# On the Possibilities and Limitations of Computational Fuzzy Extractors

Kenji Yasunaga* Kosuke Yuzawa†

November 6, 2017

**Abstract**

We present positive and negative results of fuzzy extractors with computational security. As a negative result, we show that, under a certain computational condition, the existence of a computational fuzzy extractor implies the existence of an information-theoretic fuzzy extractor with slightly weaker parameters. The condition is that the generation procedure of the fuzzy extractor is efficiently invertible by an injective function. Our result implies that to circumvent the limitations of information-theoretic fuzzy extractors, we need to employ computational fuzzy extractors that are not invertible by injective functions. As positive results, we present constructions of computational fuzzy extractors based on a leakage-resilient key encapsulation mechanism and a strong decisional Diffie-Hellman assumption.

**Keywords:** Fuzzy Extractor, Error-Correcting Code, Key Encapsulation Mechanism, Leakage-Resilient Cryptosystem

## 1 Introduction

Cryptographic primitives generally require uniformly random strings. A *fuzzy extractor* is a primitive proposed by Dodis et al. [6] that can reliably derive uniformly random keys from noisy sources, such as biometric data (fingerprint, iris, facial image, etc.) and long pass-phrases. More formally, a fuzzy extractor is defined to be a pair of procedures $(\mathsf{Gen}, \mathsf{Rep})$. The key generation procedure $\mathsf{Gen}$ receives a sample $w$ from a noisy source $W$ with some entropy, and outputs a uniformly random key $r$ and a helper string $p$. After that, the reproduction procedure $\mathsf{Rep}$ can be used to derive the same key $r$ from the helper string $p$ and a sample $w'$ that is close to the original sample $w$. Notably, this framework does not need secret keys other than $w$. The derived key $r$ is close to uniform even if the helper string $p$ was given. See [7, 3] for surveys of results related to fuzzy extractors.

To construct fuzzy extractors, Dodis et al. [6] introduced a primitive, called *secure sketch*. On input $w$, a secure sketch produces a recovery information. It enables the recovery of $w$ from any close value $w'$, but does not reveal much information about $w$. They show that a combination of a secure sketch and a strong extractor gives a fuzzy extractor.

Fuzzy extractors were defined as *information-theoretic* primitives, and several limitations regarding parameters in fuzzy extractors are also studied in [6]. The *entropy loss* is the difference

*Faculty of Electrical and Computer Engineering, Kanazawa University. yasunaga@se.kanazawa-u.ac.jp
†Division of Electrical Engineering and Computer Science, Kanazawa University.

between the entropy of $w$ and the length of the extracted key $r$. In the setting of information-theoretic security, the entropy loss is known to be inevitable [12]. This limitation is a major problem for applications using low entropy sources such as biometric data.

Fuller et al. [8] considered the *computational security* of fuzzy extractors to construct *lossless* fuzzy extractors, which circumvent the entropy loss of information-theoretic fuzzy extractors. They gave both negative and positive results. On one hand, they show that the existence of a computational secure sketch implies the existence of an information-theoretic secure sketch with slightly weaker parameters. This result means that lossless fuzzy extractors cannot be constructed by combining a computational secure sketch and a strong extractor. On the other hand, they present a direct construction of a lossless fuzzy extractor based on the hardness of learning with errors (LWE) problem.

In this work, we further study the possibilities and limitations of computational fuzzy extractors. The negative result of [8] implies that we need to avoid using computational secure sketches to construct lossless fuzzy extractors. However, it remains unclear what properties are necessary for fuzzy extractors to be lossless. Regarding the possibilities, a construction of [8] is only given based on a specific computational assumption. It was not known that lossless fuzzy extractors can be constructed based on other computational assumptions such as the decisional Diffie-Hellman (DDH) assumption or more general assumptions such as the existence of a one-way function.

First, we observe that the negative result of [8] can be applied to computational fuzzy extractors under some condition. The condition is that for the generation procedure $\mathsf{Gen}$, there is an efficient inverter that, on input $(r, p)$, recovers the same $w$ that was actually used to generate $(r, p)$ by $\mathsf{Gen}$. It is unclear if the negative result holds for an inverter without this property. We will discuss this observation in more detail in Section 1.1.

Regarding the limitations, we provide a similar negative result of computational fuzzy extractors under another condition. Specifically, we show that if $\mathsf{Gen}$ has an efficient inverter that is *almost injective*, then the existence of a computational fuzzy extractor implies the existence of an information-theoretic fuzzy extractor. This result indicates that a lossless fuzzy extractor must have a property that the generation procedure is not efficiently invertible by injective functions. In the process of proving the result, we fix a flaw in the proof of the negative results of [8], and obtain a similar lemma with a slightly weaker parameter.

Next, as a positive result, we show that lossless fuzzy extractors can be constructed based on various computational assumptions. Specifically, we give a construction of a lossless fuzzy extractor from a *leakage-resilient key encapsulation mechanism*. A key encapsulation mechanism (KEM) is a public-key primitive that enables two parties to share a random key without private communication. KEMs have practical and provably secure constructions under various computational assumptions [5]. A leakage-resilient KEM is a KEM with the security against leakage-attacks to secret keys. Such leakage-resilient cryptographic primitives have been extensively studied in recent years. See [2, 11] for surveys of leakage-resilient primitives. In our positive result, we only need a somewhat weak leakage-resilience, where the leakage function is determined before generating a public key. A generic construction of secure public-key encryption in this model is provided by Naor and Segev [10]. The construction employs only a standard KEM and a strong extractor. We observe that a combination of a leakage-resilient KEM and a secure sketch gives a computational fuzzy extractor. Compared to existing computational extractors, our construction based on leakage-resilient KEM has an advantage in "stretching" the key. We discuss these points in Section 4. Finally, we give a simple construction of a computational fuzzy extractor based on a

stronger variant of the decisional Diffie-Hellman assumption.

### 1.1 On the Negative Results of [8]

Fuller et al. noted in [8, footnote 3] that, if the generation procedure $\mathsf{Gen}$ is efficiently invertible, their negative results for computational secure sketches can also be applied to computational fuzzy extractors. We observe that this is true if the inverter of $\mathsf{Gen}$ satisfies some condition, but it is unclear without it. We describe the observation below in more detail.

