

Sample-then-Correct: Practical fuzzy extractors for biometrics beyond the iris

Noah Arce-Caliskan, Benjamin Fuller, Gadalia Montoya Weinberg O’Bryan, Maryam Rezapour,
Amey Shukla

University of Connecticut and Dapple Security
nac@uconn.edu, benjamin.fuller@uconn.edu, gadalia.obryan@dapplesecurity.com,
maryam.rezapour@uconn.edu, amey.shukla@uconn.edu

September 8, 2026

Abstract

Fuzzy extractors derive stable cryptographic keys from noisy biometric measurements (Dodis et al., EURO-CRYPT 2004). Despite nearly three decades of research, existing fuzzy extractor constructions do not provide strong security for the biometric modalities deployed on modern devices, particularly fingerprints and facial recognition. Recent attacks highlight the gap between known analysis of theoretical constructions and the security levels required for practical deployment (Zhu and Wang, PoPETS 2025).

We close this gap. Compared with sample-then-lock baselines (Shukla et al., CCS 2025), our construction improves security for the fingerprint and face biometrics from 44 to 72 bits and from 59 to 84 bits, respectively. For iris biometric, we improve security from 105 to 111 bits.

Our construction, sample-then-correct, combines two existing approaches: digital lockers and secure sketches based on error-correcting codes. We embed secure sketches within a collection of digital lockers that an honest party can open with good probability. The construction naturally achieves the benefits of both approaches. We prove security in an extension of the generic group model that measures natural attacks the adversary can conduct.

These gains are achieved while maintaining authentication accuracy comparable to prior work, with true accept rates near 90% when multiple readings are available and approximately 60% from a single reading. Our implementation authenticates in under one second on a single CPU thread.

1 Introduction

Biometrics are ubiquitous in authentication systems, yet they are typically not secured by cryptography. Modern devices routinely authenticate users with face, fingerprint, or iris recognition, but the biometric templates themselves are processed, stored, and matched inside trusted hardware components rather than being used to derive cryptographic secrets directly. The use of trusted hardware is needed to prevent devastating privacy attacks that invert the template to an individual’s biometric [GRGB⁺12, FJR15, AF20, AMF22, TKAK23, WGCJ22, LNWS23, ASW⁺24]. The fundamental challenges of biometric cryptography are that biometric measurements are noisy, highly structured, and irrevocable. Unlike passwords, biometrics cannot simply be changed after compromise.

Fuzzy extractors were introduced twenty-five years ago as a cryptographic solution to this problem [BBR88, DRS04, DORS08, ŠTO05, HAD06, DKRS06, FMR13, CFP⁺16, ACEK17, ABC⁺18, WLH18, WL18, DFR21, ACF⁺22, CHR24, ACD⁺24, SDF⁺25]. The goal is deceptively simple: derive a strong cryptographic key from a biometric measurement while publishing only helper data that reveals essentially nothing about the underlying biometric. At authentication time, a noisy measurement of the same biometric should reproduce the same key.

Despite extensive research, achieving this goal in practice remains difficult. A useful fuzzy extractor must simultaneously satisfy three competing requirements:

1. **Correctness:** noisy measurements of the same biometric must reproduce the same key with good probability.

2. **Security:** the derived key must remain private, even when the *helper* data used to reproduce the key is public.
3. **Practicality:** the construction must achieve both properties on real biometric distributions.

Real biometric feature vectors exhibit substantial dependence among features, nonuniform distributions, and modality-specific structure [Dau04,ACD⁺24]. Furthermore, practical systems require high true accept rates (TARs), of at least 90%, which necessitates tolerating significant measurement noise. Increasing error tolerance generally reduces security.

No existing fuzzy extractor simultaneously provides strong security estimates and high authentication accuracy across the major biometric modalities. Current deployments, therefore, continue to rely on trusted hardware enclaves for biometric protection rather than deriving cryptographic keys directly from biometrics. Billions of users' biometrics are stored inside trusted enclaves produced by a small number of manufacturers. Each side-channel attack [LSG⁺18,KHF⁺20] puts users' long-term privacy at risk. Trusted hardware also makes security externally enforced: the system is secure only if the hardware executes the biometric check faithfully and refuses to release secrets otherwise. In contrast, a fuzzy extractor aims for a self-enforcing guarantee. Without a sufficiently close biometric measurement, there is no computation that reproduces the same cryptographic key except by breaking the underlying cryptographic assumptions or guessing the biometric.

Fuzzy extractors consist of two algorithms, **Gen** and **Rep** [DRS04,DORS08]. The generation algorithm takes a biometric feature vector w and outputs a cryptographic key, **key**, together with public helper data **pub**. The reproduction algorithm takes a noisy measurement w' and the helper data **pub**, and reproduces the same key whenever w and w' are sufficiently close according to some metric **dis**. In this work we focus on binary strings with the Hamming distance. Formally,

$$(\text{key}, \text{pub}) \leftarrow \text{Gen}(w), \text{key} \leftarrow \text{Rep}(w', \text{pub}).$$

Over the last two decades, research has converged on two broad approaches: coding-based constructions and hash-based constructions. Both have been applied to biometrics [HAD06,SDF⁺25]. However, both have fundamental limitations that prevent them from providing secure fuzzy extractors for all practical biometrics.

Coding-Based Constructions Coding-based constructions derive from the secure sketch paradigm of Dodis et al. [DRS04,DORS08]. Their appeal is that they naturally tolerate biometric noise through error-correcting codes, writing down the *syndrome* of the input w . However, their security analysis depends critically on assumptions about the biometric distribution. Stronger guarantees are known only under strong statistical assumptions such as that features are i.i.d., which is not true of real biometric feature extractors [Dau04,ACD⁺24]. Without such assumptions, the best general statement is that publishing a syndrome reduces entropy by the syndrome length [DORS08,FRS16,DGV⁺16], which can leave too little remaining entropy for practical security.

Recent attacks have shown how to invert coding style fuzzy extractors [ZW25]. These attacks highlight the importance of providing proofs of security based on the true statistical properties demonstrated by biometrics.

Hash-Based Constructions Hash-based constructions [CFP⁺16,CFP⁺21,BMT25,SDF⁺25] create many digital lockers, each keyed by a cryptographic hash of a small subset of the biometric bits. An honest user opens a locker when the corresponding subset is unchanged across measurements, while an adversary must guess the subset value used as the locker key. These constructions' security is proportional to the entropy of the inputs to the cryptographic hash. This allows one to provide concrete security estimates on real biometric distributions, even when strong dependencies are present. As we discuss in the next subsection, for all constructions, accurate entropy estimation is an unavoidable assumption in the area.

The challenge with this hash-based construction is that security and correctness are tightly coupled. To achieve high authentication accuracy, the sampled subsets must be sufficiently small for one of the μ sampled hashes to match exactly between w and w' . Yet small subsets contain less entropy. Consequently, existing constructions face a difficult tradeoff between true accept rate and security. The recent **ClearEyes** system achieves impressive iris results [SDF⁺25] obtaining > 100 bits of security at 90% TAR. However, their results do not generalize to the face and fingerprint biometrics.

	STL	STL-M	STC	STC-M
Stored lockers	200,000	600,000	110,000	18,334
Enroll readings	1	3	1	3
Auth readings	1	2	1	2
Storage (MB)	7	19	65	11
Security metric	Min subset entropy		Min of five estimates	
Enroll hashes	200,000	600,000	110,000	18,334
Subset creation (s)	7	22	29	5
Enroll time (s)	0.2	0.5	1.2	1.4
Auth hashes	200,000	1,200,000	110,000	110,004
Auth time (s)	0.16	0.98	1.0	1.0

Table 1: System configuration and computational costs. STL is short for sample-then-lock and STC is short for sample-then-correct. The suffix “-M” denotes enrollment and authentication using multiple readings. All reported times are for all lockers.

Closing the Gap The history of fuzzy extractors is a search for a construction that is flexible in error correction and whose security can be shown based on the statistical properties demonstrated by biometrics. Existing constructions typically sacrifice one objective to improve the other. Coding-based approaches provide flexible error tolerance but rely on coarse security estimates; hash-based approaches provide sharp, concrete security analysis but often require accepting lower authentication performance or lower security. Nearly thirty years of research have not yielded a construction that exhibits the benefits of both approaches without the tradeoffs.

This work presents and analyzes *sample-then-correct*, a new fuzzy extractor construction designed to bridge this gap. The construction combines sampling, cryptographic hashing, and error-correction in a way that achieves the strength of both approaches. At worst, it achieves the minimum of the security of the hash construction and the coding construction. We provide a formal model of security that extends the generic group model [Sho97] to allow the attacker to invert the coding-theoretic fuzzy extractor and run the Rep algorithm. We prove substantial gains over the security of each individual construction.

Achieving practical security with sample-then-correct requires two technical contributions: assessing the security of biometrics from demonstrable properties, and in cryptographic modeling. The next two subsections focus on these contributions. As we discuss, the demand for concrete security in practice requires more delicate analysis to avoid losses present in previous fuzzy extractor analysis [CFP⁺16, CFP⁺21].

1.1 Summary of Experimental Results

We assess security of the three most prominent biometrics: face, fingerprint, and iris. To summarize, at $> 90\%$ TAR, we improve security for the fingerprint from 44 to 72 bits, face from 59 to 84 bits, and iris from 105 to 111 bits.

Our primary point of comparison is the recent work of Shukla et al. [SDF⁺25] who focus on the iris and achieve 105 bits of security at a 92% true accept rate (TAR). They present two versions, one with a single enrollment and authentication reading with a 64% TAR and a version with 3 enrollments and 2 authentication readings that achieves 92% TAR.

Entropy estimation is inherently heuristic [VV10, VV11] if one does not have an exponential number of samples from the underlying distribution. We rely on standard heuristic techniques in the area of biometrics [Dau04]. We also ran NIST pseudorandomness tests [BRS⁺08]. In all our testing the biometric entropy test yielded a lower estimate than the NIST pseudorandomness suite. As we discuss in the next section, our analysis relies on five statistical properties of the biometric. We transform all of our five relevant security metrics into a task of estimating entropy of a distribution. See more discussion in Section 4. An important input into our security estimate is the hardness of inverting a coding based secure sketch. The state of the art attack takes 1.5 core years [ZW25] or 2^{55} ns.¹ We

¹There are technical details that differ about IronMask [KJK⁺21], the secure sketch attacked by Zhu et al. [ZW25], and the schemes discussed in this work. The two most important are that 1) IronMask works on real values of norm 1 (the hypersphere) while we

	STL	STL-M	STC	STC-M
Face				
Subset size	85	85	70	70
Security estimate	59	59	84	84
TAR	.58	.91	.62	.93
Fingerprint				
Subset size	80	80	65	65
Security estimate	44	44	72	72
TAR	.67	.96	.62	.92
Iris				
Subset size	115	115	85	85
Security estimate	105	105	111	111
TAR	0.64	0.92	0.73	.91

Table 2: Security and authentication performance. STL is short for sample-then-lock and STC is short for sample-then-correct. Mean error rates for the biometrics are Face = 0.13, Fingerprint = 0.13, and Iris = 0.08. Iris results for sample-then-lock are from **ClearEyes** [SDF⁺25].

treat the attack as currently taking 2^{55} operations. To estimate security conservatively, we assume that the same attack will improve by 2^{20} operations, taking 2^{35} operations.

We compare sample-then-correct to the prior state of the art using the hash-based fuzzy extractor, called sample-then-lock [CFP⁺16, CFP⁺21] in Tables 1 and 2. The recent implementation of sample-then-lock for the iris, called **ClearEyes** [SDF⁺25], focused on the iris biometric. We additionally compare against sample-then-lock for the face and fingerprint modalities. Shukla et al. heavily optimized their feature extractor for the iris. We use off-the-shelf feature extractors for the fingerprint and face and Shukla et al.’s feature extractor for the iris. All feature extractors produce binary features, and Hamming distance is the relevant distance.

The number of subsets, β , is chosen so that authentication can complete in at most 1s. Once this value is chosen, the size of subsets, μ , is chosen to achieve a desired TAR: $> 60\%$ for the single enrollment and $> 90\%$ for the multiple enrollment case. The main security versus TAR tradeoff is presented in Figure 1 with the orange line representing sample-then-correct and the green line representing sample-then-lock across different subset sizes.

Table 1 presents a summary of the best parameters for sample-then-lock and sample-then-correct. Both systems have a “-M” variant. These variants represent configurations with multiple readings: 3 enrollments and 2 images used per authentication attempt, compared to single readings in the base versions.

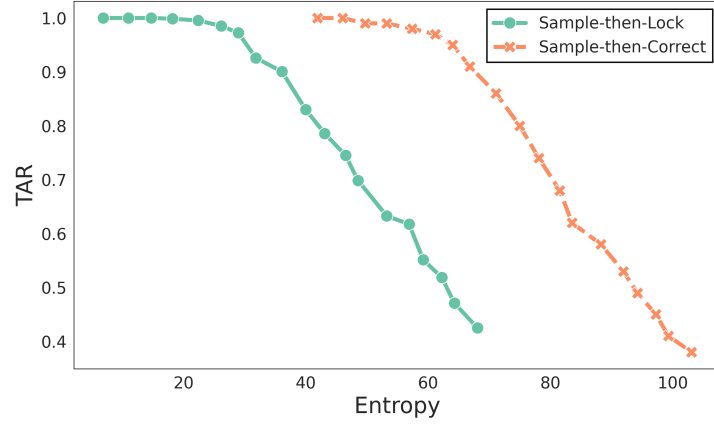
For iris authentication, sample-then-correct achieves 111 bits of security with a TAR of 73%, improving upon **ClearEyes**’s 105 bits at 64% TAR. The multi-reading variant (Sample-then-correct-M) further achieves 91% TAR (in comparison to 92% for sample-then-lock) while maintaining the same security level of 111 bits. However, our improvements for the iris are evident at high TAR for the single enrollment setting where at a 90% TAR we boost security by nearly 40 bits (from roughly 45 bits to 85 bits).

