On the Possibilities and Limitations of Computational Fuzzy Extractors

Kenji Yasunaga* Kosuke Yuzawa†

April 27, 2015

Abstract

Fuller et al. (Asiacrypt 2013) studied on computational fuzzy extractors, and showed, as a negative result, that the existence of a computational "secure sketch" implies the existence of an information-theoretically secure sketch with slightly weaker parameters. In this work, we show a similar negative result such that the existence of a computational fuzzy extractor satisfying a certain computational condition implies the existence of an information-theoretic fuzzy extractor with slightly weaker parameters. The condition is that the generation procedure of the fuzzy extractor can be efficiently invertible. This result implies that to circumvent the limitations of information-theoretic fuzzy extractors, we need to employ computational fuzzy extractors in which the generation procedure cannot be efficiently invertible. As positive results, we present a construction of computational fuzzy extractor based on a leakage-resilient key encapsulation mechanism and a construction based on a strong decisional Diffie-Hellman assumption.

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 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 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.

Dodis et al. [6] introduced the notion of *secure sketch*, which, on input w, produces an information that enables the recovery of w from any close value w' and does not reveal much information about w. Then, they show that a combination of secure sketches and strong extractors gives fuzzy extractors.

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

*Institute of Science and Engineering, Kanazawa University. Kakuma-machi, Kanazawa, 920–1192, Japan. yasunaga@se.kanazawa-u.ac.jp

†Graduate School of Natural Science and Technology, Kanazawa University. Kakuma-machi, Kanazawa, 920-1192, Japan. makku107@stu.kanazawa-u.ac.jp

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].

Fuller et al. [8] consider the *computational security* of fuzzy extractors to circumvent the limitations of information-theoretic fuzzy extractors. They gave both negative and positive results. On one hand, they show that secure sketches with computational security need to be subject to lower bounds from coding theory. In particular, they show that the existence of a computational secure sketch implies the existence of an information-theoretic secure sketch with slightly weaker parameters. On the other hand, they present a direct construction of a computational fuzzy extractor based on the hardness of learning with errors (LWE) problem.

In this work, we further study the limitations and possibilities of fuzzy extractors.

First, as a negative result, we show that, assuming that the generation procedure Gen can be efficiently invertible, computational fuzzy extractors also need to be subject to lower bounds from coding theory. Specifically, we show that if w can be efficiently computable from the pair (r, p) that can be generated by $\mathsf{Gen}(w)$, then the existence of a computational fuzzy extractor implies the existence of an information-theoretic fuzzy extractor with slightly weaker parameters. This negative result implies that in order to circumvent the limitation of the entropy loss of information-theoretic fuzzy extractors, we need to employ computational fuzzy extractors in which the generation procedure cannot be efficiently invertible. Indeed, there are extractors for structured sources that can be efficiently invertible [5].

Next, as a positive result, we give a construction of a computational fuzzy extractor based on 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. 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, which was proposed by Akavia et. al [1], where the leakage function is determined before choosing a public key. A generic construction of secure public-key encryption in this model is provided by Naor and Segev [10]. 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. See Section 4 for the details. Finally, we give a simple construction of a computational fuzzy extractor based on a stronger variant of the decisional Diffie-Hellman assumption.

Comparison to the Negative Results of Fuller et al. [8]

Fuller et al. noted in [8, footnote 3] that, if the generation procedure Gen can be efficiently invertible, their negative results for computational secure sketches can also be applied to computational fuzzy extractors. This observation is true if Gen is *injective*, but it is unclear if similar negative results hold for non-injective Gen from their results. Moreover, considering non-injective fuzzy extractors is important because it seems difficult to construct injective fuzzy extractors. We describe these facts below in more detail.

Let $(\mathsf{Gen}, \mathsf{Rep})$ be a computational fuzzy extractor. Assume that there is an efficient algorithm InvGen that, given (r, p), outputs w, where (r, p) was generated by $\mathsf{Gen}(w)$. Then, 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 secure

sketch and fuzzy extractor with information-theoretic security. 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, Gen is not injective. Namely, 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, at least one of w_1 and w_2 cannot be recovered by InvGen, and thus it seems difficult to use InvGen for constructing secure sketches.

In contrast, we give our negative result for computational fuzzy extractors even when Gen is not injective. In this sense, our result is a generalization of the negative results of [8].

