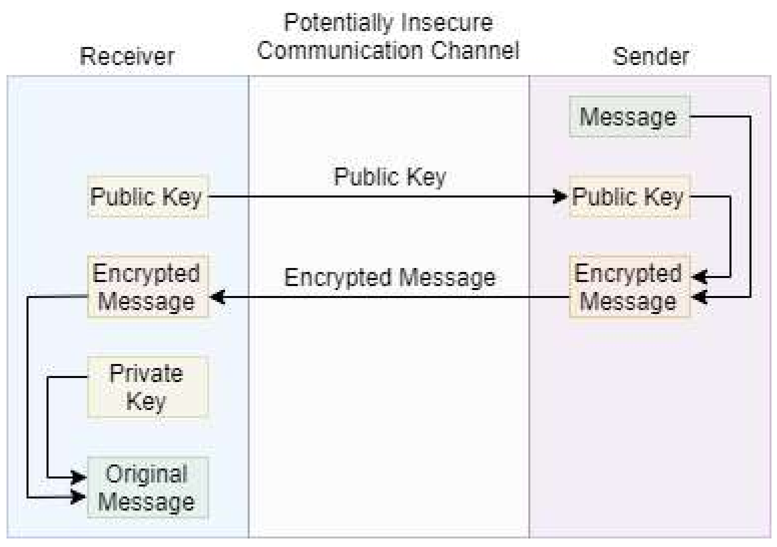
Algorithms Explained: RSA Encryption

Jin Kyu Lim Follow
Feb 20, 2019 · 4 min read

RSA (Rivest-Shamir-Adleman) algorithm is one of the many algorithms that allow you to transmit data securely over a potentially insecure medium. The basic working principle of the algorithm is illustrated below.



How the RSA algorithm works

First, the receiver generates a *public key* and a *private key*, and sends the *public key* to the sender. Then, the sender encrypts the message with the received public key, and sends it back to the receiver. The receiver then decrypts the message using the private key, and the original message is retrieved.

As we can see from the diagram above, the raw message never gets transmitted over the public domain, which is where attackers will be eavesdropping. The private key must be kept securely at all times, so that the message cannot be decrpyted by a third party.

The idea behind this algorithm is that the message is computationally expensive to decode without the private key, and even if the public key is compromised, it is also computationally expensive to derive the private key from the public key. At this point, you might be wondering what the relationship between the public and private key is, and how the private key is able to decrypt messages encrypted with the public key. The bottom line idea is that such operation is possible as both keys are derived from a common set of numbers, but let's dig into detail what this actually means.

The math behind the RSA algorithm:

We first select two prime numbers, p and q. These two numbers will be the *common building blocks* for both the public and private keys. Using these two numbers, we derive four variables. Firstly, we have **n**, the product of the two numbers.

$$n = p \times q$$

We then calculate the totient of n, **φ(n)**. In our case, this can be calculated from a simple function:

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

We then select a new variable, **e**, which fulfills the following conditions:

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

$$1 < e < \varphi(n)$$

Finally, we select a value for a new variable **d**, which has the following property:

$$d \times e \equiv 1(\mathrm{mod}\varphi(n))$$

Now, our public key will consist of (n,e) and our private key (d,e). Encrypting the message with our key is fairly simple. Let's say our message is m and the encrypted message is c. Then, m and c have the following relationship:

Encryption:

$$c \equiv m^e(\mathrm{mod}(n))$$

Decryption:

$$m \equiv c^d(\mathrm{mod}(n))$$

Example:

Let's say we choose p=3, q=5. Then, we would have our **n** variable to be:

$$n = p \times q = 7 \times 11 = 77$$

Then, we calculate the totient, **φ(n)**:

$$\varphi(N) = (p-1)(q-1) = (7-1)(11-1) = 60$$

We can then pick a value for **e** that meets the criteria mentioned above. Let's pick 17, as it meets the criteria:

$$1 < 17 < \varphi(N) = 60$$

$$gcd(17, 60) = 1$$

Then we can pick a value for **d**, that satisfies the above criterion. we can pick 53, as:

$$53 \times 17 \equiv 1(\mathrm{mod}60)$$

Now we have the public key consisted of (n, e) to be (77,17), and the public key consisted of (n,d) to be (77,53). Let's say we want to send a payload message of the number 6.

We then encrypt the message:

$$c = 6^{17}(\mathrm{mod}(77)) = 41$$

And decrypting gives us:

$$m = 41^{53}(\mathrm{mod}(77)) = 6$$

Security Algorithms Encryption Rsa Https

Discover Medium

Welcome to a place where words matter. On Medium, smart voices and original ideas take center stage - with no ads in sight. Watch

Make Medium yours

Follow all the topics you care about, and we'll deliver the best stories for you to your homepage and inbox. Explore

Become a member

Get unlimited access to the best stories on Medium — and support writers while you're at it. Just $5/month. Upgrade

Medium

About Help Legal