Pseudoentropic Isometries: A New Framework for Fuzzy Extractor Reusability

Quentin Alamélou[1], Paul-Edmond Berthier[1], Chloé Cachet[1], Stéphane Cauchie[1], Benjamin Fuller[2], Philippe Gaborit[3], and Sailesh Simhadri[2]

[1] equensWorldline, FPL-ITA-MSC, Seclin, France,
firstname.lastname@equensworldline.com,
[2] University of Connecticut, Storrs, CT, United States,
firstname.lastname@uconn.edu,
[3] Université de Limoges, XLIM-DMI, Limoges, France,
philippe.gaborit@xlim.fr,

Abstract. Fuzzy extractors (Dodis *et al.*, Eurocrypt 2004) turn a noisy secret into a stable, uniformly distributed key. *Reusable* fuzzy extractors remain secure when multiple keys are produced from a single noisy secret (Boyen, CCS 2004). Boyen proved that any information-theoretically secure reusable fuzzy extractor is subject to strong limitations. Simoens *et al.* (IEEE S&P, 2009) then showed deployed constructions suffer severe security breaks when reused. Canetti *et al.* (Eurocrypt 2016) proposed using computational security to sidestep this problem. They constructed a computationally secure reusable fuzzy extractor for the Hamming metric that corrects a *sublinear* fraction of errors.

We introduce a generic approach to constructing reusable fuzzy extractors. We define a new primitive called a *reusable pseudoentropic isometry* that projects an input metric space to an output metric space. This projection preserves distance and entropy even if the same input is mapped to multiple output metric spaces. A reusable pseudoentropy isometry yields a reusable fuzzy extractor by 1) randomizing the noisy secret using the isometry and 2) applying a traditional fuzzy extractor to derive a secret key.

We propose reusable pseudoentropic isometries for the set difference and Hamming metrics. The set difference construction is built from composable digital lockers (Canetti and Dakdouk, Eurocrypt 2008) yielding the first reusable fuzzy extractor that corrects a *linear* fraction of errors. For the Hamming metric, we show that the second construction of Canetti *et al.* (Eurocrypt 2016) can be seen as an instantiation of our framework. In both cases, the pseudoentropic isometry's reusability requires noisy secrets distributions to have entropy in each symbol of the alphabet.

Lastly, we implement our set difference solution and describe two use cases.

1 Introduction

Cryptography relies on uniformly distributed and reproducible long-term secrets to perform authentication or derive keys. Numerous high entropy randomness sources exist, such as biometrics and human-generated data [20,36], physically unclonable functions (PUFs) [50] and quantum information [8]. Many of these sources exhibit errors when read multiple times, preventing stable cryptographic key generation. The errors of each physical phenomena implicitly define a metric space.

Dodis *et al.* [23] stated that Hamming distance looks like the "most natural metric to consider" [16,23,37]. Set distance better suits some biometric matchers such as human and digital fingerprints and the exotic movie lover's problem [36]. Typical systems create a *template* reading from an initial reading; subsequent readings are directly compared to this initial template. A subsequent reading is accepted if the two readings are "close" according to the distance metric. Plaintext templates have privacy concerns [51,54]. In the worst case, a matching biometric can be reverse engineered from the template [28].

Information reconciliation [8] enables retrieving identical values from noisy data. *Privacy amplification* [8] converts values with entropy into uniform random strings. Fuzzy Extractors [23,22], are a pair of non-interactive algorithms (Gen, Rep) that simultaneously perform information reconciliation and privacy amplification. The algorithm Gen, used at enrollment, takes input ω from an entropy source and outputs a uniformly distributed key R and a public helper string P. The algorithm Rep takes P and ω' and reproduces the secret key R as long as ω' is close enough to ω, specifically $d(\omega, \omega') \leq t$ for a fixed parameter t. Fuzzy extractors exist with security against information-theoretic [22] or computational adversaries [27].

Dodis *et al.* proposed fuzzy extractor constructions for the Hamming, set difference and edit metrics adapting prior work [37,36]. We focus on the set difference metric: inputs ω are subsets of size s of a universe $\mathcal{U}$ whose cardinality is n. For this metric, Dodis *et al.* distinguished two settings, referred to as the *small* and *large* universe settings. Let λ be some security parameter. In the former case, we have that $n = \mathtt{poly}(\lambda)$ while in the latter one n is superpolynomial in s. The large universe setting occurs in practice. For example, consider a list of book titles or a list of movies (movie lover's problem due to [36]). The small universe setting benefits from a reduction to the Hamming metric, referred to as the bin-set equivalence (described in Section 2). We concentrate on the large universe setting where this transform is not applicable.

Reusability Boyen introduced *reusable* fuzzy extractors for which numerous helper strings P^j from a user's fuzzy secret do not impact user's security [13]. Boyen showed that information-theoretic fuzzy extractors must leak substantial information about ω when numerous calls to Gen are made. On the positive side, Boyen demonstrated reusable security when the exclusive OR of pairs of enrolled values reveals no sensitive information. This is a restrictive class of correlations; we have no evidence that practical sources obey this condition. We call this type of construction *weakly* reusable. Subsequent work showed that existing fuzzy extractors are not reusable in practice [10,52]. Apon *et al.* [6] added reusability to the *learning-with-errors* based fuzzy extractor of Fuller *et al.* [27]. Their work inherits the small error tolerance of the latter construction [27]; it only corrects a logarithmic fraction of errors, that is, for input ω, $t = O(\log |\omega|)$.

Recently, Canetti *et al.* [16] constructed a reusable fuzzy extractor that makes no assumption about how repeating readings are correlated. We call this type of construction *strongly* reusable. It works for the Hamming distance and provides security against computationally bounded adversaries. It uses a strong form of symmetric encryption, called *digital lockers* [15] (our construction also uses digital lockers but in a different way).

Their construction is secure for distributions with high entropy samples instead of global min-entropy. This is in contrast to traditional constructions that only assume the source has min-entropy. Their main Hamming construction can be extended through bin-set equivalence to a small set universe set difference fuzzy extractor. Their scheme allows an error rate $t = o(|\omega|)$.

Prior to this work, there were no known strongly reusable fuzzy extractors correcting a linear error rate for any common metric.

1.1 Our contributions

Most upper bounds on the security of information-theoretic (reusable) fuzzy extractors are due to strong connections to the field of coding theory (see for example [21]). The best security of a information-theoretic fuzzy extractor allowing t errors is connected to the code with the most

Scheme	Reusability	Error Tolerance	Metric
RPI + Fuzz. Ext. (this work)	Set difference/Hamming	Strong	$\Theta(\|w\|)$
Sample-then-lock [16]	Hamming	Strong	$o(\|w\|)$
Code Offset [13]	Hamming	Weak	$\Theta(\log \|w\|)$
LWE Decoding [6]	Hamming	Weak	$\Theta(\|w\|)$

Table 1. Recent constructions of reusable fuzzy extractors. The code offset and LWE decoding schemes are weakly reusable. They may leak information about the difference between repeated readings w_i, w_j.

keywords correcting t errors. Coding theory has a long history with established bounds on the best codes.

To circumvent these results, we propose a new framework to achieve reusability. The main idea is to separate the task of reusability from the task of noise elimination. Our contributions are as follows:

1. We introduce a randomization stage captured by a new primitive we call a *pseudoentropic isometry*. Informally, a pseudoentropic isometry projects fuzzy secrets while maintaining distances between two noisy readings and entropy of the original secret. To be *reusable*, a pseudoentropic isometry must generate "uncorrelated" values Ω^i from correlated noisy enrollments values of the same secret. The reusability property is that each Ω^i has sufficient entropy conditioned on knowledge of the other $\Omega^j, j \neq i$. *Reusable pseudoentropic isometries* (RPIs) do not perform any form of error correction. We do not believe they are subject to bounds from coding theory. First passing the input through an RPI makes a fuzzy extractor reusable.
2. We propose two reusable pseudoentropic isometries. The first one is an original construction with an RPI based on digital lockers [15]. This construction is in the set difference metric with a large universe. Roughly for each element of the set the construction samples a point in the new metric space and locks this point using the set element as the key.
   We then show that Construction 2 of Canetti *et al.* [16, Section 5.1] fits our framework in the Hamming metric. Both instantiations require sources with superlogarithmic entropy (in each alphabet symbol) to achieve reusability. We compare these constructions with previous reusable fuzzy extractors in Table 1.
3. We provide an implementation of our set difference RPI and describe two use cases suited to our constructions.

Notes: To the best of our knowledge all computational fuzzy extractors information-theoretically determine the key. That is, it is possible for an unbounded adversary to search through possible input values and exclude them based on the public value P. However, this does not break security as a computationally bounded adversary can only check a polynomial number of values. This is in contrast to information-theoretic fuzzy extractors that do not confirm correctness of a guess (information-theoretic constructions ensure that ω has high entropy conditioned on P). We are not aware of any constructions are both secure against information-theoretic adversaries and provide additional security against computational adversaries (such as reusability, better key length).

Digital lockers are a strong cryptographic tool[4] and our construction requires substantial structure on the input source ω. Reusable fuzzy extractors are notoriously difficult to construct. Reducing the required assumption to build an RPI is the primary future direction of this work.

[4] They require a strong assumption on the underlying cryptographic primitives. However, they can be efficiently implemented using Diffie-Hellman groups or hash functions.

