RSA ENCRYPTION: BEHIND THE SCENES

PETRO KOLOSOV

Abstract. Simple explanation on the symmetric encryption problematics and main idea behind the Rivest-Shamir-Adleman (RSA) encryption.

Contents

1. RSA Encryption: General Idea

The RSA algorithm is named after Ron **Rivest**, Adi **Shamir** and Len **Adleman** who invented it in 1977, see [1]. The basic technique was first discovered in 1973 by Clifford Cocks [2] of CESG (part of the British GCHQ) but this was a secret until 1997. The patent taken out by RSA Labs has expired.

Historically, the process of encryption is considered to be symmetric one.

> **Symmetric encryption** – is a type of encryption where only one secret key is used to both encrypt and decrypt information.

Date: December 8, 2024.

Key words and phrases. RSA encryption, Rivest-Shamir-Adleman, Asymmetric encryption, Symmetric encryption, Public key cryptography, Private key cryptography, Diffie-Hellman key exchange, One-way functions, Cryptographic protocols, Public and private keys, Key exchange problem, Encryption history, Cryptographic algorithms,

It means that prior the communication the sides must conclude and share the secret key to be used in both encryption and decryption. Such approach is highly cost since it requires to share the defined secret keys between each actor.

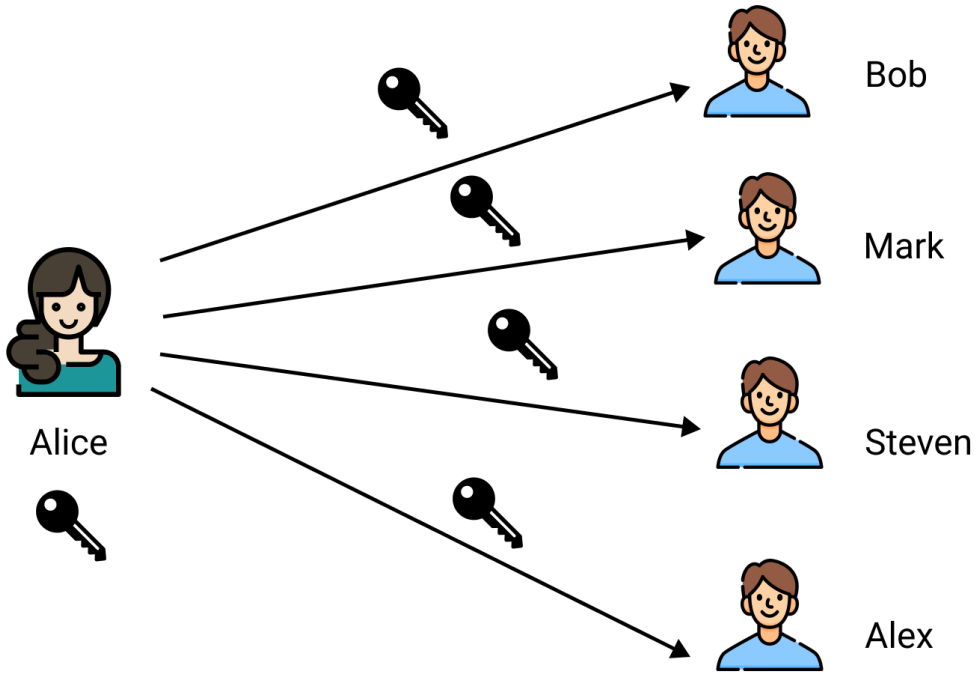

Figure 1. Symmetric encryption real life example.

The problem here is that Alice, Bob, Mark, Steven and Alex must exchange the secret keys securely, for instance by means of Diffie-Hellman key exchange.

Much more simpler is to think about secured communication channel that in terms of asymmetric encryption.

> **Asymmetric encryption** – is an encryption such that a message is encrypted using public key and decrypted using private key.

Real life example would be if Alice shares with all the actors not the secret key, but **opened lock**. Still Alice keeps the secret **private key** with herself, but now she doesn't worry that intermediate eavesdropper would read her precious messages.

