Cryptographic Authentication from the Iris

Sailesh Simhadri[1], James Steel[2], and Benjamin Fuller[2]

[1] Google Inc., Cambridge, MA USA
saileshsimhadri@gmail.com
[2] University of Connecticut, Storrs, CT USA
james.steel,benjamin.fuller@uconn.edu

Abstract. Biometrics exhibit noise between repeated readings. Due to the noise, devices store a plaintext *template* of the biometric. This stored template is an appetizing target for an attacker.
Fuzzy extractors derive a stable cryptographic key from biometrics (Dodis et al., Eurocrypt 2004). Despite many attempts, there are no iris key derivation systems that prove lower bounds on key strength.
Our starting point is a fuzzy extractor due to Canetti et al. (Eurocrypt 2016). We modify and couple the image processing and cryptographic algorithms. We then present a sufficient condition on the iris distribution for security, and analysis this condition using the ND0405 Iris dataset.
We build an iris key derivation system with 32 bits of security even when multiple keys are derived from the same iris. We acknowledge 32 bits of security is insufficient for a secure system. Multifactor systems hold the most promise for cryptographic authentication. Our scheme is suited for incorporation of additional *noiseless* factors such as a password.
Our scheme is implemented in C and Python and is open-sourced.

1 Introduction

Authentication schemes combine factors such as passwords, one-time codes, security questions, and social relationships [11]. Some providers use key derivation functions to derive cryptographic keys and then protect sensitive data using these keys. Depending on the *entropy* of the authentication factors, we can obtain bounds on how long it will take an adversary to correctly guess users' private information.

Biometrics are used to authenticate users on mobile devices (phones and tablets). In these systems, a template of the biometric reading is stored in a secure processor. Since the template is stored "in the clear," a secure processing component is necessary. Furthermore, it means that deploying biometric authentication in a client-server setting is risky. The client-server setting is still the majority of Internet authentication.

In the absence of a secure processor, two complementary lines of research emerged: interactive protocols and schemes that create a single value that allows for authentication (that is, non-interactive protocols). The interactive setting is well understood [16,28,8,32,27,18,9,26,31]. Importantly, interactive protocols do not consider server compromise in scope of the threat model. Their focus is

on ensuring an adversary that pretends to be either the client or server gains minimal information by engaging in the protocol. Furthermore, the interactive model is not applicable for a user authenticating to a device.

The non-interactive setting is not understood despite years of research. (We detail prior work in Section 1.1.) For many biometrics there is little in way of implementable work (current proposals either requiring exponential time [41,34,64] or semantically secure graded encodings [52]). We focus on building non-interactive key derivation from the iris [57]. We use the definition of fuzzy extractors [30,29]. (Our discussion applies to fuzzy commitments [45] and secure sketches [29].)

Fuzzy extractors derive stable keys from a biometric. Fuzzy extractors consist of two algorithms Gen, or generate, and Rep, or reproduce. The Gen algorithm takes an initial reading of the biometric, denoted w, deriving a key Key and a value Pub. The Rep algorithm is used at authentication time taking Pub and a later reading of the biometric, denoted w'. If the two readings of the biometric are similar enough then the same Key should be output by the algorithm. The security of a fuzzy extractor is analyzed assuming the adversary knows Pub.

The first generation of fuzzy extractors shared the same core construction and security analysis [25]. These constructions all used a variant of the one-time pad where the "pad" is a codeword from an error correcting code (rather than being uniformly distributed).

The quality of the derived key depends on the entropy of the biometric and size of the error-correcting code. Let W be a biometric of length n and suppose W has k bits of min-entropy. Suppose the error correcting code has 2^α codewords. Roughly, it is assumed the "one-time pad" leaks the entropy deficiency of the code or $n - \alpha$ bits. If one wishes to tolerate t bits of error between w and w', using bounds on the best code, this loss is at least $h_2(t/n) * n$.[3]

In many cases $h_2(t/n) * n$ is larger than k. Daugman's seminal paper on iris recognition [24] transformed iris images into a fixed length 2048 bit vector. Daugman reports error rates close to 10% in a controlled environment. For more realistic datasets the error rate is 30% (see Figure 3). In either case, $h_2(t/n) * n \geq h_2(.1) * 2048 \approx 874$ is larger than the estimated entropy of 249.[4] It is not known how to analyze the first generation of fuzzy extractors to argue security for the iris.

Biometrics cannot be changed or updated so provable, cryptographic security is crucial. A compromise affects an individual for their entire life. We focus on a strengthening of fuzzy extractors called *reusable* fuzzy extractors that allows derivation of multiple keys and multiple public values from the same biometric.

Recently, a second generation of fuzzy extractors emerged using cryptographic tools [33,20,2,1,63,62]. These constructions are reusable and only provide

[3] The quantity $h_2(t/n) * n$ is the binary entropy of t/n multiplied by n. The quantity $h_2(t/n) * n$ is larger than t (when $t \leq .5n$). For example, if $t = .1n$ then $h_2(t/n) * n \approx .427n$.

[4] Any distribution limited to people on the earth can be described using 33 bits. The estimate of 249 should be understood as the randomness involved in creating a new iris.

security against computationally bounded adversaries. Some of these constructions provide meaningful security when W has low entropy. However, this security requires W to have additional structure beyond entropy. There have been no empirical evaluations of whether biometrics exhibit this structure. Furthermore, these constructions are stated in asymptotic form and it is not clear what properties they provide for actual biometrics.

Our contribution We build the first key derivation system that provides meaningful albeit moderate provable security from the iris. Our scheme has been implemented and open-sourced [36]. The combination of cryptographic and statistical analysis estimates a security level of 32 bits. As a point of comparison, recent estimates place password entropy at 22 bits [48,10,61]. We do not believe this security level is sufficient for a stand alone system. Our hope is that this work serves as a catalyst for system designers to incorporate our construction into multi-factor authentication systems and that the overall system provides strong security. We discuss such a system below.

The starting point for our construction is the recent sample-then-lock scheme of Canetti et al. [20]. The idea of the scheme is simple: to hash the biometric in an "error-tolerant" way. Hashing the full biometric doesn't work (due to biometric noise). Instead, multiple random subsets of the biometric are hashed. That is, sample a random subset of bits, denoted $\mathcal{I}$, and hash w restricted to the bits of $\mathcal{I}$, denoted $\mathsf{Hash}(w_{\mathcal{I}})$, and use this value as a pad for a cryptographic Key. That is, store $(\mathsf{Key} \oplus \mathsf{Hash}(w_{\mathcal{I}}), \mathcal{I})$. This process is repeated with multiple subsets $\mathcal{I}_j$ and the same Key. Correctness follows if it is likely that in at least one subset $\mathcal{I}_j$, $w_{\mathcal{I}_j} = w'_{\mathcal{I}_j}$. The security analysis requires for a random subset $\mathcal{I}_j$ of bits that $w_{\mathcal{I}_j}$ has entropy with high probability over the choice of $\mathcal{I}_j$ (see Definition 1). This strengthens the requirement that the whole vector W has entropy.

We first estimate that subsampling iris bits produces a distribution with entropy (Figure 1). (Efficient entropy estimation is heuristic. Provably accurate entropy estimation [59,60] requires an exponentially large number of samples in the actual entropy of the distribution.) We use the same heuristic that occurs in previous biometric research. Roughly, the distances between transformed irises of different individuals are compared with the distances that would be produced by the binomial distribution whose entropy is computable. This is discussed in further in Section 4.