For face and fingerprint biometrics, sample-then-correct demonstrates large improvements, shown in both Figure 1 and Table 2. Sample-then-correct provides higher security estimates (84 and 72 bits respectively) in comparison to sample-then-lock (59 and 44 bits), with multi-reading configuration, achieving 0.93 and 0.92 TAR respectively. This represents an improvement of 25 and 28 bits respectively.

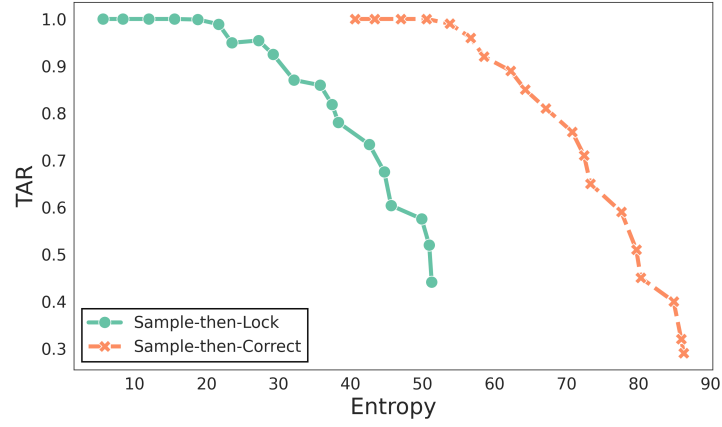
Across TAR in Figure 1, sample-then-correct usually improves security by at least 20 bits in comparison to sample-then-lock. Similarly, for the same security level sample-then-correct often additively improves TAR by 40 – 50%. These gains in TAR are not to be ignored, they elevate fuzzy extractors to being usable.

Sample-then-correct is the first time a fuzzy extractor construction has achieved the combination of 90% TAR and security of 70 bits for all three major biometrics. This combination makes fuzzy extractors practical for deployed biometrics.

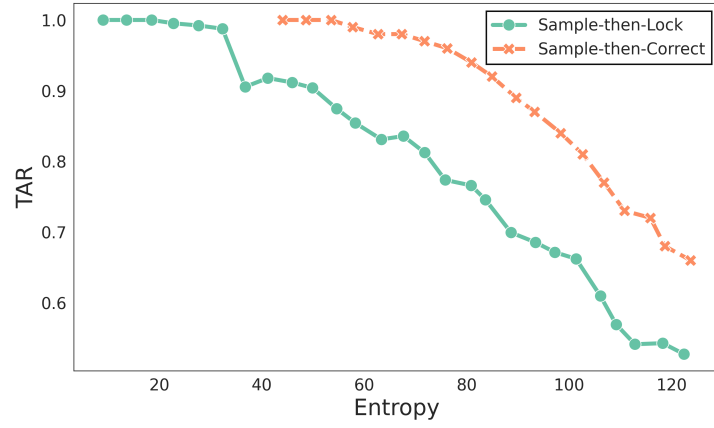
consider dimensions as bits and 2) IronMask uses a hidden matrix which isn’t allowed in the fuzzy extractor definition. Applying the attack to standard fuzzy extractors is not immediate.



(a) Security vs. TAR for Face.



(b) Security vs. TAR for Fingerprints.



(c) Security vs. TAR for Iris.

Figure 1: Trade-off between security and correctness for different biometrics using the Sample-then-Correct construction.

1.2 Technical Overview

In this section, we provide a quick overview of sample-then-correct and our security analysis. Let $w \in \{0, 1\}^n$ be the enrollment biometric and let w' be a noisy reading used for authentication.

Coding-based secure sketches. Coding-based constructions rely on a linear error-correcting code $\mathbf{A} \in \{0, 1\}^{n \times k}$ that corrects up to t errors. Let $\mathbf{H} \in \{0, 1\}^{(n-k) \times n}$ be the null space of $\mathbf{A}$. A syndrome construction publishes $\text{pub} = \mathbf{H}w$. Since $\mathbf{A}$ is an error-correcting code, for every error vector e of weight at most t , the syndrome $\mathbf{H}e$ uniquely identifies e , and there is an efficient algorithm Decode such that $\text{Decode}(\mathbf{H}e) = e$. These two properties enable reproduction as follows:

1. Compute $\mathbf{H}w' + \text{pub} = \mathbf{H}(w + w')$.
2. If $\text{dis}(w, w') \leq t$, then $w + w'$ has weight at most t , so $\text{Decode}(\mathbf{H}(w + w'))$ recovers the error pattern $w + w'$.
3. Use w' and $w + w'$ to recover w , and derive key by hashing w .²

Here we focus on the *syndrome* construction; in the body we also consider the *code-offset* construction. There are many variants of how the linear code is utilized, but they are equivalent from the security versus correctness tradeoff [DGV⁺16, Appendix A].

Sample-then-lock. The hash-based approach uses digital lockers instead of publishing a syndrome. Let Hash denote a random oracle.³ For a biometric $w \in \{0, 1\}^n$, sample-then-lock chooses β random subsets $\mathcal{I}_i \subseteq \{0, \dots, n-1\}$ of size μ and applies a salted hash to each sampled subset $w_{\mathcal{I}_i}$. Concretely, it samples a uniform key $\text{key} \leftarrow \{0, 1\}^\ell$ and stores, for each i , $c_i = \text{Hash}(i || w_{\mathcal{I}_i}) \oplus (0^\ell || \text{key})$, together with the subset $\mathcal{I}_i$. Given w' , the honest party tries each subset. If $w'_{\mathcal{I}_i} = w_{\mathcal{I}_i}$ for some i , then the corresponding locker opens and reveals key . The security of the construction is therefore controlled by the entropy of the sampled subset values, while correctness is controlled by the probability that at least one sampled subset is unchanged.

Sample-then-correct. The goal of sample-then-correct is to retain concrete security even in cases of dependence or low entropy of the biometric distribution, while also making the adversary do more than guess one correct subset to immediately recover the key. The sampled subset still plays the same role as in sample-then-lock: it is the key that opens a locker. The payload inside the locker, however, is not the final key. It is a secure sketch of the complementary bits, which lets an honest party correct the rest of the biometric.

For a subset $\mathcal{I}_i$, write $\overline{\mathcal{I}_i} := \{0, \dots, n-1\} \setminus \mathcal{I}_i$ for its complement. Define Gen as follows:

1. Sample β random subsets $\mathcal{I}_i$ of size μ .
2. For each i , publish $\text{pub}_i = \text{Hash}(i || w_{\mathcal{I}_i}) \oplus \mathbf{H}w_{\overline{\mathcal{I}_i}}$.
3. Publish $h^* := \text{Hash}(01 || w)$.
4. The final public value is $h^*, (\text{pub}_i, \mathcal{I}_i)_{i=1}^\beta$, and the final key is $\text{key} := \text{Hash}(00 || w)$.

Reproduction (**Rep**) for the honest party proceeds as follows:

1. Input w' .
2. For $i = 1$ to β , compute

$$e_{\text{cand}_i} = \text{Decode}\left(\text{Hash}(i || w'_{\mathcal{I}_i}) \oplus \text{pub}_i \oplus \mathbf{H}w'_{\overline{\mathcal{I}_i}}\right),$$

$$w_{\text{cand}_i} = w' \oplus \text{Expand}(e_{\text{cand}_i}).$$

²Some instantiations [FMR13, ACEK17] consider a random code $\mathbf{A}$ which makes syndrome decoding difficult for a large number of errors but provides additional security. Other instantiations [WLH18, WL18, WLG19, DFR21, BKM⁺18, BLMZ19] combine the syndrome with cryptographic tools with linear homomorphisms to provide useful properties such as reusability or robustness [Boy04].

³We focus on hash functions modeled as random oracles [BR93]. Previous constructions used the primitive of digital lockers, which are constructed from random oracles or very strong group-theoretic assumptions [CD08, BC10].

3. If $\text{Hash}(01||w_{\text{cand}_i}) = h^*$, output $\text{Hash}(00||w_{\text{cand}_i})$.

Expand takes a string indexed by the complement $\overline{\mathcal{I}_i}$ and maps it to length n by placing zeros in positions in $\mathcal{I}_i$. For example, when $n = 8$ and $\mu = 3$, $\text{Expand}(11011, \{1, 3, 4, 6, 7\}) = 01010011$.

Correctness follows from the two layers working together. If $w'_{\mathcal{I}_i} = w_{\mathcal{I}_i}$, then the hash mask opens correctly and reveals the syndrome $\mathbf{H}w_{\overline{\mathcal{I}_i}}$. If, in addition, $\text{dis}(w_{\overline{\mathcal{I}_i}}, w'_{\overline{\mathcal{I}_i}}) \leq t$, then decoding recovers the remaining error pattern, allowing the honest party to reconstruct w and check it using h^* .

Analyzing Security The formal proof works in an extension of the generic group model that we call *generic group with decoding*, or **GGMwD**. At a high level, the model hides syndrome-space elements behind random labels, called handles. The adversary can combine these handles algebraically and can ask decoding-style queries, but the only useful information it should learn comes from events that correspond to natural attacks on the construction: guessing the biometric, finding a close biometric, opening a true locker, or detecting algebraic structure to either decide which syndromes are correct or to correlate true syndromes.

We measure security using five biometric-dependent values (Theorem 1); terms that depend only on the strength of the hash function or the handle space are omitted here. The five terms are:

1. $\alpha := H_\infty(W)$, the min-entropy of the full biometric distribution. This captures direct attacks that guess w and check the public hash.
2. α_{fuzz}^t , the fuzzy min-entropy at decoding radius t . This captures attacks that find any point close enough to the enrolled biometric to pass reproduction. Fuzzy min-entropy [FRS16, FRS20] is stricter than false accept rate (FAR): FAR measures success for another biometric drawn from the population, whereas fuzzy min-entropy allows an adversary to choose any input.
3. α_{Subset} , the entropy of sampled subset values. This captures attacks that guess $w_{\mathcal{I}_i}$ for some locker. Distinguishing a correct guess from an incorrect one matters because a correct guess removes the hash mask from pub_i and exposes the real syndrome of the complementary bits; an incorrect guess leaves a random-looking value.
4. $\alpha_{\text{Code}}^\gamma$, the entropy of low-weight linear combinations of true syndromes. Once the adversary has many handles, it can look for algebraic relations that hold for the real syndromes but not for the values produced by incorrect locker guesses. For a parameter $\gamma \in \mathbb{Z}^+$, we directly bound relations involving at most γ syndromes by estimating $\alpha_{\text{Code}}^\gamma := \min_{0 < \text{wt}(\vec{a}) \leq \gamma} H_\infty(\langle \vec{a}, \text{Syn}(W_{\overline{\mathcal{I}}}) \rangle | \mathcal{I})$. Here $\text{Syn}(W_{\overline{\mathcal{I}}})$ denotes the vector of syndromes on the sampled complements.
5. $\alpha_{\text{Subset}}^\gamma$, the entropy needed to make a high-weight relation useful. For relations involving more than γ syndromes, the proof no longer estimates each relation directly. Instead, it uses the fact that such a relation can help only after the adversary has simultaneously guessed many correct subset values. The parameter γ is therefore the dividing line: below it we estimate linear-combination entropy, and above it we charge the adversary for guessing many lockers at once.

Depending on the biometric and the chosen parameters, different quantities dominate the final bound. When we measure security, we roughly consider the minimum of these values, with the last two terms also pay a penalty for the number of algebraic comparisons available to the adversary.

Core of analysis Our analysis is complex. This is due to our desire to provide concrete bounds on security. Canetti et al. [CFP⁺21]’s analysis of sample-then-lock divided the min-entropy of subsets, and thus security, by 2 to account for information gained by the adversary through queries. Shukla et al. [SDF⁺25] were able to remove this division by 2 through more complex techniques that show that the adversary’s queries can be viewed as deterministic and the adversary learns nothing from responses. This technique is natural in the random oracle model. We show that the same technique works in the **GGMwD** but requires complex analysis. We are able to show the adversary can simulate responses other than some small probability events where they learn the private value w .

n	Length of biometric strings
k	Dimension of error correcting code
t	Corrected distance
μ	Size of subsets (used as input to hash)
ℓ	Length of derived key
β	Number of subsets
δ	Error probability of fuzzy extractor
s	$n - \mu - k$ Length of syndromes
α	Min-entropy of distribution W
α_{Subset}	Minimum of min-entropy for selected subsets
$\alpha_{\text{Subset}}^\gamma$	Minimum of min-entropy of γ simultaneous subsets
$\alpha_{\text{code}}^\gamma$	Minimum of min-entropy of linear combinations of weight at most γ of syndromes for distribution W for selected subsets, conditioned on choice and subset
α_{fuzz}^t	Fuzzy min-entropy of distribution W
A	Error correcting code
H	Nullspace of code
λ	Output length of group elements handle space
q	Total queries allowed by adversary
q_{Invert}	Queries to Invert allowed by adversary

Table 3: Summary of notation

Notes on codes The overview uses a clean unique-decoding abstraction: within radius t the code corrects, and outside radius t it gives no useful information. This keeps the introductory security discussion tied to fuzzy min-entropy [FRS16, FP19, FRS20, Ful24]. The same construction could also use list decoding [GS98, GR05, GX13], since the published hash h^* selects the right candidate from a short list. The formal analysis includes the additional condition needed for security: decoding behavior should not let an adversary tell real syndromes from random-looking values produced by incorrect locker guesses.

Future Work We leave open analyzing sample-then-correct with respect to quantum adversaries. Sample-then-correct combines information-theoretic assumptions on biometric distributions with computational assumptions on hashing and syndrome decoding. The entropy-based components of our security analysis, including biometric min-entropy, fuzzy min-entropy, the entropy of sampled subsets and low-weight linear combinations, correspond to search problems. We expect Grover’s algorithm [Gro96] to apply the usual square root work reduction, noting that the parallelization of classical algorithms for checking hashes has been widely optimized on GPUs. Syndrome decoding for quantum algorithms is usually studied for random codes [Ber10] not the structured codes studied here. Of course, one could try and estimate concrete hardness for quantum algorithms as we do here for classical algorithms. Making precise statements requires extending our **GGMwD** to a quantum version, where adversaries may make coherent superposition queries to oracles [BDF⁺11, Zha21]. We leave this as future work.

