A trivial debiasing scheme for Helper Data Systems

Boris Škorić

Abstract We introduce a debiasing scheme that solves the more-noise-than-entropy problem which can occur in Helper Data Systems when the source is very biased. We perform a condensing step, similar to Index Based Syndrome coding, that reduces the size of the source space in such a way that some source entropy is lost while the noise entropy is greatly reduced. In addition, our method allows for even more entropy extraction by means of a 'spamming' technique. Our method outperforms solutions based on the one-pass and two-pass von Neumann algorithms.

1 Introduction

1.1 Helper Data Systems

The past decade has seen a lot of interest in the field of *security with noisy data*. In several security applications it is necessary to reproducibly extract secret data from noisy measurements on a physical system. One such application is read-proof storage of cryptographic keys using Physical Unclonable Functions (PUFs) [19,20,18,5,16]. Another application is the privacy-preserving storage of biometric data.

Storage of keys in nonvolatile digital memory can often be considered insecure because of the vulnerability to physical attacks. (For instance, fuses can be optically inspected with a microscope; flash memory may be removed and read out.) PUFs provide an alternative way to store keys, namely in analog form, which allows the designer to exploit the inscrutability of analog physical behavior. Keys stored in this way are sometimes referred to as Physically Obfuscated Keys (POKs) [12]. In both the biometrics and the PUF/POK case, one faces the problem that some form of error correction has to be performed, but under the constraint that the redundancy data, which is visible to attackers, does not endanger the secret extracted from the physical measurement. This problem is solved by a special security primitive, the *Helper Data System* (HDS). A HDS in its most general form is shown in Fig. 1. The `Gen` procedure takes as input a measurement X. It outputs a secret S and (public) Helper Data W. The helper data is stored. In the reconstruction phase, a fresh measurement X' is obtained. Typically X' is a noisy version of X, close to X (in terms of e.g. Euclidean distance or Hamming distance) but not necessarily identical. The `Rec` (reconstruction) procedure takes X' and W as input. It outputs $\hat{S}$, an estimate of S. If X' is not too noisy then $\hat{S} = S$.

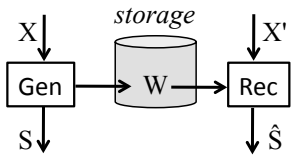

Fig. 1 *Data flow in a generic Helper Data System.*

Two special cases of the general HDS are the Secure Sketch (SS) and the Fuzzy Extractor (FE) [10]. The Secure Sketch has $S = X$ (and $\hat{S} = \hat{X}$, an estimator for X). If X is not uniformly distributed, then S is not uniform. The SS is suitable for privacy-preserving biometrics, where high entropy of S (given W) is required, but not uniformity. The Fuzzy Extractor is required to have a (nearly) uniform S given W. The FE is typically used for extracting keys from PUFs and POKs.

1.2 The Code Offset Method (COM)

By way of example we briefly present the Code Offset Method [4,14,10,9,21], the oldest and most well-known HDS. The COM makes use of a linear code $\mathcal{C}$ with k-bit messages and n-bit codewords. The syndrome function is denoted as $\texttt{Syn} : \{0,1\}^n \to \{0,1\}^{n-k}$. The code supports a syndrome decoding algorithm $\texttt{SynDec} : \{0,1\}^{n-k} \to \{0,1\}^n$ that maps syndromes to error patterns. Use is also made of a key derivation function $\texttt{KeyDeriv} : \{0,1\}^n \times \{0,1\}^* \to \{0,1\}^\ell$, with $\ell \leq k$. Below we show how the COM can be used as a FE for either uniform or non-uniform source X.

B. Škorić
TU Eindhoven
E-mail: b.skoric@tue.nl

Enrollment:
The enrollment measurement gives $X \in \{0,1\}^n$. The helper data is computed as $W = \mathtt{Syn}\, X$. The key is computed as $S = \mathtt{KeyDeriv}(X, R)$, where R is (optional) public randomness. The W and R are stored.
Reconstruction:
A fresh measurement of the PUF gives $X' \in \{0,1\}^n$. The estimator for X is computed as

$$\hat{X} = X' \oplus \mathtt{SynDec}(W \oplus \mathtt{Syn}\, X') \tag{1}$$

and the reconstructed key is $\hat{S} = \mathtt{KeyDeriv}(\hat{X}, R)$.

1.3 The problem of bias

After seeing the helper data W as specified above, an attacker has uncertainty $\mathsf{H}(X|W)$ about X.[1] As W is a function of X, we have $\mathsf{H}(X|W) = \mathsf{H}(X|\mathtt{Syn}\, X) = \mathsf{H}(X) - \mathsf{H}(\mathtt{Syn}\, X)$. Let us consider the simplest possible noise model, the Binary Symmetric Channel (BSC). In the case of a BSC with bit error rate β, the code's redundancy has to satisfy $n - k \geq nh(\beta)$ in order for the error-correction to work. (Here h denotes the binary entropy function, $h(\beta) = -\beta \log \beta - (1-\beta)\log(1-\beta)$, with 'log' the base 2 logarithm). More generally, $\mathsf{H}(\mathtt{Syn}\, X)$ has at least to be equal to the entropy of the noise. Hence the COM helper data in the BSC case leaks at least $nh(\beta)$ bits of information about X. This becomes a problem when X itself does not have much entropy, which occurs for instance if the bits in X are highly biased [13,15]. Note that the problem is not fundamental: if the bias parameter (see Section 2) is denoted as p, the secrecy capacity is $nh(p + \beta - 2p\beta) - nh(\beta)$, which is positive.

A solution was proposed by Maes et al. [17]. Their approach is to combine debiasing with error correction. For the debiasing they use the von Neumann algorithm in a single-pass or multi-pass manner. Their helper data comprises a selection 'mask' that helps the reconstruction algorithm to identify the locations where von Neumann should yield output.

In this paper we follow a simpler approach similar to the Index Based Syndrome (IBS) method [22]. In IBS the helper data consists of an ordered list of pointers to locations in X; the content of X in those locations together forms a codeword.

