = \mathbb{Z}^3$, $x_i, y_i \in \mathbb{Z}$ are Cartesian coordinates and $\theta_i \in \mathbb{Z}$ is the angle representing the orientation of the minutiae b_i. Similar encoding techniques can be used for other biometric modalities [12–14], such as iris scans and faces. We note that other representations are possible, for example, BT could be an embedding output by an appropriately trained neural network.

Fuzzy Vaults (FV) and more generally Fuzzy Extractors (FE) are cryptographic schemes that use an input BT to generate Helper Data (HD). HD encodes a secret k. It is hard to recover the secret k or BT from HD, unless prompted with BT′ close/similar enough to the original BT used to generate the HD. It then follows that even if the HD is leaked or made public, the BT is also hard to recover. Section 2.3 overviews a concrete example of such a construction and discusses its shortcoming against offline attacks.

2.2 Shamir's Secret Sharing

Shamir's K-out-of-N secret sharing [11] allows a dealer to split a secret in N shares such that subsets of at least K such shares enable recovering the original secret. Given a secret X, $[X]_j$ denotes the j-th secret share of X. Also, denote generation of N shares of X by:

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

Conversely, denote reconstruction of secret X from K shares by:

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

Such a scheme is implemented by selecting a polynomial (defined over a finite field) of degree $K-1$, i.e,:

$$P_X(x) = \sum_{i=0}^{(K-1)} a_i \times x^i, \tag{5}$$

where $a_0 = X$ (the secret) and all other a_i are random numbers in the field. Each share $[X]$ is a point in P_X, i.e, $(s, P_X(s))$ for some s. With N or more shares one can interpolate $P_X(x)$ (i.e., solving a linear system of equations) and find $a_0 = X$. Less than N shares reveal no information about secret X. Notably, shares $(s, P_X(s))$ where s is chosen as a random number are indistinguishable from a point (r_x, r_y) where both r_x and r_y are chosen randomly.

2.3 Fuzzy Vault Scheme

A Fuzzy Vault (FV) [16] is a practical construction for a Fuzzy Extractor (FE– defined in [17])[1]. It is designed to work with BTs that are represented as unordered sets of data points as shown earlier, in Equation 1. The scheme has two components:

(1) the points $\mathsf{BT} = (b_1, ..., b_m)$ are obfuscated by shuffling them with n random points, $r_1, ..., r_n \in \mathbb{D}$. The security of the scheme relies on the difficulty of identify the b_i points given the set $\{b_1, ..., b_m, r_1, ..., r_n\}$ (in random order). For this to hold it is critical that the r_i values are sampled from the same distribution as the b_i values.
(2) a mechanism to recover a hidden key k if the user can identify exactly $d+1$ of b_i points (see the definition of parameter d below).

In more detail, the FV scheme consists of two algorithms, (FV_{GEN}, FV_{OPEN}). The former is defined as a randomized algorithm

$$\mathsf{FV}_{GEN}(\mathsf{BT}, k) : \mathbb{D}^m \times \mathbb{F}_p \rightarrow \mathcal{H} \tag{6}$$

which takes U's biometric template BT as input, along with a key k sampled from large prime field $\mathbb{F}_p$. It outputs an instance of the helper data $\mathsf{HD} \in \mathcal{H}$. The scheme is further parameterized by some public parameters $m, n, d, w, p \in \mathbb{Z}$ and requires that $\forall_{i,j}$: $\mathsf{dist}(b_i, b_j) > w$ for distinct $i, j \in [m]$ where dist is some distance function (i.e., some metric). This is because points within distance w are in some sense considered to be the same across different impressions of the same biometric.

The generation algorithm samples n so called "chaff points" $r_1, ..., r_n \in \mathbb{D}$ from the same distribution[2] as $b_i \in \mathsf{BT}$. Let $\widetilde{\mathsf{BT}} = (\tilde{b}_1, ..., \tilde{b}_{m+n}) \in \mathbb{D}^{m+n}$ consist of the b_i, r_i points in a random order. As with the b_i values, the r_i values are sampled such that $\forall i, j : \mathsf{dist}(\tilde{b}_i, \tilde{b}_j) > w$ for distinct i, j.

The algorithm then samples a random polynomial $P \in \mathbb{F}[x]$ of degree $d < m$ such that $P(0) = k$, similar to the polynomial in a Shamir secret sharing scheme (see Section 2.2), where k is the secret being shared. For $\tilde{b}_i \in \tilde{\mathsf{BT}}$, if $\tilde{b}_i \in \mathsf{BT}$ then let[3] $v_i = P(\tilde{b}_i)$ and otherwise uniformly sample $v_i \leftarrow \mathbb{F}$. Finally, the algorithm outputs the helper data as $\mathsf{HD} = ((\tilde{b}_1, v_1), ..., (\tilde{b}_{m+n}, v_{m+n}), H(k)) \in \mathcal{H}$ where $\mathcal{H} = (\mathbb{D} \times \mathbb{F})^{m+n} \times \{0,1\}^\kappa$ and $H : \{0,1\}^* \rightarrow \{0,1\}^\kappa$ is a random oracle.

The FV_{OPEN} algorithm can then recover the key $k \in \mathbb{F}_p$ given a close enough biometric BT′ and HD:

$$\mathsf{FV}_{OPEN}(\mathsf{BT}', \mathsf{HD}) : \mathbb{F}_p \tag{7}$$

Close enough here means that more than d points (where d is the polynomial degree) in BT's are less than w apart from points in the original BT, i.e.:

$$|\{b'_i \in \mathsf{BT}', s.t. \exists [b_j \in \mathsf{BT} \wedge \mathsf{dist}(b'_i, b_j) \leq w]\}| > d. \tag{8}$$

As such, the parameters w, d control how similar the two biometrics must be for it to be considered a match and therefore also control the trade-off between false positive/negative rates during authentication.

In FV_{OPEN}, first the set $S = \{(b'_i, v_i) \in \mathsf{HD}$ s.t. $\exists [b_j \in \mathsf{BT}' \wedge \mathsf{dist}(b'_i, b_j) \leq w]\}$ is computed. Then for each subset S' of S s.t. $|S'| = d+1$, the algorithm interpolates the points $(\tilde{b}_i, v_i) \in S'$ to obtain the polynomial $P(x)$. If $H(P(0)) = H(k)$, then the algorithm

[1]We note, however, that the original fuzzy vault scheme [16] was proposed before the formalization of the concept of a fuzzy extractor in [17].

[2]FVs/FEs typically assume that each b_i is iid and sampled from some distribution $\mathcal{D}$, enabling a proof of security. In practice, this assumption may not hold and proper sampling is an ongoing research challenge [4, 5].

[3]Here, we assume that $\tilde{b}_i \in \mathbb{D}$ can be interpreted as an element of $\mathbb{F}$. This can be achieved by defining an injective or random function $\phi : \mathbb{D} \rightarrow \mathbb{F}$ and defining $v_i = P(\phi(\tilde{b}_i))$.

will output $P(0)$. If no such subset S' exists, then the algorithm outputs $\perp$.

If we apply this scheme in the traditional manner, the overall protocol then consists of:

(1) **Enrollment** – the user U enrolls by interacting with a trusted enrollment device. The enrollment device generates fresh $k \leftarrow \mathbb{F}_p$ and HD from U's BT and sends $k \leftarrow \mathbb{F}_p$ and HD $\leftarrow$ FV$_{GEN}$(BT, k) to the Authenticator. Authenticator persistently stores this data associated with the newly created user identity.

(2) **Authentication** –Later, when a client wishes to authenticate as U, the Authenticator will send the associated HD back to the client. The authentication will succeed if the client can successfully answer a challenge which requires knowledge of the key k, e.g., standard challenge-response protocols based on the encryption of nonces.

Ideally, this protocol would achieve the following security guarantees. When the user U enrolls, the helper data HD reveals no information about BT to Authenticator (nor any other entities aside from the trusted enrollment device itself – e.g., a biometric sensor). Similarly, the online authentication procedure would not reveal information about the newly supplied biometric BT$'$ to Authenticator, apart from whether it matched or not.

