RSA

The first PKE (1978); it's a kind of non-interactive key exchange.

Alice		**Bob**
	$\xleftarrow{\quad pk \quad}$	sk
$c \leftarrow Enc(pk, k)$	$\xrightarrow{\quad c \quad}$	$k = Dec(sk, c)$
$k \leftarrow U_\lambda$	

The "naive" idea: let $n = p \cdot q$, where p, q are λ-bit primes. Then, $pk = n$ and $sk = (p, q)$.

Encryption and decryption are based on $(\mathbb{Z}_n^*, \cdot)$;

There are 2 exponents:

- Encryption exponent e (e.g. $e = 3$)
- Decryption exponent d

$$\text{s.t.} \quad d \cdot e \equiv 1 \mod \varphi(n)$$

$$\varphi(n) = \# \mathbb{Z}_n^* = (p-1)(q-1)$$

$$PK = (n, e) \quad ; \quad SK = (n, d)$$

Observation by RSA: We can use Euler's THM to define a so called TRAPDOOR PERMUTATION:

$$\mathrm{Enc}(pk, m) = f_{e,n}(m) = m^e \bmod n = c$$

$$m \in \mathbb{Z}_n^*$$

$$\mathrm{Dec}(sk, c) = c^d \bmod n = f^{-1}_{d,n}(c)$$

$$= (m^e)^d = m^{ed} = m^{t \cdot \varphi(n) + 1}$$

$$= \underbrace{\left(m^{\varphi(n)}\right)^t}_{\equiv 1 \bmod n} \cdot m \bmod n$$

CPA secure? Of course not as it is

DETERMINISTIC.

In practice, we use RSA using the so-called PKCS standards:

-) CPA security # 1.5

-) CCA security # 2.0

<u>CPA</u>: $\hat{m} = (r \| m)$ for $r \leftarrow \{0,1\}^{\ell(\lambda)}$

Then, do the same $c = (\hat{m})^e \bmod n$

$\hat{m} = Dec(sk, c)$ and we can recover m by discarding r. It's a standard:

– 1 byte fixed; This makes sure

The modular reduction takes place.

- 1 byte encode the "mode": encryption or signatures.
- The r part; at least 8 bytes.
- Then m.

<u>CPA security</u>? Here is what we know:

- First, $\ell(\lambda)$ must be large enough ($\omega(\log \lambda)$).
- On the other hand, we can prove CPA

security for $m \in \{0,1\}$. From what assumption? NOT FACTORING, but under the so-called RSA assumption.

- For other ranges we don't know.

RSA assumption? Of course, FACTORING must be hard. Also, computing $\varphi(n)$ should be hard; but this is equivalent to factoring n:

$$\underbrace{p \cdot q}_{n} - \underbrace{(p-1)(q-1)}_{\varphi(n)} + 1 = pq - pq + p + q - 1 + 1$$

$$= p + q$$

Then, given $\varphi(n)$ we can compute

$$\begin{cases} s = n - \varphi(n) + 1 = p + q \\ p \cdot q = n \end{cases}$$

$\Rightarrow$ can compute p, q.

The RSA assumption is simply the fact that $f_{n,e}(m) = m^e \bmod n$ is a ONE-WAY FUNCTION. In fact, this is not really precise, because it's more than that: it's a TRAPDOOR PERMUTATION

TDP : $(\mathsf{Gen}, f, f^{-1})$ s.t.

$$(pk, sk) \leftarrow \mathsf{Gen}(1^\lambda)$$

$$y = f_{pk}(x) \; ; \; x = f^{-1}_{sk}(y)$$

$$\begin{array}{ccc}
\mathcal{A} & & \mathcal{C} \\
pk & \xleftarrow{\;pk,\; f(x)=y\;} & x \leftarrow \mathcal{X} \\
 & \xrightarrow{\;x'\;} & x' \stackrel{?}{=} x
\end{array}$$