1.4 Contributions and outline

We introduce an alternative solution to the bias problem in helper data systems. We follow the *condense-then-fuzzy-extract* philosophy proposed by Canetti et al. [7] as one of the available options when faced with 'more noise than entropy' scenarios. Condensing means mapping the source variable X to a smaller space such that most of the entropy of X is retained, but the noise entropy is greatly reduced. Our way of condensing the source is to restrict X to the bit positions $\mathcal{U}$, with $\mathcal{U} \subset \{1, \ldots, n\}$ a random subset containing all the rare symbols. The set $\mathcal{U}$ becomes part of the helper data.

Our $\mathcal{U}$ bears some similarity to the von Neumann mask in [17], but there are important differences. (i) The size of $\mathcal{U}$ is tunable. (ii) We can extract source information based on the legitimate party's ability to distinguish $\mathcal{U}$ from fake instances of $\mathcal{U}$ when a 'spamming' technique similar to [21] is applied.

The outline of this paper is as follows. Section 2 gives the details of the scheme. Section 3 analyzes the extractable entropy and the practicality of the 'spamming' option. In Section 5 we make some remarks on the use of min-entropy. Section 6 summarises and suggests future work.

2 Debiasing based on subset selection

We will use the following notation. The set $\{1, \ldots, n\}$ is written as $[n]$. The notation $X_{\mathcal{U}}$ means X restricted to the positions specified in $\mathcal{U}$. Set difference is written as '$\setminus$'. A string consisting of n zeroes is written as 0^n. The Hamming weight of X is denoted as $w(X)$. We will consider a source $X \in \{0,1\}^n$ made up of i.i.d. bits X_i following a Bernoulli distribution with parameter p, i.e. $\Pr[X_i = 1] = p$. Without loss of generality we take $p \in (0, \frac{1}{2})$. In particular we are interested in the case $p < \beta$ where direct application of the COM fails. The notation 'log' stands for the base-2 logarithm. Information distance (Kullback-Leibler divergence) is denoted as $D(p||q) = p \log \frac{p}{q} + (1-p) \log \frac{1-p}{1-q}$ for $p, q \in (0,1)$.

2.1 The scheme

Below we present a barebones version of our proposed scheme. We omit details concerning the protection of the stored data. There are well known ways to protect helper data, using either Public Key Infrastructure or one-way functions [6]. We also omit details that have to do with the verification of the reconstructed key. These details can be trivially added.

System setup
The following system parameters are fixed. An integer u satisfying $np < u < n$, representing the size of $\mathcal{U}$; a list length L; a pseudorandom generator f that takes

[1] The notation H stands for Shannon entropy. For information-theoretic concepts see e.g. [8].

as input a seed σ and a counter j, and outputs a subset $f(\sigma, j) \subset [n]$ such that $|f(\sigma, j)| = u$; a Secure Sketch $(\mathtt{Gen}, \mathtt{Rec})$ that acts on a source in $\{0,1\}^u$ and is able to handle bit error rate β; a key derivation function $\mathtt{KDF} : \{0,1\}^u \times [L] \to \{0,1\}^\ell$. All these parameters are public.

Enrollment

E1. Measure $X \in \{0,1\}^n$.
E2. Draw a random subset $\mathcal{U} \subset [n]$ of size u such that $X_\mathcal{U}$ contains as many '1's as possible.
E3. Compute $Y = X_\mathcal{U}$ and $W = \mathtt{Gen}(Y)$.
E4. Draw a random seed σ.
E5. Draw a random $z \in [L]$. Determine a permutation $\pi : [n] \to [n]$ such that[2] $\pi(\mathcal{U}) = f(\sigma, z)$.
E6. Derive the secret key as $S = \mathtt{KDF}(Y, z)$.
E7. Store σ, π, W.

Reconstruction

R1. Read σ, π, W.
R2. Measure $X' \in \{0,1\}^n$.
R3. Construct a set $\mathcal{M}' = \{i \in [n] : \ X'_i = 1\}$. Compute $\mathcal{M} = \pi(\mathcal{M}')$.
R4. Construct the list $\mathcal{L} = (f(\sigma, j))_{j=1}^L$.
R5. Determine the index $\hat{z} \in [L]$ for which $\mathcal{L}_{\hat{z}}$ has the largest overlap with $\mathcal{M}$.
R6. Compute $\hat{\mathcal{U}} = \pi^{\mathrm{inv}}(\mathcal{L}_{\hat{z}})$ and $\hat{Y} = \mathtt{Rec}(X'_{\hat{\mathcal{U}}}, W)$.
R7. Reconstruct the key as $\hat{S} = \mathtt{KDF}(\hat{Y}, \hat{z})$.

We will typically consider $u \geq 2np$. With an exponentially small probability it may occur that $w(X) \geq u$, leading to $X_\mathcal{U} = 1^u$ in step E2. Even if such an exceptional PUF is encountered, the scheme still works.

2.2 Explanation of the scheme

The effect of steps E4,E5 is to create a list of $\mathcal{U}$-candidates, of which only entry z is correct. To an attacker (who knows u but does not know X or X') the L candidates are indistinguishable.[3]

Steps R3–R5 allow for a quick search to identify the index z of the correct $\mathcal{U}$-candidate. Note that the reconstruction algorithm compares a permuted $\mathcal{M}$ to $\mathcal{L}$-entries instead of $\mathcal{M}$ to permuted $\mathcal{L}$-entries; this improves speed. To further optimize for speed, steps R4 and R5 can be combined in a loop to select good z values on the fly as soon as a new $\mathcal{L}_j$ is generated.

Note that extremely fast pseudorandom generators exist which spew out more than 8 bits per clock cycle [1,2]. This makes it practical to work with large values of L, as long as not too many plausible z-candidates are generated. See Section 3.3. Even on CPU-constrained devices[4] (clock speed of MHz order) it should be possible to achieve $L = 2^{10}$.

We did not explicitly specify how to map a seed to a size-u subset of $[n]$. A very straightforward algorithm would be to pick u pseudorandom locations in $[n]$.

We did not specify an algorithm for determining the permutation π, nor did we specify in which form π is stored. These are minor details and have no impact on the overall efficiency of the scheme, since steps E5 and R3 are performed only once. The computational bottleneck is R4,R5. For details about permutations we refer to [11,3].

