



Information Security System (CSX-408)

Key Management

Dr Samayveer Singh
Assistant Professor

Department of Computer Science & Engineering
National Institute Technology Jalandhar, Punjab, India
samays@nitj.ac.in

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.

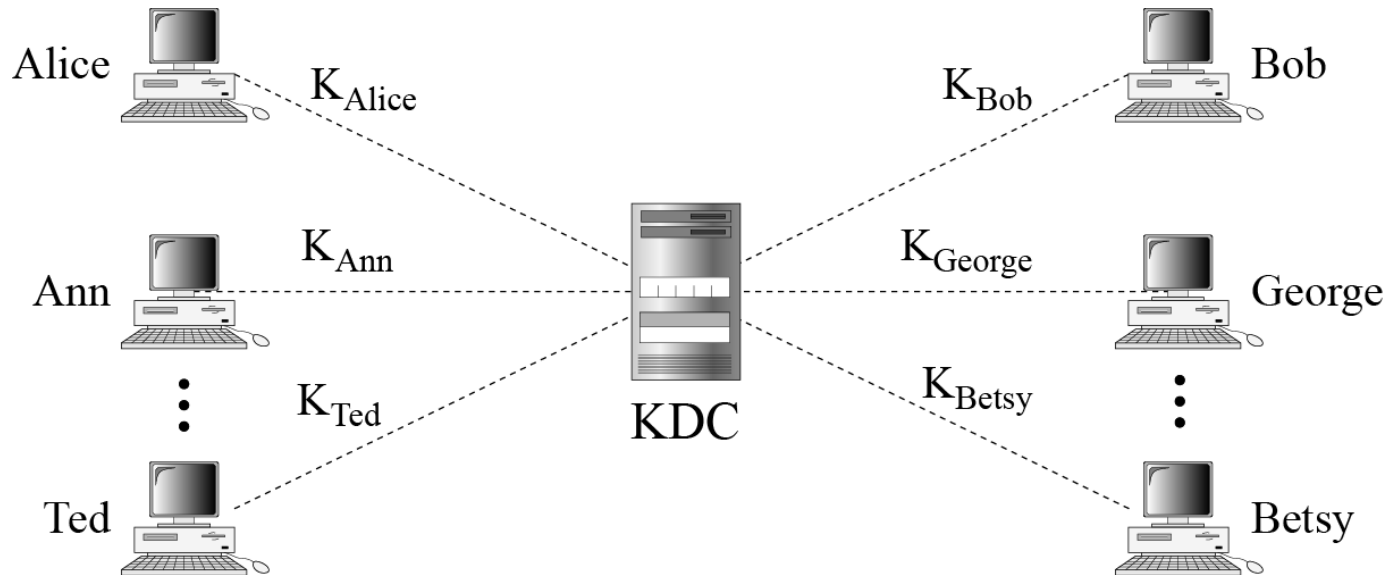
Topics discussed in this section:

15.1.1 Key-Distribution Center: KDC

15.1.2 Session Keys

15.1.1 Key-Distribution Center: KDC

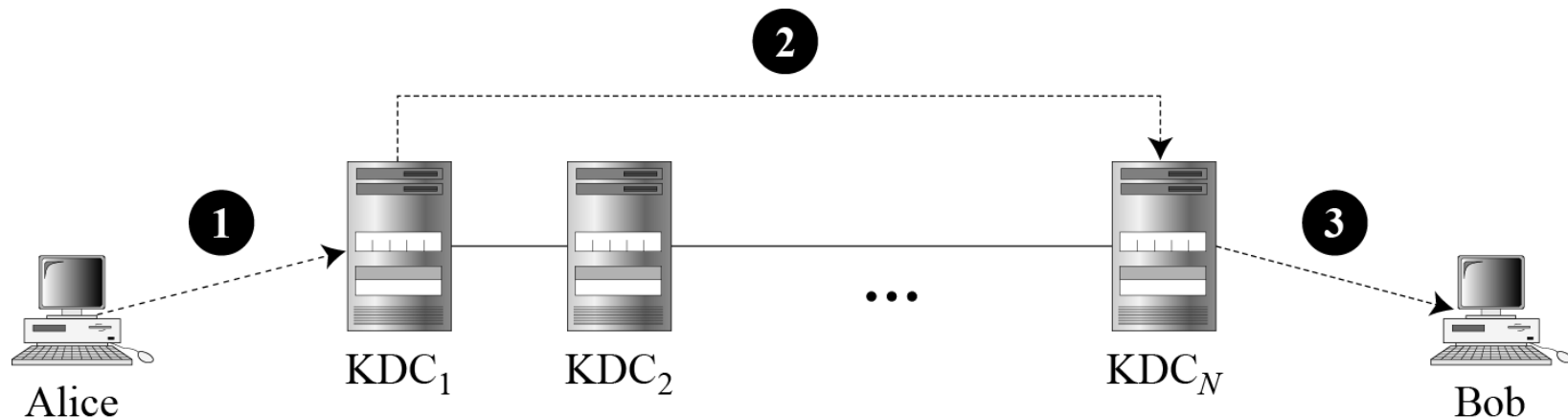
Figure 15.1 *Key-distribution center (KDC)*



15.1.1 Continued

Flat Multiple KDCs.

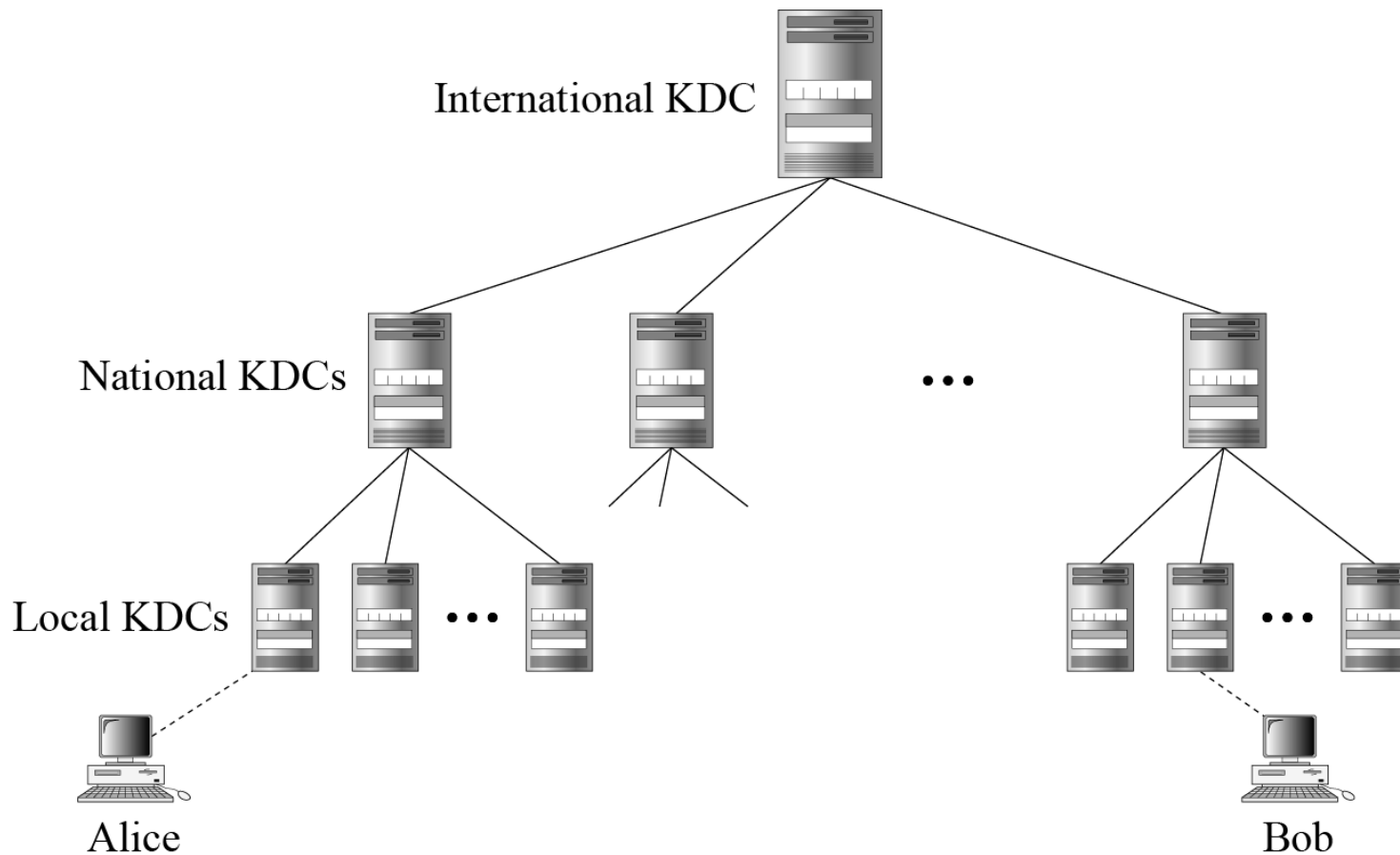
Figure 15.2 *Flat multiple KDCs*



15.1.1 Continued

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.

