CS6222 Grad Cryptography, Homework 4

Response by: Your Name, (Computing ID)

Total points: 40 awarded maximum. 51 available. Points are noted after each problem.

Instructions. For each problem, typeset your solution in the `answer` environment, and if there are sub-problems, mark them clearly. Use as much space as you require, and be sure to update your name and computing ID above, and the acknowledgements box at the end.

Policies. In short, you are encouraged to think about the problems on your own, and then discuss them and work toward solutions with your classmates. You must write and submit your own solutions. You may also read any published material that helps you come to an understanding of the problems, but you must acknowledge and/or reference any discussion or published material, with the exception of lecture notes, in-class and in-office-hours discussions, textbook sections we have covered, and basic LaTeX help or dictionary lookups. It is a violation of the honor code if any of the following occur:

- You copy text directly from any source.
- You use any material or discussion without acknowledgment or citation, excluding the above special cases.
- You are unable to explain your work orally.

See https://jackdoerner.net/teaching/2025/Fall/CS6222/#policies for more details.

Problem 1 (How Not to Authenticate a Message (14pts total))**.** Let $\{F_k : \{0,1\}^{|k|} \to \{0,1\}^{|k|}\}_{k\in\{0,1\}^*}$ be a PRF family. Your little brother came up with some ways to MAC long messages using just this PRF. Unfortunately they are all insecure. Demonstrate this fact by developing an attack for each one (but be nice about it, so you don't hurt his feelings). In each case, the $\mathsf{Gen}(1^n)$ algorithm simply samples a PRF key $k \leftarrow \{0,1\}^n$, the Ver algorithm recomputes the tag (as specified below) and compares the result to the putative tag it received, and your attack should work even when ℓ is a small constant.

(a) (3pts) The tag for a message $m_1\| \ldots \|m_\ell$ where $m_i \in \{0,1\}^{|k|}$ is $t := F_k(m_1) \oplus \ldots \oplus F_k(m_\ell)$.

(b) (3pts) The tag for a message $m_1\| \ldots \|m_\ell$ where $m_i \in \{0,1\}^{|k|/2}$ is $t := F_k(\langle 1\rangle \| m_1) \oplus \ldots \oplus F_k(\langle \ell\rangle \| m_\ell)$, where $\langle i\rangle$ denotes a $|k|/2$-bit encoding of i.

(c) (3pts) The tag for a message $m_1\| \ldots \|m_\ell$ where $m_i \in \{0,1\}^{|k|/2}$ is (r,t) where $r \leftarrow \{0,1\}^{|k|}$ and $t := F_k(r) \oplus F_k(\langle 1\rangle \| m_1) \oplus \ldots \oplus F_k(\langle \ell\rangle \| m_\ell)$, where $\langle i\rangle$ denotes a $|k|/2$-bit encoding of i.

Your little brother wasn't discouraged by your attacks. He thought really hard about it and came back with an even better way to MAC a long message using a CRHF family $\{H_s : \{0,1\}^{|s|+1} \to \{0,1\}^{|s|}\}_{s\in\{0,1\}^*}$. Prove to him that his new idea is also insecure.[1]

(d) (5pts) For every $s \in \{0,1\}^*$, let $H'_s : \{0,1\}^* \to \{0,1\}^{|s|}$ be a CRHF for arbitrary (polynomial) length inputs that is constructed from H_s using the Merkle-Damgård technique. Now the tag for a message $m \in \{0,1\}^*$ is (s,t) where $s \leftarrow \{0,1\}^{|k|}$ and $t := H'_s(k\|m)$.

[1] Unfortunately, this one was used in practice quite a lot.

Problem 2 (Random Linear Combination (3pts))**.** Fix $\ell > 0$ and a prime p. Let $\mathcal{K} = \mathbb{Z}_p^{\ell+1}$, $\mathcal{M} = \mathbb{Z}_p^{\ell}$, and $\mathcal{T} = \mathbb{Z}_p$. Let $\{h_{\vec{k}} : \mathcal{M} \to \mathcal{T}\}_{\vec{k}\in\mathcal{K}}$ be defined such that

$$h_{\vec{k}} : (m_1, \ldots, m_\ell) \mapsto \left(k_{\ell+1} + \sum_{i\in[\ell]} k_i \cdot m_i \bmod p \right).$$

Prove that $\{h_{\vec{k}}\}_{\vec{k}\in\mathcal{K}}$ is a *strongly* 2-universal hash-function family.