Note that inputting z into the key derivation function increases the entropy of S by $\log L$ bits. If the PUF has ample entropy then $L = 1$ suffices, and one can skip all steps involving the seed σ and the permutation π; the $\mathcal{U}$ itself is stored as helper data. This yields a scheme that is very fast and implementable on resource-constrained devices.

3 Analysis

3.1 Entropy after condensing

The Hamming weight of X carries very little information. Let's assume for the moment that $T = w(X) \in \{0, \ldots, u\}$ is known to the attacker[5], just to simplify the analysis.

Even if the attacker knows t and $\mathcal{U}$ (i.e. z), there are $\binom{u}{t}$ equally probable possibilities for Y. Hence

$$\begin{aligned} \mathsf{H}(Y|Z = z, T = t) &= \log \binom{u}{t} \\ &> uh(\frac{t}{u}) - \frac{1}{2} \log \frac{t(u-t)}{u} - \log \frac{e^2}{\sqrt{2\pi}}. \end{aligned} \tag{2}$$

The inequality follows from Stirling's approximation.

As (2) does not depend on z, the entropy $\mathsf{H}(Y|T = t)$ is also given by (2). A lower bound on $\mathsf{H}(Y|T)$ is obtained by taking the expectation over t. This turns out to be rather messy, since the distribution of t is a truncated binomial. (It is given that $t \leq u$, while originally $w(X) \in \{0, \ldots, n\}$). As t equals approximately np on average, the result is $\mathsf{H}(Y|T) \approx uh(\frac{np}{u})$. A more precise lower bound is given in Theorem 1 below.

[2] $\pi(\mathcal{U})$ means π applied to each element of $\mathcal{U}$ individually.

[3] Given the i.i.d. model for creating X, the set $\mathcal{U}$ itself is uniformly random. If we want a different model for X, e.g. with asymmetries between the positions, then $\mathcal{L}$ will need to be generated in a way that follows the statistics of X.

[4] We are considering devices that are able to employ the PUF key in cryptographic computations; hence they cannot be *very* resource-constrained.

[5] In the security analysis we denote random variables using capitals, and their numerical realisations in lowercase.

Theorem 1 *Let δ be defined as*

$$\delta = \min\{e^{-2np^2(\frac{u}{np}-1)^2}, e^{-np\frac{1}{3}(\frac{u}{np}-1)}, e^{-nD(\frac{u}{n}||p)}\}. \quad (3)$$

Let $\delta < p$. Let $u \geq 2np/(1-\delta)$. Then the entropy of Y given T can be lowerbounded as

$$\begin{aligned}\mathsf{H}(Y|T) > uh\left(\frac{np-n\delta}{u}\right) - \frac{1}{2}\log\frac{np}{1-\delta} \quad (4)\\ -\log\frac{e^2}{\sqrt{2\pi}} + \frac{1}{2\ln 2}\frac{np-n\delta}{u}\\ -u\left\{\frac{1-p}{np} + \frac{2\delta}{p} + \delta + \frac{\delta^3}{p^3}\right\}(1-\delta)^{-1}(1-\frac{\delta}{p})^{-2}.\end{aligned}$$

Proof: The proof in rather tedious and can be found in Appendix A. □

The entropy of Y is obtained as follows,

$$\mathsf{H}(Y) = \mathsf{H}(YT) = \mathsf{H}(T) + \mathsf{H}(Y|T). \quad (5)$$

The $\mathsf{H}(T)$ is the entropy of the *truncated* binomial distribution.

Theorem 2 *Let $q_t = \binom{n}{t}p^t(1-p)^{n-t}$ denote the probabilities in the full binomial distribution. Let δ be defined as in Theorem 1.*

$$\mathsf{H}(T) > \log\frac{2\pi}{e}\sqrt{np(1-p)} + \log(1-\delta) - (n-u)q_u\log\frac{1}{q_u}.$$

Proof: See Appendix B. □

Theorem 3 *Let δ be defined as in Theorem 1. Let $\delta < p$. Let $u \geq 2np/(1-\delta)$.*

$$\begin{aligned}\mathsf{H}(Y) > uh\left(\frac{np-n\delta}{u}\right) + \frac{1}{2\ln 2}\frac{np-n\delta}{u} - \log\frac{e^3}{(2\pi)^{3/2}}\\ +\frac{1}{2}\log(1-p) + \frac{3}{2}\log(1-\delta) - (n-u)q_u\log\frac{1}{q_u}\\ -u\left[\frac{1-p}{np} + \frac{2\delta}{p} + \delta + \frac{\delta^3}{p^3}\right](1-\delta)^{-1}(1-\frac{\delta}{p})^{-2}. \quad (6)\end{aligned}$$

Proof: Follows directly from Theorems 1 and 2 by adding $\mathsf{H}(T) + \mathsf{H}(Y|T)$. □

The complicated expression (6) can be well approximated by $uh(np/u) - u/np$. Note that the difference between the original source entropy $\mathsf{H}(X) = nh(p)$ and the condensed form $\mathsf{H}(Y) \approx uh(np/u) - u/np$ is considerable. For example, setting $n = 640$, $p = 0.1$, $u = 128$ yields $\mathsf{H}(Y) \approx 126$ and $\mathsf{H}(X) \approx 300$. A small part of this huge difference can be regained using the trick with the entropy of Z. The practical aspects are discussed in Section 3.3.

Note also that our scheme outperforms simple von Neumann debiasing by a factor of at least 2. The von Neumann algorithm takes $n/2$ pairs of bits; each pair has a probability $2p(1-p)$ of generating a (uniform) output bit; hence the extracted entropy is $np(1-p) < np$. In our scheme, if we set $u \approx 2np$ we get $\mathsf{H}(Y) \approx 2np - 2$. Furthermore, $\mathsf{H}(Y)$ is an increasing function of u.

3.2 Fuzzy Extraction after condensing

The Code Offset Method applied to $Y \in \{0,1\}^u$ leaks at least $uh(\beta)$ bits of information about Y. In case of a 'perfect' error-correcting code, the length of a noise-robust key reconstructed with the COM is $\mathsf{H}(Y) - uh(\beta)$. In Fig. 2 we plot $\mathsf{H}(Y) - uh(\beta)$ for some example parameter settings. (Note that in all cases shown here $\beta \geq p$ holds; the COM acting on the original source X would be unable to extract any entropy.) Clearly there is an optimal u for given (n, p, β). In practice one is given p and β and has to find (n, u) such that the $\mathsf{H}(Y) - uh(\beta)$ is large enough.

