CS255: Cryptography and Computer Security **Winter 2001**

Assignment #1

Due: Wednesday, January 31st, 2001.

Problem 1 Let p be a 128-bit prime and let $\mathbb{Z}_p$ be the set of integers $\{0, \ldots, p-1\}$. Consider the following encryption scheme. The secret key is a pair of integers $a, b \in \mathbb{Z}_p$ where $a \neq 0$. An encryption of a message $M \in \mathbb{Z}_p$ is defined as:

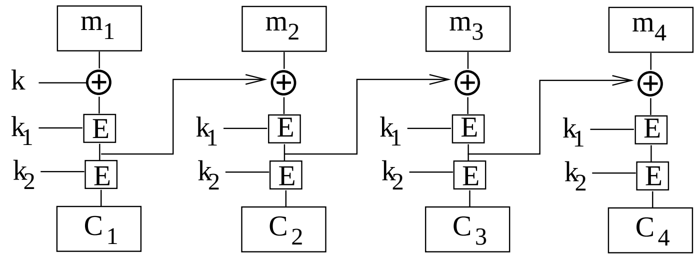
$$E_{a,b}[M] = aM + b \pmod{p}$$

a. Show that when E is used to encrypt a message $M \in \mathbb{Z}_p$ the system has perfect secrecy in the sense of Shannon.

b. Show that if the system is used to encrypt messages M_1, M_2 then the system does not have perfect secrecy. Hence, although the system has perfect secrecy for one message it is not very useful as is.
Hint: consider the case $M_1 = M_2$.

c. Show that given two random plaintext/ciphertext pairs $C_i = E_{a,b}[M_i]$ for $i = 1, 2$ it is possible to recover the key a, b with high probability.

Problem 2 Let E, D be the encryption/decryption algorithms of a certain block cipher. Consider the following chaining method for double DES like encryption:



The secret key is a triple (k, k_1, k_2) where k is as long as E's block size (64 bits for DES) and k_1, k_2 are as long as E's key size (56 bits for DES). For example, when E is DES the total key size is 64+56+56 = 176 bits.

a. Describe the decryption circuit for this system.

b. Show that using two short chosen ciphertext decryption queries an attacker can recover the full key (k, k_1, k_2) in approximately the time it takes to run algorithm D 2^ℓ times (i.e. the attack running time should be $O(2^\ell \text{time}(D))$. Here ℓ is the block cipher's key-length (56 bits for DES). Your attack shows that this system can be broken much faster than exhaustive search.
Hint: Consider the two decryption queries $\langle C_1, C_2, C_3, C_4 \rangle$ and $\langle C'_1, C_2, C'_3, C_4 \rangle$ where $C_1, \ldots, C_4$ and C'_1, C'_3 are random ciphertext blocks.

Problem 3 Before DESX was invented, the researchers at RSA Labs came up with DESV and DESW, defined by

$$\begin{aligned} DESV_{kk_1}(M) &= DES_k(M) \oplus k_1 \text{ and} \\ DESW_{kk_1}(M) &= DES_k(M \oplus k_1) \end{aligned}$$

As with DESX, $|k| = 56$ and $|k_1| = 64$. Show that both these proposals do not increase the work needed to break the cryptosystem using brute-force key search. That is, show how to break these schemes using on the order of 2^{56} DES encryptions/decryptions. You may assume that you have a moderate number of plaintext-ciphertext pairs, $C_i = DES\{V/W\}_{kk_1}(M_i)$.

Problem 4 KQIM (Internet Music Station) wishes to broadcast streamed music to its subscribers. Non-subscribers should not be able to listen in. When a person subscribes she is given a software player with a number of secret keys embedded in it. KQIM encrypts the broadcast using a 128-bit AES key K. The secret keys in each legitimate player can be used to derive K and enable legitimate subscribers to tune in. When a subscriber cancels her subscription, KQIM will encrypt future broadcasts using a new key K'. All valid players should be able to derive K', however the canceled subscriber should not.

a. Suppose the total number of potential subscribers is less than $n = 10^5$. Let $R_1, R_2, \ldots, R_n$ be 128-bit random values. The player shipped to subscriber number u contains all the R_i's except for R_u (i.e. the player contains 99999 keys). Let S be the set of currently subscribed users. Show that KQIM can construct a key K used to encrypt the broadcast so that any subscriber in S can derive K (from the R_i's in her player) while any single subscriber outside of S cannot derive K. You may assume that the set S is known to everyone (e.g. it is part of the broadcast). Briefly explain why your construction satisfies the required properties.

b. Is your construction in part (a) collusion resistant? That is, can two canceled subscribers combine the secrets embedded in their player to build a new operational player?

Remark: much better solutions to this problem exist.

Problem 5 Given a cryptosystem E_k, define the randomized cryptosystem F_k by

$$F_k(M) = (E_k(R), R \oplus M),$$

where R is a random bit string of the same size as the message. That is, the output of $F_k(M)$ is the encryption of a random one-time pad along with the original message XORed with the random pad. A new independent random pad R is chosen for every encryption.

We consider two attack models. The goal of both models is to reconstruct the actual secret key k.[1]

- In the key-reconstruction chosen plaintext attack (KR-CPA), the adversary is allowed to generate strings $M_1, M_2, \ldots$ and for each M_i learn a corresponding ciphertext.
- In the key-reconstruction random plaintext attack (KR-RPA), the adversary is given random plaintext/ciphertext pairs.

[1] This is a very strong goal - one might be able to decrypt messages without ever learning k.

Note that for the case of F_k the opponent has no control over the random pad R used in the creation of the given plaintext/ciphertext pairs. Clearly a KR-CPA attack gives the attacker more power than a KR-RPA attack. Consequently, it is harder to build cryptosystems that are secure against KR-CPA.

Prove that if E_k is secure against KR-RPA attacks then F_k is secure against $\mathsf{KR}-\mathsf{CPA}$ attacks.

Hint: It is easiest to show the contrapositive. Given an algorithm A that executes a successful $\mathsf{KR}-\mathsf{CPA}$ attack against F_k, construct an algorithm B (using A as a "subroutine") that executes a successful $\mathsf{KR}-\mathsf{RPA}$ attack against E_k. First, define precisely what algorithm A takes as input, what queries it makes, and what it produced as output. Do the same for B. Then construct an algorithm B that runs A on a certain input and properly answers all of A's queries. Show that the output produced by A enables B to complete the $\mathsf{KR}-\mathsf{RPA}$ attack against E_k.

Problem 6 We study a variant of the Linear Congruential Generator. This class of Pseudo Random Number Generators (PRNG) is insecure for cryptographic purposes, and yet it keeps appearing in security systems. Consider the following generator. The fixed public parameters of the generator are a 128-bit prime p, and three integers a, b, c. Let $\mathbb{Z}_p = \{0, 1, \ldots, p-1\}$. The seed for the generator is a pair $(s_1, s_2) \in \mathbb{Z}_p^2$. The generator works as follows:

1. Let (x_1, x_2) be the current state of the generator (initially the state is equal to the seed). Output $cx_1 + x_2 \bmod p$ as the current random block.
2. Set the new state to be the pair $(ax_1 + x_2,\ bx_2 + x_1) \bmod p$ and goto Step 1.

Show that no matter what parameters a, b, c are used, after observing a few consecutive outputs of the generator it is easy to predict all future outputs.