One method of formalizing this is with an indistinguishably based security definition. For example, given two distinct biometrics (BT$_1$, BT$_2$), the adversary should not be able to distinguish the distribution of HD$_1 \leftarrow$ FV$_{GEN}$(BT$_1$, k) from HD$_2 \leftarrow$ FV$_{GEN}$(BT$_2$, k). However, these two are trivial to distinguish since the FV$_{OPEN}$ algorithm must be efficient. Moreover, BT$_1$, BT$_2$ are directly contained in HD$_1$, HD$_2$ respectively. And yet, given that there are sufficiently many chaff points, HD does to some extent obfuscate the original biometric BT. In particular, this allows a weaker security notion. Let us assume that all BT $= (b_1, ..., b_m)$ are generated such that each b_i is sampled iid from some distribution $\mathcal{D}$ over $\mathbb{D}$. Then it follows that the adversary has a negligible in k probability of outputting k given HD alone. To see why, recall that HD $= ((\tilde{b}_1, v_1), ..., (\tilde{b}_{m+n}, v_{m+n}), H(k))$ and since all b_i, r_j are iid (by assumption), so are all of the $\tilde{b}_i$ values. As such, the adversary is tasked with identifying a set of m-out-of-$(m+n)$ points $(\tilde{b}_i, v_i)$ which lay on a degree-d polynomial where each $\tilde{b}_i \leftarrow \mathcal{D}$ and all but $m - d$ v_i values are uniform in $\mathbb{F}$. For appropriately set parameters, this problem is conjectured to be intractable [16].

We note however that, in practical deployments, biometrics might have significantly less entropy than the computational security parameter k [4, 5]. As such, statistical guessing of the biometric template BT could allow for an adversary to recover k from HD with noticeable probability. Moreover, since HD contains a hash of the k (and $m > d+1$ points that lie in the polynomial), the adversary can perform such an attack in an offline setting (after receiving HD in clear) and check whether or not the correct k (or the correct polynomial) was obtained.

Looking forward, we will mitigate this attack by not sending the helper data HD to the each user U' that claims an identity and requests to authenticate. This limits the exposure of HD to only the Authenticator. Since in many cases we can assume the Authenticator is honest, they will not perform such brute force attacks (this assumption is equivalent to that in current password-based authentication servers storing salted hashes). However, in the unlikely event that they do become corrupted, e.g. hacked, then the adversary is still tasked with performing a potentially expensive offline attack in order to recover the underlying biometric BT and key k. This can give the organization the crucial amount of time to mitigate the potential fallout.

The syntax for the FV construction and respective notation are summarized in Definition 1. Definitions 2 and 3 state FV's completeness and security guarantees.

Definition 1 (Fuzzy Vault (FV) Syntax).
A Fuzzy Vault is defined as FV $=$ (FV$_{GEN}$, FV$_{OPEN}$, Φ), *where* Φ *is a set of parameters* $\Phi = (m, n, d, \mathbb{F}, \mathbb{M}, \mathtt{dist}, \mathtt{w})$*:*
- m is the number of biometric features, referred to as ***minutiae*** *points.*
- n is the number of randomizing features, referred to as ***chaff*** *points.*
- d is a polynomial degree;
- $\mathbb{F}_p$ is a prime field with size $p-1$;
- $\mathbb{M}$ is a metric space;
- $\mathtt{dist}$ *is some distance function defined over* $\mathbb{M}$*;*
- $\mathtt{w}$ *is a distance threshold;*
FV$_{GEN}$ *and* FV$_{OPEN}$ *are algorithms:*

- FV$_{GEN}$*:*
  - ***Inputs****: k and* BT*, s.t., $k \in \mathbb{F}_p$.*
  - ***Output****:* HD
- FV$_{OPEN}$*:*
  - ***Inputs****:* HD *and* BT$'_U$
  - ***Output****: $k' \in \mathbb{F}_p$.*

Definition 2 (FV-Completeness).
FV $=$ (FV$_{GEN}$, FV$_{OPEN}$, Φ) *is complete with (w, d)-fuzziness if for every possible k and every pair* BT, BT$'$ *such that,*

$$|\{b'_i \in \mathrm{BT}', s.t. \exists [b_j \in \mathrm{BT} \wedge dist(b'_i, b_j) \leq w]\}| > d, \quad (9)$$

it holds that:

$$\mathrm{FV}_{OPEN}(\mathrm{FV}_{GEN}(k, \mathrm{BT}), \mathrm{BT}') = k \quad (10)$$

with overwhelming probability.

Definition 3 (FV-Security).
FV $=$ (FV$_{GEN}$, FV$_{OPEN}$, Φ) *is p-secure if a Probabilistic Polynomial Time (P.P.T.) adversary with access to* HD*, where:*

$$\mathrm{HD} = \mathrm{FV}_{GEN}(k, \mathrm{BT}) \quad (11)$$

is able to guess either, BT *or k, with success probability of at most p.*

2.4 Oblivious Programmable PRF

An Oblivious Programmable PRF (OPPRF) is a two party functionality consisting of a sender and receiver. The functionality is shown in Figure 2. The sender has a set of input pairs $(y_1, z_1), ..., (y_n, z_n)$ with distinct y_i. The functionality samples a key k such that $F_k(y_i) = z_i$ and at all other input points it outputs a random value. The receiver on input points $x_1, ..., x_n$ then obtains $F_k(x_i)$ for all i.

Parameters: There are two parties, a sender with input $L = \{(y_1, \tilde{y}_1), ..., (y_{n_y}, \tilde{y}_{n_y})\}$ where $y_i \in \mathbb{F}, \tilde{y} \in \{0,1\}^{\text{out}}$ and a receiver with a set $X \subseteq \mathbb{F}$ where $|X| = n_x$.

Functionality: Upon input $(\text{sender}, \text{sid}, L)$ from the sender and $(\text{receiver}, \text{sid}, X)$ from the receiver, the functionality samples a random function $F : \mathbb{F} \longrightarrow \{0,1\}^{\text{out}}$ such that $F_k(y) = \tilde{y}$ for each $(y, \tilde{y}) \in L$ and sends $X' := \{F_k(x) \mid x \in X\}$ to the receiver. Subsequently, upon input $(\text{sender}, \text{sid}, y)$ from the sender, the functionality returns $F(y)$ to the sender.

Figure 2: Ideal functionality $\mathcal{F}_{\text{opprf}}$ of Oblivious Programmable PRF.

This functionality can be realized from a standard OPRF along with polynomial interpolation or a similar encoding method. Loosely speaking, the sender samples a normal OPRF key k and sends the minimum degree polynomial P such that $P(y_i) = z_i - F_k(y_i)$. The parities compute the final output as $F_k(x) + P(x)$ where F_k is evaluated via the OPRF protocol. See [10] for efficient constructions.

As we explain in Section 3, our OE construction leverages OPPRFs to enable efficient oblivious computation of FV_{OPEN} while keeping input BT' private to the Client and input HD private to Authenticator.

3 OBLIVIOUS EXTRACTOR: INTUITION

Our Oblivious Extractor (OE) construction is based on a few simple observations that we discuss through the rest of this section. This section omits some protocol details in order to convey the general idea. Detailed specifications are presented in Section 4.

First, we note that *checking if two points are within a certain distance threshold from each other* is equivalent to *generating the set of all points that are within a certain threshold from the first point and checking for existence of the second point in the generated set.* More formally, for any distance function `dist` and two elements a and b in $\mathbb{F}$:

$$\begin{aligned} &\texttt{dist}(a,b) < w \equiv |\{b\} \cap A| = 1 \quad \text{where} \\ &A = \{a_i \mid \texttt{dist}(a, a_i) < w\} \end{aligned} \tag{12}$$

This is because set A contains all points in $\mathbb{F}$ that are sufficiently close (given threshold w and metric `dist`) to a, therefore, b must exist in set A if it is within this proximity. We note that this approach works because $\mathbb{F}$ is discrete (as opposed continuous spaces such as real numbers in $\mathbb{R}$), and $|A| = O(\text{poly}(k))$ is reasonably small for our application.

This observation allows us to use an oblivious set membership operation to obliviously perform distance-based matching of each $b'_1, ..., b'_m$ in BT$'$ to each $\tilde{b}_1, ..., \tilde{b}_{m+n}$ in HD. This matching is equivalent to the one performed in clear by the regular FV. More importantly, equation 12 is independent of the particular `dist` used for the feature matching. Thus, distance matching based on oblivious set membership testing can in principle be used to match biometric features of multiple biometric modalities, e.g., iris scans, faces, etc.

In addition to minutiae-to-minutiae matching, the regular FV also verifies if at least d minutiae are matched correctly, where d is the threshold defined by the polynomial degree (see Section 2 for details). To achieve the same property, our scheme relies on Shamir's secret sharing.

