Chapter 15

Key Management

Chapter 15

Objectives

- To explain the need for a key-distribution center
- To show how a KDC can create a session key
- To show how two parties can use a symmetric-key agreement protocol to create a session key
- To describe Kerberos as a KDC and an authentication protocol
- To explain the need for certification authorities for public keys
- To introduce the idea of a Public-Key Infrastructure (PKI) and explain some of its duties

15-1 SYMMETRIC-KEY DISTRIBUTION

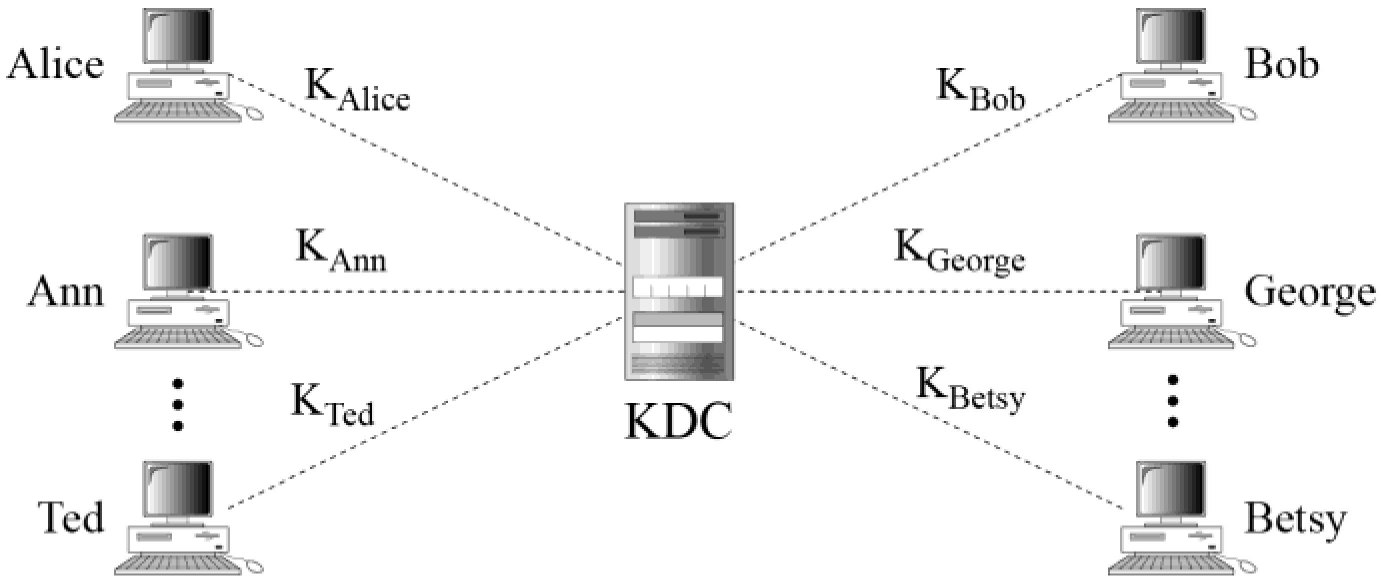
Symmetric-key cryptography is more efficient than asymmetric-key cryptography for enciphering large messages. Symmetric-key cryptography, however, needs a shared secret key between two parties. The distribution of keys is another problem.

Topics discussed in this section:

15.1.1 Key-Distribution Center: KDC
15.1.2 Session Keys

15.1.1 Key-Distribution Center: KDC

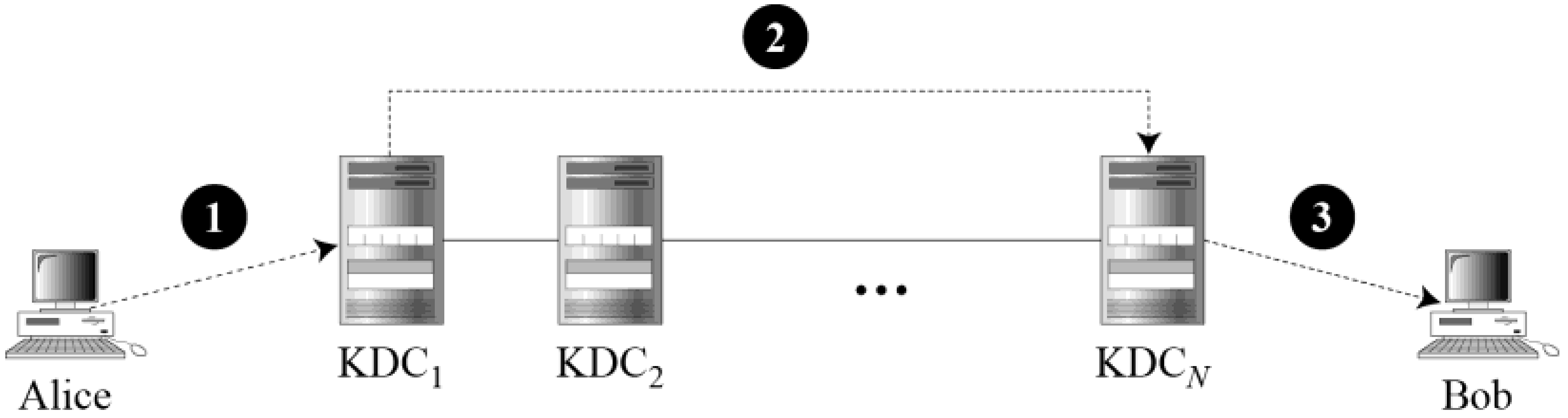
Figure 15.1 ***Key-distribution center (KDC)***



15.1.1 Continued

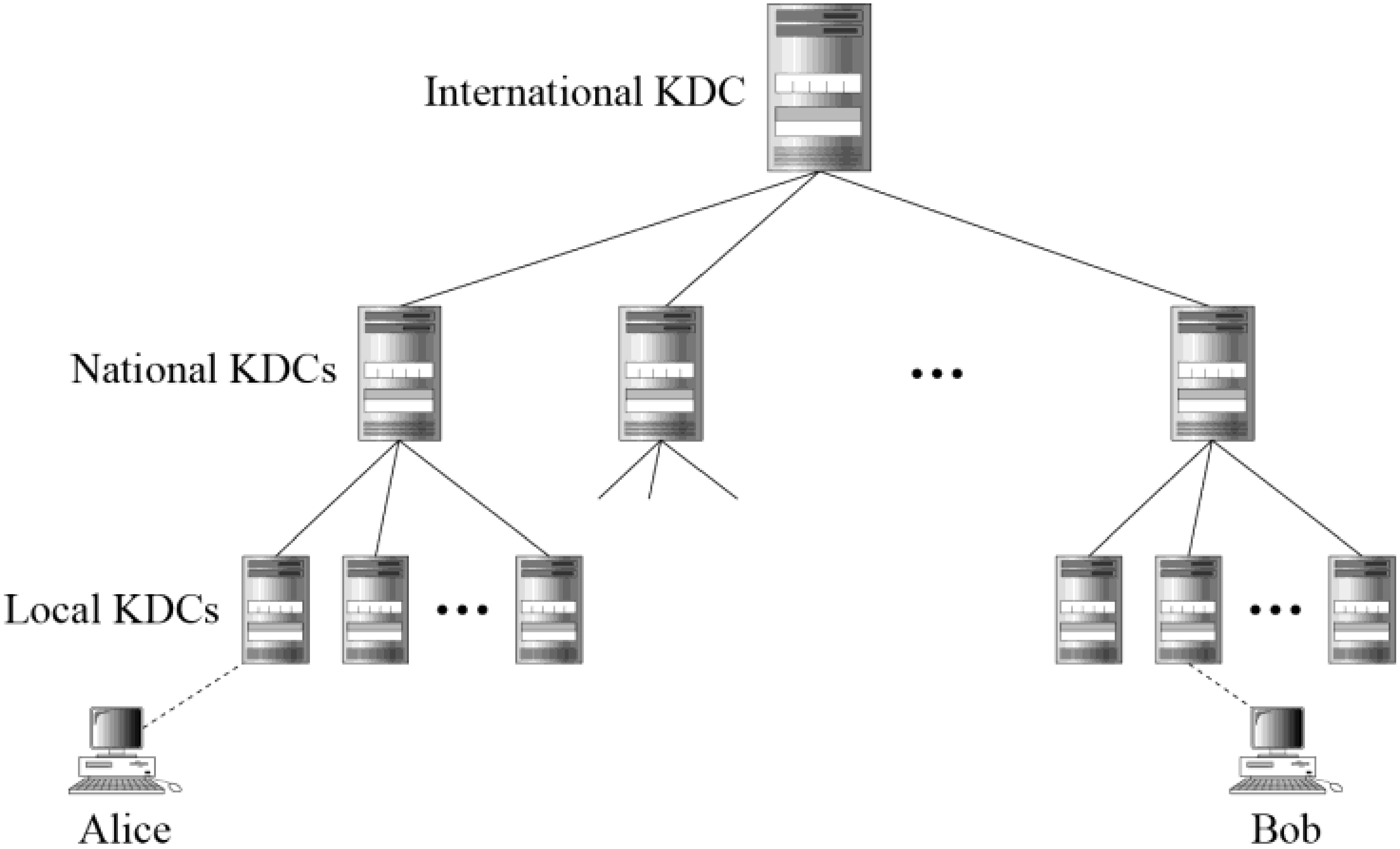
***Flat Multiple KDCs*.**

Figure 15.2 ***Flat multiple KDCs***



Hierarchical Multiple KDCs

Figure 15.3 ***Hierarchical multiple KDCs***



15.1.2 Session Keys

A KDC creates a secret key for each member. This secret key can be used only between the member and the KDC, not between two members.

