Chapter 13
Key Establishment

With the cryptographic mechanisms that we have learned so far, in particular symmetric and asymmetric encryption, digital signatures and message authentication codes (MACs), one can relatively easily achieve the basic security services (cf. Sect. 10.1.3):

- Confidentiality (with encryption algorithms)
- Integrity (with MACs or digital signatures)
- Message authentication (with MACs or digital signatures)
- Non-repudiation (with digital signatures)

Similarly, identification can be accomplished through protocols which make use of standard cryptographic primitives.

However, all cryptographic mechanisms that we have introduced so far assume that keys are properly distributed between the parties involved, e.g., between Alice and Bob. The task of key establishment is in practice one of the most important and often also most difficult parts of a security system. We already learned some ways of distributing keys, in particular Diffie–Hellman key exchange. In this chapter we will learn many more methods for establishing keys between remote parties. You will learn about the following important issues:

- How keys can be established using symmetric cryptosystems
- How keys can be established using public-key cryptosystems
- Why public-key techniques still have shortcomings for key distribution
- What certificates are and how they are used
- The role that public-key infrastructures play

13.1 Introduction

In this section we introduce some terminology, some thoughts on key freshness and a very basic key distribution scheme. The latter is helpful for motivating the more advanced methods which will follow in this chapter.

13.1.1 Some Terminology

Roughly speaking, key establishment deals with establishing a shared secret between two or more parties. Methods for this can be classified into *key transport* and *key agreement* methods, as shown in Fig. 13.1. A key transport protocol is a technique where one party securely transfers a secret value to others. In a key agreement protocol two (or more) parties derive the shared secret where all parties contribute to the secret. Ideally, none of the parties can control what the final joint value will be.

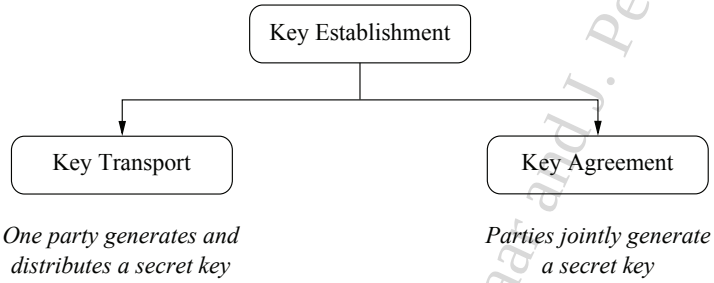

Fig. 13.1 Classification of key establishment schemes

Key establishment itself is strongly related to identification. For instance, you may think of attacks by unauthorized users who join the key establishment protocol with the aim of masquerading as either Alice or Bob with the goal of establishing a secret key with the other party. To prevent such attacks, each party must be assured of the identity of the other entity. All of these issues are addressed in this chapter.

13.1.2 Key Freshness and Key Derivation

In many (but not all) security systems it is desirable to use cryptographic keys which are only valid for a limited time, e.g., for one Internet connection. Such keys are called *session keys* or *ephemeral keys*. Limiting the period in which a cryptographic key is used has several advantages. A major one is that there is less damage if the

key is exposed. Also, an attacker has less ciphertext available that was generated under one key, which can make cryptographic attacks much more difficult. Moreover, an attacker is forced to recover several keys if he is interested in decrypting larger parts of plaintext. Real-world examples where session keys are frequently generated include voice encryption in GSM cell phones and video encryption in pay-TV satellite systems; in both cases new keys are generated within a matter of minutes or sometimes even seconds.

The security advantages of *key freshness* are fairly obvious. However, the question now is, how can key updates be realized? The first approach is to simply execute the key establishment protocols shown in this chapter over and over again. However, as we see later, there are always certain costs associated with key establishment, typically with respect to additional communication connections and computations. The latter holds especially in the case of public-key algorithms which are very computationally intensive.

The second approach to key update uses an already established joint secret key to *derive* fresh session keys. The principal idea is to use a key derivation function (KDF) as shown in Fig. 13.2. Typically, a non-secret parameter r is processed together with the joint secret k_{AB} between the users Alice and Bob.

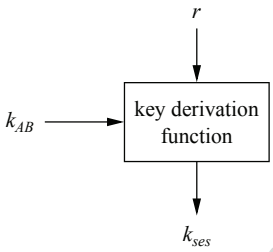

Fig. 13.2 Principle of key derivation

An important characteristic of the key derivation function is that it should be a one-way function. The one-way property prevents an attacker from deducing k_{AB} should any of the session keys become compromised, which in turn would allow the attacker to compute all other session keys.

One possible way of realizing the key derivation function is that one party sends a nonce, i.e., a numerical value that is used only once, to the other party. Both users encrypt the nonce using the shared secret key k_{AB} by means of a symmetric cipher such as AES. The corresponding protocol is shown below.