CS255: Cryptography and Computer Security **Winter 2011**

Assignment #1

Due: Monday, Jan. 31, 2011, in class.

Problem 0 Data compression is often used in data storage or transmission. Suppose you want to use data compression in conjunction with encryption. Does it make more sense to

A. Compress the data and then encrypt the result, or

B. Encrypt the data and then compress the result.

Justify your answer.

Problem 1 Simple 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 Let m be a message consisting of ℓ AES blocks (say $\ell = 100$). Alice encrypts m using CBC mode and transmits the resulting ciphertext c to Bob. Due to a hardware error, ciphertext block number $\ell/2$ is corrupted during transmission. All other ciphertext blocks are transmitted and received correctly.

a. Once Bob decrypts the received ciphertext, how many plaintext blocks will be corrupted?

b. Answer the same question for randomized counter mode.

Problem 3 Let us see why in CBC mode an unpredictable IV is necessary for CPA security.

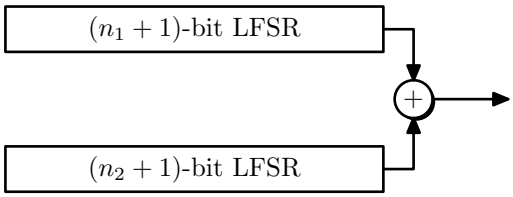
a. Suppose a defective implementation of CBC encrypts a sequence of packets by always using the last ciphertext block of packet number i as the IV for packet number $i + 1$ (up until a few years ago all web browsers implemented CBC this way). Construct an efficient adversary that wins the CPA game against this implementation with advantage close to 1. Recall that in the CPA game the attacker submits packets (a.k.a messages) to the challenger one by one and receives the encryption of those packets. The attacker then submits the semantic security challenge which the challenger treats as the next packet in the packet stream.

b. Can you suggest a simple fix to the problem from part (a) that does not add any additional bits to the ciphertext?

c. Suppose the block cipher (E, D) used for CBC encryption has a block size of n bits. Construct an attacker that wins the CPA game against CBC with a random IV (i.e. where the IV for each message is chosen independently at random) with advantage close to $1/2^n$.

Your answer for part (c) explains why CBC cannot be used with a block cipher that has a small block size (e.g. $n = 32$ bits). Note that there are many other problems with such a small block size, which is why AES has a block size of 128 bits.

Problem 4 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. Note that the taps in both LFSRs are publicly known to everyone. 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 $2^{\min(n_1,n_2)}$. An exhaustive search attack takes time 2^n to produce the entire output sequence. Your attack is much faster.
Hint: As a first step, your attack can do an exhaustive search on all possible states of one of the LFSRs.

Problem 5 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.

Side note: the AACS system used to encrypt Blu-ray and HD-DVD disks uses a related system. It was quickly discovered that bored hackers can expose player secret keys faster than the MPAA can revoke them.

Problem 6 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

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

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

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

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.

Problem 7 Psuedo-randomness. Recall that a Pseudo Random Generator (PRG) $G : X \to Y$ is a function that given a random seed in X outputs an element in Y that looks random to any "efficient" adversary. More precisely, let $\mathcal{A}$ be an efficient algorithm that takes inputs in Y and outputs 0 or 1. For $x \xleftarrow{R} X$ and $y \xleftarrow{R} Y$ define

$$p_0 := \Pr[\mathcal{A}(G(x)) = 1] \quad \text{and} \quad p_1 := \Pr[\mathcal{A}(y) = 1]\ .$$

The PRG is secure if for all efficient algorithms $\mathcal{A}$ the quantity $|p_0 - p_1|$ is negligible.

a. In class we defined what it means for a PRG to be unpredictable. Show that if G is secure then G is unpredictable.
Hint: it is best to prove the contra-positive; if there is an efficient algorithm $\mathcal{B}$ that predicts G then there is an efficient algorithm $\mathcal{A}$ that distinguishes the output of G from random.

b. Let $G : X \to X^\ell$ be a secure PRG. We treat the output of G as a vector of ℓ coordinates. Let $[\ell]$ be the set $\{0, \ldots, \ell - 1\}$ and define the PRF F over $(X, [\ell], X)$ as follows:

$$F(k, i) := G(k)[i] \quad \in X$$

Prove that F is a secure PRF. Again, you should prove the contra-positive.

c. Let F be a secure PRF defined over (K, X, Y) and suppose that $K \subseteq X$. Define the following two PRFs F_0 and F_1:

$$F_0(k, x) := \begin{cases} F(k, x) & \text{if } x \neq 0, \text{ and} \\ 0 & \text{otherwise} \end{cases} \qquad F_1(k, x) := \begin{cases} F(k, x) & \text{if } x \neq k, \text{ and} \\ 0 & \text{otherwise} \end{cases}$$

Are any of these secure PRFs? Justify your answer.