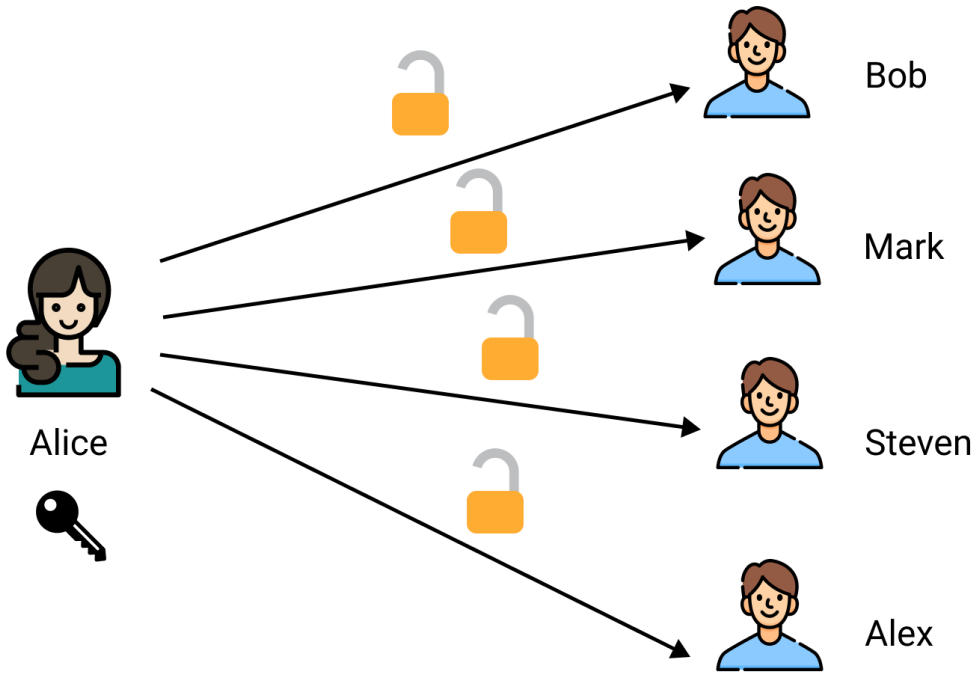

Figure 2. Asymmetric encryption real life example.

Therefore, the guys Bob, Mark, Steven and Alex have received an **opened lock** or **public key** from Alice. Now all of them is can send a secret encrypted message to Alice simply putting it to the chest closing by the lock (public key) received from Alice so that only Alice can open it with her private key. However, such a simple idea requires complex mathematical background. A concept of opened lock may be interpreted in terms of one-way functions.

> **One-way function** – is a function that is easy to compute on every input, but hard to invert given the output.

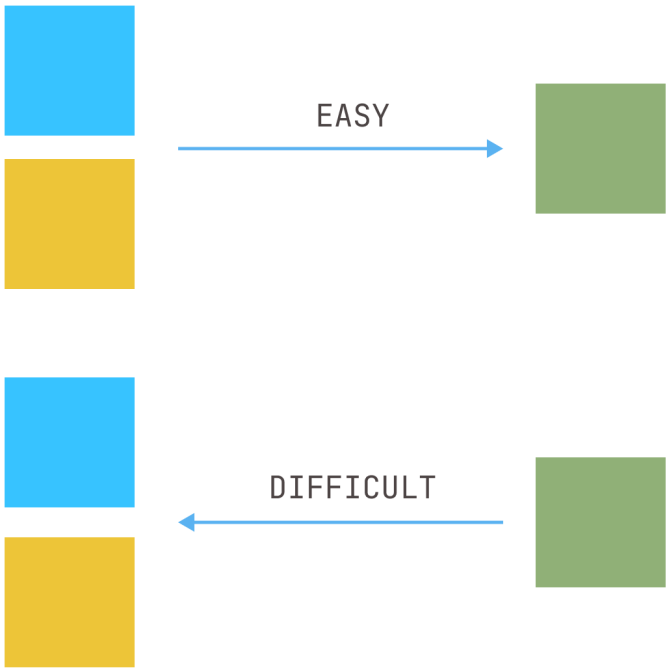

Figure 3. One-way function, analogy with paints

2. Mathematics behind the RSA

For instance, the function

$$f(m) = m^e \bmod N = C$$

$f(m) = C$ is encrypted secret message m

N is product of two private large prime numbers P, Q

(e, N) are public constants (public key)

m is a secret message

is a one-way function because it is easy to compute C given m, but it is hard to compute m given C. The constants (e, N) may be interpreted as an Alice's opened lock (public key), whereas m is a secret message from Bob. Note that constants

- e is stands for encryption, is a part of public key
- d is stands for decryption, calculated using private key

Now the only problem remains for Alice – is to define a pair of constants (e, d). Alice knows that Bob encrypts his message m, using the public key (e, N) as follows

$$m^e \bmod N = C$$

where C is encrypted message. To decrypt the message C Alice must fetch a constant d such that reverts the exponentiation of the secret message m

$$C^d = m \bmod N$$

$$m^{ed} \bmod N = m \bmod N$$

We know that Alice defined a public constant N as a product of two large prime numbers P, Q

$$N = P \cdot Q$$

so that it is hard to compute its factorization.

