RSA-256bit

Digital Circuit Lab

TA: Po-Chen Wu

Outline

- Introduction to Cryptography
- RSA Algorithm
- Montgomery Algorithm for RSA-256 bit

Introduction to Cryptography

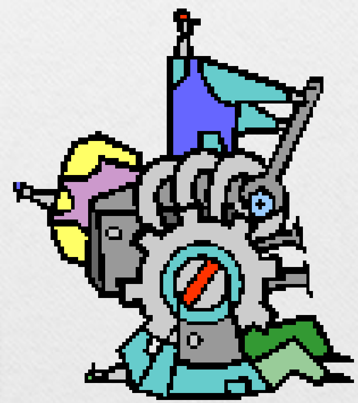
Communication Is Insecure

Alice

Bob

Paparazzi

Secure Approach: Cryptosystems



Alice

Bob

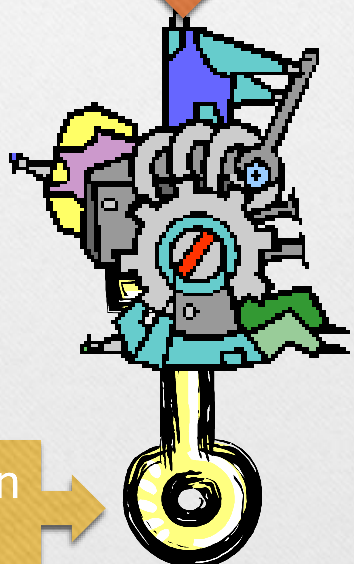
Paparazzi

Cryptosystems

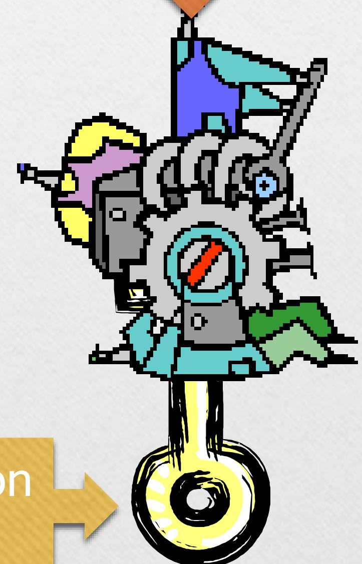
Encryption Scheme

Decryption Scheme

Alice



Encryption Key



Decryption Key

Bob

Encryption vs. Decryption

- Only Bob knows the decryption key.
- Encryption Key
  - Only Alice and Bob know the encryption key: PRIVATE cryptosystem
  - Everyone knows the encryption key: PUBLIC cryptosystem
- RSA is a public cryptosystem.

RSA Algorithm

RSA Cryptosystem

- If Bob wants to use RSA, he needs to select a key pair, and announce the encryption key.
- If Alice wants to communicate with Bob, she needs to use the encryption key announced by Bob.
- If Bob wants to receive messages from the others, he needs to use the decryption key he selected.

How to Select a key pair

- Key pair selection scheme:
  - Bob (randomly) selects 2 prime numbers p and q.
    - For security reason, $p = 2p' + 1$ and $q = 2q' + 1$, where p' and q' are also prime numbers.
  - Bob evaluates $n = pq$ and $\Phi(n) = (p - 1)(q - 1)$
  - Bob chooses e such that $\gcd(e, \Phi(n)) = 1$
  - Bob finds the integer d $(0 < d < \Phi(n))$ such that $ed - k\Phi(n) = 1$
  - Finally, Bob announces the number pair (n, e) and keeps $(d, p, q, \Phi(n))$ in secret.

Euler's totient or **phi function**, $\Phi(n)$ counts the integers between 1 and n that are coprime to n.

$\Phi(p) = p - 1$, $\Phi(q) = q - 1$
$\Phi(pq) = (p - 1)(q - 1)$

How to Encrypt

- Encryption Scheme:
  - Whenever Alice wants to tell Bob m which is less than n, she evaluate $c = m^e \bmod n$, where n and e are the numbers Bob announced.
  - Then Alice sends c to Bob.

How to Decrypt

- Decryption Scheme:
  - Whenever Bob receives an encrypted message c, he evaluate $m' = c^d \bmod n$ Hard to calculate!
  - Fact: $m' = m$
- Why the decryption scheme work?
  - Euler's theorem: if $\gcd(a, n)=1$, $a^{\Phi(n)} \bmod n = 1$
  - $c^d \bmod n = (m^e \bmod n)^d \bmod n = (m^e)^d \bmod n$
    $= m^{ed} \bmod n = m^{k\Phi(n)+1} \bmod n$
    $=(m^k)^{\Phi(n)} m \bmod n = m$

Montgomery Algorithm for RSA-256 bit

Inverse (1/4)

- For real number, x and y are the inverse of each other if
  $$xy = 1$$
  We write $y = x^{-1}$, and vice versa.
- When we say a divided by b, or a / b, we mean that a multiplied by b^{-1}.
- In the "world" of "modulo N," we want to define the inverse (and then the division operator /) such that the exponential laws hold.

