CS255: Cryptography and Computer Security **Winter 2007**

Assignment #1

Due: Wednesday, Jan. 31st, 2007.

Problem 1 Secret sharing.

a. Suppose Alice shares a secret block cipher key, K_{AB} with Bob, and a different secret block cipher key, K_{AC} with Charlie. Describe a method for Alice to encrypt an m-block message such that it can only be decrypted with the cooperation of both Bob and Charlie. The ciphertext should only be a constant size greater than m blocks. You may assume that Bob and Charlie have a pre-established secret channel on which to communicate.

b. Now, suppose Alice shares a block cipher key, K_{AB} with Bob, a block cipher key K_{AC} with Charlie, and a block cipher key K_{AD} with David. Describe a method for Alice to encrypt an m-block message such that any two of Bob, Charlie, and David can decrypt (for example, Bob and Charlie can decrypt), but none of them can decrypt the message themselves. Again, the ciphertext should only be a constant size greater than m blocks. **Hint:** Pick a random message encryption key to encrypt the message with. Then add three ciphertext blocks to the ciphertext header.

c. How does your solution from part (b) scale as we increase the number of recipients? In other words, suppose Alice has a secret key with each of n recipients and wants to encrypt so that any k out of n recipients can decrypt, but any $k-1$ cannot. What would be the length of the header as a function of n and k?

Your answer shows that this solution scales poorly. We will discuss a far more efficient solution later on in the class.

Problem 2 The movie industry wants to protect digital content distributed on DVD's. We study one possible approach. Suppose there are at most a total of n DVD players in the world (e.g. $n = 2^{32}$). We view these n players as the leaves of a binary tree of height $\log_2 n$. Each node v_i in this binary tree contains an AES key K_i. These keys are kept secret from consumers and are fixed for all time. At manufacturing time each DVD player is assigned a serial number $i \in [0, n-1]$. Consider the set S_i of $\log_2 n$ nodes along the path from the root to leaf number i in the binary tree. The manufacturer of the DVD player embeds in player number i the $\log_2 n$ keys associated with the nodes in S_i. In this way each DVD player ships with $\log_2 n$ keys embedded in it (these keys are supposedly inaccessible to consumers). A DVD movie M is encrypted as

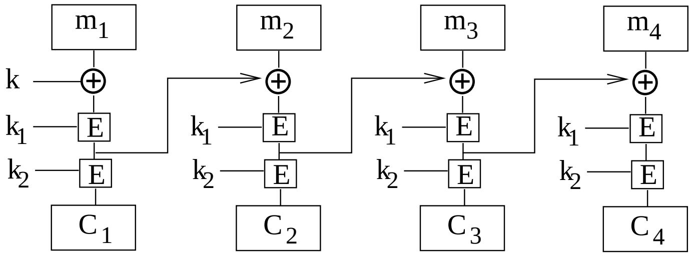
$$DVD \;=\; \underbrace{E_{K_{root}}(K)}_{\text{header}} \;\Big\|\; \underbrace{E_K(M)}_{\text{body}}$$

where K is some random AES key called a content-key. Since all DVD players have the key K_{root} all players can decrypt the movie M. We refer to $E_{K_{root}}(K)$ as the header and $E_K(M)$ as the body. In what follows the DVD header may contain multiple ciphertexts where each ciphertext is the encryption of the content-key K under some key K_i in the binary tree.

a. Suppose the $\log_2 n$ keys embedded in DVD player number r are exposed by hackers and published on the Internet (say in a program like DeCSS). Show that when the movie industry is about to distribute a new DVD movie they can encrypt the contents of the DVD using a header of size $\log_2 n$ so that all DVD players can decrypt the movie except for player number r. In effect, the movie industry disables player number r.
Hint: the header will contain $\log_2 n$ ciphertexts where each ciphertext is the encryption of the content-key K under certain $\log_2 n$ keys from the binary tree.

b. Suppose the keys embedded in k DVD players $R = \{r_1, \ldots, r_k\}$ are exposed by hackers. Show that the movie industry can encrypt the contents of a new DVD using a header of size $O(k \log n)$ so that all players can decrypt the movie except for the players in R. You have just shown that all hacked players can be disabled without affecting other consumers.