Organization The rest of this work is organized as follows, Section 2 covers preliminaries, Section 3 formally introduces the construction, cryptographic model, and presents our formal security results, Section 4 evaluates the systems for the face, fingerprint, and iris including a description of how relevant security measures are computed, and Section 5 concludes.

2 Preliminaries

We use the following notation throughout this work. The *min-entropy* of a random variable $\mathbf{x}$ is $H_\infty(\mathbf{x}) = -\log(\max_x \Pr[\mathbf{x} = x])$. The *average (conditional) min-entropy* [DORS08, Section 2.4] of $\mathbf{x}$ given $\mathbf{y}$ is $H_\infty(\mathbf{x} \mid \mathbf{y}) = -\log(\mathbb{E}_{y \in \mathbf{y}} \max_x \Pr[\mathbf{x} = x \mid \mathbf{y} = y])$. We consider the Hamming metric over $\{0, 1\}$, define $\text{dis}(x, y) = |\{i \mid x_i \neq y_i\}|$. For a vector in $x \in \{0, 1\}^n$, let $\text{wt}(x) = |\{i \mid x_i \neq 0\}|$. U_n denotes the uniformly distributed random variable on

$\{0,1\}^n$. Logarithms are base 2. We denote the vector of all zero elements over length x as 0^x . For a finite set X , we write $x \leftarrow X$ to mean sampling over the uniform distribution on elements in X .

Definition 1 (Fuzzy min-entropy [FRS16, FRS20]). *Let $t \in \mathbb{N}$ and X be distributions. The t -fuzzy min-entropy of X in a metric space $(\mathcal{M}, \text{dis})$ is*

$$\alpha_{\text{fuzz}}^t = -\log \left(\max_{x'} \sum_{x \in \mathcal{M} | \text{dis}(x, x') \leq t} \Pr[X = x] \right)$$

2.1 Coding preliminaries

Definition 2 (Linear Code). *A (n, k, t) -linear error correcting code is a pair of matrices $\mathbf{A} \in \{0,1\}^{n \times k}$, $\mathbf{H} \in \{0,1\}^{n-k \times n}$ where $\mathbf{H} = \text{null}(\mathbf{A})$ is called the dual or null space of $\mathbf{A}$, $(\forall \mathbf{x} \in \{0,1\}^k, \mathbf{H}\mathbf{A}\mathbf{x} = 0^{n-k})$ and a deterministic polynomial time procedure `Decode` with the following properties:*

1. *For all $\mathbf{x}_1, \mathbf{x}_2 \in \{0,1\}^k$ such that $\mathbf{x}_1 \neq \mathbf{x}_2$ it is true that $\text{dis}(\mathbf{A}\mathbf{x}_1, \mathbf{A}\mathbf{x}_2) \geq 2t$.*
2. *For all $\mathbf{y} \in \{0,1\}^{n-k}$ such that there exists an $\mathbf{x} \in \{0,1\}^k$ such that $\text{dis}(\mathbf{y}, \mathbf{A}\mathbf{x}) \leq t$, $\text{Decode}(\mathbf{H}\mathbf{y}) \oplus \mathbf{y} = \mathbf{A}\mathbf{x}$.*

The value $\mathbf{H}\mathbf{y}$ is called the syndrome of $\mathbf{y}$. The above procedure is called syndrome decoding. For such a procedure to work for all such pairs $\mathbf{y}, \mathbf{x}$, it must be the case that for all $\mathbf{y}$ such that $\text{dis}(\mathbf{y}, 0) \leq t$, $\mathbf{H}\mathbf{y}$ is unique.

We consider a family of error correcting codes specified by a PPT algorithm `SCode`($\mathcal{I}_i$). The procedure `SCode` outputs pairs $\mathbf{A}, \mathbf{H}$ that we abbreviate as `code`. When such a family of codes is used we include `code` as input to `Decode` and define $\text{Syn}(\mathbf{y}, \text{code})$ as parsing `code` = $\mathbf{A}, \mathbf{H}$ and outputting $\mathbf{H}\mathbf{y}$.

Our SCode In this work, we focus on a simple construction of sample code where for a parameter $\mu \in \mathbb{Z}^+$, we consider the $(n - \mu, k)$ BCH code. Recall the dimensions of $\mathbf{A} \in \{0,1\}^{n-\mu \times k}$ and $\mathbf{H} \in \{0,1\}^{n-\mu-k \times n-\mu}$. We then compute `SCode`($\mathcal{I}$) as:

1. Take as input $\mathbf{H} \in \{0,1\}^{n-\mu-k \times n-\mu}$ and $\mathcal{I}$.
2. Initialize $j = 1, i = 0$ and $\mathbf{H}' = 0^{n-\mu-k \times n}$.
3. While $j + i < n$:
 - (a) If $j + i \in \mathcal{I}$, let $\mathbf{H}'_{(\cdot, j+i)} = 0^{n-\mu-k}$. Set $i = i + 1$
 - (b) Else set $\mathbf{H}'_{(\cdot, j+i)} = \mathbf{H}_{(\cdot, j)}$. Set $j = j + 1$.
4. Output $\mathbf{H}'$

The above fills in zero columns on the positions specified in $\mathcal{I}$. These μ symbols are ignored for decoding purposes. There is a natural adaptation of syndrome decoding for $\mathbf{H}'$ that first computes the errors with respect to $\mathbf{H}$ (with the 0 columns removed) and then translates these positions to the length n vector. We call this function `Decode`.

One can of course consider more complicated sampling procedures where the code is further randomized to prevent the adversary from detecting correlations. For our current biometrics, linear correlations are rarely the weakest term, we leave sampling of more complicated codes as future work.

3 Sample-then-Correct

We now provide an overview of the sample-then-correct construction. The formal algorithms are shown in Figure 2. Instead of locking the same key in a digital locker keyed by different sub, one locks a randomized syndrome. For this construction to have a meaningful advantage, the adversary must not be able to quickly determine if they have provided the right subset to the digital locker [CJKL18]. This is in contrast to digital lockers which indicate when the wrong point is provided.

<u>Gen(w):</u>	<u>Rep(w', nonce₀, h_0, nonce_{key}, $\{\text{code}_i, p_i, \mathcal{I}_i\}_{i=1}^\beta$):</u>
1. Sample nonces nonce_{key}, nonce₀. 2. For $i = 1, \dots, \beta$: (i) Choose hash key nonce_{i}. (ii) Sample $\mathcal{I}_i$ uniform subset of size μ. (iii) Set $h_i = \text{Hash}(\text{nonce}_i, w_{\mathcal{I}_i})$. (iv) Set $\text{code}_i \leftarrow \text{SCode}(\mathcal{I}_i)$. (v) Set $p_i = \text{Syn}(w, \text{code}_i) \oplus h_i$. 3. Output $\left(\begin{array}{l} \text{Key} = \text{Hash}(\text{nonce}_{\text{key}}, w), \\ \text{pub} = (\text{nonce}_0, h_0 := \text{Hash}(\text{nonce}_0, w)), \\ \text{nonce}_{\text{key}}, \{\text{code}_i, p_i, \mathcal{I}_i\}_{i=1}^\beta \end{array} \right).$	1. For $i = 1, \dots, \beta$: (i) Set $h_i = \text{Hash}(\text{nonce}_i, w'_{\mathcal{I}_i})$. (ii) Compute $\text{Syn}_i = p_i \oplus h_i$. (iii) $\text{Syn}_{w', i} = \text{Syn}(w', \text{code}_i)$. (iv) Compute $\text{Syn}_{w \oplus w', i} = \text{Syn}_i \oplus \text{Syn}_{w', i}$. (v) Compute $\text{Decode}(\text{Syn}_{w \oplus w', i}, \text{code}_i) = e$. (vi) If $\text{dis}(e, 0) > t$ or $e = \perp$ continue. (vii) Set $w = w' \oplus e$. (viii) If $\text{Hash}(\text{nonce}_0, w) = h_0$ output $\text{Hash}(\text{nonce}_{\text{key}}, w)$. 2. Output $\perp$.

Figure 2: Syndrome then lock construction.

3.1 Generic Group Model with Decoding, GGMwD

We consider an extension of the generic group model called **GGMwD** that models attacks available to the adversary. There are several previous versions of fuzzy extractor security [DORS08, FMR13, DFR21].

GGMwD The additional oracles available to the adversary are defined relative to a family of Syn whose distribution is specified by SCode and probability distribution W of biometrics. We consider a generic group [Sho97] over binary strings of length $s := n - \mu - k$ and three independent random functions, modeled as random oracles [BR93], $\text{Hash}_1^\mathcal{O}, \text{Hash}_2^\mathcal{O}, \text{Hash}_3^\mathcal{O}$. We do not provide direct access to $\text{Hash}_1^\mathcal{O}$. The core of extension to the GGM is based on 3 oracles that we review below. The oracles are presented formally in Figure 3.

1. **Guess** represents the adversary hashing a subset and adding this to a public value to unmask the blinding applied in pub_i . We merge three operations here:

- (a) Hashing the chosen subset,
- (b) Adding this to the sampled public value,
- (c) Encoding the value in the group.

The hash function used to compute subsets is only available to the adversary through the **Guess** interface.

2. **Invert** represents the adversary's ability to invert the syndrome into realistic biometrics, discovering all points $\{w | \text{Syn}(W) = \text{syn}\}$. This is modeled by allowing the adversary to invert the generic group encoding. These operations are tracked separately as this is currently believed to be an expensive operation.
3. **DecodeECC** represents the ability to quickly:
 - (a) Run the error correction **Decode** on a candidate x ,
 - (b) Add the resulting error vector e to a point y , and
 - (c) Hash the value to see if it is correct.

<u>Setup</u> (n, k, μ, λ, ℓ): 1. Define $s = n - \mu - k$. 2. Sample random 1-1 function, $g : \{0, 1\}^s \rightarrow \{0, 1\}^\lambda.$ 3. Sample random functions, $\begin{aligned} \text{Hash}_1 &: [\beta] \times \{0, 1\}^\mu \rightarrow \{0, 1\}^s, \\ \text{Hash}_2 &: \{0, 1\}^n \rightarrow \{0, 1\}^\ell, \\ \text{Hash}_3 &: \{0, 1\}^n \rightarrow \{0, 1\}^\ell. \end{aligned}$	<u>Hash₂^O</u> (x): $\text{Hash}_2(x)$. <u>Hash₃^O</u> (x): $\text{Hash}_3(x)$. <u>+</u> (x, y): 1. $x^{-1} = g^{-1}(x), y^{-1} = g^{-1}(y)$. 2. If either value does not exist, return $\perp$. 3. Return $\text{Encode}(x^{-1} + y^{-1})$.
<u>Sample</u> (w): 1. For $i = 1$ to β: (a) Sample I_i, a uniform subset of $[1, n]$ of size μ. (b) Compute $h_i = \text{Hash}_1(i, w_{I_i})$. (c) Set $\text{code}_i \leftarrow \text{SCode}(w, I_i)$. (d) Set $\text{syn}_i = \text{Syn}(\text{code}_i, w)$. (e) Set $\text{pub}_i = \text{Encode}(\text{syn}_i + h_i)$. (f) Store $(i, \text{pub}_i, \text{code}_i, I_i)$. 2. Compute $h_{\text{check}} = \text{Hash}_2(w)$. 3. Return $\text{pub} := h_{\text{check}}, \{\text{pub}_i, \text{code}_i, I_i\}_{i=1}^\beta$.	<u>Encode</u> (x): $g(x)$. <u>Guess</u> (x, i): $\text{Encode}(\text{Hash}_1((i, x)) + \text{pub}_i)$. <u>Invert</u> (x): 1. Compute $\text{syn} = g^{-1}(x)$. 2. If no such inverse exists, output $\perp$. 3. Else return syn.
	<u>DecodeECC</u> (x, choice, y): 1. Compute $\text{syn} = g^{-1}(x)$. 2. If no such inverse exists, output $\perp$. 3. Compute $\text{Decode}(\text{syn}, \text{choice}) = e$. 4. If $\text{dis}(e, 0) > t$ or $e = \perp$, return $\perp$. 5. If $\text{Hash}_2(e + y) = h_{\text{check}}$, return $e + y$. 6. Return $\perp$.

Figure 3: GGMwD model, adversary is allowed q queries to $\text{Hash}_2^O, \text{Hash}_3^O, +, \text{Encode}, \text{Guess}, \text{DecodeECC}$ and q_{Invert} queries to Invert .

The Invert oracle takes in three arguments, the candidate syndrome x to decode, the code code to use, and the offset y to shift after decoding.

We overload the notation of the above oracles to apply to vectors and use $\mathcal{O}$ to refer to the set of oracles provide to the adversary.

