1 Last Lecture

In the last lecture, we saw the first formal definition for encryption schemes. An encryption scheme $\mathcal{E} = (\mathsf{KeyGen}, \mathsf{Enc}, \mathsf{Dec})$ with key space $\mathcal{K}$ and message space $\mathcal{M}$ satisfies no-query-semantic-security if, for any probabilistic polynomial time adversary $\mathcal{A}$, there exists a negligible function $\mu(\cdot)$ such that for all n,

$$\Pr\left[\mathcal{A} \text{ wins the } \mathsf{No\text{-}Query\text{-}Semantic\text{-}Security} \text{ game against } \mathcal{E}\right] \leq 1/2 + \mu(n)$$

where the $\mathsf{No\text{-}Query\text{-}Semantic\text{-}Security}$ game is defined in Figure **??**.

No-Query-Semantic-Security
1. Adversary sends two messages m_0, m_1 to the challenger.
2. The challenger chooses a bit $b \leftarrow \{0,1\}$, key $k \leftarrow \mathcal{K}$ and sends $\mathsf{Enc}(m_b, k)$ to the adversary.
3. The adversary sends its guess b', and wins the security game if $b = b'$.

Figure 1: The No-Query Semantic Security Game

We also saw our first security proof: we defined an encryption scheme $\mathcal{E}'$ using a secure scheme $\mathcal{E}$, and proved that if there exists an attack against $\mathcal{E}'$, then there exists an attack against $\mathcal{E}$.

1.1 Questions/clarifications from last class

1. **Why do we use negligible functions? What happens if an adversary is allowed to win with probability $1/2 + 1/n$?**

   There are at least a couple of issues why replacing $\mathsf{negl}(n)$ with $1/n$ could be a bad idea.

   - First, in *certain* cases, it is possible to boost the success probability of the adversary from $1/2+1/n$ to ≈ 1. For instance, suppose there exists an adversary that, for every m_0, m_1 and every key k, can win with probability $1/2 + 1/n$. Then, it is possible to construct an adversary that wins with prob. close to 1 (by computing the guess multiple times, and taking majority).
   - Second, suppose you have an encryption scheme where $1/n$ keys are 'bad', and the rest are good. For such an encryption scheme, any poly. time adversary would have winning probability bounded by $\approx 1/2 + 1/n$ (since the adversary would have no guessing advantage if the keys are good). But such an encryption scheme would be very bad in practice — roughly $1/n$ fraction of the people using this encryption scheme would have bad secret keys.

2. **Some more explanation on reductions and challengers**

   Suppose you wish to show that a problem P is $\mathsf{NP\text{-}complete}$. We can do this by showing a reduction from $\mathsf{3\text{-}SAT}$ to P. $\mathsf{NP\text{-}completeness}$ reductions can also be viewed through the lens of challenger-adversary games. Suppose there exists an adversary that can solve all instances of problem P in polynomial time. We can use this adversary to solve all instances of $\mathsf{3\text{-}SAT}$ in polynomial time. The reduction algorithm interacts with the $\mathsf{3\text{-}SAT}$ challenger, and receives an instance. It transforms this $\mathsf{3\text{-}SAT}$ instance into an instance for P, which it then forwards to the adversry $\mathcal{A}$. The adversary $\mathcal{A}$ sends its output, using which the reduction algorithm solves the $\mathsf{3\text{-}SAT}$ instance.

2 Question 04.02 from last lecture

Recall the exercise (Question 04.02) from last lecture.

Intuitively, this scheme looks secure — if $\mathcal{E}$ is secure, then each of the ciphertext components, individually, hides m. But how do we argue this formally? If there exists an adversary $\mathcal{A}$ to break the security of $\mathcal{E}'$, how do we use this adversary to break the security of $\mathcal{E}$?

Question 04.02. Let $\mathcal{E} = (\mathsf{KeyGen}, \mathsf{Enc}, \mathsf{Dec})$ be an encryption scheme that satisfies Definition 04.02. Consider the following modified encryption scheme $(\mathsf{KeyGen}', \mathsf{Enc}', \mathsf{Dec}')$:

- KeyGen' = Run KeyGen twice, and let k_1, k_2 be the two output keys. Output $k = (k_1, k_2)$ as the key.
- $\mathsf{Enc}'(m, k = (k_1, k_2)) = (\mathsf{Enc}(m, k_1), \mathsf{Enc}(m, k_2))$
- $\mathsf{Dec}'(\mathsf{ct} = (\mathsf{ct}_1, \mathsf{ct}_2), k = (k_1, k_2)) = \mathsf{Dec}(\mathsf{ct}_1, k_1)$

Show that $\mathcal{E}'$ satisfies Definition 04.02, assuming $\mathcal{E}$ does.