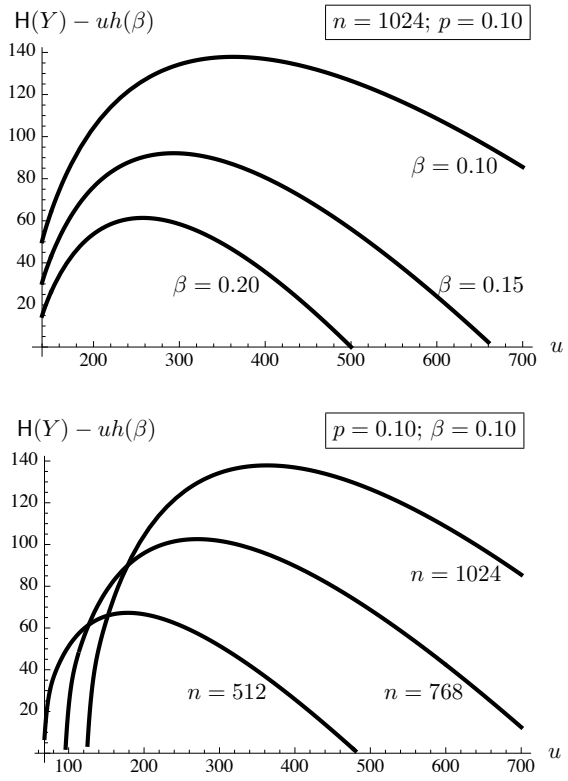

Fig. 2 *The key length $\mathsf{H}(Y) - uh(\beta)$ that the Code Offset Method can reproducibly extract after the condensing step, plotted as a function of u for various parameter values. For $\mathsf{H}(Y)$ the bound in Theorem 3 is used.*

3.3 The list size L

We briefly discuss how large L can be made before the reconstruction procedure sketched in Section 2 starts to produce too many candidates for z. We define $\tilde{p} = p + \beta - 2p\beta$.

On the one hand there is the number of '1' symbols in $X'_{\mathcal{U}}$ for the **correct** $\mathcal{U}$. The number of 1's in $X_{\mathcal{U}}$ is

on average np. Of these, on average $np(1-\beta)$ will be a '1' in X'. The $(u-np)$ zeroes in $X_{\mathcal{U}}$ will generate on average $(u-np)\beta$ 1's in X'. Hence the number of 1's in $X'_{\mathcal{U}}$ is expected to be around $np(1-\beta)+(u-np)\beta = np+(u-2np)\beta$.

On the other hand there is the number of 1's for **incorrect** $\mathcal{U}$-candidates. The number of 1's in X' is approximately $n\tilde{p}$. We pretend that $n\tilde{p}$ is integer, for notational simplicity. We denote by $A \in \{0,\ldots,n\tilde{p}\}$ the number of 1's in $X'_{\mathcal{V}}$ for a *randomly chosen* subset $\mathcal{V}$, with $|\mathcal{V}| = u$. The A follows a hypergeometric probability distribution

$$\Pr[A=a] = \frac{\binom{n\tilde{p}}{a}\binom{n-n\tilde{p}}{u-a}}{\binom{n}{u}} = \frac{\binom{u}{a}\binom{n-u}{n\tilde{p}-a}}{\binom{n}{n\tilde{p}}}. \quad (7)$$

The first expression looks at the process of selecting u out of n positions with exactly a 1's hitting the $n\tilde{p}$ existing 1's in X'; the second expression looks at the process of selecting $n\tilde{p}$ positions in X' such that exactly a of them lie in $\mathcal{V}$. We have $\mathbb{E}_a\, a = u\tilde{p}$ and $\mathbb{E}_a(a-u\tilde{p})^2 = u\tilde{p}(1-\tilde{p})(n-u)/(n-1) < u\tilde{p}$. In other words, a is sharply peaked around $u\tilde{p}$.

We can put a threshold θ somewhere in the gap between $u\tilde{p}$ and $np+(u-2np)\beta$, and declare a $\mathcal{U}$-candidate $\mathcal{V}$ to be bad if the Hamming weight of $X'_{\mathcal{V}}$ is lower than θ. Let's set $\theta = np+(u-2np)\beta - c\cdot\sqrt{np}$ with c sufficiently large to avoid false negatives (i.e. missing the correct $\mathcal{U}$). In a way analogous to (3), we can bound the false positive probability as

$$\Pr[A \geq \theta] < \min\{e^{-2u\tilde{p}^2(\theta/u\tilde{p}-1)^2},\ e^{-u\tilde{p}\frac{1}{3}(\theta/u\tilde{p}-1)},\ e^{-uD(\frac{\theta}{u}||\tilde{p})}\}. \quad (8)$$

These bounds are obtained by applying (3) and replacing $u\to\theta$, $n\to u$, $p\to\tilde{p}$. Fig. 3 shows how many bits of entropy ($b \approx -\log\Pr[A>\theta]$) can be obtained from z without running into false positives in step R5. To extract b bits of entropy, a list length $L = 2^b$ is needed. Fig. 3 serves just to show the orders of magnitude and is by no means an exhaustive treatment of the whole parameter space n, p, β, u, θ. We remark that the curves depend quite strongly on the threshold parameter c.

It is important to note that it is perfectly possible to make L extremely large. Then many false positives occur, but this is not a fundamental problem. It requires extra work: one key reconstruction and one verification (e.g. of a key hash) per false positive. Depending on the available n, the computing platform etc. this may be a viable option.

4 Comparison to other debiasing schemes

The main algorithms to compare against are given in the CHES2015 paper by Maes et al [17]. They introduce several schemes that perform debiasing in the context of a helper data system: Classic Von Neumann (CVN), Pair-Output (2O-VN) and Multi-Pass Tuple Output (MP-TO-VN). The following figures of merit are important, (i) the amount of entropy retained in Y, from the original $nh(p)$ contained in X; (ii) the amount of work required *during the reconstruction phase* to derive $\hat{Y}$ from X'.

