Public-key Cryptography:

Recall: Symmetric-Crypto : A and B share a secret/private key k.

If third person E has k $\longrightarrow$ E can decrypt messages.

$\searrow$ E can fake messages.

if k uses for identific.

Weak Point:

- One has to exchange every time a key over a secure channel
  $\Rightarrow$ key management can become costly.

Public-key Cryptography = asymmetric cryptography.
(Used in PGP)

"If A sends a message to B, A looks up the public key of B in order to encryp the message."

Every user has a pair of keys (e, d) with differ, asymmetric properties:

- e public key, it allows any party to encrypt to this key.
- d private key, it allows the owner to decrypt messages send to the matching public key e.
- There should be <u>no way</u> to compute d from e (The other way around is quite common)

Public-key cryptosystems consists of 3 algos

(i) key generation (ii) encription (iii) decription

Mathematical Background:

Let $n \in \mathbb{N}$. We consider $\mathbb{Z}/n = \mathbb{Z}/n\mathbb{Z}$ = "integers modulo n"

~~We identify $\mathbb{Z}/n$ with $\{0, 1, \dots, n-1\}$~~

For $a \in \mathbb{Z}$ we can compute (using XGCD) integers $b, \hat{b}$ such that

$$\gcd(a,n) = b \cdot a + \hat{b} \cdot n$$

if $\gcd(a,n)=1$

$$\Leftrightarrow \quad 1 = b \cdot a + \hat{b} \cdot n \implies 1 \equiv b \cdot a \bmod n \Leftrightarrow b^{-1} \equiv a \bmod n$$

In that case b is called the inverse of a modulo n.

Every a with $\gcd(a,n) = 1$ has an inverse modulo n.

Notation:

- $(\mathbb{Z}/n)^\times$ = invertible elements in $\mathbb{Z}/n = \{a \mid 1 \leq a < n, \gcd(a,n) = 1\}$
  = multiplicative group mod n.
- $|(\mathbb{Z}/n)^\times| =: \varphi(n)$ size of $(\mathbb{Z}/n)^\times$ (order of this finite group)
  φ is called Euler's totient, Euler Phi, Phi-function.

Examples:

(a) $(\mathbb{Z}/7)^\times = \{1, 2, \dots, 6\}$; $\varphi(7) = 6$

Lemma: $\varphi(p) = p - 1$ for any prime number p.

(b) $(\mathbb{Z}/6)^\times = \{1, 5\}$; $\varphi(6) = 2$

(c) $(\mathbb{Z}/10)^\times = \{1, 3, 7, 9\}$; $\varphi(10) = 4$

(d) $\varphi(15) = |\{1, 2, 4, 7, 8, 11, 13, 14\}| = 8 = (3-1)(5-1)$

Lemma: Let p, q be two prime numbers. Then,

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

Proof.

				divisible by p		divisible by q
1	2	3	$\dots$	p	copy and change $p \mapsto q$	q
$p+1$	$p+2$	$p+3$	$\dots$	$2p$		$2q$
$2p+1$	$2p+2$	$2p+3$	$\dots$	$3p$		$3q$
$\vdots$	$\vdots$	$\vdots$		$\vdots$		$\vdots$
$(q-1)p+1$	$\dots$			$q\cdot p$		$q\cdot p$
				q of them!		p of them

Thus there are $p\cdot q - q - p + 1 = (p-1)(q-1)$ $\square$

Lemma:

- $(\mathbb{Z}/n)^\times$ is a group under multiplication with $\varphi(n)$ elements.
- $a^{\varphi(n)} = 1$ and $\underbrace{a^{\varphi(n)+1} = a}$

  ↳ holds in general for any $a \in \mathbb{Z}/n$!

RSA: (Rivest, Shamir, Adleman 1977).

- Let p, q be two prime numbers, $p \neq q$ and ~~$p \neq q$~~ both have about the same size.
- Compute $n = p\cdot q$ and $\varphi(n) = (p-1)(q-1)$. → we can compute that only, because we have factorization
- Fix integer e with $\gcd(e, \varphi(n)) = 1$. that is, e has inverse. Let d be an integer such that
$$e\cdot d \equiv 1 \mod \varphi(n).$$
- Forget about $p, q, \varphi(n)$
- Publish the public key (e,n), BUT: d is kept secret ← private key

Encription. Encrypt $m \in \mathbb{Z}, m < n$ by computing $c \equiv m^e \mod n$

Decription: Decrypt c by computing $m' = c^d \mod n$.

