**Key generation**

The keys for the RSA algorithm are generated in the following way:

1. Choose two large [prime numbers](https://en.wikipedia.org/wiki/Prime_number) *p* and *q*.
   * To make factoring harder, *p* and *q* should be chosen at random, be both large and have a large difference.[[1]](https://en.wikipedia.org/wiki/RSA_(cryptosystem)#cite_note-rsa-1) For choosing them the standard method is to choose random integers and use a [primality test](https://en.wikipedia.org/wiki/Primality_test) until two primes are found.
   * *p* and *q* should be kept secret.
2. Compute *n* = *pq*.
   * *n* is used as the [modulus](https://en.wikipedia.org/wiki/Modular_arithmetic) for both the public and private keys. Its length, usually expressed in bits, is the [key length](https://en.wikipedia.org/wiki/Key_length).
   * *n* is released as part of the public key.
3. Compute *λ*(*n*), where *λ* is [Carmichael's totient function](https://en.wikipedia.org/wiki/Carmichael%27s_totient_function). Since *n* = *pq*, *λ*(*n*) = [lcm](https://en.wikipedia.org/wiki/Least_common_multiple)(*λ*(*p*), *λ*(*q*)), and since *p* and *q* are prime, *λ*(*p*) = [*φ*](https://en.wikipedia.org/wiki/Euler_totient_function)(*p*) = *p* − 1, and likewise *λ*(*q*) = *q* − 1. Hence *λ*(*n*) = lcm(*p* − 1, *q* − 1).
   * The lcm may be calculated through the [Euclidean algorithm](https://en.wikipedia.org/wiki/Euclidean_algorithm), since lcm(*a*, *b*) = ⁠|*ab*|/gcd(*a*, *b*)⁠.
   * *λ*(*n*) is kept secret.
4. Choose an integer *e* such that 1 < *e* < *λ*(*n*) and [gcd](https://en.wikipedia.org/wiki/Greatest_common_divisor)(*e*, *λ*(*n*)) = 1; that is, *e* and *λ*(*n*) are [coprime](https://en.wikipedia.org/wiki/Coprime).
   * *e* having a short [bit-length](https://en.wikipedia.org/wiki/Bit-length) and small [Hamming weight](https://en.wikipedia.org/wiki/Hamming_weight) results in more efficient encryption – the most commonly chosen value for *e* is 216 + 1 = 65537. The smallest (and fastest) possible value for *e* is 3, but such a small value for *e* has been shown to be less secure in some settings.[[15]](https://en.wikipedia.org/wiki/RSA_(cryptosystem)#cite_note-Boneh99-15)
   * *e* is released as part of the public key.
5. Determine *d* as *d* ≡ *e*−1 (mod *λ*(*n*)); that is, *d* is the [modular multiplicative inverse](https://en.wikipedia.org/wiki/Modular_multiplicative_inverse) of *e* modulo *λ*(*n*).
   * This means: solve for *d* the equation *de* ≡ 1 (mod *λ*(*n*)); *d* can be computed efficiently by using the [extended Euclidean algorithm](https://en.wikipedia.org/wiki/Extended_Euclidean_algorithm), since, thanks to *e* and *λ*(*n*) being coprime, said equation is a form of [Bézout's identity](https://en.wikipedia.org/wiki/B%C3%A9zout%27s_identity), where *d* is one of the coefficients.
   * *d* is kept secret as the *private key exponent*.