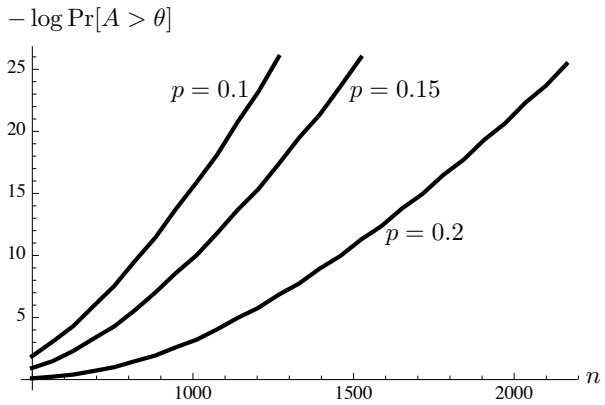

Fig. 3 *Plots of* $-\log\Pr[A>\theta]$ *as a function of* n *for* $\beta = p$, $u = 2.25np$, *and* $\theta = np+(u-2np)\beta - c\sqrt{np}$ *with* $c = 4$. *The* $\Pr[A>\theta]$ *was obtained by numerical summation of (7). The vertical axis represents approximately the number of bits* $b = \log L$ *that can be extracted from* z *without generating false positives.*

Here we will *not* include the additional entropy obtained from Z in our scheme. The procedure for reconstructing $\hat{z}$ has no equivalent in [17], which makes comparison very difficult.

scheme	*retained entropy*	*reconstruction of* $\hat{Y}$
Trivial Debiasing	$\approx 2np-2$ (at $u=2np$)	take subset of X'
CVN	$np(1-p)$	take subset of X'; $\approx np$ binary comp.
2O-VN	$np(1-p)$	take subset of X'
MP-TO-VN (two-pass)	$np(1-p)$ $+\frac{1}{2}n\frac{p^2(1-p)^2}{p^2+(1-p)^2}$	take subset of X'

The reconstruction effort is practically the same in all these schemes, and is very low.

The entropy estimates are obtained as follows. The result $np(1-p)$ for the original von Neumann algorithm was discussed in Section 3.1. The CVN and 2O-VN retain exactly this amount. In the second VN pass as described in [17], there are $\frac{n}{2} - np(1-p)$ bit pairs left after the first pass, and in each bit pair the two bits have the same value. In the second pass this gives rise to $\frac{n}{4} - \frac{1}{2}np(1-p) = \frac{n}{4}[p^2+(1-p)^2]$ von Neumann comparisons. Each comparison yields an output bit with probability $2\frac{p^2}{p^2+(1-p)^2}\frac{(1-p)^2}{p^2+(1-p)^2}$. Hence the expected number of output bits in the second pass is $\frac{1}{2}np^2(1-p)^2/[p^2+(1-p)^2]$.

Note that the two-pass MP-TO-VN adds at most 25% to the CVN entropy, while Trivial Debiasing adds slightly more than 100%.

5 Some remarks on min-entropy

One could take the point of view that the relevant quantity to study is min-entropy instead of Shannon entropy, since we are deriving cryptographic keys. The min-entropy of the source is $\mathsf{H}_{\min}(X) = n\log(1-p)^{-1}$, corresponding to the all-zero string. For small p this is significantly smaller than the Shannon entropy $\mathsf{H}(X) = nh(p)$. On the other hand, the entropy loss is also smaller when computed in terms of min-entropy.

Theorem 4 *Consider a linear binary code with message length k and codeword length n that is able to correct t errors. Let $X \in \{0,1\}^n$ consist of i.i.d. bits that are Bernoulli-distributed with parameter $p < \frac{1}{2}$.*

$$\begin{aligned}\mathsf{H}_{\min}(X|\mathtt{Syn}\,X) &> \mathsf{H}_{\min}(X) - (n-k) \\ &\quad +(t+1)\log\frac{1-p}{p} \\ &\quad -\frac{1}{2^{n-k}\ln 2}\sum_{a=0}^{t}\binom{n}{a}\left\{(\frac{1-p}{p})^{t-a+1}-1\right\}. \qquad (9)\end{aligned}$$

The proof is given in Appendix C. For codes that are far from perfect, the last term in (9) is negligible.

However, there are strong arguments against using min-entropy in the context of biased PUFs. A situation where X has a Hamming weight far below the *typical* value np can be seen as a hardware error and is likely to occur only when the chip itself is malfunctioning. If we condition all our probabilities on the premise that the hardware is functioning correctly, then we are back in the typical regime; there min-entropy is almost identical to Shannon entropy.

6 Summary

We have introduced a method for source debiasing that can be used in Helper Data Systems to solve the 'more noise than entropy' problem. Our method applies the condense-then-fuzzy-extract idea [7] in a particularly simple way: the space $\{0,1\}^n$ is condensed to $\{0,1\}^u$ in such a way that all the rare symbols are kept; meanwhile the noise entropy is reduced from $nh(\beta)$ to $uh(\beta)$. Theorem 3 gives a lower bound on the retained entropy $\mathsf{H}(Y)$. Furthermore, there is the option of extracting additional entropy from the index z, which points at the real subset $\mathcal{U}$ among the fakes. Even in its bare form, without the fake subsets, our method outperforms basic von Neumann debiasing by factor of at least 2.

Fig. 2 shows that after the condensing step the Code Offset Method can extract significant entropy in a situation where the bare COM fails. It also shows the tradeoff between the reduction of source entropy and noise entropy as u varies.

In Section 3.3 we did a very preliminary analysis of the practicality of extracting information from the index z. More work is needed to determine how this works out for real-world parameter values n, p, β and to see how the computations in steps R4 and R5 can be optimised for speed.

The entropy analysis can be improved and extended in various ways, e.g. by considering different noise models such as asymmetric noise.

Acknowledgements We thank Sebastian Verschoor, Frans Willems, Ruud Pellikaan and Niels de Vreede for useful discussions.