Definition 3. Let $\mathcal{W}$ be a family of probability distributions over the metric space $(\{0, 1\}^n, \text{dis})$. A pair of procedures $\text{Gen} : \{0, 1\}^n \rightarrow \{0, 1\}^\ell \times \{0, 1\}^*$, $\text{Rep} : \{0, 1\}^n \times \{0, 1\}^* \rightarrow \{0, 1\}^\ell$ is an $(\{0, 1\}^n, q, q_{\text{Invert}}, \ell, t, \epsilon_{\text{sec}}, \delta)$ -fuzzy extractor with parameters (β, k, λ, μ) if Gen and Rep satisfy the following properties:

- **Correctness:** If $\text{dis}(w, w') \leq t$ and $(\text{key}, \text{pub}) \leftarrow \text{Gen}(w)$, then $\Pr[\text{Rep}(w', \text{pub}) = \text{key}] \geq 1 - \delta$. We make the standard fuzzy extractor assumption that the difference between w and w' is independent of the output of Gen .
- **Security:** For any distribution $W \in \mathcal{W}$, the string key is close to random conditioned on pub for all adversaries $\mathcal{A}$ making at most q queries to some oracle $\mathcal{O} \in \{\text{Encode}, +, \text{Hash}_2^O, \text{Hash}_3^O, \text{DecodeECC}, \text{Guess}\}$ and at most q_{Invert} queries to Invert . That is, for the experiment defined in Figure 4 it is true that

$$|\Pr[\text{exp}_{FE}(n, k, \beta, \mu, \lambda, \ell, q, q_{\text{Invert}}, W) = 1]| \leq \frac{1}{2} + \epsilon_{\text{sec}}.$$

$\text{exp}_{FE}(n, k, \beta, \mu, \lambda, \ell, q, q_{\text{Invert}}, W):$

1. Sample $w \leftarrow W$.
2. Initialize $\text{Setup}(n, k, \mu, \lambda, \ell)$.
3. Sample $\text{pub} := h_{\text{check}}, \{\text{pub}_i, \text{code}_i, \mathcal{I}_i\}_{i=1}^\beta \leftarrow \text{Sample}(w)$.
4. Sample $b \leftarrow \{0, 1\}$.
5. If $b = 1$, sample $\text{key} \leftarrow \{0, 1\}^\ell$, else set $\text{key} = \text{Hash}_3(w)$.
6. Send key, pub to $\mathcal{A}$.
7. $\mathcal{A}$ queries any $\text{Encode}, +, \text{Hash}_2^\mathcal{O}, \text{Hash}_3^\mathcal{O}$, and DecodeECC up to q times and Invert at most q_{Invert} times.
8. $\mathcal{A}$ outputs b' . Output $b' == b$.

Figure 4: Experiment Definition for Fuzzy Extractor Security

3.2 Analyzing Security

Theorem 1. Let W be a distribution where $H_\infty(W) \geq \alpha$. For $1 \leq i \leq \beta$, define $\mathcal{I}_i$ as the uniform distribution over subsets of $\{0, 1\}^n$ of size μ . For a function SCode and subsets of size k let Code_i denote the distribution of: 1) Sample $I_i \leftarrow \mathcal{I}_i$, and 2) Output $\text{code}_i \leftarrow \text{SCode}(I_i)$. Let I_i and code_i represent particular outcomes of the relevant distribution. Define the following $\alpha_{\text{Subset}} := \min_{1 \leq i \leq \beta} H_\infty(W_{\mathcal{I}_i} | \mathcal{I}_i = I_i)$ and

$$\alpha_{\text{Code}}^\gamma := \min_{\substack{1 \leq i \leq \beta \\ \text{wt}(\vec{a}) \leq \gamma}} H_\infty \left(\left\langle \vec{a}, \text{Syn}(W, \text{Code}_i) \right\rangle \middle| \begin{array}{l} \text{Code}_i = \text{code}_i, \\ \mathcal{I}_i = I_i \end{array} \right)$$

$$\alpha_{\text{Subset}}^\gamma := \min_{\substack{1 \leq i \leq \beta \\ J \subseteq [1, \beta] | |J| > \gamma}} H_\infty \left(\bigwedge_{j \in J} W_{\mathcal{I}_j} \middle| \bigwedge_{i=1}^\beta \mathcal{I}_i = I_i \right)$$

The above three quantities are parameterized by I_i, code_i , however, we omit this for notational clarity. Then for all adversaries $\mathcal{A}$ making at most q queries to $\mathcal{O} \in \{\text{Encode}, +, \text{Hash}_2^\mathcal{O}, \text{Hash}_3^\mathcal{O}, \text{DecodeECC}, \text{Guess}\}$ and at most q_{Invert} queries to Invert in GGMwD :

$$\begin{aligned} \epsilon_{\text{sec}} &\leq q_{\text{Invert}} 2^{-\alpha_{\text{Subset}}^{\min}} \\ &\quad + q \left(2^{-H_{t, \infty}^{\text{fuzz}}(W)} + 2^{-H_\infty(W)} \right) \\ &\quad + (q + \beta)^2 \max \left\{ 2^{-\ell}, 2^{-(\alpha_{\text{Subset}}^{\min} + \alpha_{\text{Code}}^{\min, \gamma})}, 2^{-\alpha_{\text{Subset}}^{\min, \gamma}} \right\} \\ &\quad + q (3 \cdot 2^{-\ell} + 2^{\lambda-s}). \end{aligned}$$

Concrete measurement of security and proof techniques In Theorem 1 we focus on the actual min-entropy of the subsets and code values. Dodis et al. state a Lemma about the decrease in average min-entropy across choices [DORS08, Lemma 2.2b]. Shukla et al. [ACD⁺24, Lemma 2] then use this lemma to control the probability of the adversary guessing conditioned on none of choices being low entropy. This Lemma says that if $H_\infty(\text{Syn}(W_i, \text{Code}_i)) \geq \tilde{\alpha}_{\text{Code}}$, then with oracle access to oracles that indicate when a query is equal to $\text{Syn}(W_i, \text{code}_i)$ the chance of gaining some positive response is at most $(\beta + q)2^{-\tilde{\alpha}_{\text{Code}}/2}$. Similar analysis proceeds for subsets in

place of syndrome. We use the actual min-entropy as our measurement tool, dividing entropy by 2 is quite costly. Similarly, sample-then-lock [CFP⁺16] argues that security degrades with the square of the number of digital locker queries (this would be the cube for linear combination queries) due to information learned about incorrect accesses. Avoiding this source of quadratic dependence is critical for providing concrete security and is the reason for the proof structure discussed below.

Proof of Theorem 1. The only difference between the case when $b = 1$ and $b = 0$ is whether **key** is a random value in $\{0, 1\}^\ell$ or is $\text{Hash}_3(w)$. We bound the probability that $\mathcal{A}$, across all of their queries, queries some value x such that $\text{Hash}_3(x) = \text{key}$. For an adversary and random initialization of **Setup**, define $\text{Query}_{\text{Hash}_3} = \{x | \mathcal{A} \text{ calls } \text{Hash}_3(x)\}$. Since the adversary is unbounded, without loss of generality, we assume its queries are deterministic based on its current view. Both in the case when $b = 1$ and $b = 0$ we upper bound the probability that $\exists x \in \text{Query}_{\text{Hash}_3} \wedge \text{Hash}_3(x) = \text{key}$. Our final distinguishing probability is the sum of these two probabilities (which upper bounds the maximum of these two values).

When $b = 1$, there is no information correlated to **key**. Thus, the total probability of $\mathcal{A}$ hashing some item r^* where $\text{Hash}_3(r^*) = \text{key}$ is at most $q2^{-\ell}$. This is a union bound over the q queries, with the probability on a single query is exactly $2^{-\ell}$.

We now proceed to the case when $b = 0$. We note that the case when the $\mathcal{A}$ queries some value $x \neq w$ but $\text{Hash}_3(x) = \text{Hash}_3(w)$ is covered by the probability in case $b = 1$.

Lemma 1. *In the case that $b = 0$,*

$$\begin{aligned} & \Pr[\exists x \in \text{Query}_{\text{Hash}_3} \wedge x = w] \\ & \leq q_{\text{Invert}} 2^{-\alpha_{\text{Subset}}^{\min}} \\ & \quad + q2^{-H_{t,\infty}^{\text{fuzz}}(W)} + q2^{-H_\infty(W)} \\ & \quad + (q + \beta)^2 \max \left\{ 2^{-\ell}, 2^{-(\alpha_{\text{Subset}}^{\min} + \alpha_{\text{Code}}^{\min, \gamma})}, 2^{-\alpha_{\text{Subset}}^{\min, \gamma}} \right\} \\ & \quad + q2^{-\ell+1} + q2^{\lambda-s} \end{aligned}$$

We note that for the above to be meaningful the length of the generic group λ must be greater than $s = n - \mu - k$. Our overall strategy is as follows:

1. Show that there are only four ways of receiving information about the sampled point w .
 - (a) By computing a valid hash of a point with the same hash as w .
 - (b) By calling **Guess**, **Encode_G**, **+** and receiving a repeated value for when the value should be linearly independent of the previous values.
 - (c) Running (slow) decode syndrome on a true value syn_i , or
 - (d) Running a fast decode syndrome on a true value syn_i with a “correct” offset.
2. Show that conditioned on none of these conditions occurring, the $\mathcal{A}$ has a bounded chance of inputting w to Hash_3 .
3. A critical part of the proof is that until any of these events occur the adversary’s view is independent of the chosen point w so we can continue to ignore $\mathcal{A}$ changing their behavior based on the responses from the oracles, making $\mathcal{A}$ non-adaptive.

Proof of Lemma 1. Let hybrid $\mathcal{H}_0$ be the real execution of the protocol when $b = 0$.

Hybrid $\mathcal{H}_1$ Define hybrid $\mathcal{H}_1$ as the same as hybrid $\mathcal{H}_0$ except that $+$ is being replaced by $+'$ where if one handle has not previously been seen as an output of $\text{Encode}_{\mathbb{G}}^{\mathcal{O}}$ or then the output of $+'$ is $\perp$. We note that $\text{Encode}_{\mathbb{G}_s}$ is called inside of $+$, **Sample**, and **Guess**. Before seeing any values, the probability of any handle having an inverse in $\text{Encode}_{\mathbb{G}}$ is at exactly $2^\lambda/2^s$.

Note, the adversary sees some valid handles and can learn some handles are invalid. Knowledge of valid and invalid handles decreases the set of possible valid and invalid handles by the number of handles known to be valid/invalid. We use the inequality that for $c < a < b$, $a/b \geq (a-c)/(b-c)$. So it suffices to bound in all cases by $2^{\lambda-s}$. Taking a union bound over the q queries yields the bound of $q2^{\lambda-s}$.

Let $h_1, \dots, h_\beta$ denote the hashes computed in Step 2iii of Figure 2. Similarly, let $(i_1, g_1), \dots, (i_q, g_q)$ denote the sequence of inputs to **Guess** (noting that some of these values may be empty if the $\mathcal{A}$ doesn't query **Guess** in step i). Each value $z := x + y$ computed inside of $+'$ can be represented in the form

$$z = a_c + \sum_{i=1}^{\beta} \vec{a}_i^{\text{pub}} (h_i + \text{syn}_i) + \sum_{i=1}^q \vec{a}_i^{\text{Guess}} (\text{Hash}_1(j_i, g_j) + h_{j_i} + \text{syn}_{j_i}).$$

and the experiment and the adversary can compute the values $a_c, \vec{a}^{\text{pub}}, \vec{a}^{\text{Guess}}$. We also note that this representation is fixed, independent of query responses: for inputs x, y , the output z can be expressed as follows:

$$\begin{aligned} x &= a_{x,c} + \sum_{i=1}^{\beta} \vec{a}_{x,i}^{\text{pub}} (h_i + \text{syn}_i) \\ &\quad + \sum_{i=1}^q \vec{a}_{x,i}^{\text{Guess}} (\text{Hash}_1(j_i, g_j) + h_{j_i} + \text{syn}_{j_i}), \\ y &= a_{y,c} + \sum_{i=1}^{\beta} \vec{a}_{y,i}^{\text{pub}} (h_i + \text{syn}_i) \\ &\quad + \sum_{i=1}^q \vec{a}_{y,i}^{\text{Guess}} (\text{Hash}_1(j_i, g_j) + h_{j_i} + \text{syn}_{j_i}), \\ z &= (a_{x,c} + a_{y,c}) + \sum_{i=1}^{\beta} (\vec{a}_{x,i}^{\text{pub}} + \vec{a}_{y,i}^{\text{pub}}) (h_i + \text{syn}_i) \\ &\quad + \sum_{i=1}^q (\vec{a}_{x,i}^{\text{Guess}} + \vec{a}_{y,i}^{\text{Guess}}) (\text{Hash}_1(j_i, g_j) + h_{j_i} + \text{syn}_{j_i}), \end{aligned}$$

We now identify values $x \in \{0, 1\}^\ell$ by their characteristic values $a_c, \vec{a}^{\text{pub}}, \vec{a}^{\text{Guess}}$ denoted as $\vec{a}$. The initial values pub_i correspond to a $\vec{a} = \vec{a}^{\text{pub}} \parallel \vec{a}^{\text{Guess}}$, which is a unit vector for $\vec{a}^{\text{pub}}$ with 1 at the i th position and $\vec{a}^{\text{Guess}} = 0^q$. Any value x encoded by the adversary is represented as $x_{\mathcal{A}} = x$. We note that if addition is called at step i , it must be the case that $\vec{a}_{i+j}^{\text{Guess}} = 0$ for $j \geq 0$. That is, that all coefficients for future guesses are 0. However, our proof does not rely on this property.

For a query x we say that $x \in \text{AdvHand}$ if $\vec{a}^{\text{Guess}} = 0^q, \vec{a}^{\text{pub}} = 0^\beta$. Such items are the result of **Encode** or calling $+'$ on two items in **AdvHand**. We now consider an $\mathcal{A}_1$ that runs $\mathcal{A}$ with the following modification:

1. For each item in $x \in \text{AdvHand}$ store $x, g(x)$ we call this table **AdvHand**.
2. On input query $q_i = \text{Invert}(g(\text{syn}))$ if $\text{syn} \in \text{AdvHand}$, locally compute syn and return this value to $\mathcal{A}$.
3. On input query $q_i = \text{DecodeECC}(g(\text{syn}), c, \mathcal{I}, y)$ if $\text{syn} \in \text{AdvHand}$,

- (a) Locally compute $e' + y$ as described in **DecodeECC**,
- (b) Query $h^* \leftarrow \text{Hash}_2(e' + y)$.
- (c) If $h^* = h_{\text{check}}$, return $e' + y$ to adversary.
- (d) Else return $\perp$.

Clearly $\mathcal{A}_1$ in $\mathcal{H}_1$ has the same probability of $w \in \text{Query}_{\text{Hash}_3}$ and it makes no more queries than $\mathcal{A}$. It also makes no more queries to **Invert**.