2 Preliminaries

Notation $\log$ denotes the base 2 logarithm. $GF(n)$ denotes the finite field of n elements. $x \leftarrow f(.)$ denotes that x is an output of a function f. If f is randomized, we use the semicolon to make the randomness explicit. $f(x;\mu)$ is the result of f computed on x with randomness μ. U_ℓ denotes the uniformly distributed random variable on $\{0,1\}^l$. For a distinguisher D (or a class of distinguishers $\mathcal{D}$), we write the computational distance between X and Y as $\delta^D(X,Y) = |\mathbb{E}[D(X)] - \mathbb{E}[D(Y)]|$. $\mathcal{D}_{s_{\text{sec}}}$ denotes the class of randomized circuits which output a single bit and have size at most s_{sec}. We write $\delta^{\mathcal{D}_{s_{sec}}}(X,Y) \leq \epsilon$ if this is true for all $D \in \mathcal{D}_{s_{sec}}$.

A metric space is a finite set $\mathcal{M}$ equipped with a distance $d : \mathcal{M} \times \mathcal{M} \rightarrow \mathbb{N}$ fulfilling the properties of symmetry, triangle inequality and zero distance between equal points.

Set Difference Metric Let $\mathcal{M}$ consist of all subsets of a universe $\mathcal{U}$ whose cardinality is n. For two sets ω and ω' belonging to $\mathcal{M}$, their symmetric difference is defined as $\omega \Delta \omega' \stackrel{def}{=} \{x \in \omega \cup \omega' | x \notin \omega \cap \omega'\}$. Symmetric difference is a metric that we denote by d.

Dodis *et al.* [23] noted the bin-set equivalence: if ω denotes a set, it can be viewed a binary vector in $\{0,1\}^n$, with 1 at position x if $x \in \omega$, and 0 otherwise. Viewed in this way, set difference can be expressed as Hamming distance between these associated vectors. This transform is not efficient when the universe size n is superpolynomial.

Entropy Notions Entropy specifies the amount of information contained in some data. In security-related contexts, we care about how well an adversary can guess the value of a random variable. In the information-theoretic case, we rely on the notion of *min-entropy*. A random variable A has min-entropy m, denoted $H_\infty(A) = m$, if

$$H_\infty(A) \stackrel{\text{def}}{=} -\log(\max_{a \in A} P[A = a]).$$

The *average* min-entropy of A given B is:

$$\tilde{H}_\infty(\mathrm{A}|\mathrm{B}) \stackrel{\text{def}}{=} -\log(\mathbb{E}_{b \in \mathrm{B}} \mathrm{max}_a \mathrm{Pr}[\mathrm{A} = a|\mathrm{B} = b]).$$

HILL entropy is a commonly used computational notion of entropy [34]. It was extended to the conditional case by Hsiao, Lu, and Reyzin [35].

Definition 1. *Let (W,S) be a pair of random variables. W has* HILL entropy *at least k conditioned on S, denoted $H^{\mathtt{HILL}}_{\epsilon_{sec},s_{sec}}(W|S) \geq k$ if there exists a collection of distributions X_s (defined for each value s in the support of S) giving rise to a joint distribution (X,S), such that $\tilde{H}_\infty(X|S) \geq k$ and*

$$\delta^{\mathcal{D}_{s_{sec}}}((W,S),(X,S)) \leq \epsilon_{sec}.$$

Fuzzy Extractors The original definition of fuzzy extractors provides information-theoretic security. We state the computational definition introduced by Fuller *et al.* [27].

Definition 2 (Fuzzy Extractor). *A pair of randomized procedures "generate" (Gen) and "reproduce" (Rep) is a $(\mathcal{M},\mathcal{W},\ell,t,\gamma)$-(computational) fuzzy extractor that is $(\epsilon_{sec}, s_{sec})$-hard if Gen and Rep satisfy the following properties:*

- *Gen on input $\omega \in \mathcal{M}$ outputs an extracted string $R \in \{0,1\}^\ell$ and a helper string $P \in \{0,1\}^*$.*
- *Rep takes an element $\omega' \in \mathcal{M}$ and a bit string $P \in \{0,1\}^*$ as inputs.*
- ***Correctness:** for all ω, ω' where $d(\omega, \omega') \leq t$,*

$$\Pr[\mathsf{Rep}(\omega', P) = R : (R, P) \leftarrow \mathsf{Gen}(\omega)] \geq 1 - \gamma$$

where the probability is over the coins of Gen and Rep.
- ***Security:** for each distribution $W \in \mathcal{W}$,*

$$\delta^{s_{sec}}((R, P), (U_\ell, P)) \leq \epsilon_{sec}.$$

This definition can be naturally extended to support auxiliary input I: the security property then requires that $\delta^{s_{\text{sec}}}((R, P, I), (U_\ell, P, I)) \leq \epsilon_{\text{sec}}$ appear indistinguishable.

In prior definitions, the family of distributions $\mathcal{W}$ was all distributions with a given amount of entropy, known computational constructions require additional structure on the source beyond entropy. When we consider fuzzy extractors that are secure for all distributions of (average) min-entropy m, we replace $\mathcal{W}$ with the parameter m.

Dodis *et al.* designed FEs based on three different metrics which are Hamming, set difference and edit distances. All their constructions rely on *secure sketches*.

Definition 3. *An $(\mathcal{M}, \mathcal{W}, \tilde{m}, t, \gamma)$-*secure sketch *is a pair of randomized procedures:*

1. *The sketching procedure SS on input $\omega \in \mathcal{M}$ returns a bit string $s \in \{0,1\}^*$.*
2. *The recovery procedure Rec takes an element $\omega' \in \mathcal{M}$ and a bit string $s \in \{0,1\}^*$.*
3. Correctness *guarantees that if $d(\omega, \omega') \leq t$, then*

$$\Pr[\mathsf{Rec}(\omega', \mathsf{SS}(\omega)) = \omega] \geq 1 - \gamma$$

where the probability is taken over the coins of SS and Rec.
4. *Security: for any distribution $W \in \mathcal{W}$, $\tilde{\mathrm{H}}_\infty(W|\mathsf{SS}(W)) \geq \tilde{m}$.*

Roughly, a secure sketch performs error correction without leaking information while a fuzzy extractor also yields a uniform value. The sketch-then-extract construction [23] combines a secure sketch and an *average-case extractor* [48] to build a fuzzy extractor.

Reusable Fuzzy Extractor Reusability allow multiple calls to Gen on the noisy readings of ω while retaining security [13]. Consider ρ readings $\omega^1, \ldots, \omega^\rho$ of the same fuzzy source from which the user will be enrolled on ρ different authentication servers. Gen independently generates ρ pairs $(R^1, P^1), \ldots, (R^\rho, P^\rho)$ where $(R^j, P^j) \leftarrow \mathsf{Gen}(\omega^j)$.

Definition 4 (Reusable Fuzzy Extractor [16]). *Let $(\mathsf{Gen}, \mathsf{Rep})$ be a $(\mathcal{M}, \mathcal{W}, l, t, \gamma)$-fuzzy extractor that is $(\epsilon_{sec}, s_{sec})$-hard. Let W^1, W^2, $\ldots$, W^ρ be ρ arbitrarily correlated random variables where $W^j \in \mathcal{W}$ for all $1 \leq j \leq \rho$. Define the random variables $(R^i, P^i) \leftarrow \mathsf{Gen}(W^i)$. $(\mathsf{Gen}, \mathsf{Rep})$ is $(\epsilon_{sec}, s_{sec,\rho})$-reusable if for all $D \in \mathcal{D}_{s_{sec}}$ and for all $j = 1, \ldots, \rho$:*

$$\begin{aligned} &\Pr[D(R^1, \ldots, R^\rho, \{P^j\}_{1 \leqslant j \leqslant \rho}) = 1] \\ -&\Pr[D(R^1, \ldots, R^{j-1}, U_\ell, R^{j+1}, \ldots, R^\rho, \{P^j\}_{1 \leqslant j \leqslant \rho}) = 1] \leq \epsilon_{sec}. \end{aligned}$$

2.1 Tools

Digital Lockers Digital lockers are secure symmetric encryption schemes that retain security even when used multiple times with correlated and nonuniform keys [17]. Furthermore, an incorrect key can also be recognized with high probability. We use notation $\mathsf{c} = \mathsf{lock}(\mathsf{key}, \mathsf{val})$ for the algorithm that performs the locking of the value val using key, and $\mathsf{unlock}(\mathsf{key}, \mathsf{c})$ for the algorithm that performs the unlocking (which will output val if key is correct and $\perp$ with high probability otherwise).

Digital lockers can be constructed in the random oracle (see Lynn, Prabhakaran, and Sahai [42, Section 4]). Bitansky and Canetti [9], building on the work of [15,17], show how to obtain composable digital lockers based on a version of the Decisional Diffie-Hellman assumption without random oracles. Furthermore, it is possible that cryptographic hash functions satisfy the required functionality without resorting to the random oracle model.

We consider security via virtual-grey-box simulatability [9], a simulator is allowed unbounded running time but only a bounded number of queries to an ideal locker. Intuitively, the definition says if the keys to the ideal locker are hard to guess, the simulator will not be able to unlock the ideal locker and thus neither will the real adversary. Formally, let $\mathsf{idealUnlock}(\mathsf{key}, \mathsf{val})$ be the oracle that returns val when given key, and $\perp$ otherwise.