A Proof of Theorem 1

We start with a number of definitions and supporting lemmas. We define the binomial probability $q_t = \binom{n}{t}p^t(1-p)^{n-t}$. We define $\Delta = \Pr[w(X) > u] = \sum_{t=u+1}^{n} q_t$ and $\pi_t = q_t/(1-\Delta)$ for $t \le u$, such that the vector (π_t) is the probability distribution of t from the attacker's point of view. The notation $\mathbb{E}_t$ will refer to the binomial distribution $(q_t)_{t=0}^n$, while $\tilde{\mathbb{E}}$ will refer to the truncated binomial $(\pi_t)_{t=0}^u$.

Lemma 1 *Let $k \ge 1$, $n \ge 2$.*

$$\begin{aligned}&nh(\frac{k}{n}) - \tfrac{1}{2}\log\frac{k(n-k)}{n} - \log\frac{e^2}{\sqrt{2\pi}} \quad \le \log\binom{n}{k} \\ &\le nh(\frac{k}{n}) - \tfrac{1}{2}\log\frac{k(n-k)}{n} - \log\frac{2\pi}{e}. \qquad (10)\end{aligned}$$

Proof: We write $\binom{n}{k} = \frac{n!}{k!(n-k)!}$ and apply Stirling's inequalities $\sqrt{2\pi}N^{N+\frac{1}{2}}e^{-N} \le N! \le eN^{N+\frac{1}{2}}e^{-N}$ (valid for $N \ge 1$) to $n!$, $k!$ and $(n-k)!$. In the first line the inequalities used are $n! \ge \ldots$, $k! \le \ldots$, and $(n-k)! \le \ldots$. In the second line the inequalities point in the other direction. After some tedious rewriting the Lemma follows. □

Lemma 2 *Let $u > 2np$. For the above defined Δ it then holds that $\Delta \le \delta$, with*

$$\delta \stackrel{\text{def}}{=} \min\{e^{-2np^2(\frac{u}{np}-1)^2}, e^{-np\frac{1}{3}(\frac{u}{np}-1)}, e^{-nD(\frac{u}{n}||p)}\}. \qquad (11)$$

Proof: The three expressions result from the tail inequalities of Hoeffding, Chernoff, and Chernoff-Hoeffding respectively. Hoeffding's inequality states that $\Pr[w(X) \ge np + n\varepsilon] \le \exp(-2n\varepsilon^2)$; a version the Chernoff bound for $\varepsilon \ge p$ has $\Pr[w(X) \ge np + n\varepsilon] \le \exp(-\frac{1}{3}n\varepsilon)$; Chernoff-Hoeffding gives $\exp[-nD(p+\varepsilon||p)]$. We have $\varepsilon = u/n - p$. Only the listed form of the Chernoff bound actually needs $u > 2np$ as a condition. □

Lemma 3 *It holds that $\tilde{\mathbb{E}}_t\, t > n(p-\delta)$.*

Proof:

$$\tilde{\mathbb{E}}_t\, t = \frac{1}{1-\Delta}\sum_{t=0}^{u} q_t t > \sum_{t=0}^{u} q_t t = np - \sum_{t=u+1}^{n} q_t t$$
$$> np - \sum_{t=u+1}^{n} q_t n = np - n\Delta \geq np - n\delta. \tag{12}$$

In the last step we used Lemma 2. □

Lemma 4 *It holds that* $\tilde{\mathbb{E}}_t\, t < \frac{np}{1-\delta}$.

Proof:

$$\tilde{\mathbb{E}}_t\, t = \frac{1}{1-\Delta}\sum_{t=0}^{u} q_t t < \frac{1}{1-\Delta}\sum_{t=0}^{n} q_t t = \frac{np}{1-\Delta} \leq \frac{np}{1-\delta}. \tag{13}$$

In the last step we used Lemma 2. □

Lemma 5 *It holds that*

$$\tilde{\mathbb{E}}_t\, t^2 < \frac{(np)^2 + np(1-p)}{1-\delta}. \tag{14}$$

Proof:

$$\tilde{\mathbb{E}}_t\, t^2 = \frac{1}{1-\Delta}\sum_{t=0}^{u} q_t t^2 < \frac{1}{1-\Delta}\mathbb{E}_t\, t^2$$
$$= \frac{(np)^2 + np(1-p)}{1-\Delta} \leq \frac{(np)^2 + np(1-p)}{1-\delta}. \tag{15}$$

In the last step we used Lemma 2. □

Lemma 6 *Let* $p \in [0,1]$. *Let* $r \in (0, \frac{1}{2}]$. *Then it holds that*

$$h(p) \geq \Omega_r(p) \tag{16}$$
$$\Omega_r(p) = h(r) + (p-r)h'(r) - \frac{(p-r)^2}{r^2}[h(r) - rh'(r)].$$

The expression $h(r) - rh'(r)$ *is a nonnegative increasing function on* $r \in (0, \frac{1}{2}]$.

Proof: Ω_r is a parabola constructed such that $\Omega_r(0) = 0$, $\Omega_r(r) = h(r)$ and $\Omega_r'(r) = h'(r)$. The property $h(p) \geq \Omega_r(p)$ is verified by visual inspection. We define $g(r) = h(r) - rh'(r)$. We have $\lim_{r\to 0} g(r) = 0$. Furthermore $\frac{\mathrm{d}}{\mathrm{d}r} g(r) = -rh''(r) > 0$, which proves that g is increasing. Together with $g(0) = 0$ that implies that $g(r)$ is nonnegative on the given interval. □

Lemma 7 *Let* $\delta < p$ *and* $u \geq 2np/(1-\delta)$.

$$\tilde{E}_t\, h(\frac{t}{u}) > h(\frac{np - n\delta}{u}) \tag{17}$$
$$- \left\{ \frac{1-p}{np} + \frac{2\delta}{p} + \delta + \frac{\delta^3}{p^3} \right\} (1-\delta)^{-1}(1 - \frac{\delta}{p})^{-2}.$$

Proof: We use Lemma 6 to expand $h(\frac{t}{u})$ around $t = \tilde{\mathbb{E}}_t\, t$,

$$h(\tfrac{t}{u}) \geq h(\tilde{\mathbb{E}}_t \tfrac{t}{u}) + \text{linear} - \frac{(t - \tilde{\mathbb{E}}_t\, t)^2}{(\tilde{\mathbb{E}}_t\, t)^2}[h(\tilde{\mathbb{E}}_t \tfrac{t}{u}) - \tilde{\mathbb{E}}_t[\tfrac{t}{u}]h'(\tilde{\mathbb{E}}_t \tfrac{t}{u})]. \tag{18}$$