Inverse (2/4)

- For a positive integer x ($< N$), We define the inverse of in the "world" of "modulo N" is the positive integer y ($< N$) such that

$$xy \bmod N = 1$$

  We write $y = x^{-1}$, and vice versa.
- We define the "division" in the "world" of "modulo N" as

$$x / y \bmod N = xy^{-1} \bmod N$$

Inverse (3/4)

- Theorem: If $b = an$, then b / a <u>mod N</u> $= n$.
- Example:

  $a = 2$, $N = 35$, then $a^{-1} = 18$

  $b = 12 = 2 * 6$,

  $b / a \bmod N = ba^{-1} \bmod N$

  $= 12 * 18 \bmod 35 = 216 \bmod 35 = 6$

Inverse (4/4)

- Another example:

  $a = 2, N = 35$, then $a^{-1} = 18$

  $b = 13$

  $b\ /\ a \bmod N = ba^{-1} \bmod \mathrm{N}$

  $= 13 * 18 \bmod 35 = 234 \bmod 35 = 24$

  or

  $b\ /\ a \bmod N = (b + N)\ /\ a \bmod N$

  $= (13 + 35\ )\ /\ 2 \bmod 35$

  $= 24$

MSB-Based Modular Multiplication

- We want to evaluate $V \equiv AB \pmod{N}$, where $A = 2^{n-1}a_{n-1} + 2^{n-2}a_{n-2} + \ldots + 2a_1 + a_0$
- We can find that $V \equiv \{2[\ldots(2(2a_{n-1}B + a_{n-2}B) + a_{n-3}B) + \ldots + a_1B] + a_0B\}$
- The Algorithm for MSB-Based Modular Multiplication

$V_n \leftarrow 0$
for $i = n - 1, \ldots, 1, 0$
  $V_i \leftarrow (2V_{i+1} + a_i \cdot B) \bmod N$

$2V_{i+1} + a_iB < 3N$

Square and Multiplication Algorithms for Modular Exponentiation

- Evaluate $S = M^e \bmod N$
  where exponent $e=(1e_{k-2}\ldots e_1e_0)$

No need to be k bit

MSB-ME(M^e mod N)
$S \leftarrow M$
for $i = k-2, \ldots,1,0$
  $S \leftarrow (S \cdot S) \bmod N$
  if ($e_i = 1$) $S \leftarrow (S \cdot M) \bmod N$

LSB-ME(M^e mod N)
$S \leftarrow 1$, $T \leftarrow M$
for $i = 0,1,\ldots, k-1$
  if ($e_i = 1$) $S \leftarrow (S \cdot T) \bmod N$
  $T \leftarrow (T \cdot T) \bmod N$

$(A \cdot B) \bmod N$ is still hard to implement

Montgomery Algorithm

- Idea: Trying to compare V_i with N costs a lot.
- Idea: How about LSB first to evaluate the multiplication?

Montgomery Algorithm: Phase 1
Evaluate $V_n=(A\cdot B\cdot 2^{-n}) \bmod N$

$$A \cdot B \cdot 2^{-n} = B \cdot 2^{-n} \cdot (2^{n-1}a_{n-1}+2^{n-2}a_{n-2}+\ldots+2a_1+a_0)$$

$$= B \cdot (2^{-1}a_{n-1}+2^{-2}a_{n-2}+\ldots+2^{-(n-1)}a_1+2^{-n}a_0)$$

$$= 2^{-1}(a_{n-1}B+2^{-1}(a_{n-2}B+\ldots+2^{-1}(a_1B+2^{-1}a_0B)\ldots))$$

$V_0 \leftarrow 0$
for $i = 0,1,\ldots, n-1$
$V_{i+1} \leftarrow \left(\frac{V_i+a_iB}{2}\right) \bmod N$

$\left(\frac{V_i+a_iB}{2}\right) \bmod N = \frac{V_i+a_iB+q_iN}{2}$,
q_i = LSB of (V_i+a_iB)

LSB modular reduction $\left(\frac{V_i+a_iB}{2}\right) \bmod N$ is **easy**!

Montgomery Algorithm: Phase 2
When to substitute?

