S P Jain
School of Global
Management
DUBAI • MUMBAI • SINGAPORE • SYDNEY

RSA Cryptosystem

By Group Kappa

Thanh Nguyen	-BS20DSY029
Anh Ngo	-BS20DSY035
Minh Le	-BS20DSY033
Dang Luong	-BS20DSY034

BACHELOR OF DATA SCIENCE 2020

MATHEMATICS GROUP PROJECT

I. Introduction

Cryptography has been playing an important role in our life when humans started exchanging information. It is the encryption of text in such a method that outsiders to the code cannot understand the code, but the desired reader is able to decrypt the encryption to comprehend the message. However, before the 20th century, only symmetric-key cryptography (both communicating parties know the key) was used in encoding and decoding, leading to a problem: key distribution. That is the two communicating parties may already be sharing the key which has been distributed to them by any means or the key must be shared with the help of a key distribution center. But, using of key distribution center compromises the secrecy of the key which hampers confidentiality of the message. This problem leads to the evolution of asymmetric-key cryptography, applied to a new cryptosystem called RSA.

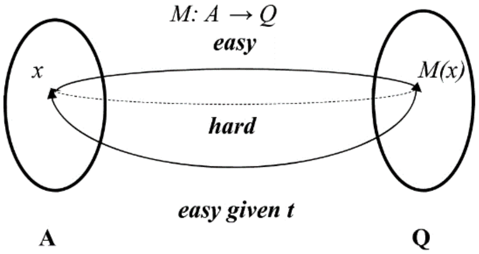
II. History

RSA is the first successful asymmetric cryptographic algorithm, which contains two distinct keys: the public key (encryption key) is circulated or published to all and hence others are aware of and the private key (decryption key) secretly kept with the user only. The discovery was made by three researchers: Rivest, Shamir and Adleman. Rivest and Shamir spent a year coming up with ideas, and Adleman spent a year shooting them down. In April 1977, Rivest, Shamir, and Adleman spent Passover at the house of a student and consumed liberal quantities of Manischewitz wine before returning to their respective homes sometime around midnight. Rivest wondered whether it is possible to find a one-way function that can be reversed only if the receiver has some special information. Suddenly, the ideas began to be clear and he had a revelation. He spent the rest of the night formalizing his idea, and by the next morning he had effectively written a complete mathematical paper. The breakthrough was Rivest’s, but it could not have come without the help of Shamir and Adleman. The system was later named RSA, standing for names: Rivest, Shamir, and Adleman.

III. RSA Algorithm

III.1. Operation

Clifford Cocks, a British mathematician and cryptographer, has constructed a special kind of one-way function called a “trapdoor” one-way function, which is easy to compute in one direction, but difficult to reverse, unless we have special information, called the trapdoor.



For this, he turned to modular exponentiation: take a number, raise it to some exponent, divide by the modulus and output the remainder.

This can be used to encrypt a message as follows: for example, B want to send a message to A, which is number *m*. B then raises m to the power of *e*, a public exponent, then divides the result by a number, *n*, and outputs *c* as the remainder:

$$m^e \; mod \; n = c \; (1)$$

This calculation is easy to perform.

However, given only *c*, *e* and *n*, it is much more difficult to reverse or determine which *m* was used:

$$?^e = c \; (mod \; n)$$

So, this is our one-way function that we can apply to *m*:

$$\xrightarrow{\text{Easy}}$$

$$m^e \; mod \; n = \; ?$$

$$\xleftarrow{\text{Difficult}}$$

$$?^e \; mod \; n = c$$

What the key? The key is the trapdoor that makes it easy to reverse the encryption. We need to raise *c* to some other exponent, *d*, which will undo the initial operation applied to *m* and return the original message *m*:

$$c^d \ mod\ n = m \ (2)$$

So, from functions (1) and (2), it is the same as *m* to the power of *e*, all raised to the power of *d*:

$$(1)(2) \rightarrow (m^e)^d \equiv m \ (mod\ n)$$

$$\rightarrow m^{ed} \equiv m \ (mod\ n)$$

Therefore, we need a way for A to construct *e* and *d*, which makes it difficult for the others to find *d*. This requires a second one-way function which is used for generating *d*.

