CS 600.442 – Modern Cryptography October 3rd, 2016

Lecture 7: Public Key Encryption

Instructor: Abhishek Jain *Scribe: Neil Fendley*

1 Semantic Security

We began lecture by formalizing the definition of Semantic Security for secret-key encryption as follows:

Definition 1 *A secret-key encryption scheme (Gen, Enc, Dec) is semantically secure if there exists a PPT simulator algorithm S s.t. the following two experiments generate computationally indistinguishable outputs:*

$$\left\{ \begin{array}{r} (m, z) \leftarrow M(1^n), \\ s \leftarrow Gen(1^n), \\ Output\ (Enc(s, m), z) \end{array} \right\} \approx \left\{ \begin{array}{c} (m, z) \rightarrow M(1^n), \\ Output\ S(1^n, z) \end{array} \right\}$$

where M is a machine that randomly samples a message from the message space and arbitrary auxiliary information.

Intuitively, semantic security promises that a PPT adversary does not learn any "new" information about a message m by looking at its ciphertext than what it already knew before (denoted as auxiliary information z). This is because adversary's view (consisting of the ciphertext and z) can be efficiently simulated given only z and no other information on m.

We now show that semantic security and IND-CPA security are, in fact, equivalent security notions.

Theorem 1 *Semantic security and IND-CPA security are equivalent.*

Proof. We prove each case separately:

- SS $\rightarrow$ IND: From SS, we have that for any m_1, $(Enc(m_1), z) \approx S(1^n, z)$. Also from SS, for any m_2, we have that $(Enc(m_2), z) \approx S(1^n, z)$. Then, by transitivity, we get IND-CPA security.
- IND-CPA $\rightarrow$ SS: From IND-CPA security, we have that encryptions of any two messages are indistinguishable. Then, we simply construct a simulator that encrypts the all zeros string, i.e., $S(1^n, z) = (Enc(0^n), z)$. SS security immediately follows from IND-CPA security.

2 Public Key Encryption

Recall that in the setting of private-key encryption, Alice and Bob were allowed to share a secret before communication. In the public-key setting, Alice and Bob don't share a secret any more. Our goal is to be able to have Alice still send a message to Bob in such a manner that no eavesdropper can distinguish between encryptions of m and m'. We formalize the notion of public-key encryption that allows us to do this.

Syntax. A public-key encryption scheme consists of three algorithms:

- $Gen(1^n) \to (pk, sk)$
- $Enc(pk, m) \to c$
- $Dec(sk, c) \to m'$ or $\perp$

Where $Gen(1^n)$ generates a public key pk and a secret key sk. Encryption takes as input a public key pk (which everyone including the adversary will have access to) and a message m and outputs a cipher text. Decryption takes as input a ciphertext c and a secret key sk, and outputs a message or $\perp$.

We say that the encryption scheme is *correct* if $Dec(sk, Enc(pk, m)) = m$, meaning if you encrypt and then decrypt a message with the public and secret key respectively, you should get the original message back.

We now provide our definition of security. We start with a weak definition (also called selective security):

Definition 2 *A public key encryption scheme is weakly-indistinguishably secure under chosen plaintext attack (IND-CPA) if for all n.u. PPT adversaries $\mathcal{A}$ there exists a negligible function $\mu(\cdot)$ such that the probability of*

$$Pr\left[\begin{array}{rl} (pk, sk) \leftarrow Gen(1^n), & \\ (m_0, m_1) \leftarrow \mathcal{A}(1^n), & : \mathcal{A}(pk, Enc(pk, m_b)) = b \\ b \xleftarrow{\$} \{0,1\} & \end{array}\right] \leq \frac{1}{2} + \mu(n)$$

What this means is that we generate two random keys, pk and sk, and allow the Adversary to send us any two messages, m_0 and m_1. The adversary $\mathcal{A}$ will be given the cipher text of a random choice of the two messages and the public key. Given this the adversary will then be unable to distinguish which message was encrypted, i.e., whether, $c = Enc(pk, m_1)$ or $Enc(pk, m_0)$.
We can then adapt this into the definition of strong security by changing a single point, we will give the Adversary the public key before we have it send us messages.

Definition 3 *A public key encryption is indistinguishably secure under chosen plaintext attack (IND-CPA) if for all n.u. PPT adversaries $\mathcal{A}$ there exists a negligible function $\mu(\cdot)$ s.t.:*

$$Pr\left[\begin{array}{rl} (pk, sk) \leftarrow Gen(1^n), & \\ (m_0, m_1) \leftarrow \mathcal{A}(1^n, pk), & : \mathcal{A}(pk, Enc(pk, m_b)) = b \\ b \xleftarrow{\$} \{0,1\} & \end{array}\right] \leq \frac{1}{2} + \mu(n)$$

This is stronger than the weak version because if you give A the public key before then it can attempt to pick messages based on generated cipher texts.