System works, i.e. $m' = m$, since
$$m' \equiv c^d \equiv (m^e)^d \equiv m^{e\cdot d} \equiv m^{k\cdot\varphi(n)+1} \equiv m \mod n,$$
for some integer k. Remark. An RSA key can also be used to sign messages.

<u>Signature</u>: to sign message m with $m < n$ compute

$$s \equiv m^d \mod n.$$

Since d is a secret, nobody else can do this computation.

<u>Verification</u>: To verify that s is a signature on m compute

$$m' = s^e \mod n$$

and accept the signature as valid if $m = m'$.

<u>Remark</u>: • This RSA "~~schoolbook~~ school book" version does not ~~require~~ satisfy modern requirements.

• We want to be able to sign messages longer than n. In practice we sign the hash $h(m)$ instead of m. (Is also better for security)

<u>Computational issue of m^e</u>:

How to compute $m^e \mod n$? (m^e gets very large for large e)

<u>Idea</u>: Reduce modulo n. ~~in every step~~ after any multiplication / squaring.

• $m^4 = \underbrace{m \cdot m \cdot m \cdot m}_{3\text{ multipl.}} = \underbrace{m^2 \cdot m^2 = (m^2)^2}_{2\text{ squarings} + 1\text{ mult}}$

• $m^9 = \underbrace{((m^2)^2)^2 \cdot m}_{3\text{ squarings} + 1\text{ multi.}}$ • $m^6 = (m^2 \cdot m)^2$ • $m^{10} = ((m^2)^2 \cdot m)^2$

• $m^{11} = ((m^2)^2 \cdot m)^2 \cdot m$, • $m^{15} = (((m^2 \cdot m)^2 \cdot m)^2 \cdot m$.

In general there are at most $\lfloor \log_2(e) \rfloor$ squarings and at most that many multiplications.

To compute the pattern we look at the binary representation of the exponent:

$4 = (100)_2$, $9 = (1001)_2$, $6 = (110)_2$, $10 = (1010)_2$, $11 = (1011)_2$, $15 = (1111)_2$.

Thus, ignoring the first position, we perform for any entry squaring and whenever the bit is one an multiplication.
We scan from left to right.

Let $e = \sum_{i=0}^{\ell-1} e_i 2^i$, $\ell = \lfloor \log_2 e \rfloor + 1$.

We compute $m^e \bmod n$ as follows:

~~Square-and-multiplication~~ Fast exponentiation algorithm:

1. $c \leftarrow m$
2. For $i = \ell-2$ to 0
   $c \leftarrow c^2 \bmod n$
   if $e_i = 1$: $c \leftarrow c \cdot m \bmod n$
3. return c.

Algo takes $\ell-1$ squarings and $\leq \ell-1$ multiplications.

Multiplications $=$ $|\{i \mid e_i \neq 0\}|$ = Hamming weight of e

Remark: • One can choose e small ~~since~~ since it belongs to the public key — small d would not work.

• Decription also needs multiplication and squares.
  • $d = 5, 17$ small would be easy to find by brute force
  • $d = \sqrt[3]{n}$ is dangerous by attack of Wiener.

• Common choices are $e = 3$, $e = 17$, $e = 2^{16}+1 = 65537$
(These e's have also small Hamming weight)

Problems using schoolbook RSA and small e:

Assume: A, B, C all use $e = 3$ and somebody sends the same message to all of them
Denote by n_A, n_B, n_C the modulus of A, B, C, respectively.

We obtain

$$\left.\begin{aligned} c_A &\equiv m^3 \bmod n_A \\ c_B &\equiv m^3 \bmod n_B \\ c_C &\equiv m^3 \bmod n_C . \end{aligned}\right\} (*)$$

~~1)~~ One of $\gcd(n_A, n_B)$, $\gcd(n_B, n_C)$, $\gcd(n_A, n_C)$ is not 1.

(A) $\Rightarrow$ we get one factor $p_i,\ i \in \{A,B,C\}$ of one of them

$\Rightarrow$ ~~factorize the~~ we can factor the public key

We compute φ and d $\Rightarrow$ completly broken

(B) two parties share a key (e,n) and can read each others messages.

2) $\gcd(\dots) = 1$.

Then $(*)$ is a system of congruences with moduli that are coprime.

So the exist a solution M modulo $N := n_A \cdot n_B \cdot n_C$ by the Chinese Rem Th.: with $M \equiv m^3 \bmod N$

Since $m^3 < N$ we have $M = m^3 \Leftrightarrow \sqrt[3]{M} = m$.

In $\mathbb{Z}$ we can efficiently compute cube roots.

Conclusion: We deduce m from the publicly avaiable information.

Remark: Same approach works for e messages to recipients all using exponent e.