CS255: Intro. to Cryptography **Winter 2025**

Assignment #2

Due: 11:59pm on Thu., Jan. 30, 2025, on Gradescope (each answer on a separate page)

Problem 1. (*The trouble with compression*) Let (E, D) be a semantically secure cipher that operates on messages in $\{0,1\}^{\leq n}$ (i.e. messages whose length is at most n bits). Suppose that the ciphertext output by the encryption algorithm is exactly 128 bits longer than the input plaintext. To reduce ciphertext size, there is a strong desire to combine encryption with lossless compression. We can think of compression as a function from $\{0,1\}^{\leq n}$ to $\{0,1\}^{\leq n}$ where, for some messages, the output is shorter than the input. As always, the compression algorithm is publicly known to everyone.

a. Compress-then-encrypt: Suppose the encryptor compresses the plaintext message m before passing it to the encryption algorithm E. Some n-bit messages compress well, while other messages do not compress at all. Show that the resulting system is not semantically secure by exhibiting a semantic security adversary that obtains advantage close to 1.

Answer: The adversary can pick up two messages m_0, m_1, two messages will be of different length after the compression, lets say l_0, l_1, then the ciphertext length will be $l_0 + 128$, $l_1 + 128$ respectively. The adversary will then output 0, if the ciphertext length is $l_0 + 128$, otherwise 1. (questions, is 1 close to 1)

b. Encrypt-then-compress: Suppose that instead, the encryptor applies compression to the output of algorithm E (here you may assume the compression algorithm takes messages of length up to $n + 128$ bits as input). Explain why this proposal is of no use for reducing ciphertext size.

Answer: Compression can only work by exploiting patterns or redundancy in the data, but the output of semantically secure cipher's output is indistinguishable from random string, meaning there is no pattern to compress. Using information theory, the output of semantically secure cipher already has the maximum entropy, so there is no room to compress further.

Problem 2. (*Broadcast encryption*) The movie industry wants to protect digital content distributed on DVDs. Here is 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 $\log_2 n$. Every node v_j in this binary tree contains an AES key $k_j \in \mathcal{K}$. These keys are kept secret from consumers and are fixed for all time. At manufacturing time every DVD player is assigned a serial number $i \in \{0, \ldots, n-1\}$. Let S_i be the set of $1 + \log_2 n$ nodes along the path from the root of the binary tree to leaf number i. The manufacturer embeds in player number i the $1 + \log_2 n$ keys associated with the nodes in S_i. In this way each DVD player ships with $1 + \log_2 n$ keys embedded in it, and these keys are supposedly inaccessible to the end user. A DVD movie m is encrypted as

$$DVD := \underbrace{E(k_{\text{root}}, k)}_{\text{header}} \Big\| \underbrace{E(k, m)}_{\text{body}},$$

where $k \overset{\text{R}}{\leftarrow} \mathcal{K}$ is a fresh random key called a content key. Since all DVD players have the key k_{root}, all players can decrypt the content m. We refer to $E(k_{\text{root}}, k)$ 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 $1 + \log_2 n$ keys embedded in DVD player number r are exposed by hackers and published on the Internet. Show that when the movie industry is about to distribute a new movie m, they can encrypt m using a header containing $\log_2 n$ short ciphertexts, 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.

Answer: To disable player r while allowing all others to decrypt, we can:
1. Generate new content key k for encrypting movie m
2. For each level l of the binary tree (except root): find the node v at level l whose sibling is on path to player r. Add $E(k_v, k)$ to the header, where k_v is key for node v
3. Structure the DVD as: header = $E(k_{v1}, k)||E(k_{v2}, k)||...||E(k_{vlogn}, k)$ and body = $E(k, m)$
Then every player except r shares at least one node v_i with the path to root, meaning all other players can decrypt by finding their shared node

b. Next, suppose the keys embedded in s DVD players $R = \{r_1, \ldots, r_s\}$ are exposed by hackers, where $s > 1$. Show that the movie industry can encrypt the contents of a new DVD using a header containing $O(s \log n)$ short ciphertexts 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.

Answer: For multiple compromised players $R = r_1, ..., r_s$, we can (almost the same as the previous question):
1. Generate new content key k for encrypting movie m
2. For each level l of the binary tree (except root): find the node v at level l that is not on path of r_i and its sibling is on path to player r_i. Add $E(k_v, k)$ to the header, where k_v is key for node v
3. Structure the DVD as: header $= E(k_{v1}, k)||E(k_{v2}, k)||...||E(k_{vslogn}, k)$ abody $= E(k, m)$
This works because:
1. Each compromised player contributes up to log n nodes, so total $ciphertexts \leq s * logn = O(slogn)$
2. Each uncompromised player shares at least one node with path to root
3. No compromised player can access any key used to encrypt k

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

Problem 3. The purpose of this problem is to exercise the concept of *advantage*. Consider the following two experiments EXP(0) and EXP(1) between a challenger and an adversary $\mathcal{A}$:

- In EXP(0) the challenger choose a uniform random number x in the set $\{1, 2, \ldots, 6\}$, and sends x to the adversary $\mathcal{A}$.
- In EXP(1) the challenger choose a uniform random number y in the set $\{1, 2, \ldots, 10\}$, and sends y to the adversary $\mathcal{A}$.