Definition 5 (Digital Lockers). *The pair* $(\mathsf{lock}, \mathsf{unlock})$ *with security parameter* λ *is an* ℓ*-composable secure digital locker with error* γ *if the following hold:*

- ***Correctness*** *For all* key *and* val*,*

$$\Pr[\mathsf{unlock}(\mathsf{key}, \mathsf{lock}(\mathsf{key}, \mathsf{val})) = \mathsf{val}] \geq 1 - \gamma.$$

- ***Wrong key detection*** *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* $z \in \{0,1\}^*$*,*

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

where S *is allowed* $q(\lambda)$ *oracle queries.*

Obfuscated Point Functions Canetti *et al.*'s construction for large alphabets [16, Section 5.1] uses a weaker primitive called an obfuscated point function. This primitive can be viewed as a digital locker without a plaintext: it outputs 1 if the key is correct, 0 otherwise. Such a function can be constructed from the above mentioned digital locker with a single possible plaintext, or from a version of the Decisional Diffie-Hellman assumption [14]. We use notation $c = \mathsf{lockPoint}(key)$ and $\mathsf{unlockPoint}(key, c)$. Point functions security is defined the same way as for digital lockers with a fixed plaintext.

3 Reusability Pseudoentropic Isometries

Reusable fuzzy extractors combine entropy extraction and error-correction without leaking information to achieve reusability. Our approach is to separate reusability from entropy extraction and error-correction. The idea is to randomly project the fuzzy secrets into a new metric space before applying a nonreusable fuzzy extractor. If we can build a randomized projection that creates unrelated values, a standard fuzzy extractor should not leak information across projected values. We call this randomization stage a pseudoentropic isometry:

Definition 6 (Pseudoentropic isometry). *Let $(\mathcal{M}_1, d_1), (\mathcal{M}_2, d_2)$ be two metric spaces. A $(\mathcal{M}_1, \mathcal{M}_2, \mathcal{W}, m_2, \epsilon_{sec}, s_{sec}, \gamma)$-pseudoentropic isometry is a pair of randomized procedures (RPIGen, RPIRep) with the following properties:*

1. *RPIGen on $\omega \in \mathcal{M}_1$ outputs $\Omega \in \mathcal{M}_2$ and $f \in \{0,1\}^*$.*
2. *RPIRep takes an element $\omega' \in \mathcal{M}_1$ and a bit string $f \in \{0,1\}^*$ as inputs to output $\Omega' \in \mathcal{M}_2$.*
3. ***Correctness:** if $(\Omega, F) \leftarrow$ RPIGen(ω), then $\Pr[d_2(\Omega, \Omega') = d_1(\omega, \omega')] \geq 1 - \gamma$. where the probability is over the randomness of (RPIGen, RPIRep).*
4. ***Security:** for any $W \in \mathcal{W}$, for $(R, F) \leftarrow$ RPIGen(W) we have $H^{\mathtt{HILL}}_{\epsilon_{sec}, s_{sec}}(R|F) \geq m_2$.*

Note: Security implies that $H^{\mathtt{HILL}}(W|V) \geq m_2$ with a slight loss in parameters as any adversary that can recover W can use RPIRep to recover U.

On their own pseudoentropic isometries are not novel (the identity function is a pseudoentropic isometry). A reusable pseudoentropic isometry or RPI is the key to our approach. To be reusable, a pseudoentropic isometry generates ρ uncorrelated values $\Omega^1, \ldots, \Omega^\rho$ from ρ enrollments values $\omega^1, \ldots, \omega^\rho$ drawn from the fuzzy secret ω.

Definition 7 (RPI). *Let $W^* \in \mathcal{W}$ be some distribution. Let $W^1, W^2, \ldots, W^\rho$ be ρ arbitrarily correlated random variables over $\mathcal{M}_1$ where each $W^i \in \mathcal{W}$. Define the random variables $(\Omega^i, F^i) \leftarrow$ RPIGen(W^i) and $(\Omega^*, F^*) \leftarrow$ RPIGen(W^*). (RPIGen, RPIRep) is $(\epsilon_{sec}, s_{sec,\rho})$-reusable if for all $j = 1, \ldots, \rho$:*

$$\delta^{\mathcal{D}_{s_{sec}}}((\Omega^1, \ldots, \Omega^\rho, F^1, \ldots, F^\rho), \\ (\Omega^1, \ldots, \Omega^{j-1}, \Omega^*, \Omega^{j+1}, \ldots, \Omega^\rho, F^1, \ldots, F^\rho)) \leq \epsilon_{sec}$$

Following this randomization stage, we can apply a traditional fuzzy extractor on the uncorrelated randomized values Ω^js. This idea is depicted in Figure 1.

Within this framework, the FE is always applied on unrelated values Ω^js. Even when reenrolling a fingerprint ω, the generated helper values P^js only yield information on the decorrelated values Ω^js and not on the original noisy secret. Thus, the combination of an RPI with a traditional, nonreusable, fuzzy extractor yields an RFE.

Relation to other fuzzy extractor primitives A pseudoentropic isometry is related to biometric embeddings used in [22]. A biometric embedding projects any fingerprint value into a metric space where a fuzzy extractor exists while loosely maintaining distances.

If we consider a slightly different primitive that is allowed to reduced distances $d_2(\Omega, \Omega') \leq d_1(\omega, \omega')$, this primitive can be constructed using either a fuzzy extractor or a secure sketch. However, the connection is not clear when distance must be maintained precisely.

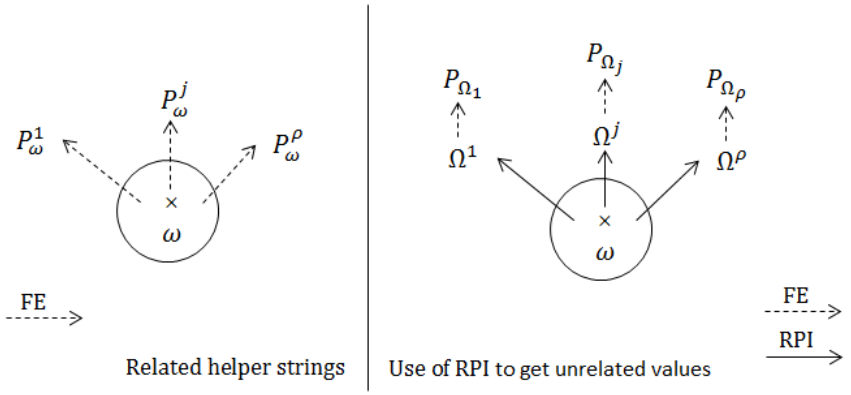

Fig. 1. Overview of reusability framework

Generation procedure Gen	**Reproduction procedure Rep**
Input: $\omega \in \mathcal{M}_1$.	Inputs: $\omega' \in \mathcal{M}_1$,
1. $(\Omega, F) \leftarrow$ RPIGen(ω).	Helper data $P \in \{0,1\}^*$.
2. $(R, Q) \leftarrow$ Gen'(Ω).	1. Parse $P = (F, Q)$
3. Set $P = (F, Q)$.	2. $\Omega' \leftarrow$ RPIRep(ω', F).
4. Return (R, P).	3. $R \leftarrow$ Rep'(Ω', Q).
	4. Return R.

Fig. 2. A generic reusable FE

3.1 RPIs imply reusable fuzzy extractors

Let (Gen', Rep') denote a (*average-case*) nonreusable fuzzy extractor. The generation procedure Gen' implicitly draws a ball $\mathcal{B}(\omega, t)$ centered on its input ω where the radius t is the error tolerance of the fuzzy extractor. Whenever a noisy reading ω' is given to procedure Rep', the secret key will be recovered as long as ω' belongs to $\mathcal{B}(\omega, t)$.

To address reusability, we randomly project the ρ fuzzy versions of ω onto unrelated values so that each of these retains original entropy independently of others. By using a ρ-RPI, the user gets unrelated values $\Omega^1, \ldots, \Omega^\rho$ that will be each enrolled once, respectively toward servers $1, \ldots, \rho$.

Let (RPIGen, RPIRep) be a ρ-RPI from $\mathcal{M}_1$ to $\mathcal{M}_2$. Let (Gen', Rep') be an average-case FE over $\mathcal{M}_2$ correcting t errors. The generation procedure Gen will first call RPIGen to randomize the input ω into Ω. The nonreusable FE is then applied on Ω. The RPI ensures that $d_2(\Omega, \Omega') = d_1(\omega, \omega')$ while the correctness of the underlying nonreusable FE ensures that Rep' recovers R from Ω' and the associated helper string as long as $d_2(\Omega, \Omega') \leqslant t$. Overall this leads to recovering R as long as $d_1(\omega, \omega') \leqslant t$.

Theorem 1. *Suppose that (RPIGen, RPIRep) is a* $(\mathcal{M}_1, \mathcal{M}_2, \mathcal{W}, m_2, \epsilon_{RPI}, s_{RPI}, \gamma_{RPI})$*-RPI that is* ρ*-reusable and (Gen', Rep') be an average-case* $(\mathcal{M}_2, m_2, \ell, t, \gamma_{FE})$*-fuzzy extractor that is* (ϵ_{FE}, s_{FE})*-hard.*

Figure 2 defines a $(\mathcal{M}_1, \mathcal{W}, l, t, \gamma_{RPI} + \gamma_{FE})$*-fuzzy extractor that is* $(\rho, \epsilon_{sec}, s_{sec})$*-reusable for* $\epsilon_{sec} = 4\epsilon_{RPI} + \epsilon_{FE}$ *and* $s_{sec} = \min\{s_{RPI} - |\mathsf{Gen}'|, s_{FE}\}$.

Proof. The correctness is straightforward and follows from aforesaid explanations. To ensure security, we first show that R is pseudorandom knowing P and then treat reusability.