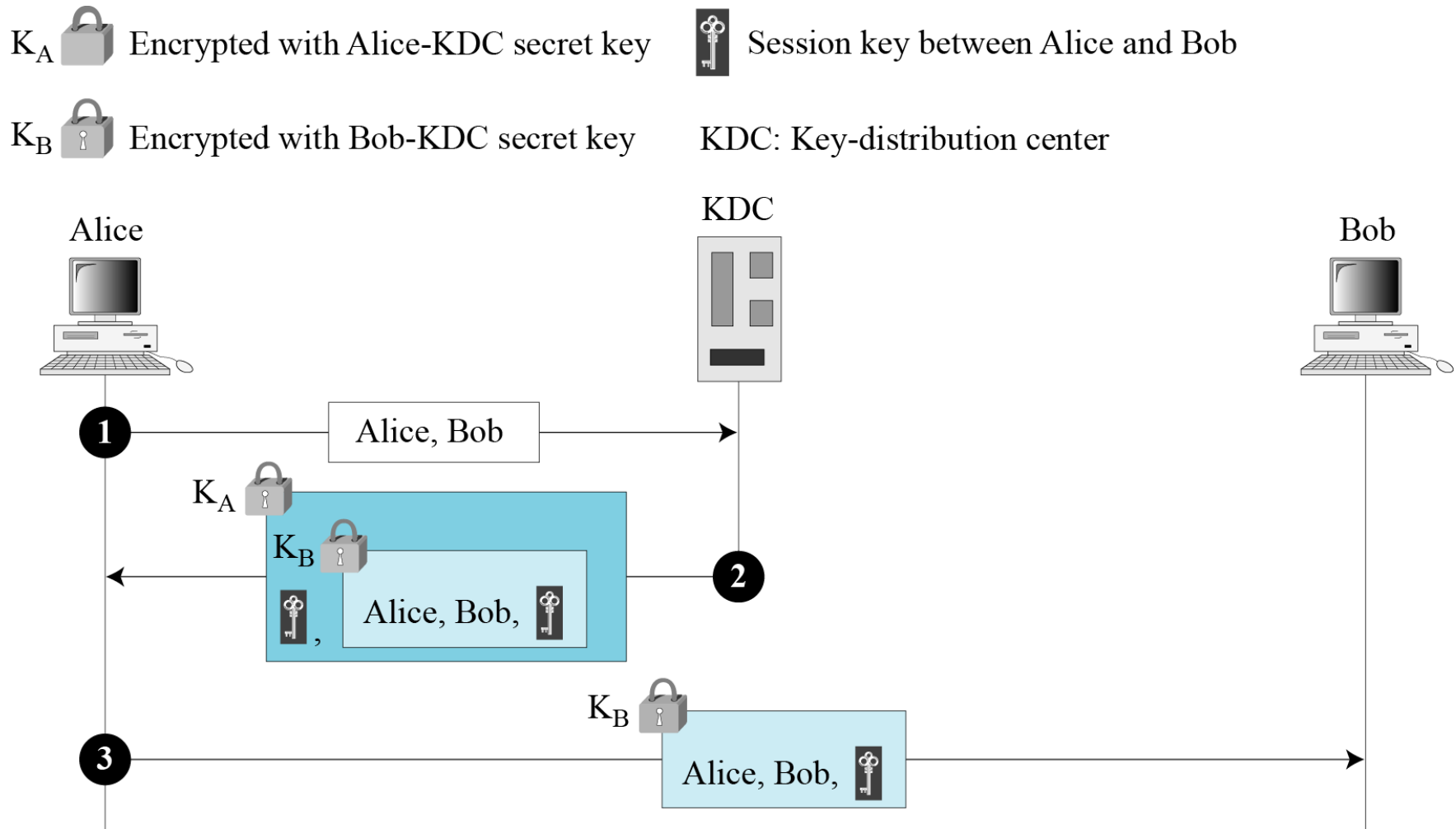
Note

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

15.1.2 Continued

A Simple Protocol Using a KDC

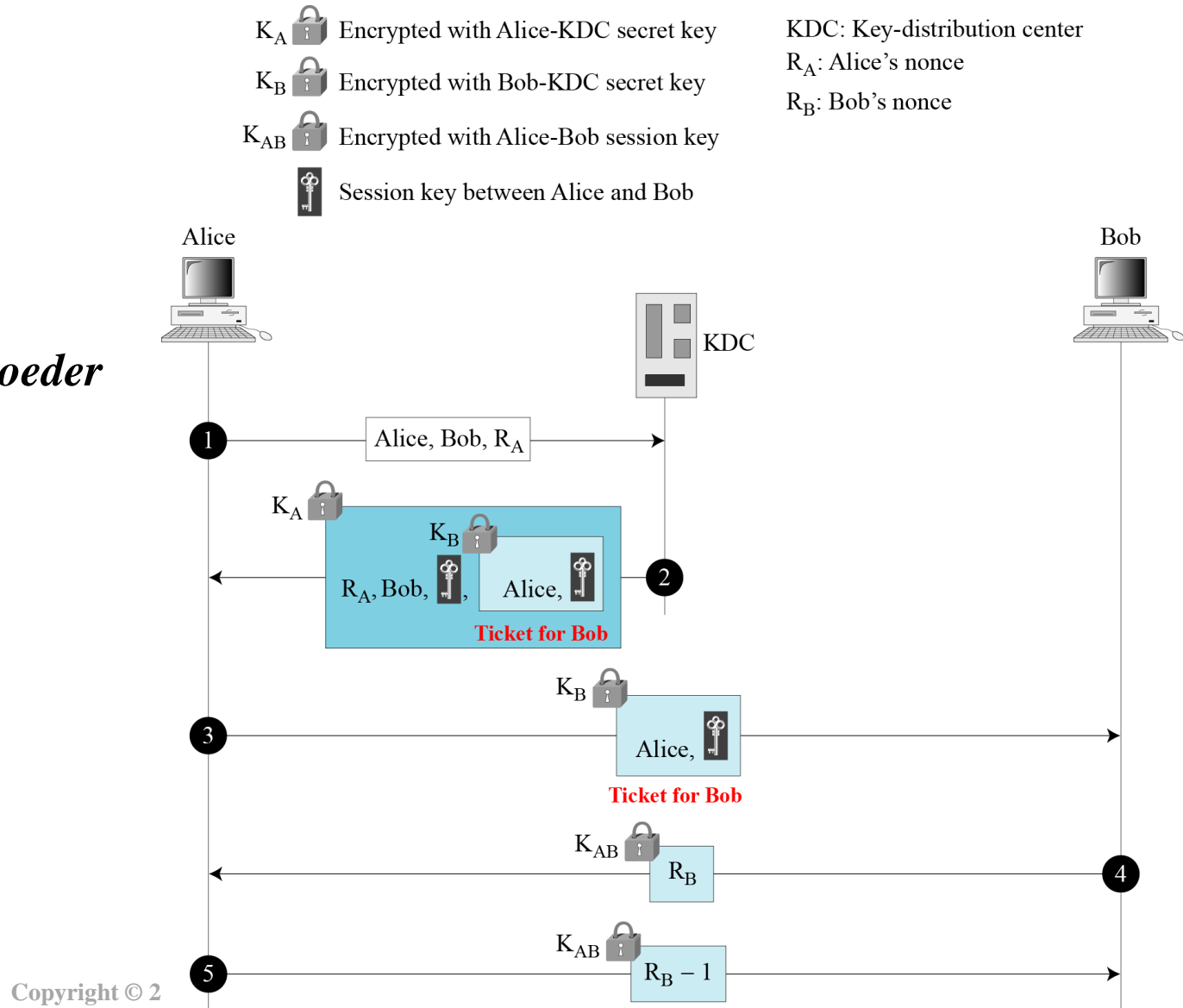
Figure 15.4 *First approach using KDC*



15.1.2 Continued

Needham-Schroeder Protocol

Figure 15.5
Needham-Schroeder protocol



15.1.2 Continued

Otway-Rees Protocol

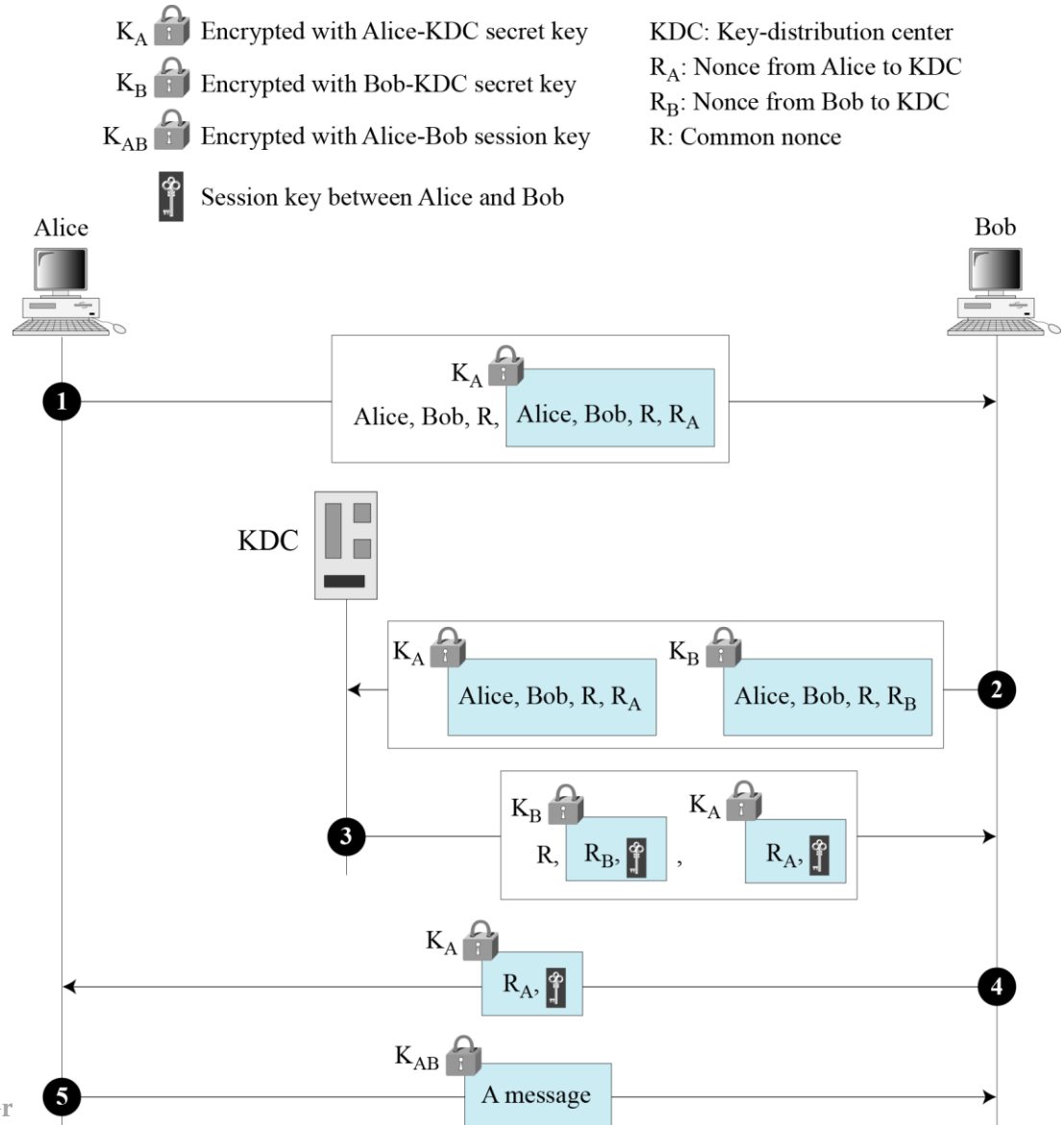


Figure 15.6
Otway-Rees protocol

10-5 Diffie Hellman Key Exchange Algorithm

The purpose of the algorithm is to enable two users to securely exchange a key that can then be used for subsequent asymmetric encryption of messages.

10.5 Procedure



Alice



Bob

Alice and Bob share a prime number q and an integer α , such that $\alpha < q$ and α is a primitive root of q

Alice and Bob share a prime number q and an integer α , such that $\alpha < q$ and α is a primitive root of q

Alice generates a private key X_A such that $X_A < q$

Bob generates a private key X_B such that $X_B < q$

Alice calculates a public key $Y_A = \alpha^{X_A} \bmod q$

Bob calculates a public key $Y_B = \alpha^{X_B} \bmod q$

Alice receives Bob's public key Y_B in plaintext

Bob receives Alice's public key Y_A in plaintext

Alice calculates shared secret key $K = (Y_B)^{X_A} \bmod q$

Bob calculates shared secret key $K = (Y_A)^{X_B} \bmod q$



10.5 Procedure

Key exchange is based on the use of the prime number $q = 353$ and a primitive root of 353, in this case $\alpha = 3$. A and B select private keys $X_A = 97$ and $X_B = 233$, respectively. Each computes its public key:

A computes $Y_A = 3^{97} \bmod 353 = 40$.

B computes $Y_B = 3^{233} \bmod 353 = 248$.

After they exchange public keys, each can compute the common secret key:

A computes $K = (Y_B)^{X_A} \bmod 353 = 248^{97} \bmod 353 = 160$.

B computes $K = (Y_A)^{X_B} \bmod 353 = 40^{233} \bmod 353 = 160$.

We assume an attacker would have available the following information:

$$q = 353; \alpha = 3; Y_A = 40; Y_B = 248$$

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

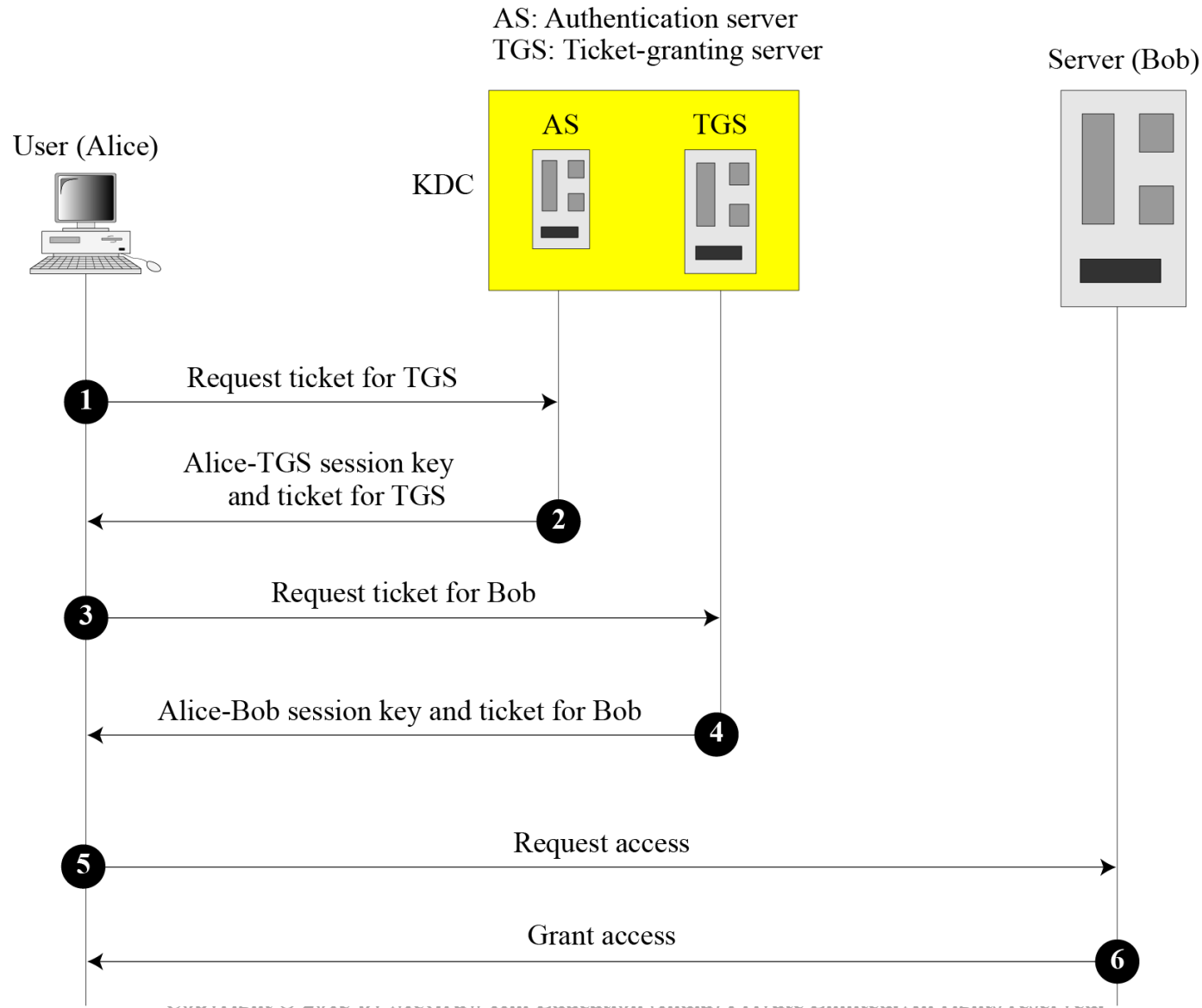
15.2.2 Operation

15.2.3 Using Different Servers

15.2.4 Kerberos Version 5

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)

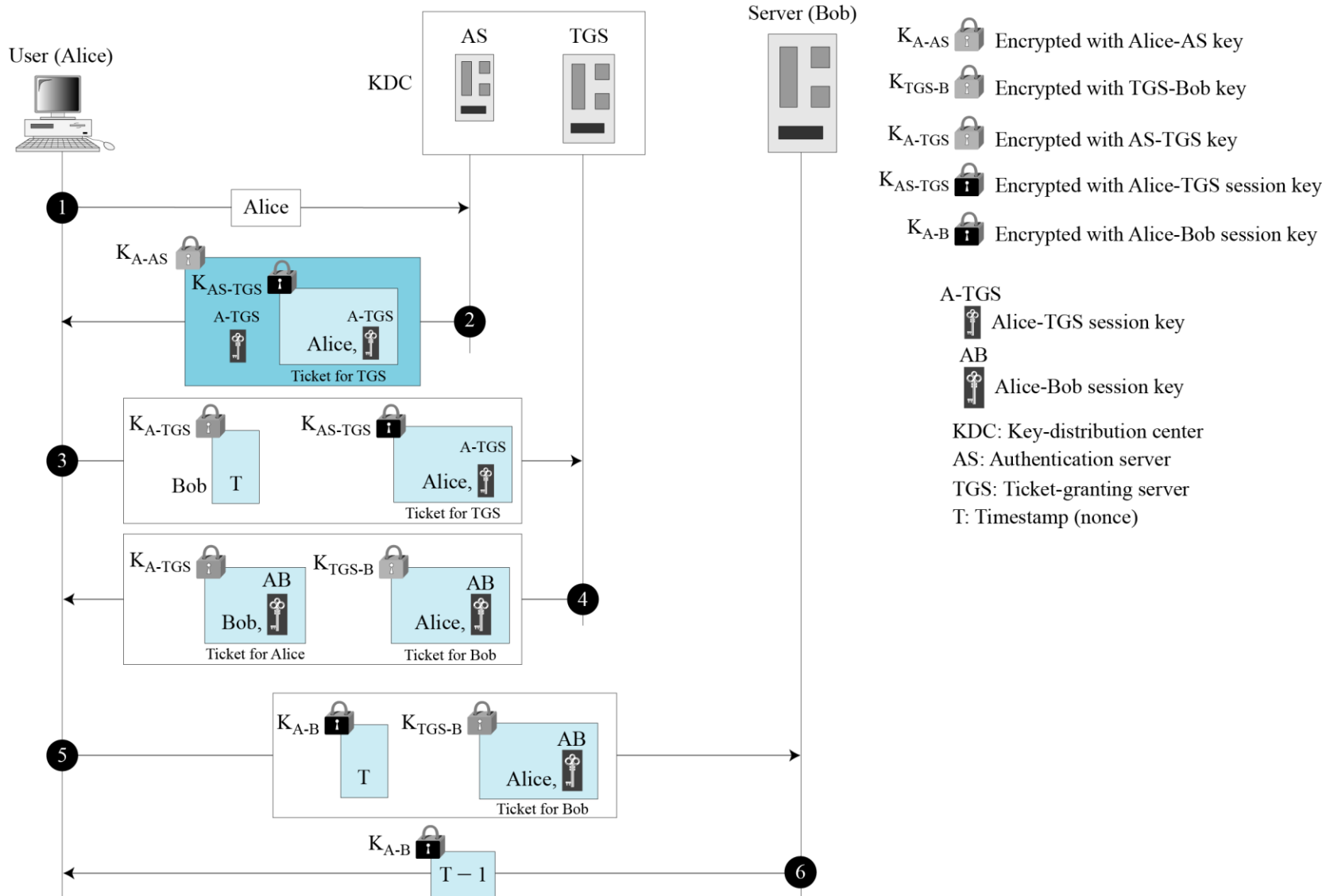
The ticket-granting server (TGS) issues a ticket for the real server (Bob).

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.*

15-3 SYMMETRIC-KEY AGREEMENT

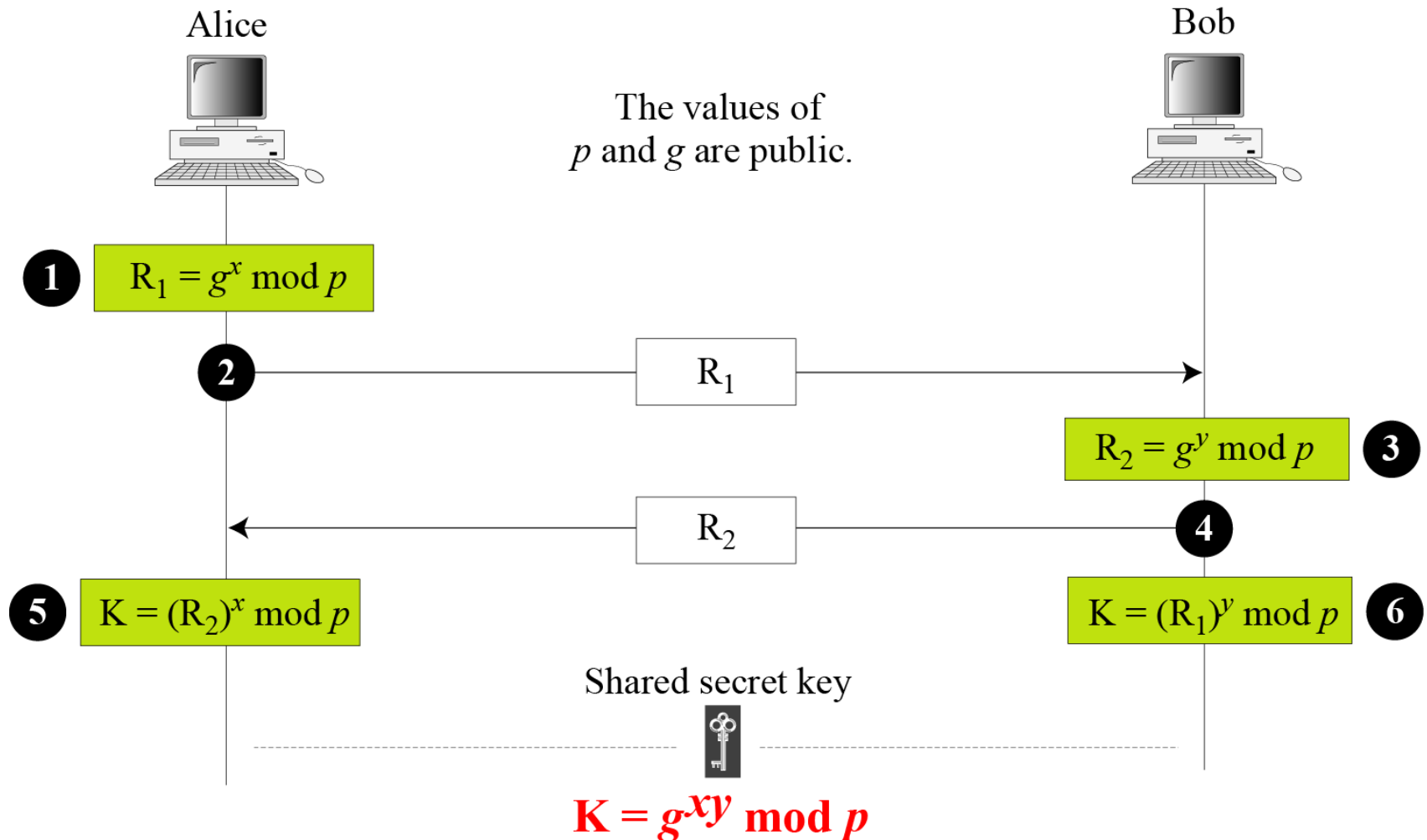
Alice and Bob can create a session key between themselves without using a KDC. This method of session-key creation is referred to as the symmetric-key agreement.