III.2. Key generation

Step 1: Choose *p* and *q*: $(p, q) = 1$; p, q are prime numbers

Step 2: Compute: $n = p$

Step 3: Find $\varphi(n)$: $\varphi(n) = (p - 1)(q - 1)$

In Euler's totient theorem:

$\varphi(n)$ is the notation for the number of numbers that are coprime to and less than n (numbers that are smaller than *n* and do not share the same divisor greater than 1 with *n*)

Given a number *n*: $n = p_1^{k_1} p_2^{k_2} \dots p_r^{k_r}$, where $p_1, p_2, \dots, p_r$ are prime numbers dividing *n*, then:

$$\varphi(n) = p_1^{k_1-1}(p_1 - 1)p_2^{k_2-1}(p_2 - 1) \dots p_r^{k_r-1}(p_r - 1)$$

For example:

$n = 8 = 2^3$, then $\varphi(8) = 2^{3-1}(2 - 1) = 4$ (*There are 4 numbers that are less than 8 and do not share the same divisors larger than 1 with 8: 1, 3, 5, 7*).

$n = 37$, then $\varphi(37) = 37^{1-1}(37 - 1) = 36$ *(There are 36 numbers coprime to and less than 37).*

Now we return to step 3:

Since p is a prime number, then: $\varphi(p) = (p - 1)$

Same with prime number q: $\varphi(q) = (q - 1)$

$\varphi(n)$ is a multiplicative function (in Euler's totient theorem), then:

$$\varphi(n) = \varphi(pq) = \varphi(p)\ \varphi(q) = (p - 1)(q - 1)$$

Step 4: Choose e: $1 < e < \varphi(n), (e, \varphi(n)) = 1$

Step 5: Find d: $de \equiv 1 \ (mod\ \varphi(n))$

Why $(e, \varphi(n)) = 1?$

Return to the condition of solving a math problem:

Find x: $ax \equiv b\ (mod\ k)$ (a, b, k are constants).

If x exists, then $gcd(a, k) \mid b$, or in words, the largest common divisor of a and k have to be divided by b. Now, look at the formula in step 5.

The condition $(e, \varphi(n)) = 1$ satisfy $(e, \varphi(n)) \mid 1$, then there exists some d.

Example:

Step 1: Choose p and q.

$p = 3$ and $q = 7$ (3 and 7 are different prime numbers).

Step 2: Multiply p and q.

$$n = pq = 21$$

Step 3: Find $\varphi(n)$.

$$\varphi\ (n) = (p - 1)(q - 1) = (3 - 1)(7 - 1) = 12$$

Step 4: Choose e.

$$e = 5\ (1 < 5 < \varphi(n) = 12; gcd(5, 12) = 1)$$

Step 5: Find d:

$$5d \equiv 1\ (mod\ 12) \rightarrow d = 5$$

III.3. Encryption and Decryption

We discussed clearly about encryption and decryption in III.1. Let us take an example:

We found $n = 21$, $e = 5$, and $d = 5$ in III.2 example. B converts the message into number m, say $m = 17$, then he takes the remainder c: $17^5 \bmod 21 = 5$ and sends "5" to A. After that, A uses $d = 5$ to find m: $5^5 \bmod 21 = 17 = m$.

- ***Condition for m***

Another example, $m = 69$. We would get $c = 6$, then A compute m: $m = 6^5 \bmod 21 = 6$. Here we raise an answer, why B sends 69 but A gets 6?

Not only when $m = 69$, the problem occurs also when $m > n$. In fact, in modulo arithmetic, we may view "m mod n" as the remainder m when divided by n. For example, $69 \bmod 21 = 6$ is also written as $69 \equiv 6 \bmod 21$. Now, when we are guaranteed by $c^d \equiv m \pmod n$, we just guaranteed that c^d is equivalent to m modulo n *but not precisely equal. Therefore, if* $m < n$, we are going to get exact m, however, if $m > n$, you are going to get number that is smaller than and equivalent to m modulo n. Since then, we can conclude a condition for m when using RSA: $\boldsymbol{m < n}$**.**

- **Explanation:** *"Why* $c^d = m \pmod n$*?".*

In III.1, we know that if A wants to get m back, he will *find the remainder when n devises c to the power d* or $c^d \equiv m \pmod n$. But why he can follow that?

