Lecture 16: Public Key Encryption:II

Instructor: Omkant Pandey

Spring 2017 (CSE 594)

Last time

- PKE from ANY trapdoor permutation
- RSA-based trapdoor permutation

Today

- ElGamal Public-Key Encryption
- Some Comments about Textbook RSA
- Some attacks on RSA
- LWE based Public-Key Encryption
- Scribe notes volunteeer?

(Weak) Indistinguishability Security for PKE

Definition (Secure Public-Key Encryption)

A public-key encryption scheme $\{\mathsf{Gen}, \mathsf{Enc}, \mathsf{Dec}\}$ is said to be secure if for all non-uniform PPT D there exists a negligible function μ such that for all $n \in \mathbb{N}$, for all pair of messages $m_0, m_1 \in \mathcal{M}$ such that $|m_0| = |m_1|$, D distinguishes between the following distributions with at most $\nu(n)$ advantage:

- $\{(pk, sk) \leftarrow \mathsf{Gen}(1^n) : (pk, \mathsf{Enc}(pk, m_0))\}$
- $\{(pk, sk) \leftarrow \mathsf{Gen}(1^n) : (pk, \mathsf{Enc}(pk, m_1))\}$

I.e., the distributions above are computationally indistinguishable.

Recall: DDH Problem

- Recall the DDH Problem: for a large prime p, and a generator g for the group $\mathbb{Z}_p^*$:

$$\left\{x \leftarrow \mathbb{Z}_p^*, y \leftarrow \mathbb{Z}_p^* : \left(g^x, g^y, g^{xy}\right)\right\} \approx_c \left\{x \leftarrow \mathbb{Z}_p^*, y \leftarrow \mathbb{Z}_p^*, z \leftarrow \mathbb{Z}_p^* : \left(g^x, g^y, g^z\right)\right\}$$

- Recall: $|\mathbb{Z}_p^*| = p - 1$ is not prime! (This makes the problem easier in some special cases)
- Recall: we work with a prime order subgroup of $\mathbb{Z}_p^*$ by picking a safe prime $p = 2q + 1$ and $g = x^2$ for a random $x \in \mathbb{Z}_p^*$.
- G_q = group generated by $g = \{g^0, g^1, \ldots, g^{q-1}\}$. $|G_q| = q$.
- There are other ways as well to obtain prime order groups G where DDH is conjectured to be hard.

Recall: DDH Problem

- DDH Assumption: Let G be a group of prime order q and $g \in G$ be a generator of G

$$\left\{x \leftarrow \mathbb{Z}_q, y \leftarrow \mathbb{Z}_q : \left(g^x, g^y, g^{xy}\right)\right\} \approx_c \left\{x \leftarrow \mathbb{Z}_q, y \leftarrow \mathbb{Z}_q, z \leftarrow \mathbb{Z}_q : \left(g^x, g^y, g^z\right)\right\}$$

ElGamal Public-Key Encryption

- ElGamal Scheme: Let G be a prime order group where DDH Assumption holds. The description of G and its order q are publicly known.
- Messages are group elements and the message space is $\mathcal{M} = G$.
  - $\mathsf{Gen}(1^n)$: sample $g \leftarrow G$, $x \leftarrow \mathbb{Z}_q$ and set $h = g^x \in G$. Output (pk, sk) where:

$$pk = (g, h) \qquad sk = x$$

  - $\mathsf{Enc}(pk, m)$ for $m \in G$: choose a random $r \leftarrow \mathbb{Z}_q$ and output:

$$(g^r, m \cdot h^r)$$

  - $\mathsf{Dec}(sk, c)$ where $c = (c_1, c_2)$: output

$$m = \frac{c_2}{c_1^x} = c_2 \times (\text{Inverse of } c_1^x)$$

- **Correctness:** $m = \frac{c_2}{c_1^x} = \frac{m \cdot h^r}{g^{rx}} = \frac{m \cdot (g^x)^r}{g^{rx}} = m.$

Security of ElGamal Scheme

- Proof based on DDH Assumption: We now prove that ElGamal scheme is secure assuming that the DDH assumption holds.
- We have to show that for all $m_0, m_1 \in G$ these two distributions are indistinguishable:
  - $\{(pk, sk) \leftarrow \mathsf{Gen}(1^n) : (pk, \mathsf{Enc}(pk, m_0))\}$
  - $\{(pk, sk) \leftarrow \mathsf{Gen}(1^n) : (pk, \mathsf{Enc}(pk, m_1))\}$
- Let D be a PPT algorithm.
- Start with the first distribution, and slowly go to the second distribution.

