Number Theory and RSA Problem

Yu Zhang

HIT/CST/NIS

Cryptography, Autumn, 2014

Outline

Content

Primes and Divisibility

- The set of **integers** $\mathbb{Z}$, $a, b, c \in \mathbb{Z}$.
- a **divides** b: $a \mid b$ if $\exists c, ac = b$ (otherwise $a \nmid b$).
  b is a **multiple** of a. If $a \notin \{1, b\}$, then a is a **factor** of b.
- $p > 1$ is **prime** if it has no factors.
- An integer > 1 which is not prime is **composite**.
- **Greatest common divisor** $\gcd(a, b)$ is the largest integer c such that $c \mid a$ and $c \mid b$. $\gcd(0, b) = b$, $\gcd(0, 0)$ undefined.
- a and b are **relatively prime (coprime)** if $\gcd(a, b) = 1$.
- **Euclid's theorem**: there are infinitely many prime numbers.

Modular Arithmetic

- Remainder $r = [a \bmod N] = a - b\lfloor a/b \rfloor$ and $r < N$. N is called **modulus**.
- $\mathbb{Z}_N = \{0, 1, \ldots, N-1\} = \{a \bmod N | a \in \mathbb{Z}\}$.
- a and b are **congruent modulo** N: $a \equiv b \pmod{N}$ if $[a \bmod N] = [b \bmod N]$.
- a is **invertible modulo** $N \iff \gcd(a, N) = 1$. If $ab \equiv 1 \pmod{N}$, then $b = a^{-1}$ is **multiple inverse** of a **modulo** N.
- **Cancellation law**: If $\gcd(a, N) = 1$ and $ab \equiv ac \pmod{N}$, then $b \equiv c \pmod{N}$.
- **Euclidean algorithm**: $\gcd(a, b) = \gcd(b, [a \bmod b])$.
- **Extended Euclidean algorithm**: Given a, N, find X, Y with $Xa + YN = \gcd(a, N)$.

Examples of Modular Arithmetic

"Reduce and then add/multiply" instead of "add/multiply and then reduce".

Compute $193028 \cdot 190301 \bmod 100$

$193028 \cdot 190301 = [193028 \bmod 100] \cdot [190301 \bmod 100] \bmod 100$
$= 28 \cdot 1 \equiv 28 \bmod 100.$

$ab \equiv cb \pmod{N}$ does *not necessarily* imply $a \equiv c \pmod{N}$.

$a = 3, c = 15, b = 2, N = 24$

$3 \cdot 2 = 6 \equiv 15 \cdot 2 \pmod{24}$, but $3 \not\equiv 15 \pmod{24}$.

Use extended Euclidean algorithm to ...

Find the inverse of $11 \pmod{17}$

$(-3) \cdot 11 + 2 \cdot 17 = 1$, so 14 is the inverse of 11.

Groups

A **group** is a set $\mathbb{G}$ with a binary operation $\circ$:

- (**Closure**:) $\forall g, h \in \mathbb{G}$, $g \circ h \in \mathbb{G}$.
- (**Existence of an Identity**:) $\exists$ **identity** $e \in \mathbb{G}$ such that $\forall g \in \mathbb{G}, e \circ g = g = g \circ e$.
- (**Existence of Inverses**:) $\forall g \in G$, $\exists\ h \in \mathbb{G}$ such that $g \circ h = e = h \circ g$. h is an **inverse** of g.
- (**Associativity**:) $\forall g_1, g_2, g_3 \in \mathbb{G}$, $(g_1 \circ g_2) \circ g_3 = g_1 \circ (g_2 \circ g_3)$.

$\mathbb{G}$ with $\circ$ is **abelian** if

- (**Commutativity**:) $\forall g, h \in \mathbb{G}, g \circ h = h \circ g$.

Existence of inverses implies **cancellation law**.
When $\mathbb{G}$ is a **finite group** and $|\mathbb{G}|$ is the **order** of group.

Group Exponentiation

$$g^m \stackrel{\text{def}}{=} \underbrace{g \circ g \circ \cdots \circ g}_{m \text{ times}}.$$

Theorem 1

$\mathbb{G}$ is a finite group. Then $\forall g \in \mathbb{G}, g^{|\mathbb{G}|} = 1$.