However, we find that the naive combination of iris processing and the fuzzy extractor provides inadequate security and efficiency. The core of our technical contribution is:

1. Modifying sample-then-lock for implementation (and proving security),
2. Modifying the iris image processing to maximize security,
3. Two open source implementations of the resulting scheme (Python and C),
4. Statistical analysis on security, correctness, storage requirements, and timing. All of our analysis uses the ND-0405 iris data set [56,12] which is a superset of the NIST Iris Challenge Evaluation Dataset [55]. Throughout our work we explicitly state what assumptions are needed for security of the scheme to hold.

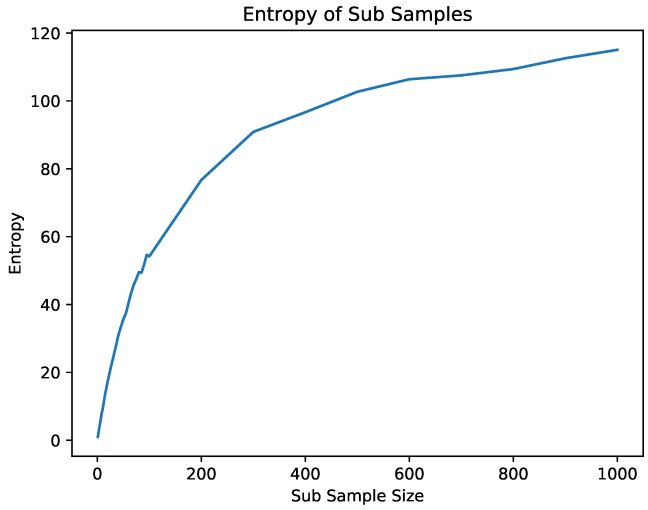

Fig. 1: In the worst case subsampling only preserves error rate. For the iris, subsampling greatly increases entropy rate from 1% to over 80%.

Adding more factors Many multi factor authentication systems do not achieve "additive security." Consider a strawman authentication system: 1) a user inputs a password and 2) an iris. Currently, the password would be hashed and compared to stored hash and the iris compared to a template. One of these comparisons has to be done first. Using a timing channel, it is possible for an adversary to separate search on each factor. A noiseless password can be used as part of the input to any fuzzy extractor and strengthen key derivation. However, previous fuzzy extractors separate the error-correction from the key derivation process using two distinct primitives called secure sketch [29] and randomness extractor respectively [51]. In such a process, the password can only be incorporated into the randomness extractor. As such, a similar timing channel may still exist. Our construction does not suffer from this problem. "Error-correction" and key derivation are performed simultaneously. The password can be prepended as input to each hash invocation without affecting storage or computational requirements.

1.1 Prior work

Boyen [15] defined reusable fuzzy extractors in 2004 and showed information-theoretic reusability requires a large decrease in security [15, Theorem 11].[5] Essentially, Boyen showed the security loss described above is *necessary*. Applied works showed that many fuzzy extractors were not reusable [58,6,7], meaning that the negative result of Boyen was not only a theoretic issue. Recent work [2,63,1,20] that provides computational security [33] sidesteps Boyen's negative result.

[5] The actual result of Boyen applies to *secure sketches* which imply fuzzy extractors. A secure sketch is a frequently used tool to construct a fuzzy extractor.

A key consideration in reusable fuzzy extractors is the type of correlation assumed between enrollments W_i and W_j. In many constructions it is assumed that $W_i \oplus W_j$ does not leak information about W_i or W_j. This assumption has not been verified in practice and was made by [15,2,63]. We make no assumption about the correlation between different enrollments, only about the statistical structure of an individual enrollment. In this model, Alamelou et al. [1] construct a reusable fuzzy extractor for the set difference metric. The iris has noise in the Hamming metric. The only construction that appears viable is the sample-then-lock construction (our starting point) [20].

Many previous works have used a fuzzy extractor in combination with the iris. These works security claims are troubling for a variety of reasons.

Hao et al. [40] use the code-offset construction with a code with 2^{140} codewords with the Iriscode transform of Daugman [24]. Using standard fuzzy extractor analysis this provides no security: it could leak as much as $2048 - 140$ bits. Hao et al. claim a key strength of 140 without justification. Hao et al. then argue an adversary providing a random iris would succeed with probability 2^{-44}. This corresponds to an adversary that does not have access to Pub (plaintext template storage suffices in this model). Note that providing an adversary providing a random iris should yields an upper bounds for the security of the system, immediately contradicting the claim of 140 bits of security. These issues have been communicated with Hao et al.

Bringer et al. [17] do not state a key strength but they report a nonzero false accept rate which implies a small effective key strength. Reporting a nonzero false accept rate is common in iris key derivation despite claimed key lengths > 40 bits (see discussion [53,43]). Using the birthday bound, false acceptances should appear when the tested dataset size approaches the square root of the claimed key size (i.e. $> 2^{20}$). No published iris datasets have close to a million individuals.

Kanade et al. [46] claim a fuzzy extractor construction but they report the entropy of the iris as over 1000 bits, much higher than other estimates. Other research states that each bit of the iris transform is independent [39] which is demonstrably not true (see for example our statistical analysis in Section 4).

The above discussion is necessarily incomplete (see the survey of Bowyer et al. [14, Section 6]). It demonstrates a large gap between theoretical fuzzy extractor constructions and their use, justifying a rigorous analysis of iris key derivation that makes assumptions explicit and accurately estimates security.

Recently, Cheon et al. [22] also modified sample-then-lock. However, their work contains a flaw in its security argument. At a high level, the authors incorrect argue that many polynomial size random oracles can't be exhausted by an unbounded adversary. This flaw has been communicated to and acknowledged by the authors. No public revision has been made.

Organization The rest of this work is organized as follows, in Section 2 we review basic definitions and cryptographic tools, Section 3 describes our scheme and software, Section 4 describes iris image processing and the transform used as input to our scheme, Section 5 evaluates the performance and correctness of our

system, Section 6 concludes. More statistical analysis and a second version of the scheme requiring additional statistical assumptions are deferred to the full version of this work [35].

2 Definitions and Cryptographic Tools

We use capital letters to refer to random variables. For a set of indices J, X_J is the restriction of X to the indices in J. U_n denotes the uniformly distributed random variable on $\{0,1\}^n$. Logarithms are base 2. The *min-entropy* of X is $\mathrm{H}_\infty(X) = -\log(\max_x \Pr[X = x])$. We use the notion of average min-entropy to measure the conditional entropy of a random variable. The *average* min-entropy of X given Y is $\tilde{\mathrm{H}}_\infty(X|Y) = -\log\left(\mathbb{E}_{y\in Y} \max_x \Pr[X = x|Y = y]\right)$.The *statistical distance* between random variables X and Y with the same domain is $\Delta(X,Y) = \frac{1}{2}\sum_x |\Pr[X = x] - \Pr[Y = x]|$. Our construction requires additional structure past entropy that we call k entropy samples [20]:

Definition 1. *Let $W = W_1, \ldots, W_n$ be a distribution over $\{0,1\}^n$. For k, α, we say that W is a source with* α-entropy k-samples *if $\tilde{\mathrm{H}}_\infty(W_{j_1}, \ldots, W_{j_k} \mid j_1, \ldots, j_k) \geq \alpha$ for uniformly random $1 \leq j_1, \ldots, j_k \leq n$.*