When we take the expectation $\tilde{\mathbb{E}}_t$, the term linear in $t - \tilde{\mathbb{E}}_t\, t$ disappears,

$$\tilde{\mathbb{E}}_t h(\tfrac{t}{u}) \geq h(\tilde{\mathbb{E}}_t \tfrac{t}{u}) - [\frac{\tilde{\mathbb{E}}_t\, t^2}{(\tilde{\mathbb{E}}_t\, t)^2} - 1]\left[h(\tilde{\mathbb{E}}_t \tfrac{t}{u}) - \tilde{\mathbb{E}}_t[\tfrac{t}{u}]h'(\tilde{\mathbb{E}}_t \tfrac{t}{u})\right]. \tag{19}$$

We use $\tilde{\mathbb{E}}_t\, t < u/2$ to bound the second occurrence of $h(\tilde{\mathbb{E}}_t \frac{t}{u})$ as $h(\cdots) < 1$ and to use $h' > 0$. For the first occurrence of $h(\tilde{\mathbb{E}}_t \frac{t}{u})$ we use that h is an increasing function and apply Lemma 3.

$$\tilde{\mathbb{E}}_t h(\tfrac{t}{u}) > h(\frac{np - n\delta}{u}) - [\frac{\tilde{\mathbb{E}}_t\, t^2}{(\tilde{\mathbb{E}}_t\, t)^2} - 1]. \tag{20}$$

Finally we bound $\tilde{\mathbb{E}}_t\, t^2$ using Lemma 5 and we bound $\tilde{\mathbb{E}}_t\, t$ using Lemma 3. □

Lemma 8

$$\tilde{\mathbb{E}}_t \log \binom{u}{t} > u\tilde{\mathbb{E}}_t h(\tfrac{t}{u}) - \log \frac{e^2}{\sqrt{2\pi}} - \tfrac{1}{2}\log \frac{np}{1-\delta} + \frac{1}{2\ln 2}\frac{np - n\delta}{u}. \tag{21}$$

Proof: We start from Lemma 1 and write

$$\log \binom{u}{t} \geq uh(\frac{t}{u}) - \tfrac{1}{2}\log \frac{t(u-t)}{u} - \log \frac{e^2}{\sqrt{2\pi}}. \tag{22}$$

We expand the second term as $-\frac{1}{2}\log \frac{t(u-t)}{u} = -\frac{1}{2}\log t - \frac{1}{2}\log(1 - \frac{t}{u}) > -\frac{1}{2}\log t + \frac{1}{2\ln 2}\frac{t}{u}$. Then we apply $\tilde{\mathbb{E}}_t$. Jensen followed by Lemma 4 gives $\tilde{\mathbb{E}}_t \log t \leq \log \tilde{\mathbb{E}}_t\, t < \log \frac{np}{1-\delta}$. Lemma 3 gives $\mathbb{E}_t \frac{t}{u} > \frac{np - n\delta}{u}$. □

With all these lemmas we can finally prove Theorem 1. From (2) we have

$$\mathsf{H}(Y|T) = \tilde{\mathbb{E}}_t \mathsf{H}(Y|T=t) = \tilde{\mathbb{E}}_t \log \binom{u}{t}. \tag{23}$$

We apply Lemma 8, and then Lemma 7 to lowerbound $\tilde{\mathbb{E}}_t h(\frac{t}{u})$.

B Proof of Theorem 2

We begin by lowerbounding the entropy of the un-truncated distribution $(q_t)_{t=0}^n$.

Lemma 9 *It holds that* $\mathsf{H}(\boldsymbol{q}) \geq \log \frac{2\pi}{e}\sqrt{np(1-p)}$.

Proof:

$$\mathsf{H}(\boldsymbol{q}) = -\mathbb{E}_t \log[\binom{n}{t} p^t (1-p)^{n-t}]$$
$$= (-\mathbb{E}_t t)\log p - (n - \mathbb{E}_t t)\log(1-p) - \mathbb{E}_t \log \binom{n}{t}$$
$$= nh(p) - \mathbb{E}_t \log \binom{n}{t}$$
$$\geq nh(p) - \log \binom{n}{\mathbb{E}_t t} = nh(p) - \log \binom{n}{np}. \tag{24}$$

In the last line we used Jensen's inequality for the function $\log \binom{n}{t}$. We use Lemma 1 to upperbound $\log \binom{n}{np}$. □

Now we write

$$\mathsf{H}(T) = \tilde{\mathbb{E}}_t \log \frac{1-\Delta}{q_t} = \log(1-\Delta) + \frac{1}{1-\Delta}\sum_{t=0}^{u} q_t \log \frac{1}{q_t}$$
$$> \log(1-\delta) + \sum_{t=0}^{u} q_t \log \frac{1}{q_t}$$
$$= \log(1-\delta) + \mathsf{H}(\boldsymbol{q}) - \sum_{t=u+1}^{n} q_t \log \frac{1}{q_t}$$
$$> \log(1-\delta) + \mathsf{H}(\boldsymbol{q}) - \sum_{t=u+1}^{n} q_u \log \frac{1}{q_u}$$
$$= \log(1-\delta) + \mathsf{H}(\boldsymbol{q}) - (n-u)q_u \log \frac{1}{q_u}. \tag{25}$$

In the last inequality we used that $q_t \log \frac{1}{q_t}$ is a decreasing function of t for $t > u$. Finally we lowerbound $\mathsf{H}(\boldsymbol{q})$ with Lemma 9.