Problem 3 (CRHF or Not (3pts each))**.** Let $\{H_s : \{0,1\}^{2|s|} \to \{0,1\}^{|s|}\}_{s\in\{0,1\}^*}$ be a CRHF family that compresses inputs by a factor of two. For each of these sub-problems you must determine whether the proposed constructions of H'_s are *necessarily* CRHF families or not. Justify your answer by providing either a short proof (if H'_s *must be* a CRHF), or disproof (if it *might not be*). A disproof consists of an algorithm that finds a collision in H'_s with non-negligible probability given s. A disproof might depend the particular details of H_s (or G, in the third part), in which case you should describe the H_s (or G) on which your disproof depends, and prove that it is a CRHF (or PRG).

(a) $H'_s(x)$ outputs the first $n-1$ bits of $H_s(x)$

(b) $H'_s : x_1 \| x_2 \mapsto H_s(H_s(x_1) \| H_s(x_2))$ where $x_1, x_2 \in \{0,1\}^{2n}$

(c) $H'_s : x \mapsto H_s(G(x))$ where $x_1 \in \{0,1\}^{n+1}$ and $G : \{0,1\}^{n+1} \to \{0,1\}^{2n}$ is a PRG

Problem 4 (OWF and Not (4pts each))**.**

(a) Let $\{H_s : \{0,1\}^{2|s|} \to \{0,1\}^{|s|}\}_{s\in\{0,1\}^*}$ be a CRHF family that compresses inputs by a factor of two. Show that $f : s\|x \mapsto s\|H_s(x)$ is necessarily a strong OWF.

Hint: It's easy to see that, given s, a reduction can pick x uniformly and then use the adversary to compute $s\|x' \leftarrow \mathcal{A}(1^{|s|}, s\|H(x))$ such that $H_s(x') = H_s(x)$. Such a reduction only finds a collision when $x' \neq x$. You need to prove that the probability that $x = x'$ is small. There are a couple of different ways to do this. It might help you to read about *averaging arguments*, or to review your solution to Homework 1 Problem 6c.

(b) Let $\{H'_s : \{0,1\}^{|s|+1} \to \{0,1\}^{|s|}\}_{s\in\{0,1\}^*}$ be a CRHF family that compresses inputs by only one bit. Show that $f' : s\|x \mapsto s\|H'_s(x)$ is *not* necessarily a strong OWF (that is, describe an H'_s that is a CRHF assuming the existence of some other CRHF, and prove that f' is not a strong OWF by constructing an adversary that inverts it with non-negligible probability).

Hint: An adversary can contradict strong OWF security even if there is some large fraction of inputs on which it inverts with probability 0.

Problem 5 (Interactive Proofs (3pts each))**.** Recall that a language $L \subseteq \{0,1\}^*$ is in NP if and only if there exists some polynomial time Turing machine M and some polynomial q such that

- for every $x \in L$, there exists some $y \in \{0,1\}^{q(|x|)}$ such that $M(x,y) = 1$, and
- for every $x \notin L$, and for every $y \in \{0,1\}^{q(|x|)}$, $M(x,y) = 0$.

Recall also that if a language L is in NP, then it has a trivial interactive proof system with a deterministic verifier.

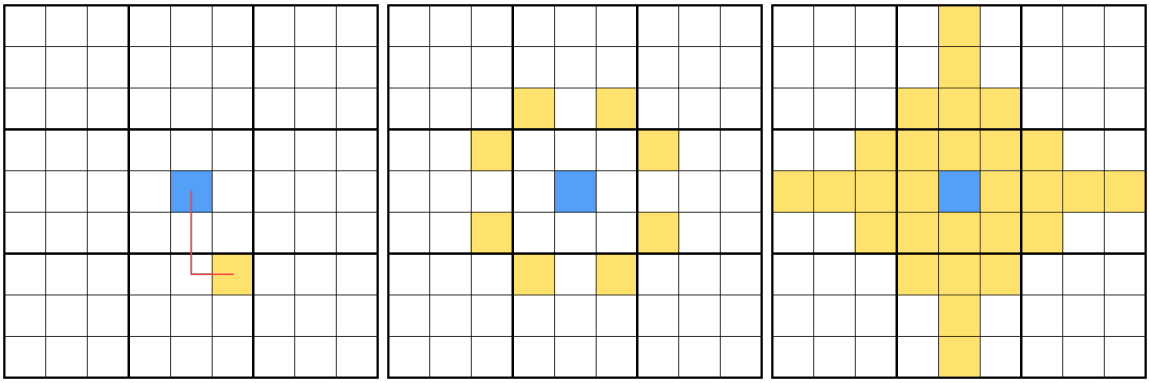

Figure 1: Illustrations of a Soduku puzzle for $n = 3$. **Left:** A single knights move. **Center:** A map of all cells that are one knights-move away from the blue cell. In any valid solution, the values in the yellow cells must be distinct from those in the blue cell, according to Rule 4. **Right:** A map of all cells that must be distinct from the blue cell, according to Rules 1-4. Illustration credit: [Sud25].

