Massachusetts Institute of Technology
6.857: Network and Computer Security
Professor Ron Rivest

Handout **??**
March 31, 2014
Due: April 11, 2014

Problem Set 4

This problem set is due on *Friday, April 11* at **11:59 PM**. Please note that no late submissions will be accepted.

Please submit via MITx.

You can work on this problem set with a group of three or four students of your choosing. If you do not have a group, please email `6.857-tas@mit.edu` and we will assign you to a group. Be sure that all group members can explain the solutions. See Handout 1 (*Course Information*) for our policy on collaboration.

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: All problems are worth 10 points.

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 in your profile on the homework submission website.

Problem 4-1. Elliptic Curves

Consider an elliptic curve $y^2 = x^3 + 2x + 3 \bmod p$, where $p = 10^{12} + 39$. A point $P = (P_x, P_y) = (60736832995, 733944796939)$ is a generator for this curve.

(a) Compute $2P$.

(b) Compute $P + Q$, where $Q = (339994762892, 473279195180)$.

(c) Compute $497P$.

(d) What is the order of Q to the base P? That is, determine a such that $aP = Q$.

Note: For this problem we require that you don't use existing implementations of discrete logarithm algorithms, but rather roll your own. You may use existing implementations of elliptic curve arithmetic, though.

Problem 4-2. The need for good randomness

In this problem we will explore how a good encryption scheme can become insecure, when the way of generating random bits is not cryptographically strong.

We say that an encryption scheme $(\mathsf{Gen}, \mathsf{Enc}, \mathsf{Dec})$ achieves computational indistinguishability for many messages if no probabilistic polynomial time adversary can win in the following game with probability more than negligibly greater than $\frac{1}{2}$.

- Challenger first samples public and secret key pair $(\mathsf{pk}, \mathsf{sk}) \leftarrow \mathsf{Gen}$ and submits pk to the adversary;
- Adversary responds with two sequences of messages $(m_1^0, m_2^0, \ldots, m_k^0)$ and $(m_1^1, m_2^1, \ldots, m_k^1)$ (for k chosen by the adversary);
- Challenger chooses a random bit b and returns to the adversary the encryptions $(c_1, c_2, \ldots, c_k)$, where $c_i = \mathsf{Enc}(\mathsf{sk}, m_i^b)$;
- Adversary outputs a bit b' and wins if $b = b'$.

One can prove that El Gamal encryption scheme, where r used in encryption are chosen uniformly at random, achieves computational indistinguishability for many messages.

(a) Usually (in practice), the r used for encryption is not generated completely at random, but by using a pseudorandom generator. Again, in practice, people use simple "pseudorandom" generators such as the linear congruential generator. A linear congruential generator LCG, given seed s_0 outputs a sequence $(s_1, s_2, \ldots, s_n)$ such that $s_i = as_{i-1} + b \pmod{|G|}$. Parameters a and b are assumed to be public.

Show that if the r's are generated using an LCG, then given *two* ciphertexts (of two messages), one can establish a non-trivial relationship between them. Here r refers to randomness used during encryption in the ElGamal scheme. More precisely, show that the ElGamal cryptosystem with LCG does not achieve computational indistinguishability for many messages.

("Lesson": Beware of using unproven, patched-up pseudorandom generators).

(b) Finally, assume that the r's are generated by a cryptographically strong pseudorandom generator. Argue that ElGamal remains secure.

(In other words, show that if there is an adversary $\mathcal{A}$ that breaks the ElGamal that uses a cryptographically strong PRG, then there is an adversary $\mathcal{B}$ that breaks the original ElGamal — one where r's are generated randomly)

Problem 4-3. Changing RSA public keys

Bitdiddle Inc. requires every employee to have an RSA public key. It also requires that the employee change his or her RSA key at the end of every month.

(a) Alice just started working at Bitdiddle, and her first public key is (n, e) where n is the product of two safe primes, and $e = 3$.

Whenever a new month starts, Alice (being lazy) changes her public key as little as possible to get by the auditors. What she does, in fact, is just to advance her public exponent e to the next prime number. So, month by month, her public keys look like:

$$(n, 3), (n, 5), (n, 7), (n, 11), \ldots$$

Explain how Alice's laziness might get her in trouble.

(b) The next year, Alice tries a different scheme.

In January, she generates a fresh public key (n, e) where n is the product of two primes, p and q.

In February, she advances p to the next prime p' after p, and q to the next prime q' after q, and sets her public key to be (n', e') for a suitable e'.

Similarly, in March, she advances p' to p'' and q' to q'', and so on.

Explain how Alice's scheme could be broken.