$\mathcal{A}$ $\qquad\qquad\qquad\qquad$ $\mathcal{C}$ $\qquad\qquad$ $\underline{\mathrm{GenRSA}(1^\lambda)}$

$$\xleftarrow{\quad pk,\ y \quad}$$

$$pk = (n, e)$$
$$sk = (n, d)$$

$$\xrightarrow{\quad x' \quad}$$

$$x \leftarrow \mathbb{Z}_n^*$$

$$y = x^e \bmod n$$

$$x' \stackrel{?}{=} x$$

RSA $\Rightarrow$ FACTORING $\quad$ Trivial.

FACTORING $\Rightarrow$ RSA $\quad$???!

Q: Can we do CPA PKE from FACTORING, for long messages? As efficient as RSA?

The answer is yes. But none of these schemes is a real standard so we won't cover it.

<u>CCA security</u>? There was a real-world CCA against PKCS # 1.5. Based on "partial decryption oracle" that just tells if a mauled $\tilde{c}$ contains $\tilde{m}$ that is padded correctly.

This is why there is PKCS # 2.0.

It's a more complex padding that can be proven CCA secure for λ-bit msg but under strong assumptions (+ RSA).

<u>OAEP</u> : $m \in \{0,1\}^{\ell}$; $m' = m \| 0^{\lambda_1}$

for $\lambda_1 = \Theta(\lambda)$. $r \leftarrow \{0,1\}^{\lambda_0}$:

$$\left.\begin{aligned} s &= m' \oplus G(r) \in \{0,1\}^{\ell+\lambda_1} \\ t &= r \oplus H(s) \in \{0,1\}^{\lambda_0} \end{aligned}\right\} \Rightarrow \hat{m} = s \| t$$

$$c = (s \| t)^e \bmod n.$$

In practice, λ_1, λ_0 are constants and ℓ can be around 256 bits.

RSA assumption + something about G, H.

(G, H are RANDOM ORACLES.)

In theory: TDP $\Rightarrow$ PKE (at least CPA secure). Recall: h is HARD-CORE for f if:

$$(f(x), h(x)) \approx_c (f(x), b)$$

$$b \leftarrow \{0,1\}$$
$$x \leftarrow \{0,1\}^n$$

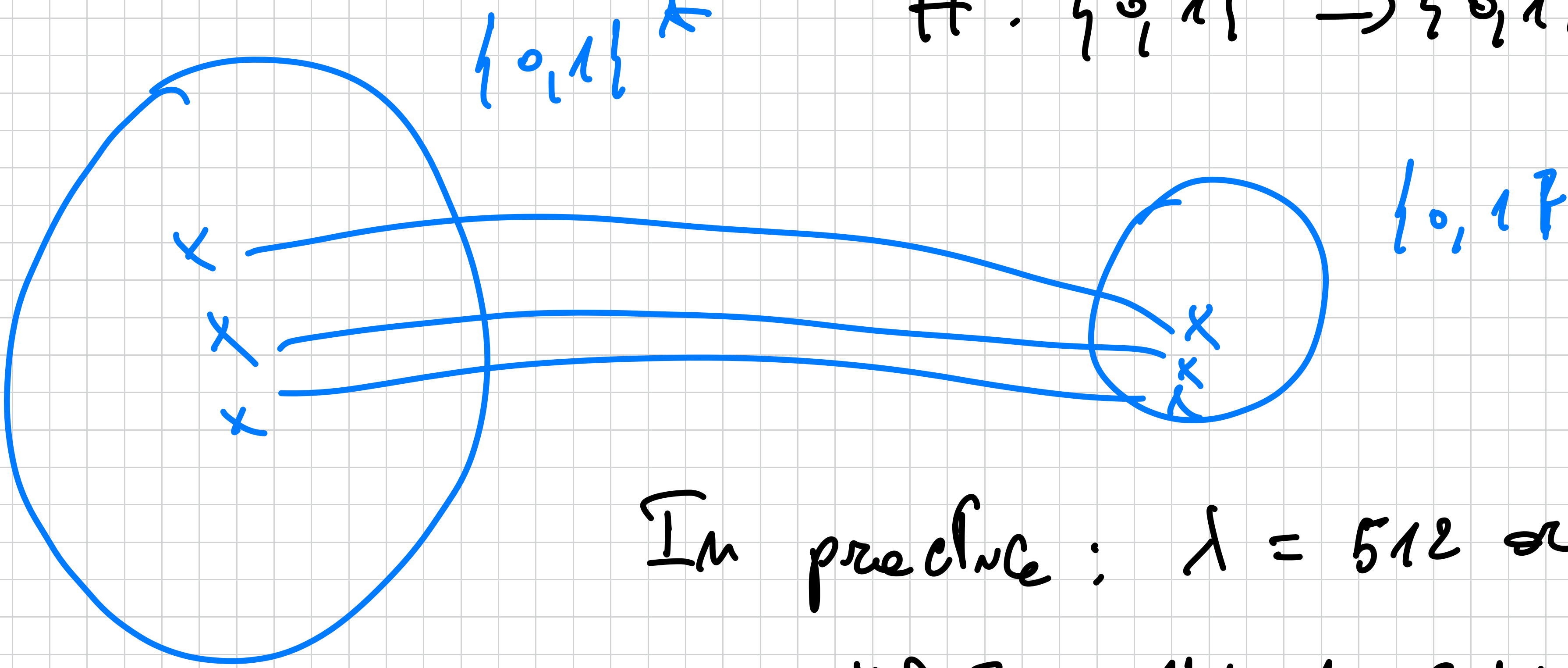
$$\mathrm{Enc}(pk, m) = \left(f_{pk}(r),\ h(r) \oplus \underbrace{m} \right)$$

$$r \leftarrow \{0,1\}^n \qquad \} \text{ 1 bit}$$

Moreover: 1-bit PKE $\Rightarrow$ poly(λ)-bit PKE.

HASH FUNCTIONS

$$H : \{0,1\}^* \rightarrow \{0,1\}^\lambda$$



In practice: $\lambda = 512$ or 256

MD5, SHA-1, SHA-2, SHA-3

$$\mathcal{H} = \{ H_s : \{0,1\}^* \rightarrow \{0,1\}^\lambda \}_{s \in \{0,1\}^\lambda}$$

The main security property: COLLISION RESISTANCE.

$$\mathrm{GAME}^{col}_{\mathcal{H},\mathcal{A}}(\lambda)$$

$$\begin{tikzcd}
\mathcal{A} \arrow[rr, bend left, leftarrow, "s"] \arrow[rr, bend right, "x, x'"'] & & \mathcal{C}
\end{tikzcd}$$

$s \leftarrow \{0,1\}^\lambda$

WIN (output 1):

$H_s(x) = H_s(x')$

$x \neq x'$

$\forall$ PPT $\mathcal{A}$:

$$\Pr\left[\mathrm{GAME}^{col}_{\mathcal{H},\mathcal{A}}(\lambda) = 1\right] \leq \mathrm{negl}(\lambda)$$

Why is there a seed? Can't we have a single hash function that is collision resistant?

We can't. Because once we fix H, there exist x, x' that are a collision and the following $A_{x,x'}$ breaks coll. res. in poly-time:

$A_{x,x'}$: Output x, x'.

General paradigm for constructing hash functions:

- First design compression function, say

$$h_s : \{0,1\}^{2\lambda} \to \{0,1\}^{\lambda}$$

$$\left(\text{or even } \{0,1\}^{\lambda+1} \to \{0,1\}^{\lambda}\right).$$

- Then, amplify this to domain $\{0,1\}^*$.

Real world constructions faithfully follow step 2, but heuristically implement step 1.