Problem Set 2

This problem set is due *on paper,* on *Thursday, September 25* at 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, seek partners by emailing TA.

Homework must be typed! Each problem answer must appear on separate sheets of paper. Mark the top of each sheet with your name(s), the course number (6.857), the problem set number and question, and the date. **Homework must be typed and clear.** We have provided templates for LaTeX and Microsoft Word on the course website.

Grading and Late Policy: Each problem is worth 10 points. Late homework will not be accepted without prior approval. Homework should not be submitted by email except with prior approval. (*Somebody* from your group should be in class on the day that the homework is due.)

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.

Problem 2-1. Two-Time Pad

Your first job out of MIT is working at a secret agency in Maryland. The agency has intercepted six messages, each one 128 bytes in length. Your supervisor tells you that they were encrypted by "the enemy" using three one-time pads. You are given the assignment of decrypting them.

"A one time pad? But that's impossible to decrypt!" you say.

Your supervisor looks at you in disgust. "Don't they teach you how to count at MIT?"

You stare blankly at your supervisor.

"Six is bigger than three."

Slowly, you realize that a one-time pad can only be used once. You have six messages — so some of the pads must have been re-used.

You take the six messages back to your cubicle. Your mentor, a spook whose name cannot be revealed in this problem set, comes over to your desk. "This is the first test that everybody gets," your mentor says. "I'll give you a hint: each pad was used just twice."

"Can I have another hint?" you ask.

"Okay," your mentor says. "One of the messages is code in a well-known programming language. One of the messages contains some song lyrics. And one of the messages is from a well-known 17th century play. But I've said too much already. . . "

This problem has three parts, each of which is worth 10 points:

(a) Discuss your approach to solving this problem. Consider the ASCII encoding system, the mathematical properties of exclusive-or, predictability of human and computer languages, and any other factors that will help you solve this problem.

(b) Determine which pairs of ciphertext were encrypted using the same "one-time" pads.

(c) Report the six plaintexts.

You can download the ciphertexts from the 6.857 website.

Problem 2-2. Hash Soup

Estimate the probability that there are two non-identical files, somewhere on the planet[1], right now, that have the same MD5 hash code. Do the same for SHA-1. State your assumptions and cite all references used.

Problem 2-3. One-Time MAC, Revisited

Recall the one-time MAC presented in class: each message uses a new key $k = (a, b)$, where a and b are chosen randomly from $\mathbf{Z}_p$ for some large prime p. The MAC of a message $M < p$ is:

$$\mathrm{MAC}_k(M) = aM + b \pmod{p}$$

One problem with this approach is that for any n-bit message M, we actually need about $2n$ bits of key. Another problem is that the authentication tag is as large as the message itself, because p must be chosen to be larger than M, and the tag is an element of $\mathbf{Z}_p$.

Consider the following modification to the one-time MAC, which solves these problems. We choose p to be an approriate size (say, 64 bits long) and divide M into chunks $M = M_1, M_2, \ldots, M_t$ in some canonical way, where $0 \leq M_i < p$. The key is $k = (a_1, a_2, \ldots, a_t, b)$ where b and each a_i are chosen at random from $\mathbf{Z}_p$. The MAC is:

$$\mathrm{MAC}_k(M) = (\sum a_i M_i) + b \pmod{p}$$

Argue (as rigorously as you can) that this one-time MAC is secure against an unbounded adversary. For very long messages, how does the key usage compare to the one-time MAC presented in class? (As always, remember to cite your sources.)

[1] Earth