Topics discussed in this section:

15.3.1 Diffie-Hellman Key Agreement

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 $K = g^{xy} \bmod p$.

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 $R_1 = 7^3 \bmod 23 = 21$.
2. Bob chooses $y = 6$ and calculates $R_2 = 7^6 \bmod 23 = 4$.
3. Alice sends the number 21 to Bob.
4. Bob sends the number 4 to Alice.
5. Alice calculates the symmetric key $K = 4^3 \bmod 23 = 18$.
6. Bob calculates the symmetric key $K = 21^6 \bmod 23 = 18$.
7. The value of K is the same for both Alice and Bob;
 $g^{xy} \bmod p = 7^{18} \bmod 23 = 18$.

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	764624298563493572182493765955030507476338096726949748923573772860925 235666660755423637423309661180033338106194730130950414738700999178043 6548785807987581
g	2
x	557
y	273

15.3.1 Continued

Example 15.2

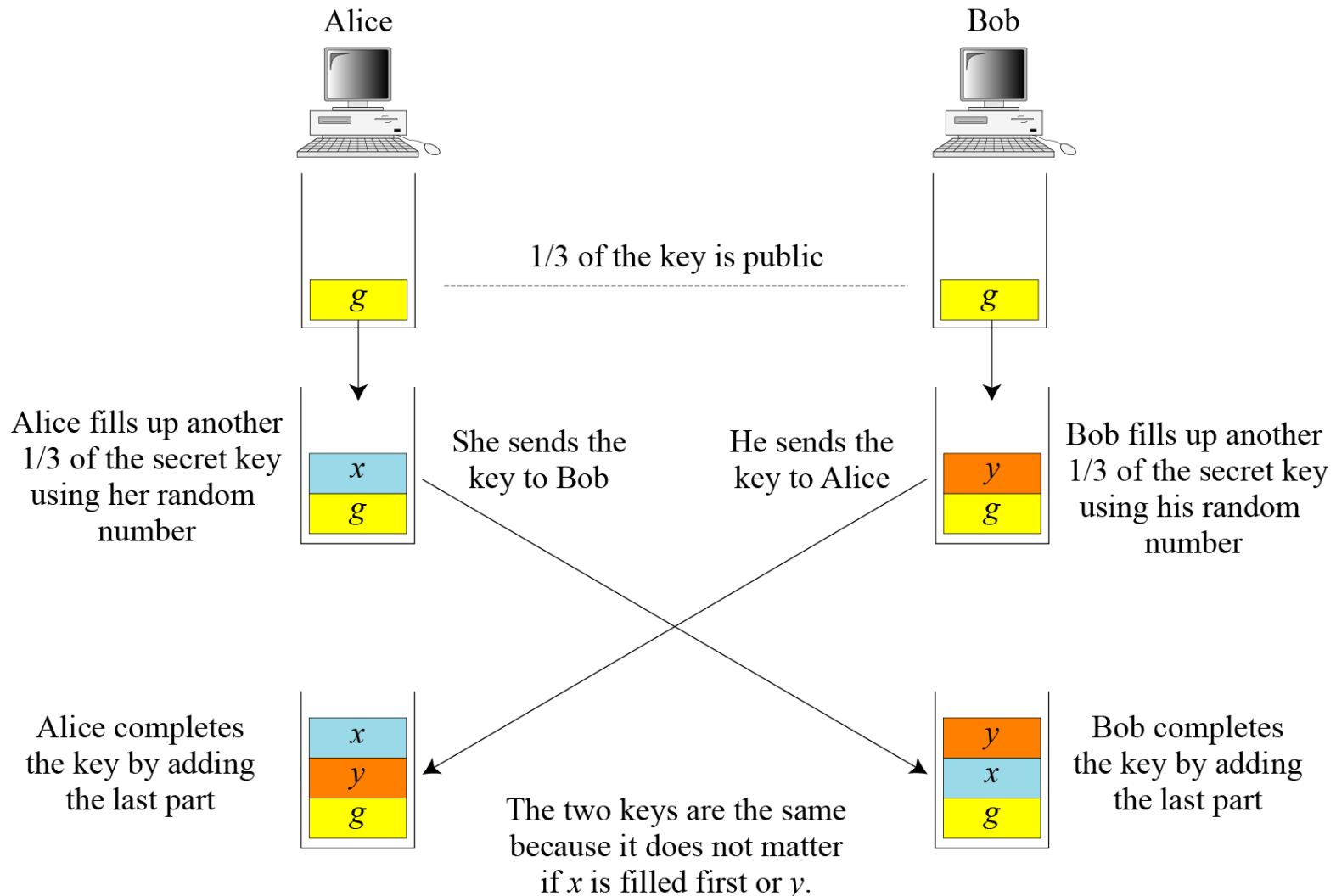
Continued

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

R_1	844920284205665505216172947491035094143433698520012660862863631067673 619959280828586700802131859290945140217500319973312945836083821943065 966020157955354
R_2	435262838709200379470747114895581627636389116262115557975123379218566 310011435718208390040181876486841753831165342691630263421106721508589 6255201288594143
K	155638000664522290596225827523270765273218046944423678520320400146406 500887936651204257426776608327911017153038674561252213151610976584200 1204086433617740