2.1 A few attempts from class (and why they don't work)

Similar to the proof from Lecture 04, we want to show a reduction: suppose there exists an adversary $\mathcal{A}$ that breaks $\mathcal{E}'$. We want to construct an algorithm $\mathcal{B}$ that, interacts with the challenger for $\mathcal{E}$, and uses $\mathcal{A}$ to break security of $\mathcal{E}$. Here are a few attempts discussed in class:

- *The reduction algorithm receives m_0, m_1 from $\mathcal{A}$, which it forwards to the $\mathcal{E}$-challenger. The challenger sends a ciphertext* ct. *The reduction algorithm sends* $(\mathsf{ct}, \mathsf{ct})$ *to $\mathcal{A}$.*

  This approach does not work because, from the point of view of $\mathcal{A}$, it is expecting two encryptions of the same message (either m_0 or m_1) using *independent keys*. Therefore, encryption of a message using two randomly chosen keys is very unlikely to result in $(\mathsf{ct}, \mathsf{ct})$.

- *The reduction algorithm receives m_0, m_1 from $\mathcal{A}$, which it forwards to the $\mathcal{E}$-challenger. The challenger sends a ciphertext* ct. *The reduction algorithm then chooses a uniformly random bit $\tilde{b}$, a uniformly random key $\tilde{k}$, computes $\widetilde{\mathsf{ct}} = \mathsf{Enc}(m_{\tilde{b}}, \tilde{k})$ and sends $(\mathsf{ct}, \widetilde{\mathsf{ct}})$ to $\mathcal{A}$.*

  Again, notice that the ciphertext $(\mathsf{ct}, \tilde{\mathsf{ct}})$ is very different from what the adversary expects. It expects either $(\mathsf{Enc}(m_0, k_1), \mathsf{Enc}(m_0, k_2))$ or $(\mathsf{Enc}(m_1, k_1), \mathsf{Enc}(m_1, k_2))$. However, if $\tilde{b}$ is not the same as the challenger's chosen bit, then one of the ciphertext components is encryption of m_0, while the other is encryption of m_1.

The core issue: in order to give the $\mathcal{A}$ a legitimate ciphertext, the reduction algorithm seems to require the knowledge of bit chosen by the challenger.

Before we prove that $\mathcal{E}'$ is secure, we will present a simple alternate formulation that is often helpful for simplifying our proofs. While this alternate formulation is stated in terms of the $\mathsf{No\text{-}Query\text{-}Semantic\text{-}Security}$ security game, it is applicable in any security game where the adversary must finally guess the bit chosen by challenger.

2.2 An alternate formulation for adversary's success in security games

Recall, from the adversary's view, it first sends two messages (m_0, m_1), then receives a challenge ciphertext, and must output a bit at the end. There are two 'worlds' - one where the adversary receives an encryption of m_0 (using a random key), and another where the adversary receives an encryption of m_1 (using a random key). Can the adversary figure out which world it is in?

If the encryption scheme is broken, then the adversary's behaviour is certainly different in the two worlds. If the adversary can break the security of our encryption scheme, then it will output 0 in the first world, and 1 in the second world.

It is not difficult to show that if the encryption scheme is secure (as per Definition 04.02), then the adversary cannot distinguish which world it is in. We can prove this by simply expanding the probability of adversary's win. Suppose there exists an adversary that wins the $\mathsf{No\text{-}Query\text{-}Semantic\text{-}Security}$ security game