In a nutshell, the modified HD is generated by $\text{OE}_{GEN}(\text{BT}, k)$ via the following process:

(1) OE_{GEN} generates $\widetilde{\text{BT}} = (\tilde{b}_1, ..., \tilde{b}_{n+m})$ where a random subset of $\widetilde{\text{BT}}$ is in BT while the remainder are random chaff points. As such, it obfuscates the original BT in the exact same way as in the original FV scheme.
(2) For each chaff point, uniformly sample an associated random pair $(x_i, y_i) \leftarrow \mathbb{F}^2$. For each $\tilde{b}_j \in \widetilde{\text{BT}}$ that is a real *minutiae* in BT, sample a random point that lies on a $d-1$ degree polynomial P (i.e. $P(x_j) = y_j$ for all j such that $\tilde{b}_j \in$ BT) and require that $P(0) = k$.
That is, every pair (x_j, y_j) that is associated with a real biometric point from BT forms one Shamir secret share of k.
(3) Output HD $= (\widetilde{\text{BT}}, X, Y, H(k))$ as the helper data, where $\widetilde{\text{BT}} = (\tilde{b}_1, ..., \tilde{b}_{n+m}), X = [x_1, ..., x_{n+m}]$ and $Y = [y_1, ..., y_{n+m}]$. By construction, it holds that for all $j \in [n+m]$ such that $\tilde{b}_j \in$ BT, $y_j = P(x_j)$. In other words, every position j that is associated with a real *minutiae* is also associated with a secret share of k. On the other hand, positions that contain chaff points are associated to random (x', y') pairs that do not lie in the polynomial P.

Given HD and a sufficiently similar biometric BT$' = \{b'_1, ..., b'_m\}$, k can be recovered by interpolating the correct (x_j, y_j) pairs which are identified based on the condition that $\text{dist}(b'_j, \tilde{b}_i) < w$ for some $b'_j \in$ BT$'$. Given that there may be several degree at most $d-1$ polynomials which fit this criteria, the correct one can be identified by requiring $H(P(0)) = H(k)$.

We note, however, that there are several challenges when converting this basic idea into an oblivious protocol. First is how to evaluate the distance function. A *naïve* method would be for all $O(m^2)$ possible i, j to check if $\text{dist}(b'_j, \tilde{b}_i) < w$ either using a generic 2PC scheme or via the idea of directly turning $\tilde{b}_i$ into a set A and performing a set membership test (e.g., using off-the-shelf protocols for private set operations, e.g., [18–22], which often use OPRFs in some form). Though possible, this would be very inefficient.

Secondly, it is critical that the Client does not learn if the binary result of $\text{dist}(b'_j, \tilde{b}_i) < w$ since this would leak if some information about each $\tilde{b}_i$ in HD. For example, a Client could query the Authenticator many times and enumerate all elements in $A = \{a_j \mid \text{dist}(a_j, \tilde{b}_i) < w\}$ and therefore learn a $\tilde{b}_i$ exactly.

We address both of these issues simultaneously with the use of an OPPRF. The idea is that, during authentication, Authenticator will sample an OPPRF key k' such that for all $i \in [n+m]$ and $a_j \in \{a_j \mid \text{dist}(a_j, \tilde{b}_i) < w\}$, the OPPRF outputs $F_{k'}(a_j) = (x_i, y_i)$. Recall that when the Client evaluates the OPPRF, they will receive either the programmed (x_i, y_i) value if they input one of the corresponding a_j values or they will receive a uniformly random (x', y') pair.

Let us assume that BT, BT′ are not similar. Therefore the Client learns at most d pairs (x_i, y_i) which correspond to the actual biometric BT. These (x_i, y_i) pairs lay on the degree d polynomial P while all others are uniformly random. Recall that it takes $d + 1$ pairs to reconstruct P and therefore the key $k = P(0)$ remains uniformly distributed in the view of the Client, since they are lacking at least one pair. Moreover, the Client can not distinguish if they obtained a programmed point (x_i, y_i) or a uniformly random point (x', y') since both are distributed uniformly random. Critically, we require that the Client only inputs $b'_j \in \mathsf{BT}'$ values which are at least distance $2w$ apart to ensure that no two b'_j fall into the same set $A = \{a_j \mid \mathsf{dist}(a_j, \tilde{b}_i) < w\}$.

Now consider the case in which BT, BT′ are similar. From the OPPRF evaluation, the Client will learn at least $d + 1$ pairs (x_i, y_i) which do lay on the degree d polynomial P (in addition to possibly some points that do not lay in P, because there might a small number of chaff points that are coincidentally close to some of the points in BT′). As such, the client can use the obtained set of points to try to interpolate all subsets of $d + 1$ points, resulting in a degree d polynomial P' at each attempt. For each interpolation, the client checks if $H(P'(0)) = H(k)$. If so, it learns that $P = P'$ and outputs $k = P'(0)$.

4 OBLIVIOUS EXTRACTOR IN DETAIL

4.1 Definitions

OE consists of two sub-protocols: Enroll and Auth. Each sub-protocol instance involves a Client and an Authenticator. Figure 4 presents OE ideal functionality $\mathcal{F}_{\mathsf{OE}}$. It answers to two queries, modeling the ideal behavior of sub-protocols Enroll and Auth.

A query to Enroll is accompanied by a reference biometric template BT (obtained securely during initial user enrollment) and parameter c, determining the maximum number of authentication attempts possible within the life-time of the particular HD to be generated. It outputs a user ID i to Client and generates a user credential in the system, represented by the Client's ID i, an associated HD and c, to be stored by Authenticator. The ideal functionality records BT, and HD and k, computed using $\mathsf{FV}'.\mathsf{Gen}(\mathsf{BT})$.

A query to Auth is initiated by Client and must contain a claimed user ID i and corresponding input biometric template BT′. The functionality verifies if there exists a registered user with ID i and if the limit c' of authentication attempts for that particular user has not been exceeded. If these checks succeed, the query returns k if BT' is sufficiently close to the reference BT and $\perp$ otherwise. k and BT used in this step are the same recorded during Enroll for ID i. Every Auth query decrements associated c to record the authentication attempt. Figure 3 shows illustrates the OE authentication protocol, with a detailed construction in Figure 5.

4.2 Construction

This section presents an OE construction fulfilling $\mathcal{F}_{\mathsf{OE}}$ (Figure 4) in the honest-but-curious model. The protocol is specified in Figure 5.

Public parameters include two random oracles H and H' and the FV scheme described in Section 2.3, including the FV parameters themselves (e.g., a metric dist, a distance threshold w, polynomial degree d, etc). Before any sub-protocol interactions, Authenticator initializes a monotonically increasing counter $id := 0$ representing unique IDs assigned to users upon successful enrollment.

Enroll:

The first part of the enrollment protocol (up to the generation of $\widetilde{\mathsf{BT}}$) remains similar to the regular FV scheme, discussed in Section 2.3. BT is sampled from the user yielding m biometric data points, sufficiently distant from each other by threshold $2w$ for chosen metric dist. A set of n chaff points are randomly sampled following the same distribution as real biometric points and also obeying the sparsity restriction (for threshold $2w$ and dist). The set of real biometric data points and chaffs are shuffled according to permutation π selected uniformly at random. The resulting shuffled list of pairs is denoted $\widetilde{\mathsf{BT}}$.

Following generation of $\widetilde{\mathsf{BT}}$, Enroll will sample randomness $r \leftarrow\!\!\$\ \mathbb{F}$ and c random polynomials defined over $\mathbb{F}$. The independent/constant term in all c random polynomials is set to r (i.e., for $j \in [c]$, $P_j(0) = r$). Each P_j is used as an independent instance of a Shamir secret sharing scheme to sample m shares of r (in the form $(x \leftarrow\!\!\$\ \mathbb{F}, P_j(x))$). For each P_j, two lists X_j and Y_j are created using the m shares. X_j and Y_j are constructed such that if index i of $\widetilde{\mathsf{BT}}$ (after shuffling) contains a real biometric data point (i.e., $\tilde{b}_i \in \mathsf{BT}$), then $Y_{j,i} = P_j(X_{j,i})$ – where $X_{j,i}$ and $Y_{j,i}$ are used to denote the i-th element of X_j and Y_j, respectively. For all other indices, elements of X_j and Y_j are selected independently, uniformly at random. The HD is then given to (and persistently stored by) Authenticator composed of $\widetilde{\mathsf{BT}}$, $X := (X_1, ..., X_c)$, $Y := (Y_1, ..., Y_c)$, $h := \mathsf{H}(r)$, $e := \mathsf{H}'(r) \oplus k$. As it will become clear, a pair of lists $X_{c'}$, $Y_{c'}$ is consumed on each Auth interaction.

Given random shuffling of $\widetilde{\mathsf{BT}}$, sufficiently large number of chaff points, and indistinguishability between Shamir secret shares and random elements in $\mathbb{F} \times \mathbb{F}$ (present in X and Y), HD produced by OE hides BT and k from Authenticator. More formally, FV security can be reduced to OE security.