Let $(\mathsf{Gen}, \mathsf{Rep})$ be a computational fuzzy extractor. Assume that there is an efficient algorithm $\mathsf{InvGen}$ that, given $(r, p)$, outputs $w$, where $(r, p)$ was generated by $\mathsf{Gen}(w)$. One can construct a computational secure sketch $(\mathsf{SS}, \mathsf{Rec})$ (see Definition 3 for the definition of secure sketches) by defining $\mathsf{SS}(w) = \{(r, p) \leftarrow \mathsf{Gen}(w); \text{Output } p\}$ and $\mathsf{Rec}(w', p) = \{r \leftarrow \mathsf{Rep}(w', p); w \leftarrow \mathsf{InvGen}(r, p); \text{Output } w\}$. Thus, by the negative results of [8], this implies the existence of an information-theoretic fuzzy extractor. However, the above observation can be applied only if $\mathsf{InvGen}(r, p)$ outputs the same $w$ from which $(r, p)$ was actually generated. In general, there could exist different $w_1$ and $w_2$ such that the outputs of $\mathsf{Gen}(w_1)$ and $\mathsf{Gen}(w_2)$ are the same. In such a case, one of $w_1$ and $w_2$ may not be recovered by $\mathsf{InvGen}$, and thus it may be difficult to use $\mathsf{InvGen}$ for constructing secure sketches.

If we assume that $\mathsf{Gen}$ is injective, then there are no different $w_1$ and $w_2$ satisfying $\mathsf{Gen}(w_1) = \mathsf{Gen}(w_2)$, and thus the negative results of [8] can be applied to such computational fuzzy extractors. However, the assumption seems too restrictive. As far as we known, there is no construction of injective fuzzy extractors. Also, there is an intuitive reason for this fact. For a fuzzy extractor $(\mathsf{Gen}, \mathsf{Rep})$, consider two input $w_1$ and $w_2$ that are close to each other. If $\mathsf{Gen}(w_1)$ outputs $(r, p)$, then it must be that $\mathsf{Rep}(w_1, p) = r$ and $\mathsf{Rep}(w_2, p) = r$. Then, it seems natural that the output range of $\mathsf{Gen}(w_2)$ also contains $(r, p)$. If so, the extractor is not injective.

## 2 Preliminaries

Let $X$ and $Y$ be random variables over some alphabet $Z$. The *min-entropy* of $X$ is $\mathrm{H}_\infty(X) = -\log(\max_x \Pr[X = x])$. The *average min-entropy* of $X$ given $Y$ is $\tilde{\mathrm{H}}_\infty(X|Y) = -\log(\mathbb{E}_{y \in Z} \max_{x \in Z} \Pr[X = x|Y = y])$. The *statistical distance* between $X$ and $Y$ is $\Delta(X, Y) = \frac{1}{2} \sum_{z \in Z} |\Pr[X = z] - \Pr[Y = z]|$. If $\Delta(X, Y) \leq \epsilon$, we say $X$ and $Y$ are *$\epsilon$-close*. The support of $X$ is $\mathrm{Supp}(X) = \{x \in Z : \Pr[X = x] > 0\}$. We denote by $U_\ell$ the uniformly distributed random variable on $\{0, 1\}^\ell$. For a finite set $S$, we denote by $t \leftarrow S$ the event that $t$ is chosen uniformly at random from $S$. For $s \in \mathbb{N}$, the *computational distance* between $X$ and $Y$ is $\Delta^s(X, Y) = \max_{D \in \mathcal{C}_s} |\mathbb{E}[D(X)] - \mathbb{E}[D(Y)]|$, where $C_s$ is the set of randomized circuits of size at most $s$ that output 0 or 1. A metric space is a set $\mathcal{M}$ with a distance function $\mathsf{dis} : \mathcal{M} \times \mathcal{M} \to \mathbb{R}^+ = [0, \infty)$. We always consider finite metric spaces and distance functions with finite images. For the Hamming metric over $Z^n$, $\mathsf{dis}(x, y)$ is the number of positions in which $x$ and $y$ differ. For a probabilistic experiment $E$ and a predicate $P$, we denote by $\Pr[E : P]$ the probability that the predicate $P$ is true after the event $E$ occurred. For a probabilistic algorithm $A$, we denote by $A(x; r)$ the output of $A$, given $x$ as input and $r$ as random coins.

We give definitions of fuzzy extractor, computational fuzzy extractor, secure sketch, and strong extractor.

**Definition 1** (Fuzzy Extractor)**.** *An* $(\mathcal{M}, m, \ell, t, \epsilon)$-fuzzy extractor *with error* $\delta$ *is a pair of randomized procedures* $(\mathsf{Gen}, \mathsf{Rep})$ *with the following properties:*

- *The generation procedure* $\mathsf{Gen}$ *on input* $w \in \mathcal{M}$ *outputs an extracted string* $r \in \{0,1\}^\ell$ *and a helper string* $p \in \{0,1\}^*$.
- *The reproduction procedure* $\mathsf{Rep}$ *takes* $w' \in \mathcal{M}$ *and* $p \in \{0,1\}^*$ *as inputs. The* correctness *property guarantees that for any* $w, w' \in \mathcal{M}$ *with* $\mathsf{dis}(w, w') \leq t$*, if* $(r, p) \leftarrow \mathsf{Gen}(w)$*, then* $\mathsf{Rep}(w', p) = r$ *with probability at least* $1 - \delta$*, where the probability is taken over the coins of* $\mathsf{Gen}$ *and* $\mathsf{Rep}$*. If* $\mathsf{dis}(w, w') > t$*, no guarantee is provided about the output of* $\mathsf{Rep}$.
- *The* security *property guarantees that for any distribution* $W$ *on* $\mathcal{M}$ *of min-entropy* $m$*, if* $(R, P) \leftarrow \mathsf{Gen}(W)$*, then* $\Delta((R, P), (U_\ell, P)) \leq \epsilon$.

