RSA

Paolo Bettelini

Abstract

This document contains the main concepts about RSA cryptography.

Contents

1 Definition and usage **3**

2 Secure data transmission **3**

2.1 Hash integrity 3

3 Algorithm **4**

3.1 Finding the keys 4

3.2 Encryption and decryption 4

1 Definition and usage

RSA (Rivest-Shamir-Adleman, 1977) is a public/private key cryptosystem used for secure data transmission.

There are a few key ideas about asymmetric cryptography algorithms.
Consider a scenario where a *sender* wants to send a message to a *receiver*.

- The public key is publicly distributed by the receiver
- The private key is kept secret by the receiver
- The sender encrypts the message with the public key
- The encrypted message can only be decrypted using the private key

The private key is hard to obtain by an attacker because it means factoring a large numbers, which is a notoriously hard problem.

2 Secure data transmission

This algorithm is used in the first steps of a HTTPS connection.

Scenario: a *client* establishes a connection with a *server*.

1. The server sends its SSL certificate to the client, which contains the public key
2. The client generates a one-use session key
3. The client encrypts the session key with the public key and sends it to the server
4. The server decrypts the session key with its private key
5. The session key will be used to encrypt and decrypt each following packet

Since the RSA algorithm is a rather slow process, it is used only in the first few steps of the connection to safely communicate a random session key. This key is used to encrypt and decrypt each following transmission using a symmetric algorithm, such as AES.

2.1 Hash integrity

When the server sends a packet to the client, the packet could be modified by an attacker.

To avoid this risk, the packet comes with a hash of its data.
This hash is encrypted with the private key, and can only be decrypt with the public key.
By doing so, the server is the only one able to create the correct encrypted hash for the packet, meaning that if an attacker was to change the transmitted data, he wouldn't be able to encrypt the hash of the data with the private key.

3 Algorithm

3.1 Finding the keys

To receive secure data the receiver must generate a public key and a private key.

The algorithm uses 3 numbers (n, e, d), which compose the public and private key.

$$k_{public} \equiv (n, e)$$
$$k_{private} \equiv (n, d)$$

Pick two prime numbers p and q such that their product is approximately N bits long.
The length of the product is proportional to the security of the key.
Common values are $\{512; 1024; 2048; 4096; \cdots\}$ bits.

The number n is given by

$$n = p \cdot q$$

Calculate $\varphi(n)$

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

Where $\varphi(x)$ is the Euler's totient function.

e is the first result of the equation

$$\gcd(e, \varphi(n)) = 1, \quad 1 < e < \varphi(n)$$

d is the first result of the equation

$$de \equiv 1 \quad (mod\, \varphi(n)), \quad 1 < d < \varphi(n)$$

3.2 Encryption and decryption

The encryption is the process of transforming some *data* into a *ciphertext* and viceversa.

$$\text{data} \rightarrow \text{ciphertext}$$
$$\text{ciphertext} \rightarrow \text{data}$$

Considering the *data* and the *ciphertext* as a number

$$\text{ciphertext} = \text{data}^e \, mod\, n$$
$$\text{data} = \text{ciphertext}^d \, mod\, n$$

An attacker knows n and e, which are part of the public key.
To find d, the attacker would need to know $\varphi(n)$, which can be computed using p and q.
It is possible to obtain p and q since $n = p \cdot q$, however it is an incredibly hard task, since it is almost impossible to factor an incredibly large number such as n.