Under notation of Definition 6, we $H^{\mathtt{HILL}}_{\epsilon_{\mathsf{RPI}},s_{\mathsf{RPI}}}(\Omega|F) \geqslant m_2$. We first show that fuzzy extractors work on distributions with HILL entropy. The proof is straightforward and delayed until Appendix A.1.

Lemma 1. *Suppose that U, V are a joint distribution where $H^{\mathtt{HILL}}_{\epsilon_{RPI},s_{RPI}}(U|V) \geq m_2$ and let (Gen', Rep') be an average-case $(\mathcal{M}_2, m_2, l, t)$-fuzzy extractor that is (ϵ_{FE}, s_{FE})-hard. Define $(R, P) \leftarrow \mathsf{Gen}'(U)$, then*

$$\delta^{\mathcal{D}_s}((R, P, V), (U_l, P, V)) \leqslant \epsilon.$$

for $\epsilon = 2\epsilon_{RPI} + \epsilon_{FE}$ and $s = \min\{s_{RPI}, s_{FE}\}$.

Lemma 1 allows us to conclude that $\delta^{\mathcal{D}_s}((R, Q, F), (U_l, Q, F)) \leqslant \epsilon$. That is,

$$\delta^{\mathcal{D}_s}((R, P), (U_l, P)) \leqslant \epsilon$$

for $P = (F, Q)$, and aforesaid parameters $\epsilon = 2\epsilon_{\mathsf{RPI}} + \epsilon_{\mathsf{FE}}$, $s = \min\{s_{\mathsf{RPI}}, s_{\mathsf{FE}}\}$. That is, Figure 2 describes a fuzzy extractor.

Reusability Let $W^1, \ldots, W^\rho$ be correlated distributions over $\mathcal{M}_1$, where $W^j \in \mathcal{W}$ for all j. The following games are between a challenger $\mathcal{C}$ and D for some fixed i_0:

$\mathcal{G}_0$ $\mathcal{C}$ honestly samples values as prescribed in Definition 4 and sends

$$(R^1, F^1, Q^1), \ldots, (R^{i_0}, F^{i_0}, Q^{i_0}), \ldots, (R^\rho, F^\rho, Q^\rho)$$

to D.

$\mathcal{G}_1$ $\mathcal{C}$ now does the following:

1. Samples each ω^js and uses RPIGen to obtain $(\Omega^1, F^1), \ldots, (\Omega^\rho, F^\rho)$.
2. Replaces ω^{i_0} with $\omega^* \leftarrow W^*$.
3. Computes $(\Omega^*, F^*) \leftarrow \mathsf{RPIGen}(\omega^*)$, $(R^*, Q^*) \leftarrow \mathsf{Gen}'(\Omega^*)$.
4. Sets $P^* = (F^{i_0}, Q^*)$.
5. Gives D R^js and P^js except for $j = i_0$ for which he receives (R^*, P^*).

If D can distinguish this game from the previous one, he would then be able to distinguish the distribution with Ω^{i_0} from the one with Ω^*. This breaks the reusability of the RPI. That is, $\mathcal{G}_1$ appears indistinguishable from $\mathcal{G}_0$ for $\epsilon = \epsilon_{\mathsf{RPI}}$ and $s = s_{\mathsf{RPI}} - |\mathsf{Gen}'|$.

$\mathcal{G}_2$ In this game, after computing $(R^*, Q^*) \leftarrow \mathsf{Gen}'(\Omega^*)$, $\mathcal{C}$ discards R^* and replaces it with randomly sampled $\eta \xleftarrow{\$} \{0,1\}^\ell$ randomly sampled. Since $H^{\mathtt{HILL}}_{\epsilon_{\mathsf{RPI}},s_{\mathsf{RPI}}}(\Omega^*|F^*) \geq m_2$ then

$$H^{\mathtt{HILL}}_{\epsilon_{\mathsf{RPI}},s_{\mathsf{RPI}}}(\Omega^*|F^{i_0}) \geq m_2.$$

Thus by Lemma 1, (U_l, P^*) and (R^*, P^*) are computationally indistinguishable. Hence, this game is indistinguishable from the previous one for $\epsilon = 2\epsilon_{\mathsf{RPI}} + \epsilon_{\mathsf{FE}}$ and $s = \min\{s_{\mathsf{RPI}}, s_{\mathsf{FE}}\}$.

$\mathcal{G}_3$ In the previous game, D is given $(R^1, F^1, Q^1), \ldots, (\eta, F^{i_0}, Q^*), \ldots, (R^\rho, F^\rho, Q^\rho)$ where η is random and does not depend on P^*. In this game, $\mathcal{C}$ sends the actual Q^{i_0} (obtained via computed $\mathsf{Gen}'(\Omega^{i_0})$ instead of Q^*.

If D can distinguish that Q^{i_0} has been given instead of Q^* (obtained via computed $\mathsf{Gen}'(\Omega^*)$, he can in particular distinguish Ω^{i_0} from Ω^*. Hence, he can distinguish

$$(\Omega^1, \ldots, \Omega^{i_0}, \ldots, \Omega^\rho, F^1, \ldots, F^{i_0}, \ldots, F^\rho)$$

from

$$(\Omega^1, \ldots, \Omega^{i_0-1}, \Omega^*, \Omega^{i_0+1}, \ldots, \Omega^\rho, F^1, \ldots, F^{i_0}, \ldots, F^\rho).$$

This contradicts the reusability of the RPI. Thus, $\mathcal{G}_3$ is indistinguishable from $\mathcal{G}_2$ for $\epsilon = \epsilon_{\text{RPI}}$ and $s = s_{\text{RPI}} - |\mathsf{Gen}'|$.

In $\mathcal{G}_3$, D is given $(R^1, P^1), \ldots, (\eta, P^{i_0}), \ldots, (R^\rho, P^\rho)$ where η is randomly sampled. By transitivity, this latter game is indistinguishable from $\mathcal{G}_0$. Indistinguishability between $\mathcal{G}_0$ and $\mathcal{G}_3$ satisfies the requirements of Definition 4.

4 Instantiating the framework

We propose two instantiations of the above mentioned framework, the first one for the set difference metric and the second one for the Hamming distance.

4.1 Set Difference-based Instantiation

Environment and Notation Set difference based FEs in [22] take as inputs subsets of a universe $\mathcal{U}$ with $n = |\mathcal{U}|$. We denote $(\mathcal{M}_\mathcal{U}, d)$, the metric space $\mathcal{M}_\mathcal{U}$ consisting of all the subsets of $\mathcal{U}$ with the set difference metric d. Let $\mathcal{M}_{\mathcal{U},s}$ denote the restriction of $\mathcal{M}_\mathcal{U}$ to s-elements subsets. $\mathcal{M}_\kappa$ denotes $(GF(2^\kappa), d)$ equipped with the set difference metric d. Similarly $\mathcal{M}_{\kappa,s}$ denotes the restriction to sets of sizes s. Let W be a probability distribution over $\mathcal{U}$ with min-entropy m. We use digital lockers to construct our set difference-based RPI. Our construction, presented in Figure 3, randomizes each set element using a digital locker.

Algorithm RPIGen
Input: $\omega = \{\omega_1, \ldots, \omega_s\}$, $\forall 1 \leqslant i \leqslant s,\ \omega_i \in \mathcal{U}$.
1. For $i = 1 \ldots s$,
   $x_i \xleftarrow{\$} \mathcal{M}_\kappa$.
   $c_i = \mathsf{lock}(\omega_i, x_i)$.
2. Set $\Omega = \{x_1, \ldots, x_s\}$ and $c = c_1, \ldots, c_s$.
3. Return (c, Ω).

Algorithm RPIRep
Inputs: $\omega' = \{\omega'_1, \ldots, \omega'_s\}$, $c = c_1, ..., c_s$.
1. $n = s$,
2. For $i = 1 \ldots s$,
   For $j = 1 \ldots n$,
   a. $x'_i \leftarrow \mathsf{unlock}(\omega'_i, c_j)$.
   b. if $x'_i = \perp \wedge j = n$: $x'_i \xleftarrow{\$} \mathcal{M}_\kappa$.
   c. else if $x'_i = \perp \wedge j \neq n$: continue.
   d. else: remove c_j; n- -; break.
3. Set $\Omega' = \{x'_1, \ldots, x'_{s'}\}$.
4. Return Ω'.

Fig. 3. A set difference-based RPI

It is possible to have a collision between x_is in Algorithm RPIGen, however this occurs with negligible probability (κ must be super-logarithmic for security). This could be addressed by using rejection sampling to ensure all output points are unique.

RPIRep adds additional elements to ensure that the output set is of size s. This step can be triggered if there was a collision in RPIGen or if unlock outputs $\perp$ in Step 2.a. This is the only time where $d(\Omega, \Omega') \geq d(\omega, \omega')$ is when unlock outputs $\perp$ when the two values actually match. Considering the worst case scenario, where ω and ω' are disjoint sets, we end up with s^2 calls to the unlock function. This construction is secure when each element of the input set has superlogarithmic min-entropy.