Furthermore, to the best of our knowledge, no construction of injective fuzzy extractors is known so far. There is an intuitive reason for the non-existence of injective fuzzy extractors. 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 $\mathsf{Rep}(w_2,p)=r$. In this case, it seems natural to consider that the output range of $\mathsf{Gen}(w_2)$ contains (r,p). If so, this 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 $\Delta(X,Y)\le\epsilon$, we say X and Y are *ϵ-close*. We denote by U_ℓ the uniformly distributed random variable on $\{0,1\}^\ell$. 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 $\mathsf{dis}:\mathcal{M}\times\mathcal{M}\to\mathbb{R}^+=[0,\infty)$. 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.

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* δ *is a pair of randomized procedures* $(\mathsf{Gen},\mathsf{Rep})$ *with the following properties:*

- *The generation procedure* 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* 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* $\mathsf{dis}(w,w')\le t$*, 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* Gen *and* Rep*. If* $\mathsf{dis}(w,w')>t$*, no guarantee is provided about the output of* 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* $\Delta((R,P),(U_\ell,P))\le\epsilon$.

Definition 2 (Computational Fuzzy Extractor)**.** *An* $(\mathcal{M},m,\ell,t,s,\epsilon)$-computational fuzzy extractor *with error* δ *is a pair of randomized procedures* $(\mathsf{Gen},\mathsf{Rep})$ *that is an* $(\mathcal{M},m,\ell,t,\epsilon)$*-fuzzy extractor with error* δ *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* $\Delta^s((R,P),(U_\ell,P))\le\epsilon$.

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

- *The sketching procedure* SS *on input* $w \in \mathcal{M}$ *outputs a string* $s \in \{0,1\}^*$.
- *The recovery procedure* 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* $\mathsf{dis}(w, w') \leq t$, $\Pr[\mathsf{Rec}(w', \mathsf{SS}(s)) = w] \geq 1 - \delta$ *where the probability is taken over the coins of* SS *and* Rec. *If* $\mathsf{dis}(w, w') > t$, *no guarantee is provided about the output of* 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, ℓ, ϵ)-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 can be efficiently invertible.

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* Gen *is* (s, η)-invertible *if there exists a deterministic circuit* InvGen *of size at most* s *such that*

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

for any p *that can be generated as* $(r, p) \leftarrow \mathsf{Gen}(w)$ *for* $w \in \mathcal{M}$, *where* $R' = U_\ell$. *We say* Gen *is* errorless-*invertible if* $\mathsf{InvGen}(r, p)$ *outputs either* $\perp$ *(failure symbol) or* $w \in \mathcal{M}$ *for which there exists* r_G *such that* $(r, p) = \mathsf{Gen}(w; r_G)$.

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

Note that, since the inverter InvGen is confined to being deterministic, 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 show that if a fuzzy extractor has the perfect correctness, we can obtain the errorless invertibility for Gen.

Lemma 1. *Let* $(\mathsf{Gen}, \mathsf{Rep})$ *be a fuzzy extractor with error* 0. *If* Gen *is* (s, η)*-invertible, then* Gen *is* $(s + s_{\mathrm{rep}}, \eta)$*-errorless-invertible, where* s_{rep} *is the size of circuit* Rep.

Proof. Let InvGen be the inverter of (s, η)-invertibility of Gen. Then, we construct an inverter InvGen' such that on input (r, p), (1) run $w \leftarrow \mathsf{InvGen}(r, p)$, (2) output w if $\mathsf{Rep}(w, p) = r$, and output $\perp$ otherwise. The correctness property of $(\mathsf{Gen}, \mathsf{Rep})$ guarantees that the output of InvGen' is a valid inverse of (r, p). □

Since we prove our negative result for computational fuzzy extractors with errorless invertibility, Lemma 1 implies that our negative result can also be applied to computational fuzzy extractors with perfect correctness.

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* ErrSmp *of size* s *such that for all* $0 \le t' \le t$ *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,ϵ)-maximal-error Shannon code *if there exists an efficient recover procedure* Rec *such that for all* $t' \le t$ *and* $c \in C$, $\Pr[\mathsf{Rec}(\mathsf{ErrSmp}(c,t')) \neq c] \le \epsilon$.

