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

Handout **??**
March 10, 2012
Due: March 24, 2012

Problem Set 3

This problem set is due on *Monday, March 24* at **11:59 PM**. Please note that no late submissions will be accepted. We are experimenting with a new submission site via MITx. More details will come later this week.

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 3-1. Kleptography

Alice has purchased a closed-source encryption box from the Acme Corporation. The specification of the box says that it will take as input a 128 bit key K and a variable length message M, and will output $EncCBC(K, M)$ with a random IV. Unfortunately, Alice can't check that the IVs are truly random. Alice can't inspect the box's internals, so it could have a mutable internal state that it uses to generate pseudo-random IVs.

(a) How can the Acme corporation effectively backdoor the hardware in a way that will be undetectable by polynomial-time black box analysis. That is, you should describe a back door which allows the Acme Corporation to find Alice and Bob's shared key in time which is polynomial in the number of messages sent by Alice.

However, if Alice (adaptively) tests her box on inputs of her choosing, she should not be able to distinguish between a back-doored box and an honest box.

You may assume that every message plaintext is distinguishable from random bytes in time polynomial in the message length. Can you make your backdoor work even if Acme sees just one of the ciphertexts generated by Alice?

Hint: The IV has as much information content as the key does, but the box generated it deterministically as a function of the key, that would be detectable.

(b) Now Alice is convinced that she can't trust the box, so she demands proof that she can trust Acme. Design a new specification for the generation of the IVs such that:

- Alice can verify the correctness of her box's outputs.
- A correct box does not compromise the confidentiality of encrypted messages.

Problem 3-2. IND-CCA

Alice and Bob have taken 6.857, and would like to send confidential messages to each other in a way that meets the IND-CCA definition given there.

Bob suggests the following method.

Let $EncCBC(K, M)$ denote the process of padding M with a 1 bit and then as many 0 bits as needed to make the message length a multiple of 128 bits, and then using AES in CBC mode to encrypt the result, using a randomly-chosen 128-bit IV. The result is:

$$EncCBC(K, M) = IV, C_1, C_2, \ldots, C_n \ .$$

Let $Rev(S)$ denote the reverse of a sequence S. That is, S is parsed as a sequence of 128-bit blocks (the length of S must be a multiple of 128 bits), and these blocks are re-arranged into reverse order.

Bob then proposes that he and Alice secretly agree on two AES keys K_1 and K_2, and then define the encryption of a message M as

$$E(K_1, K_2; M) = EncCBC(K_2, Rev(EncCBC(K_1, M))) \ .$$

The intent here is that the reversal avoids the problem noted in class, that "ciphertext prefixes decode as prefixes of the message."

Show that this method does *not* meet the IND-CCA security definition given in class. That is, show how the adversary can "win" the security game against this method.

Problem 3-3. Kalns

Ben Bitdiddle has designed a new cryptosystem called `Kalns`, but we suspect it might not be as strong as we would like to be. Therefore we ask your help to break it.

In this problem we will be working with a finite field GF_{16}. The elements of our field are all 4-bit strings. The field addition is computed as `xor`: $\mathsf{GF}_{16}(x) + \mathsf{GF}_{16}(y) = \mathsf{GF}_{16}(x \oplus y)$. We provide the two tables describing addition and multiplication laws on the course web page.

If you are curious, these tables are obtained by interpreting 4-bit field elements as degree ≤ 4 polynomials over GF_2 and performing addition and multiplication modulo the irreducible polynomial $x^4 + x + 1$.

However, for the purposes of this problem you do *not* need to understand how our GF_{16} is constructed; the solutions we know assume black-box access to GF_{16}. We have provided a GF_{16} implementation for you.

`Kalns` is a 64-bit block cipher. The secret key consists of three parts:

- an invertible 16-by-16 matrix A over GF_{16};
- a 16-element vector b over GF_{16}; and
- a permutation (bijection) S that maps GF_{16} one-to-one and onto GF_{16}.

To encrypt a 64-bit block B we first break it up in sixteen 4-bit chunks and interpret each of them as a GF_{16} element. So block B corresponds to length 16 vector $x = (x_0, \ldots, x_{15})$ over GF_{16}.

The encryptions consists of the following: $y = S(Ax + b)$, where the permutation S is individually applied to each of 16 elements of $v = Ax + b$. The 16-element vector y is later re-interpreted as 64-bit integer to obtain the encrypted block B'.

(a) Ben suspects that his cryptosystem is very secure. After all it has around $16^{16^2} \cdot 16^{16} \cdot 16! \approx 2^{1132.25}$ possible keys. However, we suspect that there are many equivalent keys. These keys have different values for (A, b, S), but produce the same ciphertext for any given plaintext. Is our suspicion well-founded?

(b) Describe a chosen-ciphertext attack on `Kalns` that recovers the unknown key (A, b, S) or an equivalent key.

(c) To demonstrate that your attack works, write an implementation that can break a randomly generated key. We have set up a web server that implements `Kalns` algorithm at

`http://6857.scripts.mit.edu/kalns/` .

Prove that your implementation works by getting your team's name on the list of successful teams. The server is powered by `scripts.mit.edu` shared hosting infrastructure. Please don't issue denial of service attacks or try to attack their servers.