We use the version of fuzzy extractors that provides security against computationally bounded adversaries [33]. Dodis et al. provide comparable definitions for information-theoretic fuzzy extractors [29, Sections 2.5–4.1]. A desirable property of a fuzzy extractor is that an individual can enroll their biometric with multiple service providers and retain security.[6] Informally, each cryptographic key should be secure if an adversary knows all public helper values and all other derived keys. We state the definition of a reusable computational fuzzy extractor:

Definition 2. *Let $\mathcal{W}$ be a family of distributions over metric space $(\mathcal{M}, \mathsf{dis})$. A pair of randomized procedures "generate" (Gen) and "reproduce" (Rep) is an* $(\mathcal{M}, \mathcal{W}, \kappa, t, \rho)$-computational fuzzy extractor *that is $(\epsilon_{sec}, s_{sec})$-hard with error δ if Gen and Rep satisfy the following properties:*
1) Correctness: *if $\mathsf{dis}(w, w') \leq t$ and $(r, p) \leftarrow \mathsf{Gen}(w)$, $\Pr[\mathsf{Rep}(w', p) = r] \geq 1 - \delta$.*
2) Security *Let $(W^1, W^2, \ldots, W^\rho)$ be ρ correlated random variables such that each $W^j \in \mathcal{W}$. Let D be an adversary. Define the following game for $j = 1, ..., \rho$:*

- ***Sampling*** *The challenger samples $w^j \leftarrow W^j$, $u \leftarrow \{0,1\}^\kappa$.*
- ***Generation*** *The challenger computes $(r^j, p^j) \leftarrow \mathsf{Gen}(w^j)$.*
- ***Distinguishing*** *For all D of size at most s_{sec}, the advantage of D is*

$$\Pr[D(r^j, \{r^i\}_{i\neq j}^{i=1,\ldots,\rho}, \{p^i\}_{i=1}^{\rho}) = 1] - \Pr[D(u, \{r^i\}_{i\neq j}^{i=1,\ldots,\rho}, \{p^i\}_{i=1}^{\rho}) = 1] \leq \epsilon.$$

[6] Unlinkability prevents an adversary from telling if two enrollments correspond to the same physical source [21,47]. Our construction satisfies unlinkability (assuming security of the underlying cryptographic tools).

Digital Lockers Our construction uses digital lockers [19]. A digital locker is an algorithm lock which takes an input val and an output key, producing an algorithm unlock, unlock reproduces key if and only if the same val is provided as input. Digital lockers have two important properties:

1. Information about key is only obtained if the combination is guessed.
2. It is possible to detect the wrong val with high probability.

Digital lockers can be constructed from variants of the Diffie-Hellman assumption [19]. Let $\mathtt{HMAC}$ be HMAC-SHA256. Our construction assumes that $\mathtt{HMAC}$ can be used to construct digital lockers. The "locking" algorithm outputs the pair

$$\mathsf{nonce}, \mathtt{HMAC}(\mathsf{nonce}, w) \oplus (0^{128} || \mathsf{key}),$$

where nonce is a nonce, $||$ denotes concatenation, 0^{128} is the all zeros string of length 128, a security parameter. Unlocking proceeds by recomputing the hash and checking for a prefix of 0^{128}. If this prefix is found then the suffix key' is output. This construction was proposed in [3] and shown to be secure in the random oracle model by Lynn, Prabhakaran, and Sahai [50, Section 4]. It is plausible that in the standard model (without random oracles) hash functions provide the necessary security [19, Section 3.2], [23, Section 8.2.3]. We now present the full formal definition [5]:

Definition 3. *The pair of algorithm* $(\mathsf{lock}, \mathsf{unlock})$ *with security parameter* λ *is an* ℓ*-composable secure digital locker with error* γ *if the following hold:*
Correctness *For any pair* $\mathsf{key}, \mathsf{val}$, $\Pr[\mathsf{unlock}(\mathsf{key}, \mathsf{lock}(\mathsf{key}, \mathsf{val})) = \mathsf{val}] \geq 1 - \gamma$. *Also, for any* $\mathsf{key}' \neq \mathsf{key}$, $\Pr[\mathsf{unlock}(\mathsf{key}', \mathsf{lock}(\mathsf{key}, \mathsf{val})) = \perp] \geq 1 - \gamma$.
Security *For every PPT adversary* A *and every positive polynomial* p*, there exists a (possibly inefficient) simulator* S *and a polynomial* $q(\lambda)$ *such that for any sufficiently large* s*, any polynomially-long sequence of values* $(\mathsf{val}_i, \mathsf{key}_i)$ *for* $i = 1, \ldots, \ell$*, and any auxiliary input* $z \in \{0,1\}^*$*,*

$$\left| \Pr\left[A\left(z, \{\mathsf{lock}\left(\mathsf{key}_i, \mathsf{val}_i\right)\}_{i=1}^{\ell}\right) = 1 \right] - \Pr\left[S\left(z, \{|\mathsf{key}_i|, |\mathsf{val}_i|\}_{i=1}^{\ell}\right) = 1 \right] \right| \leq \frac{1}{p(\mathsf{s})}$$

where S *is allowed* $q(\lambda)$ *oracle queries to the oracles* $\{\mathsf{idealUnlock}(\mathsf{key}_i, \mathsf{val}_i)\}_{i=1}^{\ell}$.

Technical Remark: Unfortunately, the security definition of digital lockers (Definition 3) is "inherently" asymptotic. A different simulator is allowed for each distance bound $p(s)$ making it difficult to argue what quality key is provided with respect to a particular adversary.

3 Our Construction & Implementation

Our construction builds on the construction of Canetti et al. [20]. The high level idea is to encrypt the same key multiple times using different subsets of w. Pseudocode for the algorithm is in Figure 2.

$\mathsf{Gen}(w)$:

1. Sample random 128 bit key.
2. For $i = 1, ..., \ell$:
   (i) Choose $1 \leq j_{i,1}, ..., j_{i,k} \leq |w|$
   (ii) Choose 512 bit hash key h_i.
   (iii) $c_i = \mathtt{Hash}(h_i, w_{j_{i,1}}, ..., w_{j_{i,k}})$.
   (iv) Set $p_i = (0^{128} || \mathsf{key}) \oplus c_i$.
3. Output $(\mathsf{key}, p_i, \{j_{i,m}\}, h_i)$.

$\mathsf{Rep}(w', p_i, \{j_{i,m}\}, h_i)$

1. For $i = 1, ..., \ell$:
   (i) $c_i = \mathtt{Hash}(h_i, w'_{j_{i,1}}, ..., w'_{j_{i,k}})$.
   (ii) If $(c_i \oplus p_i)_{1..128} = 0^{128}$,
   output $(c_i \oplus p_i)_{129..256}$.
2. Output $\perp$.

Fig. 2: Overview of generation (enrollment) and reproduction (authentication) of key derivation system.

In the description above, $x_{a..b}$ denotes the restriction of a vector to the bits between a and b. The parameters k and ℓ represent a tradeoff between correctness and security. For the scheme to be correct at least one of the ℓ subsets should have no error with high probability. Canetti et al. show it is possible to set ℓ if the expected error rate is sublinear in $|w|$. That is, when $d(w, w')/|w| = o(|w|)$. We set ℓ and k in Section 4.

A single digital locker requires storage of 32 bytes for the output of the hash and 64 bytes for each hash key h_i. In addition, the public value must store the randomly sampled locations. The two natural solutions for this are 1) storing a mask of size $|w|$ for each subset or 2) a location set of size $\log |w| * k$ for each subset. Using either approach, in our analysis, storing subsets required more space that the hash outputs and keys. This led to our main modification of the cryptographic scheme.

