CS-E4340 Cryptography: Exercise Sheet 2

—One-Way Functions & Pseudorandom Generators—

Submission Deadline: September 19, 11:30 via MyCourses

Each exercise can give up to two participation points, 2 for a mostly correct solution and 1 point for a good attempt. Overall, the exercise sheet gives at most 4 participation points. We encourage to **choose** those exercises which **interesting** and/or adequately challenging to you.

Exercise Sheet 2 is intended to help...

(a) ...understand the *definition* of pseudorandom generators (PRGs).
(b) ...understand the *relation* between PRGs and OWFs.
(c) ...develop intuition about the difficulty of *constructing* hardcore bits.
(d) ...continue familiarizing with the idea of *generic counterexamples*.
(e) ...familiarize yourself with proofs via *transformations* (which we will later call *reductions*).

Exercise 1 shows that a PRG might leak half of its input and thus, just like one-way functions, is not guaranteed to hide its input.

Ex. 2 & Ex. 3 show that PRGs are a strictly stronger notion than OWFs. Namely, Exercise 3 shows that every PRG is a OWF, while Exercise 2 shows that not every OWF is a PRG.

Counterexamples & Adversary transformation Exercise 1 and Exercise 2 help practice the notion of generic counterexamples, whereas Exercise 3 helps practice the notion of transformation of one adversary into another.

Ex. 4 & Ex. 5 aim to help understand the notion of universal hardcore bits.

Exercise 6 is advanced; you are asked to prove that the Goldreich-Levin hard-core bit is, indeed, a hard-core bit.

Exercise 7 is experimental—note that we do not know whether it has a solution.

Exercise 1 (**PRGs can leak half their input**)**.** Let $f : \{0,1\}^* \to \{0,1\}^*$ be a PRG. We define

$$g_f(x) = f(x_\ell)||x_r$$

Here, x_ℓ consists of the first $\lceil |x|/2 \rceil$ bits of x and x_r consists of the last $\lfloor |x|/2 \rfloor$ bits of x, i.e., $x = x_\ell||x_r$.

Task: Prove via reduction that if f is a PRG, then g_f is a PRG, too.

Exercise 2 (Some OWFs are not PRGs)**.** Assume the existence of length-preserving one-way functions.

Task: Show that there exists a length-expanding one-way function h which is not a PRG.

Exercise 3 (PRGs are OWFs)**.** Let g be a pseudorandom generator with $s(\lambda) := \lambda$, i.e., for all $x \in \{0,1\}^\lambda$, we have $|g(x)| = 2\lambda$.

Task: Prove that g is also a one-way function.

Exercise 4. (Universal Hardcore Bit I) A universal hardcore bit is a function $b : \{0,1\}^* \to \{0,1\}$ such that b is a hardcore bit for *all* one-way functions f. This and the next exercise show that such a universal hardcore bit cannot exist. Namely: Assume towards contradiction that there exists a universal hardcore bit b. Assume that there exists a one-way function f. Consider the one-way function $h_{b,f}$ defined as $h_{b,f}(x) := f(x)||b(x)$.

Task: Show via reduction that if f is a one-way function, then $h_{b,f}$ is a one-way function, too.

Exercise 5. (Universal Hardcore Bit II) As in the previous exercise, consider a (candidate) universal hardcore bit $b : \{0,1\}^* \to \{0,1\}$ and a one-way function f. Again, we look at the one-way function $h_{b,f}$ defined as $h_{b,f}(x) := f(x)||b(x)$.

Task: Show that b is not a hardcore bit for $h_{b,f}$.

Exercise 6. (Goldreich-Levin Hard-Core Bit) Let f_{base} be a polynomial-time computable function, and let f be defined (on even-lengths inputs) as $f(x||r) := f_{\text{base}}(x)||r$, where $|x| = |r|$. Assume that there is a PPT algorithm $\mathcal{A}$ using $p(n)$ random bits such that

$$\Pr[\mathcal{A}(f(x||r), 1^n) = b_{GL}(x, r)] = 1, \tag{1}$$

where the probability is over sampling $x \leftarrow\!\!\$\ \{0,1\}^n$, $r \leftarrow\!\!\$\ \{0,1\}^n$ and the randomness of the adversary $\mathcal{A}$.

Task: Construct a PPT algorithm $\mathcal{R}$ that uses $\mathcal{A}$ to invert f_{base} on even-length inputs.

Hint. The algorithm $\mathcal{R}$ uses $\mathcal{A}$ to recover x bitwise.

Advanced Task (optional): Does your reduction $\mathcal{R}$ still work if

$$\Pr[\mathcal{A}(f(x||r), 1^n) = b_{GL}(x, r)] = 1 - \mathsf{negl}(n). \tag{2}$$

If so, analyse why. If no, explain why not. Can you come up with a reduction which works in the case (2), where the success probability is only $1 - \mathsf{negl}(n)$ instead of 1?

Exercise 7 (Crocodile Experiment)**.** Give a function $f_{10} : \{1, .., 10\} \to \{1, .., 10\}$ which predicts the next number in the crocodile experiment in 80% of the data (based on the experimental data given in the materials section of the MyCourses page). (Doing this is considered solving the exercise, but here are further questions to consider if you like to:) Try to use less than 10 previous results, say, use only 5. Can you give a very simple prediction function f_3 or f_2 or even f_1 which is less accurate and takes only the last 3, 2, 1 numbers, respectively? Which reflections come to mind in terms of whether a more complex or a more simple function describes the experiment "better" (for this, we need a notion of what a "better" description is)? What is the easiest *distinguisher* which you can come up with for this experiment? Finally, assume that you are provided with a second crocodile. Can your function f_{10} be used for predicting the outputs of the second crocodile as well? On this line, is it possible to extend the function in such way that it includes a data collection loop at the beginning?