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

The following fact can be obtained by Markov's inequality. (See [8] for the proof.)

Lemma 2 ([8])**.** *Let* C *be a* (t,ϵ)*-average-error Shannon code with recovery procedure* Rec *such that* $\mathrm{H}_\infty(C) \ge k$. *Then, there exists a set* C' *with* $|C'| \ge 2^{k-1}$ *that is* $(t, 2\epsilon)$*-maximal-error Shannon code with recovery procedure* Rec.

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

Theorem 1. *Let* $(\mathcal{M}, \mathsf{dis})$ *be a metric space that is* (s_{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* δ. *Let* s_{rep} *denote the size of the circuit that computes* Rep. *If* Gen *is* (s_{inv}, η)*-errorless-invertible, and it holds that* $s_{\mathrm{sec}} \ge s_{\mathrm{inv}} + t s_{\mathrm{smp}} + (t+1) s_{\mathrm{rep}}$, *then there exists a value* p *and a set* C *with* $|C| \ge 2^{-\log(2^{-\ell} + \frac{\rho}{|\mathcal{M}|}) - 1}$ *that is a* $(t, 2\rho)$*-maximal-error Shannon code with recovery procedure* $\mathsf{InvGen}(\mathsf{Rep}(\cdot, p), p)$, *where* $\rho = \epsilon + \eta + (t+1)\delta$.

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 $\Delta^{s_{\mathrm{sec}}}((R,P),(U_\ell,P)) \le \epsilon$ for $(R,P) \leftarrow \mathsf{Gen}(W)$.

Let InvGen be an inverter of the (s,η)-errorless-invertibility of Gen. We consider the modified inverter InvGen':

1. On input $r \in \{0,1\}^\ell$ and $p \in \{0,1\}^*$, compute $w \leftarrow \mathsf{InvGen}(r,p)$.
2. If $w \neq \bot$ and $\mathsf{Rep}(w,p) = r$, output w. Otherwise, output a random element in $\mathcal{M}$.

The procedure InvGen' can be implemented by a circuit of size $s_{\mathrm{inv}} + s_{\mathrm{rep}}$. Define the event E_{suc} such that

$$E_{\mathrm{suc}} = \{w \neq \bot \wedge \mathsf{Rep}(w,P) = R\},$$

where $(R,P) \leftarrow \mathsf{Gen}(W), w \leftarrow \mathsf{InvGen}(R,P)$. By the correctness property of $(\mathsf{Gen}, \mathsf{Rep})$ and the failure probability of InvGen, we have that $\Pr[E_{\mathrm{suc}}] \ge 1 - (\eta + \delta)$.

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. For all $1 \le t' \le t$, do the following:
   (a) $w' \leftarrow \mathsf{ErrSmp}(w,t')$.
   (b) If $\mathsf{Rep}(w',p) \neq r$, output 0. Otherwise, do nothing.
3. Output 1.

The procedure D "efficiently" checks whether 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 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 $s_{\mathrm{inv}}+ts_{\mathrm{smp}}+(t+1)s_{\mathrm{rep}}$. Note that in the procedure D, we can easily check whether the event E_{suc} occurs or not (by checking that a random element is produced in InvGen'). Thus, by the invertibility of Gen and the correctness property of $(\mathsf{Gen},\mathsf{Rep})$, we have that $\Pr[D(R,P,t)=1 \wedge E_{\mathrm{suc}}] \ge 1-(\eta+(t+1)\delta)$. Since $\Delta^{s_{\mathrm{sec}}}((R,P),(U_\ell,P)) \le \epsilon$, if $s_{\mathrm{sec}} \ge s_{\mathrm{inv}} + ts_{\mathrm{smp}} + (t+1)s_{\mathrm{rep}}$, it holds that

$$\begin{aligned}\Pr[D(U_\ell,P,t)=1 \wedge E_{\mathrm{suc}}] &\ge 1-(\epsilon+\eta+(t+1)\delta)\\ &= 1-\rho.\end{aligned}$$

