Problem Set 3

This problem set is due *via email,* to `6.857-submit@theory.csail.mit.edu` on *Monday, October 16* by the beginning of class.

You are to work on this problem set in groups of three or four people. Problems turned in by individuals, pairs, pentuples, etc. will not be accepted. Be sure that all group members can explain the solutions. See Handout 1 (*Course Information*) for our policy on collaboration. If you do not have a group, let us know.

Homework must be submitted electronically! Each problem answer must appear on a separate page. Mark the top of each page with your group member names, the course number (6.857), the problem set number and question, and the date. We have provided templates for LaTeX and Microsoft Word on the course website (see the *Resources* page).

Grading and Late Policy: Each problem is worth 10 points. Late homework will not be accepted without prior approval.

With the authors' permission, we will distribute our favorite solution to each problem as the "official" solution—this is your chance to become famous! If you do not wish for your homework to be used as an official solution, or if you wish that it only be used anonymously, please note this on your homework.

Outside Source Policy: Please try to do these problems without using any outside sources (except perhaps the TA's.)–you'll learn more that way. **If you need to use outside sources, you must cite them in your writeup.**

Problem 3-1. Two-Time Pad

(a) Your group was e-mailed an encrypted file describing your grades from Problem Set 1. This file was encrypted using a one-time pad randomly generated especially for your group. Your task for this assignment is to decrypt the file and report the plaintext. (**Note:** you need to report the exact plaintext of the file—not just the grades!) You will find the information below useful in completing this task.

Your intrepid—and always security-minded—TAs decided it would be smart to encrypt your group's PS1 grades with a one-time pad. This way, we figured, we could securely communicate to you the scores. Accordingly, we generated a random one-time pad for each group. We then used each group's pad to encrypt a plaintext, 82-character sentence (encoded in standard ASCII), that describes their scores on each of the three problems. This is the file your group was e-mailed.

Feeling like we were on a roll, we decided to go ahead and also encrypt each group's submitted PS1 solution. We were stupid, however, and used the *same* one-time pad to encrypt each group's solutions as we had used to encrypt their grades. Furthermore, because our one-time pads were only 82-characters long, we just kept re-using it again and again to encrypt the entire problem set.

Soon realizing our folly, we deleted the encrypted the solutions. But as is often the case, deleting a file doesn't always erase all traces of its contents. A sneaky student was subsequently able to recover from our hard drives—using clever digital forensic tools—fragments of each group's encrypted problem set solution. Specifically:

- She was able to recover one 82-character fragment from each group's encrypted problem set.
- Each fragment is aligned with the one-time pad used to encrypt that group's grades. That is, if you XOR'd the fragment with the corresponding group's one-time pad, you would get the plaintext of the fragment.
- The plaintext corresponding to each of these fragments contains only standard ASCII characters.

- You can find a link to these fragments under the *Resources* pages of the course web site. Notice, the clever student wasn't able to determine which encrypted fragment belonged to which group, so she just assigned each a random name.
- Finally, remember that we encrypted the actual PS or PDF file you submitted to us for PS1—not just the text of your answers. That is, your group's fragment was drawn from the source code of the file you submitted (i.e., what you see when you open the PS or PDF file in a text editor).

Problem 3-2. Finite Fields

Arithmetic modulo a prime p is a known form of a finite field, denoted GF(p). Actually, there exists a finite field GF(q) (having q elements) whenever q is a prime power, e.g. $q = p^k$, for some positive integer k.

To create such a field, consider the set of univariate polynomials with indeterminate x and coefficients modulo p. Choose such a polynomial f(x) of degree k which is "irreducible" (i.e. "prime"—not the product of polynomials of lesser degree) and monic (highest term is x^k). Then do all operations modulo f(x).

In this problem, we ask you to demonstrate your understanding by giving an explicit construction for GF(9), a finite field with 9 elements.

Remember, an element of GF(9) will be a polynomial of the form $ax + b$ where a and b are in GF(3). We denote such a polynomial as "ab" for short (e.g. "12" denotes $x + 2$ and "20" denotes $2x$). There are 9 such elements. Specifically, we ask:

(a) Give the (9×9) addition table for GF(9).

(b) Let $f(x) = x^2 + 2x + 2$, which is irreducible. Give the (9×9) multiplication table for GF(9).

(c) Give a table showing the inverse of every nonzero element of $GF(9)$.

(d) Show that "10" (i.e. x) is a generator of $GF^*(9)$ (for this choice of $f(x)$).

(e) Find the order of each nonzero element of GF(9). (The order of a nonzero element c is the least positive integer t such that $c^t = 1$.

Problem 3-3. One-Time Commitment

A "commitment scheme" is a protocol between a sender S and a receiver R. The sender privately generates a message M (in an arbitrary manner). Then there are two phases:

- **Phase 1 (commitment):** The sender computes a value $c = C(M, v)$ and sends c to R. (Here v is some auxiliary information known only to S; it might for example be randomly chosen.)
- **Phase 2 (opening):** The sender sends the pair (M, v) to R. The receiver verifies that $C(M, v) = c$, the value received in phase 1. We say that S "opens" the commitment to "reveal" the message M.

The scheme is "binding" if, after phase 1, the sender can't open c in two different ways, revealing different values of M. It is "perfectly binding" if this is the case even when S is computationally unbounded.

The scheme is "hiding" if R, after phase 1, hasn't learned anything about the value of M from seeing the commitment c. It is "perfectly hiding" if this is the case if when R is computationally unbounded.

It is not hard to argue that a commitment scheme can't be both perfectly binding and perfectly hiding—if an unbounded receiver can't figure out M, there must be many solutions with different values of M to the equation $C(M, v) = c$, so it isn't binding.

However, it *is* possible to have a "one-time" commitment scheme that is both perfectly hiding and perfectly binding, with a little prior "setup" using the help of a trusted friend. It's a bit like setup for a one-time pad or a one-time MAC, except that S and R will get *different* secret setup values. Each pair of setup values (v, w) is used for only one commitment/reveal instance.

During setup, the friend gives v to S and w to R, where v and w may be related somehow (the friend is trusted to distribute appropriately related values of v and w). These values are distributed privately—S doesn't see w and R doesn't see v. The friend doesn't need to stay around after setup is done.

The commitment and reveal phases are the same, except that during the reveal phase R also checks that v and w are "appropriately related"; if not, S is caught cheating.

(a) Explain how to make such a "one-time commitment scheme" work. (i.e., describe the scheme).

Hint: Work modulo a large prime p, where the message M is a value modulo p, and where v and w are pairs of values modulo p.