Canetti et al. [20, Section 4] note that rather than using independent subsets they could be selected using a sampler [37]. We show the security argument holds as long as each subset is random on its own. That is, the different subsets can be arbitrarily correlated. We will use this fact to reduce the storage requirement of the scheme. We now state security of the modified scheme.

Theorem 1. *Let λ be a security parameter, Let $\mathcal{W}$ be a family with α-entropy k-samples for $\alpha = \omega(\log \lambda)$. Suppose the HMAC construction is a secure digital locker. Let $\mathcal{I}_j$ be the jth subset generated in* Gen*. The fuzzy extractor in Fig. 2 is secure if each individual $\mathcal{I}_j$ is uniformly distributed (but different subsets $\mathcal{I}_j, \mathcal{I}_\ell$ are potentially correlated). More formally, for any $s_{sec} = \mathtt{poly}(\lambda)$ there exists some $\epsilon_{sec} = \mathtt{ngl}(\lambda)$ such that* sample-then-lock *is a $(\mathcal{Z}^n, \mathcal{W}, \kappa, t)$-computational fuzzy extractor that is $(\epsilon_{sec}, s_{sec})$-hard with error $\delta = \mathtt{negl}(\lambda)$. No claim about correctness is made if $\mathcal{I}_j$ and $\mathcal{I}_\ell$ are correlated.*

We only show security when Gen is run once, reusability follows using the same argument as in Canetti et al. [20].

Proof. Let $V_1, ..., V_\ell$ be random variables corresponding to W restricted to the bits selected in subset $\mathcal{I}_i$. Similarly, let P_i be the random variable corresponding to the public part of the output produced in iteration i. Let R denote the distribution over output key values. Lastly, let U denote the uniform distribution

over $\{0,1\}^{|\mathsf{key}|}$. We show for all $s_{sec} = \mathtt{poly}(\lambda)$ there exists $\epsilon_{sec} = \mathtt{ngl}(\lambda)$ such that

$$\delta^{\mathcal{D}_{s_{sec}}}((R, \{P_i\}_{i=1}^{\ell}), (U, \{P_i\}_{i=1}^{\ell})) \leq \epsilon_{sec}.$$

Fix some polynomial s_{sec} and let D be a distinguisher of size at most s_{sec}.

We proceed by contradiction: supposing $|\mathbb{E}[D(R, \{P_i\}_{i=1}^{\ell})] - \mathbb{E}[D(U, \{P_i\}_{i=1}^{\ell})]|$ is not negligible. Suppose there is a polynomial $p(\cdot)$ such that for all λ_0 there exists some $\lambda > \lambda_0$ such that

$$|\mathbb{E}[D(R, \{P_i\}_{i=1}^{\ell})] - \mathbb{E}[D(U, \{P_i\}_{i=1}^{\ell})]| > 1/p(\lambda).$$

By Definition 3, there is a polynomial q and an unbounded time simulator S (making at most $q(\lambda)$ queries to the oracles $\{\mathsf{idealUnlock}(v_i, r)\}_{i=1}^{\ell}$) such that

$$\frac{1}{3p(\lambda)} \geq |\mathbb{E}[D(R, P_1, ..., P_\ell)] - \mathbb{E}\left[S^{\{\mathsf{idealUnlock}(v_i,r)\}_{i=1}^{\ell}}\left(R, \{\mathcal{I}_i\}_{i=1}^{\ell}, k, |\mathsf{key}|\right)\right]| \tag{1}$$

This is also true if we replace R with an independent uniform random variable U over $\{0,1\}^{|\mathsf{key}|}$. We now prove the following lemma, which shows that S cannot distinguish between R and a independent U.

Lemma 1. *Let all variables be as above. Then*

$$\begin{aligned}&\Big|\mathbb{E}\left[S^{\{\mathsf{idealUnlock}(v_i,r)\}_{i=1}^{\ell}}\left(R, \{\mathcal{I}_i\}_{i=1}^{\ell}, k, |\mathsf{key}|\right)\right] \\ &-\mathbb{E}\left[S^{\{\mathsf{idealUnlock}(v_i,r)\}_{i=1}^{\ell}}\left(U, \{\mathcal{I}_i\}_{i=1}^{\ell}, k, |\mathsf{key}|\right)\right]\Big| \\ &\quad\leq \frac{q(q+1)}{2^{\alpha}} \leq \frac{1}{3p(\lambda)}\end{aligned}$$

where q is the maximum number of queries S can make.

Proof. Fix some $u \in \{0,1\}^{|\mathsf{key}|}$. The only information about whether the value is r or u can obtained by S through the query responses. First, modify S slightly to quit immediately if it gets a response not equal to $\perp$. There are $q+1$ possible values for the view of S on a given input (q of those views consist of some number of $\perp$ responses followed by the first non-$\perp$ response, and one view has all q responses equal to $\perp$). By [29, Lemma 2.2b], $\tilde{\mathrm{H}}_{\infty}(V_i|View(S), \{\mathcal{I}_j\}) \geq \tilde{\mathrm{H}}_{\infty}(V_i|\{\mathcal{I}_j\}) - \log(q+1) \geq \alpha - \log(q+1)$. Therefore, at each query, the probability that S gets a non-$\perp$ answer (equivalently, guesses V_i) is at most $(q+1)2^{-\alpha}$. Since there are q queries of S, the overall probability is at most $q(q+1)/2^{\alpha}$. Then since 2^{α} is $\mathtt{ngl}(\lambda)$, there exists some λ_0 such that for all $\lambda > \lambda_0$, $q(q+1)/2^{\alpha} \leq 1/(3p(\lambda))$.

The overall theorem follows using the triangle inequality with equation 1, equation 1 with R replaced with U, and Lemma 1 yielding $\delta^{D}((R, P), (U, P)) \leq 1/p(\lambda)$. This completes the proof of Theorem 1.

This theorem gives us a mechanism for saving on storage size. Instead of choosing independent subsets, the implementation chooses a master subset and then generates permutations π_j to create new subsets based on public cryptographic keying material. The new scheme works as follows:

1. Choose a master subset $\mathcal{I}$ uniformly at random where $|\mathcal{I}| = k$.
2. For each locker j generate a permutation $\pi_j : \{0,1\}^{|w|} \rightarrow \{0,1\}^{|w|}$.
3. Apply π_j to each element of $\mathcal{I}$ to get $\mathcal{I}_j$.

To efficiently generate permutations we do the following:

1. Select a single master CHACHA20 key
2. Encrypt the permutation number j, creating $\log|w| * |w|$ bits of output c.
3. We split c into $\log|w|$ bit sections $c_1, ..., c_{|w|}$.
4. Define $\pi_j(i) = c_i$

The output of CHACHA20 is not a permutation: it is not guaranteed that $\log|w|$ consecutive bits do not repeat. Furthermore, looking ahead to Section 4, our iris processing results in a vector of 12000 bits. The above algorithm only works if $|w|$ is a power of 2. We adapt our algorithm by adding a check for each section c_i. If $c_i > |w|$ or c_i is repeated it is discarded. To compensate for these two failure conditions it is necessary to produce more than $|w|$ sections. Producing 2000 additional sections was sufficient to always output a permutation in our experiments. This modification reduces overall storage to a single CHACHA20 key, the single randomly generated subset, and 96 byte per subset storage. Generating these permutations takes additional computation. One can tradeoff between storing all subsets and a single master subset, storing some fraction of subsets and regenerating the rest. We are not aware of how to reduce the 96 byte per subset storage. An idea is to use a single nonce, we were not able to argue security of this modified scheme. We leave this as an open problem.