Theorem 2. *Let λ be a security parameter and let $\kappa = \omega(\log \lambda)$. Let $\mathcal{W}$ be the set of all joint distributions $W_1, W_2, \ldots W_s$ where, for any $i \leqslant s$, $H(W_i) \geq \kappa$. Let $(\mathsf{lock}, \mathsf{unlock})$ be a $(s \cdot \rho)$-composable digital locker with error γ. Then for any $s_{sec} = \mathtt{poly}(\lambda)$ there exists a ϵ_{sec} such that Figure 3 defines a $(\mathcal{M}_{\mathcal{U},s}, \mathcal{M}_\kappa, \mathcal{W}, s \cdot \kappa, \epsilon_{sec}, s_{sec}, \gamma')$-RPI for the set difference metric for*

- *$\epsilon_{sec} = q(q+1)2^{-\kappa} + q2^{-s \cdot \kappa} = \mathtt{ngl}(\lambda)$,*
- *$\gamma' = s^2 \cdot \gamma + (1 - e^{-s^2/|\mathcal{M}_\kappa|})$.*

Our proof is similar to the proof of Canetti *et al.* [16]. We first prove a present proposition that the construction is a PI and then consider reusability. The proof of this proposition is delayed until Appendix A.2.

Proposition 1. *Let λ be a security parameter and let $\kappa = \omega(\log \lambda)$. Let $\mathcal{W}$ be the set of all joint distributions $W_1, W_2, \ldots W_s$ where, for any $i \leqslant s$, $H(W_i) \geq \kappa$. Let $(\mathsf{lock}, \mathsf{unlock})$ be a s-composable digital locker with error γ. Then for any $s_{sec} = \mathtt{poly}(\lambda)$ there exists a $\epsilon_{sec} = \mathtt{ngl}(\lambda)$ such that Figure 3 defines a $(\mathcal{M}_{\mathcal{U},s}, \mathcal{M}_\kappa, \mathcal{W}, s \cdot \kappa, \epsilon_{sec}, s_{sec}, \gamma')$-pseudoentropic isometry for the set difference metric for $\gamma' = s^2 \cdot \gamma + (1 - e^{-s^2/|\mathcal{M}_\kappa|})$.*

Reusability Reusability follows from the security of digital lockers. For each $i \in \{1, ..., \rho\}$, we can treat the outputs

$$\Omega^1, \ldots, \Omega^{i-1}, \Omega^{i+1}, \ldots, \Omega^\rho$$

as auxiliary input to the digital locker adversary. The result follows by simulatability of this adversary, but requires additional composability from the digital locker.

Corollary 1. *Let λ be a security parameter and suppose there exists $(\mathsf{lock}, \mathsf{unlock})$ with that is ℓ composable for any $\ell = \mathtt{poly}(\lambda)$ with error $\delta = \mathtt{ngl}(\lambda)$. Using the RPI defined in Figure 3 for the family $\mathcal{W}$ defined above one can construct a reusable FE for any $s_{sec} = \mathtt{poly}(\lambda), \rho = \mathtt{poly}(\lambda)$ such that $\epsilon_{sec} = \mathtt{ngl}(\lambda)$ and where $t = \Theta(n)$.*

Discussion Our instantiation of an RPI for the set difference metric (large universe) allows construction of the first reusable fuzzy extractor correcting a linear error rate that makes no assumption about how individual readings are correlated. The previous work of Boyen [13] assumed that the exclusive OR of two repeated enrollments leaked no information. Canetti *et al.* [16] achieves a sublinear error rate (and works for Hamming or set difference in the small universe setting).

Looking ahead, we efficiently implement this RPI and there exist fast implementations of fuzzy extractors for the set difference metric (c.f. Section 5).

It is easy to adapt this construction to handle variable sizes. To do so, the RPIGen algorithm needs to pad with random elements up to a maximal set size to hide the actual number of elements in ω. If ω' is not the same size as ω it suffices to loop over the size of ω' in step 2 of RPIRep. Then, one could couple this RPI with any nonreusable fuzzy extractor that can handle sets of variable sizes (see [22, Subsection 6.3]) to add reusability.

4.2 Hamming Distance Instantiation

The work of Canetti *et al.* [16] presents three constructions of fuzzy extractors. They only claim reusability for Construction 1. However, under certain conditions Construction 2 is reusable. Furthermore, it implicitly uses the RPI framework, it first computes a map that preserves distance then applies an error-correcting code.[5]

Adapted construction of Canetti et al. Let $\mathcal{Z}$ be an alphabet and let $W = W_1, ..., W_s$ be a distribution over $\mathcal{Z}^s$. Let $C \subset \{0,1\}^s$ a Hamming error correcting code that corrects t errors. Let $(\mathsf{lockPoint}, \mathsf{unlockPoint})$ be an s-composable secure obfuscated point function with error γ (for keys over $\mathcal{Z}$). The algorithms $\mathsf{Gen}, \mathsf{Rep}$ are defined in Figure 4.

Algorithm RPIGen
Input: $\omega = \omega_1, ..., \omega_s$
1. Sample $c \leftarrow C, r \leftarrow \mathcal{U}_s$.
2. For $j = 1, ..., s$:
   a. If $r_j = 0$: $p_j = \mathsf{lockPoint}(\omega_j)$.
   b. Else:
      $t_j \xleftarrow{\$} \mathcal{Z}$.
      Let $p_j = \mathsf{lockPoint}(t_j)$.
3. Output (c, p), where
$p = p_1 \dots p_s, o = r \oplus c$.

Algorithm RPIRep
Input: (ω', p)
1. For $j = 1, ..., s$:
   a. If $\mathsf{unlockPoint}(\omega'_j, p_j) = 1$:
   set $r'_j = 0$.
   b. Else: set $r'_j = 1$.
2. Set $c = \mathsf{Decode}(o \oplus r')$.
3. Output $c \oplus r$.

Fig. 4. A Hamming distance-based RPI

The construction in Figure 4 does not produce a uniformly random r, it is necessary to apply a randomness extractor (technically, an average-case computational randomness extractor) to r, see [16, Section 5] for more information.

In the work of Canetti *et al.* [16], this construction is not presented as reusable because not all symbols w_i are assumed to have entropy. If each symbol w_i individually has entropy (there is requirement on the correlation between symbols), the construction leaks no information. The only information about r is revealed by $c \oplus r$ because c is chosen from a code. The digital locker is an RPI mapping w to r. In Rep, the computed string $d(r, r') \leq d(w, w') \leq t$ (assuming the locker never has an error). The string c is functioning as a secure sketch (specifically, the code-offset secure sketch). Thus, this construction is actually a combination of a RPI and a (non-reusable) secure sketch.

5 Implementation

Here we describe a basic prototype implementation of our set difference based RPI in Python. The construction can be found in Figure 5. Our construction assumes that SHA512 can be used to construct a digital locker as follows: Let H be SHA512. The locking algorithm $\mathsf{lock}(\mathsf{key}, \mathsf{val})$ outputs the pair $\mathsf{nonce}, H(\mathsf{nonce}, \mathsf{key}) \oplus (0^{256} || \mathsf{val})$, where nonce is a nonce and $||$ denotes concatenation.

We stress our construction is only a construction of an RPI, not a full fuzzy extractor. In practice, we feed our function into the improved Juels-Sudan [36] implementation due to Harmon, Johnson and Reyzin [33].

[5] We change the construction of Canetti *et al.* slightly to illustrate the connection to RPI but our construction is secure under the same conditions.

```
    omega = []
    c = []
    length = hash().digest_size
    for wi in w:
        xi = generate_random(length/2)
        seed = generate_random(16)
        zeros = bytearray([0 for x in range(length/2)])
        lock = bytearray(hmac.new(seed, wi, hash).digest())
        ci = xor(lock, (zeros+xi))
        c.append((ci, seed))
        omega.append(xi)
    return c, omega

def RPIRep(w, c, hash=sha512):
    length = hash().digest_size
    n = s = len(w)
    omega = []
    for i in range(s):
        for j in range(n):
            cj, seed = c[j]
            h = bytearray(hmac.new(seed, w[i], hash).digest())
            xi = xor(cj, h)
            res = check_result(xi)
            if(not res and j==(n-1)):
                omega.append(generate_random(length/2))
            elif(not res and j != (n-1)):
                continue
            else:
                c.pop(j)
                n -= 1
                omega.append(xi[len(xi)/2:])
                break
    return omega


def xor(b1, b2):
    return bytearray([x ^ y for x,y in zip(b1, b2)])

def generate_random(length):
    return bytearray([random.SystemRandom().randint(0, 255) \
        for x in range(length)])

def check_result(res):
    return all(v ==0 for v in res[:len(res)/2])
```

Fig. 5. Python implementation of set difference RPI (implementing Figure 3). We assume that a keyed hash function acts as a digital locker.

6 Use cases

Many companies now prefer *multi-factor authentication* (also called strong authentication), with factors falling in at least two of the following categories: *knowledge* ("what you know"), *possession* ("what you own") and *inherence* ("what you are") [3]. Mobile applications usually use possession and knowledge factors to achieve strong authentication:

- *Possession* The ownership of the mobile device is often proved via a *One Time Password* (OTP) sent by SMS. Once having received the OTP, the user enters it on the authentication web page to prove that she indeed has the device in her possession.
- *Knowledge* A password chosen by the user.

Both of these means are subject to strong limitations. The SMS OTP is vulnerable to numerous attacks (see [46] and references therein) and the SMS channel has been deprecated by authorities such as NIST, which recommend to move to more secure means of authentication [49]. Human memorable passwords do not achieve sufficient entropy, recent estimates place the password entropy at 34 bits [38].

6.1 New Trends for Mobile Authentication

Alternative methods of authentication have emerged such as biometrics and PUFs, which could respectively fulfill inherence and possession proofs. While these solutions have received attention in the authentication literature, they rely on dedicated hardware sensors and components. This is problematic in the case of mobile authentication as the availability of such hardware components (e.g. biometric sensors) varies greatly between devices.