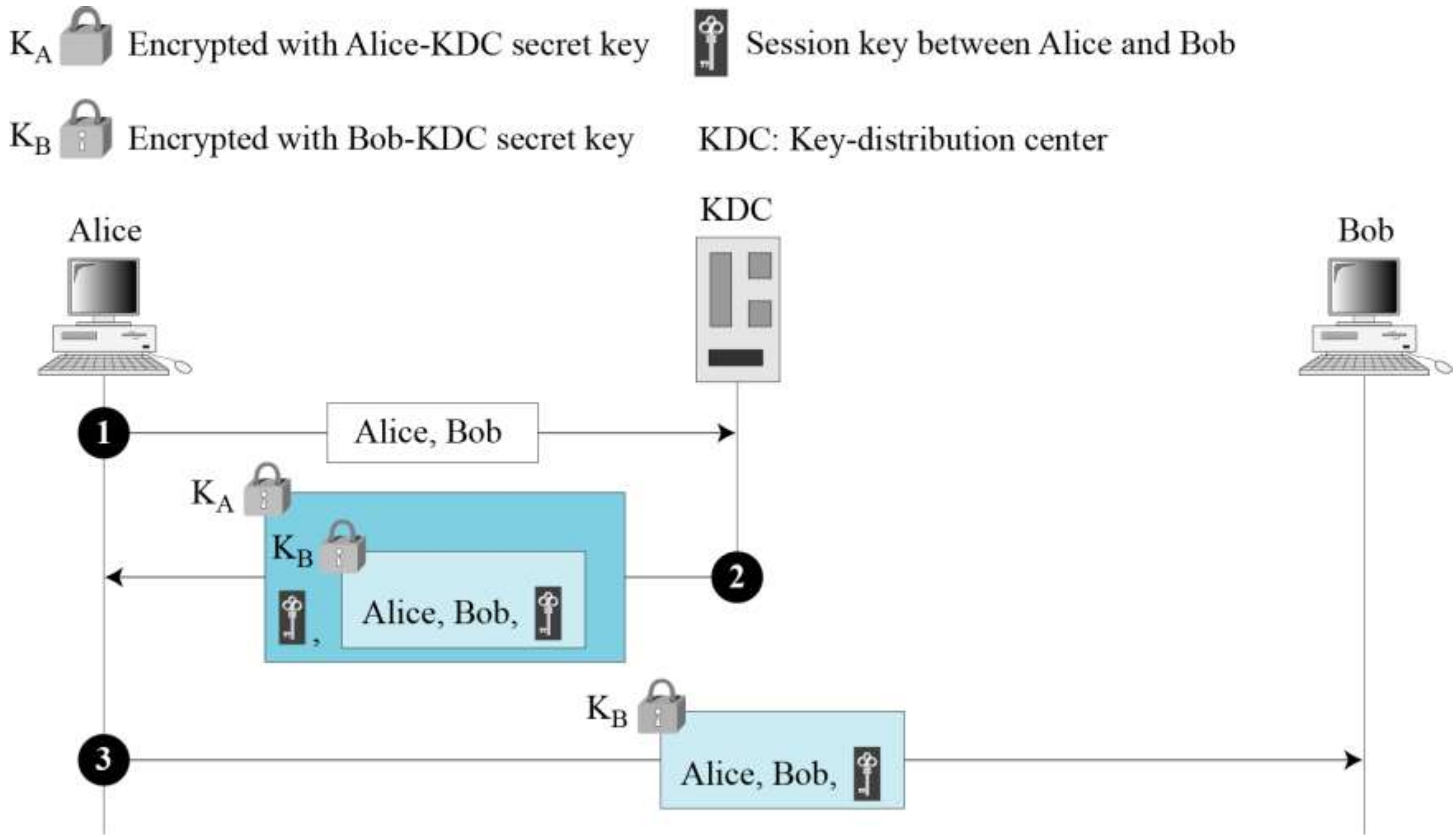
Note

A session symmetric key between two parties is used only once.