Just as in the secret key encryption schemes we want to obtain multiple message security as well. However examining our definition we can realize that

Lemma 2 *One-message security implies multi-message security for public-key encryption*

We now briefly sketch the proof. For simplicity, we only show that one-message security implies two-message security. The general case can be derived in a similar manner.

Suppose that (m_0^1, m_1^1) is the first message pair and (m_0^2, m_1^2) is the second message pair. Now, consider the following sequence of hybrid experiments:

- H_0: This corresponds to the case when $b = 0$, i.e., $c^1 = Enc(pk, m_0^1)$ and $c^2 = Enc(pk, m_0^2)$
- H_1: We now change the first ciphertext to encryption of m_1^1, but keep the second intact. That is, $c^1 = Enc(pk, m_1^1)$ and $c^2 = Enc(pk, m_0^2)$
- H_2: This corresponds to the case when $b = 1$, i.e., $c^1 = Enc(pk, m_1^1)$ and $c^2 = Enc(pk, m_1^2)$

We want to show that H_0 is indistinguishable from H_2, meaning that there is no PPT adversary $\mathcal{A}$ that can correctly predict the challenge bit b except with probability $\frac{1}{2}$. We start by proving that H_0 and H_1 are indistinguishable. The proof of indistinguishability of H_1 and H_2 is analogous and left as an exercise. Combining the two, we obtain that H_0 and H_2 are indistinguishable, as required.

Suppose that there is a PPT distinguisher $\mathcal{A}$ that distinguishes between H_0 and H_1 with noticeable probability. Using $\mathcal{A}$, we will construct an adversary $\mathcal{A}'$ that breaks one-message security of the encryption scheme.

First, $\mathcal{A}'$ receives public key pk from its challenger and forwards it to $\mathcal{A}$. Now, $\mathcal{A}$ sends two message pairs: (m_0^1, m_1^1) and (m_0^2, m_1^2). $\mathcal{A}'$ sets its own challenge message pair to $(m_0 = m_0^1, m_1 = m_1^1)$ and sends it to its challenger. Upon receiving a challenge ciphertext c, it sets $c^1 = c$ and computes $c^2 \leftarrow Enc(pk, m_0^2)$. It sends (c^1, c^2) to $\mathcal{A}$. Whenever $\mathcal{A}$ responds with its guess b', $\mathcal{A}'$ forwards it to its challenger.

Now, note that if c is an encryption of $m_0 = m_0^1$, then the above corresponds to H_0, otherwise, it corresponds to H_1. Therefore, if $\mathcal{A}$ distinguishes with noticeable probability, then so does $\mathcal{A}'$. This contradicts the one-message security of the encryption scheme. ■

3 Trapdoor Permutations

We start by defining a collection of one-way functions and permutations, and then proceed to give a definition of trapdoor permutations.

Definition 4 *A collection of one way functions is a family of functions*

$$F = \{f_i : D_i \to R_i\},\ i \in I$$

that satisfy the following conditions:

- *Sampling Function: There exists a PPT Gen such that $Gen(1^n)$ outputs $i \in I$*
- *Sampling from Domain: There exists a PPT algorithm that on input i outputs a uniformly random element of D_i*
- *Evaluation: There exists a PPT algorithm that on input $i \in D_i$ outputs $f_i(x)$*
- *Hard to invert: For every n.u. PPT adversary $\mathcal{A}$, there exists a negligible function $\mu(\cdot)$ s.t.*

$$Pr[i \leftarrow Gen(1^n),\ x \leftarrow D_i,\ y \leftarrow f_i(x) : f_i(A(1^n, y, i)) = y] \leq \mu(\cdot)$$

Theorem 3 *There exists a collection of one-way functions iff there exists a strong one-way function.*

Proof. The forward direction is obvious. If there exists a strong one-way function we can have the collection that is the single function, and therefore collections of one way functions exist. The other direction we must construct a single one-way function given a collection F. To do this define $g(r_1, r_2)$ to be $i, f_i(x)$, where $f_i \in F$, i is random bits generated from r_1 and x is sampled randomly from D_i using r_2. This is a strong one way function because the ability to invert any single element of it would mean you could invert some $f_i \in F$, which is not possible. ■

This leads us into our next definition, which is collections of One-way permutations.

Definition 5 *A collection*

$$F = \{f_i : D_i \to R_i \; i \in I\}$$

is a collection of one-way permutations if F is a collections of OWF and for every $i \in I$, f_i is a one-way permutation.

Trapdoor Permutations. Next, we formalize the definition of a collection of Trapdoor permutations.

