Public-key Cryptography with RSA

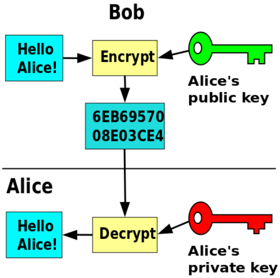
Wittawat Jitkrittum
wittawat@gatsby.ucl.ac.uk

Gatsby Tea Talk

18 Nov 2014

Overview

- Symmetric key cryptography uses same secret key for encryption and decryption.
    - Need to agree in advance upon which key to use.
    - Need a secure channel to exchange key.
- Public key cryptography uses one public key for encryption and private key for decryption.
- Public key available to anyone.
- Private key known only to the owner
- Can use private key to encrypt as well. Equivalent to a digital signature.



Public-key cryptography.
(image from Wikipedia)

RSA Cryptosystem

- Ron **R**ivest, Adi **S**hamir, and Leonard **A**dleman first published RSA in 1977.
- Assume B wants to send a message m (integer) to A.
- A has key pair: (public key, private key) $= (e, d)$ and pre-chosen n.
- RSA relies on

$$F(m, k) = m^k \mod n$$

B encrypts with public key e:

$$c = F(m, e) = m^e \mod n$$

A decrypts with private key d:

$$m = F(c, d) = c^d \mod n$$

- $x \mod y =$ remainder of x/y. For example, $12 \mod 5 = 2$.
- Need to find e, d, n that work.

Divisibility

- $\gcd(x, y)$: greatest common divisor of x and y.
  - $\gcd(8, 12) = 4$
  - $\gcd(5, 9) = 1$
- An integer $p > 1$ is a prime iff its divisors are 1 and p.
  - Prime: $2, 11, 23$
  - Not prime: $6, 10$
- Arbitrary integers x and y are said to be relatively prime or coprime iff $\gcd(x, y) = 1$.
  - Examples: $(5, 9), (8, 15)$
  - Does not mean x and y are prime.

Modular Arithmetic

- $x \mod n :=$ remainder when x is divided by n e.g., $12 \mod 5 = 2$.
  - n is called modulus.
- x, y are congruent modulo n if $(x \mod n) = (y \mod n)$, written as

$$x \equiv y \pmod{n}$$

  - Examples: $3 \equiv 5 \pmod{2}$.
- $\pmod{n}$ operator maps all integers into set $Z_n = \{0, 1, \ldots, (n-1)\}$.
- Modular arithmetic performs arithmetic operations within confines of Z_n.

Properties of Modular Arithmetic

$$
\begin{aligned}
(x+y) \mod n &= [(x \mod n) + (y \mod n)] \mod n \\
(x-y) \mod n &= [(x \mod n) - (y \mod n)] \mod n \\
(x \times y) \mod n &= [(x \mod n) \times (y \mod n)] \mod n
\end{aligned}
$$

- x is multiplicative inverse of y if $x \times y \equiv 1 \pmod{n}$. Denoted by x^{-1}.
  - Example: $3 \times 4 \equiv 1 \pmod{11}$.
  - Not all integers have a multiplicative inverse.
  - 2^{-1} does not exist under $\pmod{4}$ because $2 \times y - 1$ is not divisible by 4.

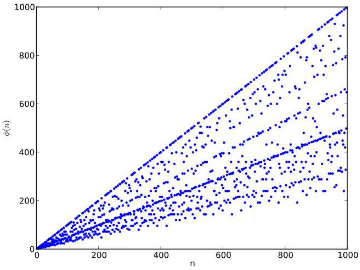
Lemma

The multiplicative inverse of y (modulo n) exists iff y and n are relatively prime.

Euler's Totient Function

Define Euler's totient function $\phi(n) :=$ number of integers in $\{1, 2, \ldots, n-1\}$ relatively prime to n.

- i.e., number of $x < n$ such that $\gcd(x, n) = 1$
- $\phi(1) = 1$
- For prime p, $\phi(p) = p - 1$
- For primes p and q, $\phi(pq) = (p-1)(q-1)$



(image from Wikipedia)

RSA Key Generation

Generate public key e, private key d, and n.

1. **Large Prime Number Generation.** Generate large primes p and q. Can be done with Rabin-Miller primality test (probabilistic test).
2. **Modulus.** Set $n = pq$**.**
3. **Totient**. Compute $\phi(n) = (p-1)(q-1)$.
4. **Public key** e. Pick a prime e in $[3, \phi(n))$ that is relatively prime to $\phi(n)$ i.e., $\gcd(e, \phi(n)) = 1$.
5. **Private key** d. By the lemma, the multiplicative inverse of e exists (modulo $\phi(n)$). Can be determined with the Extended Euclidean Algorithm. Set it to d.

Observations

- We have $ed \equiv 1 \pmod{\phi(n)}$ by design.
- Imply $ed = k\phi(n) + 1$ for some positive integer k.

