Today:

RSA: Public key encryption

Digital signature scheme.

Diffie & Hellman's vision: ('New Directions in Cryptography', 1976)

Construct public-key cryptography from trapdoor functions

Def: A trapdoor permutation family \( \{ f_k \} \) is a family of permutations \( f_k : D_k \rightarrow D_k \) s.t.

For key generation alg \( \text{Gen} \) s.t on input \( 1^t \) outputs \( (k, z) \) s.t.

\[ \begin{align*}
&\text{trapdoor} \quad (k, x \in D_k) \quad \text{easy} \rightarrow f_k(x) \\
&\quad (z, f_k(x)) \quad \text{easy} \rightarrow x \\
&\quad (k, f_k(x)) \quad \text{hard} \rightarrow x
\end{align*} \]

Given such family one can construct a (det.) public key enc. scheme \( (\text{Gen}, \text{Enc}, \text{Dec}) \):

\[ \text{Gen}(1^t) : \text{Generate} \ (PK, SK) = (k, z) \text{ by running} \]

...
key gen alg of the trapdoor family.

\[ \text{Enc}(k, m) = f_k(m) \]

\[ \text{Dec}(z, c) = f_k^{-1}(f_k(m)) \]

This is clearly not CPA secure!

Was suggested as a heuristic (w.o. formal security def).

Moreover, using such a family one can digitally sign!

**Def.** A digital signature scheme consists of PPT alg \( (\text{Gen}, \text{Sign}, \text{Ver}) \):

- \( \text{Gen}(1^n) \) generates \( (\text{PK}, \text{SK}) \) \((\text{PK} \text{ often denoted by } VK\text{ for verification key})\)

- \( \text{Sign}(sk, m) \) outputs signature \( \sigma \)

- \( \text{Ver}(pk, m, \sigma) \) outputs 0/1.

**Correctness:** \[ \Pr \left[ \text{Ver}(pk, m, \text{Sign}(sk, m)) = 1 \right] = 1 \]

\((pk, sk) \leftarrow \text{Gen}(1^n)\)
Signature scheme based on trapdoor permutations:

Gen \( (1^k) \) : generates \( (PK, SK) = (k, \mathcal{L}) \) corresponding to the trapdoor family

\[
\text{Sign} (k, m) = f_k^{-1}(m)
\]

\[
\text{Ver} (k, m, \mathcal{L}) = 1 \text{ iff } f_k(\mathcal{L}) = m.
\]

(We will talk about security later).

**RSA (RSA 1977)**

**First (and only) construction of trapdoor permutation family:**

**Key Gen \( (1^k) \) :** Choose at random primes \( p, q \in \mathbb{Z}^\star \)

Let \( n = p \cdot q \). Choose e s.t. \( \gcd(e, \varphi(n)) = 1 \)

\[
k = (n, e)
\]

\[
d = e^{-1} \mod \varphi(n)
\]

\[
\mathcal{L} = (n, d)
\]

\[
f_k : \mathbb{Z}_n^\star \rightarrow \mathbb{Z}_n^\star
\]

\[
f_k (x) = x^e \mod n.
\]

\[
f_k^{-1} (y) = y^d \mod n
\]
Note: \((x^e)^d \mod n = x\) \(\checkmark\)

For efficiency, often \(e=3\).

Need: \(\gcd(3, (p-1)(q-1)) = 1\) \((p,q = 2 \mod 3)\)

Note: \(d = e^{-1} \mod \lambda(n)\) is computed using the extended GCD alg.

\[d = e \left( \frac{1}{\lambda(n)} \right) \mod \lambda(n)\]

Given \(a, b\) outputs \(x, y \in \mathbb{Z}\) s.t. \(ax + by = \gcd(a, b)\).

If \(\gcd(a, b) = 1\) then

\(x = a^{-1} \mod b\)
\(y = b^{-1} \mod a\).