Auth:

To authenticate, a user initiates an interaction with Authenticator by claiming an identity id' and locally sampling $\mathsf{BT}' := (b'_1, ..., b'_m)$ at the Client machine. Authenticator looks up HD based on claimed id'. Authenticator also checks if the maximum number of authentication attempts allowed for the lifetime of the associated HD has not been exceeded, aborting otherwise. In practical systems that employ throttling to prevent online guessing, an additional check should occur to determine if the maximum number of attempts within a pre-defined time-window (e.g., 10 attempts per day) has been exceeded. This step is omitted from the protocol for simplicity. If the aforementioned checks succeed, Authenticator will initiate an instance of the oblivious biometric matching phase, based on BT′ (in possession of Client) and $\mathsf{HD}_{id'} = (\widetilde{\mathsf{BT}}, X, Y, h, e)$ (stored by Authenticator associated to id').

The c'-th instance of Auth consumes list $X'_c \in X$ and list $Y'_c \in Y$. To prevent information leakage across multiple executions of Auth, each X'_c and Y'_c pair is only used once, hence the cap c on the number of Auth interactions per HD. For each element

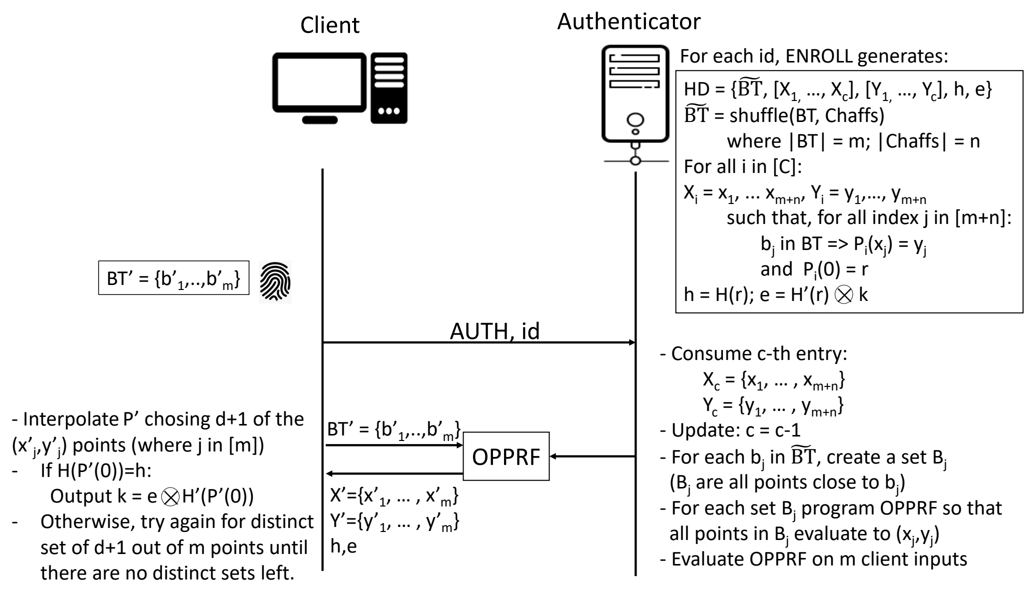
Client
Authenticator
For each id, ENROLL generates:
HD = {$\widetilde{BT}$, [X$_1$, ..., X$_c$], [Y$_1$, ..., Y$_c$], h, e}
$\widetilde{BT}$ = shuffle(BT, Chaffs)
where |BT| = m; |Chaffs| = n
For all i in [C]:
X$_i$ = x$_1$, ... x$_{m+n}$, Y$_i$ = y$_1$,..., y$_{m+n}$
such that, for all index j in [m+n]:
b$_j$ in BT => P$_i$(x$_j$) = y$_j$
and P$_i$(0) = r
h = H(r); e = H'(r) ⊗ k
BT' = {b'$_1$,..,b'$_m$}
AUTH, id
- Consume c-th entry:
X$_c$ = {x$_1$, ... , x$_{m+n}$}
Y$_c$ = {y$_1$, ... , y$_{m+n}$}
- Update: c = c-1
- For each b$_j$ in $\widetilde{BT}$, create a set B$_j$
(B$_j$ are all points close to b$_j$)
- For each set B$_j$ program OPPRF so that
all points in B$_j$ evaluate to (x$_j$,y$_j$)
- Evaluate OPPRF on m client inputs
BT' = {b'$_1$,..,b'$_m$}
OPPRF
X'={x'$_1$, ... , x'$_m$}
Y'={y'$_1$, ... , y'$_m$}
h,e
- Interpolate P' chosing d+1 of the
(x'$_j$,y'$_j$) points (where j in [m])
- If H(P'(0))=h:
Output k = e ⊗ H'(P'(0))
- Otherwise, try again for distinct
set of d+1 out of m points until
there are no distinct sets left.

Figure 3: Illustration of OE **authentication combining** HD**, oblivious set membership testing via OPPRF, and secret sharing.**

Parameters: An Authenticator and one or more clients, each generically denoted as Client. A FV scheme described in Section 2.3.
Functionality: Initialize id := 0. The functionality answers the following queries.

(1) Upon receiving $(\textsc{Enroll}, \mathrm{BT}, \mathrm{HD}, k, c)$ from Client, where $(\mathrm{HD}, k) \leftarrow \mathrm{FV}'.\mathrm{Gen}(\mathrm{BT}, k, c)$, record the tuple $(\mathrm{id}, c, \mathrm{BT}, \mathrm{HD}, k)$. FV$'$ is defined as the procedure which computes HD in Figure 5 (step f of Enroll).
Output (i) to Client and (i, c, HD) to Authenticator. Update id as id := id + 1.

(2) Upon receiving $(\textsc{Auth}, i, \mathrm{BT}')$ from Client and $(\textsc{Auth}, i)$ from Authenticator, if there exists a tuple $(i, c, \mathrm{BT}, \mathrm{HD}, k)$ for i such that
$$c > 0 \wedge |\{b'_i \in \mathrm{BT}', s.t. \exists [b_j \in \mathrm{BT} \wedge \mathrm{dist}(b'_i, b_j) \leq w]\}| > d, \text{ where } d \text{ is the polynomial degree of HD,}$$
then output (k, BT^*) to Client where BT* are these d' points in BT$'$ which are similar to BT. Otherwise output $\perp$ to Client. Update $c = c - 1$.

Figure 4: Ideal oblivious extractor functionality $\mathcal{F}_{\mathbf{OE}}$.

$\tilde{b}_i \in \widetilde{\mathrm{BT}}$ – including both chaff and real biometric points (recall that Authenticator cannot distinguish between them) – Authenticator generates the set of all points in $\mathbb{F}$ that are sufficiently close to $\tilde{b}_i$, i.e., all $b \in \mathbb{F}$ such that $\mathrm{dist}(b, \tilde{b}_i) < w$. All such b close to $\tilde{b}_i$ are associated to the same pair $(X_{c',i}, Y_{c',i})$ and added to a list L, where each $l \in L$ is in the form $(b, (X_{c',i}, Y_{c',i}))$, i.e., $l \in (\mathbb{F}, \mathbb{F}^2)$. As a result, L contains all points b in $\mathbb{F}$ that are sufficiently close to any $\tilde{b}_i \in \widetilde{\mathrm{BT}}$. By construction (recall Enroll sub-protocol), all $\tilde{b}_i \in \mathrm{BT}$ and close enough points appear in L associated to a secret share of randomness r. On the other hand, all $\tilde{b}_i \notin \mathrm{BT}$ (i.e., chaff points) and close enough points are associated to random pairs in $(\mathbb{F} \times \mathbb{F})$.

To perform oblivious authentication, Client and Authenticator invoke $\mathcal{F}_{\mathrm{OPPRF}}$ on their respective inputs: $\mathrm{BT}' = (b'_1, ..., b'_m)$ and L. For each $b'_u \in \mathrm{BT}'$, if b'_u is sufficiently close to any point in $\widetilde{\mathrm{BT}}$ (real or chaff), it also exists in L, thus Client receives an associated pair $(X_{c',v}, Y_{c',v})$, for some index $v \in [m + n]$. If b'_u is in fact close to a real biometric data point from BT (the reference template used to construct HD in Enroll), it is also the case that $Y_{c',v} = P'_c(X_{c',v})$, i.e., Client receives a secret share of randomness r (recall from Enroll that $P'_c(0) = r$). If b'_u does not exist in L (b'_u is close neither to real biometric data points nor chaff points), $\mathcal{F}_{\mathrm{OPPRF}}$ returns a random element from $(\mathbb{F} \times \mathbb{F})$.