**Definition 2** (Computational Fuzzy Extractor)**.** *An* $(\mathcal{M}, m, \ell, t, s, \epsilon)$-computational fuzzy extractor *with error* $\delta$ *is a pair of randomized procedures* $(\mathsf{Gen}, \mathsf{Rep})$ *that is an* $(\mathcal{M}, m, \ell, t, \epsilon)$*-fuzzy extractor with error* $\delta$ *in which the security property is replaced by the following one:*

- *For any distribution* $W$ *on* $\mathcal{M}$ *of min-entropy* $m$*, if* $(R, P) \leftarrow \mathsf{Gen}(W)$*, then* $\Delta^s((R, P), (U_\ell, P)) \leq \epsilon$.

**Definition 3** (Secure Sketch)**.** *An* $(\mathcal{M}, m, \tilde{m}, t)$-secure sketch *with error* $\delta$ *is a pair of randomized procedures* $(\mathsf{SS}, \mathsf{Rec})$ *with the following properties:*

- *The sketching procedure* $\mathsf{SS}$ *on input* $w \in \mathcal{M}$ *outputs a string* $s \in \{0,1\}^*$.
- *The recovery procedure* $\mathsf{Rec}$ *takes* $w' \in \mathcal{M}$ *and* $s \in \{0,1\}^*$ *as inputs. The* correctness *property guarantees that for any* $w, w' \in \mathcal{M}$ *with* $\mathsf{dis}(w, w') \leq t$*,* $\Pr[\mathsf{Rec}(w', \mathsf{SS}(w)) = w] \geq 1 - \delta$ *where the probability is taken over the coins of* $\mathsf{SS}$ *and* $\mathsf{Rec}$*. If* $\mathsf{dis}(w, w') > t$*, no guarantee is provided about the output of* $\mathsf{Rec}$.
- *The* security *property guarantees that for any distribution* $W$ *on* $\mathcal{M}$ *of min-entropy* $m$*,* $\tilde{\mathrm{H}}_\infty(W|\mathsf{SS}(W)) \geq \tilde{m}$.

**Definition 4.** *We say that* $\mathsf{Ext} : \{0,1\}^n \to \{0,1\}^\ell$ *is an* $(n, m, \ell, \epsilon)$-strong extractor *if for any* $W$ *on* $\{0,1\}^n$ *of min-entropy* $m$*,* $\Delta((\mathsf{Ext}(W; X), X), (U_\ell, X)) \leq \epsilon$*, where* $X$ *is the uniform distribution on* $\{0,1\}^r$.

## 3 Negative Results

In this section, we show that the existence of a computational fuzzy extractor satisfying some computational condition implies the existence of an information-theoretic fuzzy extractor with slightly weaker parameters. The condition is that the generation procedure of a fuzzy extractor is efficiently invertible by an almost-injective function.

We follow a similar approach to Fuller et al. [8], who showed that a computational secure sketch implies an information-theoretic secure sketch. They proved that the existence of a computational secure sketch implies the existence of a code correcting random errors. The result follows by observing that such a code is sufficient to construct an information-theoretic secure sketch [6].

We start from the existence of a computational fuzzy extractor $(\mathsf{Gen}, \mathsf{Rep})$. To show the existence of an error-correcting code, we assume that the generation procedure $\mathsf{Gen}$ of the fuzzy extractor is efficiently invertible. The idea for constructing a code is that the inverter of $\mathsf{Gen}$ can work as a generator of a codeword from a message. Here, a sample $w$ and an extracted string $r$ from $w$ are considered a codeword and a message, respectively. By fixing the helper string $p$, we can see that the inverter of $\mathsf{Gen}$ is an encoder and the reproduction procedure $\mathsf{Rep}$ is a decoder of an error-correcting code. The injectiveness of the inverter of $\mathsf{Gen}$ is used to guarantee a high information-rate of the resulting code. The structure used in our approach is different from that in [8]. For a secure sketch $(\mathsf{SS}, \mathsf{Rec})$, they used the fact that by fixing the sketch $ss = \mathsf{SS}(W)$, the procedure of sampling $W$ conditioned on $ss$ is a random sampling of codewords and the recovery procedure $\mathsf{Rec}$ can work as a decoder that outputs a corrected codeword, not message.

We give a formal definition of invertibility of the generation procedure.

**Definition 5.** *Let* $(\mathsf{Gen}, \mathsf{Rep})$ *be a fuzzy extractor for a metric space* $\mathcal{M}$*. We say* $\mathsf{Gen}$ *is* $(s, \eta)$-invertible *if there exists a deterministic circuit* $\mathsf{InvGen}$ *of size at most* $s$ *such that*

$$\Pr\left[W' \leftarrow \mathsf{InvGen}(R', p) : \begin{array}{c} \exists\, r_G \in \{0,1\}^* \ s.t. \\ \mathsf{Gen}(W'; r_G) = (R', p) \end{array}\right] \geq 1 - \eta$$

*for any* $p$ *that can be generated as* $(r, p) \leftarrow \mathsf{Gen}(w)$ *for* $w \in \mathcal{M}$*, where* $R' = U_\ell$*. In addition, if the inverter* $\mathsf{InvGen}$ *has the property such that* $|\{w' : w' \leftarrow \mathsf{InvGen}(U_\ell, p)\}| \geq (1 - \xi)2^\ell$ *for any* $p$ *that can be generated as* $(r, p) \leftarrow \mathsf{Gen}(w)$*, we say* $\mathsf{Gen}$ *is* $(s, \eta, \xi)$-almost-injectively-invertible.

In the definition, we consider that $\mathsf{InvGen}$ succeeds in inverting $\mathsf{Gen}$ if it outputs $w'$ from which the input $(r', p)$ can be generated by $\mathsf{Gen}$, and thus $w'$ is not necessarily the same as $w$ from which $p$ was actually generated.

Note that, since the inverter $\mathsf{InvGen}$ is confined to being deterministic, $\mathsf{InvGen}$ has the property of *output uniqueness*. That is, for any $r$ and $p$, $\mathsf{InvGen}(r, p)$ does not output two different values $w_1, w_2 \in \mathcal{M}$ such that $(r, p) = \mathsf{Gen}(w_1; r_1) = \mathsf{Gen}(w_2; r_2)$ for some $r_1, r_2 \in \{0,1\}^*$.

We will prove that the existence of a computational fuzzy extractor implies the existence of an error-correcting code. We provide some notions regarding coding theory.