Example: \(a=5, b=7\)

\[5 = 1 \cdot 5 + 0 \cdot 7\]
\[7 = 0 \cdot 5 + 1 \cdot 7\]
\[2 = (-1) \cdot 5 + 1 \cdot 7\]
\[1 = 3 \cdot 5 + (-2) \cdot 7 \Rightarrow 3 = 5^{-1} \mod 7\]
Claim: \{fu\} is a trapdoor permutation family assuming

**RSA Assumption**

\[ n, e, x \mod n \xrightarrow{\text{HARD}} x \]

\[ n = p \cdot q \]

\[ x \leftarrow \mathbb{Z}^*_n. \]

RSA scheme based on trapdoor permutation
(presented above) is called "textbook RSA".

**CPA secure version (in Random Oracle Model)**

\[ \text{Gen}(\mathbb{F}) : \text{Same} \quad \mathcal{V} \xrightarrow{\text{secure}} (\mathcal{P}, \mathcal{E}) \]

\[ \text{Enc}(\mathcal{P}, m) : \text{Choose} \quad r \leftarrow \mathbb{Z}^*_n \quad \text{output} \]

\[ (r^e \mod n, \mathcal{H}(r \oplus m)) \]

\[ \text{Dec}(\mathcal{E}, (c_1, c_2)) : \text{Compute} \quad \mathcal{R} = c_1^d \mod n \]

Output \[ m = \mathcal{H}(r) \oplus c_2. \]
Security: The only way to distinguish 
\[(PK, \text{Enc}(PK, m)) \text{ from random } \]
\[(r, \text{modn}, H(r)||m)\]
on input t, but this breaks RSA assumption.

Making RSA CCA secure (in the ROM)

Def: An encryption scheme \((\text{Gen}, \text{Enc}, \text{Dec})\) is CCA secure

\[
\text{if } \forall \text{ PPT adv } A \ (PK) \text{ that chooses } m_0, m_1 \in \mathbb{Z}_n \text{ st. } |m_0| = |m_1|,
\]

\[
\Pr[A^{(PK, \text{Enc}(PK, m_0)) = b}] = \frac{1}{2} + \text{negl}(\lambda)
\]

\[(PK, SK) \leftarrow \text{Gen}(1^\lambda) \]
\[b \leftarrow \{0, 1\}\]
randomness of Enc.
assuming A does not send \(c_b\) as an oracle query.

Note: RSA & El-Gamal are not CCA secure
Making RSA CCA secure.

\[ \text{Gen}(1^n) : \text{Same} \]

\[ \text{Enc}((n,e), m) : x^e \mod n \quad x = \text{Pad}_r(m) \]

where \( \text{Pad}_r(m) \xrightarrow{\text{easy}} m \)

\[ \text{Dec}((n,d), C) : \text{Compute} \quad x = C^d \mod n \]

\[ \text{Compute} \quad \text{Pad}^{-1}(x) = m. \]

**PAD**: Optimal Asymmetric Encryption Padding (OAEP)

[ Bellare-Rogaway 94]

**Intuition**: To make dec oracle useless let

\[ \text{pad}(m) = (r, G(r) \oplus (m \cdot 0^{k_1})) \]

Intuitively, to generate \( x^e \) for valid encoding \( x \) must know \( r \), but may not know \( m \). To fix this, replace \( r \) with \( r \oplus H(x_0) \).

\[ x_0 = G(r) \oplus (m \cdot 0^{k_1}) \]

\[ x_1 = r \oplus H(x_0) \]

**CPA security** (in the ROM)

Partial information about \( m \) is leaked only if

\( r \) is leaked, which happens only if \( x_0 \) is leaked entirely.

\( \Rightarrow (x_0, x_1) \) leaked entirely.
CCA security follows since $A^{\text{dec}(sk, \cdot)}( (x_0 \cdot x_1)^e \mod n)$ cannot use its decryption oracle on C-s that are a func. of $(x_0 \cdot x_1)^e \mod n$.

Suppose $A$ generates $(x_0 \cdot x_1)^e \mod n$ without knowing $(x_0 \cdot x_1)^e$.

$\Rightarrow$ A does not know $x_i \oplus H(x_i) \Rightarrow \text{w.h.p. } x_0 \neq G(r) \oplus (\cdot \cdot \cdot 0^{k_1}) x_i \oplus H(x_i)$. 