



Public Key Cryptography

Murat Kantarcioglu



Definition of Public Key Encryption

- ★ A public key encryption is a triple (G, E, D) of PPT algorithm
 - ▶ Given security parameter k , $(e, d) \leftarrow G(1^k)$
where e is the public key and d is the private key
 - ▶ Given message m , $c \leftarrow E(1^k, e, m)$
 - ▶ Given ciphertext c , $m \leftarrow D(1^k, d, c)$
 - ▶ The system is secure!!! (More on this later)

- ★ Input size should be polynomial in terms of k

- ★ Encryption and Decryption could be probabilistic



Trapdoor Function Model

- ★ Trapdoor function model (G, E, D) are PPT algorithms
- ★ $G(1^k)$ outputs (f, t_f) where f is a trapdoor function
 - ▶ RSA: $G(1^k)$ outputs $(e, d, N = pq)$
- ★ For every message $m \in M$, E s.t. $E(f, m) = f(m) = c$
 - ▶ RSA: $E(f, m) = m^e \bmod N$
- ★ Given $c \in E(f, m)$ and t_f , $D(t_f, c) = f^{-1}(c) = m$
 - ▶ RSA: $D(t_f, c) = D(d, c) = c^d \bmod N = m$
- ★ For every PPT A , for randomly chosen f and $c = f(m)$
 - ▶ $Pr[A(f, c) = m]$ is negligible in term of k
 - ▶ RSA: Given c, e, N it is hard to find m



Problems with Trapdoor Function Model

- ★ Trapdoor functions are assumed to be hard to invert on average
- ★ It may be easy to invert them on special message
 - ▶ RSA: For $m = 1$, $E(m) = m^e \bmod N = 1$
- ★ Partial Information may be revealed
 - ▶ RSA: $J_n(m) = J_n(m^e \bmod N)$
- ★ Relationship between Encrypted Messages
 - ▶ it is easy to detect when message is resend
 - ▶ RSA: If the same exponent e for encrypting fixed m with different N s, then m could be recovered
- ★ Low exponent attack for RSA ($e=3$)
$$c_1 = m^3, c_2 = (m + 1)^3 \Rightarrow \frac{c_2 + 2c_1 - 1}{c_2 - c_1 + 2} = m$$



Rabin's Public Key System

- ★ For Rabins system $G(1^k)$ outputs $n = pq$, p and q
- ★ Define $f_n(m) = m^2 \bmod n$
- ★ Define $f^{-1}(m^2) = x$ s.t. $x^2 = m^2 \bmod n$
- ★ Note that inverse of rabin function has four outputs
 - ▶ $x^2 = m^2 \bmod p$ has two solutions
 - ▶ $x^2 = m^2 \bmod q$ has two solutions
 - ▶ Total four solutions due to CRT
- ★ In practice, some additional information is needed for unique inverse
 - ▶ It is easy if Message space M is sparse in Z_n^*



Rabins's Public Key Cryptosystem

- ★ Inverting Rabins function is as hard as factoring
- ★ Note if p, q is known inverting the Rabins function is easy
- ★ Assume you have an adversary A that inverts Rabins function
- ★ Defining adversary B for factorization using A is easy
 - ▶ Adversary $B(n)$
 - 1 $i \xleftarrow{\$} Z_n^*$
 - 2 $y \leftarrow A(i^2 \bmod n, n)$
 - 3 if $y^2 = i^2 \bmod n$ and $y \neq \pm i$ then
 - 4 return $\gcd(i \pm y, n)$
 - 5 else
 - 6 jump to [1]



Rabin's Public Key Cryptosystem

- ★ Note if $y^2 = i^2 \pmod n$ and $y \neq \pm i$ then
 - ▶ $y - i \neq 0$ and $y + i \neq 0$
 - ▶ $y^2 = i^2 \Rightarrow (y - i)(y + i) = 0 \pmod n$
 - ▶ \Rightarrow either $\gcd(y + i, n) \neq 0$ or $\gcd(y - i, n) \neq 0$