15.1.2 Continued

A Simple Protocol Using a KDC

Figure 15.4 *First approach using KDC*



15.1.2 Continued

Needham-Schroeder Protocol

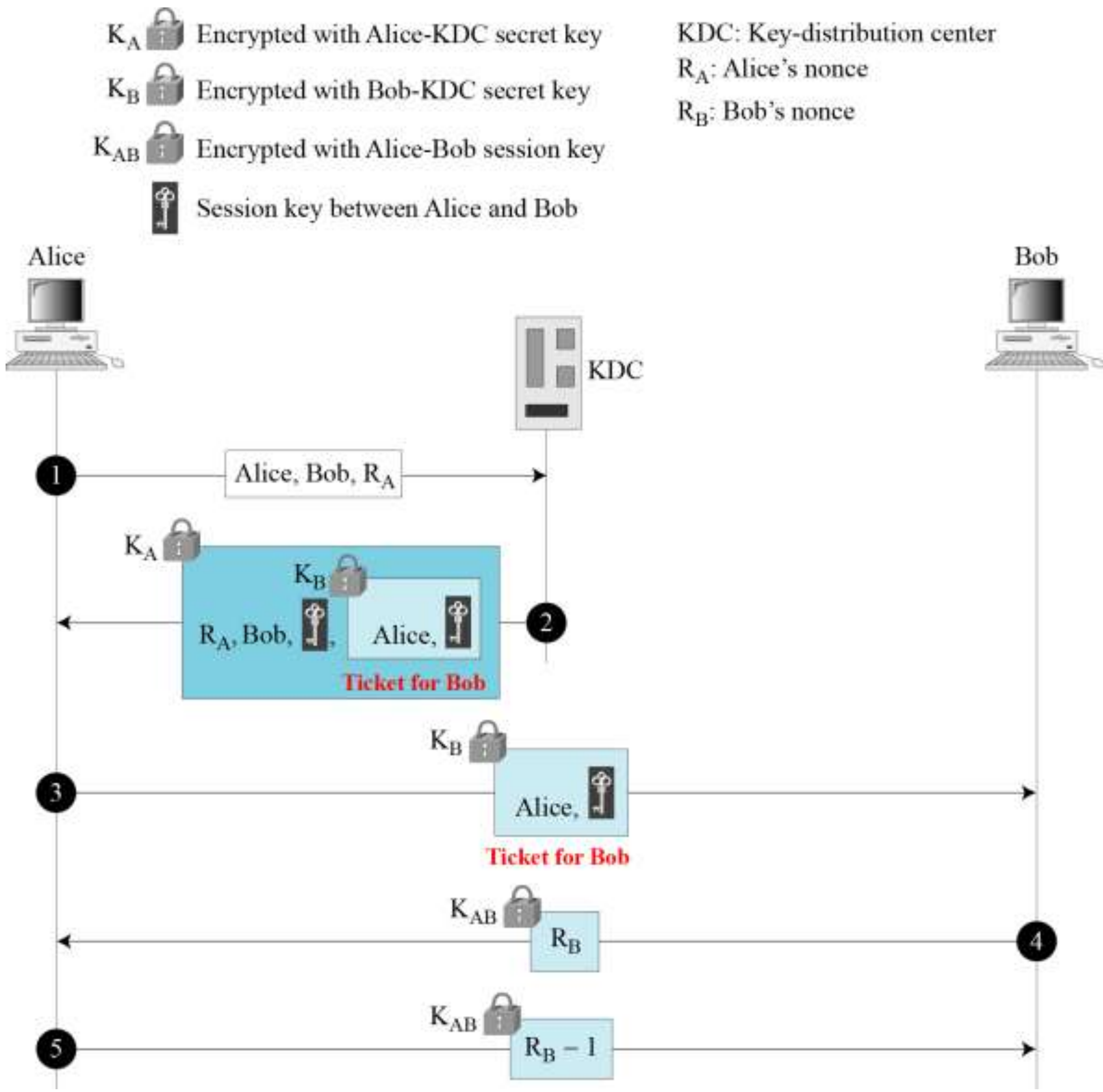


Figure 15.5

Needham-Schroeder protocol

15.1.2 Continued

Otway-Rees Protocol

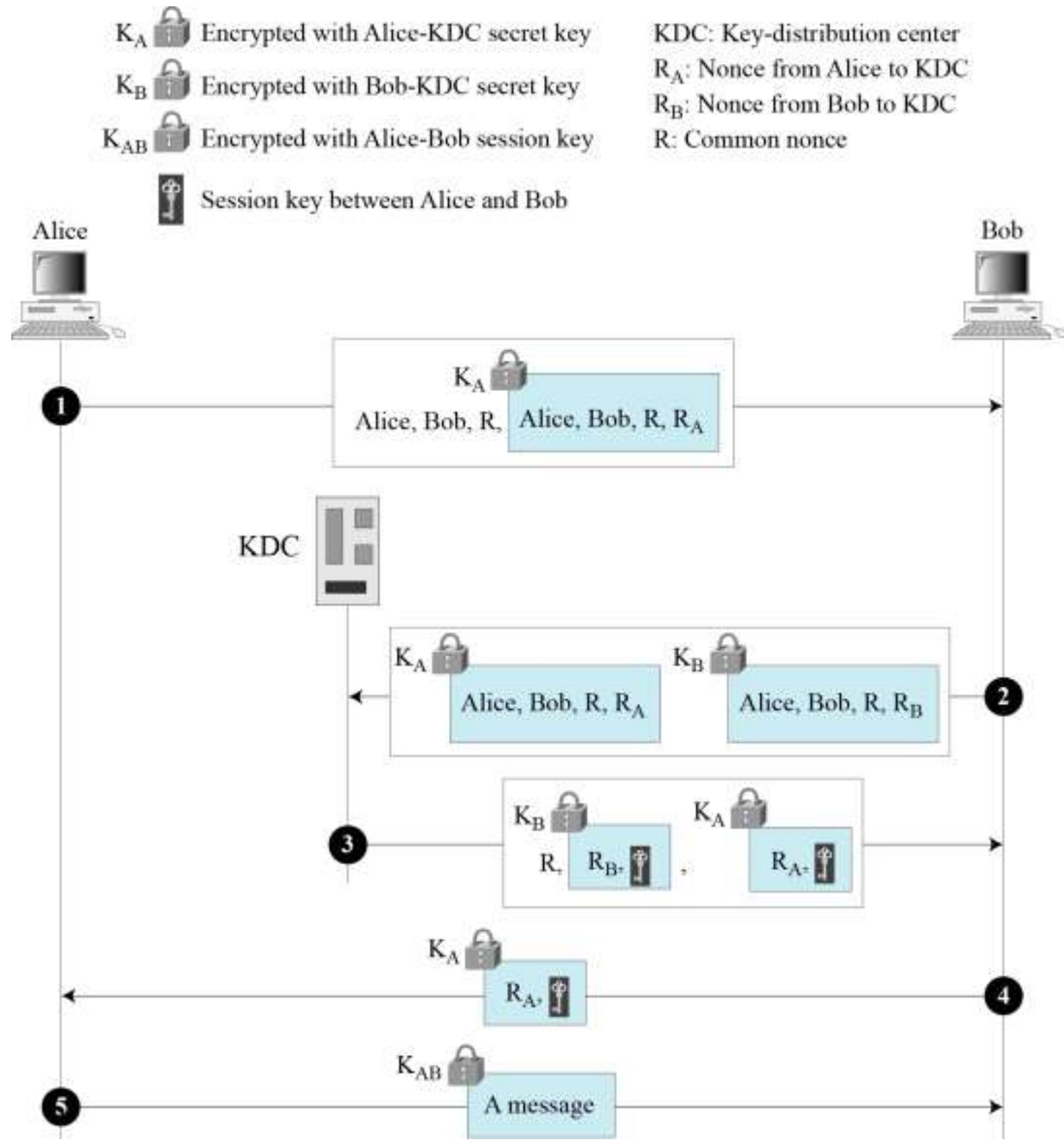


Figure 15.6

Otway-Rees protocol

15-2 KERBEROS

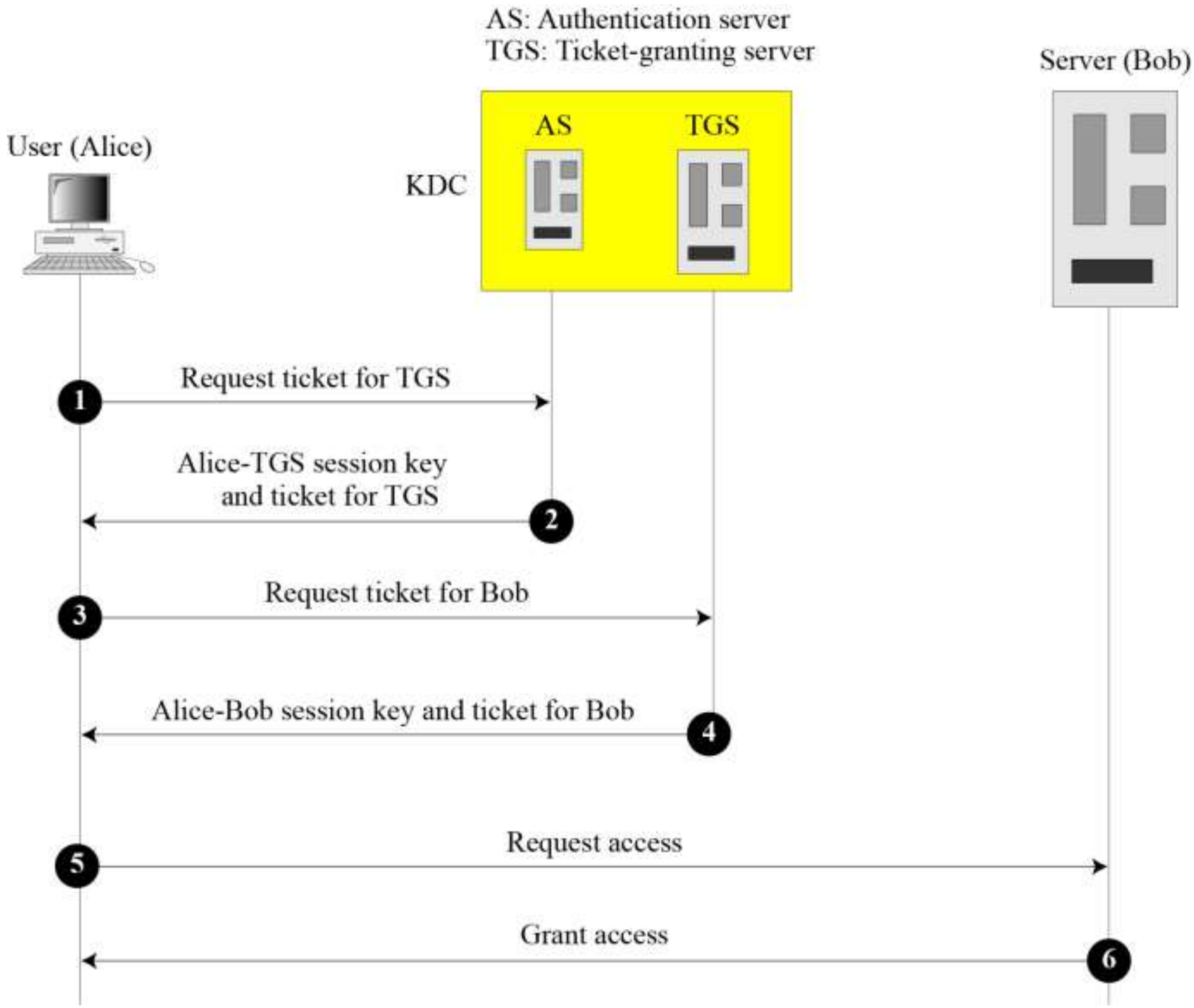
Kerberos is an authentication protocol, and at the same time a KDC, that has become very popular. Several systems, including Windows 2000, use Kerberos. Originally designed at MIT, it has gone through several versions.

Topics discussed in this section:

15.2.1 Servers
15.2.2 Operation
15.2.3 Using Different Servers
15.2.4 Kerberos Version 5
14.2.5 Realms

15.2.1 Servers

Figure 15.7 *Kerberos servers*



15.2.1 Continued

Authentication Server (AS)

The authentication server (AS) is the KDC in the Kerberos protocol.

Ticket-Granting Server (TGS)

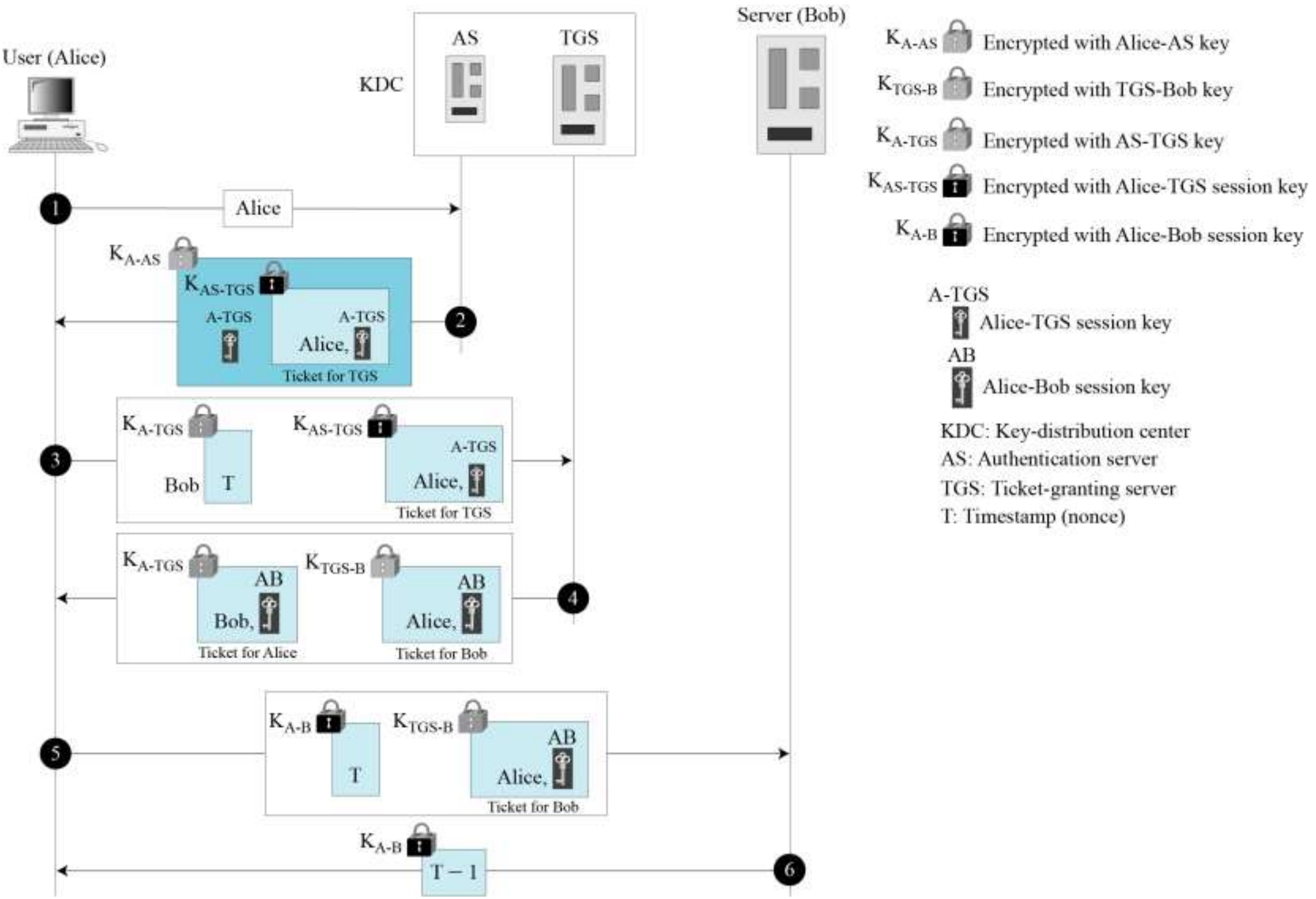
The ticket-granting server (TGS) issues a ticket for the real server (Bob).

Real Server

The real server (Bob) provides services for the user (Alice).