Problem 3 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 4 Traitor tracing. Satellite content providers (such as satellite radio) often use hardware players to enforce specific usage policy (e.g. the content cannot be saved after it is played). Player i contains an encryption key K_i that it uses to decrypt and play the broadcast content. Now suppose some user j breaks open his player, recovers key K_j, and builds a pirate player P that decrypts and saves all broadcast content in the clear. When this pirate player P is somehow found, the content provider would like to tell whose key K_j was used to construct P (supposedly, this user j will have to answer some tough questions). Finding the key K_j that was used to build P is called *tracing* and the key K_j is called the *traitor key.*

Let $n = 32$ and suppose there are at most 2^n players in existence. Consider the following encryption system:

Setup: generate $2n$ keys:

$k_{0,0}$	$k_{1,0}$	$k_{2,0}$	$\cdots$	$k_{n-1,0}$
$k_{0,1}$	$k_{1,1}$	$k_{2,1}$	$\cdots$	$k_{n-1,1}$

Player number ℓ (for $\ell = 0, 1, \ldots, 2^n - 1$) is given key K_ℓ defined as follows. Let $b_{n-1}b_{n-2}\ldots b_0 \in \{0,1\}^n$ be the binary representation of ℓ (so that $\ell = \sum_{i=0}^{n-1} b_i 2^i$). Then key K_ℓ is

$$K_\ell = (\ k_{0,b_0},\ k_{1,b_1},\ \ldots,\ k_{n-1,b_{n-1}}\ )$$

Encrypt: to transmit content m, the content provider picks a random $i \in \{0, 1, \ldots, n-1\}$ and broadcasts via satellite the ciphertext:

$$C = (\ i,\ E(k_{i,0}, m),\ E(k_{i,1}, m)\ )$$

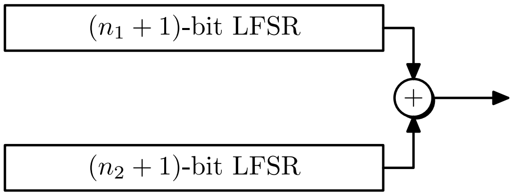
a. Show that all players $\ell = 0, 1, \ldots, 2^n - 1$ can decrypt the broadcast and obtain m.

b. Suppose key K_j is used to create a pirate decoder P. Show that the content provider can use P as a *black-box* and recover the index j. The content owner need not reverse engineer player P — it only uses P as a black box feeding it ciphertexts and observing the result. We are assuming that users do not collude so that P is created using knowledge of a single secret key K_j.
Hint: try to recover one bit of j at a time by feeding P a total of n carefully crafted ciphertexts $C_0, C_1, \ldots, C_{n-1}$.

c. Suppose a pirate is able to obtain two player keys K_i and K_j for some i, j (where $i \oplus j$ is not a power of 2). Show how the pirate can build a player P that will evade detection by your tracing algorithm from part (b). That is, your tracing algorithm will fail to output either i or j.

Side note: traitor tracing is a somewhat evil use of cryptography. Fortunately, traitor tracing systems have a number of applications unrelated to content protection.

Problem 5 Consider the following CSS-like pseudo random generator. Assume the generator is used as a stream cipher to encrypt the contents of a DVD.



The secret key is $n = n_1 + n_2$ bits (recall that $n_1 = 16$ and $n_2 = 24$ for CSS). The top LFSR is initialized with $1\|k_1$ where k_1 is the left most n_1 bits of the key. The bottom LFSR is initialized with $1\|k_2$ where k_2 is the right most n_2 bits of the key. The output of the two LFSR's is Xored and the resulting bit stream is the pseudo random sequence used to encrypt the plaintext. Show that an attacker who is only given the initial $2n$ bits of output of this generator can produce the rest of the output sequence in time approximately $2^{\min(n_1,n_2)}$.
Hint: Do an exhaustive search on all possible states of one LFSR and try to deduce the state of the other LFSR.

An exhaustive search attack takes time 2^n to produce the entire output sequence. Your attack is much faster. For CSS, your attack takes approximately 2^{16} steps which is only a few milliseconds on a modern processor. Consequently, the resulting stream cipher is completely insecure once a few bits of a plaintext/ciphertext pair are known.

Problem 6 Given a cipher (E, D), define the randomized cipher (F, G) by

$$F(k, m) := (E(k, r),\ r \oplus m),$$

where r is a random bit string of the same length 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. The decryption algorithm $G(k,\ (c_1, c_2))$ works by first running $D(k, c_1)$ to recover r and then recovering m from c_2.

We consider two attack models. The goal of both models is to reconstruct the actual secret key k (this is a very strong goal – one might be able to decrypt messages without ever learning k).

- In the key-reconstruction chosen plaintext attack (KR-CPA), the adversary is allowed to issue q message queries $m_1, m_2, \ldots, m_q$ and for each m_i learn the encryption of m_i.
- In the key-reconstruction random plaintext attack (KR-RPA), the adversary is given q random plaintext/ciphertext pairs.

Note that for $F(k, \cdot)$ 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 the cipher (E, D) is secure against KR-RPA attacks then the cipher (F, G) is secure against $\mathsf{KR} - \mathsf{CPA}$ attacks.

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