15.3.1 Continued

Figure 15.10 *Diffie-Hellman idea*



15.3.1 Continued

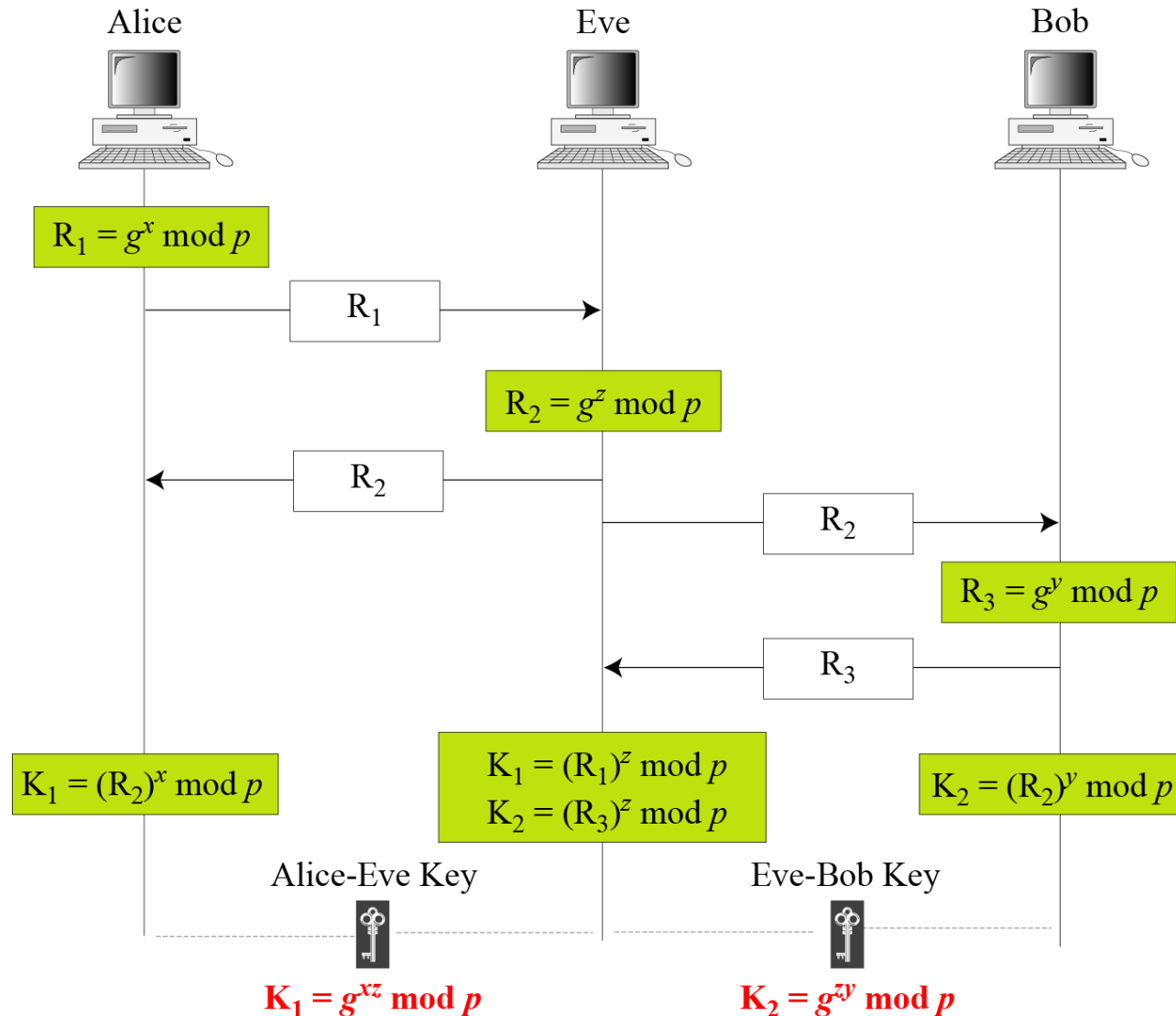
Security of Diffie-Hellman

Discrete Logarithm Attack

Man-in-the-Middle Attack

15.3.1 Continued

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



15-4 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

15.4.2 Trusted Center

15.4.3 Controlled Trusted Center

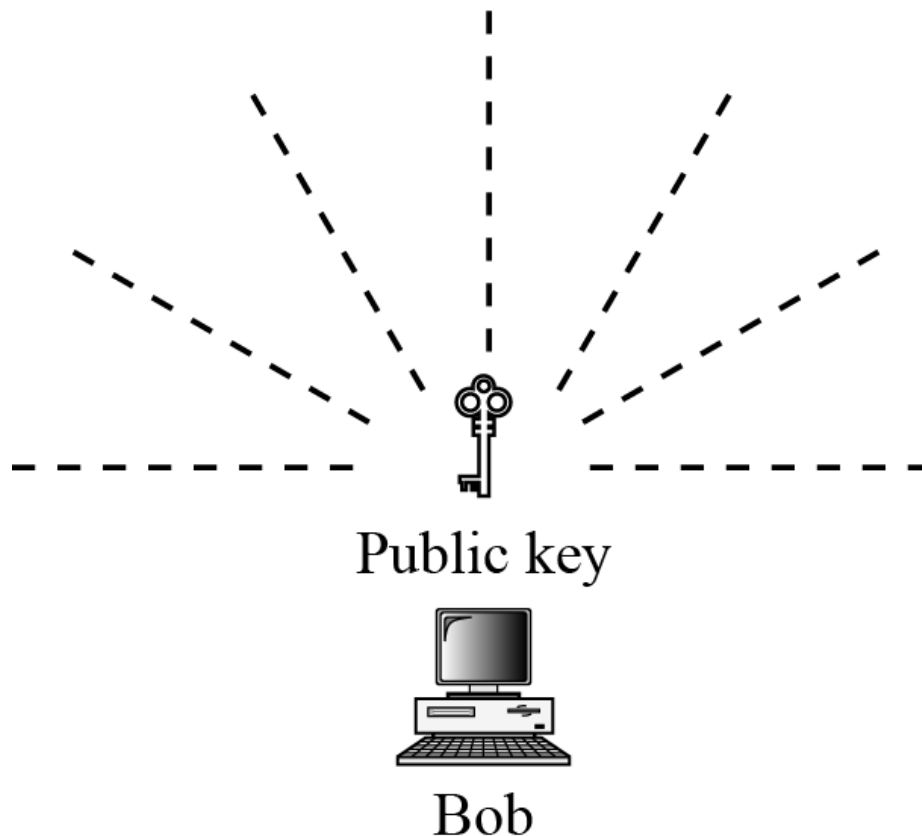