15.2.2 Operation

Figure 15.8 *Kerberos example*



15.2.3 Using Different Servers

Note that if Alice needs to receive services from different servers, she need repeat only the last four steps.

15.2.4 Kerberos Version 5

The minor differences between version 4 and version 5 are briefly listed below:

1) ***Version 5 has a longer ticket lifetime.***
2) ***Version 5 allows tickets to be renewed.***
3) ***Version 5 can accept any symmetric-key algorithm.***
4) ***Version 5 uses a different protocol for describing data types.***
5) ***Version 5 has more overhead than version 4.***

15.2.5 Realms

Kerberos allows the global distribution of ASs and TGSs, with each system called a realm. A user may get a ticket for a local server or a remote server.

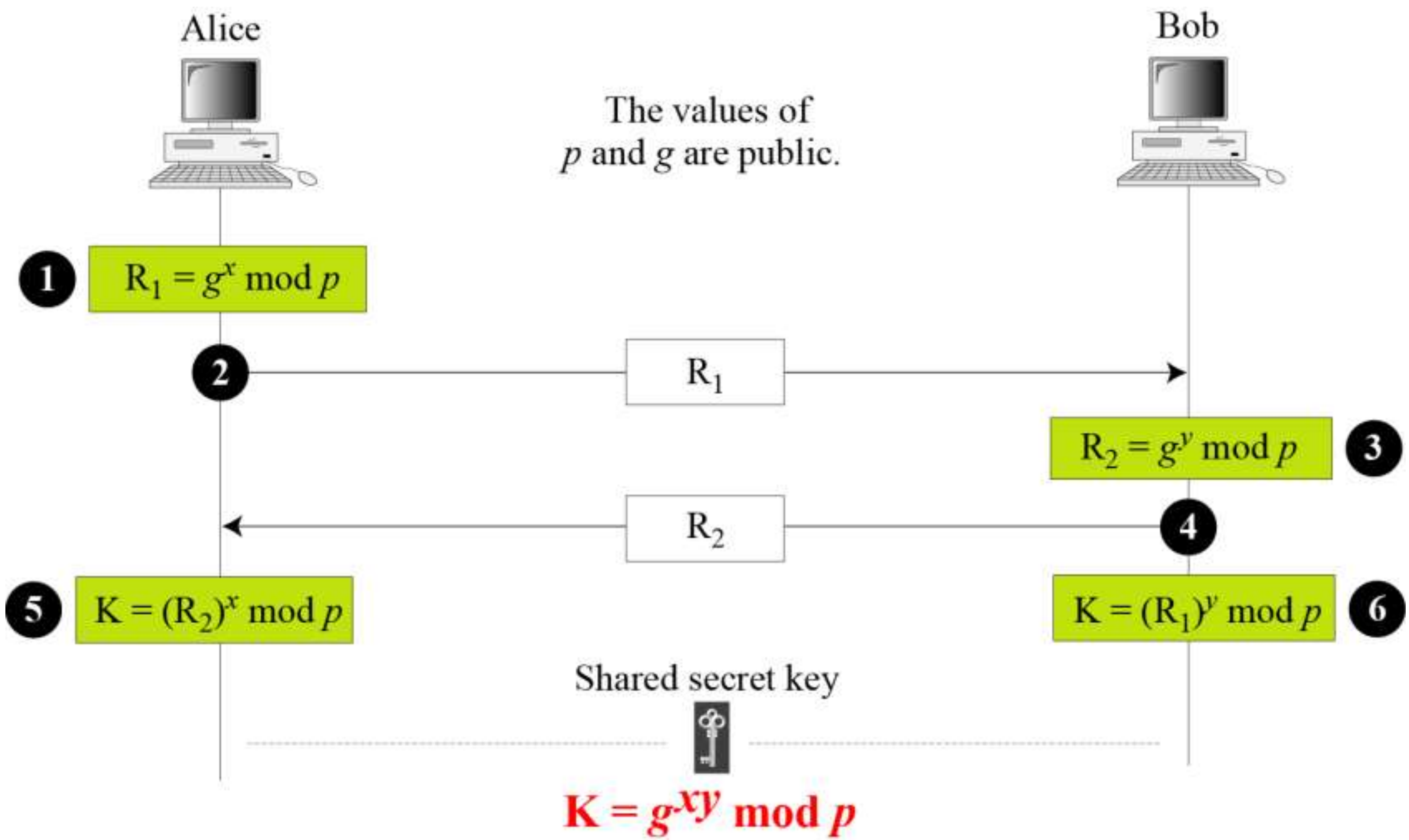
15-3 SYMMETRIC-KEY AGREEMENT

Alice and Bob can create a session key between themselves without using a KDC. This method of session-key creation is referred to as the symmetric-key agreement.

<u>Topics discussed in this section:</u>

15.3.1 Diffie-Hellman Key Agreement

Figure 15.9 ***Diffie-Hellman method***



15.3.1 Continued

Note

The symmetric (shared) key in the Diffie-Hellman method is K = g^{xy} mod p.

15.3.1 *Continued*

Example 15.1

Let us give a trivial example to make the procedure clear. Our example uses small numbers, but note that in a real situation, the numbers are very large. Assume that g = 7 and p = 23. The steps are as follows:

1. **Alice chooses x = 3 and calculates R_1 = 73 mod 23 = 21.**
2. **Bob chooses y = 6 and calculates R_2 = 76 mod 23 = 4.**
3. **Alice sends the number 21 to Bob.**
4. **Bob sends the number 4 to Alice.**
5. **Alice calculates the symmetric key K = 43 mod 23 = 18.**
6. **Bob calculates the symmetric key K = 216 mod 23 = 18.**
7. **The value of K is the same for both Alice and Bob; g^{xy} mod p = 718 mod 35 = 18.**

15.3.1 *Continued*

Example 15.2

Let us give a more realistic example. We used a program to create a random integer of 512 bits (the ideal is 1024 bits). The integer *p* is a 159-digit number. We also choose *g*, *x*, and *y* as shown below:

p	7646242985634935721824937659550305074763380967269497489235737728609252356666607554236374233096611800333381061947301309504147387009991780436548785807987581
g	2
x	557
y	273

15.3.1 Continued

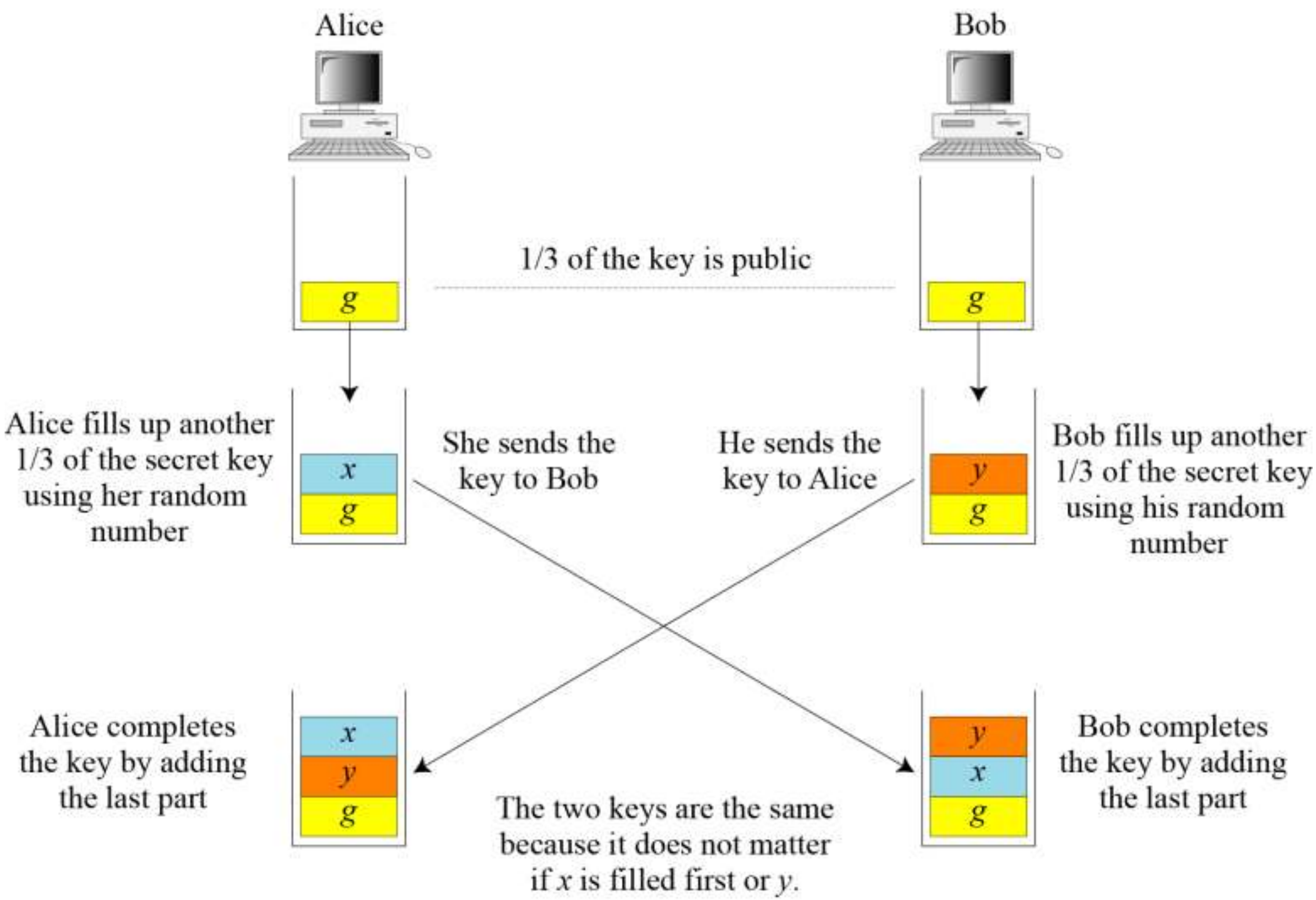
Example 15.2 *Continued*

The following shows the values of R_1, R_2, and K.