Useful Theorems

For proving correctness of RSA,

Fermat's Little Theorem

If p is prime, for m relatively prime to p, it holds that $m^{p-1} \equiv 1 \pmod{p}$.

- Example: $2^{5-1} = 16 \equiv 1 \pmod{5}$

Chinese Remainder Theorem

Let p and q be relatively prime. If $a \equiv m \pmod{p}$ and $a \equiv m \pmod{q}$, then $a \equiv m \pmod{pq}$.

- Example: $22 \equiv 2 \pmod{5}$ and $22 \equiv 2 \pmod{4}$.
  $\Rightarrow 22 \equiv 2 \pmod{5 \cdot 4}$.

Known So Far

Fermat's Little Theorem

If p is prime, for m relatively prime to p, it holds that $m^{p-1} \equiv 1 \pmod{p}$.

Chinese Remainder Theorem

Let p and q be relatively prime. If $a \equiv m \pmod{p}$ and $a \equiv m \pmod{q}$, then $a \equiv m \pmod{pq}$.

Known

1. $[(x \mod p) \times (y \mod p)] \mod p = (x \times y) \mod p$
2. $n = pq$.
3. $\phi(n) = (p-1)(q-1)$
4. $ed \equiv 1 \pmod{\phi(n)}$ by design. So, $ed = k\phi(n) + 1$ for some k.
5. Encrypt with public key e by $c = m^e \mod n$.
6. Decrypt with private key d by $m = c^d \mod n$.

RSA Algorithm and Correctness

- Encrypt with public key e by $c = m^e \mod n$.
- Decrypt with private key d by $m = c^d \mod n$.

Proof of Correctness. Need to show $m = c^d \mod n$.

- Suffices to show $m \equiv c^d \pmod p$ and $m \equiv c^d \pmod q$. Then use Chinese remainder theorem to get $m \equiv c^d \pmod n$.
- $c^d \pmod p = (m^e \pmod n)^d \pmod p = m^{ed} \pmod p = m^{k\phi(n)+1} \pmod p = m^{k(p-1)(q-1)+1} \pmod p$.

$$
\begin{aligned}
m^{ed} \pmod p &= m \cdot m^{k(p-1)(q-1)} \pmod p \\
&= m \cdot \left(m^{p-1}\right)^{k(q-1)} \pmod p \\
\text{(modular arithmetic)} &= m \cdot \left(m^{p-1} \pmod p\right)^{k(q-1)} \pmod p \\
\text{(Fermat's little theorem)} &= m \cdot (1)^{k(q-1)} \pmod p \\
&= m \pmod p \quad \square
\end{aligned}
$$

Security

- **Public**: n, e (public key), c (cipher text)
- **Secret**: p, q (factors of n), $\phi(n)$, d (private key)

Mathematical attacks:

1. Factor n into $n = pq$.
2. Determine $\phi(n)$ directly without $n = pq$. Can use it to find $d = e^{-1}$ modulo $\phi(n)$.
3. Determine d (private key) directly from n, e. As hard as (1).

Comments:

- Factoring n is considered fastest (still difficult). Used as measure of RSA security.
- http://en.wikipedia.org/wiki/RSA_Factoring_Challenge
- For factorizing $n = pq$, best published asymptotic running time is the general number field sieve (GNFS) algorithm: $O\left(\exp\left(\left(\frac{64}{9}b\right)^{1/3}(\log b)^{2/3}\right)\right)$ for b-bit number.
  (See Integer factorization, Wikipedia)

More on RSA

- In 1994, Peter Shor showed that a quantum computer (exists ?) would be able to factor n in polynomial time.
- As of 2010, the largest factored RSA number was 768 bits long (232 decimal digits).
  - State-of-the-art distributed implementation took around 1500 CPU years.
- Practical RSA keys: 1024 to 2048 bits.

Practical uses

- For exchanging a symmetric key
- Digital signature. Encrypt a message with one's private key.

Related Theorems

Euler's Theorem 1

For every x and n that are relatively prime, $x^{\phi(n)} \equiv 1 \pmod{n}$.

Euler's Theorem 2

For every positive integers x and n, $x^{\phi(n)+1} \equiv x \pmod{n}$

Fermat's Little Theorem 2

Let x be a positive integer. If p is prime, then $x^p \equiv x \pmod{p}$

- Example: $3^5 = 243 \equiv 3 \pmod{5}$

References I

- http://doctrina.org/How-RSA-Works-With-Examples.html
- http://doctrina.org/Why-RSA-Works-Three-Fundamental-Ques
- http://ict.siit.tu.ac.th/~steven/css322/
- http://en.wikipedia.org/wiki/Integer_factorization
- http://www.cse.cuhk.edu.hk/~taoyf/course/bmeg3120/notes/r