Given the degree d of P'_c, if at least $d + 1$ points in BT$'$ are sufficiently close to points in BT, Client retrieves enough shares of r to reconstruct k. Most importantly, if less than $d + 1$ points are sufficiently close to points in the original BT, Client cannot distinguish any of the received elements from random in $(\mathbb{F} \times \mathbb{F})$, irrespective of whether each element was generated as a share of r, as random pair during construction of $X_{c'}$ and $Y_{c'}$) (see Enroll), or as a result of $\mathcal{F}_{\mathrm{OPPRF}}$ evaluation on an element that does not exist in L and thus has not been programmed by the OPPRF. It follows that, if Client fails to authenticate, nothing is learned by Client about BT or HD. At the same time, BT and BT' are hidden from Authenticator.

Upon completion of Auth, Authenticator decrements c'. This assures that fresh $X_{c'}$ and $Y_{c'}$ are used in different Auth sessions

Parameters: An Authenticator and one or more clients, each generically denoted as Client. An FV scheme described in Section 2.3 (and associated parameters, e.g., dist, m, n, w, d, etc). Two random oracles $\mathsf{H} : \mathbb{F} \to \{0,1\}^{\kappa}, \mathsf{H}' : \mathbb{F} \to \{0,1\}^{\kappa}$.

Protocol: Authenticator will initialize id := 0.

[Enroll] Upon the command (Enroll, BT, k, c) from Client, the Client performs

(a) **[Parse]** Parse $(b_1, ..., b_m) = \mathsf{BT}$ where $b_i \in \mathbb{D}$. Abort if for distinct $i, i' \in [m]$, $\mathrm{dist}(b_i, b_{i'}) > w$.
(b) **[Add Chaff]** Sample $b_{m+1}, ..., b_{m+n} \leftarrow \mathcal{D}$ s.t. for all distinct $i, i' \in [m+n]$, $\mathrm{dist}(b_i, b_{i'}) > w$.
(c) **[Shuffle]** Sample a random permutation $\pi : [m+n] \to [m+n]$ and define $\widetilde{\mathsf{BT}} := (\tilde{b}_1, ..., \tilde{b}_{m+n})$ where $\tilde{b}_i := b_{\pi(i)}$.
(d) **[Secret Share]** Sample $r \leftarrow \mathbb{F}$. For $j \in [c]$, sample a random degree d polynomial $P_j \in \mathbb{F}[x]$ such that $P_j(0) = r$.
(e) **[Shares]** Sample $X_j \leftarrow \mathbb{F}^{n+m}$ and define $(x_{j,1}, ..., x_{j,m+n}) := X_j$. For $j \in [c], i \in [n+m]$, if $\tilde{b}_i \in \mathsf{BT}$ then define $y_{j,i} := P_j(x_{j,i})$. Otherwise define $y_{j,i} \leftarrow \mathbb{F}$. Let $Y_j := (y_{j,1}, ..., y_{j,m+n})$. Define $X := (X_1, ..., X_c)$ and $Y := (Y_1, ..., Y_c)$.
(f) **[Output]** Send $\mathsf{HD} := (\widetilde{\mathsf{BT}}, X, Y, h, e)$ to Authenticator where $h := \mathsf{H}(r), e := \mathsf{H}'(r) \oplus k$.

Authenticator receives HD and sends id back. Authenticator records the tuple (id, c, HD) and increments id as id := id+1.

[Auth] Upon the command (Auth, id$'$, BT$'$) from Client, Authenticator looks up the tuple (id$'$, c', HD). If none exists, Authenticator sends back $\perp$. Otherwise let $(\widetilde{\mathsf{BT}}, X, Y, h, e) := \mathsf{HD}$ and $X, Y \in \mathbb{F}^{c' \times (m+n)}$. If $c' = 0$, send $\perp$ to Client and abort. Otherwise:

(a) **[Program OPPRF]** Let $(\tilde{b}_1, ..., \tilde{b}_{m+n}) := \widetilde{\mathsf{BT}}$. Define $L \in (\mathbb{F} \times \mathbb{F}^2)$ where for each $i \in [m+n]$ and $b \in \{b \in \mathbb{D} \mid \mathrm{dist}(\tilde{b}_i, b) < w\}$, it holds that $(b, (X_{c',i}, Y_{c',i})) \in L$.
(b) **[Invoke OPPRF]** Authenticator and Client invoke $\mathcal{F}_{\mathsf{OPPRF}}$ where Authenticator is the sender with input L. The Client inputs $\mathsf{BT}' = (b'_1, ..., b'_m)$, restricted that for distinct $i, i' \in [m]$, $\mathrm{dist}(b'_i, b'_{i'}) > w$, and receives $(x'_i, y'_i) = F(b'_i) \in \mathbb{F}^2$ for $i \in [m]$.
(c) **[Remove Row]** Authenticator updates X, Y by removing rows $X_{c'}, Y_{c'}$ and $c' := c' - 1$.
(d) **[Interpolate]** Authenticator sends h and e to Client. For each $S_j \subset [m]$ of size $d+1$, Client defines the degree d polynomial $P_{S_j} \in \mathbb{F}_p[x]$ s.t. $P_S(x'_i) = y'_i$ for all $i \in S_j$.
(e) **[Output]** If there exists a S_j s.t. $\mathsf{H}(P_{S_j}(0)) = h$, then Client outputs $k := \mathsf{H}'(P_{S_j}(0)) \oplus e$. Otherwise Client outputs $\perp$.

Figure 5: Oblivious extractor protocol Π_{OE}.

even with the same HD, preventing leakage/linkability across multiple/successive authentication attempts.

5 SECURITY ANALYSIS

Our OE construction does not affect the FV-Completeness and FV-Security guarantees provided by the original FV scheme. For completeness, this follows from the equivalence in Eq. 12 (implying that the accuracy of the distance-based matching of individual elements in BT and BT$'$ is not affected) and the fact that secret shares used in our scheme are generated with the same polynomial degree (therefore, the number of individual matches required to reconstruct k is the same). For security, we note that the only difference in the HD stored by Authenticator in OEs versus that stored by Authenticator in FVs is that the polynomial is evaluated on additional randomly generated points during enrollment. The obfuscation of BT, by shuffling minutiae and chaff points, which yields FV security notion (per analysis in [16]) is still performed in the exact same way as in the original scheme.

Therefore, in the remainder of this section, we stress to prove that our Π_{OE} protocol securely realizes the $\mathcal{F}_{\mathsf{OE}}$ functionality of Figure 4 in the semi-honest UC model [23]. In practical terms this means that the messages received during the protocol can be simulated given only the input of that party and the output of the ideal functionality $\mathcal{F}_{\mathsf{OE}}$.

Proof. **Corrupt** Authenticator.

First, we consider a semi-honest Authenticator. When interacting with the ideal functionality, Authenticator receives (i, c, HD) each time a Client enrolls. By definition this is effectively the same information that Authenticator receives from the Client in the real interaction, i.e. the simulator outputs HD to Authenticator.

As discussed in Section 2.3, it is the case that HD reveals some information about BT. However, the functionality explicitly allows Authenticator to learn this information. Moreover, this leakage is inherently required for this type of functionality due to the possibility of Authenticator running the Auth protocol with themselves and thereby learn information about BT.

In the ideal world Authenticator participates in the Auth protocol by sending (Auth, i) to the ideal functionality. They receive no output from the functionality. In the real protocol the view of Authenticator consists of the $\mathcal{F}_{\mathsf{OPPRF}}$ query, which they also receive no output from. Therefore the simulation follows directly. □

Proof. **Corrupt** Client.

For a corrupt Client, the view of the Enroll protocol is trivial to simulate. Effectively, it consists of the Client receiving their identifier id. This is also provided by the ideal functionality which the simulator can forward to the Client.

For proving the security of the Auth protocol we consider two cases. The first is the corrupt Client is authenticating on a id which they registered or one which an honest party registered. In the former, the simulation is to simply run the real protocol. Observe that this is secure due to the adversary already knowing the underlying HD value.

The most interesting case is the latter, when a corrupt Client requests to authenticate on an id which was registered by an honest user. The view of the Client in the ideal worlds is either k if their biometric BT' matches and otherwise $\perp$.