Hybrid $\mathcal{H}_2$ In $\mathcal{H}_2$, we:

- 1. Create a table **AdvHand** that values called to **Encode** and their sums.
- 2. **Invert** so that on input $q_i = \text{Invert}(g(\text{syn}))$ it first checks if $\text{syn} \in \text{AdvHand}$ and if so returns $\perp$, and
- 3. **DecodeECC** so that on input $q_i = \text{DecodeECC}(g(\text{syn}), c, \mathcal{I}, y)$ it first checks if $\text{syn} \in \text{AdvHand}$ and if so returns $\perp$.

Since $\mathcal{A}_1$ never queries an item in **AdvHand** in $\mathcal{H}_1$, $\mathcal{A}_1$ has the same probability of $w \in \text{Query}_{\text{Hash}_3}$ in $\mathcal{H}_2$.

Hybrid $\mathcal{H}_3$ We now explicitly let the adversary win if:

- 1. They make a query $\vec{a}$ that has not been made before but results in the same group element.
- 2. They make a query to **Invert** on some value syn_i that was generated in **Sample**.
- 3. They make a query to **DecodeECC** with some value within distance t of w .
- 4. They make a query to **Hash**₂ that matches h_2 .

The rest of the proof proceeds by showing that:

- 1. The probability of each of these events is low,
- 2. If none of these events occur, the adversary's view of received handles is independent of the sampled point w .

Define hybrid $\mathcal{H}_3$ is the same as $\mathcal{H}_2$ except that we make five modifications at the same time and consider a new adversary $\mathcal{A}_2$:

- 1. Inside of **Sample** in Step 1f, we augment the storage of item i to be $(i, \text{pub}_i, \text{code}_i, \text{syn}_i, \text{Hash}_i, I_i)$ and add an additional step which stores the input w .
- 2. On query q_i , **Invert** $(g(\text{syn}^*))$ outputs a special ("Win", w) when provided with an input

$$\text{syn}^* \in \{\text{syn}_i\}_{i=1}^\beta.$$

- 3. On query q_i , **DecodeECC** $(x, \text{choice}, \mathcal{I}, y)$ outputs a special ("Win", w) when provided with either:

- (a) An input $\mathcal{I}, y$ such that $\text{dis}(y, w) \leq t$, or
- (b) The value $e' + y$ computed inside of **DecodeECC** has $\text{Hash}_2(e' + y) = h_{\text{check}}$.

If neither of these conditions occur then **DecodeECC** outputs $\perp$.

- 4. The experiment internally manages a table **HandleLinear** that contains pairs of points $z, \vec{a}_z$. This table is initialized to be $(\text{pub}_i, \vec{a}_{\text{pub}_i})$ where $\vec{a}_{\text{pub}_i}$ is a unit vector with a 1 at the $i + 1$ st position. On a query
 - (a) q_i that is $+_\ell(g(x), g(y))$ the experiment internally computes $z = x + y$ and $\vec{a}_z$.
 - (b) q_i that is **Guess** (x, j) the experiment internally computes $z := \text{Hash}_1(j, x) + \text{pub}_j$ and $\vec{a}_z$ which is a unit vector with a 1 at the i th position of λ^{Guess} .

(c) q_i that is $g(z)$ compute $\vec{a}_z$ which is $\vec{a}_z = (z, 0, 0, \dots, 0)$.

Before returning the value, the experiment checks if there $\exists(z', \vec{a}_{z'}) \in \mathbf{HandleLinear}$ such that $z' = z$ but $\vec{a}_{z'} \neq \vec{a}_z$. If so, it outputs (“Win”, w). Otherwise, it inserts (z, λ_z) into $\mathbf{HandleLinear}$ and operates as before. We note that for all $1 \leq i \leq q$ here we treat $\lambda_{i+j}^{\text{Guess}} = 0$ for $0 < j \leq q - (i + 1)$ relying on the fact that $+'_\ell$ only works for previously seen handles.

5. On query q_i that is $\mathbf{Hash}_2(x)$ checks if $\mathbf{Hash}_2(x) = h_{\text{check}}$ if so it outputs (“Win”, w).

We consider $\mathcal{A}_2$ that runs $\mathcal{A}_1$ except when receiving (“Win”, w) from some oracle it immediately queries $\mathbf{Hash}_3(w)$ and outputs whether this value is equal to its first input. Clearly, the probability that $\mathcal{A}_2$ when interacting with $\mathcal{H}_3$ queries w to $\mathbf{Hash}_3$ is at least as high the probability that $\mathcal{A}_1$ when interacting with $\mathcal{H}_2$ queries w to $\mathbf{Hash}_3$.

Hybrid $\mathcal{H}_4$ Define $\mathcal{H}_4$ as the same as $\mathcal{H}_3$ the following change:

1. On initialization it randomly samples $p_1, \dots, p_\beta \leftarrow \{0, 1\}^\ell$.
2. On a query q_i to guess $\mathbf{Guess}(x, i)$ return h_i^q defined as follows:
 - (a) If $\mathbf{Hash}_1(x, i)$ is equal to h_j it defines $h_i^q = \text{syn}_j$.
 - (b) If $\mathbf{Hash}_1(x, i)$ has been previously defined in query j return the defined h_j^q .
 - (c) Else define $h_i^q \leftarrow \{0, 1\}^\ell$. Record $(\mathbf{Hash}_1(x, i), h_i^q)$.
3. On a query q_i to $\mathbf{Invert}(g(\text{syn}), c)$,
 - (a) If $\text{syn} \in \{\text{syn}_i\}_{i=1}^\beta$ it outputs the special symbol (“Win”, w) as in $\mathcal{H}_3$.
 - (b) Otherwise, it computes $\vec{a}_{\text{syn}}$ and responds with

$$\text{res}_{\text{syn}} := \vec{a}_{\text{syn}, c} + \sum_{i=1}^{\beta} \vec{a}_{\text{syn}, i}^{\text{pub}} p_i + \sum_{i=1}^q \vec{a}_i^{\text{Guess}} h_i^q.$$

The above distribution of responses has the same distribution as in $\mathcal{H}_3$. We also note that it is independent of the value w as long as for queried points in $\mathbf{Invert}$ it is true that $\mathbf{Hash}_1(x, i) \neq h_i$. To see this, considering setting $\mathbf{Hash}_1(x, i) = r - \text{pub}_i$, this induces the same distribution of responses on $\mathbf{Hash}_1$ when $\mathbf{Hash}_1(x, i)$ is defined randomly.

Hybrid $\mathcal{H}_5$ Define $\mathcal{H}_5$ as the same as $\mathcal{H}_4$ with the following change:

1. Each value $p_1, \dots, p_\beta, h_1^q, \dots, h_q^q$ is initialized as an indeterminate except when h_j^q is defined through a hash match in $\mathcal{H}_4$. Initialize table $\mathbf{ConsistentSol}$ as containing $((1, 0, \dots, 0), 1)$.
2. For a query $\mathbf{Invert}(g(\text{syn}))$, compute $\vec{a}_{\text{syn}}$,
 - (a) If $\vec{a}_{\text{syn}}$ is linearly dependent on previous queries to $\mathbf{Invert}$, return the appropriate linear combination of previous queries.
 - (b) Else if $\vec{a}_{\text{syn}}$ is linearly independent of previous queries, randomly sample $r \leftarrow \{0, 1\}^\ell$. Define $\mathbf{A}, \vec{r}$ as the concatenation of all values in $\mathbf{ConsistentSol}$ and $(\vec{a}_{\text{syn}}, r)$. Check if

$$\mathbf{A} \begin{pmatrix} 1 \\ p_1 \\ \dots \\ p_\beta \\ h_1^q \\ \dots \\ h_q^q \end{pmatrix} = \vec{r}$$

has a solution. If it does have a solution output r and add $(\vec{a}_{\text{syn}}, r)$ to **ConsistentSol**.

If not, fix all constrained hash variables required for **ConsistentSol**'s unique solution. (Note that this may not constrain all hash outputs involved in **ConsistentSol** but hash outputs are either unconstrained or fixed.) Respond to the query as the linear combination of

$$\vec{a}_{\text{syn}} \begin{pmatrix} 1 \\ p_1 \\ \dots \\ p_\beta \\ h_1^q \\ \dots \\ h_q^q \end{pmatrix}.$$

Note for this case to be triggered it is true that all variables where $\vec{a}_{\text{syn}}$ has a nonzero coefficient are constrained otherwise $\vec{a}_{\text{syn}}$ could have been added to **ConsistentSol**.

As before, we note that the distribution of responses in $\mathcal{H}_5$ is equal in distribution to the responses in $\mathcal{H}_4$.

Hybrid $\mathcal{H}_6$ In $\mathcal{H}_6$ we make the following change as well as changing the adversary

1. The values of r are no longer added to **ConsistentSol**.
2. For a query **Invert**($g(\text{syn})$), compute $\vec{a}_{\text{syn}}$,
 - (a) If $\vec{a}_{\text{syn}}$ is linearly dependent on previous queries to **Invert** return $\perp$.
 - (b) Else if $\vec{a}_{\text{syn}}$ is linearly independent of previous queries. Define **A** as the concatenation of all values in **ConsistentSol** and $(\vec{a}_{\text{syn}})$. Check if

$$\mathbf{A} \begin{pmatrix} 1 \\ p_1 \\ \dots \\ p_\beta \\ h_1^q \\ \dots \\ h_q^q \end{pmatrix} = \vec{r}$$

has a solution for all vectors $\vec{r}$, if so output $\perp$. Otherwise, (“Win”, w).

We consider an $\mathcal{A}_3$ that has two changes:

1. If $\mathcal{A}_2$ asks to compute a query to **Invert** that is linearly dependent on previous queries $\mathcal{A}_3$ computes the response itself and sends it to $\mathcal{A}_2$.
2. Upon receiving $\perp$ from **Invert** generate r and send this to $\mathcal{A}_2$.

We note that an $\mathcal{A}_3$ wins in $\mathcal{H}_6$ as least as frequently as $\mathcal{A}_2$ it wins in $\mathcal{H}_5$.

Hybrid $\mathcal{H}_7$ We now consider two changes simultaneously. We consider in $\mathcal{H}_7$ interacting with $\mathcal{A}_4$:

1. Both $+$ ' and **Invert** taking vectors $\vec{a}$ instead of handles and **Guess** records the queried value but no longer returns handles. In addition,
 - (a) $+$ ' return $\perp$ if the oracle in $\mathcal{H}_6$ would have returned a new handle and (“Win”, w) if the oracle returns (“Win”, w).
 - (b) **Guess** calls the underlying guess from $\mathcal{H}_6$. If it returns a handle, the new **Guess** records the input and returns $\perp$. If the oracle **Guess** in $\mathcal{H}_6$ returns (“Win”, w), return (“Win”, w).

We consider a $\mathcal{A}_4$ that randomly generates the new handles on its own that are missing from $+'_\ell$ and **Guess**. It additionally translates $\mathcal{A}_3$'s queries from handles to vectors $\vec{a}$.

We now note that all responses $\mathcal{A}_4$ receives from its oracles are either $\perp$ or (“Win”, w). Recall that since $\mathcal{A}_4$ is unbounded it can be deterministic (choosing the best randomness). We also note that this $\mathcal{A}_4$ stops upon receiving “Win” from any oracle we can analyze the probability that type of “Win” is triggered for a single query and use the union bound over the number of queries to that oracle to bound total success.

Claim 1.

$$\Pr[\text{Invert Outputs Win}] \leq q_{\text{Invert}} 2^{-\alpha_{\text{Subset}}^{\min}}.$$

Proof. We note that **Invert** only outputs “Win” if the responses cannot be chosen randomly and independently of the values syn_i . This occurs if and only if the adversary makes queries where the only difference between the query to **Invert** and a linear combination of previous queries to **Invert** is the addition of nonzero coefficients on h_i^q where the corresponding hashed value was equal to h_i . The probability of this occurring is upper bounded by the case where there is only one such coefficient and the probability is $2^{-\alpha_{\text{Subset}}^{\min}}$ by assumption. $\square$

Claim 2.

$$\Pr[\text{DecodeECC Outputs Win}] \leq q \left(2^{-H_{t,\infty}^{\text{fuzz}}(W)} + 2^{-\ell} \right)$$

Proof. Since the value of e' computed inside of **DecodeECC** has weight at most t , meaning that $\text{dis}(0^\ell, e') \leq t$, for $\text{Hash}_2(e' + y) = h_{\text{check}}$ it must be the case that either

1. The value y is within distance t of w which occurs with probability at most $2^{-H_{t,\infty}^{\text{fuzz}}(W)}$, or
2. $e' + y \neq w$ but $\text{Hash}_2(e' + y) = \text{Hash}_2(w)$. On average the number of points with x such that $|\{x | \text{Hash}_2(x) = \text{Hash}_2(w)\}| = 2^{n-\ell}$, furthermore, this set is independent of w . Thus, until a point $e' + y$ has been evaluated through Hash_2 (either through calling **DecodeECC** with a $\text{syn}, \text{choice}, y$ that produces $e' + y$ or by directly calling Hash_2^O on $e' + y$, the probability of a point having the same hash is $2^{-\ell}$. A union bound across all queries yields a total probability of $q2^{-\ell}$.