In the same vein, some hardware components (SIM cards [53], dedicated Secure Element [29,44], TEE [30]) are used as possession proofs. Such solutions present several drawbacks and are not widely deployed. While SIM cards and a TEE are widespread, their usage by third party applications remains marginal. As the security component is owned by either the mobile operator (for SIM card-based solutions) or its manufacturer (for SE/TEE-based solutions), a third party application will need to be granted access rights. This leads to additional costs and makes the design of such an authentication solution much more complicated.

As a result, mobile applications are often left using purely software implementations, even critical applications such as banking. To address this limitation, such applications integrate numerous layers of protection including white-box cryptography, tamper-resistance mechanisms and code obfuscation.

In the specific case of HCE-based payment [1], major payment schemes [43,55,7] require use of device fingerprinting [24,39]. In addition, new regulations, such as GDPR (General Data Protection Regulation) in the European Union [26], will soon strictly regulate the usage of personal data and limit their sharing with a server.

Eckersley showed how to create a fingerprint from characteristics of a web browser (user agent, list of fonts, list of plug-ins,...) [24]. Subsequent studies deployed similar systems for personal computers [11,45,47,4]. This research naturally led to studying the practicability of fingerprints on mobile devices.

While early mobile solutions were insufficient [41,18,19,57], recently Kurtz *et al.* provided a comprehensive analysis in the mobile setting [39]. In their work, the list of installed applications and the top 50 songs are among the most identifying values present on a device. These fingerprints

reflect the user's behavior and are candidates to be an inherence authentication factor. Many of these device fingerprints draw on features coming from large universes with variation according to the set difference metric (*e.g.* songs and applications).

The usage of device and behavioral fingerprints, respectively as possession (of the mobile device) and inherence factors can enable strong authentication software only authentication of a user. Device fingerprints can be constructed from the following values:

- *IMEI* (International Mobile Equipment Identity) [56] This value is 15 digits long, the first ones identifying the manufacturer while the 6 last digits are randomly chosen to produce a serial number that identifies the device. The IMEI carries roughly $6 \times \log(10) \approx 20$ bits of entropy.
- *IMSI* (International Mobile Subscriber Identity) [25] The IMSI's first digits are specific to the country and the mobile network, while the remaining 8 digits are randomly chosen. Based on the latter, we can assume that the IMSI carries more than $8 \times \log(10) \approx 26$ bits of entropy.
- *AndroidID* [31] The AndroidID is a 64 bits random number which originally was generated at the device's first boot and remained constant throughout its lifetime, enabling its identification. As of Android 8, it is specific to an application and randomly generated at its installation.

This list is far from being exhaustive, other values could also be used such as the device's Wifi and Bluetooth MAC addresses, the battery's serial number, etc.

Behavioral sources include the following:

- *Apps* The user's applications list [5,39].
- *Music* The user's most listened to songs [5,39].
- *Contacts* The user's favorite contacts. The most common first name in the United States is 'James' [2] with a frequency of 3.318% which yields an approximate min-entropy of $-\log(3.318 \times 10^{-2}) \approx 5$ bits. The first name of the top 20 contacts list should exhibit min-entropy of around 100 bits.
- *TopNetworks* The Wi-Fi networks with the strongest signal at given time and given location. More precisely, the n strongest networks -whose main identifiers are the service set identifiers (SSIDs) and the basic service set identifiers (BSSIDs) define a kind of trusted place (*e.g.* the user's home or workplace). Alternatively we also consider *RegSSID*, the SSIDs of Wifi networks already registered on the device.

Discussion These authentication factors do not require user interaction, contrary to passwords and SMS OTP. This improves user experience which is critical in some use cases (*e.g.* mobile payment applications). However, some of these values may require specific permissions to be available. Other applications may ask for the same permissions, hence, malevolent ones may gain access to some of the above mentioned values. Yet, such applications are steadily and more and more efficiently hunted and removed from the Play Store [32]. Users are also more and more conscious of permissions and privacy issues and grant permissions sparingly [40].

Our use cases both consider a bank authenticating a user on their Android device [6] using software collectible fingerprints to construct possession and inherence authentication factors. We consider a security level of 100 bits to be sufficient.

[6] Similar solutions could also be deployed for iOS devices [39].

6.2 Use Case 1

Our first construction uses a source that combines hardware and software identifiers to prove possession of the phone. Alternative software fingerprints that reflect the user's behavior are used for the inherence factor [39,12,5].

Notation and examples We denote ω_P and ω_I respectively as sources intended to prove possession and inherence. We will consider ω_I as a source that varies according to the set difference metric in the large universe setting. An example of ω_P is described in Figure 6. An example of ω_I is the first name of the user's top 20 contacts.

Data	Estimated Entropy
IMEI	20 bits
IMSI	26 bits
AndroidID	64 bits
$\omega_P = IMEI \| IMSI \| AndroidID$	110 bits

Fig. 6. Device fingerprint's entropy

Both ω_P and ω_I individually carry enough entropy for strong authentication purposes (see respectively Figure 6 and Subsection 6.1). However, while the value ω_P is static, ω_I is a noisy secret that fits set difference metric. At first glance, we could use our reusable FE construction on just ω_I. However, each element of ω_I individually lacks entropy to fulfill the condition of Theorem 2 for reusability.

To solve this problem, we propose to concatenate ω_P to each element ω_I. We denote this augmented source as $\tilde{\omega}_i = \omega_P || \omega_{I,i}$. This augmented source has min-entropy which is the sum of the individual sources. Furthermore, it ties together the two sources in a cryptographic way. Recall that our RPI instantiation (Figure 3) is built from digital lockers which only require min-entropy for security. Let $(\mathsf{Gen}, \mathsf{Rep})$ be the RFE from Theorem 1.Then we describe the instantiation of a fuzzy extractor for ω_P and ω_I in Figure 7.

RPIGen
Input: $\omega_P, \omega_I = \{\omega_{I,1}, \ldots, \omega_{I,s}\}$.
1. For $i = 1 \ldots s$:
   Set $\tilde{\omega}_i = \omega_P \| \omega_{I,i}$.
2. Set $\tilde{\omega} = \{\tilde{\omega}_1, \ldots, \tilde{\omega}_s\}$.
3. $(R, P) \leftarrow \mathsf{Gen}(\tilde{\omega})$.
4. Return (R, P).

RPIRep
Inputs: ω'_P, ω'_I, P
1. For $i = 1 \ldots s$:
   Set $\tilde{\omega}'_i = \omega'_P \| \omega'_{I,i}$.
2. Set $\tilde{\omega}' = \{\tilde{\omega}'_1, \ldots, \tilde{\omega}'_s\}$.
3. $R \leftarrow \mathsf{Rep}(\tilde{\omega}', P)$.
4. Return R.

Fig. 7. A practical use case for set difference based metric

Security ω_P remains the same whereas ω_I can be subject to changes. Therefore, $\omega'_P = \omega_P$ and $\tilde{\omega}'_i = \omega_P \| \omega'_{I,i}$. As a result, $d(\omega_I, \omega'_I) = d(\tilde{\omega}, \tilde{\omega}')$ and the correctness of the underlying RFE $(\mathsf{Gen}, \mathsf{Rep})$ ensures the correctness of use case 1. The designed fingerprint enables application of Theorem 1.

6.3 Use Case 2

Our second use case uses identifying data that leads to distributions under the set difference metric with sufficient entropy for security. This construction can be seen as an extension of use case 1. We use the following sources of data:

- ω_{hard}, a device fingerprint based on the device's hardware elements (*e.g.* IMEI, MAC addresses).
- ω_{soft}, a software fingerprint relying on the device's software components.
- ω_{user}, a user fingerprint based on the user's personal information.
- $\omega_{regSSID}$, a fingerprint based on the SSIDs registered on the device (see subsection 6.1).
- ω_{misc}, a fingerprint that can be based on miscellaneous data.

Some of these sources may not achieve sufficient entropy (e.g. ω_{user} or ω_{soft}). In this case, we augment the sources by using *AndroidID* in the same way as in the last use case.

Potential Fingerprints for ω_{misc} We propose some behavioral fingerprints as candidates for ω_{misc}. The idea would be to extract $\tilde{R}$ from the noisy $\tilde{\omega}$ and use the output value as ω_{misc}. These potential fingerprints include:

- $\tilde{\omega}_{ctcs}$, a top contacts fingerprint based on the user's n_1 favorite contacts.
- $\tilde{\omega}_{songs}$, a top songs fingerprint based on the user's n_2 most listened to songs.
- $\tilde{\omega}_{apps}$, a top applications fingerprint based on the user's n_3 most used applications.

Setting n_1, n_2, n_3 accordingly should enable to fulfill the targeted security level (*e.g.* $n_1 = 20$).

Security Policy Based on these sources, we assume a server that authenticates a user only if a few ones have changed. This policy's threshold corresponds here to the RFE correction capacity t.

We then define the noisy secret

$$\omega = \{\omega_{hard}, \omega_{soft}, \omega_{user}, \omega_{RegSSID}, \omega_{misc1}, \omega_{misc2}\}$$

for which ω_{hard} and ω_{user} should be static while the others are prone to changes. Here, the authentication server could set $t = 2, 3$.

Security As long as the number of errors remains inferior or equal to t, the correctness of the underlying RFE ensures the correctness of the construction. Each element of ω has sufficient entropy for our set difference instantiation to be reusable (see Theorem 1).