Security of ElGamal Scheme

- **Game-0**: $\{(pk, sk) \leftarrow \mathsf{Gen}(1^n) : (pk, \mathsf{Enc}(pk, m_0))\}$

  $= \{g, h, g^r, m_0 \cdot h^r\} \quad = \{g, g^x, g^r, m_0 \cdot g^{xr}\}$
- **Game-1**: Use g^z for a random z instead of g^{xr}. We get:

  $= \{g, g^x, g^r, m_0 \cdot g^z\}$
- **Claim:** Game-0 and Game-1 are indistinguishable.
- **Proof:** Suppose that D can distinguish Game-0 and Game-1.
  - We construct D' which can break DDH Assumption
  - D' gets as input (g, g^x, g^y, g^α) where $\alpha = xy$ or $\alpha = z$.
  - D' sends $(g, g^x, g^y, m_0 \cdot g^\alpha)$ to D,
  - D' outputs whatever D outputs.
- If $\alpha = xy$, D is in Game-0. If $\alpha = z$, D is in Game-1.
- If D tells Game-0, Game-1 apart, D' tells DDH tuples apart! □

Textbook RSA-Encryption

- **Public-Key Encryption:**
  - $\mathsf{Gen}(1^n)$: Sample $p, q \leftarrow \Pi_n$ and set $N \leftarrow pq$.
    Sample $e \leftarrow \mathbb{Z}^*_{\phi(N)}$ and compute d s.t. $ed = 1 \mod \phi(N)$.
    Output $pk = (N, e)$ and $sk = (N, d)$.
  - Message space $\mathcal{M} = \mathbb{Z}^*_N$
  - $\mathsf{Enc}(pk, m)$ for $pk = (N, e)$ outputs $f_{N,e}(m) = m^e \mod N$.
  - $\mathsf{Dec}(sk, c)$ for $sk = (N, d)$ outputs $c^d \mod N$.
- The correct way to encrypt: construction from previous class.
- More efficient way to encrypt: RSA-OAEP+

Textbook RSA-Signature

- RSA can be used as a signature as well! Simply use e to verify and d to sign instead of decrypt!
- **Signature scheme:**
  - $\mathsf{Gen}(1^n)$: Sample $p, q \leftarrow \Pi_n$ and set $N \leftarrow pq$.
    Sample $e \leftarrow \mathbb{Z}^*_{\phi(N)}$ and compute d s.t. $ed = 1 \mod \phi(N)$.
    Output $vk = (N, e)$ and $sk = (N, d)$.
  - Message space $\mathcal{M} = \mathbb{Z}^*_N$
  - $\mathsf{Sign}(sk, m)$ for $sk = (N, d)$ outputs $\sigma = m^d \mod N$.
  - $\mathsf{Verify}(vk, m, \sigma)$ for $vk = (N, e)$ outputs 1 iff $\sigma^e = m \mod N$.

Remarks on Textbook RSA-Signature

- Signature function $\mathsf{Sign}(sk, m)$ for $sk = (N, d)$:

$$f_{N,d}(m) = m^d \mod N.$$

  Verification checks $m = \sigma^e \mod N$.

- Signature is deterministic but that is not a problem !
- Can you **forge** a signature?
- Not if someone gives you a random challenge (RSA Assumption).
- However: what if you select your own messages?
- **Forgery**: Choose a random $\alpha \leftarrow \mathbb{Z}_N^*$.
  Adversary knows the verification key $vk = (N, e)$.
  It can compute:

$$\beta = \alpha^e \mod N.$$

- Notice that (β, α) is a valid (message, signature) pair!
- Read: how to sign from any trapdoor permutation.

Attacks on the RSA Function

- To speed up encryption, choose a short e: e.g., $e = 3$.
- This is often a big problem!
- **A Simple Example** (Coppersmith, Hastad, and Boneh):
  - Suppose Alice broadcasts m to 3 people with keys $(N_1, 3), (N_2, 3), (N_3, 3)$.
  - $c_1 = m^3 \mod N_1, c_2 = m^3 \mod N_2$ and $c_3 = m^3 \mod N_3$
  - Suppose that N_1, N_2, N_3 are co-primes (no common factors, otherwise easy to get m).
  - You can compute (by Chinese Remainder Theorem):

$$C' = m^3 \mod N_1 N_2 N_3.$$

  - m is less than $N_1, N_2, N_3 \Rightarrow m^3 < N_1 N_2 N_3$.
  - Therefore, $m = \sqrt[3]{C'}$ on integers (modulus plays no role)!