$\square$

Claim 3.

$$\begin{aligned} & \Pr[+_ \ell' \vee \text{Encode}_{\mathbf{G}_s} \vee \text{Guess Outputs Win}] \\ & \leq (q + \beta)^2 \max \left\{ 2^{-\ell}, 2^{-(\alpha_{\text{Subset}}^{\min} + \alpha_{\text{Code}}^{\min, \gamma})}, 2^{-\alpha_{\text{Subset}}^{\min, \gamma}} \right\}. \end{aligned}$$

Proof. These oracles output Win if and only if they encounter two representations $\vec{a}, \vec{b}, \vec{a} \neq \vec{b}$ such that

$$\vec{a}_{1,c} + \sum_{i=1}^{\beta} \vec{a}_i^{\text{pub}} p_i + \sum_{i=1}^q \vec{a}_i^{\text{Guess}} h_i^q = \vec{b}_c + \sum_{i=1}^{\beta} \vec{b}_i^{\text{pub}} p_i + \sum_{i=1}^q \vec{b}_i^{\text{Guess}} h_i^q.$$

Redefine $\vec{a} := \vec{a} - \vec{b}$, this is equivalent to

$$-\vec{a}_c = \sum_{i=1}^{\beta} \vec{a}_i^{\text{pub}} p_i + \sum_{i=1}^q \vec{a}_i^{\text{Guess}} h_i^q$$

Conditioned on not yet having received “Win” from some oracle the adversary’s view is independent of the actual values syn_i . We split our analysis into three cases:

1. If there exists some $\vec{a}_i^{\text{pub}} \neq 0$ then the overall sum contains the hash h_i which is uniformly distributed and unknown, so this probability is $2^{-\ell}$.
2. If $\vec{a}^{\text{pub}} = 0$ and $\text{wt}(\vec{a}^{\text{Guess}}) \leq \gamma$ then,

$$\Pr[\langle \vec{a}^{\text{Guess}}, \text{syn}_i \rangle = -\vec{a}_c] \leq \max\{2^{-\ell}, 2^{-\alpha_{\text{Subset}}^{\min}} 2^{-\alpha_{\text{Code}}^{\min, \gamma}}\}$$

as either the hash collides with λ_c or the hash is removed and the syndrome collides.

3. If $\vec{a}^{\text{pub}} = 0$ and $\text{wt}(\vec{a}^{\text{Guess}}) > \gamma$ then it must be the case that for all $\vec{a}_i^{\text{Guess}}$, it must be true that either 1) h_q^i was defined to syn_i or 2) that there was some hash value that collided with $-\vec{a}_c$. By assumption the chance of the first case occurring is at most $2^{-\alpha_{\text{Subset}}^{\min, \gamma}}$. The probability of the second case occurring is at most $(q + \beta)2^{-\ell}$. (We note this second case is over the number of evaluations of hashes seen by the adversary and does not have to consider the pairs below.)

The statement of the claim follows by considering the total number of such pairs, $\vec{a}, \vec{b}$. □

The proof of Lemma 1 completes by noting that, conditioned on not having received win, the first query to $\text{Hash}_2/\text{Hash}_3$ succeeds with probability $2^{-H_\infty(W)}$ for the true point with the probability that some other point has the same hash as w of $2^{-\ell}$. □

This completes the proof of Theorem 1. □

Using code-offset in place of syndrome In the Introduction and the above proof, we focus on the syndrome construction which stores $\mathbf{H}w$. All of the above analysis naturally extends to the code offset construction which samples a random $\mathbf{x} \in \{0, 1\}^k$ and stores $\mathbf{A}\mathbf{x} + w$. Supporting this construction requires changing:

1. s to $n - \mu$ (instead of $n - \mu - k$) and
2. In `DecodeECC`, the value $z = g^{-1}(x)$ is now potentially close to a codeword (rather than being a syndrome) so an additional step is needed to compute $\text{syn} = \mathbf{H}z$.

However, all of the analysis remains the same as we allow the adversary to win as soon as they call `Invert` on an item that doesn't contain any unknown hash values.

Reusability with independent subsets We do not formally define reusability in this work [Boy04, CFP⁺16], informally it is security when the adversary supplies $W_1, \dots, W_{\text{Reuse}}$ correlated probability distributions and sees all keys except one may be replaced with a random value.

Shukla et al. showed sample-then-lock is fully reusable with arbitrary correlation between randomness used in `Gen` [SDF⁺25]. Unfortunately, this does not seem to be the case for Sample-then-Correct. Consider the case of two enrollments that use the same subsets and two distributions W_1, W_2 that only differ in the first two bits. There will likely be some subset $\mathcal{I}^*$ that contains the first two bits. Because the rest of $w_1, w_2 \leftarrow (W_1, W_2)$ are identical, they will have the same $\text{syn}_{\overline{W_{\mathcal{I}^*}}}$, this allows an adversary to use whether the unmasked `syn` values are the same between w_1 and w_2 as a check whether the hash value is correct. In technical terms, the problem is that $\alpha_{\text{Subset}}^\gamma$ may be as small as α_{Subset} and α_{Code} may be 0. Looking ahead to Section 4, when subsets are uniform, we are able to bound the maximum intersection between sampled subsets, and treat the problem of predicting γ subsets into a problem of predicting one larger subset.

However, this attack does not apply if subsets are not reused between enrollments as this can be modeled as just increasing β .

4 Evaluation

Our primary goal is to estimate the tradeoff between True Accept Rate (TAR) and entropy for iris, face, and fingerprint biometrics in our sample-then-correct fuzzy extractor. Our main comparison point is the sample-then-lock fuzzy extractor using the ClearEyes [SDF⁺25] biometric feature extractor.

4.1 Datasets

We use three datasets for our evaluations: the Hong Kong Polytechnic University 3D Fingerprint Images Database [LK17], the CBSR NIR Face Dataset [LCLZ07], and the IIT Delhi Iris Dataset [KP10]. For faces, we reuse the AdaFace [KJL22] face recognition model, which is pretrained on the MS1MV3 Dataset [DGA⁺21] and finetuned on the Webface4M Dataset [ZHD⁺21], see more details below.

Both the IIT Delhi and CBSR NIR Face datasets are captured at near-infrared wavelengths. Hong Kong Polytechnic University 3D Fingerprint Images Database contains multiple datasets; we utilize only the processed grayscale fingerprint images from the first session. All datasets were preprocessed to remove duplicates.

We note that all of these datasets are academic, limited in scale, and exhibit substantial demographic biases. The datasets often focus on a particular ethnic group of college-age individuals. Understanding feature extractor performance on balanced data is a current research problem in the biometrics community, who are designing synthetic biometric generation tools [MUD23, CRR⁺24].

As a piece of notation, we refer to a single class in the dataset as a biometric and use readings to refer to different images of biometrics. We use like readings to refer to readings from the same biometric, and unlike readings to refer to readings from different biometrics.

4.2 Metrics

We now discuss how to assess the concrete security of sample-then-correct. For correctness, we use the standard metric of TAR which measures the fraction of the time that two readings from the same biometric produce the same key.

Measuring security is more complicated. Since fuzzy extractor security is upper bounded by the fuzzy min-entropy (and min-entropy) [FRS16, FRS20] of the distribution W that biometrics are drawn from, estimating entropy of distributions becomes a necessary component of evaluation. Unfortunately, entropy estimation is an inherently heuristic prospect [VV10, VV11] if one does not have an exponential number of samples in the underlying entropy. As academic data sets often have size of roughly 10^4 we are forced to resort to heuristics. Common biometric tests on rely on the distribution of distances between readings of different biometrics. Of course, this distribution is being estimated distances between readings in the biometric dataset. The basic Daugman [Dau04] test for estimating entropy is $\text{EntTest}(\text{DSet})$:

1. Compute histogram of all fractional Hamming distances between readings of different biometrics.
2. Find the mean $\bar{w}$ and standard deviation σ of this histogram.
3. Compute the degrees of freedom $\text{dF} = \bar{w}(1 - \bar{w})/\sigma^2$.
4. Min-entropy is

$$e = \min\{-\log(\bar{w}), -\log(1 - \bar{w})\} * \text{dF}.$$

The intuition for the above is that the higher entropy the biometric, the tighter that readings of different biometrics will center at $\bar{w}$. Also as a reminder, we additionally ran NIST test suite [BRS⁺08], but the results of Daugman’s test were always smaller. Daugman argued for validity of the test based on the fact that the distance histogram fit a binomial with parameters $(\text{dF}, \bar{w})$ well and that distribution has the above min-entropy. We now review how we transform the evaluation of the relevant entropy metrics using the heuristic entropy test of Daugman [Dau06].

Estimating Different Min-entropies We now summarize how our different entropy measurements, α , α_{fuzz} , α_{Subset} , $\alpha_{\text{Subset}}^\gamma$, $\alpha_{\text{code}}^\gamma$ are estimated using the above entropy test.

1. α , the overall entropy of the distribution. Run the Daugman test above.
2. α_{fuzz}^t , the fuzzy min-entropy, we take the binomial computed from estimating α and for a distance t corresponding to the error-correction capacity of the used $\mathbf{A}$ we compute the negative log of the probability of a binomial with parameters $e, \bar{w}$ having an outcome of weight less than t/n .

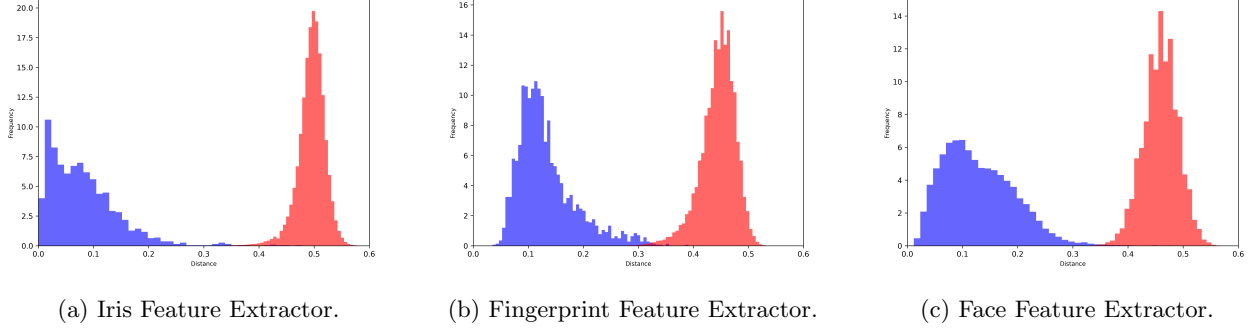


Figure 5: Fractional Hamming distance comparisons for the three feature extractors evaluated using Sample-then-Correct. Blue distributions represent readings of the same biometric while red distributions represent readings of different biometrics.

This calculation approximates the probability that two points in the distribution are within distance t . Fuzzy min-entropy captures the probability of multiple points being within distance t of any point w^* (not necessarily in the distribution).

3. α_{Subset} , the entropy of subsets of the distribution. For a size μ , we compute β subsets $\mathcal{I}_i$ of size μ ,
 - (a) Assess the entropy of each subset by using the Daugman test by restricting $\bar{w}$ to the subset $\mathcal{I}_i$, and
 - (b) Output the minimum.

We set $\beta = 110\text{K}$ for our single enrollment parameters and $\beta = 18334$ for our multiple enrollment parameters.

4. $\alpha_{\text{Subset}}^\gamma$, the min-entropy of $\gamma + 1$ subsets together. Across β subsets we compute the size of the maximum intersection between two subsets, denoted as **Inter**. We compute $\mu_\gamma = \mu + \gamma(\mu - \text{Inter})$. The number μ_γ is the minimum number of unique bits contained in $\gamma + 1$ subsets. We then run the Daugman test on subsets of size μ_γ .
5. $\alpha_{\text{code}}^\gamma$ the min-entropy of linear combinations of syndromes. This is the most complicated quantity to assess as it should be computed across all linear combinations λ (of size of the syndrome or code offset) and there are too many to assess this exhaustively. We assess linear combinations of weight at most γ where the coefficients are small $[-2^{10}, 2^{10}]$ using the above method. Taking the minimum over these values assessment yields a large number often in the thousands. Our best hypothesis is that the entropy test additionally assesses the randomness of λ .

We add a cap to this number of $\alpha_{\text{code}}^\gamma \leq \alpha - \alpha_{\text{Subset}}$. This would be the value if all linear combinations of $\mathbf{H}w_{\mathcal{I}_i}$ had all of the entropy not contained in $w_{\mathcal{I}_i}$. We also note that the relevant term in the analysis is $\alpha_{\text{code}}^\gamma + \alpha_{\text{Subset}}$. When the cap is used, this evaluates to α and is independent of the subset size.

In our evaluation we vary γ from 1 to 15 and compute

$$\max_{1 \leq \gamma \leq 15} \min\{\alpha_{\text{code}}^\gamma, \alpha_{\text{Subset}}^\gamma\},$$

to decide what γ to use for a parameter configuration. Across biometrics, usually $\gamma = 15$ considers subsets whose length is nearly the entire feature vector.

4.3 Feature Extractors and Training

To ensure fair and rigorous comparisons, we adopt different strategies for feature extraction across biometrics. We partition each dataset into two halves: one for training the feature extractors and the other reserved exclusively for

testing. This is called the open-dataset setting. This split ensures that the system is evaluated on completely unseen biometrics, providing a realistic assessment of performance for a larger unseen population. This approach also serves a critical security purpose: it prevents adversaries from exploiting model inversion attacks [GRGB⁺12, FJR15, AF20, AMF22, TKAK23, WGCJ22, LNWS23, ASW⁺24] or similar techniques that could potentially reconstruct original biometric data from the protected templates. By ensuring that the feature extractor has no prior exposure to individual subjects during training, we remove the risk of information leakage through the feature extractor itself.

Iris For iris features, we directly reuse the feature extractor trained by Shukla et al. [SDF⁺25]. Their dataset is split into three disjoint sets:

- **Train** All images of the left iris from all subjects.
- **stats_W** 40 of the 202 right eyes, used by Shukla et al. to compute statistics **stats_W** to sample non-uniform subsets. We exclude this set from any evaluation.
- **Test** The remaining 162 right eyes used for performance evaluation.

Shukla et al. [SDF⁺25] introduce a two-stage approach: a first-stage feature extractor produces initial embeddings, followed by a second-stage feature extractor that learns the non-uniform sampling algorithm. This stage-two extractor analyzes correlations in the bit representation across higher dimensions to select stable bits optimized for sample-then-lock fuzzy extractor [CFP⁺16].

We do not present results of the two stage training of the feature extractor. We found that this second stage sacrificed overall entropy to increase the entropy of subsets. Since our construction depends on many other properties of the biometric distribution we found the first stage of training produced superior results. However, when describing their results, we use their stage two feature extractor (see Table 2 and Figure 1).

