8. Public-Key Encryption

Asymmetric encryption.

Idea by Diffie & Hellman 1976.

Earlier, but unpublished paper by James Ellis (1970).

Paper released by British government 1997.

"The possibility of non-secret encryption".

- All users share the same e, d.
- Each user has a pair of keys (K, L) such that

$$d(e(M, K), L) = M \quad \forall M \in \mathcal{M}$$

K is made public, L is private (secret).

- Requirements
  - (i) $C = e(M, K)$ "easy" given M and K
    solving for M "infeasible" given C and K.
  - (ii) $M = d(C, L)$ "easy" given C, L

Hence: $f_K(M) = e(M, K)$ is a one-way function with "trapdoor" L.

- Further requirements:
  - (i) (K, L) easy to generate.
  - (ii) There are sufficiently many (K, L), exhaustive search is impossible.

8.1. The RSA Cryptosystem

(Rivest, Shamir, Adleman, 1977)

Prior invented by Clifford Cox (1973), released in 1997 by British Government.

RSA-System:

(i) Choose $p \neq q$ (large prime)
compute $n = p \cdot q$

(ii) Choose $d \in \mathbb{Z}^*_{(p-1)(q-1)}$, i.e., $\gcd(d, (p-1)(q-1)) = 1$
Compute $e = d^{-1} \bmod (p-1)(q-1)$
Remark $\varphi(n) = (p-1)(q-1)$

(iii) Public key: $(e, n) = (d^{-1}, n)$, private key: d

(iv) Message $m \in \{1, \dots, n-1\}$
Encryption: $c = m^e \bmod n$
Decryption: $m = c^d \bmod n$

Question: 1.) Is m the original message?
2.) Security
3.) Implementation

Proposition 8.1. $p \neq q$ prime, $x, y \in \mathbb{N}$.
$x \equiv y \pmod p$ and $x \equiv y \pmod q \iff x \equiv y \pmod{pq}$.

Proof.
$p \mid (x-y)$ and $q \mid (x-y) \Rightarrow p \cdot q \mid (x-y)$
$\Leftarrow$ since, $p \neq q$ prime.

Prop. 8.2. Let $p \neq q$ prime, $n = p \cdot q$, $d, d^{-1} \in \mathbb{Z}^*_{\varphi(n)}$.
$0 \leq m < n$, $c = m^{d^{-1}} \bmod n$. Then
$m = c^d \bmod n$.

Proof. $d^{-1}d \equiv 1 \pmod{(p-1)(q-1)}$
$\Rightarrow \exists\, t \in \mathbb{N}: d^{-1}d = t(p-1)(q-1) + 1$

(i) $\gcd(m, p) = 1$:
$$(m^{d^{-1}})^d \equiv m^{d^{-1}d} = m^{t(p-1)(q-1)} \cdot m$$
$$\equiv (m^{p-1})^{t(q-1)} \cdot m \pmod p$$
$$\overset{\text{Fermat}}{\equiv} 1 \cdot m \equiv m \pmod p$$

(ii) $\gcd(m, p) = p$
$p \mid m$, i.e., $m \equiv 0 \pmod p \Rightarrow m^{d^{-1}d} \equiv 0 \equiv m \pmod p$

Analogously: $(m^{d^{-1}})^d \equiv m \pmod q$.
Using Prop. 8.1: $(m^{d^{-1}})^d \equiv m \bmod (p \cdot q)$. ∎

Security of RSA:

Chosen plaintext is most relevant.

Known: d^{-1}, n, arbitrary many pairs (m, c).

a) Factoring n, computing $\varphi(n) = (p-1)(q-1)$
computing $(d^{-1})^{-1} \bmod \varphi(n) = d$
But factoring infeasible.

b) Computing square roots modulo n
allows factoring.

Prop. 8.3 $n = p \cdot q$, $p \neq q$ prime, x is a nontrivial solution of $x^2 \equiv 1 \pmod{n}$, i.e., $x \not\equiv \pm 1 \pmod{n}$
Then $\gcd(x+1, n) \in \{p, q\}$.

Proof. Ex!

Hence: Computing square roots is no easier than factoring.

c) Computing $\varphi(n)$ without factoring.
Any eff. alg. for comp. $\varphi(n)$ allows for an eff. alg. to factor.

Because:

Let $n = p \cdot q$ (p, q prime, unknown)

$\varphi(n) = (p-1)(q-1)$ (known)

$$\varphi(n) = (p-1)(q-1) = p \cdot q - p - q + 1$$

$$\Leftrightarrow \quad p + q = n - \varphi(n) + 1 \qquad (1)$$

$$(p-q)^2 - (p+q)^2 = -4pq$$

$$\Leftrightarrow \quad (p-q)^2 = (p+q)^2 - 4n \qquad (2)$$

$$q = \frac{1}{2}\left((p+q) - (p-q)\right) \qquad (3)$$

(1) yields $(p+q)$, from (2) obtain $p-q$, q follows from (3). ∎

Computing $\varphi(n)$ is no easier than factoring.

d) Computing $(d^{-1})^{-1} \bmod n$ (without knowing $\varphi(n)$)

Prop. 8.4. Let $n = p \cdot q$, p, q prime. Any eff. alg. for comp. $b^{-1} \bmod \varphi(n)$ leads to an eff. prob. alg. for factoring n with error prob. $< \frac{1}{2}$. ⊥

Proof. Stinson p. 139-141

Hence, comp. $b^{-1} \bmod \varphi(n)$ is no easier than factoring.

Remarks:

a) If d is known, n can be eff. factored (see Prg. 8.4)
If d is detected, it is not sufficient to exchange d, also use new p, q!

b) Never let somebody observe your decryption process! (Side-channel attack → AMC)

c) Conjecture of RSA (78):
"An eff. alg. to break the RSA system leads to an eff. alg. for factoring."
(Still open question.)

RSA speed.

RSA is ~ 1000 times slower than DES in hardware.
" " ~ 100 times slower " " in software.
(in hardware 1995: 1 MB/s)

8.1.2. Implementation of RSA

- Large prime p, q → Miller-Rabin primality test
- Choice of $d \in \mathbb{Z}^*_{(p-1)(q-1)}$
  → Start with some ~~d_0~~ d_0,
  $d_0 = d_0 + 1$ until $\gcd(d_0, \varphi(n)) = 1$
- Inverse $d^{-1} \bmod \varphi(n)$ → extended Euclidean alg.
- Exponentiation → square-and-multiply
- Table concerning RSA hardware,
  see Schneier p. 469.