Corollary 2

$\forall g \in \mathbb{G}$ and i, $g^i = g^{[i \bmod |\mathbb{G}|]}$.

Corollary 3

Define function $f_e : \mathbb{G} \to \mathbb{G}$ by $f_e(g) = g^e$.
If $\gcd(e, |\mathbb{G}|) = 1$, then f_e is a permutation.
Let $d = [e^{-1} \bmod |\mathbb{G}|]$, then f_d is the inverse of f_e. ($f_d(f_e(g)) = g$)
*e**'th root of** c: $g^e = c$, $g = c^{1/e} = c^d$.*

The Group $\mathbb{Z}_N^*$

$$\mathbb{Z}_N^* \stackrel{\text{def}}{=} \{a \in \{1, \ldots, N-1\} \,|\, \gcd(a, N) = 1\}$$

Euler's phi function: $\phi(N) \stackrel{\text{def}}{=} |\mathbb{Z}_N^*|$.

Theorem 4

$N = \prod_i p_i^{e_i}$, $\{p_i\}$ *are distinct primes,* $\phi(N) = \prod_i p_i^{e_i - 1}(p_i - 1)$.

Corollary 5 (Euler's theorem & Fermat's little theorem)

$a \in \mathbb{Z}_N^*$. $a^{\phi(N)} \equiv 1 \pmod{N}$.
If p *is prime and* $a \in \{1, \ldots, p-1\}$, *then* $a^{p-1} \equiv 1 \pmod{p}$.

Corollary 6

Define function $f_e : \mathbb{Z}_N^* \to \mathbb{Z}_N^*$ *by* $f_e(x) = [x^e \bmod N]$.
If $\gcd(e, \phi(N)) = 1$, *then* f_e *is a permutation.*
Let $d = [e^{-1} \bmod \phi(N)]$, *then* f_d *is the inverse of* f_e.
e***'th root of*** c: $g^e = c$, $g = c^{1/e} = c^d$.

Examples on Groups

- $\mathbb{Z}$ is an abelian group under '$+$', not a group under '$\cdot$'.
- The set of real numbers $\mathbb{R}$ is not a group under '$\cdot$'.
- $\mathbb{R} \setminus \{0\}$ is an abelian group under '$\cdot$'.
- $\mathbb{Z}_N$ is an abelian group under '$+$' modulo N.
- If p is prime, then $\mathbb{Z}_p^*$ is an abelian group under '$\cdot$' modulo p.
- $\mathbb{Z}_{15}^* = \{1, 2, 4, 7, 8, 11, 13, 14\}$, $|\mathbb{Z}_{15}^*| = 8$.
- $\mathbb{Z}_3^*$ is a subgroup of $\mathbb{Z}_{15}^*$, but $\mathbb{Z}_5^*$ is not.
- $2^{1/3} \bmod 5 = 2^3 \bmod 5 = 3$. $(3^{-1} = 3 \pmod 4)$
- g^3 is a permutation on $\mathbb{Z}_{15}^*$, but g^2 is not (e.g., $8^2 \equiv 2^2 \equiv 4$).

$N = pq$ where p, q are distinct primes. $\phi(N) = ?$

$\phi(N) = (N-1) - (q-1) - (p-1) = (p-1)(q-1)$.

Arithmetic algorithms

- **Addition/subtraction**: linear time $O(n)$.
- **Mulplication**: naively $O(n^2)$. Karatsuba (1960): $O(n^{\log_2 3})$
  Basic idea: $(2^b x_1 + x_0) \times (2^b y_1 + y_0)$ with 3 mults.
  Best (asymptotic) algorithm: about $O(n \log n)$.
- **Division with remainder**: $O(n^2)$.
- **Exponentiation**: $O(n^3)$.

Algorithm 1: Exponentiating by Squaring

---

input : $g \in G$; exponent $x = [x_n x_{n-1} \ldots x_2 x_1 x_0]_2$

output: g^x

$y \leftarrow g; z \leftarrow 1$

for $i = 0$ **to** n **do**

    if $x_i == 1$ **then** $z \leftarrow z \times y$

    $y \leftarrow y^2$

return z

Algorithms for Factoring

- **Factoring** $N = pq$. p, q are of the same length n.
- **Trial division**: $\mathcal{O}(\sqrt{N} \cdot \text{polylog}(N))$.
- **Pollard's** $p-1$ method: effective when $p-1$ has "small" prime factors.
- **Pollard's rho** method: $\mathcal{O}(N^{1/4} \cdot \text{polylog}(N))$.
- **Quadratic sieve** algorithm [Carl Pomerance]: sub-exponential time $\mathcal{O}(\exp(\sqrt{n \cdot \log n}))$.
- The best-known algorithm is the **general number field sieve** [Pollard] with time $\mathcal{O}(\exp(n^{1/3} \cdot (\log n)^{2/3}))$.