- ★ Also existence of B implies chosen ciphertext attacks



Defining Security

- ★ Goal: Model security as an opaque envelope
- ★ Indistinguishable Security

Definition 7.2 We say that a Public Key Cryptosystem (G, E, D) is *polynomial time indistinguishable* if for every PPT M, A , and for every polynomial Q, \forall sufficiently large k

$$\Pr(A(1^k, e, m_0, m_1, c) = m \mid (e, d) \stackrel{R}{\leftarrow} G(1^k) ; \{m_0, m_1\} \stackrel{R}{\leftarrow} M(1^k) ; m \stackrel{R}{\leftarrow} \{m_0, m_1\} ; c \stackrel{R}{\leftarrow} E(e, m)) < \frac{1}{2} + \frac{1}{Q(k)} \quad (7.1)$$



Polynomial Indistinguishability

- ★ The difference between public key and private key encryption is A given the encryption function
- ★ Note that any deterministic scheme fails the security definition
 - ▶ Given f, m_0, m_1, c where $c \in \{f(m_0), f(m_1)\}$, finding $f^{-1}(c)$ is easy
- ★ Even if adversary know either m_0 or m_1 is encrypted, could not tell exactly which one is encrypted.



Semantic Security

- ★ Inspired by the Shannons perfect security definition
- ★ It is assumed that adversary is computationally bounded

Definition 7.3 We say that an encryption scheme (G, E, D) is *semantically secure* if for all PPT algorithms M and A , functions h , polynomials Q there is a PPT B such that for sufficiently large k ,

$$\begin{aligned} \Pr(A(1^k, c, e) = h(m) \mid (e, d) \stackrel{R}{\leftarrow} G(1^k) ; m \stackrel{R}{\leftarrow} M(1^k) ; c \stackrel{R}{\leftarrow} E(e, m)) \\ \leq \Pr(B(1^k) = h(m) \mid m \stackrel{R}{\leftarrow} M(1^k)) + \frac{1}{Q(k)} \end{aligned} \quad (7.2)$$

- ★ A public key cryptosystem passes Indistinguishable Security iff it passes Semantic Security



Trapdoor Hardcore Predicates

-
- ★ Trapdoor predicate model (G, E, D, S) are PPT algorithms and $B : M \mapsto \{0, 1\}$
 - ★ $G(1^k)$ outputs (f, t_f) where $f : M \mapsto C$ is a trapdoor function
 - ▶ RSA: $G(1^k)$ outputs $(e, d, N = pq)$
 - ★ Given $B : M \mapsto \{0, 1\}$, $\exists S(b)$ PPT such that given b , S outputs random $m \in M$ s.t. $B(m) = b$
 - ★ For every message $m \in M$, E s.t. $E(f, m) = f(m)$
 - ▶ RSA: $E(f, m) = m^e \bmod N$
 - ★ Given $c \in E(f, m)$ and t_f , $D(t_f, c) = B(m)$
 - ▶ RSA: Assume B is the least significant bit of m
 - ▶ RSA: $D(t_f, c) = D(d, c) = LSB(m)$
 - ★ For every PPT A , for randomly chosen f and $c = f(m)$
 - ▶ $Pr[A(f, c) = B(m)]$ is negligible in term of k
 - ▶ RSA: Given c, e, N it is hard to find $LSB(m)$



PKE using Hard Core Predicates: Single Bit Case

- ★ Given hard core predicates B , define PKE as $(G, E, D)_B$
- ★ $G(1^k)$ outputs (f, t_i)
 - ▶ RSA: $G(1^k)$ outputs $(e, d, n = pq)$
- ★ $E(i, m)$ ($m \in \{0, 1\}$) using S finds x s.t $B(x) = m$ and outputs $f(x)$
 - ▶ RSA: To encrypt bit m choose x s.t $LSB(x) = m$ and output $x^e \bmod n$
- ★ $D_i(t_i, c)$ computes $f(x) = c$ and sets $m = B(x)$
 - ▶ RSA: To decrypt c , calculate $x = c^d \bmod n$ and set $m = LSB(x)$



