Chapter 7
The RSA Cryptosystem

After Whitfield Diffie and Martin Hellman introduced public-key cryptography in their landmark 1976 paper [58], a new branch of cryptography suddenly opened up. As a consequence, cryptologists started looking for methods with which public-key encryption could be realized. In 1977, Ronald Rivest, Adi Shamir and Leonard Adleman (cf. Fig. 7.1) proposed a scheme which became the most widely used asymmetric cryptographic scheme, RSA.

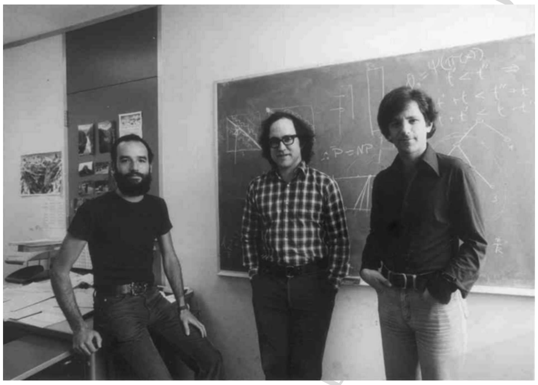

Fig. 7.1 An early picture of Adi Shamir, Ron Rivest, and Leonard Adleman (reproduced with permission from Ron Rivest)

In this chapter you will learn:

- How RSA works
- Practical aspects of RSA, such as computation of the parameters, and fast encryption and decryption
- Security estimations
- Implementational aspects

7.1 Introduction

The RSA crypto scheme, sometimes referred to as the Rivest–Shamir–Adleman algorithm, is currently the most widely used asymmetric cryptographic scheme, even though elliptic curves and discrete logarithm schemes are gaining ground. RSA was patented in the USA (but not in the rest of the world) until 2000.

There are many applications for RSA, but in practice it is most often used for:

- encryption of small pieces of data, especially for key transport
- digital signatures, which is discussed in Chap. 10, e.g., for digital certificates on the Internet

However, it should be noted that RSA encryption is not meant to replace symmetric ciphers because it is several times slower than ciphers such as AES. This is because of the many computations involved in performing RSA (or any other public-key algorithm) as we learn later in this chapter. Thus, the main use of the encryption feature is to securely exchange a key for a symmetric cipher (key transport). In practice, RSA is often used together with a symmetric cipher such as AES, where the symmetric cipher does the actual bulk data encryption.

The underlying one-way function of RSA is the integer factorization problem: Multiplying two large primes is computationally easy (in fact, you can do it with paper and pencil), but factoring the resulting product is very hard. Euler's theorem (Theorem 6.3.3) and Euler's phi function play important roles in RSA. In the following, we first describe how encryption, decryption and key generation work, then we talk about practical aspects of RSA.

7.2 Encryption and Decryption

RSA encryption and decryption is done in the integer ring $\mathbb{Z}_n$ and modular computations play a central role. Recall that rings and modular arithmetic in rings were introduced in Sect. 1.4.2. RSA encrypts plaintexts x, where we consider the bit string representing x to be an element in $\mathbb{Z}_n = \{0,1,\ldots,n-1\}$. As a consequence the binary value of the plaintext x must be less than n. The same holds for the ciphertext. Encryption with the public key and decryption with the private key are as shown below:

RSA Encryption Given the public key $(n,e) = k_{pub}$ and the plaintext x, the encryption function is:

$$y = e_{k_{pub}}(x) \equiv x^e \bmod n \tag{7.1}$$

where $x,y \in \mathbb{Z}_n$.

RSA Decryption Given the private key $d = k_{pr}$ and the ciphertext y, the decryption function is:

$$x = d_{k_{pr}}(y) \equiv y^d \bmod n \tag{7.2}$$

where $x, y \in \mathbb{Z}_n$.

In practice, x, y, n and d are very long numbers, usually 1024 bit long or more. The value e is sometimes referred to as *encryption exponent* or *public exponent*, and the private key d is sometimes called *decryption exponent* or *private exponent*. If Alice wants to send an encrypted message to Bob, Alice needs to have his public key (n, e), and Bob decrypts with his private key d. We discuss in Sect. 7.3 how these three crucial parameters d, e, and n are generated.

Even without knowing more details, we can already state a few requirements for the RSA cryptosystem:

1. Since an attacker has access to the public key, it must be computationally infeasible to determine the private-key d given the public-key values e and n.
2. Since x is only unique up to the size of the modulus n, we cannot encrypt more than l bits with one RSA encryption, where l is the bit length of n.
3. It should be relatively easy to calculate $x^e \bmod n$, i.e., to encrypt, and $y^d \bmod n$, i.e., to decrypt. This means we need a method for fast exponentiation with very long numbers.
4. For a given n, there should be many private-key/public-key pairs, otherwise an attacker might be able to perform a brute-force attack. (It turns out that this requirement is easy to satisfy.)

7.3 Key Generation and Proof of Correctness

A distinctive feature of all asymmetric schemes is that there is a set-up phase during which the public and private key are computed. Depending on the public-key scheme, key generation can be quite complex. As a remark, we note that key generation is usually not an issue for block or stream ciphers.

Here are the steps involved in computing the public and private-key for an RSA cryptosystem.