Content

The RSA Problem

Recall group exponentiation on $\mathbb{Z}_N^*$

Define function $f_e : \mathbb{Z}_N^* \to \mathbb{Z}_N^*$ by $f_e(x) = [x^e \bmod N]$.
If $\gcd(e, \phi(N)) = 1$, then f_e is a permutation.
If $d = [e^{-1} \bmod \phi(N)]$, then f_d is the inverse of f_e.
e**'th root of** c: $g^e = c$, $g = c^{1/e} = c^d$.

Idea: factoring is hard

$\implies$ for $N = pq$, finding p, q is hard

$\implies$ computing $\phi(N) = (p-1)(q-1)$ is hard

$\implies$ computations modulo $\phi(N)$ is not available

There is a gap.

$\implies$ **RSA problem** [Rivest, Shamir, and Adleman] is hard:
Given $y \in \mathbb{Z}_N^*$, compute y^{-e}, e^{th}-root of y modulo N.

Open problem

RSA problem is easier than factoring?

Generating RSA Problem

Algorithm 2: GenRSA

input : Security parameter 1^n

output: N, e, d

$(N, p, q) \leftarrow \text{GenModulus}(1^n)$
$\phi(N) := (p-1)(q-1)$
find e such that $\gcd(e, \phi(N)) = 1$
compute $d := [e^{-1} \bmod \phi(N)]$
return N, e, d

The RSA Assumption

The RSA experiment $\mathsf{RSAinv}_{\mathcal{A},\mathsf{GenRSA}}(n)$:

1. Run $\mathsf{GenRSA}(1^n)$ to obtain (N, e, d).
2. Choose $y \leftarrow \mathbb{Z}_N^*$.
3. $\mathcal{A}$ is given N, e, y, and outputs $x \in \mathbb{Z}_N^*$.
4. $\mathsf{RSAinv}_{\mathcal{A},\mathsf{GenRSA}}(n) = 1$ if $x^e \equiv y \pmod{N}$, and 0 otherwise.

Definition 7

RSA problem is hard relative to GenRSA if $\forall$ PPT algorithms $\mathcal{A}$, $\exists$ negl such that

$$\Pr[\mathsf{RSAinv}_{\mathcal{A},\mathsf{GenRSA}}(n) = 1] \leq \mathsf{negl}(n).$$

Constructing One-Way Functions

Algorithm 3: Algorithm computing $f_{\mathsf{GenModulus}}$

input : String x

output: String N

compute n such that $p(n) \leq |x| < p(n+1)$

compute $(N, p, q) := \mathsf{GenModulus}(1^n; x)$

`/* run` $\mathsf{GenModulus}(1^n)$ `using` x `as the random tape */`

return N

Reduce the factoring problem to the inverting problem.

Theorem 8

If the factoring problem is hard relative to GenModulus, *then* $f_{\mathsf{GenModulus}}$ *is a one-way function.*

Constructing One-Way Permutations

Construction 9

Define a family of permutations with GenRSA*:*

- Gen*: on input* 1^n*, run* GenRSA(1^n) *to obtain* (N, e, d) *and output* $I = \langle N, e\rangle$*, Set* $\mathcal{D}_I = \mathbb{Z}_N^*$*.*
- Samp*: on input* $I = \langle N, e\rangle$*, choose a random elements of* $\mathbb{Z}_N^*$*.*
- f*: on input* $I = \langle N, e\rangle$ *and* $x \in \mathbb{Z}_N^*$*, output* $[x^e \bmod N]$*.*

Reduce the RSA problem to the inverting problem.

Content

“Textbook RSA”

Construction 10

- Gen: *on input* 1^n *run* $\mathsf{GenRSA}(1^n)$ *to obtain* N, e, d. $pk = \langle N, e \rangle$ *and* $sk = \langle N, d \rangle$.
- Enc: *on input* pk *and* $m \in \mathbb{Z}_N^*$, $c := [m^e \bmod N]$.
- Dec: *on input* sk *and* $m \in \mathbb{Z}_N^*$, $m := [c^d \bmod N]$.