R_1	844920284205665505216172947491035094143433698520012660862863631067673619959280828586700802131859290945140217500319973312945836083821943065966020157955354
R_2	4352628387092003794707471148955816276363891162621155579751233792185663100114357182083900401818764868417538311653426916302634211067215085896255201288594143
K	1556380006645222905962258275232707652732180469444236785203204001464065008879366512042574267766083279110171530386745612522131516109765842001204086433617740

15.3.1 Continued

Figure 15.10 ***Diffie-Hellman idea***



15.3.1 Continued

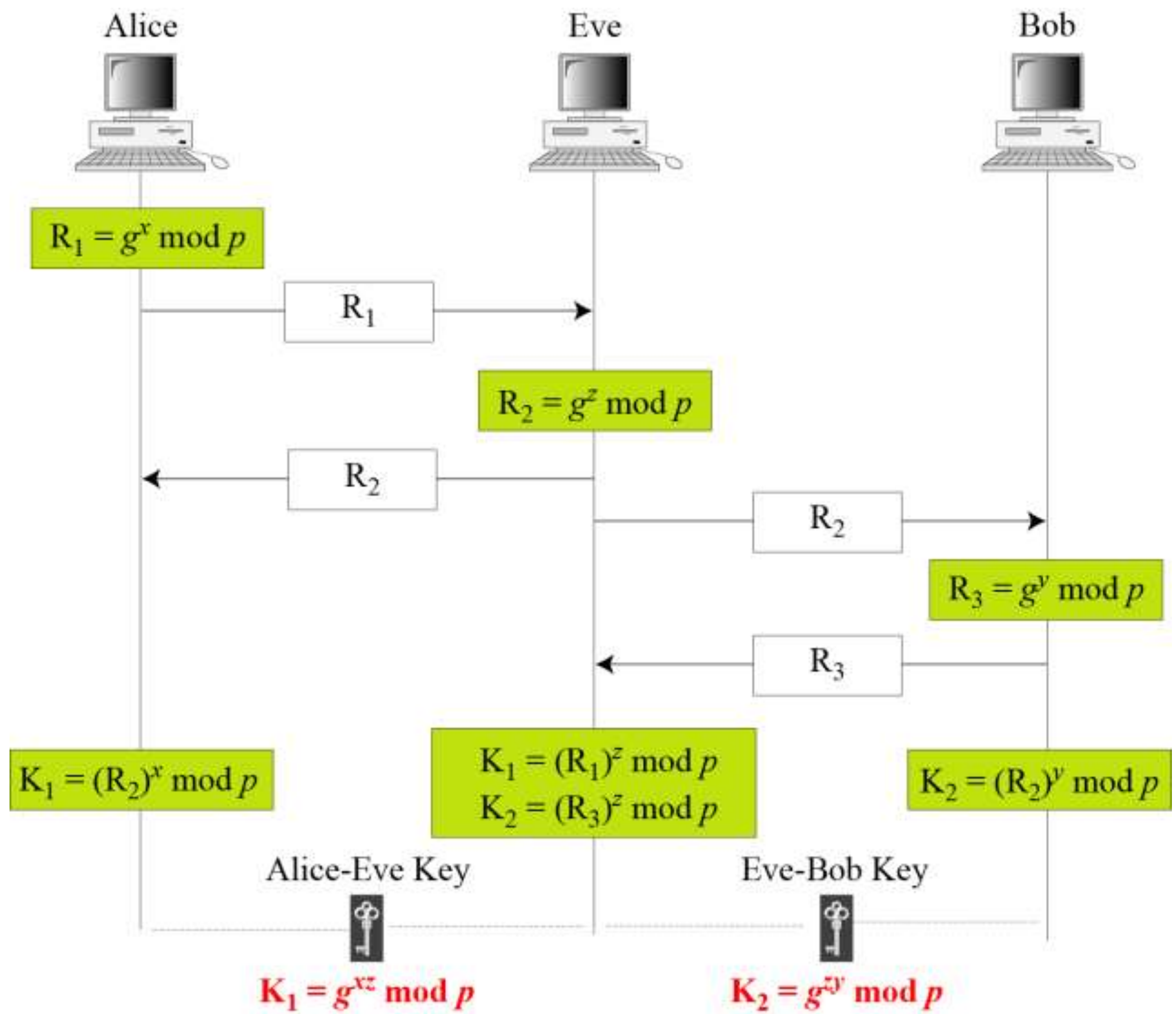
Security of Diffie-Hellman

Discrete Logarithm Attack

Man-in-the-Middle Attack

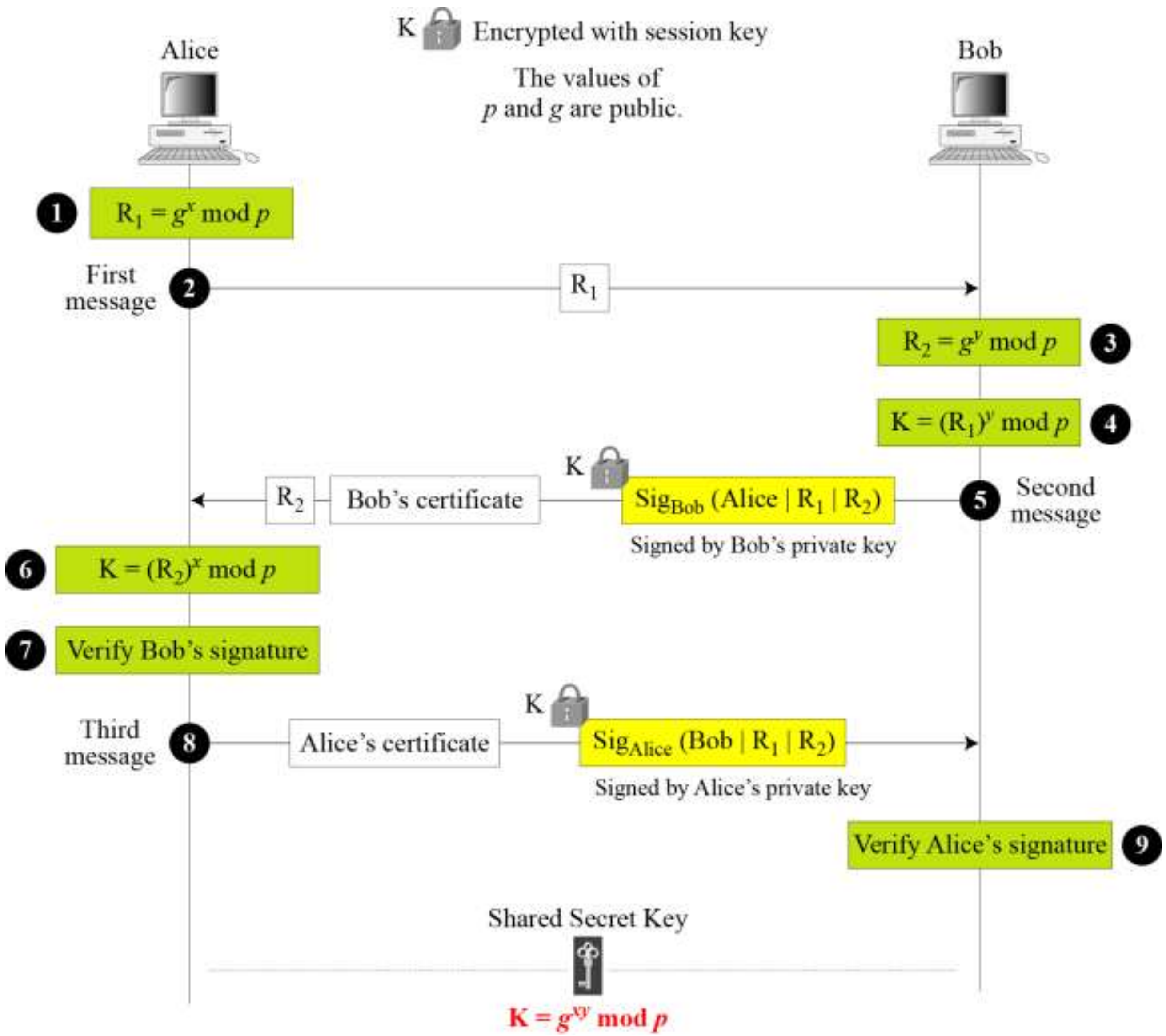
15.3.1 Continued

Figure 15.11 *Man-in-the-middle attack*



15.3.2 Station-to-Station Key Agreement

Figure 15.12 *Station-to-station key agreement method*



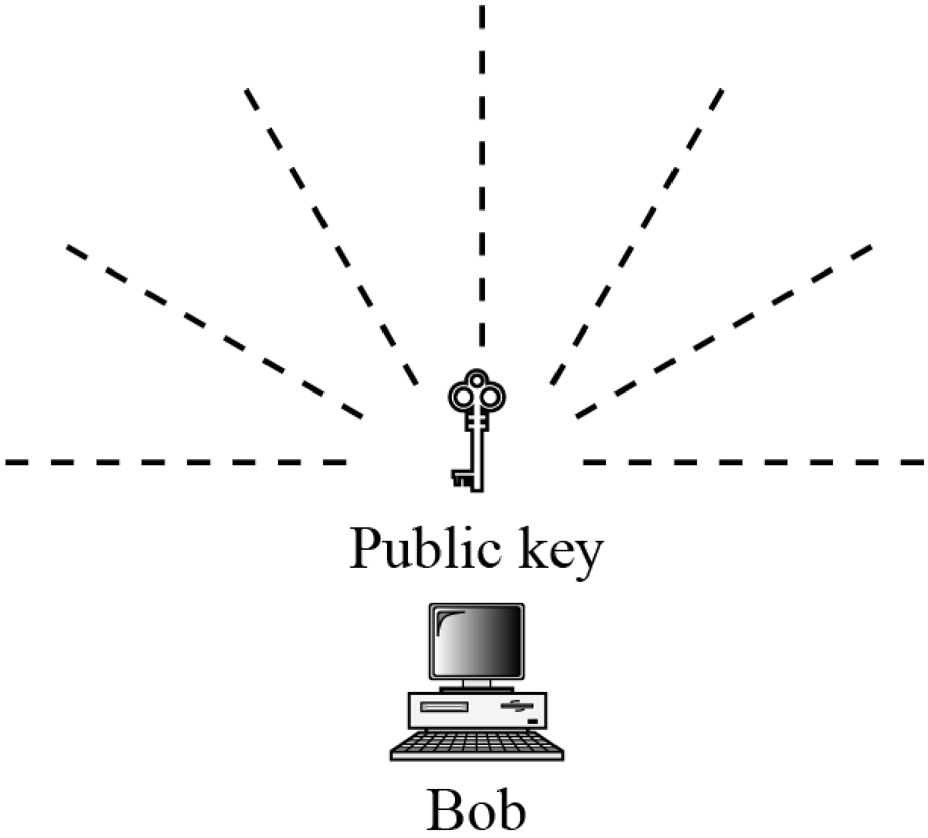
15-4 PUBLIC-KEY DISTRIBUTION

In asymmetric-key cryptography, people do not need to know a symmetric shared key; everyone shields a private key and advertises a public key.