In the fifth step in key generation:

$$ed \equiv 1 \pmod{\varphi(n)}$$

By Euler's totient function:

$$ed \equiv 1 \pmod{(p - 1)(q - 1)}$$

Hence, there exists some integer k such that:

$$ed = 1 + k(p - 1)(q - 1)$$

Back to functions (1)(2)(III.1), we have:

$$c^d = (m^e)^d = m^{ed} = m^{1+k(p-1)(q-1)} \equiv m \pmod n$$

Assume $(m, p) = 1$ and $(m, q) = 1$, by Fermat's Little Theorem [(3)]:

$$m^{p-1} \equiv 1 \ (mod\ p)$$

$$m^{q-1} \equiv 1 \ (mod\ q)$$

Using these two above functions, we have:

$$c^d \equiv m^{1+k(p-1)(q-1)} = m.(m^{p-1})^{k(q-1)} \equiv m.1 \equiv m \ (mod\ p)$$

$$c^d \equiv m^{1+k(p-1)(q-1)} = m.(m^{q-1})^{k(p-1)} \equiv m.1 \equiv m \ (mod\ q)$$

Since $(p, q)=1$, by Chinese remainder Theorem [(4)]:

$$c^d \equiv m \ (mod\ pq)$$

$$or\ c^d \equiv m \ (mod\ n)$$

[(3)] Fermat's Little Theorem:

It states that if p is a prime number, then for any integer a, the number a^p-a is an integer multiple of p.

We denote: $a^p \equiv a \ (mod\ p)$.

Because $(a, p) = 1$, we have $a^{p-1} \equiv 1 \ (mod\ p)$.

[(4)] Chinese remainder theorem:

It states that if one knows the remainders of the Euclidean division of an integer n by several integers, then one can determine uniquely the remainder of the division of n by the product of these integers, under the condition that the divisors are pairwise coprime.

- <u>General case</u>:

Given $(n_1, n_2) = 1$, $(n_1, n_3) = 1$ and $(n_2, n_3) = 1$

$$x \equiv b_1 \ (mod\ n_1)$$

$$x \equiv b_2 \ (mod\ n_2)$$

$$x \equiv b_3 \ (mod\ n_3)$$

$$N = n_1n_2n_3$$

$$N_i = N/n_i$$

x_i: modular multiplication inverse of N_i ($N_ix_i \equiv 1 \pmod{n_i}$)

b_i	N_i	x_i	$b_iN_ix_i$
b_1	$N_1 = n_2n_3$	x_1	$b_1N_1x_1$
b_2	$N_2 = n_1n_3$	x_2	$b_2N_2x_2$
b_3	$N_3 = n_1n_2$	x_3	$b_3N_3x_3$

The solution is $x = \sum_{i=1}^{3} b_iN_ix \pmod{N}$.

- *Special case:*

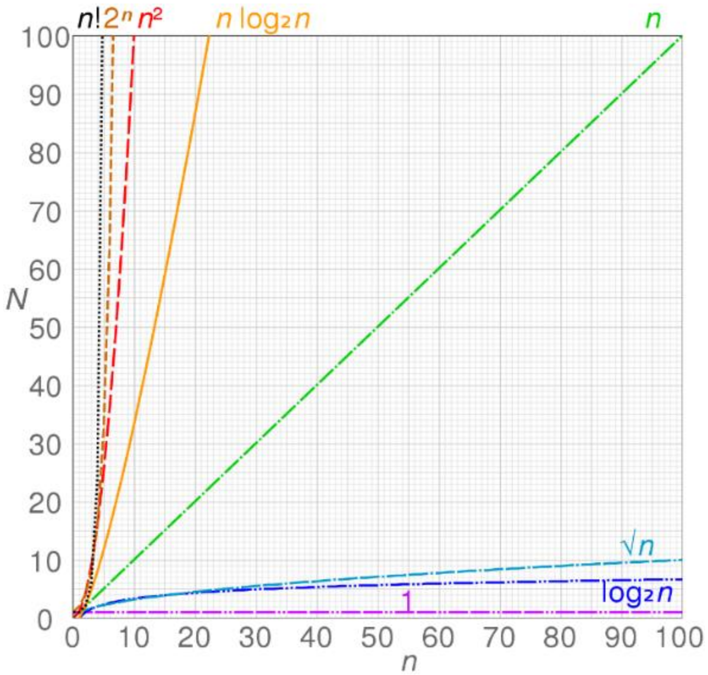
Suppose $(p, q) = 1$. If $x \equiv a \pmod p$ and $x \equiv a \pmod q$, then $x \equiv a \pmod{pq}$.

