RSA crypto-system

Saturday, 30 November 2019 11:54 AM

ASSYMETRIC ALGORITHM:

Alice   Bob

$\xleftarrow{k_{pub}}$ K_{pub}, K_{pr}

$y = e_{k_{pub}}(x)$

$\xrightarrow{y}$ $d_{k_{pr}}(y) = x$

Some facts about RSA:
- → invented in 1977 by Rivest, Shamir, Adleman
- → based on the paper by DIFFIE-HELLMAN
- → most popular PK cryptosystem.
- → patented in the USA until 2000.

RSA ALGORITHM

KEY GENERATION:

unlike symm algorithm (AES, 3DES), PK algorithms require the computation of the pair (k_{pub}, k_{pr})

1. choose large primes p, q → $p, q \geq 2^{512} \Rightarrow n \geq 2^{1024}$
2. $n = p \cdot q$
3. $\phi(n) = (p-1)(q-1)$
4. choose $k_{pub} = e \in \{1, \ldots\ldots, \phi(n)-1\}$ s.t. $\gcd(e, \phi(n)) = 1$ (this guarantees that the inverse exist (d))
5. compute $k_{pr} = d$ s.t. $d \cdot e \equiv 1 \mod \phi(n)$ (by Extended Euclidean Algorithm)

$K_{pub} = (n, e)$, $k_{pr} = (d)$

RSA ENCRYPTION & DECRYPTION:

Encryption:

given $k_{pub} = (n, e)$, $x \in \mathbb{Z}_n = \{0, 1, \ldots\ldots n-1\}$

$$y = e_{k_{pub}}(x) = x^e \mod n$$

Decryption:

given $K_{pr} = d$, $y \in \mathbb{Z}_n$

$$x = d_{k_{pr}}(y) \equiv y^d \mod n$$

example:

Alice   Bob

① $p = 3$ $q = 11$

2 Remarks:

→ proof of correctness:

What is interesting is that the message x is first raised to the eth power during encryption and the result y is raised to the dth power in the decryption, and the result of this is again equal to the message x. Expressed as an equation, this process is:

$$d_{k_{pr}}(y) = d_{k_{pr}}(e_{k_{pub}}(x)) \equiv (x^e)^d \equiv x^{de} \equiv x \bmod n. \quad (7.3)$$

This is the essence of RSA. We will now prove why the RSA scheme works.

Proof. We need to show that decryption is the inverse function of encryption, $d_{k_{pr}}(e_{k_{pub}}(x)) = x$. We start with the construction rule for the public and private key: $d \cdot e \equiv 1 \bmod \Phi(n)$. By definition of the modulo operator, this is equivalent to:

$$d \cdot e = 1 + t \cdot \Phi(n),$$

where t is some integer. Inserting this expression in Eq. (7.3):

$$d_{k_{pr}}(y) \equiv x^{de} \equiv x^{1+t \cdot \Phi(n)} \equiv x^{t \cdot \Phi(n)} \cdot x^1 \equiv (x^{\Phi(n)})^t \cdot x \bmod n. \quad (7.4)$$

This means we have to prove that $x \equiv (x^{\Phi(n)})^t \cdot x \bmod n$. We use now Euler's Theorem from Sect. 6.3.3, which states that if $\gcd(x,n) = 1$ then $1 \equiv x^{\Phi(n)} \bmod n$. A minor generalization immediately follows:

$$1 \equiv 1^t \equiv (x^{\Phi(n)})^t \bmod n, \quad (7.5)$$

where t is any integer. For the proof we distinguish two cases:

First case: $\gcd(x,n) = 1$

Euler's Theorem holds here and we can insert Eq. (7.5) into (7.4):

$$d_{k_{pr}}(y) \equiv (x^{\Phi(n)})^t \cdot x \equiv 1 \cdot x \equiv x \bmod n. \qquad q.e.d.$$

This part of the proof establishes that decryption is actually the inverse function of encryption for plaintext values x which are relatively prime to the RSA modulus n. We provide now the proof for the other case.