Implementation We implemented our construction in both Python and C and both implementations are open sourced [36]. Previous implementations of fuzzy extractors required expertise in error-correcting codes. Our construction only requires repeated evaluation of a hash function.

The entire Python library is 100 lines of code with dependencies on numPy (for array manipulation), random, and hashlib. Our Gen code is single threaded because the majority of execution time is spent generating the subsets $j_{i,1}, ..., j_{i,k}$. The Rep functionality is embarrassingly parallel. We implemented a parallel version that simply partitions the hashes to be performed. Rep succeeds when one of these threads returns. Unfortunately, neither implementation is fast enough with authentication taking seconds (see Section 5).

We also developed an optimized C implementation designed for fast Rep performance. As Rep is used at every authentication its speed is more important than Gen which is only used when a user enrolls with a new service. For this implementation we used Libsodium [4] as the cryptographic backend and HMAC-SHA-512 to instantiate the digital locker. This library makes use of low level

bit level operations for quickly packing and selecting bits the iris vector. In preliminary testing a major obstacle to fast Rep was disk load time. Recall, each subset selected in Gen requires storage of 96 bytes.

4 Iris Image Processing and Setting Parameters

This section provides a brief overview of iris image processing and the transform used in our system. Iris image processing is an entire field [13]. Our scheme can be used with techniques that produce a vector with Hamming errors (fraction of symbols that are the same).

The starting point for our transform is open-source OSIRIS package [49]. This package is open source and uses representative techniques. OSIRIS takes a near infrared iris image and produces a 32768 bit vector w. The stages of OSIRIS are:

1. Iris and Pupil Localization: This step finds the inner and outer boundaries of the iris accounting for pupil dilatation and occlusions.
2. Iris Unwrapping: The iris is converted into a 2D matrix. This array is indexed by (r, θ) which is the polar position of the pixel in the original image.
3. Featurization: 2D Gabor filters [38] centered at different positions are convolved with the image yielding a complex values at locations (r, θ). This produces a 64×512 vector of complex valued numbers.
4. Binarization: Complex numbers are quantized based on sign to produce two bits.

The OSIRIS library includes six transforms. These transforms are the real and imaginary components of three different sets of Gabor filters. Our experiments showed the histogram with the lowest error rate (for images of the same iris) was Transform 5. We thus used Transform 5 for all of our analysis.

Daugman [24] reports mean error rates of 11%, but we are unaware of any subsequent work that an error rate that lowachieves as low an error rate as 11%.[7]

All of our statistical analysis is performed using the ND-0405 dataset [12] which is a superset of the NIST Iris Challenge Evaluation Dataset [55]. The ND-0405 dataset includes 356 persons and 64964 total images. We observe a mean error rate of 32% using the ND-0405 Iris data set [56,12].

Our analysis includes *intraclass* comparisons which are comparisons of the Hamming distance between two transformed images of the same iris and *interclass* comparisons which are comparisons of the Hamming distance between two transformed images of different irises. The ND-0405 dataset contains images from the left and right eye of the same individual. These are treated as interclass comparisons.

Figure 3 shows the histograms for fractional Hamming distance between two images of the same individual (*same*) and different individuals (*different*) for

[7] The security/correctness tradeoff of our system immediately improves with an iris transform with lower error rate.

the dataset. This histogram is produced by computing the fractional Hamming distance of every iris with every other iris (for a total of $\approx 10^9$ comparisons). The fractional Hamming distances were then grouped into *interclass/different* comparisons corresponding to the same iris and *intraclass/same* comparisons corresponding to different irises. The error rate of the data is defined as the expected fractional Hamming distance between two images of the same iris. For intraclass comparisons we observed a mean error rate of .32. For different irises, we observe the interclass mean and interclass variance as $\mu = .494$ and $\sigma = .0008$.

The standard method for estimating the entropy of the iris [24] is to compare the interclass histogram with a Binomial distribution with the same mean μ and variance σ. If the observed distribution and the Binomial distribution have very similar histograms, then the observed distribution is assumed to have the same entropy as the Binomial distribution. This technique is necessarily a heuristic.

We computed this heuristic generating a binomial distribution with mean $\mu = .494$ and variance $\sigma = .0008$. The statistical distance between the interclass histogram and the binomial distribution was computed with a total statistical distance of .005. We use the entropy of the Binomial distribution as a stand in for the entropy of the observed distribution. The entropy of the Binomial is calculated using the following equations (where dF stands for degrees of freedom):

$$\begin{aligned} \mathrm{dF} &= \frac{\mu(1-\mu)}{\sigma} = 311 \\ \text{entropy} &= (-\mu \log \mu - (1-\mu)\log(1-\mu)) * \mathrm{dF} \\ &= 311. \end{aligned}$$

Our entropy estimate is different from Daugman's. It is common for this estimate to vary across data sets, this estimate is capturing useful information of the underlying biologic process and noise which is less useful. However, since the construction has to "correct" the noise, the noise should also be counted for security.

Entropy of Subsamples Our security theorem requires not only overall entropy, but entropy of random subsets (see Definition 1 and Theorem 1). In the worst case, sampling only preserves the entropy rate of a distribution which for OSIRIS is $311/32768 \approx 1\%$.

Iris entropy is believed to be geographically distributed throughout the iris. The OSIRIS output is produced by convolving a fixed Gabor filter at overlapping regions of the unwrapped iris. So one would expect nearby bits to be correlated. If only nearby bits are correlated, subsampling random bits will increase the entropy rate. To test this hypothesis, we performed the following analysis (for subset size k) with 10 trials for each subset size:

1. Randomly sample k distinct positions.
2. Compute the intraclass and interclass histograms for the dataset restricted to these positions.
3. Compute the μ and σ for the interclass histogram. (Using the same method as in Figure 3).

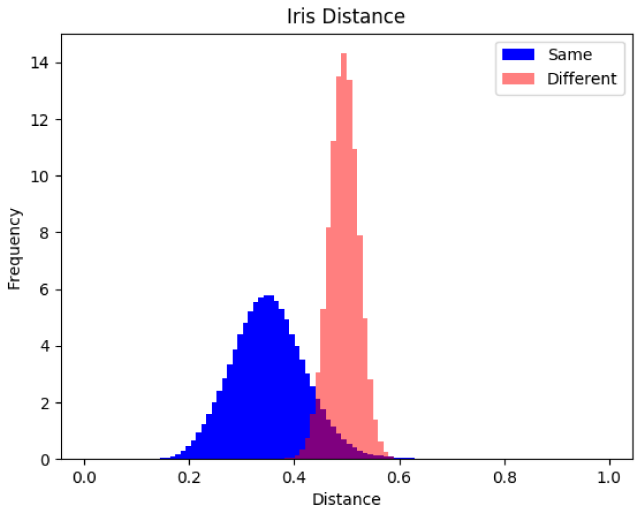

Fig. 3: Distribution of distances for the ND-0405 data set.

4. Estimate the entropy e_i for trial i.
5. Compute the average min-entropy as $e = -\log \mathbb{E}_i\, 2^{-e_i}$.

In this last step we average the entropy calculation using average min-entropy (Section 2). This technique is preferable to averaging the entropies e_i. We are targeting security which requires that the entropy should be high in all settings, not just on average. Consider five possible events where the entropy conditioned on the events is $1, 100, 100, 100, 100$ respectively. Then the "average entropy" is ≈ 80 while the average min-entropy is ≈ 3. However, clearly in this situation the individual with an entropy of 1 is in trouble. We find the average of entropies and the *average min-entropy* differ substantially. The average min-entropy heavily weights low entropy trials.