By the averaging argument, there exists a value p such that $\Pr[D(U_\ell,p,t)=1 \wedge E_{\mathrm{suc}}] \ge 1-\rho$. This implies that, for all $1 \le t' \le t$,

$$\Pr\left[\begin{array}{l} w \leftarrow \mathsf{InvGen}'(R,p),\\ w' \leftarrow \mathsf{ErrSmp}(w,t') \end{array} : \mathsf{Rep}(w',p)=R \wedge E_{\mathrm{suc}}\right] \ge 1-\rho, \tag{1}$$

where $R = U_\ell$. Since the event E_{suc} implies that $\mathsf{InvGen}(R,p)=w$, we have that, for all $1 \le t' \le t$,

$$\Pr\left[\begin{array}{l} w \leftarrow \mathsf{InvGen}'(U_\ell,p),\\ w' \leftarrow \mathsf{ErrSmp}(w,t') \end{array} : \mathsf{InvGen}(\mathsf{Rep}(w',p),p)=w\right] \ge 1-\rho.$$

This implies that the distribution $\mathsf{InvGen}'(U_\ell,p)$ is a (t,ρ)-average-error Shannon code with recovery procedure $\mathsf{InvGen}(\mathsf{Rep}(\cdot,p),p)$. By applying Lemma 2, we can show that there is a set C with $|C| \ge 2^{k-1}$ that is a $(t,2\rho)$-maximal-error Shannon code for $k \le \mathrm{H}_\infty(\mathsf{InvGen}'(U_\ell,p))$.

Finally, we prove that $\mathrm{H}_\infty(\mathsf{InvGen}'(U_\ell,p)) \ge -\log(2^{-\ell}+\frac{\rho}{|\mathcal{M}|})$. Define

$$R_{\mathrm{good}} = \left\{ r \in \{0,1\}^\ell : \begin{array}{l} w \leftarrow \mathsf{InvGen}(r,p),\\ w \neq \perp \wedge \mathsf{Rep}(w,p)=r \end{array} \right\}.$$

By equation (1), it holds that $|R_{\mathrm{good}}| \ge (1-\rho)2^\ell$. Let $W_{\mathrm{good}} = \{\mathsf{InvGen}(r,p) : r \in R_{\mathrm{good}}\}$. By the definition of InvGen', we have that

$$\Pr[\mathsf{InvGen}'(U_\ell,p)=w] = \begin{cases} 2^{-\ell}+\frac{\rho}{|\mathcal{M}|} & w \in \mathcal{M} \cap W_{\mathrm{good}} \\ \frac{\rho}{|\mathcal{M}|} & w \in \mathcal{M} \setminus W_{\mathrm{good}}. \end{cases}$$

Therefore, the min-entropy of $\mathsf{InvGen}'(U_\ell,p)$ is $-\log(2^{-\ell}+\frac{\rho}{|\mathcal{M}|})$. □

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,δ)-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 δ 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 δ, and 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:*

- $\mathsf{Gen}(w; r, x)$ *: 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 Theorem 1 and Lemmas 3 and 4, we obtain the following corollary.

Corollary 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 δ. Let s_{rep} denote the size of the circuit that computes Rep. If Gen is (s_{inv}, η)-errorless-invertible, and it holds that $s_{\mathrm{sec}} \geq s_{\mathrm{inv}} + tn\log|Z| + (t+1)s_{\mathrm{rep}}$, then there exists a $(Z^n, m, \ell', t, \epsilon')$ (information-theoretic) fuzzy extractor with error 2ρ for any $\ell' \leq m - n\log|Z| - \log(2^{-\ell} + \frac{\rho}{|Z|^n}) - 2\log(\frac{1}{\epsilon'}) + 1$, where $\rho = \epsilon + \eta + (t+1)\delta$.*

In particular, in the above corollary, if we choose $m = n\log|Z|$ and $\frac{\rho}{|Z|^n} \leq 2^{-\ell}$, then a $(Z^n, n\log|Z|, \ell, t, s_{\mathsf{sec}}, \epsilon)$-computational fuzzy extractor implies a $(Z^n, n\log|Z|, \ell - 2\log(\frac{1}{\epsilon'}), t, \epsilon')$-fuzzy extractor with error 2ρ.