Let us assume that the biometric does not match and therefore the simulator obtained $\perp$ from $\mathcal{F}_{\mathsf{OE}}$. In the real protocol recall that $X_{c'}, Y_{c'}$ consists of $m+n$ values in F. Out of these a random set of m lay on a degree $d-1$ polynomial. Since the functionality would have output $\perp$, the Client would have received at most $d-1$ of the points which lay on the degree $d-1$ polynomial. Critically, the distribution of these points (and all others) are uniformly random. Therefore, the simulator will simply sample a uniformly random set of points and use these in place of $X_{c'}, Y_{c'}$. The view of the Client is identical when modeled in the $\mathcal{F}_{\mathsf{OPPRF}}$-hybrid.

In the case that there is a match, the simulator learns the key k and the $d' \geq d$ biometric points BT^* which matched from the functionality. With this the simulator can identify which of the $X_{c'}, Y_{c'}$ points should lay on a degree $d-1$ polynomial. Since there was a match there are $d' \geq d$ such points. The simulator samples $X_{c'}, Y_{c'}$ such that these points lay on a random degree $d-1$ polynomial P which has $P(0)=k$ while all other points are uniform. The Client will then reconstruct k as described by the protocol. □

5.1 **Active Malicious** Authenticator

We argue that an actively malicious Authenticator does not gain any advantage. To see why, note that no message sent by Client depends on Authenticator behavior. Hence, an actively malicious Authenticator does not learn Client's inputs that an honest-but-curious Authenticator would not.

The remaining possibility is to deviate from the protocol to tamper with Client's output (i.e., k). At best, this case prevents Client from authenticating itself to Authenticator, hence causing Authenticator to refuse access/service to Client. However, a malicious Authenticator can always refuse service to a Client, irrespective of the OE scheme (e.g., by simply ignoring Client's request to authenticate).

5.2 **Active Malicious** Client

Leakage in the case where Client may deviate from the protocol is due to the fact that a malicious Client may not respect the restriction that, for a new biometric sample provided for authentication $(b'_1, ..., b'_m) = \mathsf{BT}'$, it must hold that for distinct $i, i' \in [m]$, $\mathsf{dist}(b'_i, b'_{i'}) > w$. As a consequence, different instance of the OPPRF for different b'_i and $b'_{i'}$ in the same BT' may yield the same result if both points lie close enough to the same point in $\widetilde{\mathsf{BT}}$. In turn, this allows Client to learn whether or not a point close to b'_i and b'_j (either real biometric point or chaff point) exists in $\widetilde{\mathsf{BT}}$. Though strictly better than sending HD as a whole to Client, this still reveals some small amount of information about HD's structure, which may be undesirable.

To prevent this, the Client should always be required to prove to Authenticator usage of sufficiently distant b'_i and $b'_{i'}$ for all distinct i, i'. Importantly, this proof should not reveal anything about b'_i and $b'_{i'}$ to Authenticator, which can be hard and expensive to achieve in practice and may significantly complicate the protocol.

Instead, we suggest a simpler approach based on a slightly modified version of the protocol presented in Section 4.2. Instead of preventing, it allows Authenticator to detect Client's malicious behavior whenever a Client learns that small piece of information about HD. Upon detection, a malicious Client device can be blacklisted and banned from the system. This modified version of the protocol works as follows:

(1) Let S be the set of all points that are not sufficiently close to any point in $\widetilde{\mathsf{BT}}$, i.e., if $(b, (X, Y)) \notin L$ for some X and Y (see L in **[Program OPPRF]** step in Figure 5), then $b \in \mathsf{S}$.
(2) Before the start of the protocol Authenticator generates an additional key K_m and produces $|\mathsf{S}| + m + n$ shares of K_m in an m out of $(|\mathsf{S}| + m + n)$ Shamir secret sharing scheme.
(3) L in **[Program OPPRF]** step of Figure 5 is augmented to also include every element in S. Each $b \in \mathsf{S}$ is programmed with a distinct random pair $(X, Y) \in \mathbb{F}^2$.
(4) All elements in the new augmented L are also programmed with a secret share of K_m restricted that: for all $(b_i, (X_i, Y_i))$ and $(b_{i'}, (X_{i'}, Y_{i'}))$ in L, if $(X_i, Y_i) = (X_{i'}, Y_{i'})$, then b_i and $b_{i'}$ are programmed to yield <u>the same share</u> of K_m.
(5) Conversely, for all $(b_i, (X_i, Y_i))$ and $(b_{i'}, (X_{i'}, Y_{i'}))$ in L, if $(X_i, Y_i) \neq (X_{i'}, Y_{i'})$, then b_i and $b_{i'}$ are programmed to yield <u>different shares</u> of K_m.

The basic idea behind this approach is that any honest Client authentication terminal that follows the protocol will always receive m secret shares $[K_m]$ and will be able to reconstruct it and prove knowledge of K_m to Authenticator irrespective of whether the user succeeded in authenticating herself using the biometric.

On the other hand, a Client authentication terminal that cheats by selecting b'_i and b'_j close to each other and learns that in fact a corresponding point exists in $\widetilde{\mathsf{BT}}$ (because both of them return the same (X, Y) pair) will also be unable to reconstruct K_m. This client will obtain at most $m-1$ shares of K_m, because at least one of the shares will be repeated. Hence, the Client will fail to prove knowledge of K_m. Therefore, Authenticator is able to detect this malicious behavior and block/ban the malicious Client authentication terminal accordingly.

6 IMPLEMENTATION AND EVALUATION

To evaluate OE's practicality, we implement a prototype of an OE-based authentication system using fingerprint biometrics. This section describes this implementation and respective evaluation.

6.1 Fingerprint Pre-Processing & Parameters

Pre-processing and extraction procedures generate a biometric template BT from a fingerprint image. As discussed in Section 2, each data point in BT is the position and orientation (x_i, y_i, θ) of a fingerprint minutiae. To extract the BT we use NIST Biometric Image Software (NBIS) [24]. NBIS returns a set of identified minutiae points with corresponding confidence levels. From NBIS output, we select 20 points with the highest confidence and encode them as data points in $\mathbb{F}$. Following the FV implementation guidelines

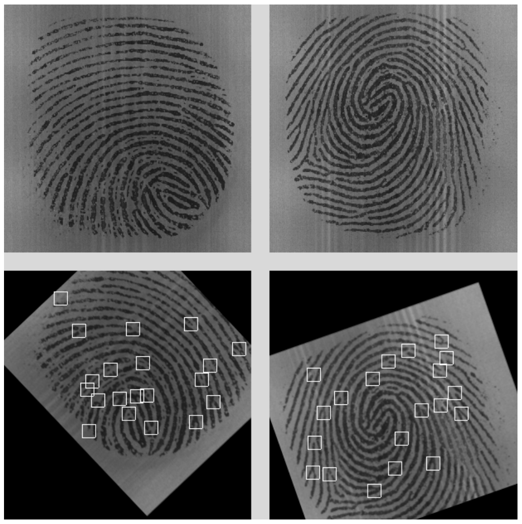

Figure 6: Fingerprint pre-processing and feature extraction examples highlighting identified minutiae points in white squares.

from [25] and [26], in our prototype, we implement OE using the following set of public parameters $\Phi = (m, n, d, \mathbb{F}, \mathbb{M}, \texttt{dist}, \texttt{w})$:

- Number of minutiae $m = 20$;
- Number of chaff points $n = 200$;
- Polynomial degree $d = 9$;
- $\mathbb{F}$ is a prime field with prime of at least 128 bits;
- Distance threshold $w = 20$;
- The distance function used to compare fingerprint features (based on the empirical characterization from [26]) and generate the sets is given by:

$$D(b_i, b_j) = \sqrt{(x_i - x_j)^2 + (y_i - y_j)^2} + 0.2 \times min(|\theta_i - \theta_j|, 360 - |\theta_i - \theta_j|) \tag{13}$$

The OE polynomial degree is set to 9 (also based on [26]). Finite field polynomial operations were implemented using the Number Theory Library (NTL) [27].

In **Auth**, points are matched from HD based on their distance to minutiae points in the new template BT′ sampled from the user. Similar to [26], we the distance function between $p_i \in$ HD and $p'_j \in$ BT′ defined as in Equation 13. These parameters must be empirically calibrated to yield the best accuracy results. We rely on these parameters based the work by Nandakumar et al. [26], that focuses on biometric matching accuracy with FVs. To improve accuracy results for noisy fingerprint readings before extracting the template, during the biometric sampling, we also run the fingerprint pre-alignment algorithm from [28]. Figure 6 illustrates the result of the template extraction for two pre-aligned fingerprint images. White squares highlight the $n = 20$ minutiae points detected in these fingerprints.