**Definition 6.** *We say a metric space* $(\mathcal{M}, \mathsf{dis})$ *is* $(s, t)$-bounded-error samplable *if there exists a randomized circuit* $\mathsf{ErrSmp}$ *of size* $s$ *such that for all* $0 \leq t' \leq t$ *and* $w \in \mathcal{M}$*,* $\mathsf{ErrSmp}(w, t')$ *outputs a random point* $w' \in \mathcal{M}$ *satisfying* $\mathsf{dis}(w, w') = t'$*.*

**Definition 7.** *Let* $C$ *be a set over a metric space* $\mathcal{M}$*. We say* $C$ *is a* $(t, \epsilon)$-maximal-error Shannon code *if there exists an efficient recover procedure* $\mathsf{Rec}$ *such that for all* $0 \leq t' \leq t$ *and* $c \in C$*,* $\Pr[\mathsf{Rec}(\mathsf{ErrSmp}(c, t')) \neq c] \leq \epsilon$*.*

**Definition 8.** *Let* $(\mathcal{M}, \mathsf{dis})$ *be a metric space that is* $(s, t)$*-bounded-error samplable by a circuit* $\mathsf{ErrSmp}$*. For a distribution* $C$ *over* $\mathcal{M}$*, we say* $C$ *is a* $(t, \epsilon)$-average-random-error Shannon code *if there exists an efficient recover procedure* $\mathsf{Rec}$ *such that* $\Pr[c \leftarrow C, t' \leftarrow \{0, \ldots, t\} : \mathsf{Rec}(\mathsf{ErrSmp}(c, t')) \neq c] \leq \epsilon$*.*

The following fact can be obtained by Markov's inequality.[1]

[1] A similar lemma was given in [8], but the proof has a flaw, which was pointed out by an anonymous reviewer. In their proof, a code was chosen by a probabilistic argument for every $t' \in \{0, \ldots, t\}$, but it was not guaranteed that the code is the same for every $t'$. Instead, we consider a code that corrects random errors for "random" $t'$, which is guaranteed to correct random errors for every $t'$ with a worse decoding error probability.

**Lemma 1.** *Let $C$ be a $(t,\epsilon)$-average-random-error Shannon code with recovery procedure* $\mathsf{Rec}$ *such that* $\mathrm{H}_\infty(C) \geq k$*. Then, there exists a set $C'$ with $|C'| \geq 2^{k-1}$ that is $(t, 2\epsilon(t+1))$-maximal-error Shannon code with recovery procedure* $\mathsf{Rec}$*.*

*Proof.* Since $C$ is a $(t,\epsilon)$-average-random-error Shannon code, we have that

$$\sum_{c\in\mathrm{Supp}(C)} \Pr[c \leftarrow C] \Pr_{t'\leftarrow\{0,\ldots,t\}}[\mathsf{Rec}(\mathsf{ErrSmp}(c,t')) \neq c] \leq \epsilon.$$

For $c \in \mathrm{Supp}(C)$, let $\epsilon_c = \Pr_{t'\leftarrow\{0,\ldots,t\}}[\mathsf{Rec}(\mathsf{ErrSmp}(c,t')) \neq c]$. By Markov's inequality, it holds that

$$\Pr_{c\leftarrow C}[\epsilon_c \leq 2\epsilon] = \Pr_{c\leftarrow C}[\epsilon_c \leq 2\mathbb{E}_{c'\leftarrow C}[\epsilon_{c'}]] \geq \frac{1}{2}.$$

Since $\mathrm{H}_\infty(C) \geq k$, there are at least $2^{k-1}$ codewords $c \in \mathrm{Supp}(C)$ satisfying $\epsilon_c \leq 2\epsilon$. Let $C'$ be the set of such codewords. For every $c \in C'$, we have that

$$\sum_{t'\in\{0,\ldots,t\}} \Pr[t' \leftarrow \{0,\ldots,t\}] \Pr[\mathsf{Rec}(\mathsf{ErrSmp}(c,t')) \neq c] \leq 2\epsilon, \tag{1}$$

which implies that $\Pr[\mathsf{Rec}(\mathsf{ErrSmp}(c,t')) \neq c] \leq 2\epsilon(t+1)$ for every $t' \in \{0,\ldots,t\}$. Otherwise, there exists $t' \in \{0,\ldots,t\}$ such that $\Pr[t' \leftarrow \{0,\ldots,t\}] \Pr[\mathsf{Rec}(\mathsf{ErrSmp}(c,t')) \neq c] > \frac{1}{t+1} 2\epsilon(t+1) = 2\epsilon$, which contradicts (1). Therefore, $C'$ is a $(t, 2\epsilon(t+1))$-maximal-error Shannon code. □

We prove that if the generation procedure is injectively-invertible, then the existence of a computational fuzzy extractor implies the existence of a maximal-error Shannon code.

**Lemma 2.** *Let $(\mathcal{M}, \mathsf{dis})$ be a metric space that is $(s_{\mathrm{smp}}, t)$-bounded-error samplable. Let $(\mathsf{Gen}, \mathsf{Rep})$ be an $(\mathcal{M}, m, \ell, t, s_{\mathsf{sec}}, \epsilon)$-computational fuzzy extractor with error 0. Let $s_{\mathrm{rep}}$ denote the size of the circuit that computes* $\mathsf{Rep}$*. If* $\mathsf{Gen}$ *is $(s_{\mathrm{inv}}, \eta, \xi)$-almost-injectively-invertible, and it holds that $s_{\mathrm{sec}} \geq s_{\mathrm{inv}} + s_{\mathrm{smp}} + s_{\mathrm{rep}}$, then there exists a value $p$ and a set $C$ with $|C| \geq (1-\xi)2^{\ell-1}$ that is a $(t, 2(\epsilon+\eta)(t+1))$-maximal-error Shannon code with recovery procedure* $\mathsf{InvGen}(\mathsf{Rep}(\cdot, p), p)$*.*

*Proof.* Let $W$ be an arbitrary distribution on $\mathcal{M}$ of min-entropy $m$. By the security property of the computational fuzzy extractor $(\mathsf{Gen}, \mathsf{Rep})$, we have that $\Delta^{s_{\mathrm{sec}}}((R,P),(U_\ell,P)) \leq \epsilon$ for $(R,P) \leftarrow \mathsf{Gen}(W)$.