General PKE using Trapdoor Hard core Predicates

- ★ Given hard core predicates B , define PKE as $(G, E, D)_B$
- ★ $G(1^k)$ outputs (f, t_i)
 - ▶ RSA: $G(1^k)$ outputs $(e, d, n = pq)$
- ★ $E(i, m)$ where $m = m_0 || m_1 || \dots || m_k$ where $m_i \in \{0, 1\}$ using S finds x_i s.t $B(x_i) = m_i$ and outputs $f(x_0) || f(x_1) \dots || f(x_k)$
 - ▶ RSA: To encrypt $m = m_0 || m_1 || \dots || m_k$ choose x_i s.t $LSB(x_i) = m_i$ and output $x_0^e || x_1^e \dots || x_k^e$
- ★ $D_i(t_i, c)$ computes $f(x_i) = c_i$ where $c = c_0 || c_1 \dots || c_k$ and sets $m_i = B(x_i)$
 - ▶ RSA: To decrypt $c = c = c_0 || c_1 \dots || c_k$, calculate $x_i = c_i^d \bmod n$ and set $m_i = LSB(x_i)$
- ★ The above construction is too inefficient but secure



Proof of Security

★ Given a collection of trapdoor permutations and a hard core predicates then PKE is indistinguishably secure

★ Proof:

- ▶ We will use the Hybrid argument
- ▶ Given k bit long m_0 and m_1 define $s_i = pre_i(m_1) || suf_{(k-i)}(m_0)$
- ▶ Note $s_0 = m_0$ and $s_k = m_1$
- ▶ Note s_i and s_{i+1} differs at most one location
- ▶ Assum PKE is not secure
- ▶ $\Rightarrow \exists A$ for inf. many k s.t. $Pr[A \text{ outputs correct bit}] > \frac{1}{2} + \frac{1}{Q(k)}$



Proof of Security

$$\star \Rightarrow \Pr[A \text{ outputs correct bit} | c \in \{E(m_o), E(m_1)\}]$$

$$\star = (1 - \Pr[A \text{ outputs 1} | c \in \{E(m_o)\}]) \cdot \Pr[c \in \{E(m_o)\}] + \Pr[A \text{ outputs 1} | c \in \{E(m_1)\}] \cdot \Pr[c \in \{E(m_1)\}]$$

► Let $P_j = \Pr[A \text{ outputs 1} | c \in \{E(s_j)\}]$

$$\star = (1 - \Pr[A \text{ outputs 1} | c \in \{E(s_o)\}]) \cdot \Pr[c \in \{E(s_o)\}] + \Pr[A \text{ outputs 1} | c \in \{E(s_k)\}] \cdot \Pr[c \in \{E(s_k)\}]$$

$$\star = \frac{1}{2}((1 - P_0) + P_k) = \frac{1}{2}(1 + P_k - P_0) = \frac{1}{2}(1 + \sum_{j=0}^{k-1} (P_{j+1} - P_j))$$

$$\star \Rightarrow \frac{1}{2}(1 + \sum_{j=0}^{k-1} (P_{j+1} - P_j)) > \frac{1}{2} + \frac{1}{Q(k)}$$

$$\star \Rightarrow (\sum_{j=0}^{k-1} (P_{j+1} - P_j)) > \frac{2}{Q(k)}$$

$$\star \Rightarrow \exists j, (P_{j+1} - P_j) > \frac{2}{Q(k) \cdot k}$$



Proof of Security

-
- ★ Now by using the $\exists j, (P_{j+1} - P_j) > \frac{2}{Q(k) \cdot k}$, we can define an adversary C that attacks trapdoor predicates problem efficiently.
 - ★ Assume that C wants to predict $B(x)$ given $c = f(x)$
 - ★ Assume s_j and s_{j+1} differs at location l
 - ★ C puts c to location l of the s_j
 - ★ if A outputs 1, C returns $s_{j+1,l}$ else s_j
 - ★ Note C predicts $B(x)$ with probability $> \frac{1}{2} + \frac{2}{Q(k) \cdot k}$



Efficient Probabilistic Encryption

★ Given hard core predicates B , define PKE as $(G, E, D)_B$

★ $G(1^k)$ outputs (f, t_i)