The adversary's goal is to distinguish these two experiments: at the end of each experiment the adversary $\mathcal{A}$ outputs a bit 0 or 1 for its guess for which experiment it is in. For $b = 0, 1$ let W_b be the event that in experiment b the adversary outputs 1. The adversary tries to maximize its distinguishing advantage, namely the quantity

$$\text{Adv}[\mathcal{A}] := \Big| \Pr[W_0] - \Pr[W_1] \Big| \quad \in [0, 1] \ .$$

The advantage Adv captures the adversary's ability to distinguish the two distributions. If the advantage is 0 then the adversary behaves exactly the same in both experiments and therefore does not distinguish between them. If the advantage is 1 then the adversary can tell perfectly what experiment it is in. If the advantage is negligible for all efficient adversaries (as defined in class) then we say that the two experiments are indistinguishable.

a. Consider the following adversary $\mathcal{A}_0$ that takes as input an integer z and outputs $\{0, 1\}$:

```
if $z \in \{1, \ldots, 4\}$: output 0
else: output 1
```

What is the advantage $\text{Adv}[\mathcal{A}_0]$ of this adversary in distinguishing the two experiments?

Answer: YOUR ANSWER: $\text{Adv}[\mathcal{A}_0] = \frac{4}{15}$.

b. Describe an adversary $\mathcal{A}_1$ that achieves the highest advantage you can think of.

Answer: To fully take advantage of {1,. . . , 6} has higher probability of coming from EXP(0), and {7, 8, 9, 10} is exclusively from EXP(1). $\mathcal{A}_1$ will be

```
if $z \in \{1, \ldots, 6\}$: output 0
else: output 1
```

Problem 4. (*PRFs*) Let F be a secure PRF defined over $(\mathcal{K}, \mathcal{X}, \mathcal{Y})$, where $\mathcal{K} = \mathcal{X} = \mathcal{Y} = \{0,1\}^n$.

a. Show that $F_1\big((k_1, k_2),\ (x_1, x_2)\big) := F(k_1, x_1) \oplus F(k_2, x_2)$ is not a secure PRF. That is, construct a PRF adversary $\mathcal{A}_1$ that has non-negligible advantage in distinguishing $F_1(k, \cdot)$ from a random function in $\text{Funs}[\mathcal{X}^2, \mathcal{Y}]$. Hint: your attacker will query F_1 at four points.

Answer:
Query (x,x), (x,y), (y,x), (y,y) for some distinct x, y
For F_1, it is always 0 for the following expression $F_1((k_1,k_2),(x,x)) \oplus F_1((k_1,k_2),(x,y)) \oplus F_1((k_1,k_2),(y,x)) \oplus F_1((k_1,k_2),(y,y)) = 0$
For a random function: the above expression has negligible probability to be 0
$\mathcal{A}_1$ constructed as follows: query (x,x), (x,y), (y,x), (y,y), then XOR four output of F_1. If it is 0, output 0, otherwise output 1.

b. Show that $F_2(k, x) := F(k, x) \parallel F(k, F(k, x))$ is not a secure PRF. Here $\parallel$ means the concatenation of the two outputs. Hint: your adversary $\mathcal{A}_2$ will query F_2 at two inputs, where the second query depends on the answer to the first query.

Answer: Essentially F_2 appends the encryption of encryption. We construct $\mathcal{A}_2$ as follows:

query message m, $F_2(k, m) := F(k, m) \parallel F(k, F(k, m)) := a||b$
query message $F_2(k, m)$, $F_2(k, F_2(k, m)) := F(k, F(k, m)) \parallel F(k, F(k, F(k, m))) := c||d$
output 0 if $b == c$, otherwise 1

For F_2, the second half of first output must equal first half of second output. For a random function, this equality would only happen with negligible probability

c. Prove that $F_3(k, x) := F(k, x) \oplus x$ is a secure PRF. Do so by proving the contrapositive: show that if an adversary $\mathcal{A}_3$ can distinguish $F_3(k, \cdot)$ from a random function then there is adversary $\mathcal{B}$ (that is a wrapper around $\mathcal{A}_3$) that can distinguish F from a random function. This $\mathcal{B}$ will play the role of PRF challenger to $\mathcal{A}_3$, and attack F.