IV. Security

The process of creating keys is not time-consuming to follow, however, why RSA was chosen to be one of the ways for people to protect their information when exchanging them via the Internet?

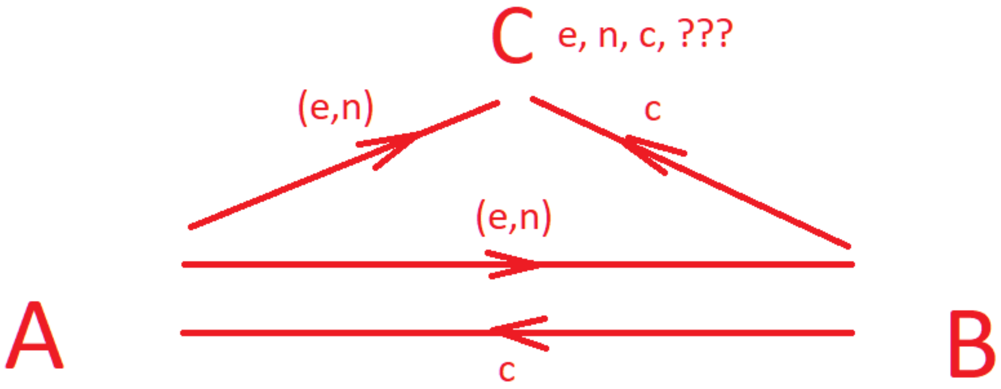
RSA cryptography depends strictly on the complexity of finding two prime divisors of a number. As in step 1 of generating keys, we need to define 2 numbers *p* and *q*, such that they are both prime numbers. When p and q multiply with each other, we have an n, which has only 2 divisors *p* and *q*. When we are given that *n* but not *q* or *p*, if *n* is a small number, for example, 21, we can easily and quickly find 2 prime divisors of it: 3 and 7. However, it will take more time to find these *p* and *q* when n increases then becomes a relatively large number. What are prime divisors of 523099?

Will computers find difficulty when solving this problem: given *n = 523099,* find their *p* and *q* (prime divisors)? The answer is “No!”. They can compute the exact result (632 and 829) in approximately 0.005 seconds, 300.000 times faster than a normal human can do. However, computers also face up to the same problem as us. The graph illustrates the computational complexity of mathematical operation:



As numbers and complexity increase, computers need more time to come up with results.

As in the previous part, we need a way to construct e and d, which makes it difficult for anyone else to find d. And "time complexity" above is the key to this difficulty. Suppose there is another person, say C. He can absorb the information that A and B sent to each other. All he can get after the exchanging are *e, n* and *c*, then he has to find *d*, using $ed \equiv 1 \pmod{\varphi(n)}$, then finds *m* directly. However, when *n* is super large, C and his computer need such a large amount of time, which will be several years, to find either *p, q, or φ(n)*. When he finishes computing the result, the cryptographer may even improve the cipher or the information he gets may no longer important.



Therefore, when creating the key, a cryptographer has to define *p* and *q* that are large enough, find d using the process as in 2 then hide all of them. But what is large enough *p* and *q*?

As time passing by, more and more advanced technology is applied in cipher decryption. As the result, cryptographers must improve their encrypting method. In RSA, all they have to do is increasing p and q to a particular “bit length”. In 2003, RSA Security announced that a 1024-bit long number cannot be broken until 2020, 2048-bit long number may be used after that until 2030, and after 2030, storing information in 3072 - bit long number will definitely safe.

IV. Another application: Digital signature

1. Definition

Digital signature is a form of electronic signature, which uses the Public Key Infrastructure. It requires each user to have a public key and a private key. The private key is used to create the digital signature while the public key is used to authenticate and verify it.

2. Creating and verifying

RSA is used in the field of digital signature. The digital signature is created using an one-way hash function[5] on the original message. This process creates a message digest (or fingerprint).

Then the message digest is encrypted by the private key and the whole thing is attached to the original message. Both are sent to the receiver. When the receiver receives the file, the original message is once again hashed using the same hash function, the public key is used to decrypt the digital signature and get the message digest. He then compares the two message digests and check if they match. If they do, then the message is verified.

