Public Key Encryption

CS 276: Introduction to Cryptography

Sanjam Garg

March 4, 2026

Overview

1. Motivation and Definitions
2. Trapdoor Functions
3. PKE from Trapdoor Functions
4. PKE from CDH
5. Improving Efficiency (ROM)
6. Fujisaki-Okamoto

Public Key Encryption: Motivation

Goal

Two parties communicate with **privacy** against an eavesdropper who sees all communication. Only the holder of the secret key can decrypt.

Public Key Encryption: Motivation

Goal

Two parties communicate with **privacy** against an eavesdropper who sees all communication. Only the holder of the secret key can decrypt.

Why not one-way functions?

Digital signatures and symmetric-key encryption can be built from OWFs. **Lower bounds** suggest PKE *cannot* be built from OWFs alone. We need **stronger assumptions** (e.g. trapdoor functions, CDH) that imply OWFs.

PKE Scheme and Correctness

Definition 1 (Public key encryption)

A **public key encryption scheme** is a tuple $(\mathsf{Gen}, \mathsf{Enc}, \mathsf{Dec})$:

- $\mathsf{Gen}(1^n) \to (\mathsf{pk}, \mathsf{sk})$: outputs public key and secret key.
- $\mathsf{Enc}(\mathsf{pk}, m) \to c$: (probabilistic) encryption of m.
- $\mathsf{Dec}(\mathsf{sk}, c) \to m$ or $\perp$: decryption.

PKE Scheme and Correctness

Definition 1 (Public key encryption)

A **public key encryption scheme** is a tuple $(\mathsf{Gen}, \mathsf{Enc}, \mathsf{Dec})$:

- $\mathsf{Gen}(1^n) \to (\mathsf{pk}, \mathsf{sk})$: outputs public key and secret key.
- $\mathsf{Enc}(\mathsf{pk}, m) \to c$: (probabilistic) encryption of m.
- $\mathsf{Dec}(\mathsf{sk}, c) \to m$ or $\perp$: decryption.

Definition 2 (Correctness)

For all n, $(\mathsf{pk}, \mathsf{sk}) \leftarrow \mathsf{Gen}(1^n)$, $m \in \{0,1\}^*$:

$$\Pr[\mathsf{Dec}(\mathsf{sk}, \mathsf{Enc}(\mathsf{pk}, m)) = m] = 1$$

(Can be relaxed to negligible decryption error.)

IND-CPA Security

Game $\mathrm{Exp}^{\mathrm{CPA}}_{\mathcal{A}}(n)$

1. $(\mathsf{pk}, \mathsf{sk}) \leftarrow \mathrm{Gen}(1^n)$; $b \leftarrow \{0,1\}$.
2. $\mathcal{A}(\mathsf{pk})$ outputs (m_0, m_1, st).
3. $c^* \leftarrow \mathrm{Enc}(\mathsf{pk}, m_b)$.
4. $\mathcal{A}(c^*, \mathsf{st})$ outputs b'.
5. **Output** 1 if $b = b'$, else 0.

IND-CPA Security

Game $\mathsf{Exp}^{\mathsf{CPA}}_{\mathcal{A}}(n)$

1. $(\mathsf{pk}, \mathsf{sk}) \leftarrow \mathsf{Gen}(1^n)$; $b \leftarrow \{0,1\}$.
2. $\mathcal{A}(\mathsf{pk})$ outputs (m_0, m_1, st).
3. $c^* \leftarrow \mathsf{Enc}(\mathsf{pk}, m_b)$.
4. $\mathcal{A}(c^*, \mathsf{st})$ outputs b'.
5. **Output** 1 if $b = b'$, else 0.

Definition 3 (IND-CPA)

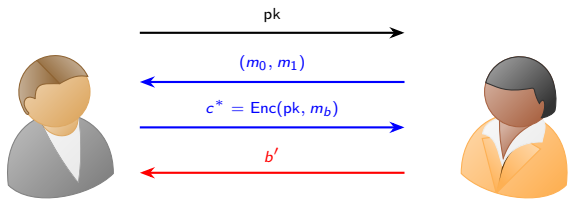
The scheme is **IND-CPA secure** if for all non-uniform PPT $\mathcal{A}$:

$$\left| \Pr[\mathsf{Exp}^{\mathsf{CPA}}_{\mathcal{A}}(n) = 1] - \frac{1}{2} \right| = \mathsf{negl}(n).$$

IND-CPA: Challenger vs Adversary

$$\mathrm{Exp}_{\mathcal{A}}^{\mathrm{CPA}}(n)$$

sample $(\mathsf{pk}, \mathsf{sk})$, $b \leftarrow \{0,1\}$



output 1 if $b' = b$; else 0

IND-CCA Security

Game $\mathrm{Exp}_{\mathcal{A}}^{\mathrm{CCA}}(n)$

Same as CPA, but $\mathcal{A}$ gets a **decryption oracle** $\mathrm{Dec}(\mathrm{sk}, \cdot)$ in both phases. After receiving c^*, the oracle returns $\perp$ when queried on c^*.