(a) Prove that if L has an interactive proof system in which the verifier is deterministic, then $L \in \mathsf{NP}$.

(b) Prove that if L has a perfectly sound interactive proof system,[2] then $L \in \mathsf{NP}$.

Problem 6 (A Practical Application of ZK Proofs (10pts))**.** *Knight's-move Sudoku* [Sud25] is a variant of the popular Sudoku puzzle game. Solutions to knights-move sudoku puzzles are only valid if every cell contains a number that is distinct from the numbers in the 8 cells that are one knights-move away (where a knights move is defined as in chess).

More formally, for $n \in \mathbb{N}^+$, a sudoku puzzle is a game played on a $n^2 \times n^2$ grid of cells. The grid is partitioned into n^2 disjoint sub-grids, arranged in an $n \times n$ pattern, and each sub-grid contains n^2 cells arranged in an $n \times n$ pattern. Some cells are empty and some contain a value from $[n^2]$.

A valid solution to a soduko puzzle is an assignment of integer values to empty cells that satisfies the following rules:

1. Every row of the puzzle contains a permutation on $[n^2]$.[3]
2. Every column of the puzzle contains a permutation on $[n^2]$.
3. Every sub-grid of the puzzle contains a permutation on $[n^2]$.

The knights-move variant adds a fourth rule:

4. Every pair of cells that are one knights-move apart from each other contain distinct values. The cells at locations $(x_1, y_1) \in [n^2] \times [n^2]$ and $(x_2, y_2) \in [n^2] \times [n^2]$ are one knights-move apart if and only if $\{|x_1 - x_2|, |y_1 - y_2|\} = \{1, 2\}$. This rule is illustrated in Figure 1

Solving a knights-move sodoku puzzle is hard! You want to prove that you can solve some particular puzzle (denoted x), without giving away the solution. Consider the language L that is defined to contain only solvable puzzles. For every $x \in L$, there exists at least one valid solution w that

[2] That is, one in which the verifier *never* accepts $x \notin L$ as a true statement, not even with negligible probability.
[3] That is, each of the integers from 1 to n^2 appears in the row exactly once.

satisfies the four rules enumerated above. It is easy to see that there is an efficient algorithm M such that $M(x, w) = 1$ if and only if $x \in L$ and w is a solution for x.

Design an interactive zero-knowledge proof system for the language of valid sudoku puzzles. Your solution should achieve perfect completeness, computational zero knowledge, and statistical soundness with a soundness error no greater than $1 - p(n)$ for some non-negligible function p. You must prove that all three properties hold. You can assume the existence of one-way permutations, and use the OWP-based commitment scheme for bits [Ps10, Section 4.7.1] that we discussed in class, which has perfect binding and computational hiding.

Problem 7 (The Revenge of Security through Obscurity (1pt))**.** Some of you have managed to find my homework problems (or equivalent ones) in various textbooks this semester. You can solve *this* problem either by solving it in the usual sense,[4] or by telling me exactly which textbook it came from.

This problem is hard in ways that will waste your time instead of teaching you cryptography. It might be fun if you're bored, but make sure you spend enough time on all your other obligations first! A solution is worth only 1 point, but the point it's worth is made out of gold. For this problem specifically, you may give the problem statement to ChatGPT or another LLM for help translating or finding the source; as usual you must understand the solution and write it yourself!

> 6. Пусть $G = \langle g \rangle$ – группа простого нечетного порядка q. Алгоритм решения задачи CDH: $(g, g^a, g^b) \mapsto g^{ab}$ может быть преобразован в алгоритм решения задачи SqDH: $(g, g^a) \mapsto g^{a^2}$. Доказать, что верно и обратное: алгоритм решения SqDH может быть преобразован в алгоритм решения CDH.

Acknowledgments

In this box, you should acknowledge your collaborators and the resources you used, if any.

Instructor's Acknowledgments

Problems in this homework have been borrowed from or inspired by a number of sources. Citations can be provided on request, after the homework is completed.

References

[Ps10] Rafael Pass and abhi shelat. *A Course in Cryptography*. 2010. http://www.cs.cornell.edu/courses/cs4830/2010fa/lecnotes.pdf.

[Sud25] Mastering Sudoku. What is chess sudoku? https://masteringsudoku.com/chess-sudoku/, 2025.

[4] "Obscured" solutions do not count.