Remark 4: *We implement our own* BT *extraction for the sake of completeness, so as to have a fully working prototype and report on its accuracy. We stress that there is no difference in the accuracy of oblivious versus regular* FVs*, which is determined by the underlying biometric pre-processing techniques. These techniques are orthogonal and not affected by our work.*

6.2 Performance Analysis

Setup: Results presented in this section reflect measurements performed on an Intel Core i7-3770 octa-core CPU @3.40GHz, with 16GB of RAM, running Linux (Ubuntu 18.04LTS). Client and Authenticator 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 (LAN).

Protocol	Avg. Time	Std. Dev.
OE$_{GEN}$ (User Enrollment)	945.9 ms	24.1 ms
OE$_{OPEN}$ (User Authentication)	1.2 s	0.1 s

Table 1: Total execution times for OE protocols.

Operation	Avg. Time	Std. Dev.
OPPRF (Authenticator)	340 ms	71 ms
Network delay (2 LAN RTTs)	20 ms	none
Poly. Interpolations (Client)	876 ms	26.2 ms

Table 2: Break-down of operations in OE$_{OPEN}$ for parameters specified in Section 6.1.

Our protocol has two main costs, the OPPRF and the Client performing interpolation. We implement the OPPRF based on the protocol of [29] with optimizations provided by [30, 31]. The resulting overhead is that a OPPRF with n programmed points has communication overhead of $32 * 1.3 * n$ bytes plus a small setup cost of [31]. Using the parameter specified above this results in programming approximately $n = 154000$ points with an overhead 5.6MB per authentication and requires 0.34 seconds. The successive interpolation operations to reconstruct k take on average 876ms for the selected parameters.

Accuracy of the underlying biometric matching is not affected by our use-case. Improving its accuracy is an orthogonal effort. Nonetheless, for completeness, we report on the accuracy considering the implementation used in our prototype. Similar accuracy analysis for biometric matching using fuzzy vaults (also considering other biometrics modalities) can be found in [12, 25, 26]. We report on our prototype's accuracy considering metrics for:

- **Genuine Acceptance Rate (GAR):** Percentage of biometric samples correctly matched to other samples acquired from the same biometric.
- **False Acceptance Rate (FAR):** Percentage of biometric samples incorrectly matched to any sample not acquired from the same biometric.

We conducted accuracy experiments using FVC2000 publicly available[4] fingerprint database. FVC2000 includes multiple fingerprint

[4]Database and further information available at: http://bias.csr.unibo.it/fvc2000/.

images (10 different noisy images of each fingerprint) acquired using 4 types of low-cost biometric sensors. As discussed in Section 2.3, the polynomial degree allows configuring the number of matching data points in two biometric samples necessary to consider that the samples belong to the same user. Therefore, accuracy results are presented as a function of FV polynomial degree in Table 3.

Polynomial Degree (d)	GAR	FAR
5	97.70%	21.93%
6	93.96%	11.48%
7	89.65%	4.84%
8	86.20%	2.04%
9	80.17%	0
10	70.68%	0

Table 3: OE_{OPEN} accuracy as a function of polynomial degree

According to the results in Table 3, for a security-critical task such as authentication, an ideal choice would be degree 9 with zero false acceptances. The same degree results in GAR of 80%, meaning that 1 out of 5 times a genuine user would be rejected and required to attempt authentication again.

7 RELATED WORK

Confidentiality of biometrics has been widely studied over the last decade. These works resulted in cryptographic schemes that can be used as building blocks in systems utilizing biometrics. Fuzzy Vaults (FVs) [16] were developed (and implemented in [32]) to ensure privacy of reference biometric templates (BTs). An FV generates random looking data from BTs, only storing such data (HD) in the backends, and is still able to authenticate users from HD. Subsequently, the notion of Fuzzy Extractors (FE) was formalized, and derived from secure sketches [17], and also applied to biometrics [33]. Most FV/FE provide statistical security. Computational FE schemes were only recently introduced [34]. These computational FE schemes rely on hardness of the Learning With Errors (LWE) problem. FE re-usability was identified as an important issue to ensure security for repeated usage with the same biometric. Re-usability enables one to extract multiple HD from the same biometric without leaking any additional information. Not every FE/FV can be reused and still ensure security (illustrated in [8, 9]). In fact, from the two Helper Data HD-1 and HD-2, created with two instances of the scheme on the same biometric, an attacker can learn the original biometric inputs. New (indistinguishability based) definitions for re-usability were presented [35] and theoretical analysis demonstrated that the computational FE scheme in [34] is not (weakly or strongly) reusable.

Secure two/multi-party computation (2PC/MPC) protocols enable mutually distrusting parties to compute functions of their private inputs, while guaranteeing output correctness and input privacy, against misbehaving parties. 2PC/MPC has become increasingly practical over the past three decades [36–47] with both, generic (that allow evaluation of any computable function) and function-specific methods. SNUSE [25] used MPC in the context of biometric-based authentication to address the FV re-enrollment issue, i.e., how to refresh users' cryptographic keys without requiring active user participation to re-sample the BT. In SNUSE secret sharing and MPC, in the honest-but-curious model, are used to enable non-interactive re-enrollment in biometric authentication systems based on FV/FE. It focuses on biometric template privacy and non-interactive re-enrollment, hence, it does not target oblivious authentication using FE, which is the core focus of this paper. Other approaches [48–50] use 2PC/MPC to verify whether a biometric (e.g., a face) exists in a database, a problem commonly referred to as "identification". We target the related yet different problem of oblivious authentication with template privacy "vis-a-vis" Authenticator, which demands not only oblivious fuzzy matching of templates, but also subsequent key agreement for cryptographic operations (e.g.,challenge-response protocols, decryption).

The problem of strengthening password-based user authentication has also been well-studied under the common umbrella of Password-Authenticated Key Agreement (PAKE) protocols [51–55]. Since they work with passwords, PAKE protocols typically rely on Client and Authenticator sharing and storing the exact same secret (either in clear or hashed form). Therefore, they are not appropriate for authentication using biometrics, in which Authenticator and Client inputs are always slightly different. To enable authentication from noisy versions of a common secret, as in the case of biometric-based authentication, the notion of fuzzy PAKE (fPAKE) protocols was introduced in [56]. However, fPAKE protocols do not handle the case where the reference secret (e.g., the biometric) must also be cryptographically protected when stored at the Authenticator. We believe OE bridges this gap by enabling oblivious biometric-based authentication from HD, instead of requiring a reference BT to be stored in clear by Authenticator.

8 CONCLUSION

In this work we defined a new primitive called Oblivious Extractor (OE). We argued that OEs could be used to enhance the security of existing biometric-based authentication systems and provide examples of such applications. Finally we proposed, implemented, and evaluated a concrete construction for OEs.

There remain several possibilities for future work. Specifically, we believe that further improving the computation and communication costs of our scheme is a relevant direction, that would increase the chances of practical adoption. We also highlight the possibility of proposing OEs that are provably secure from other fuzzy extractor assumptions (e.g., computational). Our security analysis focused on passive adversaries with covert security against malicious clients. While we believe this adversary model covers security requirements of many practical biometric-based authentication settings, developing a protocol secure in the malicious model would be interesting from a theoretical perspective, and a promising future direction. Finally, it would be interesting to study the different practical settings in which oblivious extractors may be useful.

REFERENCES

[1] W. Post, "5.6 million fingerprints stolen in cyberattack." https://www.washingtonpost.com/news/the-switch/wp/2015/09/23/opm-now-says-more-than-five-million-fingerprints-compromised-in-breaches/?noredirect=on&utm_term=.669777a1e5ce (accessed 2018-12-03).

[2] CNN, “Alleged breach of india’s biometric database could put 1.2bn users at risk.” https://www.cnn.com/2018/01/11/asia/india-security-breach-biometric-database-intl/index.html (accessed 2018-12-03).

[3] Y. Dodis, L. Reyzin, and A. Smith, “Fuzzy extractors: How to generate strong keys from biometrics and other noisy data,” in *International conference on the theory and applications of cryptographic techniques*, pp. 523–540, Springer, 2004.

[4] C. Rathgeb and A. Uhl, “Statistical attack against iris-biometric fuzzy commitment schemes,” in *CVPR 2011 WORKSHOPS*, pp. 23–30, IEEE, 2011.

[5] B. Tams, P. Mihăilescu, and A. Munk, “Security considerations in minutiae-based fuzzy vaults,” *IEEE Transactions on Information Forensics and Security*, vol. 10, no. 5, pp. 985–998, 2015.

[6] 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*, pp. 1–18, Springer, 2017.

[7] X. Boyen, “Reusable cryptographic fuzzy extractors,” in *Proceedings of the 11th ACM Conference on Computer and Communications Security*, CCS ’04, (New York, NY, USA), pp. 82–91, ACM, 2004.

