Public-key encryption: two keys.
- One key is made public and used to encrypt.
- The other key is kept private and enables to decrypt.

Alice wants to send a message to Bob:
- She encrypts it using Bob’s public-key.
- Only Bob can decrypt it using his own private-key.
- Alice and Bob do not need to meet to establish a secure communication.

Security:
- It must be difficult to recover the private-key from the public-key
- but not enough in practice.
The RSA algorithm is the most widely-used public-key encryption algorithm
- Invented in 1977 by Rivest, Shamir and Adleman.
- Used for encryption and signature.
- Widely used in electronic commerce protocols (SSL).
Key generation:

- Generate two large distinct primes $p$ and $q$ of same bit-size.
- Compute $n = p \cdot q$ and $\phi = (p - 1)(q - 1)$.
- Select a random integer $e$, $1 < e < \phi$ such that $\gcd(e, \phi) = 1$
- Compute the unique integer $d$ such that

$$e \cdot d \equiv 1 \mod \phi$$

using the extended Euclidean algorithm.
- The public key is $(n, e)$. The private key is $d$. 
RSA encryption

Encryption
- Given a message \( m \in [0, n - 1] \) and the recipient’s public-key \((n, e)\), compute the ciphertext:

\[
c = m^e \mod n
\]

Decryption
- Given a ciphertext \( c \), to recover \( m \), compute:

\[
m = c^d \mod n
\]
Definition:

- \( \phi(n) \) for \( n > 0 \) is defined as the number of integers \( a \) comprised between 0 and \( n - 1 \) such that \( \gcd(a, n) = 1 \).
- \( \phi(1) = 1, \phi(2) = 1, \phi(3) = 2, \phi(4) = 2 \).

Equivalently:

- Let \( \mathbb{Z}_n^* \) be the set of integers \( a \) comprised between 0 and \( n - 1 \) such that \( \gcd(a, n) = 1 \).
- Then \( \phi(n) = |\mathbb{Z}_n^*| \).
If $p \geq 2$ is prime, then

$$\phi(p) = p - 1$$

More generally, for any $e \geq 1$,

$$\phi(p^e) = p^{e-1} \cdot (p - 1)$$

For $n, m > 0$ such that $\gcd(n, m) = 1$, we have:

$$\phi(n \cdot m) = \phi(n) \cdot \phi(m)$$
Euler’s theorem

Theorem
For any integer \( n > 1 \) and any integer \( a \) such that \( \gcd(a, n) = 1 \), we have \( a^{\phi(n)} \equiv 1 \mod n \).

Proof
Consider the map \( f : \mathbb{Z}_n^* \rightarrow \mathbb{Z}_n^* \), such that \( f(b) = a \cdot b \) for any \( b \in \mathbb{Z}_n^* \).

\( f \) is a permutation, therefore:

\[
\prod_{b \in \mathbb{Z}_n^*} b = \prod_{b \in \mathbb{Z}_n^*} (a \cdot b) = a^{\phi(n)} \cdot \left( \prod_{b \in \mathbb{Z}_n^*} b \right)
\]

Therefore, we obtain \( a^{\phi(n)} \equiv 1 \mod n \).
Fermat’s little theorem

Theorem
For any prime $p$ and any integer $a \neq 0 \mod p$, we have $a^{p-1} \equiv 1 \mod p$. Moreover, for any integer $a$, we have $a^p \equiv a \mod p$.

Proof
Follows from Euler’s theorem and $\phi(p) = p - 1$. 
Proof that decryption works

Since \( e \cdot d \equiv 1 \mod \phi \), there is an integer \( k \) such that \( e \cdot d = 1 + k \cdot \phi \).

If \( m \not\equiv 0 \mod p \), then by Fermat’s little theorem \( m^{p-1} \equiv 1 \mod p \), which gives:

\[
m^{1 + k \cdot (p-1) \cdot (q-1)} \equiv m \mod p
\]

This equality is also true if \( m \equiv 0 \mod p \).

This gives \( m^{ed} \equiv m \mod p \) for all \( m \).

Similarly, \( m^{ed} \equiv m \mod q \) for all \( m \).

By the Chinese Remainder Theorem, if \( p \neq q \), then

\[
m^{ed} \equiv m \mod n
\]
Key generation:
- Public modulus: $N = p \cdot q$ where $p$ and $q$ are large primes.
- Public exponent: $e$
- Private exponent: $d$, such that $d \cdot e \equiv 1 \pmod{\phi(N)}$

To sign a message $m$, the signer computes:
- $s = m^d \mod N$
- Only the signer can sign the message.

To verify the signature, one checks that:
- $m = s^e \mod N$
- Anybody can verify the signature
There are many attacks on basic RSA signatures:

- Existential forgery: \( r^e = m \mod N \)
- Chosen-message attack: \((m_1 \cdot m_2)^d = m_1^d \cdot m_2^d \mod N\)

To prevent from these attacks, one usually uses a hash function. The message is first hashed, then padded.

\[
m \rightarrow H(m) \rightarrow 1001...0101\|H(m)
\]

Example: PKCS#1 v1.5:
\[
\mu(m) = 0001\text{ FF}...\text{FF}00\|c_{\text{SHA}}\|\text{SHA}(m)
\]

ISO 9796-2: \( \mu(m) = 6A\|m[1]\|H(m)\|\text{BC} \)

The signature is then \( \sigma = \mu(m)^d \mod N \)
Attacks against RSA

- Factoring
  - Equivalence between factoring and breaking RSA?