▶ RSA: $G(1^k)$ outputs $(e, d, n = pq)$

★ $E(i, m)$ where $|m| = l$ where m_i

1 Choose $r \in M$

2 Compute $f(r), f^2(r), \dots, f^l(r)$

3 Let $p = B(r) || B(f(r)) || \dots || B(f^{l-1}(r))$

4 Set $c = (p \oplus m, f^l(r))$

$$f(r) = r^e \pmod n$$
$$f^2(r) = f(f(r)) = (r^e)^e \pmod n$$

Secure pseudo-random generator



Efficient Probabilistic Encryption

- ★ To decrypt a ciphertext $c = (m^l, a)$, $D(t_i, c)$ runs as follows $l = |m|$
 1. Compute r from $a = f^l(r)$ using t_i
 2. Compute $p = B(r) || B(f(r)) || B(f^2(r)) || \dots || B(f^{l-1}(r))$
 3. Set $m = m^l \oplus p$

- ★ Note that $|c| = |m| + k$ where k is the security parameter
 - ▶ Compare this with the previous one $|c| = |m|.k$

- ★ RSA: $f(m) = m^e \bmod n$
 - ▶ $f^l(m) = (m^{e^l}) \bmod n$
 - ▶ $f^{-l}(c) = c^{(e^l)^{-1}} \bmod n$

- ★ Above construction is semantically secure given trapdoor functions



“More” Practical Probabilistic Encryption

- ★ Let $p = q = 7 \pmod 8$ and $n = pq$ where $|n| = k$
- ★ $f_n(x) = x^2 \pmod n$ and $B(x) = LSB(x)$
 - ▶ $LSB(x)$ is a hard core bit iff factoring is hard
- ★ We define $EPE(G, E, D)$
- ★ $G(1^k) = (n, (p, q))$, $n = pq$ where p, q defined as above
- ★ $E(n, m)$ where $l = |m|$
 - 1 Choose random quadratic residue $r \in \mathcal{Z}_n^*$
 - 2 Compute r^2, r^4, \dots, r^{2^l}
 - 3 Let $p = LSB(r) || LSB(r^2) || \dots || LSB(r^{2^{l-1}})$
 - 4 Set $c = (m \oplus p, r^{2^l} \pmod n)$



“More” Practical Probabilistic Encryption

- ★ Decryption: $D((p, q), c)$ where $c = (m^l, a)$, $l = |m|$
 - 1 Compute r s.t $r^{2^l} = a \pmod n$
 - 2 Let $p = LSB(r) || LSB(r^2) || \dots || LSB(r^{2^{l-1}})$ $r = b^2 \pmod n$
 - 3 Set $\underline{m} = m^l \oplus p$ a is QR mod n
- ★ Note that $p = 8t + 7, q = 8s + 7$ $\Rightarrow a$ is QR mod p
 $\approx a$ is QR mod q
 - ★ $J_p(a) = a^{\frac{p-1}{2}} = 1 \pmod p$ iff a is QR mod p
 - ▶ $a = a \cdot a^{\frac{p-1}{2}} = a^{4t+4} = (a^{2t+2})^2 \pmod p$ $a = a$
 - ▶ $\sqrt{a} = a^{(2t+2)} \pmod p$ $a = a^{-1}$
 - ▶ $r_p = a^{\frac{1}{2^l}} = a^{(2t+2)^l} \pmod p$ $= a \cdot a^{\frac{p-1}{2}}$
- ★ Similarity $r_q = a^{\frac{1}{2^l}} = a^{(2s+2)^l} \pmod q$ $= a \cdot a^{\frac{8t+6}{2}}$
- ★ Use CRT with r_p, r_q to calculate $r = a^{\frac{1}{2^l}} \pmod n = a^{4t+4}$
- ★ Above construction is semantically secure with comp. cost $O(k^3)$



Optimal Asymmetric Encryption Padding (OAEP)