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, and (2) constructing a maximal-error Shannon code from an average-error one (Lemma 2) in Theorem 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 Π 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*

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

  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 say such a scheme an (n, ℓ, s, ϵ)-*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 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* 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* 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 $s \in \mathbb{N}$,

$$\Delta^s(\mathsf{Expt}^{\mathsf{leak}}_{\Pi,A}(0), \mathsf{Expt}^{\mathsf{leak}}_{\Pi,A}(1)) \leq \Delta^s(\mathsf{Expt}^{\mathsf{leak}}_{\Pi,A}(0), \mathsf{Expt}^{\mathsf{leak}'}_{\Pi,A}(0)) \tag{2}$$
$$+ \Delta^s(\mathsf{Expt}^{\mathsf{leak}'}_{\Pi,A}(0), \mathsf{Expt}^{\mathsf{leak}'}_{\Pi,A}(1)) \tag{3}$$
$$+ \Delta^s(\mathsf{Expt}^{\mathsf{leak}'}_{\Pi,A}(1), \mathsf{Expt}^{\mathsf{leak}}_{\Pi,A}(1)). \tag{4}$$

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 Ext. Thus, for any $s \in \mathbb{N}$, equations (2) and (4) are upper-bounded by ϵ_{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 (3) is upper-bounded by ϵ_{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* δ*. Let* s_{gen}*,* s_{enc}*, and* s_{ss} *denote the sizes of circuits that computes* $\mathsf{KEM.Gen}$*,* $\mathsf{KEM.Enc}$*, and* 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* δ*:*

- $\mathsf{Gen}(w; r_1, r_2)$ *: 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 $\tilde{\mathrm{H}}_\infty(W|SS(W)) \geq \tilde{m}$ 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.

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 with longer stretch. Although a PRG with any polynomial-length-stretch can be constructed from any PRG with one-bit-stretch, in its construction, we need to use the one-bit-stretch PRG in a nested manner. Namely, 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. Thus, 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 $\Omega(n)$.

We assume that, for some $n' < n$, there exist efficiently computable mappings B from $\mathbb{Z}_q$ to $\{0,1\}^{n'}$ and B' from $\{0,1\}^{n'}$ to $\mathbb{Z}_q$ that "preserve" the entropy of the input random variable. Specifically, we require that for a uniformly random variable X over $\mathbb{Z}_q$, $\Delta(B(X), U_{n'})$ is upper-bounded by a negligible function in n, and that for any random variable Y over $\{0,1\}^{n'}$ of min-entropy $\Omega(n)$, $\mathrm{H}_\infty(B'(Y)) \geq \Omega(n)$.

Theorem 3. *Assume that the strong DDH assumption holds. Let* $(\mathsf{SS}, \mathsf{Rec})$ *be a* $(\{0,1\}^{n'}, m, \Omega(n), t)$*-secure sketch with error* δ*. Then, the following* $(\mathsf{Gen}, \mathsf{Rep})$ *is a* $(\{0,1\}^{n'}, m, n', t, s, \epsilon)$ *computational fuzzy extractor with error* δ *for any polynomial* s *and a negligible function* ϵ *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* $R = B(g^{aB'(w)})$*, and output* (R, P)*.*
- $\mathsf{Rep}(w', (g, g^a, ss))$ *: recover* $w = \mathsf{Rec}(w', ss)$ *and output* $B(g^{aB'(w')})$*.*

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

For the security property, we know that $\tilde{\mathrm{H}}_\infty(W|P) \geq \Omega(n)$ 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,

$$\begin{aligned}
&\Delta^s((R,P),(U_{n'},P))\\
&=\Delta^s((B(g^{aB'(W)}),g,g^a,\mathsf{SS}(W)),(U_{n'},g,g^a,\mathsf{SS}(W)))\\
&\leq\Delta^s((B(g^c),g,g^a,\mathsf{SS}(W)),(U_{n'},g,g^a,\mathsf{SS}(W)))+\epsilon(n)\\
&\leq\epsilon'(n),
\end{aligned}$$

where $c\in\mathbb{Z}_q$ is chosen uniformly at random, and $\epsilon(\cdot)$ and $\epsilon'(\cdot)$ are negligible functions. The first inequality follows from the efficient computability of B and B', the entropy-preserving property of B', and the strong DDH assumption. The last inequality follows from the entropy-preserving property of B. □

Acknowledgments

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

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

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] M. Cheraghchi, F. Didier, and A. Shokrollahi. Invertible extractors and wiretap protocols. *IEEE Transactions on Information Theory*, 58(2):1254–1274, 2012.

[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.