Discussion As mentioned in Subsection 6.1, new regulations such as GDPR will enforce restrictions concerning the collect and processing of personal data. Our use cases cope with these restrictions by executing on the mobile device a security policy decided by the server and thus avoiding the latter collecting the data. Moreover, the usage of software accessible fingerprints which do not require user interactions can be seen as a strong asset in mobile payment applications such as HCE-based ones.

7 Conclusion and Future Works

We present a framework for constructing reusable fuzzy extractors by using a randomization step called a reusable pseudoentropic isometry. Since multiple readings of a fuzzy secret may be correlated, the RPI decorrelates them while preserving entropy and distances. We show how to build reusable fuzzy extractors out of any efficient nonreusable fuzzy extractors and RPIs.

Relying on this new framework, we use digital lockers to construct an RPI and design the first reusable fuzzy extractor for the set difference metric. Our construction is also the first reusable fuzzy extractor handling a linear error rate that makes no assumption about how repeated readings are correlated. We also show that the framework can be applied to the Hamming distance through the example of Canetti *et al.*'s second construction. We then propose a Python implementation of our set difference metric RPI construction.

In the two last sections, we described a prototype implementation of our set difference instantiation and two use cases for our set difference instantiation. These use cases show the applicability of our RPI construction in the context of industrial mobile authentication.

Future works The major open question is designing new RPI instantiations, especially those that weaken the cryptographic assumption or the required structure on the noisy secret. Another complementary path could be to focus on constructing a fuzzy extractor *directly* reusable. From a more practical point of view, an extensive study of mobile devices' fingerprints would allow accurate assessment of their entropy and their variation over time. These two points will improve our RPI framework.

References

1. Host-based card emulation. `https://developer.android.com/guide/topics/connectivity/nfc/hce.html`.
2. Most common first names and last names in the u.s. `https://names.mongabay.com/male_names.htm`.
3. Multi-factor authentication. https://www.pcisecuritystandards.org/pdfs/Multi-Factor-Authentication-Guidance-v1.pdf.
4. G. Acar, C. Eubank, S. Englehardt, M. Juarez, A. Narayanan, and C. Diaz. The web never forgets: Persistent tracking mechanisms in the wild. In *Proceedings of the 2014 ACM SIGSAC Conference on Computer and Communications Security*, CCS '14, pages 674–689, New York, NY, USA, 2014. ACM.
5. J. P. Achara, G. Acs, and C. Castelluccia. On the unicity of smartphone applications. In *Proceedings of the 14th ACM Workshop on Privacy in the Electronic Society*, WPES '15, pages 27–36. ACM, 2015.
6. D. Apon, C. Cho, K. Eldefrawy, and J. Katz. Efficient, reusable fuzzy extractors from lwe. Cryptology ePrint Archive, Report 2017/755, 2017. `http://eprint.iacr.org/2017/755`.
7. Bancontact. SEPA Rulebooks Scheme Manuals Remote Domain 46D0 Mobile App Security Guidelines, 2016.
8. C. H. Bennett, G. Brassard, and J.-M. Robert. Privacy amplification by public discussion. *SIAM Journal on Computing*, 17(2):210–229, 1988.
9. N. Bitansky and R. Canetti. On strong simulation and composable point obfuscation. In *Advances in Cryptology–CRYPTO 2010*, pages 520–537. Springer, 2010.
10. M. Blanton and M. Aliasgari. Analysis of reusability of secure sketches and fuzzy extractors. *IEEE Trans. Information Forensics and Security*, 8(9):1433–1445, 2013.
11. K. Boda, A. M. Földes, G. G. Gulyás, and S. Imre. User tracking on the web via cross-browser fingerprinting. In *Proceedings of the 16th Nordic Conference on Information Security Technology for Applications*, NordSec'11, pages 31–46, Berlin, Heidelberg, 2012. Springer-Verlag.
12. H. Bojinov, Y. Michalevsky, G. Nakibly, and D. Boneh. Mobile device identification via sensor fingerprinting. *CoRR*, abs/1408.1416, 2014.
13. X. Boyen. Reusable cryptographic fuzzy extractors. In *Proceedings of the 11th ACM Conference on Computer and Communications Security*, CCS '04, pages 82–91. ACM, 2004.

14. R. Canetti. Towards realizing random oracles: Hash functions that hide all partial information. In *Advances in Cryptology - CRYPTO '97, 17th Annual International Cryptology Conference, Santa Barbara, California, USA, August 17-21, 1997, Proceedings*, pages 455–469, 1997.
15. R. Canetti and R. R. Dakdouk. Obfuscating point functions with multibit output. In *Advances in Cryptology–EUROCRYPT 2008*, pages 489–508. Springer, 2008.
16. R. Canetti, B. Fuller, O. Paneth, L. Reyzin, and A. Smith. *Advances in Cryptology – EUROCRYPT 2016*, chapter Reusable Fuzzy Extractors for Low-Entropy Distributions, pages 117–146. Springer Berlin Heidelberg, Berlin, Heidelberg, 2016.
17. R. Canetti, Y. Tauman Kalai, M. Varia, and D. Wichs. *On Symmetric Encryption and Point Obfuscation*, pages 52–71. Springer Berlin Heidelberg, Berlin, Heidelberg, 2010.
18. M. Chen, J. Fridrich, M. Goljan, and J. Lukás. Determining image origin and integrity using sensor noise. *IEEE Transactions on Information Forensics and Security*, 3(1):74–90, 2008.
19. A. Das, N. Borisov, and M. Caesar. Do you hear what i hear?: Fingerprinting smart devices through embedded acoustic components. In *Proceedings of the 2014 ACM SIGSAC Conference on Computer and Communications Security*, CCS '14, pages 441–452, 2014.
20. J. Daugman. How iris recognition works. *IEEE Transactions on Circuits and Systems for Video Technology*, 14:21–30, 2002.
21. 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*, pages 412–431. Springer, 2016.
22. 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, Mar. 2008.
23. 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 2004*, volume 3027 of *Lecture Notes in Computer Science*, pages 523–540. Springer Berlin Heidelberg, 2004.
24. P. Eckersley. How unique is your web browser? In *Privacy Enhancing Technologies, 10th International Symposium, PETS 2010, Berlin, Germany, July 21-23, 2010. Proceedings*, pages 1–18, 2010.
25. ETSI. Digital cellular telecommunications system (phase 2+). *Security related network functions*, 1992.
26. European Parliament. General Data Protection Regulation (GDPR). `http://ec.europa.eu/justice/data-protection/reform/files/regulation_oj_en.pdf`, 2016.
27. B. Fuller, X. Meng, and L. Reyzin. *Computational Fuzzy Extractors*, pages 174–193. Springer Berlin Heidelberg, Berlin, Heidelberg, 2013.
28. J. Galbally, A. Ross, M. Gomez-Barrero, J. Fierrez, and J. Ortega-Garcia. Iris image reconstruction from binary templates: An efficient probabilistic approach based on genetic algorithms. *Computer Vision and Image Understanding*, 117(10):1512–1525, 2013.
29. Gemalto. Gemalto eSE secure end-to-end solutions. `https://www.gemalto.com/iot/consumer-electronics/embedded-secure-element`.
30. GlobalPlatform. GlobalPlatform made simple guide: Trusted Execution Environment (TEE) Guide. `https://www.globalplatform.org/mediaguidetee.asp`.
31. Google. Android Developer Reference, Settings.Secure, ANDROID_ID. `https://developer.android.com/reference/android/provider/Settings.Secure.html#ANDROID_ID`.
32. Google. Android Security 2016 Year in Review. `https://source.android.com/security/reports/Google_Android_Security_2016_Report_Final.pdf`, 2017.
33. K. Harmon, S. Johnson, and L. Reyzin. An implementation of syndrome encoding and decoding for binary bch codes, secure sketches and fuzzy extractors, 2006. Available at `http://www.cs.bu.edu/~reyzin/code/fuzzy.html`.
34. J. Håstad, R. Impagliazzo, L. A. Levin, and M. Luby. A pseudorandom generator from any one-way function. *SIAM Journal on Computing*, 28(4):1364–1396, 1999.
35. C.-Y. Hsiao, C.-J. Lu, and L. Reyzin. Conditional computational entropy, or toward separating pseudoentropy from compressibility. In *Annual International Conference on the Theory and Applications of Cryptographic Techniques*, pages 169–186. Springer, 2007.
36. A. Juels and M. Sudan. A fuzzy vault scheme. *Des. Codes Cryptography*, 38(2):237–257, Feb. 2006.
37. A. Juels and M. Wattenberg. A fuzzy commitment scheme. In *Proceedings of the 6th ACM Conference on Computer and Communications Security*, CCS '99, pages 28–36. ACM, 1999.
38. S. Komanduri, R. Shay, P. G. Kelley, M. L. Mazurek, L. Baueř, 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.