Attacks on the RSA Function

- How often can you apply this attack?
- When same e is used by at least $k \geqslant e$ parties
- This takes modulus out of the equation and you can solve over integers (easy)
- If e is large enough, attack is not practical.
- Current wisdom: low exponent RSA when used carefully with appropriate padding is still secure.
- You can use e of special form, e.g., $e = 2^{16} + 1$ to speed up exponentiation and use appropriate padding.
- **(M. Weiner):** If $d < \frac{1}{3}N^{0.25}$, easy to get d from (N, e).
- **(Boneh-Durfee):** If $d < N^{0.292}$, east to get d from (N, e).

LWE-based Public Key Encryption

- Let $q \geqslant 2$ be a modulus, n the security parameter (a.k.a dimension), and $\alpha \ll 1$ an erroer parameter such that $\alpha q > \sqrt{n}$.
- <u>LWE Instance:</u>
  - choose a random (column) vector $\mathbf{s} \xleftarrow{\$} \mathbb{Z}_q^n$ (secret)
  - choose a random matrix of coefficients $\mathbf{A} \xleftarrow{\$} \mathbb{Z}_q^{m \times n}$
  - choose a Gaussian error vector $\mathbf{e} \xleftarrow{\chi} \mathbb{Z}^m$ (column) where χ is a Gaussian distribution over $\mathbb{Z}$ with parameter αq
  - Let
$$\mathbf{b} = \mathbf{A} \cdot \mathbf{s} + \mathbf{e}$$
  - The LWE instance is: $(\mathbf{A}, \mathbf{b})$
- <u>Decisional LWE Assumption:</u> hard to distinguish an LWE pair from a random instance.

$$(\mathbf{A}, \mathbf{b}) \;\approx_c (\mathbf{A}, \mathbf{u})$$

where $\mathbf{u} \in \mathbb{Z}_q^m$ is a random column vector.

LWE-based Public Key Encryption

- Regev's scheme based on LWE.
- Key Generation:
  - choose $\mathbf{s} \xleftarrow{\$} \mathbb{Z}_q^n, \mathbf{A} \xleftarrow{\$} \mathbb{Z}_q^{m\times n}, \mathbf{e} \xleftarrow{\chi} \mathbb{Z}^m$, $\mathbf{b} = \mathbf{A}\cdot\mathbf{s} + \mathbf{e}$ (as before).
  - the keys are:
  $$pk = (\mathbf{A}, \mathbf{b}), \quad sk = \mathbf{s}$$
- Encryption (for a bit): pick a row-vector of bits $\mathbf{x} \xleftarrow{\$} \{0,1\}^m$, output:
$$(\mathbf{c} = \mathbf{x}\mathbf{A}, c' = \mathbf{x}\mathbf{b} + \mathsf{bit}\cdot\frac{q}{2})$$
- Decryption:
$$c' - \mathbf{c}\cdot\mathbf{s} = (\mathbf{x}\mathbf{b} + \mathsf{bit}\cdot\frac{q}{2}) - \mathbf{x}\mathbf{A}\mathbf{s} = (\mathbf{x}\mathbf{b} + \mathsf{bit}\cdot\frac{q}{2}) - \mathbf{x}\mathbf{b} + \mathbf{x}\mathbf{e} \approx \mathsf{bit}\cdot\frac{q}{2}.$$
- Parameters: $n^2 \leqslant q \leqslant 2n^2, m = 1.1n\log q, \alpha = 1/(\sqrt{n}\log^2 n)$.

LWE-based Public Key Encryption

- Correctness: if not for the error term, the value would be either 0 or $q/2$.
  - The error is adding at most m independent normally distributed variables whose standard deviation is $\sqrt{m}\alpha q < q/\log n$.
  - The probability that it goes over $q/4$ is negligible.
- Security: (LWE + LHL)
  - Game 0: Real pk = LWE instance = $(\mathbf{A}, \mathbf{b})$
  - Game 1: change pk to a random instance = $(\mathbf{A}, \mathbf{u})$
  - Game 2: change bit from 0 to 1 (one-time pad, due to LHL)
  - Game 3: change pk back to LWE instance