with prob. significantly greater than 1/2, say 3/4. Then

$$\begin{aligned}
3/4 &= \Pr\Big[\mathcal{A} \text{ wins the } \mathsf{No\text{-}Query\text{-}Semantic\text{-}Security} \text{ game}\Big] \\
&= \Pr\Big[\text{Challenger chooses } b=0 \ \wedge\ \mathcal{A} \text{ guesses } b'=0\Big] \\
&\quad + \Pr\Big[\text{Challenger chooses } b=1 \ \wedge\ \mathcal{A} \text{ guesses } b'=1\Big] \\
&= \Pr\Big[\text{Challenger chooses } b=0 \ \wedge\ \mathcal{A} \text{ guesses } b'=0\Big] \\
&\quad + \Big(\Pr\Big[\text{Challenger chooses } b=1\Big] - \Pr\Big[\text{Challenger chooses } b=1 \ \wedge\ \mathcal{A} \text{ guesses } b'=0\Big]\Big) \\
&= \Pr\Big[\text{Challenger chooses } b=0 \ \wedge\ \mathcal{A} \text{ guesses } b'=0\Big] \\
&\quad + \Big(\frac{1}{2} - \Pr\Big[\text{Challenger chooses } b=1 \ \wedge\ \mathcal{A} \text{ guesses } b'=0\Big]\Big)
\end{aligned}$$

From the last line, it follows that if $\mathcal{A}$ breaks the security of an encryption scheme, then the following probabilities are 'far apart'.

$$p_0 = \Pr\Big[\text{Challenger chooses } b=0 \ \wedge\ \mathcal{A} \text{ guesses } b'=0\Big] \tag{1}$$

$$p_1 = \Pr\Big[\text{Challenger chooses } b=1 \ \wedge\ \mathcal{A} \text{ guesses } b'=0\Big] \tag{2}$$

This leads to an alternate formulation of security definitions. We say that an encryption scheme satisfies $\mathsf{No\text{-}Query\text{-}Semantic\text{-}Security}$ if, for any adversary $\mathcal{A}$, the following probabilities are negligibly close:

$$p_0 = \Pr\Big[\text{Challenger chooses } b=0 \ \wedge\ \mathcal{A} \text{ guesses } b'=0\Big]$$

$$p_1 = \Pr\Big[\text{Challenger chooses } b=1 \ \wedge\ \mathcal{A} \text{ guesses } b'=0\Big]$$

Intuitively, this makes sense: if an encryption scheme is secure, then the adversary's behaviour should not change depending on whether it gets an encryption of m_0 or an encryption of m_1. We formalize our observation below (the proof is left as a simple exercise).

Claim 05.01. An encryption scheme $\mathcal{E}$ satisfies $\mathsf{No\text{-}Query\text{-}Semantic\text{-}Security}$ (Def. 04.02) if and only if the following holds: for any prob. poly. time adversary $\mathcal{A}$, there exists a negl. function $\mu(\cdot)$ s.t. for all n,

$$\Big|\Pr\Big[\text{Chall. chooses } b=0 \ \wedge\ \mathcal{A} \text{ outputs } 0\Big] - \Pr\Big[\text{Chall. chooses } b=1 \ \wedge\ \mathcal{A} \text{ outputs } 0\Big]\Big| \le \mu(n)$$

2.3 Back to Question 04.02

As before, we will proceed by considering the contrapositive, but now, we will use Claim **??**. Suppose there exists an efficient adversary that breaks the security of $\mathcal{E}'$. Then, for this adversary, the probability of outputting 0 are significantly different in the following two 'worlds':

<u>World 0:</u>

- The adversary sends two messages m_0, m_1.
- Chall. chooses key $k = (k_1, k_2) \leftarrow \mathsf{KeyGen}'(1^n)$, sends $\mathsf{Enc}'(m_0, k) = \Big(\mathsf{Enc}(m_0, k_1),\ \mathsf{Enc}(m_0, k_2)\ \Big)$.
- The adversary finally outputs a bit b.

World 1:

- The adversary sends two messages m_0, m_1.
- Chall. chooses key $k = (k_1, k_2) \leftarrow \mathsf{KeyGen}'(1^n)$, sends $\mathsf{Enc}'(m_1, k) = \Big(\mathsf{Enc}(m_1, k_1)\ ,\ \mathsf{Enc}(m_1, k_2)\ \Big)$.
- The adversary finally outputs a bit b.