Second case: $\gcd(x,n) = \gcd(x, p \cdot q) \neq 1$

Since p and q are primes, x must have one of them as a factor:

$$x = r \cdot p \quad \text{or} \quad x = s \cdot q,$$

where r, s are integers such that $r < q$ and $s < p$. Without loss of generality we assume $x = r \cdot p$, from which follows that $\gcd(x,q) = 1$. Euler's Theorem holds in the following form:

$$1 \equiv 1^t \equiv (x^{\Phi(q)})^t \bmod q,$$

where t is any positive integer. We now look at the term $(x^{\Phi(n)})^t$ again:

$$(x^{\Phi(n)})^t \equiv (x^{(q-1)(p-1)})^t \equiv ((x^{\Phi(q)})^t)^{p-1} \equiv 1^{(p-1)} = 1 \bmod q.$$

Using the definition of the modulo operator, this is equivalent to:

$$(x^{\Phi(n)})^t = 1 + u \cdot q,$$

where u is some integer. We multiply this equation by x:

$$\begin{aligned} x \cdot (x^{\Phi(n)})^t &= x + x \cdot u \cdot q \\ &= x + (r \cdot p) \cdot u \cdot q \\ &= x + r \cdot u \cdot (p \cdot q) \\ &= x + r \cdot u \cdot n \\ x \cdot (x^{\Phi(n)})^t &\equiv x \bmod n. \end{aligned}$$

Inserting Eq. (7.6) into Eq. (7.4) yields the desired result:

$$d_{k_{pr}} = (x^{\Phi(n)})^t \cdot x \equiv x \bmod n.$$

□

FAST EXPONENTIATION:

problem:

$y \equiv x^e \mod n$

$x \equiv y^d \mod n$

with very large numbers.

$x^{2^{1024}} = ?$

naïve way	better way
$x \cdot x = x^2$	$x \cdot x = x^2$
$x^2 \cdot x = x^3$	$x^2 \cdot x^2 = x^4$
$x^3 \cdot x = x^4$	$x^4 \cdot x^4 = x^8$
⋮	⋮
$x^{2^{1024}-1} \cdot x = x^{2^{1024}}$	$x^{2^{1023}} \cdot x^{2^{1023}} = x^{2^{1024}}$

example:

Alice		Bob
		(1) $p = 3,\ q = 11$
$x = 4$		(2) $n = pq = 33$
		(3) $\phi(n) = 2 \times 10 = 20$
		(4) Choose $e = 3$
		$\gcd(3, 20) = 1$ ✓
	$\xleftarrow{K_{pub}(33,3)}$	(5) $d = e^{-1} = 7 \mod 20$
$y = 4^3 = 64 \mod 33 = 31$	$\xrightarrow{\quad}$	$x = y^d = 31^7 = 4 \mod 33$

TRICK:

$$31^7 \equiv (-2)^7 \mod 33$$
$$\equiv -128 \mod 33$$
$$\equiv -4 \cdot 33 + 4 \mod 33$$
$$\equiv 4 \mod 33 \quad (\text{dd}).$$

$$x^{2^{1024}-1} \cdot x = x^{2^{1024}}$$

$2^{1024}-1$ MULTIPLICATIONS

$$x^{2^{1023}} \cdot x^{2^{1023}} = x^{2^{1024}}$$

1024 MULTIPLICATIONS

SQUARE and MULTIPLY ALGORITHM

(binary method / left to Right exponentiation)

Ex: x^{26} $\quad (26)_{10} = (11010)_2$

SQ	$x \cdot x = x^2$	$(x^1)^2 = x^{(10)_2}$
MUL	$x \cdot x^2 = x^3$	$x^1 \cdot x^{(10)_2} = x^{(11)_2}$
SQ	$x^3 \cdot x^3 = x^6$	$(x^{11})^2 = x^{(110)_2}$
SQ	$x^6 \cdot x^6 = x^{12}$	$(x^{110})^2 = x^{(1100)_2}$
MUL	$x \cdot x^{12} = x^{13}$	$x^{(1100)_2} \cdot x^{(1)_2} = x^{(1101)_2}$
SQ	$x^{13} \cdot x^{13} = x^{26}$	$(x^{(1101)_2})^2 = x^{(11010)_2}$

working scheme: Scan the exponent bits left to Right:

1) in every iteration we SQUARE
2) IF current bit is 1: MULTIPLY by x