Face For faces, we fine-tune AdaFace [KJL22] to better adapt to the open-set setting by unfreezing only the deeper residual stages of the model and training them with the AdaFace loss, while keeping the early layers fixed. The training data is a custom subset of WebFace4M [ZHD⁺21] constructed using a sampling script that randomly selects 10,000 training identities with 10 images each (100,000 total training images) and 2,000 disjoint validation identities with 10 images each (20,000 total validation images), ensuring no identity overlap between splits. This approach preserves the general face representation learned from MS1MV3 while allowing the higher-level features to better generalize to the open-set face recognition regime. We then evaluate on the CBSR-NIR benchmark [LCLZ07], which is segmented using RetinaFace [DGV⁺20]. All classes from CBSR-NIR are used to test.

Fingerprint For fingerprints, we retrain the same feature extractor architecture [SDF⁺25] from scratch using the processed grayscale fingerprint images from the Hong Kong Polytechnic University 3D Fingerprint Images Database [LK17], as pre-trained models optimized for these biometrics were not available from the original work. We select first 50% of classes to be used for testing and 50% of remaining classes to be used for training.

Quality of Features The histograms resulting from the three training pipelines are visible in Figure 5 with comparisons between readings of the same biometric in blue and comparisons between readings of different biometrics in red. As a reminder a perfect feature extractor will have the entire blue distribution at distance 0 and the entire red distribution at distance .5. The iris has the lowest distance between readings of the same biometric and the smallest variance in readings of different biometrics. These values directly impact the size of subsets that can be used and the entropy estimates discussed above. Roughly, the size of subsets that can be locked decreases exponentially with the fractional error distance to be supported. Having the iris with the lowest distance makes sense as Shukla et al. [SDF⁺25] heavily optimized their feature extractor for the iris. We use off the shelf models for the other two biometrics.

Size	Face			Fingerprint			Iris		
	$\alpha_{\text{Subset}}^\gamma$	$\alpha_{\text{Subset}+35}$	TAR	$\alpha_{\text{Subset}}^\gamma$	$\alpha_{\text{Subset}+35}$	TAR	$\alpha_{\text{Subset}}^\gamma$	$\alpha_{\text{Subset}+35}$	TAR
20	88	50	0.99	63	47	1.00	72	54	1.00
25	63	53	0.99	74	51	1.00	80	58	0.99
30	72	57	0.98	58	54	0.99	100	63	0.98
35	83	61	0.97	62	57	0.96	107	67	0.98
40	88	64	0.95	67	59	0.92	107	72	0.97
45	97	68	0.91	72	62	0.89	107	76	0.96
50	103	70	0.86	74	64	0.85	107	81	0.94
55	109	75	0.80	77	67	0.81	120	85	0.92
60	109	78	0.74	80	71	0.76	114	90	0.89
65	109	82	0.68	81	72	0.71	120	93	0.87
70	109	84	0.62	85	73	0.65	114	98	0.84
75	109	87	0.58	88	78	0.59	120	103	0.81
80	109	91	0.53	89	80	0.51	114	107	0.77
85	109	94	0.49	92	80	0.45	120	111	0.73
90	109	97	0.45	93	78	0.40	128	116	0.72
95	109	101	0.41	94	78	0.32	128	119	0.68
100	109	103	0.38	96	77	0.29	135	124	0.66

Table 4: Comparison of Face, Fingerprint, and Iris entropy and TAR. At size 60, we switch the code from correcting $.15n$ errors to $.20n$ errors. At larger subset sizes correcting $.20n$ fuzzy min-entropy becomes the limiting factor in the security estimate, see Table 5.

	Face	Fingerprint	Iris
α	223	206	383
$\alpha_{\text{fuzz}}^{t/n=0.15}$	84	74	153
$\alpha_{\text{fuzz}}^{t/n=0.20}$	57	51	108
$(\alpha_{\text{code}}^\gamma + \alpha_{\text{Subset}})/2$	111	103	191

Table 5: Comparison of Entropy Parameters that don’t vary with subset size. Note in our experiments, $\alpha_{\text{code}}^\gamma + \alpha_{\text{Subset}} = \alpha$ so this value is independent of subset size. As discussed above, for μ past 60 fuzzy min-entropy at $t = .2$ becomes the limiting term in the security estimate so we switch to $t = .15$.

Further improvements in feature extraction Training new open-dataset feature extractors which produce embeddings specifically optimized for fingerprint and face biometrics is a substantial research challenge. Here we focus on the building stronger cryptographic algorithms rather than optimizing feature extraction for any biometrics. We continue to encourage the biometric research community to develop improved feature extraction methods tailored to these biometrics and to ensure fair performance across demographics. Advances directly enhance the security and accuracy of our fuzzy extractor.

4.4 Correctness and Security

Correctness We measure correctness by TAR, which quantifies the system’s ability to correctly authenticate legitimate users. Following the methodology of prior work [SDF⁺25], we compute TAR in two distinct ways to provide a comprehensive evaluation of system performance under different enrollment strategies.

For single-enrollment TAR, we enroll a single reading for each biometric subject in the test set and then attempt to authenticate with all other available readings for that same subject. Each authentication attempt is evaluated independently; a successful key reproduction is counted as a true accept, while a failed key reproduction is counted

as a false rejection. These values are summarized in Table 4. As mentioned above, the smaller the mean (and variance) of the blue distribution in Figure 5, the better the biometric is able to scale to larger subsets while maintaining TAR.

To evaluate performance under more practical enrollment conditions, we also compute a multi-enrollment TAR, similar to ClearEyes [SDF⁺25]. In this setting, we enroll a set of 3 readings for each subject and then authenticate using a separate set of 2 new readings. A subject is considered successfully authenticated if any of the 6 authentication attempts (i.e., at least one enrolled-authentication pair) produces a successful key reproduction. In this evaluation, we reduce the number of lockers by a factor of 6 to ensure that the overall authentication still completes in 1s. These results are summarized in Table 6 where we can roughly lift from a TAR of .6 to .9.

Security We set our primary security measure to be

$$\min \left\{ \alpha, \alpha_{\text{Subset}} + 35, \frac{\alpha_{\text{Subset}}^\gamma}{2}, \frac{\alpha_{\text{code}} + \alpha_{\text{Subset}}}{2}, \alpha_{\text{fuzz}} \right\}$$

A summary of these statistics is shown in Tables 4 and 5. In Table 4, we highlight the subset sizes that are presented in the Introduction in Table 2.

4.5 Implementation

We implement our Sample-then-Correct construction in C to ensure computational efficiency and portability across different platforms. The implementation leverages OpenSSL as the primary cryptographic library, providing well-tested and standardized implementations of hash functions, PRFs, and other cryptographic operations required by our construction. For BCH error correction, we utilize the code provided by Morelos-Zaragoza [MZ06], which offers an implementation suitable for our requirements.

Our implementation uses BCH codes over the Galois field $\text{GF}(2^6)$, yielding a code length of $n = 63$ bits. We configure the code with two error correction modes depending on the subset size. For subsets smaller than 60 positions, we use a high-tolerance mode with $t = 12$, correcting up to 12 bit errors per codeword. This configuration yields $k = 16$ information bits per 63-bit codeword with 47 redundancy bits, offering approximately 20% error correction capacity. For larger subsets (60 positions or more), we switch to $t = 9$, allowing correction of up to 9 bit errors per codeword. This produces a message length of $k = 24$ information bits with 39 redundancy bits, providing approximately 15% error tolerance relative to the codeword length. The complement bitstring is partitioned into chunks of k bits, with each chunk independently encoded. We note that this correction is slightly weaker than a full correction of either 15 or 20% of errors as decoding fails if a single chunk has too many errors. We found choosing subsets and thus the complement being placed in the syndrome to sufficiently randomize the position of errors that this did not occur.

4.6 Results

We demonstrate the effectiveness of Sample-then-Correct across iris, fingerprint, and face biometrics. All results use conservative parameters with a maximum authentication time of 1 second on a single thread. In reality, the average reading for an iris takes only 200ms to authenticate.

As shown in Table 6, for iris recognition, we achieve a TAR of 0.91 at 111 bits of entropy and a TAR of 0.95 at 98 bits of entropy, with fuzzy min-entropy values of 107 and 153 at decoding thresholds of $t = .20n$ and $t = .15n$, respectively. For fingerprints, we obtain a TAR of 0.92 at 72 bits of entropy and a TAR of 0.96 at 71 bits of entropy, with fuzzy min-entropy values of 51 and 74, respectively. For face recognition, we achieve a TAR of 0.93 at 84 bits of entropy and a TAR of 0.95 at 82 bits of entropy, with fuzzy min-entropy values of 57 and 84, respectively. As a reminder, the TAR values switch from $t = .20n$ to $t = .15n$ at $\mu = 60$ as fuzzy min-entropy begins to become the limiting factor in the security estimate. All values in Table 6 are presented with $t = 0.15n$.

For fair comparison, we use the same number of enrollment and authentication readings as ClearEyes [SDF⁺25], since the iris dataset contains only 5 readings per class. However, performance can be readily improved through simple optimizations. For instance, increasing authentication readings from 2 to 3 for face recognition yields a TAR of 0.96 while maintaining the same 1s single-thread constraint.

Biometric	Size	Sec. Estimate	TAR	Multi-TAR
Face	65	82	.68	.95
	70	84	.62	.93
Fingerprint	60	71	.76	.96
	65	72	.71	.92
Iris	70	98	.84	.95
	85	111	.73	.91

Table 6: Amplification of TAR by enrolling and authenticating with 3 and 2 readings respectively. At all of these sizes TAR and Multi-TAR are presented with correcting capacity of $.15n$.

Parallelization enables faster authentication of more lockers. Using 6 cores (e.g., the Apple A18 processor), we can authenticate in $< .2s$ instead of 1s. We do not implement the low-entropy locker filtering used in **ClearEyes**. This filtering samples subsets and then rejects subsets which have little entropy in comparison to other subsets. This method provides approximately 5 bits of entropy gain.

5 Conclusion

This work proposes a new family of fuzzy extractors called sample-then-correct that combine the best of sample-then-lock and error-correction based fuzzy extractors. Our construction overcomes the unrealistic independence assumptions of information-theoretic constructions and the low security achieved by sample-then-lock on the fingerprint and face. This is the first time a fuzzy extractor construction has achieved the combination of:

- practical TAR ($> 90\%$),
- security of > 70 bits for all three major biometrics, and
- reliance on only entropy-based statistical assumptions.

For all three tested biometrics, the fuzzy min-entropy of the feature extractor becomes a bottleneck indicating that the cryptography is extracting all achievable security.

All schemes are implemented and complete authentication in under a second on commodity hardware. Perhaps the most interesting open problems resulting from this work are understanding security with respect to quantum adversaries and finding codes that have strong controls $\alpha_{\text{code}}^\gamma$. Specifically can one design codes that ensure γ -wise independence of syndromes.

Acknowledgements

The authors would like to thank Dapple Security for proposing the initial concept of sample-then-correct, which is formalized and analyzed in this manuscript. The authors thank anonymous reviewers and Leonid Reyzin for their helpful comments in improving the manuscript. The work of the UConn authors was supported in part by grants from Dapple Security and by NSF grants #2141033 and #2232813.

Ethical Considerations

This work focuses on the secure storage of biometrics and therefore touches on biometric privacy, bias, and the deployment of cryptography. This work develops new cryptographic techniques for securing biometrics. We believe that providing strong cryptography is critical to securing biometrics, which are already widely deployed. As mentioned in Section 4, all used datasets are academic in nature and have demographic biases. We strongly recommend evaluation on fairer datasets before large-scale deployment.

Open Science

Code and checkpointed models are available via this [Zenodo record](#). Some datasets used are licensed and can not be included. Code is at this [Github repo](#).

References

- [ABC⁺18] Quentin Alamélou, Paul-Edmond Berthier, Chloé Cachet, Stéphane Cauchie, Benjamin Fuller, Philippe Gaborit, and Sailesh Simhadri. Pseudoentropic isometries: A new framework for fuzzy extractor reusability. In *AsiaCCS*, 2018.
- [ACD⁺24] Sohaib Ahmad, Sixia Chen, Luke Demarest, Benjamin Fuller, Caleb Manicke, Alexander Russell, and Amey Shukla. IrisLock: Iris biometric key derivation with 42 bits of security. *Cryptology ePrint Archive*, Paper 2024/100, 2024.
- [ACEK17] Daniel Apon, Chongwon Cho, Karim Eldefrawy, and Jonathan Katz. Efficient, reusable fuzzy extractors from LWE. In *International Conference on Cyber Security Cryptography and Machine Learning*, pages 1–18. Springer, 2017.
- [ACF⁺22] Daniel Apon, Chloe Cachet, Benjamin Fuller, Peter Hall, and Feng-Hao Liu. Nonmalleable digital lockers and robust fuzzy extractors in the plain model. In Shweta Agrawal and Dongdai Lin, editors, *Advances in Cryptology – ASIACRYPT 2022*, pages 353–383, Cham, 2022. Springer Nature Switzerland.
- [AF20] Sohaib Ahmad and Benjamin Fuller. Resist: Reconstruction of irises from templates. In *2020 IEEE International Joint Conference on Biometrics (IJCB)*, pages 1–10. IEEE, 2020.
- [AMF22] Sohaib Ahmad, Kaleel Mahmood, and Benjamin Fuller. Inverting biometric models with fewer samples: Incorporating the output of multiple models. In *2022 IEEE International Joint Conference on Biometrics (IJCB)*, pages 1–11. IEEE, 2022.
- [ASW⁺24] Sani M Abdullahi, Shuifa Sun, Beng Wang, Ning Wei, and Hongxia Wang. Biometric template attacks and recent protection mechanisms: A survey. *Information Fusion*, 103:102144, 2024.
- [BBR88] Charles H. Bennett, Gilles Brassard, and Jean-Marc Robert. Privacy amplification by public discussion. *SIAM Journal on Computing*, 17(2):210–229, 1988.
- [BC10] Nir Bitansky and Ran Canetti. On strong simulation and composable point obfuscation. In *Advances in Cryptology–CRYPTO 2010*, pages 520–537. Springer, 2010.
- [BDF⁺11] Dan Boneh, Özgür Dagdelen, Marc Fischlin, Anja Lehmann, Christian Schaffner, and Mark Zhandry. Random oracles in a quantum world. In *International conference on the theory and application of cryptography and information security*, pages 41–69. Springer, 2011.
- [Ber10] Daniel J Bernstein. Grover vs. mceliece. In *International Workshop on Post-Quantum Cryptography*, pages 73–80. Springer, 2010.
- [BKM⁺18] Allison Bishop, Lucas Kowalczyk, Tal Malkin, Valerio Pastro, Mariana Raykova, and Kevin Shi. A simple obfuscation scheme for pattern-matching with wildcards. In *Annual International Cryptology Conference*, pages 731–752. Springer, 2018.
- [BLMZ19] James Bartusek, Tancrede Lepoint, Fermi Ma, and Mark Zhandry. New techniques for obfuscating conjunctions. In *Eurocrypt*, pages 636–666, 2019. <https://eprint.iacr.org/2018/936>.
- [BMT25] Alexandra Boldyreva, Deep Inder Mohan, and Tianxin Tang. May the force not be with you: Brute-force resistant biometric authentication and key reconstruction. In *Proceedings of the 2025 ACM SIGSAC Conference on Computer and Communications Security*, pages 3620–3634, 2025.