Since we assumed that $\mathcal{A}$ breaks $\mathcal{E}'$, its behaviour must be different in the two worlds. Let us assume, for simplicity, that $\mathcal{A}$ outputs 0 w.p. 3/4 in World 0, and w.p. 1/4 in World 1. Here is the crucial idea, called the **hybrid technique**. Consider the following 'hybrid world':

Hybrid World:

- The adversary sends two messages m_0, m_1.
- Chall. chooses key $k = (k_1, k_2) \leftarrow \mathsf{KeyGen}'(1^n)$, sends $\Big(\mathsf{Enc}(m_0, k_1),\ \mathsf{Enc}(m_1, k_2)\ \Big)$.
- The adversary finally outputs a bit b.

What can we say about $\mathcal{A}$'s probability of outputting 0 in the hybrid world?
Can we say that this probability is somewhere between 3/4 and 1/4? No, we cannot conclude that the probability is in $[1/4, 3/4]$. *But what we can conclude is that the probability cannot be close to both* 3/4 *and* 1/4*!*

We can use this simple observation to conclude our security proof. Here is the template for the proof, you must fill in the details.

1. Suppose $\mathcal{A}$ breaks the security of $\mathcal{E}'$. Then the probability of $\mathcal{A}$ outputting 0 in World 0 must be significantly different from the probability of $\mathcal{A}$ outputting 0 in World 1.
2. This implies that one of the following must hold true:
   - The probability of $\mathcal{A}$ outputting 0 in World 0 is significantly different from the probability of $\mathcal{A}$ outputting 0 in the Hybrid World.
   - The probability of $\mathcal{A}$ outputting 0 in Hybrid World is significantly different from the probability of $\mathcal{A}$ outputting 0 in World 1.

In either of these two cases, we can construct a reduction algorithm that uses $\mathcal{A}$ and breaks the security of $\mathcal{E}$.

3 Our first computationally-secure encryption scheme

We will now present the first secure encryption scheme (with $|\mathcal{K}| \ll |\mathcal{M}|$). The idea for this construction was already proposed by one of the students in Lecture 03. We will instantiate the idea using an appropriate building block called *pseudorandom generators*.

Let us first recall the idea. Take $\mathcal{K} = \{0,1\}^n$ and $\mathcal{M} = \{0,1\}^\ell$ where $\ell > n$. Let G be an **efficiently computable**, **deterministic** function that maps n bits to ℓ bits. This is a public function (that is, everyone knows the description of this function). Using this function, we can define an encryption scheme $\mathcal{E}_G$, parameterized by G, with key space $\mathcal{K}$ and message space $\mathcal{M}$.

- $\mathsf{KeyGen}(1^n)$: chooses a uniformly random key $k \in \{0,1\}^n$
- $\mathsf{Enc}(m \in \mathcal{M}, k \in \mathcal{K}) = m \oplus G(k)$
- $\mathsf{Dec}(\mathsf{ct}, k) = \mathsf{ct} \oplus G(k)$

Note that this scheme is very similar to Shannon's One Time Pad, except that the message is masked by $G(k)$ (instead of k). Clearly, the scheme satisfies correctness. What about security? That depends on the structure of G. Below are a few bad choices of G. Convince yourself that the corresponding encryption schemes will not satisfy No-Query-Semantic-Security.

Example 1: Let $G : \{0,1\}^n \to \{0,1\}^{2n}$ be defined as follows: $G(s) = s \mid\mid s$ for all $s \in \{0,1\}^n$, where $\mid\mid$ denotes string concatenation. Show that the encryption scheme $\mathcal{E}_G$ does not satisfy No-Query-Semantic-Security.

Solution: First, it helps to see a few attempts that **do not** work. Consider $m_0 = 0^{2n}$, $m_1 = 1^{2n}$. Encryption of m_0 using key k will be $(k||k)$, while encryption of m_1 with k will be $(\overline{k}||\overline{k})$, where $\overline{k} = 11\ldots1 \oplus k$. Note that both these encryptions are indistinguishable, hence these are not good choices for m_0, m_1.

Next, let us try $m_0 = 0^n||1^n$, and $m_1 = 1^n||0^n$. Encryption of m_0 using key k is $(k||\overline{k})$, while encryption of m_1 using key k is $(\overline{k}||k)$. Again, both encryptions are indistinguishable.