[8] 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*, pp. 68–77, July 2011.

[9] M. Blanton and M. Aliasgari, “Analysis of reusability of secure sketches and fuzzy extractors,” *IEEE Transactions on Information Forensics and Security*, vol. 8, pp. 1433–1445, Sept 2013.

[10] V. Kolesnikov, N. Matania, B. Pinkas, M. Rosulek, and N. Trieu, “Practical multi-party private set intersection from symmetric-key techniques,” in *Proceedings of the 2017 ACM SIGSAC Conference on Computer and Communications Security*, pp. 1257–1272, 2017.

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

[12] 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*, pp. 800–808, Springer, 2007.

[13] A. Kumar and A. Kumar, “Development of a new cryptographic construct using palmprint-based fuzzy vault,” *EURASIP Journal on Advances in Signal Processing*, vol. 2009, pp. 1–11, 2009.

[14] Y. Wang and K. N. Plataniotis, “Fuzzy vault for face based cryptographic key generation,” in *2007 Biometrics Symposium*, pp. 1–6, IEEE, 2007.

[15] https://fidoalliance.org/ (accessed 2020-01-10).

[16] A. Juels and M. Sudan, “A fuzzy vault scheme,” *Des. Codes Cryptography*, vol. 38, pp. 237–257, Feb. 2006.

[17] Y. Dodis, R. Ostrovsky, L. Reyzin, and A. Smith, “Fuzzy extractors: How to generate strong keys from biometrics and other noisy data,” *SIAM journal on computing*, vol. 38, no. 1, pp. 97–139, 2008.

[18] M. J. Freedman, K. Nissim, and B. Pinkas, “Efficient private matching and set intersection,” in *International conference on the theory and applications of cryptographic techniques*, pp. 1–19, Springer, 2004.

[19] C. Hazay and Y. Lindell, “Efficient protocols for set intersection and pattern matching with security against malicious and covert adversaries,” in *Theory of Cryptography Conference*, pp. 155–175, Springer, 2008.

[20] S. Jarecki and X. Liu, “Efficient oblivious pseudorandom function with applications to adaptive ot and secure computation of set intersection,” in *Theory of Cryptography Conference*, pp. 577–594, Springer, 2009.

[21] E. De Cristofaro, J. Kim, and G. Tsudik, “Linear-complexity private set intersection protocols secure in malicious model,” in *International Conference on the Theory and Application of Cryptology and Information Security*, pp. 213–231, Springer, 2010.

[22] E. De Cristofaro, P. Gasti, and G. Tsudik, “Fast and private computation of cardinality of set intersection and union,” in *International Conference on Cryptology and Network Security*, pp. 218–231, Springer, 2012.

[23] Y. Lindell, “How to simulate it–a tutorial on the simulation proof technique,” *Tutorials on the Foundations of Cryptography*, pp. 277–346, 2017.

[24] K. Ko, “User’s guide to nist biometric image software (nbis),” *NIST Interagency/Internal Report (NISTIR)-7392*, 2007.

[25] I. D. O. Nunes, K. Eldefrawy, and T. Lepoint, “Snuse: A secure computation approach for large-scale user re-enrollment in biometric authentication systems,” *Future Generation Computer Systems*, 2019.

[26] 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.

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

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

[29] P. Rindal and P. Schoppmann, “VOLE-PSI: fast OPRF and circuit-psi from vector-ole,” *Eurocrypt*, vol. 2021.

[30] M. R. N. T. Gayathri Garimella, Benny Pinkas and A. Yanai, “Oblivious key-value stores and amplification for privateset intersection,” *Crypto*, vol. 2021.

[31] S. R. Geoffroy Couteau and P. Rindal, “Silver: Silent vole and oblivious transfer from hardness of decoding structured ldpc codes,” *Crypto*, vol. 2021.

[32] K. Nandakumar, A. K. Jain, and S. Pankanti, “Fingerprint-based fuzzy vault: Implementation and performance,” *IEEE Transactions on Information Forensics and Security*, vol. 2, pp. 744–757, Dec 2007.

[33] 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*, EUROCRYPT’05, (Berlin, Heidelberg), pp. 147–163, Springer-Verlag, 2005.

[34] 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*, pp. 174–193, 2013.

[35] 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*, pp. 1–18, 2017.

[36] 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*, pp. 218–229, ACM, 1987.

[37] A. C.-C. Yao, “How to generate and exchange secrets,” in *Proceedings of the 27th Annual Symposium on Foundations of Computer Science*, SFCS ’86, (Washington, DC, USA), pp. 162–167, IEEE Computer Society, 1986.

[38] M. Ben-Or, S. Goldwasser, and A. Wigderson, “Completeness theorems for non-cryptographic fault-tolerant distributed computation,” in *Proceedings of the Twentieth Annual ACM Symposium on Theory of Computing*, STOC ’88, (New York, NY, USA), pp. 1–10, ACM, 1988.

[39] D. Chaum, C. Crépeau, and I. Damgard, “Multiparty unconditionally secure protocols,” in *Proceedings of the Twentieth Annual ACM Symposium on Theory of Computing*, STOC ’88, (New York, NY, USA), pp. 11–19, ACM, 1988.

[40] D. Beaver, *Foundations of Secure Interactive Computing*, pp. 377–391. Berlin, Heidelberg: Springer Berlin Heidelberg, 1992.

[41] G. Asharov, Y. Lindell, and T. Rabin, *Perfectly-Secure Multiplication for Any $t < n/3$*, pp. 240–258. Berlin, Heidelberg: Springer Berlin Heidelberg, 2011.

[42] Z. Beerliová-Trubíniová and M. Hirt, *Perfectly-Secure MPC with Linear Communication Complexity*, pp. 213–230. Berlin, Heidelberg: Springer Berlin Heidelberg, 2008.

[43] 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*, STOC ’96, (New York, NY, USA), pp. 639–648, ACM, 1996.

[44] 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*, STOC ’02, (New York, NY, USA), pp. 494–503, ACM, 2002.

[45] I. Damgård, V. Pastro, N. Smart, and S. Zakarias, “Multiparty computation from somewhat homomorphic encryption,” in *Proceedings of the 32Nd Annual Cryptology Conference on Advances in Cryptology — CRYPTO 2012 - Volume 7417*, (New York, NY, USA), pp. 643–662, Springer-Verlag New York, Inc., 2012.

[46] 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*, pp. 1–18. Berlin, Heidelberg: Springer Berlin Heidelberg, 2013.

[47] D. W. Archer, D. Bogdanov, B. Pinkas, and P. Pullonen, “Maturity and performance of programmable secure computation,” *IEEE Security & Privacy*, vol. 14, no. 5, pp. 48–56, 2016.

[48] Z. Erkin, M. Franz, J. Guajardo, S. Katzenbeisser, I. Lagendijk, and T. Toft, “Privacy-preserving face recognition,” in *International symposium on privacy enhancing technologies symposium*, pp. 235–253, Springer, 2009.

[49] A.-R. Sadeghi, T. Schneider, and I. Wehrenberg, “Efficient privacy-preserving face recognition,” in *International Conference on Information Security and Cryptology*, pp. 229–244, Springer, 2009.

[50] M. Osadchy, B. Pinkas, A. Jarrous, and B. Moskovich, “Scifi-a system for secure face identification,” in *2010 IEEE Symposium on Security and Privacy*, pp. 239–254, IEEE, 2010.

[51] M. Bellare, D. Pointcheval, and P. Rogaway, “Authenticated key exchange secure against dictionary attacks,” in *International conference on the theory and applications of cryptographic techniques*, pp. 139–155, Springer, 2000.

[52] S. M. Bellovin and M. Merritt, “Encrypted key exchange: Password-based protocols secure against dictionary attacks,” 1992.

[53] V. Boyko, P. MacKenzie, and S. Patel, “Provably secure password-authenticated key exchange using diffie-hellman,” in *International Conference on the Theory and Applications of Cryptographic Techniques*, pp. 156–171, Springer, 2000.

[54] R. Canetti, S. Halevi, J. Katz, Y. Lindell, and P. MacKenzie, “Universally composable password-based key exchange,” in *Annual International Conference on the Theory and Applications of Cryptographic Techniques*, pp. 404–421, Springer, 2005.

[55] R. Gennaro, “Faster and shorter password-authenticated key exchange,” in *Theory of Cryptography Conference*, pp. 589–606, Springer, 2008.

[56] P.-A. Dupont et al., “Fuzzy password-authenticated key exchange,” in *EUROCRYPT*, pp. 393–424, Springer, 2018.