This analysis was performed for subset sizes $k \in \{1, 2, ..., 10\} \cup \{15, 20, ..., 100\} \cup \{200, 300, ..., 1000\}$ with 10 trials for each size.

Since we are randomly subsampling from a distribution that fits the binomial well, the distribution was assumed to also fit a binomial distribution. The figure is in the Introduction in Figure 1. We note that the entropy rate is significantly higher than the worst case of 1%. At some points in Figure 1 the entropy rate exceeds 80%.

4.1 Choosing reference parameters

In this subsection, we define some reference parameters for an instantiation of the scheme. For our construction the two tunable parameters are the number of subsets, ℓ, and the number of bits in each subset, k. Increasing k improves

security but hurts correctness, increasing ℓ improves correctness but costs time and storage. The two parameters are related by

$$1 - (1 - (1 - \text{error rate})^k)^\ell = \Pr[\text{correct}]. \tag{2}$$

We will set the number of lockers $\ell = 10^6$. This results in storage of approximately 100 MB which is dominated by the per locker storage of the HMAC key and output. We assume a *correctness target* of 50% true positive rate. While this number is unacceptable for an authentication system, correctness rate is an "s-curve" in error rate. Correctness increases quickly once it hits 50%, achieving correctness of $1-2^{-x}$ for some x requires multiplicatively increasing the number of lockers by 2^{x-1}. So 93.75% correctness requires 8×10^6 lockers (roughly 800MB of storage). We note that in many biometric authentication settings, the sensor can rapidly collect multiple images, allowing multiple chances to authenticate. We consider these parameters fixed.

Optimizing the transform A technique commonly used to improve iris transforms is called *masking*. (Bowyer et al. survey iris processing techniques [13].) In most iris transforms in addition to the binary vector w the transform additionally outputs a second vector $mask$. Bits set in $mask$ indicate an error in the transform perhaps due to an eyelash or eyelid (known as an occlusion). Rather than comparing the Hamming distance $d(w, w')$, the authentication only compares locations i where $mask_i = 0 = mask'_i$. The intuition behind the mask vector is that occluded locations are expected to have higher error rates and should be ignored.

A possible way to incorporate $mask$ into *sample-then-lock* is to only sample from positions that are not masked. This technique limits "comparison" to locations where $mask_i = 0$. However, $mask$ may be correlated to the underlying values w, so choosing subsets in this way may leak information to the attacker. Locations to be masked are not uniformly distributed throughout the iris. Rather masked bits usually occur on the top, bottom, inside and outside of the iris [42].

Instead, we will restrict the 32768 bit vector to locations that are unlikely to be masked across the dataset. We denote by pr_{mask} the vector of mask probabilities for each bits. To find the right restriction we did the following for a threshold $thres \in \{1, .0975, .095, .0925, ..., .05, .025\} \cup \{.015\}$.

1. Restrict the input locations to positions j where $pr_{mask,j} > thres$.
2. Compute the mean error rate restricted to these bits.
3. Compute the maximum subset size k such that $\Pr[\text{correct}] \geq .5$. (see Equation 2).
4. Repeat 10 times:
   (a) Sample k random bits $\mathcal{I}$ from locations where $pr_{mask,j} > thres$.
   (b) Restrict the input dataset to locations in $\mathcal{I}$. Compute interclass histogram across the entire dataset.
   (c) Compute $\mu_{thres,i}, \sigma_{thres,i}$ for trial i.
   (d) Compute the entropy $e_{thres,i}$ for trial i.
5. Compute the overall entropy as $e_{thres} = -\log \mathbb{E}_i\, 2^{-e_{thres,i}}$

Pr of mask	Number of Bits	Subsample Size	Entropy
1	32768	32	28
0.9	31810	33	29
0.8	31256	33	29
0.7	30528	33	29
0.6	29455	34	29
0.5	27910	34	30
0.4	26115	35	30
0.3	23861	37	32
0.2	20109	39	31
0.1	15953	41	33
0.075	14572	42	32
0.05	12718	43	32
0.025	9661	44	30
0.015	7619	45	30

Table 1: Average min-entropy of input subset to *sample-then-lock* when restricting to bits that are unlikely to be masked.

A subset of this analysis is in Table 1. This analysis has a minimum entropy, 28, and subset size, 32, when all bits are likely to be included. The entropy is maximized at 33, while subset size is maximized at 45. In the full version, we compare this approach with restricted to bits that demonstrate the highest error rate [35]. Both approaches result in similar parameters. We include the 12000 bits that are least likely to be masked as our "iris transform." This was the size that allowed the highest subset size where entropy was close to the maximum.

5 Evaluation

In this section we evaluate the running time and correctness of our system. The basis of our security argument is Theorem 1 and Table 1 which give a necessary condition for security and the estimated entropy of subsets being used in our construction respectively.

This performance analysis was performed on a Dell Precision Tower 7000 Series with 4 Xeon E5-2620 v4 processors and 64GB of RAM. The computation was parallelism bound.

We report performance numbers for both the Python and C implementations. In the Python implementation Gen takes 220s. We implemented a parallel version of Rep which takes 12s. Since Rep must be performed on every authentication this is not fast enough for most use cases. These performance numbers do not include disk read time, which was greater than the computation time.

For the C implementation, we consider the speed of three different operations, Gen, Rep and subset generation. We do not include time for subset generation in Gen and Rep. Furthermore, we do not include disk read time. The reported times for Rep assumes the data structure is already in memory. Depending on the use case the data structure for Pub may be stored in memory, on disk, or regenerated as needed. Importantly, subset generation is independent of the iris

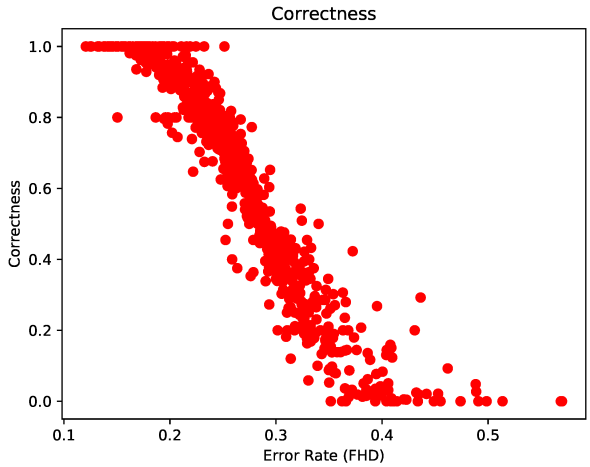

Fig. 4: Correlation between correctness of Rep and the error rate of an individual's eye. The mean error rate across the corpus is 60%. Based on subset size of 43 bits with the 12000 bits that have probability of being masked of at less than 5%.

value and can be performed ahead of time (e.g., prior to an employee starting their shift).

On across 100 runs, on average, the parallel hash computation in Gen takes .57s, the parallel hash computation in Rep takes .57s, and subset generation takes 12.79s. (Standard deviations of .48s, .48s, and .51s respectively.)

We tested correctness across the data corpus with our Python implementation. Our implementation was tested with these parameters: 1) starting with 12000 bits that are unlikely to be masked 2) using a subset size of 43 bits. Specifically, Gen was run based on the first alphanumeric image from a particular iris, followed by Rep on all other images of that iris. Our target correctness was 50% across the corpus. Our observed mean correctness was higher at 60%. As expected correctness is highly correlated with the error rate of the underlying iris. This correlation is demonstrated in Figure 4.