15.4.4 Certification Authority

15.4.5 X.509

15.4.6 Public-Key Infrastructures (PKI)

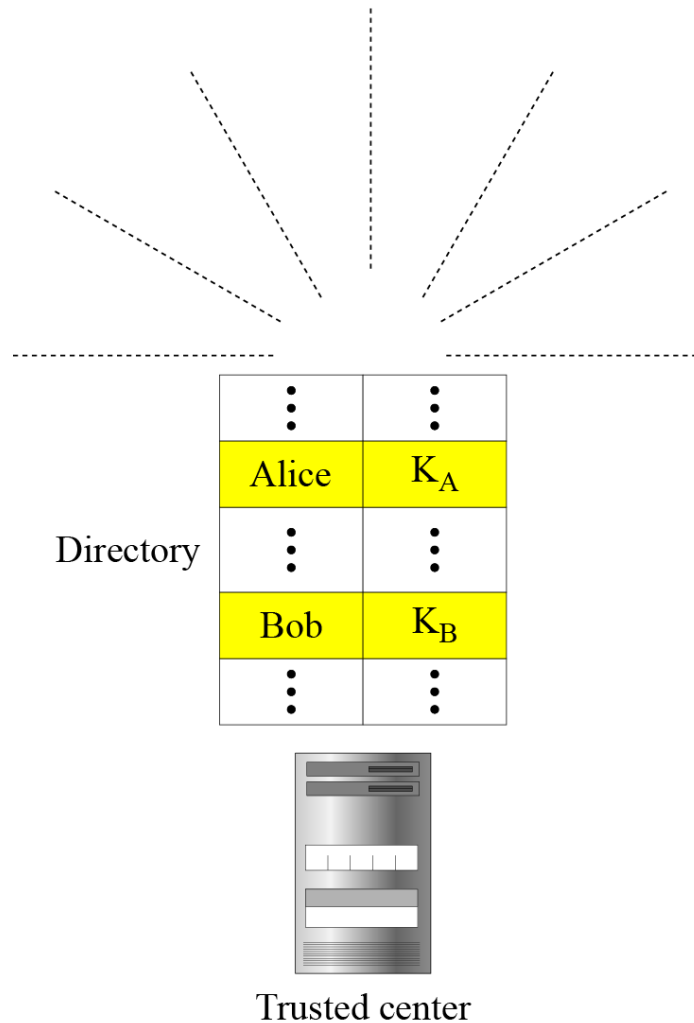
15.4.1 Public Announcement

Figure 15.13 *Announcing a public key*



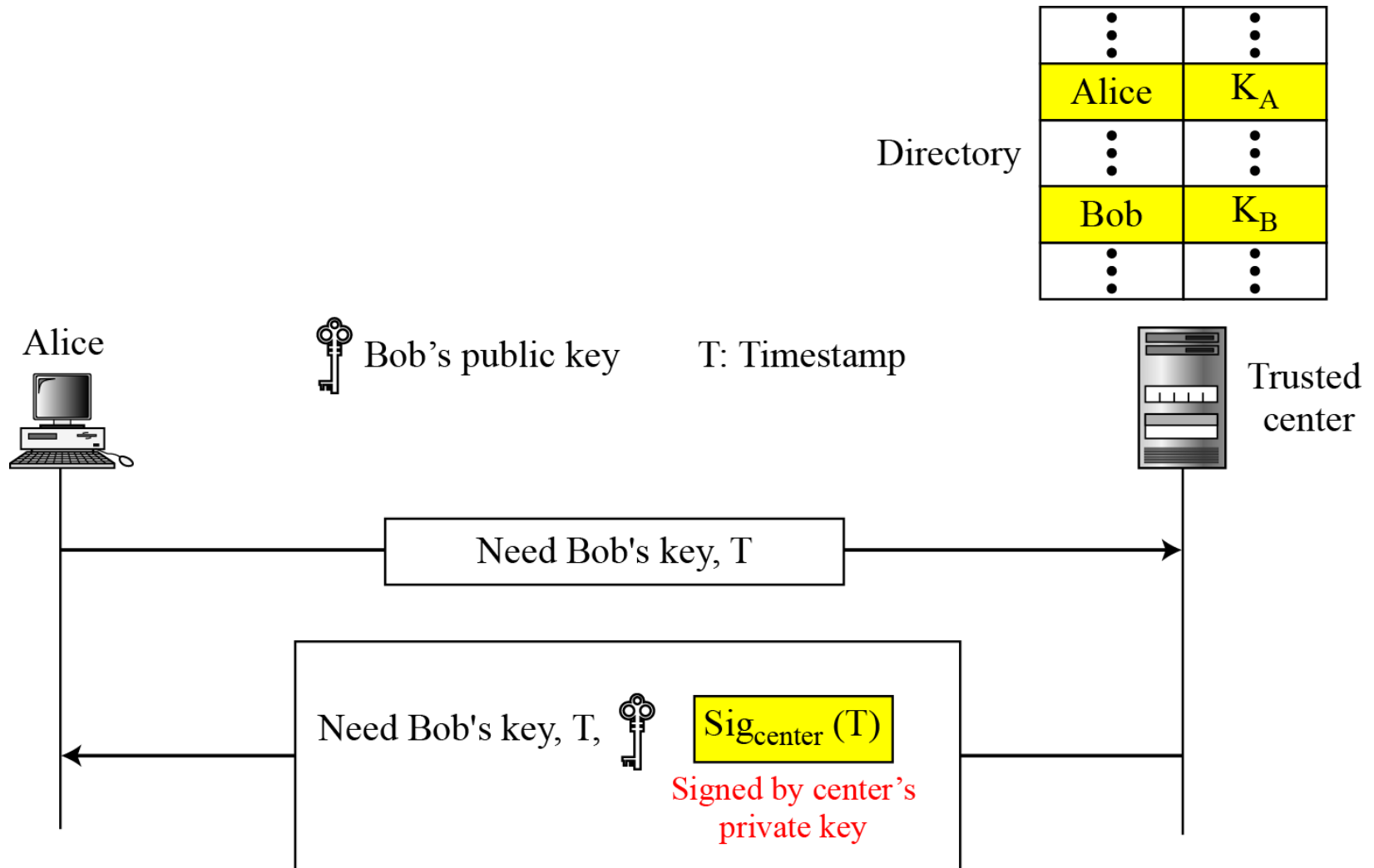
15.4.2 Trusted Center

Figure 15.14 *Trusted center*



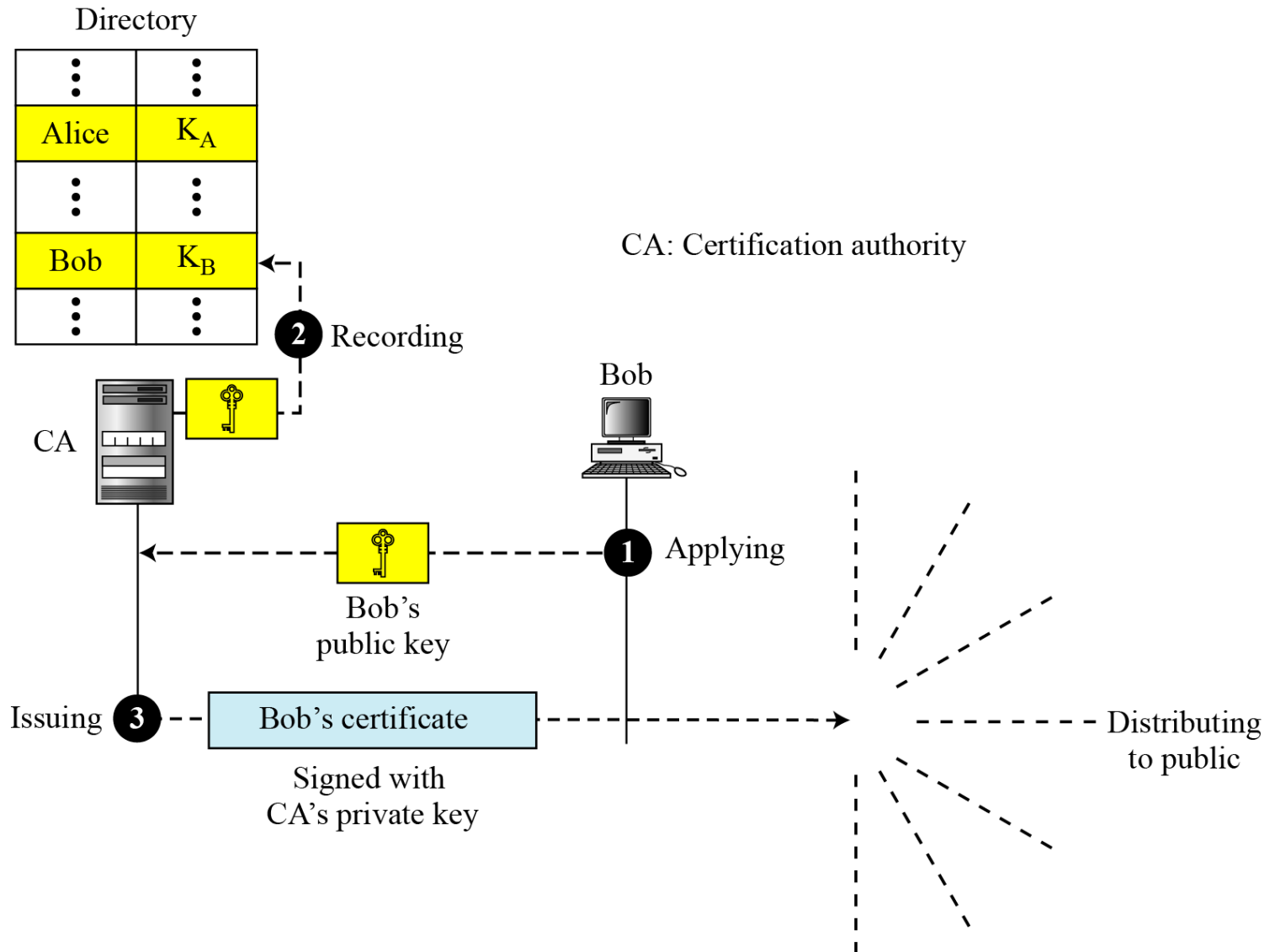
15.4.3 Controlled Trusted Center

Figure 15.15 *Controlled trusted center*



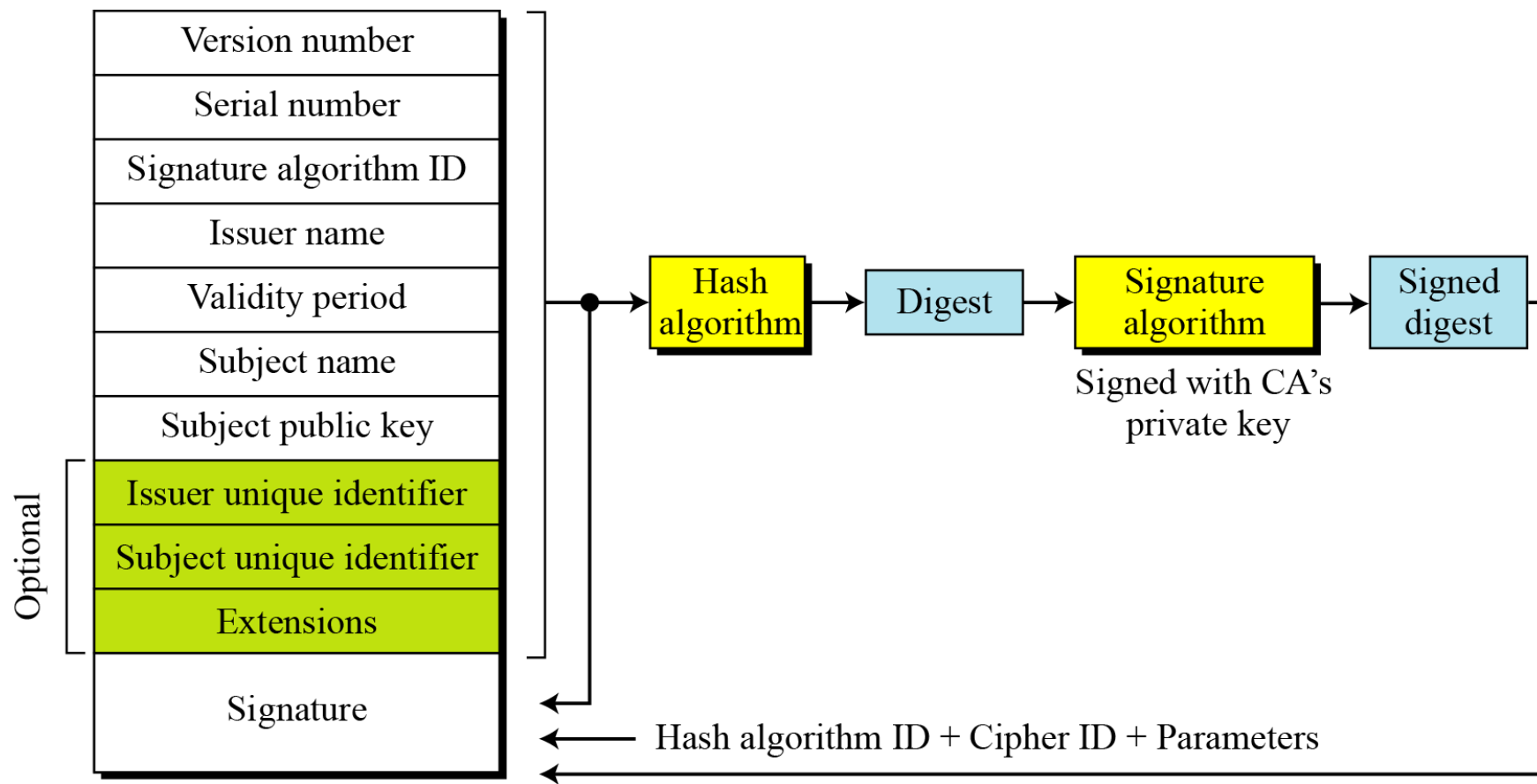
