RSA and Its proof
The RSA Algorithm

1. Motivation:

   Alice  \[\rightarrow\]  Bob

   m

   Eve

   <1> Alice and Bob do not want to take the time to send a courier with a key.

   <2> All information that Alice sends to Bob will be obtained by Eve.
Question: Is it possible for a message to be sent in such a way that Bob can read it but Eve cannot?

② Solution to ①: public key cryptosystem

It was first publicly suggested by Diffie and Hellman in their classic paper [1976]. (key exchange) However, they did not yet have a practical implementation.
③ In 1977, RSA (Rivest, Shamir, Addleman) proposed RSA algorithm, based on the idea that factorization of integers into their prime factors is hard.

④ It had long been claimed that government cryptographic agencies had discovered the RSA algorithm several years earlier, but secrecy rules prevented them from releasing any evidence.
5 In 1997, documents released by CESG, a British crypto agency, showed that in 1970, James Ellis had discovered public key crypto, and in 1973, Clifford Cocks had written an internal document describing a version of RSA algorithm.
Here is how the RSA algorithm works,

1. Bob chooses two distinct large primes \( p \) and \( q \) and let \( n = pq \).
2. Bob also chooses an encryption exponent \( e \) such that \( \gcd(e, (p-1)(q-1)) = 1 \)
3. He sends the pair \( (n, e) \) to Alice but keeps the values of \( p \) and \( q \) secret.
Alice writes her message as number \( m \). If \( m \) is larger than \( n \), she breaks the message into blocks, each of which is less than \( n \). However, for simplicity let's assume \( m < n \).

Alice computes

\[
    c = m^e \pmod{n}
\]

and sends \( c \) to Bob.
Since Bob knows $p$ and $q$, he computes $(p-1)(q-1)$ and finds the decryption exponent $d$ with $d \cdot e \equiv 1 \pmod{(p-1)(q-1)}$.

(As we'll see later)

$$m = c^d \pmod{n}$$
7 Summary of the RSA Algo

1. Bob chooses secret primes $p$ and $q$ and compute $n=pq$.
2. Bob chooses $e$ with $\gcd(e,(p-1)(q-1)) = 1$.
3. Bob computes $d$ with $de \equiv 1 \pmod{(p-1)(q-1)}$.
4. Bob makes $n$ and $e$ public, and keeps $p, q, d$ secret.
5. Alice encrypts $m$ as $c \equiv m^e \pmod{n}$ and sends $c$ to Bob.
6. Bob decrypts by computing $m \equiv c^d \pmod{n}$.
\[ C = m^e \pmod{n} \]

\[ m = C^d \pmod{n} \]
(8) Eq. Bob chooses

\[ p = 885320963, \quad q = 238855417 \]

\[ n = pq = 21463707796206571 \]

encryption exponent \( e = 9007 \)

\[ m = 30120 \]

Alice computes

\[ c \equiv m^e \equiv 30120^{9007} \equiv 11353585035722866 \pmod{n} \]

Alice sends \( c \) to Bob.
Bob uses the extended Euclidean algorithm to compute decryption exponent $d$:
\[ d \equiv e^{-1} \pmod{(p-1)(q-1)} \]
\[ d = 116402471153538991 \]

Bob computes
\[ c^d \equiv 113535859035722866^{116402471153538991} \]
\[ \equiv 30120 \pmod{n} \]

So Bob obtains the original message.
9. Why RSA works?

By Euler's Thm,

If \( \gcd(a, n) = 1 \), then \( a^{\phi(n)} \equiv 1 \pmod{n} \)

So \( \gcd(m, n) = 1 \)

Since \( de \equiv 1 \pmod{\phi(n)} \)

\( de = 1 + k\phi(n) \)

So \( c^d \equiv (me)^d \equiv m^{1 + k\phi(n)} \equiv m \cdot (m^{\phi(n)})^k \equiv m \cdot 1^k \equiv m \pmod{n} \)
<2> If \( \gcd(m, n) \neq 1 \)

1. \( \gcd(m, p) = 1 \)

\[ m^{p-1} = 1 \pmod{p} \quad (\text{Fermat's little Theorem}) \]

\[ m^{1 + k(p-1)(q-1)} = m \pmod{p} \quad (*) \]

2. \( \gcd(m, p) = p \) then \( m = 0 \pmod{p} \)

\((*)\) is valid again
3. Hence \( m^{ed} = m \pmod{p} \)

4. Similar argument \( m^{ed} = m \pmod{q} \)

5. By CRT

\[ c^d = (m^e)^d = m^{ed} = m \pmod{n} \]

QED.
The security of RSA is provided by the assumption that \( n \) cannot be factored.

If we know \( n \) and \( \phi(n) \), then we can quickly find \( p \) and \( q \).

\[
\text{pf} \quad n - \phi(n) + 1 = pq - (p-1)(q-1) + 1 = p+q
\]

\[
\therefore \quad x^2 - (n - \phi(n) + 1)x + n = x^2 - (p+q)x + pq = (x-p)(x-q)
\]
\[ p, q = \frac{(n - \phi(n) + 1) \pm \sqrt{(n - \phi(n) + 1)^2 - 4n}}{2} \]

Eg. \( n = 221, \phi(n) = 192 \)
\[ x^2 - 30x + 221 \]
\[ p, q = \frac{30 \pm \sqrt{30^2 - 4 \cdot 221}}{2} = 13, 17 \]

\(<2>\) If we know \( d \) and \( e \), then we can probably factor \( n \).

\(<pf>\) \( \text{mod} \equiv m (\text{mod} \ n) \ \forall \gcd(m, n) = 1 \)
\[ m^{ed-1} \equiv 1 \quad (\text{mod } n) \]

\[ ed-1 = k \phi(n) \]

test possible \( k \) to find \( \phi(n) \)

and then find \( p, q \) by \( \langle 1 \rangle \)

The point of claims \( \langle 1 \rangle \) and \( \langle 2 \rangle \) is that finding \( \phi(n) \) or finding \( d \) is as hard as factoring \( n \).
In practice, the RSA algorithm is not quite fast enough for sending massive amounts of data. Therefore, the RSA algorithm is often used to send a key for a fast encryption method such as DES.