Rivest-Shamir-Adelman
Public Key/Private Key Encryption

NOTE: This is taken from a wikipedia article but it was verified using the sources listed at the end of this handout. Normally the use of wikipedia as a source is frowned upon. In general, RSA encryption involves three main functions: Key generation, encryption, and decryption. Also, RSA may be used to securely sign a message.

Key Generation

RSA involves a public key and a private key. The public key can be known by everyone and is used for encrypting messages. Messages encrypted with the public key can only be decrypted in a reasonable amount of time using the private key. The keys for the RSA algorithm are generated the following way:

1) Choose two distinct prime numbers p and q. For security purposes, the integers p and q should be chosen at random, and should be of similar bit-length. Prime integers can be efficiently found using a primality test. In practise, both p and q should be very large prime numbers.
2) Compute $n = p \times q$. n is the modulus for both the public and private keys. The length of n in bits is the key length.
3) Compute $\varphi(n) = (p-1)(q-1)$. This value must be kept private.
4) Choose a random integer e[1] such that $1 < e < \varphi(n)$ and $gcd(e, \varphi(n)) = 1$; i.e., e and $\varphi(n)$ are co-prime.
   - e is released as part of the public key (the exponent)
   - e having a short bit-length and small Hamming weight results in more efficient encryption − most commonly $2^{16} + 1 = 65,537$. However, much smaller values of e (such as 3) have been shown to be less secure in some settings.
5) Determine d as $d \equiv e^{-1} (\mod \varphi(n))$; i.e., d is the modular multiplicative inverse of $e (\mod \varphi(n))$.
   - This is more clearly stated as: solve for d given $d \times e \equiv 1 (\mod \varphi(n))$
   - d is kept as the private key exponent.

The public key consists of the modulus n and the public (or encryption) exponent e. The private key consists of the modulus n and the private (or decryption) exponent d, which must be kept secret. $p, q,$ and $\varphi(n)$ must also be kept secret because they can be used to calculate d.

Encryption

Alice transmits her public key (n, e) to Bob and keeps the private key d secret. Bob then wishes to send message $\mathbf{M}$ to Alice.

He first turns $\mathbf{M}$ into an integer m, such that $0 \leq m < n$ and $\gcd(m, n) = 1$ by using an agreed-upon reversible protocol known as a padding scheme. He then computes the ciphertext c corresponding to $c \equiv m^e \mod n$. This can be done efficiently, even for 500-bit numbers, using Modular exponentiation. Bob then transmits c to Alice.

Note that at least nine values of m will yield a ciphertext c equal to m.

Decryption

Alice can recover m from c by using her private key exponent d via computing $m \equiv c^d \mod n$. Given m, she can recover the original message $\mathbf{M}$ by reversing the padding scheme.

(In practice, there are more efficient methods of calculating c^d using the precomputed values below.)

[1] e does not need to be prime, but it makes it more likely that e and $\varphi(n)$ are co-prime.

Signing Messages (Authentication)

Suppose Alice uses Bob's public key to send him an encrypted message. In the message, she can claim to be Alice but Bob has no way of verifying that the message was actually from Alice since anyone can use Bob's public key to send him encrypted messages. In order to verify the origin of a message, RSA can also be used to sign a message.

Suppose Alice wishes to send a signed message to Bob. She can use her own private key to do so. She produces a hash value of the message, raises it to the power of $d(\mod n)$ (as she does when decrypting a message), and attaches it as a "signature" to the message. When Bob receives the signed message, he uses the same hash algorithm in conjunction with Alice's public key. He raises the signature to the power of $e(\mod n)$ (as he does when encrypting a message), and compares the resulting hash value with the message's actual hash value. If the two agree, he knows that the author of the message was in possession of Alice's private key, and that the message has not been tampered with since.

A Worked Example

Here is an example of RSA encryption and decryption. The parameters used here are artificially small, but one can also use OpenSSL to generate and examine a real keypair.

Choose two distinct prime numbers, such as:

$$p = 61 \text{ and } q = 53.$$

Compute $n = p \times q$ giving

$$n = 61 \times 53 = 3233.$$

Compute the totient of the product as $\varphi(n) = (p-1)(q-1)$ giving

$$\varphi(3233) = (61-1)(53-1) = 3120.$$

Choose any number $1 < e < 3120$ that is coprime to 3120. Choosing a prime number for e leaves us only to check that e is not a divisor of 3120.

Let:

$$e = 17.$$

Compute d, the modular multiplicative inverse of $e(\mod \varphi(n))$ yielding,

$$d = 2753.$$

Worked example for the modular multiplicative inverse:

$$e \times d \mod \varphi(n) = 1$$

$$17 \times 2753 \mod 3120 = 1.$$

The public key is $(n = 3233, e = 17)$. For a padded plaintext message m, the encryption function is

$$c(m) = m^{17} \mod 3233.$$

The private key is $(d = 2753)$. For an encrypted ciphertext c, the decryption function is

$$m(c) = c^{2753} \mod 3233.$$

For instance, in order to encrypt $m = 65$, we calculate

$$c = 65^{17} \mod 3233 = 2790$$

To decrypt $c = 2790$, we calculate

$$m = 2790^{2753} \mod 3233 = 65.$$

Both of these calculations can be computed efficiently using the square-and-multiply algorithm for modular exponentiation. In real-life situations the primes selected would be much larger; in our example it would be trivial to factor n, 3233 (obtained from the freely available public key) back to the primes p and q. Given e, also from the public key, we could then compute d and so acquire the private key.

Additional Sources

1) Stallings, William. Network and internetwork security: principles and practice. Vol. 1. Englewood Cliffs: Prentice Hall, 1995.
2) Trappe, Wade, and Lawrence C. Washington. Introduction to cryptography with coding theory. Pearson Education, 2006.
3) Stallings, William, and Lawrie Brown. "Computer security." Principles and Practice (2008).