$V_0 \leftarrow 0$
for $i = 0,1,\ldots, n-1$
$\quad q_i \leftarrow (V_i + a_i B) \bmod 2$
$\quad V_{i+1} \leftarrow \left(\frac{V_i + a_i B + q_i N}{2}\right)$
if $(V_n \geq N)$ $V \leftarrow V_n - N$

$A=(a_{n-1}a_{n-2}\ldots a_1 a_0)_2$, $A,B<N$

$V_0=0<2N$, $V_{i+1} \leq \left(\frac{V_i + a_i B + N}{2}\right) < 2N$, $i=0,1,\ldots,n-1$

Montgomery Algorithm: Modified Version (1/2)

$$A \cdot B \cdot 2^{-n} = B \cdot 2^{-n} \cdot (2^{n-1}a_{n-1} + 2^{n-2}a_{n-2} + \ldots + 2a_1 + a_0)$$

$$= B \cdot (2^{-1}a_{n-1} + 2^{-2}a_{n-2} + \ldots + 2^{-(n-1)}a_1 + 2^{-n}a_0)$$

$$= 2^{-2}((2a_{n-1} + a_{n-2})B + 2^{-2}((2a_{n-3} + a_{n-4})B + \ldots$$

$$+ 2^{-2}((2a_3 + a_2)B + 2^{-2}(2a_1 + a_0)B)\ldots))$$

$V_0 \leftarrow 0$

for $i = 0, 2, \ldots, n-2$

$\quad V_{i+2} \leftarrow \left(\frac{V_i + 2a_{i+1}B + a_iB}{4}\right) \bmod N$

$$\left(\frac{V_i + 2a_{i+1}B + a_iB}{4}\right) \bmod N = \frac{V_i + 2a_{i+1}B + a_iB + q_iN}{4},$$

$q_i = (k_i = 0)?\ 0 : (4 - k_i)$, $k_i = (V_i + 2a_{i+1}B + a_iB) \bmod 4$

Montgomery Algorithm: Modified Version (2/2)

$V_0 \leftarrow 0$

for $i = 0,2,\ldots, n-2$

$\quad k_i = (V_i + 2a_{i+1}B + a_iB) \bmod 4$

$\quad q_i = (k_i = 0)?\ 0: (4-k_i);$

$$V_{i+2} \leftarrow \frac{V_i + 2a_{i+1}B + a_iB + q_iN}{4}$$

if $(V_n \geq N)$ $V \leftarrow V_n - N$

$A=(a_{n-1}a_{n-2}\ldots a_1a_0)_2$, $A,B<N$

$$V_0=0<2N, \quad V_{i+2} \leq \left(\frac{V_i + 2a_{i+1}B + a_iB + 3N}{4}\right) < 2N, \quad i=0,1,\ldots,n-1$$

Modular Exponentiation Using Montgomery Algorithm (1/2)

- Observation on

$$V_n = \mathrm{MA}(A, B) = (A \cdot B \cdot 2^{-n}) \bmod N$$

  - Define $A' = 2^n A \bmod N$ (A “packed”)
  - Fact: If $V = AB \bmod N$, then $V = \mathrm{MA}(A', B)$
  - Fact: If $V = AB \bmod N$, then $V' = \mathrm{MA}(A', B')$
- Idea: “Pack” the integers we want to evaluate, and use Montgomery Algorithm instead of direct modular multiplication.

Modular Exponentiation Using Montgomery Algorithm (2/2)

- Evaluate $S = M^e \bmod N$

Constant $C = 2^{2n} \bmod N$

MSB-ME($M^e \bmod N$)
$M' \leftarrow \mathrm{MA}(C \cdot M)$ (pre-processing)
$S \leftarrow M'$
for $i = k-2, \ldots, 1, 0$
    $S \leftarrow \mathrm{MA}(S \cdot S)$
    if $(e_i = 1)$ $S \leftarrow \mathrm{MA}(S \cdot M')$
$S \leftarrow \mathrm{MA}(S \cdot 1)$ (post-processing)

LSB-ME($M^e \bmod N$)
$T \leftarrow \mathrm{MA}(C \cdot M)$ (pre-processing)
$S \leftarrow 1$
for $i = 0, 1, \ldots, k-1$
    if $(e_i = 1)$ $S \leftarrow \mathrm{MA}(S \cdot T)$
    $T \leftarrow \mathrm{MA}(T \cdot T)$

MSB-ME($M^e \bmod N$)
$S \leftarrow M$
for $i = k-2, \ldots, 1, 0$
    $S \leftarrow (S \cdot S) \bmod N$
    if $(e_i = 1)$ $S \leftarrow (S \cdot M) \bmod N$

LSB-ME($M^e \bmod N$)
$S \leftarrow 1$, $T \leftarrow M$
for $i = 0, 1, \ldots, k-1$
    if $(e_i = 1)$ $S \leftarrow (S \cdot T) \bmod N$
    $T \leftarrow (T \cdot T) \bmod N$

The End.

Any question?

Reference

- [1] P.L. Montgomery, “Modular multiplication without trial division,”Mathematics of Computation, vol.44, pp.519-521, April 1985.