Topics discussed in this section:

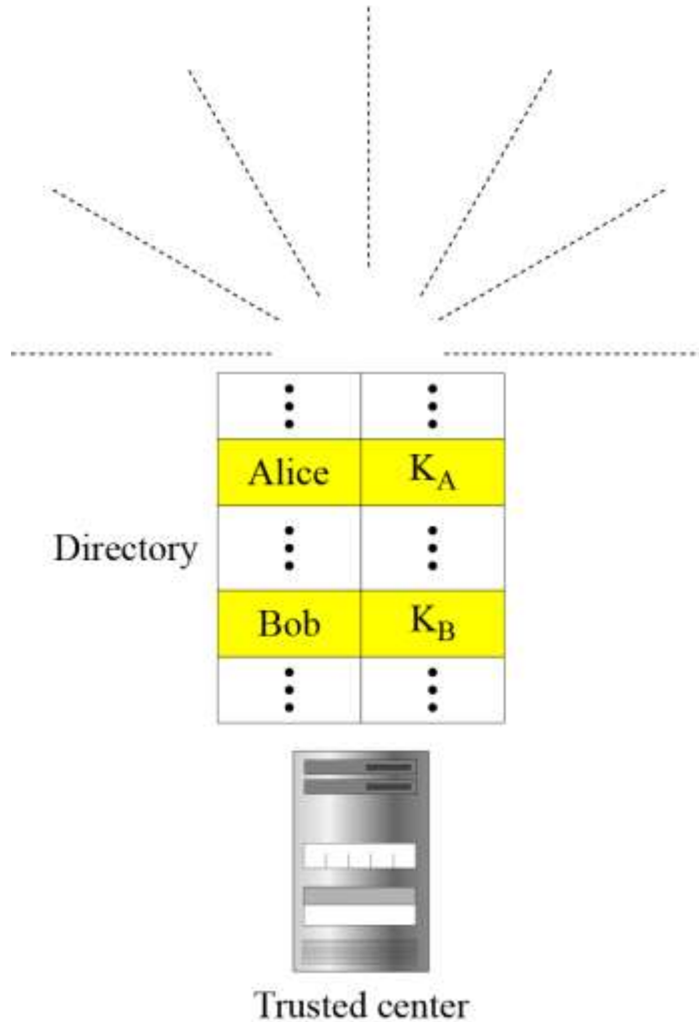
15.4.1 Public Announcement

Figure 15.13 ***Announcing a public key***



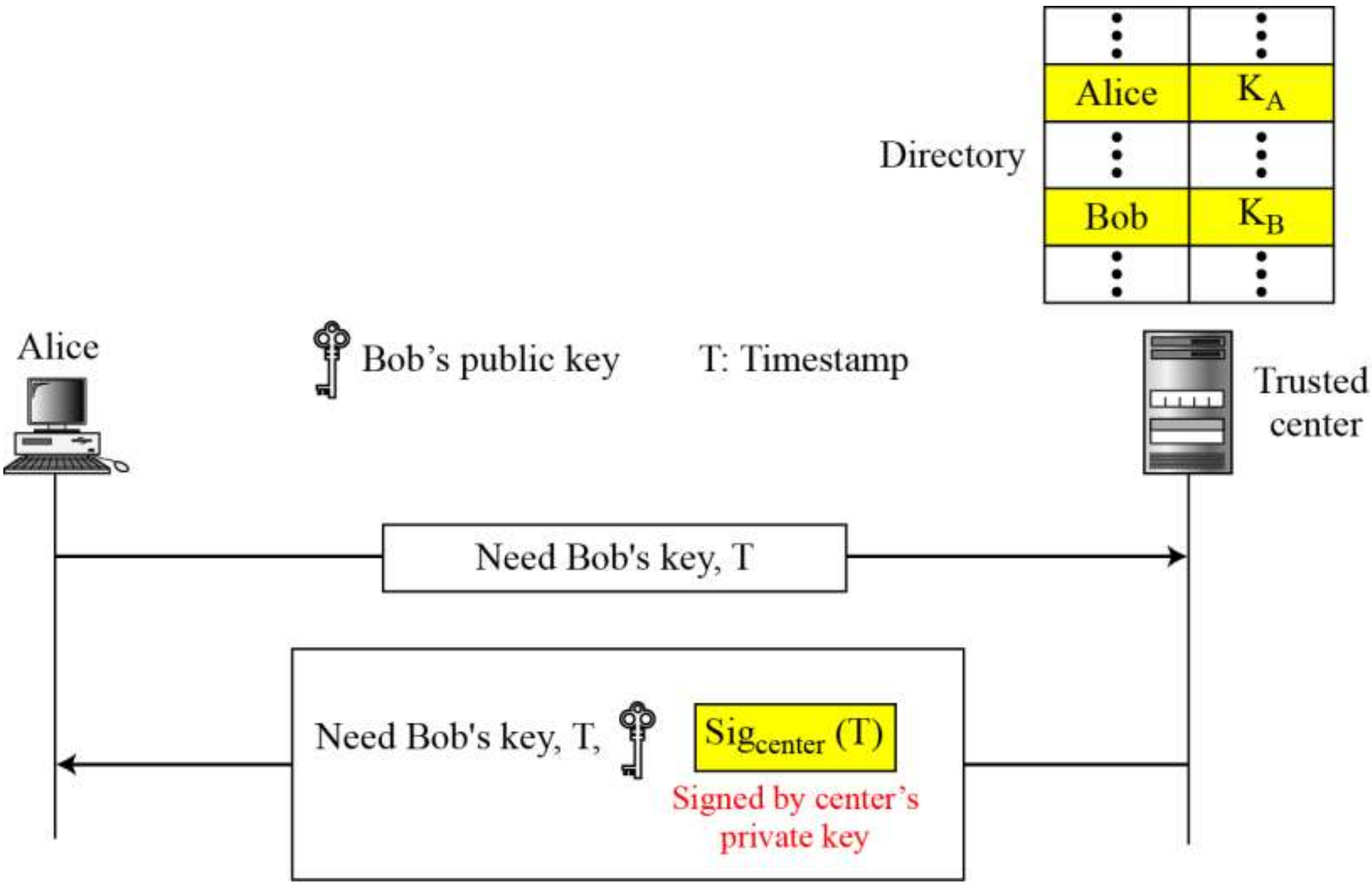
15.4.2 Trusted Center

Figure 15.14 ***Trusted center***



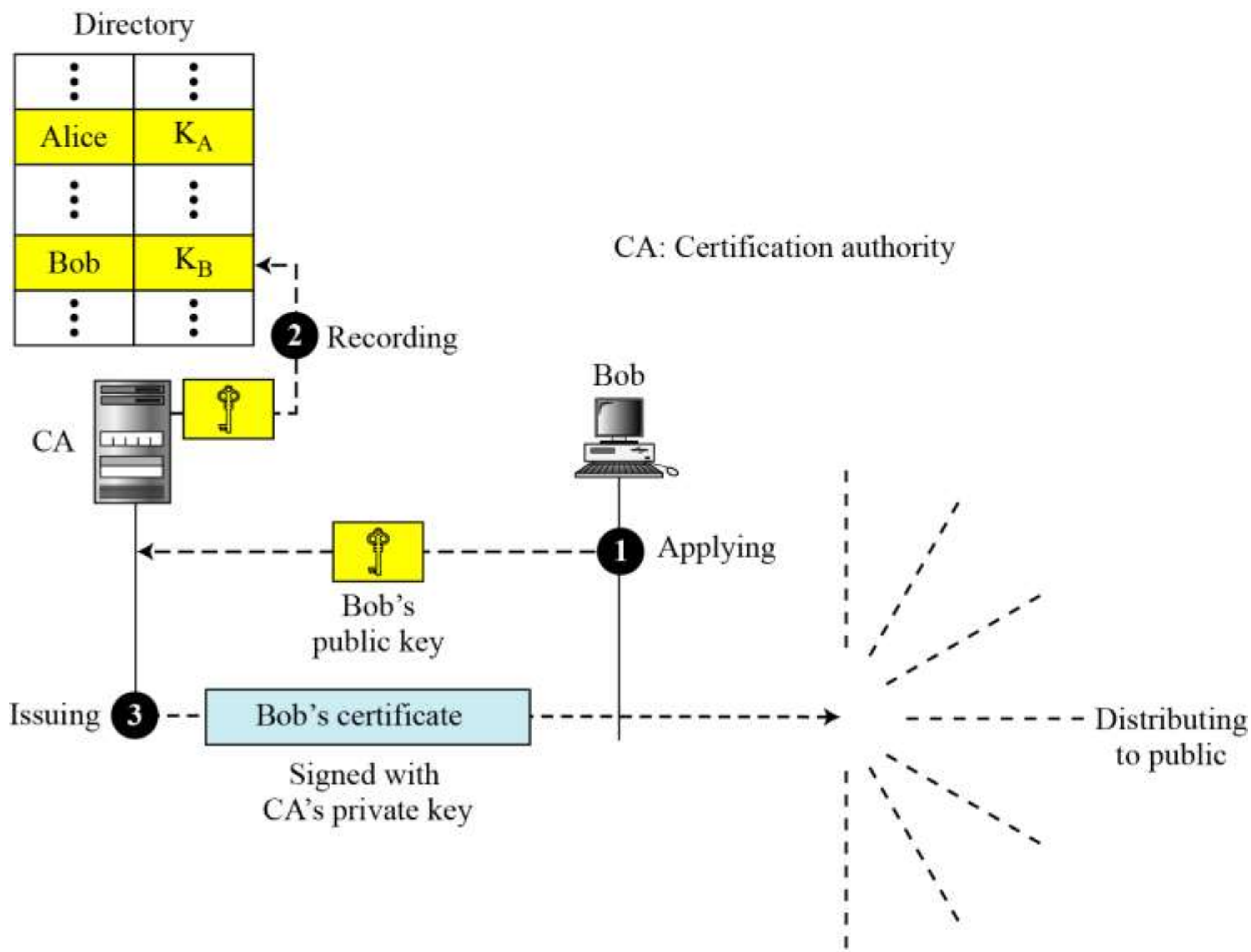
15.4.3 Controlled Trusted Center

Figure 15.15 *Controlled trusted center*



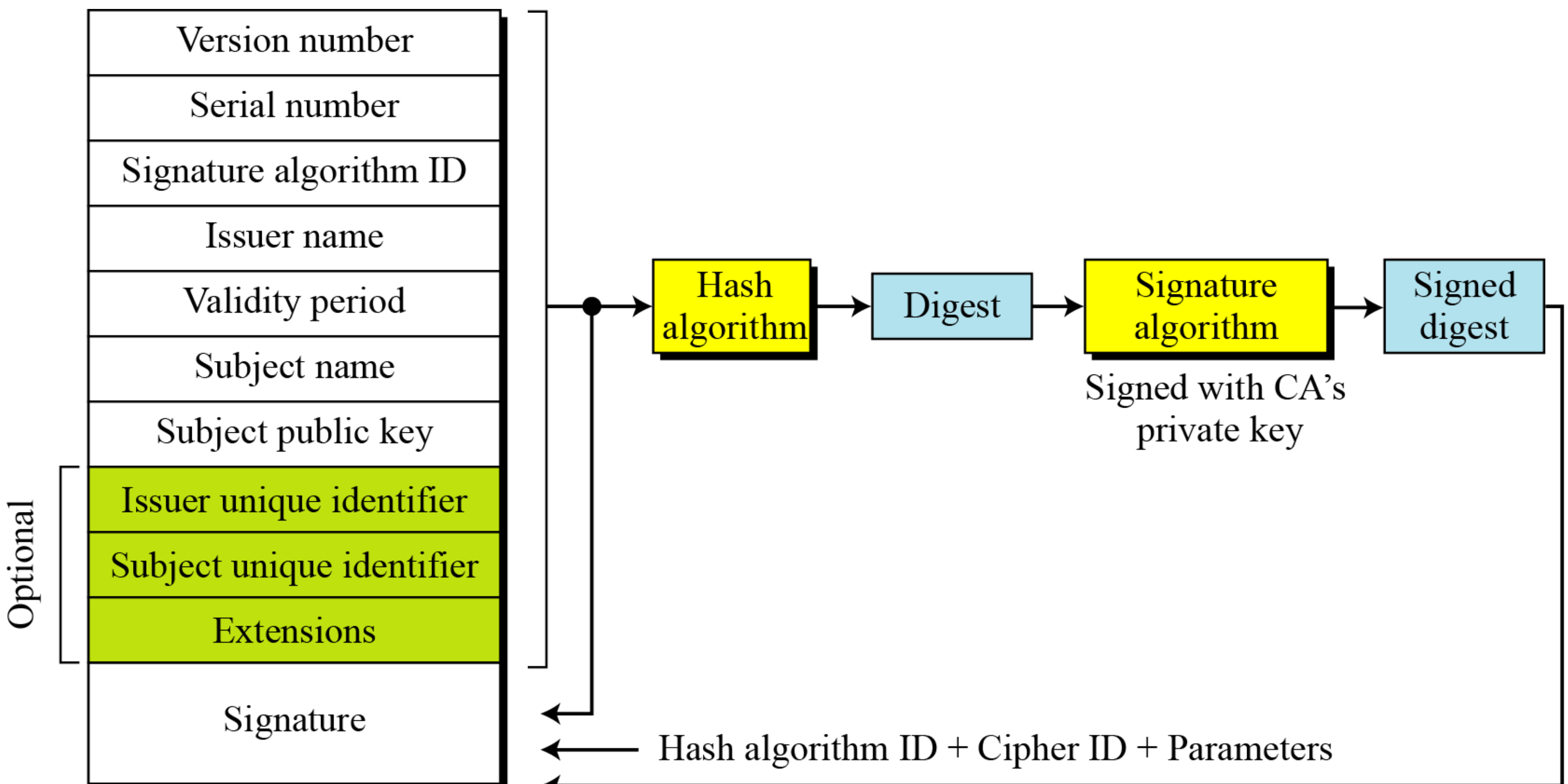
15.4.4 Certification Authority

