CS 290G (Fall 2014) – Introduction to Cryptography Nov 20th, 2014

Lecture 11: Building PRF and Public Key Encryption

Instructor: Rachel Lin *Scribe: Binyi Chen*

1 Recap

Last class we proposed the notion of multi-message security, in which for any two sequences of messages, the adversary cannot distinguish with non-negligible probability. In order to obtain multi-message security, we define pseudorandom function (PRF). Based on the existence of PRF, we finally construct a multi-message secure secret key encryption. This class we will give a construction of PRF based on a length-doubling PRG, and we will extend the encryption scheme into Public Key Encrytion.

2 Building a PRF from PRG

Theorem 1 (Goldreich, Goldwasser, and Micali 1985) *If there is a length doubling PRG G, such that $|G(x)| = 2|x| = 2n$, then there is a PRF $F : \{0,1\}^n \times \{0,1\}^m \to \{0,1\}^n$, where m is any polynomial of n.*

Since the length of the output can be made arbitrary (see homework 2), we can build any PRF $F : \{0,1\}^n \times \{0,1\}^m \to \{0,1\}^l$, where m, l are any polynomial of n.

In the proof sketch from class slides, the core idea is using PRG to build the truth-table of Pseudo-Random Function, nevertheless naively building the truth-table in a sequential order is infeasible, since the length of truth-table is exponential, and the output cannot be even pseudo-random after exponential times of invoking PRG.

Fortunately, the number of queries is polynomial to n, thus we can calculate the entries of truth-table in a dynamic way. In the final construction, it uses a tree to build the truth table, in which each node is an independent PRG. Each time when adversary query x, it traverse from root to the leaf corresponding x and build the corresponding $f(x)$ by iteratively using PRG. It only costs m steps, where m is the depth of the tree.

For more details of the proof sketch please see the class slides. For the formal details of theorem's proof, please refer to the Section 3.6 "Pseudorandom Functions" in the Book [1] by Oded Goldreich.

3 Public Key Encryption Scheme

3.1 Motivation

Suppose a lot of customers want to communicate and send private information to Bank of America(BOA). Using only secret key encryption scheme, BOA has to share and exchange a secret key with each customer, which is a super burden for BOA. Fortunately, we can hope to build another encryption scheme, in which BOA publish one single public key pk, every customer who wants to send message to BOA can encrypt the message using pk, but only BOA who has the secret key sk can decrypt the ciphertext. In this way, BOA only needs to secretly generate a pair of key (pk, sk), and no key exchange is needed.

3.2 Definition

Informally speaking, Alice and Bob want to communicate in a secure way, Eve is the adversary who try to extract information from the communication between Alice and Bob. Bob firstly generate the key pair (pk, sk), then publish pk (Alice and Eve all know pk now), whenever Alice want to send message m to Bob, She encrypts m as $c = Enc(pk, m)$ and send ciphertext c to Bob, Bob can then use sk to decrypt the ciphertext and recover the message $m = Dec(sk, c)$. When Bob want to send message to Alice, Alice can also do similar things (generating pair of keys).

Definition 1 *A public key encryption algorithm with parameter n and message space $M = \{0,1\}^m$ consists of three parts* ***(Gen, Enc, Dec)****.*

- *$Gen(1^n)$ is a Probabilistic Polynomial Time (PPT) Turing Machine that samples a key pair (pk, sk) given input 1^n.*
- *$Enc(pk, m)$ is a Probabilistic Polynomial Time (PPT) Turing Machine that generates ciphertext c using public key k and message m.*
- *$Dec(sk, c)$ is a Deterministic Polynomial Time (DPT) Turing Machine that generates plaintext m given secret key sk and ciphertext c.*

(Gen, Enc, Dec) *satisfies the* ***correctness*** *property, i.e: for any parameter n and message m, we have:*

$$\mathbf{Pr}[(pk, sk) \leftarrow Gen(1^n) : Dec(sk, Enc(pk, m)) = m] = 1$$

We have to define security for public encryption scheme, the first step, similar to secret key encryption, is single-message security:

Definition 2 *A public encryption scheme is single-message secure iff* $\forall \{m_{0n}\}, \{m_{1n}\}$:

$$\{(pk, sk) \leftarrow Gen(1^n) : (Enc(pk, m_{0n}), pk)\} \approx \{(pk, sk) \leftarrow Gen(1^n) : (Enc(pk, m_{1n}), pk)\}$$

Single-message security is definitely a weak notion of security, similar to the analysis in secret key encryption scheme, we hope the scheme to be multi-message secure.

Definition 3 *A public encryption scheme is multi-message secure iff* $\forall \{\vec{m}_{0n}\}, \{\vec{m}_{1n}\}$:

$$\begin{aligned} &\{(pk, sk) \leftarrow Gen(1^n) : Enc(pk, \vec{m}_{0n}[1]), ..., Enc(pk, \vec{m}_{0n}[l_n]), pk\} \\ \approx &\{(pk, sk) \leftarrow Gen(1^n) : Enc(pk, \vec{m}_{1n}[1]), ..., Enc(pk, \vec{m}_{1n}[l_n]), pk\} \end{aligned}$$

Fortunately, different from Secret Encryption Scheme, in PKE scheme, single-message security directly implies multi-message security.

Theorem 2 *If PKE (Gen, Enc, Dec) is single-message secure, then it is also multi-message secure.*

4 Construction

Recall that RSA collection of OWP is a collection of functions such that $\forall f$ in the collection, there exists a d that:

1. f is easy to evaluate.
2. It is hard to invert given only f and y.
3. It is easy to invert given f, y and d.

Consider the property of public key and secret key in PKE scheme, we find RSA collection seems to be a perfect match. In RSA collection, (N, e) defines a function $f_{N,e}(x) = x^e \bmod N$, there exists an unique trapdoor d, such that $e \cdot d = 1 \bmod \phi(N)$. When given d, it becomes easy to invert the function $f_{N,e}$. A natural thought is making N, e to be the public key, while d to be the secret key. This leads to our first attempt to construct a PKE.

4.1 First Attempt

1. $Gen(1^n)$: Sample primes $p, q \in \{0,1\}^n$, set $N = p \cdot q$, sample $e \in Z_{\phi(N)}$ and calculate corresponding d. Set $(pk, sk) = ((N, e), d)$.
2. $Enc(pk, m)$: Given message m, set ciphertext $c = Enc(pk, m) = m^e \bmod N$.
3. $Dec(sk, c)$: Given ciphertext c, we recover the message $m = Dec(sk, c) = c^d \bmod N$.

Obviously, the above scheme satisfies the correctness property. Nevertheless, it is not single-message secure, because the Encryption algorithm is determinitic instead of random. That means, given message m_0, m_1, and challenging ciphertext c, we can efficiently

check whether $Enc(pk, m_0) = c$ or $Enc(pk, m_1) = c$, then know which message that ciphertext is come from, thus the security will be easily broken.

To overcome the obstacle, what we need to do is adding randomness on encryption scheme, which make the random variable $Enc(pk, m_0)$ insdistinguishable to $Enc(pk, m_1)$. Next class we will show the details of the construction.

References

[1] Oded Goldreich. Foundations of cryptography: Volume 1, basic tools.