Insecurity

Since the “textbook RSA” is deterministic, it is insecure with respect to any of the definitions of security we have proposed.

RSA Implementation Issues

- **Encoding binary strings as elements of** $\mathbb{Z}_N^*$: $\ell = \|N\|$. Any binary string m of length $\ell - 1$ can be viewed as an element of Z_N. Although m may not be in Z_N^*, RSA still works.
- **Choice of** e: Either $e = 3$ or a small d are bad choices. Recommended value: $e = 65537 = 2^{16} + 1$
- **Using the Chinese remainder theorem**: to speed up the decryption.

$$[c^d \bmod N] \leftrightarrow ([c^d \bmod p], [c^d \bmod q]).$$

Assume that exponentiation modulo a v-bit integer takes v^3 operations. RSA decryption takes $(2n)^3 = 8n^3$, whereas using CRT takes $2n^3$.

Example of "Textbook RSA"

$N = 253$, $p = 11$, $q = 23$, $e = 3$, $d = 147$, $\phi(N) = 220$.

$m = 0111001 = 57$.
Encryption: $250 := [57^3 \bmod 253]$.
Decryption: $57 := [250^{147} \bmod 253]$.

Using CTR,

$$[250^{[147 \bmod 10]} \bmod 11] = [8^7 \bmod 11] = 2$$

$$[250^{[147 \bmod 22]} \bmod 23] = [20^{15} \bmod 23] = 11$$

$57 \leftrightarrow (2, 11)$.

Attacks on "Textbook RSA" with a small e

Small e and small m make modular arithmetic useless.

- If $e = 3$ and $m < N^{1/3}$, then $c = m^3$ and $m = c^{1/3}$.
- In the hybrid encryption, 1024-bit RSA with 128-bit DES.

A general attack when small e is used:

- $e = 3$, the same message m is sent to 3 different parties.
- $c_1 = [m^3 \bmod N_1]$, $c_2 = [m^3 \bmod N_2]$, $c_3 = [m^3 \bmod N_3]$.
- N_1, N_2, N_3 are coprime, and $N^* = N_1 N_2 N_3$, $\exists$ unique $\hat{c} < N^*$: $\hat{c} \equiv c_1 \pmod{N_1}$, $\hat{c} \equiv c_2 \pmod{N_2}$, $\hat{c} \equiv c_3 \pmod{N_3}$.
- With CRT, $\hat{c} \equiv m^3 \pmod{N^*}$. Since $m^3 < N^*$, $m = \hat{c}^{1/3}$.

Common Modulus Attacks

Common Modulus Attacks: the same modulus N.

Case I: for multiple users with their own secret keys.
Each user can find $\phi(N)$ with his own e, d, then find others' d.

Case II: for the same message encrypted with two public keys.
Assume $\gcd(e_1, e_2) = 1$, $c_1 \equiv m^{e_1}$ and $c_2 \equiv m^{e_2} \pmod N$.
$\exists X, Y$ such that $Xe_1 + Ye_2 = 1$.

$$c_1^X \cdot c_2^Y \equiv m^{Xe_1} m^{Ye_2} \equiv m^1 \pmod N.$$

CCA in "Textbook RSA" Encryption

Recovering the message with CCA

$\mathcal{A}$ choose a random $r \leftarrow \mathbb{Z}_N^*$ and compute $c' = [r^e \cdot c \bmod N]$, and get m' with CCA. Then $m = [m' \cdot r^{-1} \bmod N]$.

$$m' \cdot r^{-1} \equiv (c')^d r^{-1} \equiv (r^e \cdot m^e)^d r^{-1} \equiv r^{ed} m^{ed} r^{-1} \equiv rmr^{-1} \equiv m.$$

Doubling the bid at an auction

The ciphertext of an bid is $c = [m^e \bmod N]$. $c' = [2^e c \bmod N]$.

$$(c')^d \equiv (2^e m^e)^d \equiv 2^{ed} m^{ed} \equiv 2m.$$

Content

Padded RSA

Idea: add randomness to improve security.