- ★ Previously discussed schemes are secure but inefficient
- ★ Goal: Efficient PKE with provable security
- ★ RSA-OAEP is secure against chosen ciphertext Attacks under the Random Oracle assumption
- ★ Random Oracle Assumption
 - ▶ Use hash function H in your design
 - ▶ Give security proofs assuming that H is a random function
 - ▶ Replace H with some cryptographic hash function in practice
- ★ Random Oracle Assumption is not valid in general but feasible and efficient in practice

OAEP

- ★ Let k_o be chosen s.t. 2^{k_o} steps are large
- ★ Let $f : \{0, 1\}^k \mapsto \{0, 1\}^k$ is a secure trapdoor function
- ★ Let $n = k - k_o - k_1$ and r is a random k_o bit string
- ★ $G : \{0, 1\}^{k_o} \mapsto \{0, 1\}^{n+k_1}$ is pseudo-random generator
- ★ Let $H : \{0, 1\}^{n+k_1} \mapsto \{0, 1\}^{k_o}$ be hash function
- ★ $E^{G,H}(m) = f((\underbrace{m}_{n} \parallel \underbrace{0^{k_1}}_{k_1} \oplus \underbrace{G(r)}_{n+k_1}) \parallel \underbrace{r \oplus H((\cancel{x} \parallel 0^{k_1}) \oplus G(r))}_{k_o})$
- ★ $D^{G,H}(c)$
 - ★ $a \parallel b = f^{-1}(c)$ where $|a| = k - k_o, |b| = k_o$
 - ★ $r = H(a) \oplus b, m = G(r) \oplus a$
 - ★ If $su f_k(m) \neq 0^{k_1}$ reject else output $pre_n(m)$



E 1-Gamal Scheme

★ $G(1^n)$ returns a group G , generator g and random $x \in G$

★ Set public key $X = g^x$, Usually Z_p^* is used as G

★ For $m \in G$ $E_X(m)$:

- ▶ $y \xleftarrow{\$} G$, $Y = g^y$
- ▶ $C = (X^y) \cdot M$
- ▶ Return (Y, C)

$$Y = g^y \quad C = X^y \cdot M$$

★ $D_x((Y, C)) = C \cdot (Y^x)^{-1}$

$$= X^y \cdot M \cdot (Y^x)^{-1} = M$$
$$Y^x = X^y \Rightarrow (g^y)^x = (g^x)^y$$

★ El-gamal is not secure against chosen-plaintext attacks if $G = Z_p^*$

★ El-gamal is not secure against chosen-ciphertext attacks for any G

★ El-gamal is secure against chosen-plaintext attacks if DDH is satisfied for chosen G



PKE + Symmetric Encryption (SE)= Hybrid Encryption

★ Even RSA-OAEP is inefficient for encrypting large amounts of data

★ Practice Hybrid Encryption

- ▶ Use PKE to encrypt the SE key, encrypt message using SE

★ Define $\bar{E}_{pk}(M)$

- ▶ Generate random K for SE
- ▶ $C^s = E_K(M)$ and $C^a = E_{pk}(K)$
- ▶ Return (C^a, C^s)

PKC
SE

★ Define $\bar{D}_{pr}(C)$

- ▶ Let $C = (C^a, C^s)$
- ▶ $\hat{K} = D_{pr}(C^a)$
- ▶ $M = D_{\hat{K}}(C^s)$



Hybrid Encryption

- ★ If PKE and SE are secure against chosen plain text attacks then Hybrid Encryption is secure against chosen-plaintext attacks
- ★ If PKE and SE are secure against chosen-ciphertext attacks then Hybrid Encryption is secure against chosen-ciphertext attacks
- ★ Examples:
 - 1 $E^1(M) = \{K = H(r), \text{return } (r^e \bmod n, AES - CBC_K(M))\}$
 - 2 $E^2(M) = (r^e \bmod n, G(r) \oplus M)$ for some pseudo-random generator G
 - 3 $E^3(M) = (r^e \bmod n, G(r) \oplus M, H(r||M))$ for some pseudo-random generator G and hash function H
- ★ E_1, E_2 are secure against CPA and E_3 is secure against CCA under random oracle assumption