Figure 15.16 *Certification authority*



Certificate

Figure 15.17 shows the format of a certificate.



15.4.5 Continued

Certificate Renewal

Each certificate has a period of validity. If there is no problem with the certificate, the CA issues a new certificate before the old one expires.

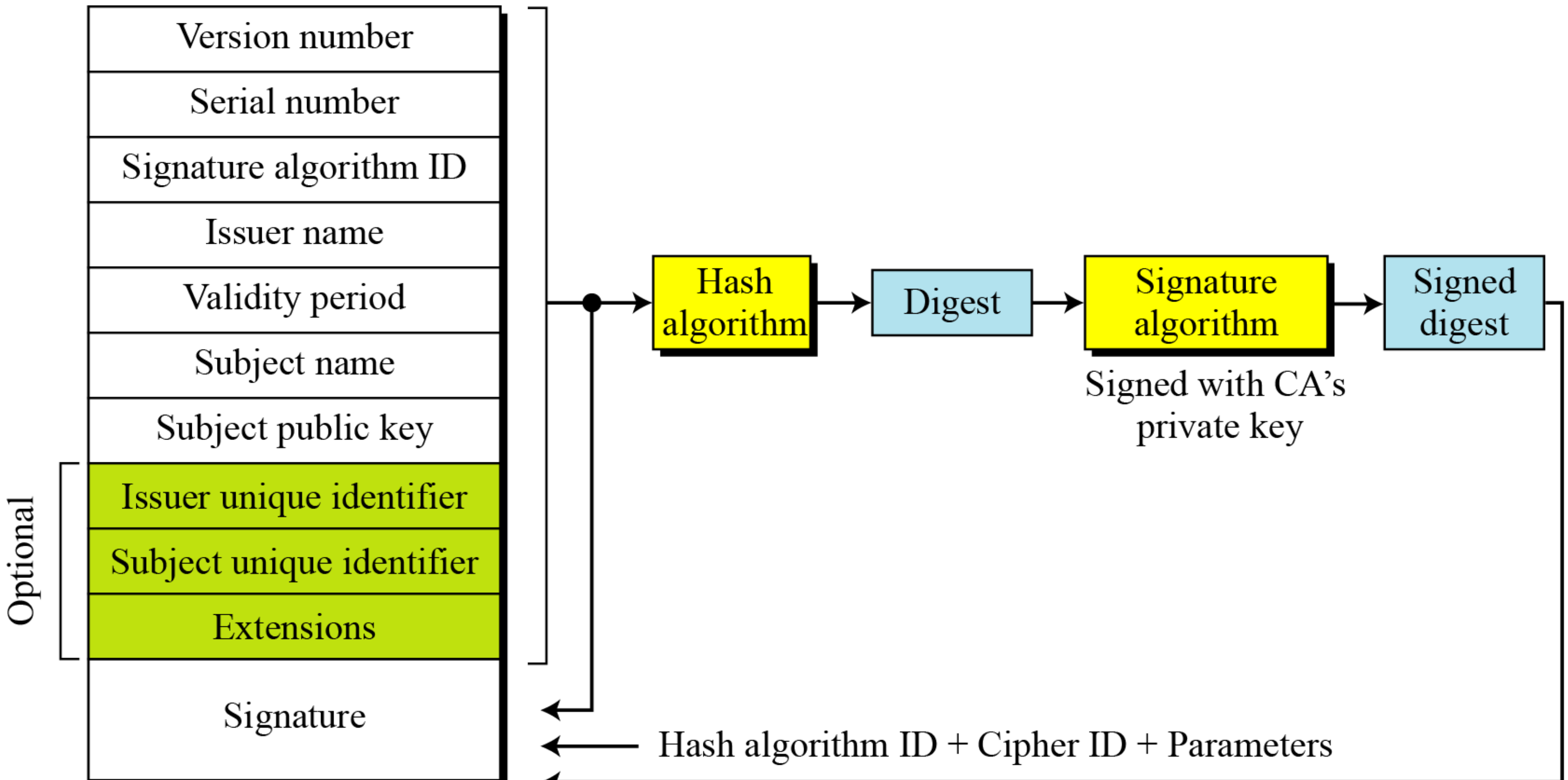
Certificate Renewal

In some cases a certificate must be revoked before its expiration.

