CS 388H: Cryptography **Fall 2021**

Homework 1: Symmetric Cryptography

Due: September 16, 2021 at 11:59pm (Submit on Gradescope) **Instructor:** David Wu

Instructions. You **must** typeset your solution in LaTeX using the provided template:

https://www.cs.utexas.edu/~dwu4/courses/fa21/static/homework.tex

You must submit your problem set via Gradescope (accessible through Canvas).

Collaboration Policy. You may discuss your general *high-level* strategy with other students, but you may not share any written documents or code. You should not search online for solutions to these problems. If you do consult external sources, you must cite them in your submission. You must include the names of all of your collaborators with your submission. Refer to the official course policies for the full details.

Problem 1: Pseudorandom Generators [20 points]. Let $G\colon \{0,1\}^\lambda \to \{0,1\}^n$ be a secure PRG. For each of the following functions G', indicate whether it is a secure PRG or not. If it is secure, give a *formal* proof. If not, describe an explicit attack.

(a) $G'(s) := G(s) \| (G(s) \oplus 1^n)$, where 1^n denotes the all-ones string of length n.

(b) $G'(s_1 \| s_2) := G(s_1) \oplus G(s_2)$.

(c) $G'(s_1 \| s_2) := s_1 \| G(s_2)$.

Recall that for two strings $s_1, s_2 \in \{0,1\}^\lambda$, we write $s_1 \| s_2 \in \{0,1\}^{2\lambda}$ to denote the *concatenation* of s_1 and s_2. Please refer to this handout for examples of how to formally show whether a construction is secure or not.

Problem 2: Encrypting Twice? [15 points]. Intuitively, encrypting a message twice should not harm security. It turns out that this is not always true. Let $(\mathsf{Encrypt}, \mathsf{Decrypt})$ be a cipher and define the "encrypt-twice" cipher $(\mathsf{Encrypt}_2, \mathsf{Decrypt}_2)$ where $\mathsf{Encrypt}_2(k,m) := \mathsf{Encrypt}(k, \mathsf{Encrypt}(k,m))$.

(a) Give an example of a cipher $(\mathsf{Encrypt}, \mathsf{Decrypt})$ that is semantically secure, but $(\mathsf{Encrypt}_2, \mathsf{Decrypt}_2)$ is not semantically secure.

(b) Suppose $(\mathsf{Encrypt}, \mathsf{Decrypt})$ is CPA-secure. Prove that $(\mathsf{Encrypt}_2, \mathsf{Decrypt}_2)$ is also CPA-secure.

Problem 3: Key Leakage in PRFs [20 points]. Let $F\colon \{0,1\}^n \times \{0,1\}^n \to \{0,1\}$ be a secure PRF. Use F to construct a function $F'\colon \{0,1\}^{n+1} \times \{0,1\}^n \to \{0,1\}$ with the following two properties:

- F' is a secure PRF.
- If the adversary learns the last bit of the key, then F' is no longer secure.

(a) Describe your construction $F'\colon \{0,1\}^{n+1} \times \{0,1\}^n \to \{0,1\}$. **Hint:** Consider changing the value of F at a single point.

(b) Show that if F is a secure PRF, then your construction F' is a secure PRF.

(c) Show that your construction F' is insecure against an adversary that learns the last bit of the key (i.e., at the beginning of the PRF security game, the challenger gives the last bit of the PRF key to the adversary). Specifically, give a complete description of your adversary and compute its advantage. This problem shows that leaking even a *single* bit of the secret key can break PRF security.

Problem 4: PRFs from PRGs [20 points]. We saw in lecture how to obtain a PRG from a PRF (e.g., "counter mode"). In this problem, we will explore the converse. Let $G\colon \{0,1\}^\lambda \to \{0,1\}^{2\lambda}$ be a *length-doubling* PRG. Define a candidate PRF $F\colon \{0,1\}^\lambda \times \{0,1\}^2 \to \{0,1\}^\lambda$ as follows:

> On input a key $s \in \{0,1\}^\lambda$ and an input $b_1 b_2 \in \{0,1\}^2$:
>
> 1. Compute $(s_0, s_1) \leftarrow G(s)$ where $s_0, s_1 \in \{0,1\}^\lambda$.
> 2. Compute $(s_{b_1 0}, s_{b_1 1}) \leftarrow G(s_{b_1})$ where $s_{b_1 0}, s_{b_1 1} \in \{0,1\}^\lambda$.
> 3. Output $s_{b_1 b_2}$.

(a) Show that if G is a secure PRG, then F is a secure PRF. **Hint:** Use a hybrid argument. Note that while you may define multiple intermediate hybrids, arguing indistinguishability between each pair of hybrids will likely be similar. If this is the case, you can just show the argument for one case and say that the others are similar.

(b) Suppose we wanted to use G to construct a PRF on a larger domain $\{0,1\}^n$, where $n = \text{poly}(\lambda)$. Show how to generalize the above construction to construct a PRF. In your construction, evaluating the PRF on an input $x \in \{0,1\}^n$ should require *at most n* invocations of G. While you do *not* need to formally prove the security of your construction, such a proof should be possible.

Problem 5: Time Spent [3 extra credit points]. How long did you spend on this problem set? This is for calibration purposes, and the response you provide does not affect your score.

Optional Feedback. Please answer the following *optional* questions to help us design future problem sets. You do not need to answer these questions. However, we do encourage you to provide us feedback on how to improve the course experience.

(a) What was your favorite problem on this problem set? Why?

(b) What was your least favorite problem on this problem set? Why?

(c) Do you have any other feedback for this problem set?

(d) Do you have any other feedback on the course so far?