# RSA ALGORITHM

## Key generation

1. Generate two large prime numbers, *p* and *q*
2. Let $n = pq$
3. Let $\varphi(n) = (p-1)(q-1)$
4. Choose a small number *e*, coprime to $\varphi(n)$
5. Find *d*, such that $d*e \bmod \varphi(n) = 1$
6. Publish *e* and *n* as the public key.
7. Keep *d, p, q* secret. *d* and *n* constitute the private key.

## Encryption

$c = m^e \, modn$

## Decryption

$m = c^d \, modn$

**Notes**: Its security comes from the computational difficulty of factoring large numbers. To be secure, very large numbers must be used for *p* and *q*: 100 decimal digits at the very least.

$c = m^e \, modn$

## Why does it work?

$$m = c^d \bmod n = \left(m^e\right)^d \bmod n = \left(m^{ed}\right) \bmod n = \left(m^{ed \bmod \varphi(n)}\right) \bmod n =$$

$$\left(m^{1 \bmod \varphi(n)}\right) \bmod n = \left(m^1\right) \bmod n = m \,\mathrm{modn}$$

If m and n are coprime, which is always true in this case as n = p*q (p and q are large prime numbers), then the Euler formula is valid:

$$m^{\varphi(n)} = 1 \bmod n$$

$$m^{ed} m^{\varphi(n)} = m^{ed} \bmod n$$

$$m^{ed} m^{2\varphi(n)} = m^{ed} \bmod n$$

$ed = r + q \times \varphi(n)$

$m^{ed} = m^{r+q\times\varphi(n)} \bmod n = m^r m^{q\times\varphi(n)} \bmod n = m^r \bmod n = m^{ed \bmod \varphi(n)} \bmod n = m^{1 \bmod \varphi(n)} \bmod n =$
$m \bmod n$

**Example 1**:

1) Generate two large prime numbers, *p* and *q*

To make the example easy to follow I am going to use small numbers, but this is not secure. To find random primes, we start at a random number and go up ascending odd numbers until we find a prime. Let's have:

*p = 7*
*q = 19*

2) Let *n = pq*

*n = 7 * 19*
*= 133*

3) Let *Phi(n) = (p - 1)(q - 1)*

*Phi(n) = (7 - 1)(19 - 1) = 6 * 18 = 108*

4) Choose a small number, *e* coprime to *Phi(n)*

*e* coprime to $\varphi(n)$, means that the largest number that can exactly divide both *e* and $\varphi(n)$ (their greatest common divisor, or GCD) is 1. Euclid's algorithm is used to find the GCD of two numbers, but the details are omitted here.

*e = 2 => GCD(e, 108) = 2 (no)*
*e = 3 => GCD(e, 108) = 3 (no)*
*e = 4 => GCD(e, 108) = 4 (no)*
*e = 5 => GCD(e, 108) = 1 (yes!)*

5) Find *d*, such that *de mod* $\varphi(n)$ *= 1 using the F.I.S.H. (extended Euclid algorithm): d = 65*

| Public Key | Secret Key |
|---|---|
| $n = 133$<br>$e = 5$ | $n = 133$<br>$d = 65$ |

**Encryption**

The message must be a number less than the smaller of p and q. However, at this point we don't know p or q, so in practice a lower bound on p and q must be published. This can be somewhat below their true value and so isn't a major security concern. For this example, lets use the message "6".

$$\begin{aligned} c &= m^e \bmod n \\ &= 6^5 \bmod 133 \\ &= 7776 \bmod 133 \\ &= 62 \end{aligned}$$

**Decryption**

This works very much like encryption, but involves a larger exponentiation, which is broken down into several steps.

$$\begin{aligned} m &= c^d \bmod n \\ &= 62^{65} \bmod 133 = 6 \end{aligned}$$