We know that encryption of 0^{2n} will be some string, repeated twice. Therefore, it suffices to find a message m_1 whose encryption is not of this form. Check that if $m_1 = 0^n||1^n$, then its encryption, using any key, is not of this form. Therefore, we can distinguish between encryptions of m_0 and m_1.

Example 2: Let $G : \{0,1\}^n \to \{0,1\}^{2n}$ be some deterministic function that satisfies the following property: for any $s \in \{0,1\}^n$, the XOR of the bits of $G(s)$ is 0. Show that the encryption scheme $\mathcal{E}_G$ does not satisfy No-Query-Semantic-Security.

Solution: For this problem, we are only given one piece of information about G: the parity[a] of its output, on any input, is 0.

Recall, the encryption of a message m using key k is $m \oplus G(k)$. If we consider the parity of the encryption of m, then this is simply $\mathsf{parity}(m) \oplus \mathsf{parity}(G(k)) = \mathsf{parity}(m)$. Therefore, it suffices to find two messages whose parity is different. Set $m_0 = 00\ldots00$ and $m_1 = 00\ldots01$. On receiving the ciphertext, compute its parity — if the parity is 0, then the message m_0 was encrypted, else m_1 was encrypted.

[a] Recall, the parity of a string is the XOR of all its bits. If s_1 and s_2 are two strings, then $\mathsf{parity}(s_1 \oplus s_2) = \mathsf{parity}(s_1) \oplus \mathsf{parity}(s_2)$.

Notice that if the output of G (on a random input s) is sufficiently distinguishable from a uniformly random $2n$-bit string, then we cannot hope for $\mathcal{E}_G$ to satisfy No-Query-Semantic-Security. On the other hand, if we replace $G(k)$ with a uniformly random $2n$-bit string, then we get Shannon's OTP – a perfectly secure encryption scheme. This motivates us to use G whose output 'looks random'.

3.1 Pseudorandom Generators

Functions whose output on random inputs looks random to any efficient algorithm are called pseudorandom generators. As in the case of encryption schemes, we will first define an appropriate security game to capture the intuition "output on random inputs looks random to any efficient algorithm". In this security game between a challenger and an adversary, the challenger chooses a bit b at random. If $b = 0$, it chooses a random n-bit string, and sends $G(s)$ to the adversary. If $b = 1$, it chooses a uniformly random $2n$-bit string and sends it to the adversary. The adversary must guess whether $b = 0$ or 1.

PRG-Security

1. The challenger chooses a bit $b \leftarrow \{0,1\}$, string $s \leftarrow \{0,1\}^n$, $u_1 \leftarrow \{0,1\}^\ell$. It computes $u_0 = G(s)$, and sends u_b to the adversary.
2. The adversary sends its guess b', and wins the security game if $b = b'$.

Figure 2: The PRG Security Game

The formal definition of PRG security is given below.

Definition 05.01. A deterministic polynomial time computable function $G : \{0,1\}^n \to \{0,1\}^\ell$ is a secure pseudorandom generator (PRG) if $\ell > n$, and for any prob. poly. time adversary $\mathcal{A}$, there exists a negligible function $\mu(\cdot)$ such that for all n,

$$\Pr\left[\mathcal{A} \text{ wins the PRG security game against } G\right] \leq 1/2 + \mu(n),$$

where the PRG game is defined in Figure **??**.

3.2 Exponential time/space attacks on PRGs

The following are two simple generic attacks on PRGs. For concreteness, let $G : \{0,1\}^n \to \{0,1\}^{2n}$.

- Exponential time attack: Given $u \in \{0,1\}^{2n}$, we can iterate over all inputs $x \in \{0,1\}^n$, and check if $u = G(x)$ for some x.
- Exponential space attack: Since the description of G is public, one can precompute a $O(n \cdot 2^n)$ sized (sorted) database, containing $G(x)$ for all $x \in \{0,1\}^n$. Later, when the adversary receives $u \in \{0,1\}^{2n}$, it can perform a binary search in $\mathsf{poly}(n)$ time to check if u is PRG output.

(☕) **Challenge Question:** Given the description of G, is it possible to prepare a database of size $2^{n/2}$ s.t., when the adversary receives $u \in \{0,1\}^{2n}$, it can check if u is pseudorandom or not, in $O(2^{n/2})$ steps.