Define the following procedure $D$:

1. On input $r \in \{0,1\}^\ell, p \in \{0,1\}^*$, and $t \in \mathbb{N}$, compute $w \leftarrow \mathsf{InvGen}(r,p)$.
2. $t' \leftarrow \{0,\ldots,t\}$.
3. $w' \leftarrow \mathsf{ErrSmp}(w,t')$.
4. If $\mathsf{Rep}(w',p) \neq r$, output 0. Otherwise, output 1.

The procedure $D$ "efficiently" checks whether $\mathsf{Rep}$ can correctly output the string $r$ from the corresponding $p$ and $w$ with random $t$-bounded errors. We need the efficiency of $D$ since otherwise the "error-correcting" property of $\mathsf{Rep}$ may not be taken over from the computational security of $(\mathsf{Gen}, \mathsf{Rep})$. The procedure $D$ can be implemented by a circuit of size $s_{\mathrm{inv}} + s_{\mathrm{smp}} + s_{\mathrm{rep}}$.

By the invertibility of $\mathsf{Gen}$ and the correctness property of $(\mathsf{Gen}, \mathsf{Rep})$, we have that $\Pr[D(R, P, t) = 1] \geq 1 - \eta$, where $(R, P) \leftarrow \mathsf{Gen}(W)$. Since $\Delta^{s_{\mathrm{sec}}}((R, P), (U_\ell, P)) \leq \epsilon$, if $s_{\mathrm{sec}} \geq s_{\mathrm{inv}} + s_{\mathrm{smp}} + s_{\mathrm{rep}}$, it holds that

$$\Pr[D(U_\ell, P, t) = 1] \geq 1 - (\epsilon + \eta).$$

By the averaging argument, there exists a value $p$ such that $\Pr[D(U_\ell, p, t) = 1] \geq 1 - (\epsilon + \eta)$. This implies that

$$\Pr\left[\begin{array}{l} w \leftarrow \mathsf{InvGen}(R, p), \\ t' \leftarrow \{0, \ldots, t\}, \\ w' \leftarrow \mathsf{ErrSmp}(w, t') \end{array} : \mathsf{Rep}(w', p) = R \right] \geq 1 - (\epsilon + \eta), \tag{2}$$

where $R = U_\ell$. Thus, the distribution $\mathsf{InvGen}(U_\ell, p)$ is a $(t, \epsilon + \eta)$-average-random-error Shannon code with recovery procedure $\mathsf{InvGen}(\mathsf{Rep}(\cdot, p), p)$. By applying Lemma 1, we can show that there is a set $C$ with $|C| \geq 2^{k-1}$ that is a $(t, 2(\epsilon + \eta)(t + 1))$-maximal-error Shannon code for $k \geq \mathrm{H}_\infty(\mathsf{InvGen}(U_\ell, p))$.

It follows from the almost-injective invertibility of $\mathsf{Gen}$ that $|\{w' : w' \leftarrow \mathsf{InvGen}(U_\ell, p)| \geq (1 - \xi)2^\ell$. Thus, $\mathrm{H}_\infty(\mathsf{InvGen}(U_\ell, p)) \geq \ell - \log(1/(1 - \xi))$. Therefore, the statement follows. □

It is known that a secure sketch can be constructed from a Shannon code, which is explicitly presented in [8], and implicitly stated in [6, Section 8.2].

**Lemma 3** ([6, 8])**.** *For an alphabet $Z$, let $C$ be a $(t, \delta)$-maximal-error Shannon code over $Z^n$. Then, there exists a $(Z^n, m, m - (n \log |Z| - \log |C|), t)$ secure sketch with error $\delta$ for the Hamming metric over $Z^n$.*

An information-theoretic fuzzy extractor can be constructed from a secure sketch and a strong extractor [6]. In particular, if we use universal hashing as strong extractor, we obtain the following result.

**Lemma 4** ([6])**.** *Let $(\mathsf{SS}, \mathsf{Rec})$ be an $(\mathcal{M}, m, \tilde{m}, t)$-secure sketch with error $\delta$, and $\mathsf{Ext}$ an $(n, \tilde{m}, \ell, \epsilon)$-strong extractor given by universal hashing (any $\ell \leq \tilde{m} - 2 \log(\frac{1}{\epsilon}) + 2$ can be achieved). Then, the following $(\mathsf{Gen}, \mathsf{Rep})$ is an $(\mathcal{M}, m, \ell, t, \epsilon)$-fuzzy extractor with error $\delta$:*

- $\mathsf{Gen}(w; r, x)$ *: set* $P = (\mathsf{SS}(w; r), x)$*,* $R = \mathsf{Ext}(w; x)$*, and output* $(R, P)$*.*
- $\mathsf{Rep}(w', (s, x))$ *: recover* $w = \mathsf{Rec}(w', s)$ *and output* $R = \mathsf{Ext}(w; x)$*.*

By combining Lemmas 2, 3, and 4, we obtain the following theorem.

**Theorem 1.** *Let $Z$ be an alphabet. Let $(\mathsf{Gen}, \mathsf{Rep})$ be a $(Z^n, m, \ell, t, s_{\mathsf{sec}}, \epsilon)$-computational fuzzy extractor with error 0. Let $s_{\mathrm{rep}}$ denote the size of the circuit that computes $\mathsf{Rep}$. If $\mathsf{Gen}$ is $(s_{\mathrm{inv}}, \eta, \xi)$-almost-injectively-invertible, and it holds that $s_{\mathrm{sec}} \geq s_{\mathrm{inv}} + n \log |Z| + s_{\mathrm{rep}}$, then there exists a $(Z^n, m, \ell', t, \epsilon')$ (information-theoretic) fuzzy extractor with error $2(\epsilon + \eta)(t + 1)$ for any $\ell' \leq m + \ell - n \log |Z| - \log(\frac{1}{1-\xi}) - 2 \log(\frac{1}{\epsilon'}) + 1$.*

In particular, in the above theorem, if we choose $m = n \log |Z|$, then a $(Z^n, n \log |Z|, \ell, t, s_{\mathsf{sec}}, \epsilon)$-computational fuzzy extractor implies a $\left(Z^n, n \log |Z|, \ell - \log\left(\frac{1}{1-\xi}\right) - 2 \log\left(\frac{1}{\epsilon'}\right) + 1, t, \epsilon'\right)$-fuzzy extractor with error $2(\epsilon + \eta)(t + 1)$.