C Proof of Theorem 4

$$\begin{aligned}
\mathsf{H}_{\min}(X|\mathtt{Syn}\,X) &= -\log \mathbb{E}_\sigma \max_x \Pr[X=x|\mathtt{Syn}\,X=\sigma] \\
&= -\log \sum_\sigma \max_x \Pr[X=x \wedge \mathtt{Syn}\,X=\sigma] \\
&= -\log \sum_\sigma \max_{x:\,\mathtt{Syn}x=\sigma} p^{w(x)}(1-p)^{n-w(x)} \\
&= -\log (1-p)^n \sum_\sigma \max_{x:\,\mathtt{Syn}x=\sigma} (\frac{p}{1-p})^{w(x)} \\
&= \mathsf{H}_{\min}(X) - \log \sum_\sigma \max_{x:\,\mathtt{Syn}x=\sigma} (\frac{p}{1-p})^{w(x)}.
\end{aligned}$$

The $\max_x$ selects the smallest weight $w(x)$. Among the strings $x \in \{0,1\}^n$ that have syndrome σ (a coset), the one with the lowest Hamming weight is called the coset leader. For each weight a, there are possibly multiple cosets whose leader has weight a. The *coset leader weight enumerator*, denoted as c_a, counts how many cosets have a leader of weight a. The $\sum_\sigma$ summation in the expression above can be written in terms of the c_a,

$$\sum_\sigma \max_{x:\,\mathtt{Syn}x=\sigma} (\frac{p}{1-p})^{w(x)} = \sum_{a\geq 0} c_a (\frac{p}{1-p})^a. \tag{26}$$

The code can correct t errors, so for $a \leq t$ it holds that $c_a = \binom{n}{a}$. A perfect code has $c_a = 0$ for $a > t$. We consider codes that are far from perfect. We split $\sum_\sigma$, which has 2^{n-k} terms, into a part with coset leader weights $\leq t$ and a part with weights $> t$. In the latter part we write $(\frac{p}{1-p})^{w(x)} \leq (\frac{p}{1-p})^{t+1}$. This yields

$$\begin{aligned}
&\mathsf{H}_{\min}(X|\mathtt{Syn}\,X) - \mathsf{H}_{\min}(X) \geq \\
&-\log \left[\sum_{a=0}^{t} \binom{n}{a} (\frac{p}{1-p})^a + (\frac{p}{1-p})^{t+1} \{2^{n-k} - \sum_{a=0}^{t} \binom{n}{a}\} \right].
\end{aligned} \tag{27}$$

For a far-from-perfect code, the dominant term in the logarithm is $(\frac{p}{1-p})^{t+1} 2^{n-k}$. We write the logarithm term in (27) as $-\log[(\frac{p}{1-p})^{t+1} 2^{n-k}(1+\varepsilon)]$, with $\varepsilon \ll 1$, which equals $-(n-k) + (t+1)\log\frac{1-p}{p} - \log(1+\varepsilon)$. Finally we apply $-\log(1+\varepsilon) \geq -\varepsilon$.

References

1. `http://gjrand.sourceforge.net/`.
2. `http://www.digicortex.net/node/22`.
3. E.F. Beckenbach. *Applied combinatorial mathematics*. John Wiley and sons, 1964.
4. C.H. Bennett, G. Brassard, C. Crépeau, and M. Skubiszewska. Practical quantum oblivious transfer. In *CRYPTO*, pages 351–366, 1991.
5. C. Böhm and M. Hofer. *Physical Unclonable Functions in Theory and Practice*. Springer, 2013.
6. X. Boyen. Reusable cryptographic fuzzy extractors. In *ACM Conference on Computer and Communications Security*, pages 82–91, 2004.
7. R. Canetti, B. Fuller, O. Paneth, L. Reyzin, and A. Smith. Reusable fuzzy extractors for low-entropy distributions. In *Eurocrypt 2016*, 2016. `eprint.iacr.org/2014/243`.
8. T.M. Cover and J.A. Thomas. *Elements of Information Theory*. John Wiley & Sons, Inc., second edition, 2005.
9. Y. Dodis, R. Ostrovsky, L. Reyzin, and A. Smith. Fuzzy Extractors: how to generate strong keys from biometrics and other noisy data. *SIAM J. Comput.*, 38(1):97–139, 2008.
10. Y. Dodis, M. Reyzin, and A. Smith. Fuzzy Extractors: How to generate strong keys from biometrics and other noisy data. In *Eurocrypt 2004*, volume 3027 of *LNCS*, pages 523–540. Springer-Verlag, 2004.
11. R. Durstenfeld. ACM Algorithm 235: Random Permutation. *Communications of the ACM*, 7(7):420, 1964.
12. B. Gassend. Physical Random Functions. Master's thesis, Massachusetts Institute of Technology, 2003.
13. T. Ignatenko and F.M.J. Willems. Information leakage in fuzzy commitment schemes. *IEEE Transactions on Information Forensics and Security*, 5(2):337–348, 2010.
14. A. Juels and M. Wattenberg. A fuzzy commitment scheme. In *ACM Conference on Computer and Communications Security (CCS) 1999*, pages 28–36, 1999.
15. P. Koeberl, J. Li, A. Rajan, and W. Wu. Entropy loss in PUF-based key generation schemes: The repetition code pitfall. In *IEEE International Symposium on Hardware-Oriented Security and Trust (HOST) 2014*, pages 44–49, 2014.
16. R. Maes. *Physically Unclonable Functions: Constructions, Properties and Applications*. Springer, 2013.
17. R. Maes, V. van der Leest, E. van der Sluis, and F.M.J. Willems. Secure key generation from biased PUFs. In *Cryptographic Hardware and Embedded Systems (CHES) 2015*, volume 9293 of *LNCS*, pages 517–534. Springer, 2015.
18. A.-R. Sadeghi and D. Naccache, editors. *Towards hardware-intrinsic security*. Springer, 2010.
19. P. Tuyls, G.-J. Schrijen, B. Škorić, J. van Geloven, R. Verhaegh, and R. Wolters. Read-proof hardware from protective coatings. In *Cryptographic Hardware and Embedded Systems (CHES) 2006*, volume 4249 of *LNCS*, pages 369–383. Springer-Verlag, 2006.
20. P. Tuyls, B. Škorić, and T. Kevenaar. *Security with Noisy Data: Private Biometrics, Secure Key Storage and Anti-Counterfeiting*. Springer, London, 2007.
21. B. Škorić and N. de Vreede. The Spammed Code Offset Method. *IEEE Transactions on Information Forensics and Security*, 9(5):875–884, May 2014.
22. M.-D. Yu and S. Devadas. Secure and robust error correction for Physical Unclonable Functions. *IEEE Design & Test of Computers*, 27(1):48–65, 2010.