3.3 Secure PRG $\implies$ secure encryption

We will prove this formally in the next lecture. The intuition behind this proof is that if there exists an adversary that breaks the No-Query-Semantic-Security of $\mathcal{E}_G$, then either there exists an adversary that breaks the PRG security of G, or there exists an adversary that breaks the No-Query-Semantic-Security of Shannon's OTP. Since Shannon's OTP is perfectly secure, no adversary can break the No-Query-Semantic-Security of Shannon's OTP. Hence, an attack on $\mathcal{E}_G$ implies an attack on the PRG security of G.

3.4 Some incorrect practical uses of PRGs

PRGs are used in a number of real-world applications. Sometimes, they have been used incorrectly, leading to attacks.

- The two-time attack: Similar to Shannon's OTP, the PRG based construction is also insecure if an adversary sees two messages encrypted with the same key.
- Malleability attack: same as Shannon's OTP.
- The related key attack: Let G be a secure PRG. Suppose you have two inputs s and s', where s and s' are closely correlated (say $s' = s \oplus 00\ldots01$). Can we say that both $G(s)$ and $G(s')$ will look random? Unfortunately, no! Such attacks are called *related key attacks*.

  In fact, there exist functions that are secure PRGs, but insecure against related key attacks. See Sample Assignment 0 for more details.

4 Additional Resources

Relevant sections from textbook [Boneh-Shoup]: Sections 3.1 and 3.3.

5 Questions

Question 05.01. Let $\mathcal{E} = (\mathsf{KeyGen}, \mathsf{Enc}, \mathsf{Dec})$ be an encryption scheme that satisfies Definition 04.02. Consider the following modified encryption scheme $(\mathsf{KeyGen}', \mathsf{Enc}', \mathsf{Dec}')$:

- $\mathsf{KeyGen}' = \mathsf{KeyGen}$
- $\mathsf{Enc}'(m, k) = \Big(\mathsf{Enc}(m, k), \mathsf{Enc}(m, k \oplus 00\ldots01)\Big)$

  In words: the first component of the ciphertext is an encryption of m using key k. The second component is an encryption of m using key k' which is same as k, except that the last bit of k is flipped.
- $\mathsf{Dec}'(\mathsf{ct} = (\mathsf{ct}_1, \mathsf{ct}_2), k) = \mathsf{Dec}(\mathsf{ct}_1, k)$

Can we show that $\mathcal{E}'$ satisfies Definition 04.02, assuming $\mathcal{E}$ does?

Hint: We cannot construct a reduction in this case. In fact, it is possible that $\mathcal{E}$ satisfies No-Query-Semantic-Security, but is insecure if the adversary sees two ciphertexts, one generated using key k and another using $k \oplus 00\ldots01$.

Question 05.02. Listed below are a few deterministic functions $G : \{0,1\}^n \to \{0,1\}^{2n}$. For each of them, show that the corresponding encryption scheme $\mathcal{E}_G$ does not satisfy No-Query-Semantic-Security.

1. $G : \{0,1\}^n \to \{0,1\}^{2n}$ is a deterministic function that satisfies the following property: for any $s \in \{0,1\}^n$, the $(n+1)^{\text{th}}$ bit of $G(s)$ is equal to the XOR of the first n bits of $G(s)$.

   Hint: Set $m_0 = 0^{2n}$, $m_1 = 0^n\ 1\ 0^{n-1}$. Let ct be the challenge ciphertext. Check the XOR of first $n+1$ bits.

2. $G : \{0,1\}^n \to \{0,1\}^{2n}$ such that, for all s, the output $G(s)$ has more than $3n/2$ 1s. What if G satisfies a weaker property: $\Pr\Big[G(s) \text{ has more than } 3n/2 \text{ 1s}\Big] \geq 3/4$ where the probability is over choice of uniformly random $s \leftarrow \{0,1\}^n$.

   Hint: Set $m_0 = 0^{2n}$, $m_1 = 1^{2n}$. Count the number of 1s in challenge ciphertext.

3. $G : \{0,1\}^n \to \{0,1\}^{2n}$ such that, for any s, the output $G(s)$ does not contain the substring "0000".

   Hint: Set $m_0 = 0^{2n}$, pick m_1 uniformly at random. If m_0 is encrypted, then the last four bits cannot be "0000". What can we say about encryption of m_1?