As in the negative result of [8], we do not claim about the efficiency of the resulting fuzzy extractor. In our case, the non-explicit parts are (1) fixing the value $p$ in Lemma 2, and (2) constructing a maximal-error Shannon code from an average-random-error one in Lemma 1.

# 4 Positive Results

## 4.1 A Construction based on LR-KEM

We present a construction of a computational fuzzy extractor based on a leakage-resilient key encapsulation mechanism. First, we give a definition of leakage-resilient key encapsulation mechanism.

**Definition 9** (Leakage-Resilient Key Encapsulation Mechanism (LR-KEM))**.** *An* $(n, \ell, m, s, \epsilon)$*-LR-KEM scheme* $\Pi$ *is a tuple of randomized procedures* $(\mathsf{KEM.Gen}, \mathsf{KEM.Enc}, \mathsf{KEM.Dec})$ *with the following properties.*

- *The key generation procedure* $\mathsf{KEM.Gen}$ *on input a random string* $r \in \{0,1\}^n$ *outputs a pair* $(pk, sk)$ *of a public key and a secret key.*
- *The encryption procedure* $\mathsf{KEM.Enc}$ *on input a public key* $pk$ *outputs a ciphertext* $c$ *and a key* $k \in \{0,1\}^\ell$*.*
- *The decryption procedure* $\mathsf{KEM.Dec}$ *on input a secret key* $sk$ *and a ciphertext* $c$ *outputs a key* $k$*. The* correctness *property guarantees that for any* $(pk, sk) \leftarrow \mathsf{KEM.Gen}(1^n)$*,* $\Pr[(c, k) \leftarrow \mathsf{KEM.Enc}(pk) : \mathsf{KEM.Dec}(sk, c) = k] = 1$*.*
- *The* security *property guarantees that for any circuit* $A$ *of size at most* $s$ *and for any* $f : \{0,1\}^n \to \{0,1\}^\ell$ *satisfying* $\tilde{\mathrm{H}}_\infty(r|f(r)) \geq m$*, where* $r \leftarrow U_n$*, it holds that*

$$\Delta^s\left(\mathsf{Expt}^{\mathsf{leak}}_{\Pi,A}(0), \mathsf{Expt}^{\mathsf{leak}}_{\Pi,A}(1)\right) \leq \epsilon,$$

*where the experiment* $\mathsf{Expt}^{\mathsf{leak}}_{\Pi,A}(b)$ *is defined as follows:*

1. $r \leftarrow U_n$*.*
2. $(pk, sk) \leftarrow \mathsf{KEM.Gen}(r)$*.*
3. $(c, k) \leftarrow \mathsf{KEM.Enc}(pk)$*.*
4. $k_0 = k$*, and* $k_1 \leftarrow U_\ell$*.*
5. $b' \leftarrow A(pk, c, k_b, f(r))$*.*
6. *Output* $b'$*.*

A usual (non-leakage-resilient) KEM scheme is a special case of an $(n, \ell, m, s, \epsilon)$-LR-KEM scheme in which $f(r)$ is not given to $A$. We call such a scheme an $(n, \ell, s, \epsilon)$*-KEM scheme.*

Definition 9 is slightly different from the corresponding security of leakage-resilient public-key encryption considered in [1, 10] (cf. [10, Section 8]). In [1, 10], the leakage function can be applied to the secret key $sk$, and the restriction on $f$ is the output length $|f(sk)|$. Instead, in Definition 9, we consider the leakage of the random string $r$ of $\mathsf{Gen}$, and the restriction on $f$ is the residual entropy of $r$. Nevertheless, the difference is not crucial. Indeed, the same construction as [10, Section 8] gives a generic construction of a leakage-resilient KEM scheme from any KEM scheme and a strong extractor. Although the proof is almost the same as that of [10, Theorem 8.1], we give the proof for completeness and for a detailed analysis due to the treatment of the exact security in this paper.

**Lemma 5.** *Let* $\Pi = (\mathsf{KEM.Gen}, \mathsf{KEM.Enc}, \mathsf{KEM.Dec})$ *be an* $(n, \ell, s_{\mathrm{kem}}, \epsilon_{\mathrm{kem}})$*-KEM scheme, and* $\mathsf{Ext}$ *an* $(n, m, k, \epsilon_{\mathrm{ext}})$*-strong extractor. Then, the following* $\Pi' = (\mathsf{KEM.Gen}', \mathsf{KEM.Enc}', \mathsf{KEM.Dec}')$ *is an* $(n + t, \ell, m, s, \epsilon_{\mathrm{kem}} + 2\epsilon_{\mathrm{ext}})$*-LR-KEM scheme for any* $s \leq s_{\mathrm{kem}} - s_f$*, where* $t$ *is the length of a random string in* $\mathsf{Ext}$ *and* $s_f$ *is the size of the circuit for computing the leakage function* $f$*.*

- $\mathsf{KEM.Gen}'$ *: Choose* $r \in \{0,1\}^n$ *and* $x \in \{0,1\}^t$ *uniformly at random, and compute* $r' = (\mathsf{Ext}(r; x), x)$ *and* $(pk, sk) \leftarrow \mathsf{KEM.Gen}(r')$*. Output* $pk' = (pk, x)$ *and* $sk' = r$*.*
- $\mathsf{KEM.Enc}'$ *: On input* $pk' = (pk, x)$*, compute* $(c, k) \leftarrow \mathsf{KEM.Enc}(pk)$*. Output* $c' = (c, x)$ *and* $k$*.*
- $\mathsf{KEM.Dec}'$ *: On input* $sk' = r$ *and* $c' = (c, x)$*, compute* $(pk, sk) \leftarrow \mathsf{KEM.Gen}(\mathsf{Ext}(r; x), x)$ *and* $k = \mathsf{KEM.Dec}(sk, c)$*. Output* $k$*.*

*Proof.* Consider the following experiment $\mathsf{Expt}^{\mathsf{leak}'}_{\Pi,A}(b)$ for $b \in \{0,1\}$:

1. $r \leftarrow U_n$, $x \leftarrow U_t$, and $r' \leftarrow U_{n+k}$.
2. $(pk, sk) \leftarrow \mathsf{KEM.Gen}(r')$. Let $pk' = (pk, x)$ and $sk' = r$.
3. $(c, k) \leftarrow \mathsf{KEM.Enc}(pk)$. Let $c' = (c, x)$.
4. $k_0 = k$, and $k_1 \leftarrow U_\ell$.
5. $b' \leftarrow A(pk, c, k_b, f(r))$.
6. Output $b'$.