- [Boy04] Xavier Boyen. Reusable cryptographic fuzzy extractors. In *Proceedings of the 11th ACM conference on Computer and Communications Security*, pages 82–91, 2004.
- [BR93] Mihir Bellare and Phillip Rogaway. Random oracles are practical: A paradigm for designing efficient protocols. In *ACM Conference on Computer and Communications Security*, pages 62–73, 1993.
- [BRS⁺08] Lawrence Bassham, Andrew Rukhin, Juan Soto, James Nechvatal, Miles Smid, Elaine Barker, Stefan Leigh, Mark Levenson, Mark Vangel, David Banks, et al. A statistical test suite for random and pseudorandom number generators for cryptographic applications. Technical report, National Institute of Standards and Technology, 2008.
- [CD08] Ran Canetti and Ronny Ramzi Dakdouk. Obfuscating point functions with multibit output. In *Advances in Cryptology–EUROCRYPT 2008*, pages 489–508. Springer, 2008.
- [CFP⁺16] Ran Canetti, Benjamin Fuller, Omer Paneth, Leonid Reyzin, and Adam Smith. Reusable fuzzy extractors for low-entropy distributions. In *Advances in Cryptology – EUROCRYPT*, pages 117–146. Springer, 2016.
- [CFP⁺21] Ran Canetti, Benjamin Fuller, Omer Paneth, Leonid Reyzin, and Adam Smith. Reusable fuzzy extractors for low-entropy distributions. *Journal of Cryptology*, 34(1):1–33, 2021.
- [CHRF24] Chloe Cachet, Ariel Hamlin, Maryam Rezapour, and Benjamin Fuller. Upgrading fuzzy extractors. In *International conference on applied cryptography and network security*, pages 156–182. Springer, 2024.
- [CJKL18] Jung Hee Cheon, Jinhyuck Jeong, Dongwoo Kim, and Jongchan Lee. A reusable fuzzy extractor with practical storage size: Modifying Canetti et al.’s construction. In *Australasian Conference on Information Security and Privacy*, pages 28–44. Springer, 2018.
- [CRR⁺24] Praveen Kumar Chandaliya, Kiran Raja, Raghavendra Ramachandra, Zahid Akhtar, and Christoph Busch. Towards inclusive face recognition through synthetic ethnicity alteration. In *2024 IEEE 18th International Conference on Automatic Face and Gesture Recognition (FG)*, pages 1–8. IEEE, 2024.
- [Dau04] John Daugman. How iris recognition works. *Circuits and Systems for Video Technology, IEEE Transactions on*, 14(1):21 – 30, January 2004.
- [Dau06] J. Daugman. Probing the uniqueness and randomness of iriscodes: Results from 200 billion iris pair comparisons. *Proceedings of the IEEE*, 94(11):1927–1935, 2006.
- [DFR21] Luke Demarest, Benjamin Fuller, and Alexander Russell. Code offset in the exponent. In *2nd Conference on Information-Theoretic Cryptography (ITC 2021)*, 2021.
- [DGA⁺21] Jiankang Deng, Jia Guo, Xiang An, Zheng Zhu, and Stefanos Zafeiriou. Masked face recognition challenge: The insightface track report. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 1437–1444, 2021.
- [DGV⁺16] Jeroen Delvaux, Dawu Gu, Ingrid Verbauwhede, Matthias Hiller, and Meng-Day Yu. Efficient fuzzy extraction of puf-induced secrets: Theory and applications. In *International Conference on Cryptographic Hardware and Embedded Systems*, pages 412–431. Springer, 2016.
- [DGV⁺20] Jiankang Deng, Jia Guo, Evangelos Ververas, Irene Kotsia, and Stefanos Zafeiriou. Retinaface: Single-shot multi-level face localisation in the wild. In *CVPR*, 2020.
- [DKRS06] Yevgeniy Dodis, Jonathan Katz, Leonid Reyzin, and Adam Smith. Robust fuzzy extractors and authenticated key agreement from close secrets. In Cynthia Dwork, editor, *Advances in Cryptology - CRYPTO 2006*, volume 4117 of *Lecture Notes in Computer Science*, pages 232–250. Springer Berlin Heidelberg, 2006.

- [DORS08] Yevgeniy Dodis, Rafail Ostrovsky, Leonid Reyzin, and Adam Smith. Fuzzy extractors: How to generate strong keys from biometrics and other noisy data. *SIAM Journal on Computing*, 38(1):97–139, 2008.
- [DRS04] Yevgeniy Dodis, Leonid Reyzin, and Adam Smith. Fuzzy extractors: How to generate strong keys from biometrics and other noisy data. In *EUROCRYPT*, pages 523–540. Springer, 2004.
- [FJR15] Matt Fredrikson, Somesh Jha, and Thomas Ristenpart. Model inversion attacks that exploit confidence information and basic countermeasures. In *Proceedings of the 22nd ACM SIGSAC conference on computer and communications security*, pages 1322–1333, 2015.
- [FMR13] Benjamin Fuller, Xianrui Meng, and Leonid Reyzin. Computational fuzzy extractors. In *Advances in Cryptology-ASIACRYPT 2013*, pages 174–193. Springer, 2013.
- [FP19] Benjamin Fuller and Lowen Peng. Continuous-source fuzzy extractors: source uncertainty and insecurity. In *2019 IEEE International Symposium on Information Theory (ISIT)*, pages 2952–2956. IEEE, 2019.
- [FRS16] Benjamin Fuller, Leonid Reyzin, and Adam Smith. When are fuzzy extractors possible? In *International Conference on the Theory and Application of Cryptology and Information Security*, pages 277–306. Springer, 2016.
- [FRS20] Benjamin Fuller, Leonid Reyzin, and Adam Smith. When are fuzzy extractors possible? *IEEE Transactions on Information Theory*, 66(8):5282–5298, 2020.
- [Ful24] Benjamin Fuller. Impossibility of efficient information-theoretic fuzzy extraction. *Designs, Codes and Cryptography*, 92(7):1983–2009, 2024.
- [GR05] Venkatesan Guruswami and Atri Rudra. Limits to list decoding reed-solomon codes. In *Proceedings of the thirty-seventh annual ACM symposium on Theory of computing*, pages 602–609, 2005.
- [GRGB⁺12] Javier Galbally, Arun Ross, Marta Gomez-Barrero, Julian Fierrez, and Javier Ortega-Garcia. From the iriscodes to the iris: A new vulnerability of iris recognition systems. *Black Hat Briefings USA*, 1:8, 2012.
- [Gro96] Lov K Grover. A fast quantum mechanical algorithm for database search. In *Proceedings of the twenty-eighth annual ACM symposium on Theory of computing*, pages 212–219, 1996.
- [GS98] Venkatesan Guruswami and Madhu Sudan. Improved decoding of reed-solomon and algebraic-geometric codes. In *Foundations of Computer Science, 1998. Proceedings. 39th Annual Symposium on*, pages 28–37. IEEE, 1998.
- [GX13] Venkatesan Guruswami and Chaoping Xing. List decoding reed-solomon, algebraic-geometric, and gabidulin subcodes up to the singleton bound. In *Proceedings of the forty-fifth annual ACM symposium on Theory of computing*, pages 843–852, 2013.
- [HAD06] Feng Hao, Ross Anderson, and John Daugman. Combining crypto with biometrics effectively. *Computers, IEEE Transactions on*, 55(9):1081–1088, 2006.
- [KHF⁺20] Paul Kocher, Jann Horn, Anders Fogh, Daniel Genkin, Daniel Gruss, Werner Haas, Mike Hamburg, Moritz Lipp, Stefan Mangard, Thomas Prescher, et al. Spectre attacks: Exploiting speculative execution. *Communications of the ACM*, 63(7):93–101, 2020.
- [KJK⁺21] Sunpill Kim, Yunseong Jeong, Jinsu Kim, Jungkon Kim, Hyung Tae Lee, and Jae Hong Seo. Ironmask: Modular architecture for protecting deep face template. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 16125–16134, 2021.

- [KJL22] Minchul Kim, Anil K Jain, and Xiaoming Liu. Adaface: Quality adaptive margin for face recognition. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 18750–18759, 2022.
- [KP10] Ajay Kumar and Arun Passi. Comparison and combination of iris matchers for reliable personal authentication. *Pattern recognition*, 43(3):1016–1026, 2010.
- [LCLZ07] Stan Z Li, RuFeng Chu, ShengCai Liao, and Lun Zhang. Illumination invariant face recognition using near-infrared images. *IEEE Transactions on pattern analysis and machine intelligence*, 29(4):627–639, 2007.
- [LK17] Chenhao Lin and Ajay Kumar. Tetrahedron based fast 3d fingerprint identification using colored leds illumination. *IEEE transactions on pattern analysis and machine intelligence*, 40(12):3022–3033, 2017.
- [LNWS23] Song-Hong Lee, Cing-Ping Nien, Shun-Chi Wu, and A Lee Swindlehurst. Reconstruction attacks in template-based ecg biometric recognition systems. *IEEE Internet of Things Journal*, 2023.
- [LSG⁺18] Moritz Lipp, Michael Schwarz, Daniel Gruss, Thomas Prescher, Werner Haas, Stefan Mangard, Paul Kocher, Daniel Genkin, Yuval Yarom, and Mike Hamburg. Meltdown. *arXiv preprint arXiv:1801.01207*, 2018.
- [MUD23] Andrey Makrushin, Andreas Uhl, and Jana Dittmann. A survey on synthetic biometrics: Fingerprint, face, iris and vascular patterns. *Ieee Access*, 11:33887–33899, 2023.
- [MZ06] Robert H Morelos-Zaragoza. *The art of error correcting coding*. John Wiley & Sons, 2006.
- [SDF⁺25] Amey Shukla, Luke Demarest, Benjamin Fuller, Sohaib Ahmad, Caleb Manicke, Alexander Russell, and Sixia Chen. Fuzzy extractors are practical: Cryptographic strength key derivation from the iris. In *CCS*, 2025.
- [Sho97] Victor Shoup. Lower bounds for discrete logarithms and related problems. In *International Conference on the Theory and Applications of Cryptographic Techniques*, pages 256–266. Springer, 1997.
- [ŠTO05] Boris Škorić, Pim Tuyls, and Wil Ophey. Robust key extraction from physical uncloneable functions. In *Applied Cryptography and Network Security: Third International Conference, ACNS 2005, New York, NY, USA, June 7-10, 2005. Proceedings 3*, pages 407–422. Springer, 2005.
- [TKAK23] Gioacchino Tangari, Shreesh Keskar, Hassan Jameel Asghar, and Dali Kaafar. On the adversarial inversion of deep biometric representations. *arXiv preprint arXiv:2304.05561*, 2023.
- [VV10] Gregory Valiant and Paul Valiant. A CLT and tight lower bounds for estimating entropy. In *Electronic Colloquium on Computational Complexity (ECCC)*, volume 17, page 9, 2010.
- [VV11] Gregory Valiant and Paul Valiant. Estimating the unseen: an $n/\log(n)$ -sample estimator for entropy and support size, shown optimal via new clts. In *Proceedings of the forty-third annual ACM symposium on Theory of computing*, pages 685–694, 2011.
- [WGCJ22] Kanishka P Wijewardena, Steven A Grosz, Kai Cao, and Anil K Jain. Fingerprint template invertibility: Minutiae vs. deep templates. *IEEE Transactions on Information Forensics and Security*, 18:744–757, 2022.
- [WL18] Yunhua Wen and Shengli Liu. Robustly reusable fuzzy extractor from standard assumptions. In *International Conference on the Theory and Application of Cryptology and Information Security*, pages 459–489. Springer, 2018.
- [WLG19] Yunhua Wen, Shengli Liu, and Dawu Gu. Generic constructions of robustly reusable fuzzy extractor. In *IACR International Workshop on Public Key Cryptography*, pages 349–378. Springer, 2019.

- [WLH18] Yunhua Wen, Shengli Liu, and Shuai Han. Reusable fuzzy extractor from the decisional Diffie–Hellman assumption. *Designs, Codes and Cryptography*, Jan 2018.
- [Zha21] Mark Zhandry. Redeeming reset indistinguishability and applications to post-quantum security. In *International Conference on the Theory and Application of Cryptology and Information Security*, pages 518–548. Springer, 2021.
- [ZHD⁺21] Zheng Zhu, Guan Huang, Jiankang Deng, Yun Ye, Junjie Huang, Xinze Chen, Jiagang Zhu, Tian Yang, Jiwen Lu, Dalong Du, et al. Webface260m: A benchmark unveiling the power of million-scale deep face recognition. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 10492–10502, 2021.
- [ZW25] Pengxu Zhu and Lei Wang. Hypersphere secure sketch revisited: Probabilistic linear regression attack on IronMask in multiple usage. *PoPETS*, 2025.