Delta Revocation

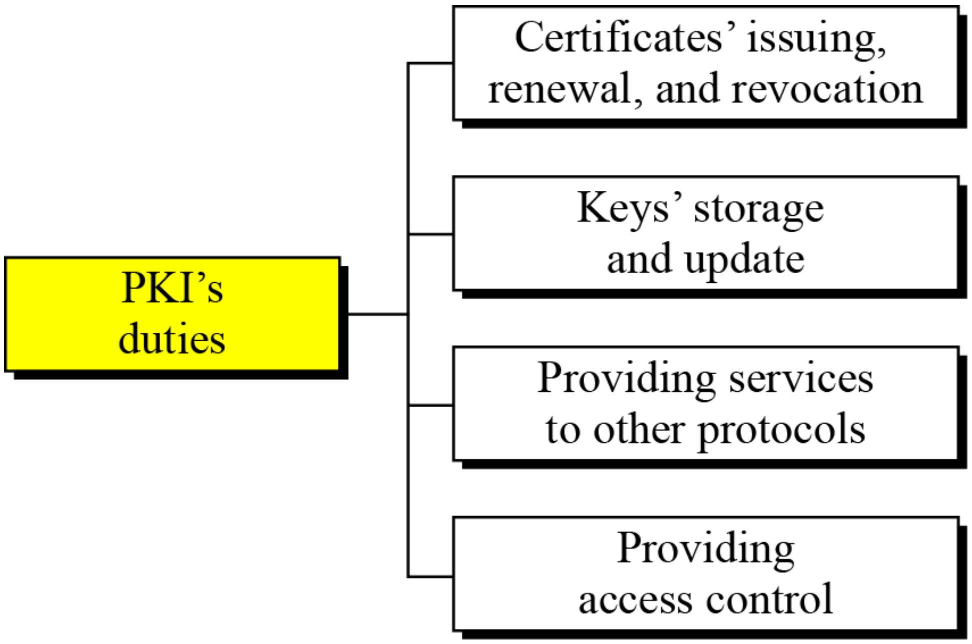
To make revocation more efficient, the delta certificate revocation list (delta CRL) has been introduced.

Figure 15.17 *Certificate revocation format*



15.4.6 Public-Key Infrastructures (PKI)

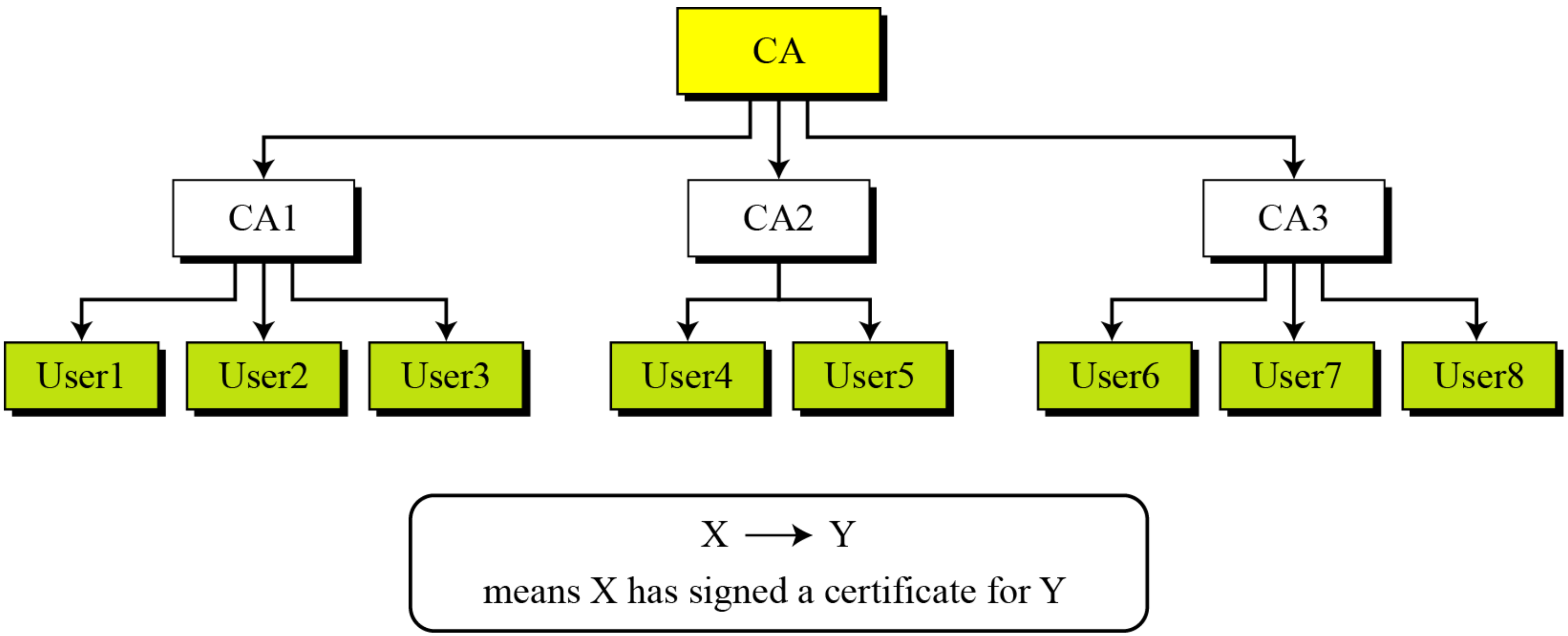
Figure 15.19 ***Some duties of a PKI***



15.4.6 Continued

Trust Model

Figure 15.20 *PKI hierarchical model*



Example 15.3

Show how User1, knowing only the public key of the CA (the root), can obtain a verified copy of User3's public key.

Solution

User3 sends a chain of certificates, CA<<CA1>> and CA1<<User3>>, to User1.

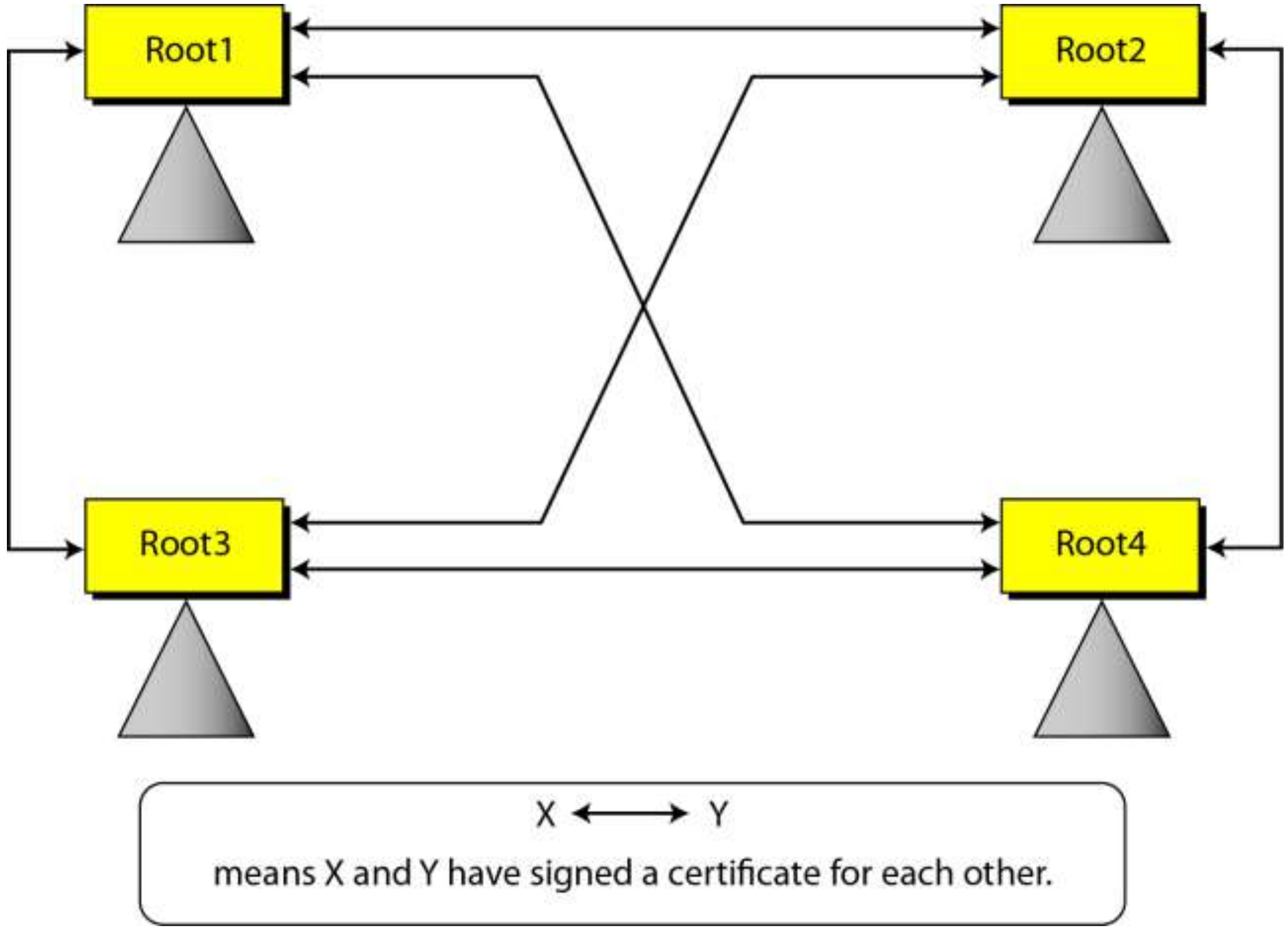
a. User1 validates CA<<CA1>> using the public key of CA.
b. User1 extracts the public key of CA1 from CA<<CA1>>.
c. User1 validates CA1<<User3>> using the public key of CA1.
d. User1 extracts the public key of User 3 from CA1<<User3>>.

Example 15.4

Some Web browsers, such as Netscape and Internet Explorer, include a set of certificates from independent roots without a single, high-level, authority to certify each root. One can find the list of these roots in the Internet Explorer at Tools/Internet Options/Contents/Certificate/Trusted roots (using pull-down menu). The user then can choose any of this root and view the certificate.

15.4.6 Continued

Figure 15.21 *Mesh model*



15.4.6 *Continued*

Example 15.5

Alice is under the authority Root1; Bob is under the authority Root4. Show how Alice can obtain Bob's verified public key.

Solution

Bob sends a chain of certificates from Root4 to Bob. Alice looks at the directory of Root1 to find Root1<<Root1>> and Root1<< Root4>> certificates. Using the process shown in Figure 15.21, Alice can verify Bob's public key.