It follows from the triangle inequality that for any $s \in \mathbb{N}$,

$$\begin{aligned}
&\Delta^s(\mathsf{Expt}^{\mathsf{leak}}_{\Pi,A}(0), \mathsf{Expt}^{\mathsf{leak}}_{\Pi,A}(1)) \\
&\leq \Delta^s(\mathsf{Expt}^{\mathsf{leak}}_{\Pi,A}(0), \mathsf{Expt}^{\mathsf{leak}'}_{\Pi,A}(0)) && (3)\\
&\quad + \Delta^s(\mathsf{Expt}^{\mathsf{leak}'}_{\Pi,A}(0), \mathsf{Expt}^{\mathsf{leak}'}_{\Pi,A}(1)) && (4)\\
&\quad + \Delta^s(\mathsf{Expt}^{\mathsf{leak}'}_{\Pi,A}(1), \mathsf{Expt}^{\mathsf{leak}}_{\Pi,A}(1)). && (5)
\end{aligned}$$

The experiment $\mathsf{Expt}^{\mathsf{leak}'}_{\Pi,A}(b)$ is different from $\mathsf{Expt}^{\mathsf{leak}}_{\Pi,A}(b)$ only in the key generation phase, in which the uniformly random string $r'$ is used instead of the output of the strong extractor $\mathsf{Ext}$. Thus, for any $s \in \mathbb{N}$, equations (3) and (5) are upper-bounded by $\epsilon_{\mathrm{ext}}$.

The experiment $\mathsf{Expt}^{\mathsf{leak}'}_{\Pi,A}(b)$ is almost the same as the experiment for non-leakage-resilient KEM. The only difference is in the guessing phase, where $A$ is given $f(r)$. Thus, for any $s \in \mathbb{N}$, equation (4) is upper-bounded by $\epsilon_{\mathrm{kem}}$ if $s_{\mathrm{kem}} \geq s + s_f$.

Therefore, for any $s \leq s_{\mathrm{kem}} - s_f$, $\Delta^s(\mathsf{Expt}^{\mathsf{leak}}_{\Pi,A}(0), \mathsf{Expt}^{\mathsf{leak}}_{\Pi,A}(1))$ is upper-bounded by $\epsilon_{\mathrm{kem}} + 2\epsilon_{\mathrm{ext}}$. □

We give a construction of a computational fuzzy extractor based on a leakage-resilient KEM scheme.

**Theorem 2.** *Let* $(\mathsf{KEM.Gen}, \mathsf{KEM.Enc}, \mathsf{KEM.Dec})$ *be an* $(n, \ell, \tilde{m}, s_{\mathrm{sec}}, \epsilon)$*-LR-KEM scheme, and* $(\mathsf{SS}, \mathsf{Rec})$ *be an* $(\mathcal{M}, m, \tilde{m}, t)$*-secure sketch with error* $\delta$*. Let* $s_{\mathrm{gen}}$*,* $s_{\mathrm{enc}}$*, and* $s_{\mathrm{ss}}$ *denote the sizes of circuits that computes* $\mathsf{KEM.Gen}$*,* $\mathsf{KEM.Enc}$*, and* $\mathsf{SS}$*, respectively. Then, for any* $s \leq s_{\mathrm{sec}} - (s_{\mathrm{gen}} + s_{\mathrm{enc}} + s_{\mathrm{ss}})$*, the following* $(\mathsf{Gen}, \mathsf{Rep})$ *is a* $(\{0,1\}^n, m, \ell, t, s, \epsilon)$*-computational fuzzy extractor with error* $\delta$*:*

- $\mathsf{Gen}(w; r_1, r_2)$ *: compute* $(pk, sk) \leftarrow \mathsf{KEM.Gen}(w)$ *and* $(c, k) \leftarrow \mathsf{KEM.Enc}(pk; r_1)$*, set* $p = (c, \mathsf{SS}(w; r_2))$ *and* $r = k$*, and output* $(r, p)$*.*
- $\mathsf{Rep}(w', (c, ss))$ *: recover* $w = \mathsf{Rec}(w', ss)$*, compute* $(pk, sk) \leftarrow \mathsf{KEM.Gen}(w)$ *and* $K \leftarrow \mathsf{KEM.Dec}(sk, c)$*, and output* $K$*.*

*Proof.* The correctness property immediately follows from the correctness of the LR-KEM scheme and the secure sketch.

For the security property, we know that $\tilde{\mathrm{H}}_\infty(W|\mathsf{SS}(W)) \geq \tilde{m}$ from the security of the secure sketch, where $W$ is any random variable of min-entropy $m$. Thus, from the security of the LR-KEM scheme, for any $s \leq s_{\mathrm{sec}} - (s_{\mathrm{gen}} + s_{\mathrm{enc}} + s_{\mathrm{ss}})$, we have that $\Delta^s((R, P), (U_\ell, P)) = \Delta^s((K, C, \mathsf{SS}(W)), (U_\ell, C, \mathsf{SS}(W))) \leq \epsilon$. □

As for the LWE-based construction in [8], the above *KEM-and-sketch* construction does not require the entropy of $W$ conditioned on $P = (C, \mathsf{SS}(W))$. Indeed, $W$ may have no information-theoretic entropy conditioned on $P$.

Note that the above construction can be instantiated by a *secret-key* KEM scheme, in which the secret key is used for encryption.