Construction 11

Let ℓ be a function with $\ell(n) \leq 2n-2$ for all n.

- Gen*: on input 1^n, run* GenRSA(1^n) *to obtain (N, e, d). Output $pk = \langle N, e \rangle$, and $sk = \langle N, d \rangle$.*
- Enc*: on input $m \in \{0,1\}^{\ell(n)}$, choose a random string $r \leftarrow \{0,1\}^{\|N\|-\ell(n)-1}$. Output $c := [(r\|m)^e \bmod N]$.*
- Dec*: compute $\hat{m} := [c^d \bmod N]$, and output the $\ell(n)$ low-order bits of $\hat{m}$.*

ℓ should neither be too large (r is too short in theory) nor be too small (m is too short in practice).

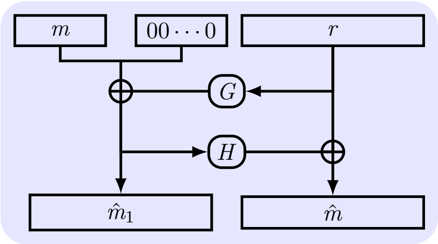
Theorem 12

If the RSA problem is hard relative to GenRSA*, then Construction with $\ell(n) = \mathcal{O}(\log n)$ is CPA-secure.*

PKCK #1 v2.1 (RSAES-OAEP)

Optimal Asymmetric Encryption Padding (OAEP): encode m of length $n/2$ as $\hat{m}$ of length $2n$. G, H are **Random Oracles**.

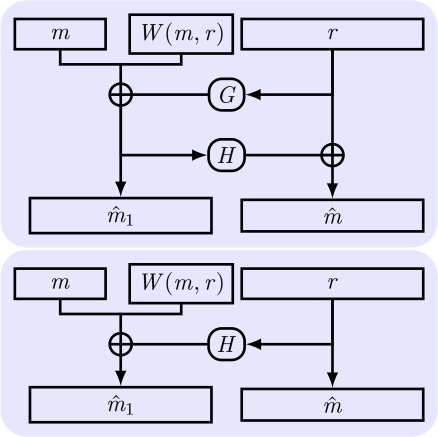
$$\hat{m}_1 := G(r) \oplus (m \| \{0\}^{n/2}), \hat{m} := \hat{m}_1 \| (r \oplus H(\hat{m}_1)).$$



RSA-OAEP is CCA-secure in Random Oracle model. [1] [RFC 3447]

[1] It may not be secure when RO is instantiated.

OAEP Improvements



OAEP+: $\forall$ trap-door permutation F, F-OAEP+ is CCA-secure.

SAEP+: RSA (e=3) is a trap-door permutation, RSA-SAEP+ is CCA-secure.

W, G, H are Random Oracles.

Remarks on RSA in Practice

Key lengths with comparable security :

Symmetric	RSA
80 bits	1024 bits
128 bits	3072 bits
256 bits	15360 bits

Implementation attacks:

Timing attack: The time it takes to compute c^d can expose d.

Power attack: The power consumption of a smartcard while it is computing c^d can expose d.

Key generation trouble (in OpenSSL RSA key generation):
Same p will be generated by multiple devices (due to poor entropy at startup), but different q (due to additional randomness).
N_1, N_2 from different devices, $\gcd(N_1, N_2) = p$.
Experiment result: factor 0.4% of public HTTPS keys.

Faults Attack on RSA

Faults attack: A computer error during $c^d \bmod N$ can expose d.

Using CRT to speed up the decryption:

$$[c^d \bmod N] \leftrightarrow ([m_p \equiv c^d \pmod p], [m_q \equiv c^d \pmod q]).$$

Suppose error occurs when computing m_q, but no error in m_p.

Then output is m' where $m' \equiv c^d \pmod p$, $m' \not\equiv c^d \pmod q$.
So $(m')^e \equiv c \pmod p$, $(m')^e \not\equiv c \pmod q$.

$$\gcd((m')^e - c, N) = p.$$

A common defense: check output. (but 10% slowdown)

Summary

- Primes, modular arithmetic.
- e^{th}-root modulo N, RSA.

Textbook

"*A Computational Introduction to Number Theory and Algebra*" (Version 2) by Victor Shoup

- RSA, "textbook RSA", padded RSA, PKCS.
- small e, common modulus attacks, CCA, faults attack.