- Mathematical attacks
  - Attacks against plain RSA encryption and signature
  - Heuristic countermeasures
  - Low private / public exponent attacks
  - Provably secure constructions

- Implementation attacks
  - Timing attacks, power attacks and fault attacks
  - Countermeasures
Factoring attack

- Factoring large integers
  - Best factoring algorithm: Number Field Sieve
  - Sub-exponential complexity
    \[
    \exp \left( (c + o(1)) n^{1/3} \log^{2/3} n \right)
    \]
    for \( n \)-bit integer.
  - Current factoring record: 768-bit RSA modulus.
- Use at least 1024-bit RSA moduli
  - 2048-bit for long-term security.
Breaking RSA:

- Given \((N, e)\) and \(y\), find \(x\) such that \(y = x^e \mod N\)

Open problem

- Is breaking RSA equivalent to factoring?

Knowing \(d\) is equivalent to factoring

- Probabilistic algorithm (RSA, 1978)
Plain RSA encryption: dictionary attack

- If only two possible messages $m_0$ and $m_1$, then only $c_0 = (m_0)^e \mod N$ and $c_1 = (m_1)^e \mod N$.

  $\Rightarrow$ encryption must be probabilistic.

PKCS#1 v1.5

- $\mu(m) = 0002||r||00||m$
- $c = \mu(m)^e \mod N$
- Still insufficient (Bleichenbacher’s attack, 1998)
Attacks against Plain RSA signature

- Existential forgery
  - \( r^e = m \mod N \), so \( r \) is signature of \( m \)

- Chosen message attack
  - \((m_1 \cdot m_2)^d = m_1^d \cdot m_2^d \mod N \)

To prevent from these attacks, one first computes \( \mu(m) \), and lets \( s = \mu(m)^d \mod N \)

- ISO 9796-1:
  \[
  \mu(m) = \bar{s}(m_z)s(m_{z-1})m_zm_{z-1} \ldots s(m_1)s(m_0)m_06
  \]

- ISO 9796-2:
  \[
  \mu(m) = 6A || m[1] || H(m) || BC
  \]

- PKCS#1 v1.5:
  \[
  \mu(m) = 0001 \ FF \ldots FF00 || c_{SHA} || SHA(m)
  \]
Attacks against RSA signatures

- Desmedt and Odlyzko attack (Crypto 85)
  - Based on finding messages $m$ such that $\mu(m)$ is smooth (product of small primes only)
  - $\mu(m_i) = \prod_j \rho_j^{\alpha_{i,j}}$ for many messages $m_i$.
  - Solve a linear system and write $\mu(m_k) = \prod_i \mu(m_i)$
  - Then $\mu(m_k)^d = \prod_i \mu(m_i)^d \mod N$

- Application to ISO 9796-1 and ISO 9796-2 signatures
  - Cryptanalysis of ISO 9796-1 (Coron, Naccache, Stern, 1999)
  - Cryptanalysis of ISO 9796-2 (Coron, Naccache, Tibouchi, Weinmann, 2009)
  - Extension of Desmedt and Odlyzko attack.
  - For ISO 9796-2 the attack is feasible if the output size of the hash function is small enough.
To reduce decryption time, one could use a small $d$
- Wiener’s attack: recover $d$ if $d < N^{0.25}$

Boneh and Durfee’s attack (1999)
- Recover $d$ if $d < N^{0.29}$
- Based on lattice reduction and Coppersmith’s technique
- Open problem: extend to $d < N^{0.5}$

Conclusion: devastating attack
- Use a full-size $d$
To reduce encryption time, one can use a small $e$
- For example $e = 3$ or $e = 2^{16} + 1$

Coppersmith’s theorem:
- Let $N$ be an integer and $f$ be a polynomial of degree $\delta$. Given $N$ and $f$, one can recover in polynomial time all $x_0$ such that $f(x_0) = 0 \mod N$ and $x_0 < N^{1/\delta}$.

Application: partially known message attack:
- If $c = (B||m)^3 \mod N$, one can recover $m$ if $|m| < |N|/3$
- Define $f(x) = (B \cdot 2^k + x)^3 - c \mod N$.
- Then $f(m) = 0 \mod N$ and apply Coppersmith’s theorem to recover $m$. 
Coppersmith’s short pad attack

Let $c_1 = (m| r_1)^3 \mod N$ and $c_2 = (m| r_2)^3 \mod N$
One can recover $m$ if $r_1, r_2 < N^{1/9}$
Let $g_1(x, y) = x^3 - c_1$ and $g_2(x, y) = (x + y)^3 - c_2$.
$g_1$ and $g_2$ have a common root $(m| r_1, r_2 - r_1)$ modulo $N$.
$h(y) = \text{Res}_x(g_1, g_2)$ has a root $\Delta = r_2 - r_1$, with $\deg h = 9$.
To recover $m| r_1$, take $\gcd$ of $g_1(x, \Delta)$ and $g_2(x, \Delta)$.

Conclusion:
Attack only works for particular encryption schemes.
Low public exponent is secure when provably secure construction is used. One often takes $e = 2^{16} + 1$. 