Another approach to constructing computational fuzzy extractor is to apply a pseudorandom generator to the output of (information-theoretic) fuzzy extractor. We say this approach *FE-then-PRG*. Compared to the LWE-based construction [8] and the FE-then-PRG construction, our construction has an advantage in "stretching" the key. In the LWE-based construction, it seems necessary also to stretch the input $W$ to stretch the key, which is undesirable if the length of $W$ cannot be stretched (e.g., biometric data). In the FE-then-PRG construction, a straightforward way of stretching the key is to use a PRG multiple times. To stretch the output length, we need to apply the PRG in a nested manner. Namely, for a PRG $\mathsf{G} : \{0,1\}^\ell \to \{0,1\}^{2\ell}$, we can obtain the key with length $4\ell$ by computing $(\mathsf{G}(x_0), \mathsf{G}(x_1))$, where $\mathsf{G}(x) = (x_0, x_1)$ and $x, x_0, x_1 \in \{0,1\}^\ell$. Thus, we need *sequential* computation to obtain the final output. In the KEM-and-sketch construction, in order to stretch the key, we can use the same public key to generate multiple ciphertexts. Hence, the computation of encrypting keys and decrypting ciphertexts can be done in *parallel*.

### 4.2 A Construction based on Strong DDH

We give a simple construction of a computational fuzzy extractor based on a stronger variant of the Decisional Diffie-Hellman (DDH) assumption. Several stronger variants of the DDH assumption have been proposed in the literature (e.g., [4, 9]). We use a weaker variant of the strong DDH assumption used in [9].

**The strong DDH assumption.** For any polynomial $s(n)$, $\Delta^{s(n)}((g, g^a, g^b, g^{ab}), (g, g^a, b^b, g^c))$ is upper-bounded by a negligible function, where $g$ is a random generator of a group $\mathbb{G}$, $\mathbb{G}$ is a group of an $n$-bit prime order $q$, $a \in \mathbb{Z}_q$ and $c \in \mathbb{Z}_q$ are chosen uniformly at random, and $b \in \mathbb{Z}_q$ is chosen from a source of min-entropy $\alpha n$ for some constant $\alpha > 0$.

**Theorem 3.** *Assume that the strong DDH assumption holds. Let* $(\mathsf{SS}, \mathsf{Rec})$ *be a* $(\mathbb{Z}_q, m, \beta n, t)$*-secure sketch with error* $\delta$*. Then, the following* $(\mathsf{Gen}, \mathsf{Rep})$ *is a* $(\mathbb{G}, m, \log q, t, s, \epsilon)$ *computational fuzzy extractor with error* $\delta$ *for any polynomial* $s$ *and a negligible function* $\epsilon$ *in* $n$*:*

- $\mathsf{Gen}(w)$ *: Choose a random generator* $g \in \mathbb{G}$ *and a random element* $a \in \mathbb{Z}_q$*, set* $P = (g, g^a, \mathsf{SS}(w))$ *and* $R = g^{aw}$*, and output* $(R, P)$*.*
- $\mathsf{Rep}(w', (g, g^a, ss))$ *: recover* $w = \mathsf{Rec}(w', ss)$ *and output* $g^{aw}$*.*

*Proof.* The correctness property immediately follows from the correctness of the secure sketch.

For the security property, we know that $\tilde{\mathrm{H}}_\infty(W|P) \geq \beta n$ from the security of the secure sketch, where $W$ is a random variable of min-entropy $m$. Then, we have that, for a sufficiently large polynomial $s$,

$$\begin{aligned}
&\Delta^s((R, P), (U_{\log q}, P)) \\
&= \Delta^s((g^{aW}, g, g^a, \mathsf{SS}(W)), (U_{\mathbb{G}}, g, g^a, \mathsf{SS}(W))) \\
&\leq \Delta^s(g^c, g, g^a, \mathsf{SS}(W)), (U_{\mathbb{G}}, g, g^a, \mathsf{SS}(W))) + \epsilon(n) \\
&= \epsilon(n),
\end{aligned}$$

where $U_{\mathbb{G}}$ is the uniform distribution over $\mathbb{G}$, $c \in \mathbb{Z}_q$ is chosen uniformly at random, and $\epsilon(\cdot)$ is a negligible function. The inequality follows from the strong DDH assumption. □

## Acknowledgment

The authors are grateful to Masahiro Mambo for his helpful comments.

This work was supported in part by JSPS/MEXT Grant-in-Aid for Scientific Research Numbers 23500010, 23700010, 24240001, 25106509, 15H00851, 16H01705, and 17H01695.

## References

[1] A. Akavia, S. Goldwasser, and V. Vaikuntanathan. Simultaneous hardcore bits and cryptography against memory attacks. In O. Reingold, editor, *TCC*, volume 5444 of *Lecture Notes in Computer Science*, pages 474–495. Springer, 2009.

[2] J. Alwen, Y. Dodis, and D. Wichs. Survey: Leakage resilience and the bounded retrieval model. In *ICITS*, pages 1–18, 2009.

[3] X. Boyen. Robust and reusable fuzzy extractor. In P. Tuyls, B. Skoric, and T. Kevenaar, editors, *Security with Noisy Data*, pages 101–112. Springer, 2007.

[4] R. Canetti. Towards realizing random oracles: Hash functions that hide all partial information. In *CRYPTO*, pages 455–469, 1997.

[5] R. Cramer and V. Shoup. Design and analysis of practical public-key encryption schemes secure against adaptive chosen ciphertext attack. *SIAM J. Comput.*, 33(1):167–226, 2003.

[6] 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.

[7] Y. Dodis, L. Reyzin, and A. Smith. Fuzzy extractors. In P. Tuyls, B. Skoric, and T. Kevenaar, editors, *Security with Noisy Data*, pages 79–99. Springer, 2007. An updated version is available at http://www.cs.bu.edu/~reyzin/fuzzysurvey.html.

[8] B. Fuller, X. Meng, and L. Reyzin. Computational fuzzy extractors. In K. Sako and P. Sarkar, editors, *ASIACRYPT (1)*, volume 8269 of *Lecture Notes in Computer Science*, pages 174–193. Springer, 2013.

[9] Y. T. Kalai, X. Li, A. Rao, and D. Zuckerman. Network extractor protocols. In *FOCS*, pages 654–663. IEEE Computer Society, 2008.

[10] M. Naor and G. Segev. Public-key cryptosystems resilient to key leakage. *SIAM J. Comput.*, 41(4):772–814, 2012.

[11] K. Pietrzak. Provable security for physical cryptography. In *Western European Workshop on Research in Cryptology - WEWoRC 2009*, 2009.

[12] J. Radhakrishnan and A. Ta-Shma. Bounds for dispersers, extractors, and depth-two superconcentrators. *SIAM J. Discrete Math.*, 13(1):2–24, 2000.