Answer:
Construct $\mathcal{B}$ as follows

$\mathcal{B}$ receives oracle O, that either outputs $F(k, \bullet)$ or random output
$\mathcal{B}$ simulates $F_3(k, x) := O(x) \oplus x$ and then query $\mathcal{A}_3$
$\mathcal{B}$ outputs whatever from $\mathcal{A}_3$

By this construction, when oracle is a truly random function or PRF, $\mathcal{B}$ perfectly simulate the challenger that $\mathcal{A}_3$ faces, meanig $\mathcal{B}$ has the same advantage of distinguishing $F(k, \bullet)$ as $\mathcal{A}_3$ distinguishes F_3, which is non negligible. This contradicts the assumption that F is a secure PRF.

Problem 5. (*Broken Even-Mansour ciphers*) Let $\pi : \mathcal{X} \to \mathcal{X}$ be a fixed public one-to-one function, where $\mathcal{X} := \{0,1\}^n$, and where π and π^{-1} are efficiently computable. The Even-Mansour pseudorandom permutation (E, D) derived from π, is defined as

$$E\big((k_0, k_1),\ x\big) := \pi(x \oplus k_0) \oplus k_1 \quad \text{and} \quad D\big((k_0, k_1),\ c\big) := \pi^{-1}(c \oplus k_1) \oplus k_0.$$

This permutation corresponds to one round of AES, and can be shown to be a secure PRP when π is chosen at random from Perms$[\mathcal{X}]$, and when $|\mathcal{X}|$ is sufficiently large. Let's look at some broken variants of Even-Mansour.

a. Show that $E_1(k_1, x) := \pi(x) \oplus k_1$ is not a secure PRP.

Answer:
Given $\pi : \mathcal{X} \to \mathcal{X}$ be a fixed public one-to-one function, for any x, we can recover the key k_1 by $k_1 = E_1(k_1, x) \oplus \pi(x)$

b. Show that $E_2(k_0, x) := \pi(x \oplus k_0)$ is not a secure PRP.

Answer: Given $\pi : \mathcal{X} \to \mathcal{X}$ be a fixed public one-to-one function, for any x, we can recover the key k_0 by $k_0 = \pi^{-1}(E_2(k_0, x)) \oplus x$

Problem 6. (*Exercising the definition of semantic security*) Let (E, D) be a semantically secure cipher defined over $(\mathcal{K}, \mathcal{M}, \mathcal{C})$, where $\mathcal{M} = \mathcal{C} = \{0,1\}^L$. Which of the following encryption algorithms yields a semantically secure scheme? Either give an attack or provide a security proof. To prove security, prove the contrapositive, that is prove that a semantic security attacker $\mathcal{A}$ on the proposed system gives a semantic security attacker $\mathcal{B}$ on (E, D), with the same advantage.

a. $E_1(k, m) := 0 \parallel E(k, m)$

Answer: E_1 is semantically secure. Given an adversary $\mathcal{A}$ against E_1 with non-negligible advantage ϵ, we construct an adversary $\mathcal{B}$ against E:
1. $\mathcal{B}$ receives challenge ciphertext c (encryption of either m_0 or m_1)
2. $\mathcal{B}$ simulates E_1 by computing $c' = 0 \parallel c$
3. $\mathcal{B}$ acts as challenger and provides c' to $\mathcal{A}$
4. Since $\mathcal{A}$ can distinguish c' with non-negligible advantage ϵ
5. $\mathcal{B}$ outputs what $\mathcal{A}$ outputs, achieving the same advantage ϵ

This contradicts the assumption that E is semantically secure.

b. $E_2(k, m) := E(k, m) \parallel \text{parity}(m)$

Answer: E_2 is not semantically secure. Here's an attack:
1. Choose $m_0 = 0^L$ and $m_1 = 1 \parallel 0^{L-1}$
2. Note that $\text{parity}(m_0) = 0$ and $\text{parity}(m_1) = 1$
3. Given challenge ciphertext $c \parallel b$, output 0 if $b = 0$, 1 if $b = 1$

The adversary always wins (aka it has advantage of 1), breaking semantic security.

c. $E_3(k, m) := \text{reverse}(E(k, m))$

Answer: E_3 is semantically secure. Given an adversary $\mathcal{A}$ against E_3 with non-negligible advantage ϵ, we construct an adversary $\mathcal{B}$ against E:
1. $\mathcal{B}$ receives challenge ciphertext c (encryption of either m_0 or m_1)
2. $\mathcal{B}$ simulates E_3 by computing $c' = \text{reverse}(c)$
3. $\mathcal{B}$ acts as challenger and provides c' to $\mathcal{A}$
4. Since $\mathcal{A}$ can distinguish c' with non-negligible advantage ϵ
5. $\mathcal{B}$ outputs what $\mathcal{A}$ outputs, achieving the same advantage ϵ

This contradicts the assumption that E is semantically secure.

d. $E_4(k, m) := E(k, \text{reverse}(m))$

Answer: $E_4(k, m) := E(k, \text{reverse}(m))$ is semantically secure. The reversing operation is just a bijective mapping of the message space to itself. Since E is semantically secure for all messages in M, and reverse(m) is still in M, the composed scheme remains semantically secure.