IND-CCA Security

Game $\mathrm{Exp}_{\mathcal{A}}^{\mathrm{CCA}}(n)$

Same as CPA, but $\mathcal{A}$ gets a **decryption oracle** $\mathrm{Dec}(\mathrm{sk}, \cdot)$ in both phases. After receiving c^*, the oracle returns $\perp$ when queried on c^*.

Definition 4 (IND-CCA)

The scheme is **IND-CCA secure** if for all non-uniform PPT $\mathcal{A}$:

$$\left| \Pr[\mathrm{Exp}_{\mathcal{A}}^{\mathrm{CCA}}(n) = 1] - \frac{1}{2} \right| = \mathrm{negl}(n).$$

Stronger than CPA; needed for secure messaging (e.g. non-malleability).

Trapdoor Functions

Definition 5 (Trapdoor function)

A **trapdoor function** is a tuple $(\mathrm{Gen}, D, f, f^{-1})$:

- $\mathrm{Gen}(1^n) \to (s, t)$: index s, trapdoor t.
- $D(s)$: description of domain.
- $f(s, x) \to y$: evaluate on $x \in D(s)$.
- $f^{-1}(t, y) \to x$: invert using trapdoor.

Trapdoor Functions

Definition 5 (Trapdoor function)

A **trapdoor function** is a tuple $(\mathsf{Gen}, D, f, f^{-1})$:

- $\mathsf{Gen}(1^n) \to (s, t)$: index s, trapdoor t.
- $D(s)$: description of domain.
- $f(s, x) \to y$: evaluate on $x \in D(s)$.
- $f^{-1}(t, y) \to x$: invert using trapdoor.

Properties

- **Correctness**: $f^{-1}(t, f(s, x)) = x$ (w.o.p.).
- **One-wayness**: Given (s, y) with $y = f(s, x)$ for random x, no PPT can find x with non-negligible probability.

PKE from TDF: Construction

Construction

Let $(\mathsf{TDF.Gen}, D, f, f^{-1})$ be a TDF. Assume a **hard-core predicate** $\mathsf{HC}(x, r)$ (e.g. from OWP).

- $\mathsf{Gen}(1^n)$: $(s, t) \leftarrow \mathsf{TDF.Gen}(1^n)$; $\mathsf{pk} = s$, $\mathsf{sk} = t$.
- $\mathsf{Enc}(\mathsf{pk}, m)$: $x \leftarrow D(s)$, $y \leftarrow f(s, x)$, $r \leftarrow \{0,1\}^{|x|}$; output $c = (y, r, m \oplus \mathsf{HC}(x, r))$.
- $\mathsf{Dec}(\mathsf{sk}, c)$: parse $c = (y, r, z)$; $x \leftarrow f^{-1}(t, y)$; output $m = z \oplus \mathsf{HC}(x, r)$.

PKE from TDF: Construction

Construction

Let $(\mathsf{TDF.Gen}, D, f, f^{-1})$ be a TDF. Assume a **hard-core predicate** $\mathsf{HC}(x, r)$ (e.g. from OWP).

- $\mathsf{Gen}(1^n)$: $(s, t) \leftarrow \mathsf{TDF.Gen}(1^n)$; $\mathsf{pk} = s$, $\mathsf{sk} = t$.
- $\mathsf{Enc}(\mathsf{pk}, m)$: $x \leftarrow D(s)$, $y \leftarrow f(s, x)$, $r \leftarrow \{0,1\}^{|x|}$; output $c = (y, r, m \oplus \mathsf{HC}(x, r))$.
- $\mathsf{Dec}(\mathsf{sk}, c)$: parse $c = (y, r, z)$; $x \leftarrow f^{-1}(t, y)$; output $m = z \oplus \mathsf{HC}(x, r)$.

Proof idea

Message space $\{0,1\}$. If $\mathcal{A}$ distinguishes encryptions of 0 vs 1, we can use it to distinguish $\mathsf{HC}(x, r)$ from random and thus invert the TDF (via hard-core theorem).

PKE from CDH

Setting

$\mathbb{G}$ prime order p, generator g. CDH: given (g, g^a, g^b), computing g^{ab} is hard.

PKE from CDH

Setting

$\mathbb{G}$ prime order p, generator g. CDH: given (g, g^a, g^b), computing g^{ab} is hard.

Construction

- $\mathsf{Gen}(1^n)$: $\mathsf{sk} \leftarrow \mathbb{Z}_p$, $\mathsf{pk} \leftarrow g^{\mathsf{sk}}$; output $(\mathsf{pk}, \mathsf{sk})$.
- $\mathsf{Enc}(\mathsf{pk}, m)$: $\alpha \leftarrow \mathbb{Z}_p$, $y \leftarrow g^{\alpha}$, $k \leftarrow \mathsf{pk}^{\alpha}$; $r \leftarrow \{0,1\}^{|B|}$; output $c = (y, r, m \oplus \mathsf{HC}(k, r))$.
- $\mathsf{Dec}(\mathsf{sk}, c)$: parse $c = (y, r, z)$; $k \leftarrow y^{\mathsf{sk}}$; output $m = z \oplus \mathsf{HC}(k, r)$.