Definition 6 *A collection of trapdoor permutations is a family of permutations*

$$F = \{f_i : D_i \to R_i \; i \in I\}$$

satisfying the following conditions:

- *Sampling Function: $\exists$ a PPT Gen s.t. $Gen(1^n) \to (i, t) \in I$*
- *Sampling from Domain: : $\exists$ a PPT algorithm that on input i outputs a uniformly random element of D_i*
- *Evaluation: $\exists$ a PPT that on input $i, x \in D_i$ outputs $f_i(x)$*
- *Hard to invert: $\forall$ n.u. PPT adversary $A, \exists$ a negligble function $\mu(\cdot)$ s.t.:*

$$Pr[i \leftarrow Gen(1^n), x \leftarrow D_i, y \leftarrow f_i(x) : f_i(A(1^n, i, y)) = y] \leq \mu(\cdot)$$

- *Inversion with a trapdoor: $\exists$ a PPT algorithm that given (i, t, y) will return $f_i^{-1}(y)$*

Roughly speaking, a trapdoor permutation is essentially a one way permutation that has a "trapdoor" t, that allows you to invert. We will now show how to build a public key encryption scheme from a family of one way trapdoor permutations.

4 Public-key Encryption from Trapdoor Permutations

Let $F = \{f_i : D_i \to R_i\}$ $i \in I$ be a family of trapdoor permutations and let h_i be the hardcore predicate associated with f_i:

- $Gen(1^n) : (f_i, f_i^{-1}) \leftarrow Gen_T(1^n)$ Outputs $(pk, sk) \leftarrow ((f_i, h_i), f_i^{-1})$
- $Enc(pk, m)$: Pick $r \xleftarrow{\$} \{0,1\}^n$ Outputs $(f_i(r), h_i(r) \oplus m)$
- $Dec(sk, (c_1, c_2)) : r \leftarrow f^{-1}(c_1)$. Outputs $c_2 \oplus h_i(r)$

Theorem 4 *(Gen, Enc, Dec) is an IND-CPA secure public encryption scheme.*

Proof. Consider the following sequence of hybrids:

- $H_0 : b = 0, c = (f_i(r), h_i(r) \oplus m_0)$
- $H_1 : b = 0, c = (f_i(r), z \oplus m_0)$
- $H_2 : b = 1, c = (f_i(r), z \oplus m_1)$
- $H_3 : b = 1, c = (f_i(r), h_i(r) \oplus m_1)$

where $z \xleftarrow{\$} \{0,1\}$ is a random bit. Note that in order to prove the IND-CPA security of the encryption scheme, it suffices to prove that H_0 and H_3 are computationally indistinguishable.

The indistinguishability of H_0 and H_1 follows from the security of the one-time pad. Indeed, note that the only difference between H_0 and H_1 is that in H_0, the message m_0 is masked using $h_i(r)$, whereas it is masked with a random bit z in H_1. Then, if there exists a PPT distinguisher between H_0 and H_1, we can use it to construct an adversary for hard-core predicate h_i. The reduction is left as an exercise.

Next, the indistinguishability of H_1 and H_2 follows immediately from the security of one-time pad. Finally, the indistinguishability of H_2 and H_3 can be argued in an analogous manner as for H_0 and H_1.

Combining the above, we obtain that H_0 and H_3 are indistinguishable. ■

5 Trapdoor Permutations from RSA

The rest of the lecture was spent outlining trapdoor permutations from the RSA assumption. We begin by defining the RSA collection.

Definition 7 *$RSA = \{f_i : D_i \to R_i\}$ where:*

- $i = \{(N, e) | N = p * q \ s.t. \ p, q \in \prod_n, e \in \mathbb{Z}^*_{\Phi(N)}\}$
- $D_i = \{x | x \in \mathbb{Z}^*_N\}$
- $R_i = \mathbb{Z}^*_N$
- $Gen(1^n) \to ((N, e), d)$ *where* $(N, e) \in I$ *and* $e * d \ mod \ \Phi(N)$
- $f_{N,e}(x) = x^e \ mod \ N$
- $f^{-1}_{N,e}(y) = y^d \ mod \ N$

It is easy to verify that that $f_{N,e}$ is a permutation.

Next, we describe the RSA assumption:

Definition 8 *For any n.u. PPT adversary $\mathcal{A}$, there exists a negligible function $\mu(\cdot)$ s.t.:*

$$Pr\left[\begin{array}{c} p,q \leftarrow_{\$} \prod_n,\ N = p * q,\ e \leftarrow_{\$} \mathbb{Z}^*_{\Phi(N)} \\ y \leftarrow_{\$} \mathbb{Z}^*_N; x \leftarrow \mathcal{A}(N,e,y): \quad x^e = y\ mod\ N \end{array}\right] \leq \frac{1}{2} + \mu(n)$$

Finally, we note that based on the RSA assumption, we can establish that the RSA collection is a family of trapdoor permutations. The proof is left as an exercise.

Theorem 5 *Assuming the RSA assumption, the RSA collection is a family of trapdoor permutations.*