But how to fetch the secret constant d to decrypt? The Euler's totient function helps. Given a number M and its prime factorization $p_1^{e_1} \cdot p_2^{e_2} \cdots p_k^{e_k}$, the Euler's totient function $\phi(M)$ is defined as

$$\phi(M) = (p_1^{e_1} - p_1^{e_1-1}) \cdot (p_2^{e_2} - p_2^{e_2-1}) \cdots (p_k^{e_k} - p_k^{e_k-1})$$

Given the public key (e, N), for the positive number N such that its factorization is $P \cdot Q$, the $\phi(N)$ is

$$\phi(N) = (P - 1) \cdot (Q - 1)$$

Euler's theorem relates the modular division and exponent as follows. Given number m then

$$m^{\phi(N)} = 1 \bmod N$$

It means that reminder of division $m^{\phi(N)}$ by N is always 1. By the equality $1^K = 1$

$$m^{K \cdot \phi(N)} = 1 \bmod N$$

If we multiply both parts by M, we get

$$m \cdot m^{K \cdot \phi(N)} = m^{K \cdot \phi(N)+1} = m \bmod N$$

It follows that Alice is able to define the secret d as follows

$$e \cdot d = K \cdot \phi(N) + 1$$
$$d = \frac{K \cdot \phi(N) + 1}{e}$$

The private exponent d is computed as the modular multiplicative inverse of $e \mod \phi(N)$.

$$e \cdot d \equiv 1 \pmod{\phi(N)}$$

So that

$$e \cdot d = K \cdot \phi(N) + 1$$

where K is any integer that satisfies the equation.

The following image demonstrates the concept of RSA approach

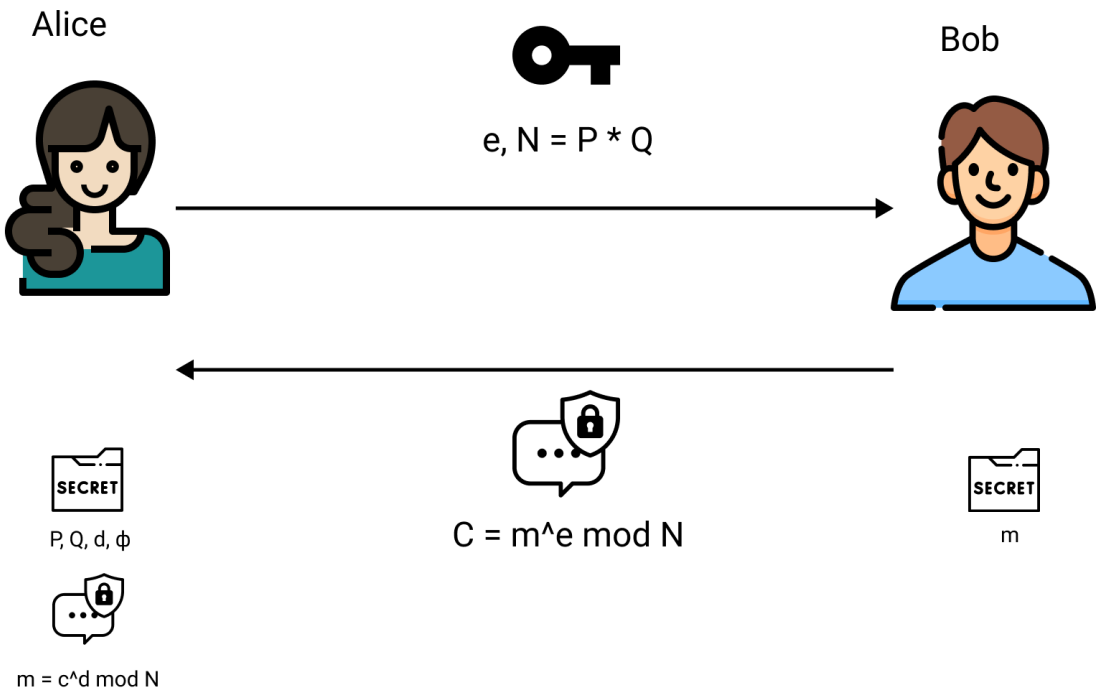

Figure 4. RSA algorithm concept diagram.

To summarize, the process by the steps is as follows

- Alice defines two secret prime numbers $P,\ Q$.
- Alice computes a part of public key $N = P \cdot Q$ and $\phi = (P-1)(Q-1)$
- Alice chooses a part of public key e, $1 < e < \phi$ such that $\gcd(e, \phi) = 1$.
- Alice computes secret exponent d, $1 < d < \phi$ such that $ed \equiv 1 \bmod \phi$.
- Alice shares public key (e, N) with Bob and keeps private key (d, p, q) in secret.
- Bob defines the message m, encrypts it as $C = m^e \bmod N$.
- Bob sends C to Alice.
- Alice decrypts C using her secret d, so she gets m

$$m = C^d \bmod N$$

Security of the RSA approach is based on the complexity of fundamental problem of prime factorization, which takes decades to solve having enough large number.

References

[1] Rivest, Ronald L. and Shamir, Adi and Adleman, Leonard. A method for obtaining digital signatures and public-key cryptosystems. *Communications of the ACM*, 21(2):120–126, 1978. https://web.williams.edu/Mathematics/lg5/302/RSA.pdf.

[2] Clifford C. Cocks. A note on non-secret encryption. *CESG Memo*, 1973.

Version: `Local-0.1.0`