Proof: same idea as TDF-based; reduce to CDH (distinguishing g^{ab} from random via HC).

Efficiency Issue

Problem

TDF/CDH construction encrypts **one bit**; ciphertext and cost are $O(\text{poly}(n))$ per bit. For long messages: $O(M \cdot \text{poly}(n))$.

Efficiency Issue

Problem

TDF/CDH construction encrypts **one bit**; ciphertext and cost are $O(\text{poly}(n))$ per bit. For long messages: $O(M \cdot \text{poly}(n))$.

Idea

Replace hard-core bit with a **random oracle** H to derive a **symmetric key** $k = H(x)$. Encrypt long m with a symmetric scheme under k. Ciphertext (y, z) with $z = \text{Sym.Enc}(H(x), m)$; one TDF evaluation per message.

PKE from TDF in ROM: Construction

Construction

- $\mathsf{Gen}(1^n)$: $(s, t) \leftarrow \mathsf{TDF.Gen}(1^n)$; $\mathsf{pk} = s$, $\mathsf{sk} = t$.
- $\mathsf{Enc}(\mathsf{pk}, m)$: $x \leftarrow D(s)$, $y \leftarrow f(s, x)$; $z \leftarrow \mathsf{Sym.Enc}(H(x), m)$; output $c = (y, z)$.
- $\mathsf{Dec}(\mathsf{sk}, c)$: parse $c = (y, z)$; $x \leftarrow f^{-1}(t, y)$; output $m \leftarrow \mathsf{Sym.Dec}(H(x), z)$.

PKE from TDF in ROM: Construction

Construction

- $\mathsf{Gen}(1^n)$: $(s, t) \leftarrow \mathsf{TDF.Gen}(1^n)$; $\mathsf{pk} = s$, $\mathsf{sk} = t$.
- $\mathsf{Enc}(\mathsf{pk}, m)$: $x \leftarrow D(s)$, $y \leftarrow f(s, x)$; $z \leftarrow \mathsf{Sym.Enc}(H(x), m)$; output $c = (y, z)$.
- $\mathsf{Dec}(\mathsf{sk}, c)$: parse $c = (y, z)$; $x \leftarrow f^{-1}(t, y)$; output $m \leftarrow \mathsf{Sym.Dec}(H(x), z)$.

Theorem 6

This scheme is IND-CPA secure in the **random oracle model** *assuming the TDF is one-way and the symmetric scheme is secure.*

Reduction (TDF + ROM): Sketch

$\mathcal{B}$ gets (s, y^*) (TDF challenge); goal: compute $x^* = f^{-1}(t, y^*)$.

1. Give $\mathsf{pk} = s$ to $\mathcal{A}$. Simulate H: for query q, if $f(s, q) = y^*$ then set $H(q) = k^*$ (random key used in challenge) and later output q as preimage. Else sample $H(q)$ at random and store.
2. $\mathcal{A}$ outputs (m_0, m_1). $\mathcal{B}$ picks $b \leftarrow \{0,1\}$, $k^* \leftarrow \{0,1\}^\ell$, $z^* \leftarrow \mathsf{Sym.Enc}(k^*, m_b)$, gives $c^* = (y^*, z^*)$.
3. If $\mathcal{A}$ never queries H on x^* with $f(s, x^*) = y^*$, then it never sees k^* but must distinguish $\mathsf{Sym.Enc}(k^*, m_0)$ from $\mathsf{Sym.Enc}(k^*, m_1)$ $\Rightarrow$ breaks symmetric security. So w.n.p. $\mathcal{A}$ queries $H(x^*)$; $\mathcal{B}$ outputs x^*. $\square$

Fujisaki-Okamoto and CCA

Goal

Convert **CPA-secure** (or one-way) PKE + one-time symmetric encryption into **IND-CCA secure** hybrid PKE in the **random oracle model** (Fujisaki–Okamoto, 1999; refined 2013).

Idea (high level)

- **Encrypt**: sample random σ; $K = G(\sigma)$, $c \leftarrow \mathrm{Sym.Enc}(K, m)$, $r = H(\sigma, c)$, $e \leftarrow \mathrm{Enc}(\mathrm{pk}, \sigma; r)$. Output (e, c). Two random oracles G, H; r binds e to (σ, c).
- **Decrypt**: recover σ' from e; recompute $K' = G(\sigma')$, $r' = H(\sigma', c)$; check **re-encryption** $e = \mathrm{Enc}(\mathrm{pk}, \sigma'; r')$; if yes output $\mathrm{Sym.Dec}(K', c)$, else $\perp$.

Any tampering with (e, c) fails the re-encryption check. Kyber and other NIST PKE use FO-style variants (e.g. HHK17, QROM-safe).