[5] *A hash function is any function that is used to turn data of any size (message) into fixed-size values (digest). The hash function is a type of one-way function, which means it is impossible to invert it in practice. Therefore, the hash function is mainly used for comparison purposes, not for encryption.*

A message is an input for the hash function. The hash function will first cut the message into equal blocks. If the bit length of the message is not a multiple of the output hash-code, then a technique called padding will add nonsense values to the message to satisfy the condition. After this process, the hash algorithm runs through the blocks to give the final hash-code.

Properties of hash function: nearly irreversible and unique (every message gives out its own hash-code, so it is unlikely for a hash function to create the same hash-code for two different messages).

3. Properties

a. Authentication

The digital signature is widely used in e-commerce to verify the sender. The document does not need full encryption, but the hash-code of that document only (usually with fixed length and shorter than the document itself, reducing the cost of encrypting). The receiver only needs to decrypt the fingerprint and hash the original document, then compare the two of them. If they match then it can be assured that the document comes from the owner of the private key.

b. Non-repudiation

When the sender has signed in the message, he cannot deny that the signature is not his. To prevent the situation that one denies having sent one message, the sender may require a message with a digital signature. Should there be a conflict, this signature may be used as proof for a third party to deal with the situation.

c. Integrity

To confirm the integrity of the message, whether or not it has been changed during transmitting. Since a slightest change in the document can result in a great change in the hash-code and be detected easily. During the encrypting process, the context will be hidden from a third party. The hash function makes it complicated for any third party to change the document.

REFERENCES

- en.wikipedia.org (2020). Wikipedia Website. [online] Available at: https://en.wikipedia.org/wiki/Fermat%27s_little_theorem. (Accessed: 24 December 2020)
- en.wikipedia.org (2020). Wikipedia Website. [online] Available at: https://en.wikipedia.org/wiki/Chinese_remainder_theorem. (Accessed: 24 December 2020)
- en.wikipedia.org (2020). Wikipedia Website. [online] Available at: https://en.wikipedia.org/wiki/Modular_multiplicative_inverse. (Accessed: 24 December 2020)
- khanacademy.org (2020). Khan Academy Website. [online] Available at: https://www.khanacademy.org/computing/computer-science/cryptography#modern-crypt. (Accessed: 24 December 2020)
- tailieu.vn (2020). TaiLieu Website. [online] Available at: https://tailieu.vn/doc/luan-van-nghien-cuu-mot-so-chu-ky-so-dac-biet-va-ung-dung-1230603.html. (Accessed: 24 December 2020)
- en.wikipedia.org (2020). Wikipedia Website. [online] Available at: https://en.wikipedia.org/wiki/RSA_(cryptosystem). (Accessed: 24 December 2020)
- en.wikipedia.org (2020). Wikipedia Website. [online] Available at: https://en.wikipedia.org/wiki/Euler%27s_totient_function. (Accessed: 24 December 2020)
- researchgate.net (2020). ResearchGate Website. [online] Available at: https://www.researchgate.net/publication/318729097_RSA_Public_Key_Cryptography_Algorithm_-_A_Review. (Accessed: 24 December 2020)
- kdientu.duytan.edu.vn (2020). Duy Tan University Website [online] Available at: http://kdientu.duytan.edu.vn/vi-vn/hoc-lieu/phuong-phap-thiet-ke-ma-cong-khai-rsa/. (Accessed: 24 December 2020)
- ieexplore.ieee.org (2020). IEEE Xplore Website. [online] Available at: https://ieeexplore.ieee.org/document/6021216. (Accessed: 24 December 2020)
- searchsecurity.techtarget.com (2020). SearchSecurity Website. [online] Available at: https://searchsecurity.techtarget.com/definition/RSA?fbclid=IwAR1R5f1ly3hQiR9GSM0yq9Xy-ZadHR5qYSjwVqEE2BfmR-acB5atlUgbouY#:~:text=The%20RSA%20algorithm%20is%20the,network%20such%20as%20the%20internet. (Accessed: 24 December 2020)
- binaryterms.com (2020). Binary Terms Website. [online] Available at: https://binaryterms.com/rsa-algorithm-in-cryptography.html?fbclid=IwAR1kESWgvX8ws3n3EQIpnFHXp3d2vAp0gzAu3O0FsLvlA-bjWDnOIGprRcY. (Accessed: 24 December 2020)
- slideshare.net (2020). Slideshare Website. [online] Available at: https://www.slideshare.net/trongthuy3/luan-van-xay-dung-chuong-trinh-ma-hoa-va-giai-ma-rsa-hot. (Accessed: 24 December 2020)