6 Conclusion

We described the first key derivation system from the human iris that provides meaningful albeit modest security. The system allows enrollment with multiple devices and services. Security rests on clear cryptographic assumptions (Theorem 1) and statistical properties (Table 1).

Our system uses repeated evaluation of cryptographic hashes. There are many promising memory hard hash functions (designed to defeat parallel GPU cracking) such as scrypt [54] and argon2i [44]. Using these constructions in sample-then-lock is nontrivial as the hash function must be computed many times by

the system. Ideally, one could use a hash function that is easy to compute in parallel with fast access to a large memory but hard for GPUs. We are unaware of any such candidates.

Biometric authentication is a fact of life. This work explores how secure cryptographic techniques can be made for real biometrics. While our system does not achieve "cryptographic" security levels, we believe they are in reach. We hope this work encourages further research into the iris and other biometric modalities. Lastly, porting to mobile platforms is a natural goal. We believe satisfactory performance on mobile devices requires new cryptographic and architectural techniques. We leave this as future work.

Acknowledgements We thank the anonymous reviews for their helpful suggestions and comments. Mariem Ouni and Tyler Cromwell contributed to software described in this work. We thank Leonid Reyzin and Alexander Russell for helpful discussions and insights. This work was supported in part through a grant with Comcast Inc. Work of S. Simhadri was done while at University of Connecticut.

References

1. Q. Alamélou, P.-E. Berthier, C. Cachet, S. Cauchie, B. Fuller, P. Gaborit, and S. Simhadri. Pseudoentropic isometries: A new framework for fuzzy extractor reusability. In *AsiaCCS*, 2018.
2. D. Apon, C. Cho, K. Eldefrawy, and J. Katz. Efficient, reusable fuzzy extractors from LWE. In *International Conference on Cyber Security Cryptography and Machine Learning*, pages 1–18. Springer, 2017.
3. M. Bellare and P. Rogaway. Random oracles are practical: A paradigm for designing efficient protocols. In *ACM Conference on Computer and Communications Security (CCS)*, pages 62–73, 1993.
4. D. J. Bernstein, T. Lange, and P. Schwabe. The security impact of a new cryptographic library. In *International Conference on Cryptology and Information Security in Latin America*, pages 159–176. Springer, 2012.
5. N. Bitansky and R. Canetti. On strong simulation and composable point obfuscation. In *Advances in Cryptology–CRYPTO 2010*, pages 520–537. Springer, 2010.
6. M. Blanton and M. Aliasgari. On the (non-) reusability of fuzzy sketches and extractors and security improvements in the computational setting. *IACR Cryptology ePrint Archive*, 2012:608, 2012.
7. M. Blanton and M. Aliasgari. Analysis of reusability of secure sketches and fuzzy extractors. *IEEE transactions on information forensics and security*, 8(9-10):1433–1445, 2013.
8. M. Blanton and P. Gasti. Secure and efficient protocols for iris and fingerprint identification. In *European Symposium on Research in Computer Security*, pages 190–209. Springer, 2011.
9. C. Blundo, E. De Cristofaro, and P. Gasti. EsPRESSo: efficient privacy-preserving evaluation of sample set similarity. In *Data Privacy Management and Autonomous Spontaneous Security*, pages 89–103. Springer, 2013.
10. J. Bonneau. The science of guessing: analyzing an anonymized corpus of 70 million passwords. In *2012 IEEE Symposium on Security and Privacy*, pages 538–552. IEEE, 2012.

11. J. Bonneau, C. Herley, P. C. Van Oorschot, and F. Stajano. The quest to replace passwords: A framework for comparative evaluation of web authentication schemes. In *IEEE Symposium on Security and Privacy*, pages 553–567. IEEE, 2012.
12. K. W. Bowyer and P. J. Flynn. The ND-IRIS-0405 iris image dataset. *arXiv preprint arXiv:1606.04853*, 2016.
13. K. W. Bowyer, K. Hollingsworth, and P. J. Flynn. Image understanding for iris biometrics: A survey. *Computer vision and image understanding*, 110(2):281–307, 2008.
14. K. W. Bowyer, K. P. Hollingsworth, and P. J. Flynn. A survey of iris biometrics research: 2008–2010. In *Handbook of iris recognition*, pages 15–54. Springer, 2013.
15. X. Boyen. Reusable cryptographic fuzzy extractors. In *Proceedings of the 11th ACM conference on Computer and communications security*, CCS '04, pages 82–91, New York, NY, USA, 2004. ACM.
16. X. Boyen, Y. Dodis, J. Katz, R. Ostrovsky, and A. Smith. Secure remote authentication using biometric data. In *EUROCRYPT*, pages 147–163. Springer, 2005.
17. J. Bringer, H. Chabanne, G. Cohen, B. Kindarji, and G. Zémor. Optimal iris fuzzy sketches. In *Biometrics: Theory, Applications, and Systems, 2007. BTAS 2007. First IEEE International Conference on*, pages 1–6. IEEE, 2007.
18. J. Bringer, H. Chabanne, and A. Patey. SHADE: Secure hamming distance computation from oblivious transfer. In *International Conference on Financial Cryptography and Data Security*, pages 164–176. Springer, 2013.
19. R. Canetti and R. R. Dakdouk. Obfuscating point functions with multibit output. In *Advances in Cryptology–EUROCRYPT 2008*, pages 489–508. Springer, 2008.
20. R. Canetti, B. Fuller, O. Paneth, L. Reyzin, and A. Smith. Reusable fuzzy extractors for low-entropy distributions. In *Advances in Cryptology–Eurocrypt 2016*, pages 117–146. Springer, 2016.
21. F. Carter and A. Stoianov. Implications of biometric encryption on wide spread use of biometrics. In *EBF Biometric Encryption Seminar (June 2008)*, 2008.
22. J. H. Cheon, J. Jeong, D. Kim, and J. 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.
23. R. R. Dakdouk. *Theory and Application of Extractable Functions*. PhD thesis, Yale University, 2009. http://www.cs.yale.edu/homes/jf/Ronny-thesis.pdf.
24. J. Daugman. How iris recognition works. *Circuits and Systems for Video Technology, IEEE Transactions on*, 14(1):21 – 30, January 2004.
25. J. Delvaux, D. Gu, I. Verbauwhede, M. Hiller, and M.-D. M. Yu. Efficient fuzzy extraction of PUF-induced secrets: Theory and applications. In *International Conference on Cryptographic Hardware and Embedded Systems (CHES)*, pages 412–431. Springer, 2016.
26. S. Deshmukh, H. Carter, G. Hernandez, P. Traynor, and K. Butler. Efficient and secure template blinding for biometric authentication. In *Communications and Network Security (CNS), 2016 IEEE Conference on*, pages 480–488. IEEE, 2016.
27. Y. Dodis, B. Kanukurthi, J. Katz, L. Reyzin, and A. Smith. Robust fuzzy extractors and authenticated key agreement from close secrets. *IEEE Transactions on Information Theory*, 58(9):6207–6222, 2012.
28. Y. Dodis, J. Katz, L. Reyzin, and A. Smith. Robust fuzzy extractors and authenticated key agreement from close secrets. In C. Dwork, editor, *Advances in Cryptology - CRYPTO 2006*, volume 4117 of *Lecture Notes in Computer Science*, pages 232–250. Springer Berlin Heidelberg, 2006.