39. A. Kurtz, H. Gascon, T. Becker, K. Rieck, and F. Freiling. Fingerprinting mobile devices using personalized configurations. *Proceedings on Privacy Enhancing Technologies*, 2016(1):4–19, 2016.
40. J. Lin, B. Liu, N. M. Sadeh, and J. I. Hong. Modeling users' mobile app privacy preferences: Restoring usability in a sea of permission settings. In *Tenth Symposium on Usable Privacy and Security, SOUPS 2014, Menlo Park, CA, USA, July 9-11, 2014*, pages 199–212, 2014.
41. J. Lukas, J. Fridrich, and M. Goljan. Digital camera identification from sensor pattern noise. *IEEE Transactions on Information Forensics and Security*, 1(2):205–214, June 2006.
42. B. Lynn, M. Prabhakaran, and A. Sahai. Positive results and techniques for obfuscation. In *Advances in Cryptology–EUROCRYPT 2004*, pages 20–39. Springer, 2004.
43. MasterCard. MasterCard CloudBased Payments Security Guidelines for MPA Development Version 1.1, 2015.
44. Morpho. Secure Elements. `https://www.morpho.com/en/commercial-identity/solutions-telecom/secure-elements`.
45. K. Mowery and H. Shacham. Pixel perfect: Fingerprinting canvas in HTML5. In M. Fredrikson, editor, *Proceedings of W2SP 2012*. IEEE Computer Society, May 2012.
46. C. Mulliner, R. Borgaonkar, P. Stewin, and J. Seifert. Sms-based one-time passwords: Attacks and defense - (short paper). In *Detection of Intrusions and Malware, and Vulnerability Assessment - 10th International Conference, DIMVA 2013, Berlin, Germany, July 18-19, 2013. Proceedings*, pages 150–159, 2013.
47. N. Nikiforakis, A. Kapravelos, W. Joosen, C. Kruegel, F. Piessens, and G. Vigna. Cookieless monster: Exploring the ecosystem of web-based device fingerprinting. In *Proceedings of the 2013 IEEE Symposium on Security and Privacy*, SP '13, pages 541–555, Washington, DC, USA, 2013. IEEE Computer Society.
48. N. Nisan and D. Zuckerman. Randomness is linear in space. *Journal of Computer and System Sciences*, 52(1):43–52, 1996.
49. NIST. Digital identity guidelines: Authentication and lifecycle management. `http://nvlpubs.nist.gov/nistpubs/SpecialPublications/NIST.SP.800-63b.pdf`.
50. R. Pappu, B. Recht, J. Taylor, and N. Gershenfeld. Physical one-way functions. *Science*, 297(5589):2026–2030, 2002.
51. S. Prabhakar, S. Pankanti, and A. K. Jain. Biometric recognition: Security and privacy concerns. *IEEE Security & Privacy*, 1(2):33–42, 2003.
52. K. Simoens, P. Tuyls, and B. Preneel. Privacy weaknesses in biometric sketches. In *2009 30th IEEE Symposium on Security and Privacy*, pages 188–203.
53. E. Telecommunications Standards Institute. SIM. `http://www.etsi.org/technologies-clusters/technologies/smart-cards/sim`.
54. U. Uludag, S. Pankanti, S. Prabhakar, and A. K. Jain. Biometric cryptosystems: issues and challenges. *Proceedings of the IEEE*, 92(6):948–960, June 2004.
55. VISA. Security Requirements and Evaluation Guidance for Mobile Applications. Visa Digital Solutions. Version 1.0, 2014.
56. Wikipedia. International Mobile Equipment Identity (IMEI). `https://en.wikipedia.org/wiki/International_Mobile_Equipment_Identity`.
57. Z. Zhou, W. Diao, X. Liu, and K. Zhang. Acoustic fingerprinting revisited: Generate stable device id stealthily with inaudible sound. In *Proceedings of the 2014 ACM SIGSAC Conference on Computer and Communications Security*, CCS '14, pages 429–440. ACM, 2014.

A Proofs

A.1 Proof of Lemma 1

Proof (Proof of Lemma 1). Suppose not, that is suppose that there exists some D of size at most s such that $\delta^s((R, P, V), (U_l, P, V)) > \epsilon$. Let X_v be a collection of distributions (defined for each $v \in V$) giving rise to a joint distribution such that $\tilde{H}_\infty(X|V) \geq m_2$. Consider a D_1 that does the following:

1. Receive input α, β.
2. Run $\gamma, \nu \leftarrow \mathsf{Gen}'(\alpha)$.
3. Output $D(\gamma, \nu, \beta)$.

Also consider a D_2 that does the following:

1. Receive input α, β.
2. Run $\gamma, \nu \leftarrow \mathsf{Gen}'(\alpha)$.
3. Sample random string $u \leftarrow U_\ell$.
4. Output $D(u, \nu, \beta)$.

Denote $R', P' \leftarrow \mathsf{Gen}'(X)$. By the triangle inequality we have the following:

$$\begin{aligned}
&\delta^{D_1}((U,V),(X,V)) + \delta^{D_2}((U,V),(X,V)) \\
&\quad = \delta^{D}((R,P,V),(R',P',V)) + \delta^{D}((U_\ell,P,V),(U_\ell,P',V)) \\
&\quad \geq \delta^{D}((R,P,V),(U_\ell,P,V)) - \delta^{D}((U_\ell,P',V),(R',P',V)) \\
&\quad \geq \epsilon - \epsilon_{FE} = 2\epsilon_{RPI}
\end{aligned}$$

Thus, either D_1 or D_2 distinguishes U, V from X, V with advantage at least ϵ_{RPI}. Either of these distinguishers contradict the HILL entropy of U, V. This completes the proof of Lemma 1.

A.2 Proof of Proposition 1

Proof (Proof of Proposition 1). We have to prove both isometric and security properties.

Isometry property. Ω is of size s. By a birthday bound calculation, the probability of any collision in the x_is is $1 - e^{-s^2/|\mathcal{M}_\kappa|}$. For any $\omega_i = \omega'_i$, if no digital locker outputs $\perp$ then $x_i = x'_i$ and the total number of calls to function unlock is s^2. Thus,

$$\Pr[d(\Omega, \Omega') = d(\omega, \omega')] \geq 1 - s^2\gamma - (1 - e^{-s^2/|\mathcal{M}_\kappa|}).$$

Security. Our goal is to show that for all $s_{sec} = \mathtt{poly}(\lambda)$ there exists $\epsilon_{sec} = \mathtt{ngl}(\lambda)$ such that $\delta^{\mathcal{D}_{s_{sec}}}((R,P),(U,P)) \leq \epsilon_{sec}$. Fix some polynomial s_{sec} and let D be a distinguisher of size at most s_{sec}. We want to bound

$$|\,\mathbb{E}[D(\Omega, P)] - \mathbb{E}[D(U_{\mathcal{M}_\kappa}, P)]|$$

by a negligible function.

We proceed by contradiction: suppose this difference is not negligible. That is, suppose that there is some polynomial $p(\cdot)$ such that for all λ_0 there exists some $\lambda > \lambda_0$ such that

$$|\,\mathbb{E}[D(\Omega, P)] - \mathbb{E}[D(U_{\mathcal{M}_\kappa}, P)]| > 1/p(\lambda).$$

Note that λ is a function of λ_0 but we omit this notation for the remainder of the proof. By the security of digital lockers (Definition 5), there is a polynomial q and an unbounded time simulator S (making at most $q(\lambda)$ queries to the oracles $\{\mathsf{idealUnlock}(\omega_i, x_i)\}_{i=1}^s$) such that

$$\left|\mathbb{E}[D(\Omega, C_1, ..., C_s)] - \mathbb{E}\left[S^{\{\mathsf{idealUnlock}(\omega_i,x_i)\}_{i=1}^s}(\Omega, \kappa)\right]\right| \leq \frac{1}{3p(\lambda)}. \tag{1}$$

The same is true if we replaced Ω above by an independent uniform random variable U over $\mathcal{M}_\kappa$. We now prove the following lemma, which shows that S cannot distinguish between Ω and $U_{\mathcal{M}_\kappa}$.

Lemma 2. *Let U denote the uniform distribution over $\mathcal{M}_\kappa$. Then*

$$\Big| \mathbb{E}\left[S^{\{\mathsf{idealUnlock}(\omega_i,x_i)\}_{i=1}^s}(R,\kappa)\right] \tag{2}$$

$$- \mathbb{E}\left[S^{\{\mathsf{idealUnlock}(\omega_i,x_i)\}_{i=1}^s}(U_{\mathcal{M}_\kappa},\kappa)\right] \Big| \tag{3}$$

$$\leq \frac{q(q+1)}{2^m} \leq \frac{1}{3p(\lambda)}\,, \tag{4}$$

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

Proof. Fix any $u \in \mathcal{M}_\kappa$ (the lemma will follow by averaging over all u). Let Ω^* be the correct value of Ω. The only information that S can learn about whether the value is Ω^* or u is through the query responses. First, modify S slightly to quit immediately if it gets a response not equal to $\perp$ (we assume such as soon as S gets back a non-$\perp$ response it distinguishes with probability 1). 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 [22, Lemma 2.2b], $\tilde{\mathrm{H}}_\infty(V_i|View(S),\{j_{ik}\}) \geq \tilde{\mathrm{H}}_\infty(V_j|\{j_{ik}\}) - \log(q+1) \geq m - \log(q+1)$. Therefore, at each query, the probability that S gets a non-$\perp$ answer is at most $(q+1)2^{-m}$. Since there are q queries of S, the overall probability is at most $q(q+1)/2^m$. Then since 2^m is $\mathtt{ngl}(\lambda)$, there exists some λ such that for all $\lambda > \lambda_0$, $q(q+1)/2^m \leq 1/(3p(\lambda))$. This completes the proof of Lemma 2.

Adding together Equation 1, Equation 4, and Equation 1 in which Ω is replaced with $U_{\mathcal{M}_\kappa}$, we obtain that

$$\delta^D((\Omega,P),(U_{\mathcal{M}_\kappa},P)) \leq \frac{1}{p(\lambda)}.$$

This is a contradiction and completes the proof of Proposition 1.