Public-Key Cryptography

Eberhard Stickel
Hochschule der Sparkassen - Finanzgruppe
University of Applied Sciences, Bonn GmbH, Germany

INTRODUCTION

Secure transmission of private information is a crucial issue in today's highly computerized world. Cryptographic algorithms are used to provide privacy of message transmission and to certify authenticity of sender and/or receiver of a message and message integrity as well as undeniability of transmitted messages (Menezes et al., 1996).

The message that is sent by the sender is called the plaintext, the secured message is called the ciphertext. To get the ciphertext, the plaintext is encoded by the sender. The receiver reconstructs the plaintext from the ciphertext by decoding. For encoding and decoding so-called keys are used (Koblitz, 1994, p. 55).

In the simplest setting, sender and receiver have agreed on a common private key, which is kept secret. This is called symmetric key cryptography. The secret private key is utilized for encoding and decoding messages sent between the two parties. For encoding the plaintext is XOR-ed with the secret key. The decoding is done in the same way using the ciphertext and the secret key on the receiver's side (Menezes et al., 1996, p. 15).

To be specific, if for example the plaintext message is given by the bit string 10011101 and the secret key is 11011100, the ciphertext is then given by 01000001. By using XOR-operations with ciphertext and key once more the plaintext is returned. Note that the XOR operation between two bit is defined as follows: $0 + 0 = 1 + 1 = 0$, $1 + 0 = 0 + 1 = 1$.

If key length and length of plaintext do not coincide, the plaintext may be blocked. This leads to the concept of block ciphers (Menezes et al., 1996, p. 223).

Symmetric key techniques generally can be implemented very efficiently. The corresponding algorithms are very fast. The problem, however, lies in the fact that two parties must have agreed on a common key before they can start to communicate and exchange messages. This would be highly impractical, for example, for transactions in electronic commerce (Menezes et al., 1996, p. 31).

BACKGROUND

A major breakthrough was the publication of the Diffie-Hellman key exchange scheme (Diffie & Hellman, 1976). The technique developed by Diffie & Hellman allows one to agree on a secret key through an insecure channel,for example, the Internet. The authors rely on a problem that is hard to solve, at least using today's knowledge and computing power: Let p be a large prime or the power of a large prime. Let g be a number with $1 < g < p$. g and p are publicly known. Given g^a mod p and g^b mod p, compute g^{ab} mod p. This is the so-called Discrete Logarithm problem. Here mod refers to division with remainder. a mod b is the remainder, if a is divided by b. Details may be found in Menezes et al. (1996, p. 515). The security of the method relies on the fact that it is impossible to solve the sketched problem in feasible time if the prime p is large enough. No efficient algorithms for solving the Discrete Logarithm problem are yet known.

Suppose that Alice and Bob want to agree on a secret key over an insecure channel. The Diffie-Hellman method runs as follows:

- Alice has published p, g and g^a mod p as her public key. a is kept secret as private key.
- Bob chooses b which is kept secret, forms g^b mod p and $(g^a)^b$ mod p and submits the latter number to Alice.
- Alice computes $(g^{ab})^{1/a} = g^b \bmod p$. This is the secret key exchanged.

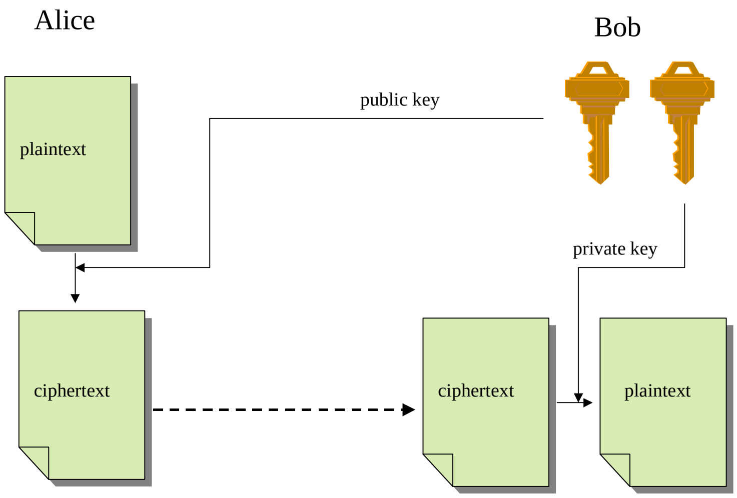
Note that various components of the algorithm are now publicly known. It is therefore called a public-key algorithm.

Various public-key algorithms are discussed in the literature and employed in practical applications. Probably the most famous algorithm is the RSA-algorithm, denoted by RSA after its founders Rivest, Shamir, & Adleman (Rivest et al., 1978). *Figure 1* shows the details.

RSA relies on the problem of factoring large numbers, for example, the product of two 1024-bit prime numbers. No algorithm exists so far that allows one to solve such a problem in feasible execution time. The basic idea of RSA is the following:

- Suppose Alice wants to transmit a secret message to Bob. Alice gets Bob's public key,for example, by accessing his Internet site.
- Alice encrypts the plaintext using the public key of Bob. Then she transmits the ciphertext to Bob.

Figure 1. Public-key cryptography



- Bob uses his secret private key to decrypt the ciphertext and recover the plaintext.

Generally public-key algorithms require a pair of keys. The public key is published, while the private key is kept secret. The public key is used for enciphering a message, while the private key is used to decipher the ciphertext.

Applying public-key methods for transmitting secret data is not efficient. The underlying algorithms are much slower and more complex than symmetric key methods (Menezes et al., 1996, p. 31). Typically a public-key algorithm is used to agree on a symmetric session key. During the session, data are protected by encrypting plaintext with the symmetric key obtained. This approach combines the advantages of symmetric and public-key technology.

Note, that there is still a problem that needs to be solved. It is necessary to assure the authenticity of the public key of a person, for example, of Bob's in the explanation of the RSA-algorithm. Otherwise, it is possible that a potential third party adversary pretends to be Bob (bogus identity). He would provide his public key to Alice, Alice would encrypt the plaintext using this key and the adversary would be able to read the secret message that was addressed to Bob. To solve this problem, trust-centers that certify the authenticity of public keys have been implemented (Menezes et al., 1996, pp. 491, 547, 581).

DIGITAL SIGNATURES

Public-key algorithms as discussed in the last section allow one to encipher secret messages. Thus, they provide privacy of information transmission. Especially in electronic commerce it is necessary to assure authenticity of messages as well as of communication partners. This is done by means of digital signatures (*Figure 2*).

Fortunately, most of the well-known public-key techniques need only slight modifications in order to be used as digital signatures. The basic idea is as follows (Zhang & Wang, 2003, p. 24):

- Suppose Alice wants to sign a message *m* and forward it to Bob. First, Alice computes a so-called digital fingerprint *d* of the message *m*. This fingerprint usually consists of a bit string of a certain length that corresponds to the message *m*. Alice signs *d* with her private key. This means that Alice encrypts *d* using her secret private key.
- Bob fetches the public key of Alice and decrypts the signature. He thus holds *d* as well as the message *m*. A digital fingerprint *d'* is computed from the transmitted message. If *d* = *d'*, he accepts message and signature, otherwise the transaction is rejected.

Digital fingerprints are computed using so-called hash functions. A hash function compresses a message to a bit string of specified length. Hash functions should be collision resistant and not invertible. Collision resistance implies that two different (meaningful) messages should have different fingerprints (hash values). Lack of invertibility assures that it is infeasible to construct a meaningful message given a specified digital fingerprint. For details refer to Goldreich (2001, p.136).

Note, that in the process of signing the roles of private and public key are simply interchanged.