29. Y. Dodis, R. Ostrovsky, L. Reyzin, and A. Smith. Fuzzy extractors: How to generate strong keys from biometrics and other noisy data. *SIAM Journal on Computing*, 38(1):97–139, 2008.
30. Y. Dodis, L. Reyzin, and A. Smith. Fuzzy extractors: How to generate strong keys from biometrics and other noisy data. In *Advances in Cryptology–Eurocrypt*, pages 523–540. Springer, 2004.
31. P.-A. Dupont, J. Hesse, D. Pointcheval, L. Reyzin, and S. Yakoubov. Fuzzy password-authenticated key exchange. In *Annual International Conference on the Theory and Applications of Cryptographic Techniques*, pages 393–424. Springer, 2018.
32. D. Evans, Y. Huang, J. Katz, and L. Malka. Efficient privacy-preserving biometric identification. In *Proceedings of the 17th conference Network and Distributed System Security Symposium, NDSS*, 2011.
33. B. Fuller, X. Meng, and L. Reyzin. Computational fuzzy extractors. In *Advances in Cryptology-ASIACRYPT 2013*, pages 174–193. Springer, 2013.
34. B. Fuller, L. Reyzin, and A. Smith. When are fuzzy extractors possible? In *International Conference on the Theory and Application of Cryptology and Information Security*, pages 277–306. Springer, 2016.
35. B. Fuller, S. Simhadri, and J. Steel. Reusable authentication from the iris. Cryptology ePrint Archive, Report 2017/1177, 2017. https://eprint.iacr.org/2017/1177.
36. B. Fuller, S. Simhadri, and J. Steel. Computational fuzzy extractors. https://github.com/benjaminfuller/CompFE, 2018.
37. O. Goldreich. A sample of samplers: A computational perspective on sampling. In *Studies in Complexity and Cryptography. Miscellanea on the Interplay between Randomness and Computation*, pages 302–332. Springer, 2011.
38. A. Grossmann and J. Morlet. Decomposition of Hardy functions into square integrable wavelets of constant shape. *SIAM journal on mathematical analysis*, 15(4):723–736, 1984.
39. Z. Guo, N. Karimian, M. M. Tehranipoor, and D. Forte. Hardware security meets biometrics for the age of IoT. In *Circuits and Systems (ISCAS), 2016 IEEE International Symposium on*, pages 1318–1321. IEEE, 2016.
40. F. Hao, R. Anderson, and J. Daugman. Combining crypto with biometrics effectively. *IEEE Transactions on Computers*, 55(9):1081–1088, 2006.
41. T. Holenstein and R. Renner. One-way secret-key agreement and applications to circuit polarization and immunization of public-key encryption. In V. Shoup, editor, *Advances in Cryptology - CRYPTO 2005: 25th Annual International Cryptology Conference, Santa Barbara, California, USA, August 14-18, 2005, Proceedings*, volume 3621 of *Lecture Notes in Computer Science*, pages 478–493. Springer, 2005.
42. K. P. Hollingsworth, K. W. Bowyer, and P. J. Flynn. The best bits in an iris code. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 31(6):964–973, 2009.
43. G. Itkis, V. Chandar, B. W. Fuller, J. P. Campbell, and R. K. Cunningham. Iris biometric security challenges and possible solutions: For your eyes only? using the iris as a key. *IEEE Signal Processing Magazine*, 32(5):42–53, 2015.
44. S. Josefsson. The memory-hard argon2 password hash function. *memory*, 2015.
45. A. Juels and M. Wattenberg. A fuzzy commitment scheme. In *Sixth ACM Conference on Computer and Communication Security*, pages 28–36. ACM, Nov. 1999.
46. S. Kanade, D. Camara, E. Krichen, D. Petrovska-Delacrétaz, and B. Dorizzi. Three factor scheme for biometric-based cryptographic key regeneration using iris. In *Biometrics Symposium, 2008. BSYM'08*, pages 59–64. IEEE, 2008.

47. E. J. Kelkboom, J. Breebaart, T. A. Kevenaar, I. Buhan, and R. N. Veldhuis. Preventing the decodability attack based cross-matching in a fuzzy commitment scheme. *Information Forensics and Security, IEEE Transactions on*, 6(1):107–121, 2011.
48. S. Komanduri, R. Shay, P. G. Kelley, M. L. Mazurek, L. Bauer, N. Christin, L. F. Cranor, and S. Egelman. Of passwords and people: measuring the effect of password-composition policies. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, pages 2595–2604. ACM, 2011.
49. E. Krichen, A. Mellakh, S. Salicetti, and B. Dorizzi. OSIRIS (open source for IRIS) reference system, 2017.
50. B. Lynn, M. Prabhakaran, and A. Sahai. Positive results and techniques for obfuscation. In *Advances in Cryptology–EUROCRYPT 2004*, pages 20–39. Springer, 2004.
51. N. Nisan and D. Zuckerman. Randomness is linear in space. *Journal of Computer and System Sciences*, pages 43–52, 1993.
52. R. Pass, K. Seth, and S. Telang. Obfuscation from semantically-secure multi-linear encodings. Cryptology ePrint Archive, Report 2013/781, 2013. http://eprint.iacr.org/.
53. V. M. Patel, N. K. Ratha, and R. Chellappa. Cancelable biometrics: A review. *IEEE Signal Processing Magazine*, 32(5):54–65, 2015.
54. C. Percival and S. Josefsson. The scrypt password-based key derivation function. Technical report, 2016.
55. P. J. Phillips, K. W. Bowyer, P. J. Flynn, X. Liu, and W. T. Scruggs. The iris challenge evaluation 2005. In *Biometrics: Theory, Applications and Systems, 2008. BTAS 2008. 2nd IEEE International Conference on*, pages 1–8. IEEE, 2008.
56. P. J. Phillips, W. T. Scruggs, A. J. O'Toole, P. J. Flynn, K. W. Bowyer, C. L. Schott, and M. Sharpe. FRVT 2006 and ICE 2006 large-scale experimental results. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2006.
57. S. Prabhakar, S. Pankanti, and A. K. Jain. Biometric recognition: Security and privacy concerns. *IEEE Security & Privacy*, 1(2):33–42, 2003.
58. K. Simoens, P. Tuyls, and B. Preneel. Privacy weaknesses in biometric sketches. In *IEEE Symposium on Security and Privacy*, pages 188–203. IEEE, 2009.
59. G. Valiant and P. Valiant. A CLT and tight lower bounds for estimating entropy. In *Electronic Colloquium on Computational Complexity (ECCC)*, volume 17, page 9, 2010.
60. G. Valiant and P. 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. ACM, 2011.
61. D. Wang, Z. Zhang, P. Wang, J. Yan, and X. Huang. Targeted online password guessing: An underestimated threat. In *Proceedings of the 2016 ACM SIGSAC conference on computer and communications security*, pages 1242–1254. ACM, 2016.
62. Y. Wen and S. Liu. Robustly reusable fuzzy extractor from standard assumptions. In *Advances in Cryptology –ASIACRYPT*, 2018.
63. Y. Wen, S. Liu, and S. Han. Reusable fuzzy extractor from the decisional Diffie–Hellman assumption. *Designs, Codes and Cryptography*, Jan 2018.
64. J. Woodage, R. Chatterjee, Y. Dodis, A. Juels, and T. Ristenpart. A new distribution-sensitive secure sketch and popularity-proportional hashing. In *Advances in Cryptology - CRYPTO*, pages 682–710. Springer, 2017.