Here, for a bit string s, parity(s) is 1 if the number of 1's in s is odd, and 0 otherwise; also, reverse(s) is the string obtained by reversing the order of the bits in s, e.g., reverse$(1011) = 1101$.

Problem 7. (*Why double-AES128 is a bad idea*) Let $\mathcal{E} := (E, D)$ be a cipher defined over $(\mathcal{K}, \mathcal{M}, \mathcal{C})$ where $\mathcal{C} \subseteq \mathcal{M}$. One can define a cipher with double the key length, called $2\mathcal{E}$, defined over $(\mathcal{K}^2, \mathcal{M}, \mathcal{C})$ as follows:

$$2E\big((k_1, k_2), m\big) := E\big(k_1, E(k_2, m)\big).$$

That is, we apply the encryption algorithm E twice with independent keys k_1 and k_2.

a. Write out the decryption algorithm: $2D\big((k_1, k_2), c\big) := \dots$

Answer: $2D\big((k_1, k_2), c\big) := D(k_1, D(k2, c))$

Recall that the AES block cipher can take either 128, 192, or 256 bit keys, denoted AES128, AES192, and AES256, respectively. You are probably wondering why is there a need for AES256. We can simply define AES256 to be 2AES128, namely define

$$\text{BadAES256}\big((k_1, k_2), m\big) := \text{2AES128}\big((k_1, k_2), m\big) = \text{AES128}\big(k_1, \text{AES128}(k_2, m)\big).$$

The key for this BadAES256 is (k_1, k_2) which is 256 bits, as required. So why is AES256 a separate algorithm? Why can't we simply use 2AES128?

b. Your goal is to show that the $2\mathcal{E}$ cipher is no more secure than the underlying $\mathcal{E}$ cipher. This means that $2AES128$ is no more secure than AES128, which is not what we want for AES256.

For a multi-block message $M = (m_1, m_2, \dots, m_n)$ write $E(k, M) := \big(E(k, m_1), \dots, E(k, m_n)\big)$.

You are given a pair (M, C) where $C = 2E\big((k_1, k_2), M\big)$. You may assume that there is a *unique* $(k_1, k_2) \in \mathcal{K}^2$ that satisfies $C = 2E\big((k_1, k_2), M\big)$. Your goal is to find this (k_1, k_2).

Given (M, C) as input, an exhaustive search algorithm over all possible (k_1, k_2) will take time $O(|\mathcal{K}|^2)$. Your goal is to design an algorithm that finds (k_1, k_2) in time $O(|\mathcal{K}|)$. This is the time to break $\mathcal{E}$ by exhaustive search which means that $2\mathcal{E}$ is no more secure than $\mathcal{E}$.

Hint: First, observe that if $C = 2E\big((k_1, k_2), M\big)$ then

$$E(k_2, M) = D(k_1, C). \tag{1}$$

Try building a table T of pairs $\big(k,\ E(k, M)\big)$ for all $k \in \mathcal{K}$. Constructing this table takes $|\mathcal{K}|$ evaluations of $E(k, M)$, and the table will contain all possible values of the left-hand side of (1). Now that T is built, show that you can find (k_1, k_2) in time $O(|\mathcal{K}|)$. To do so, use the right-hand side of (1). You can assume that testing if T contains a pair $(*, c)$ can be done in constant time using an appropriate data structure for T.

Answer: Algorithm to find (k_1, k_2) in $O(|\mathcal{K}|)$ time:
1. Build table T: for each $k_2 \in \mathcal{K}$, we compute $v = E(k_2, M)$ then store (k_2, v) in table T. It costs $O(|\mathcal{K}|)$ evaluations of E
2. Find matching keys: for each $k_1 \in \mathcal{K}$, we compute $w = D(k_1, C)$ then look up w in table T (constant time). If we found with key k_2, return (k_1, k_2). This costs $O(|\mathcal{K}|)$ evaluations of D
So the total cost will be $O(|\mathcal{K}|)$ operations
Correctness: By equation (1), the correct (k_1, k_2) must satisfy $E(k_2, M) = D(k_1, C)$. Step 1 computes all possible left-hand sides, Step 2 finds the matching right-hand side. Since we assumed a unique solution exists, the algorithm will find it.

Discussion: The algorithm from part (b) is called a *meet in the middle attack* because you are attacking the mid-point of algorithm $2\mathcal{E}$. Meet in the middle attacks come up often in cryptography.