15.4.4 Certification Authority

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

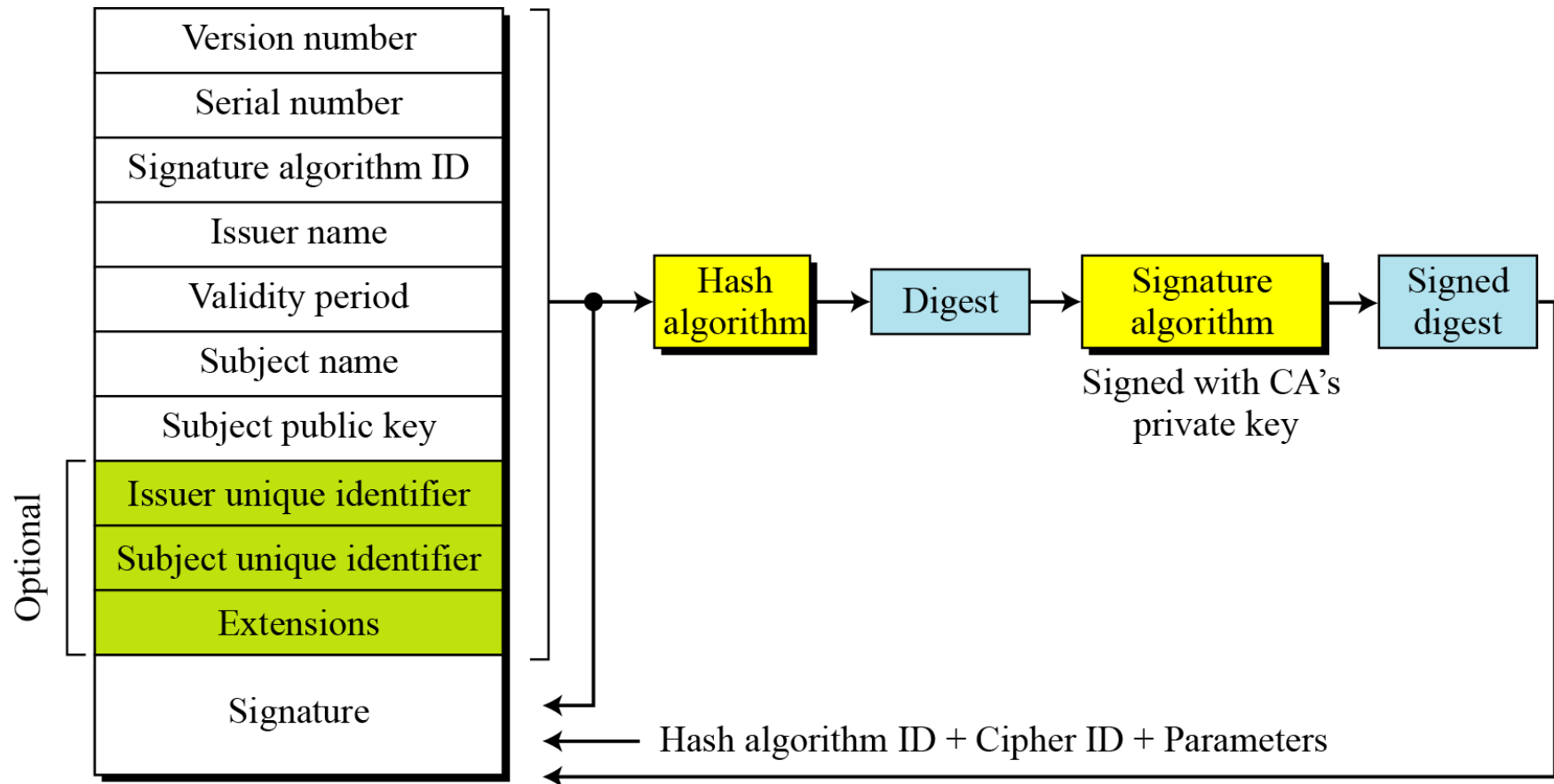
In some cases a certificate must be revoked before its expiration.

Delta Revocation

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

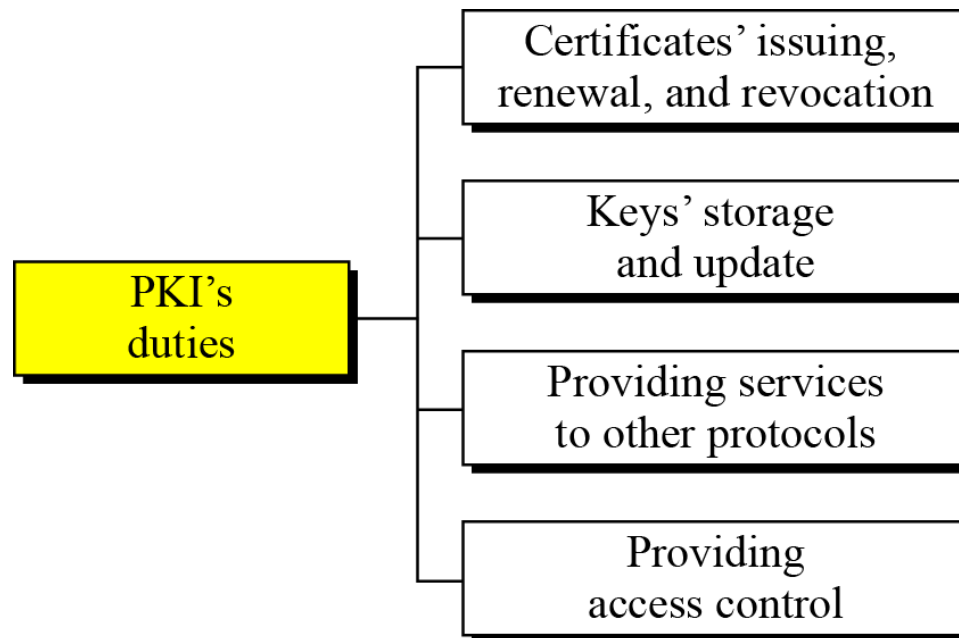
15.4.5 Continued

Figure 15.17 *Certificate revocation format*



15.4.6 Public-Key Infrastructures (PKI)

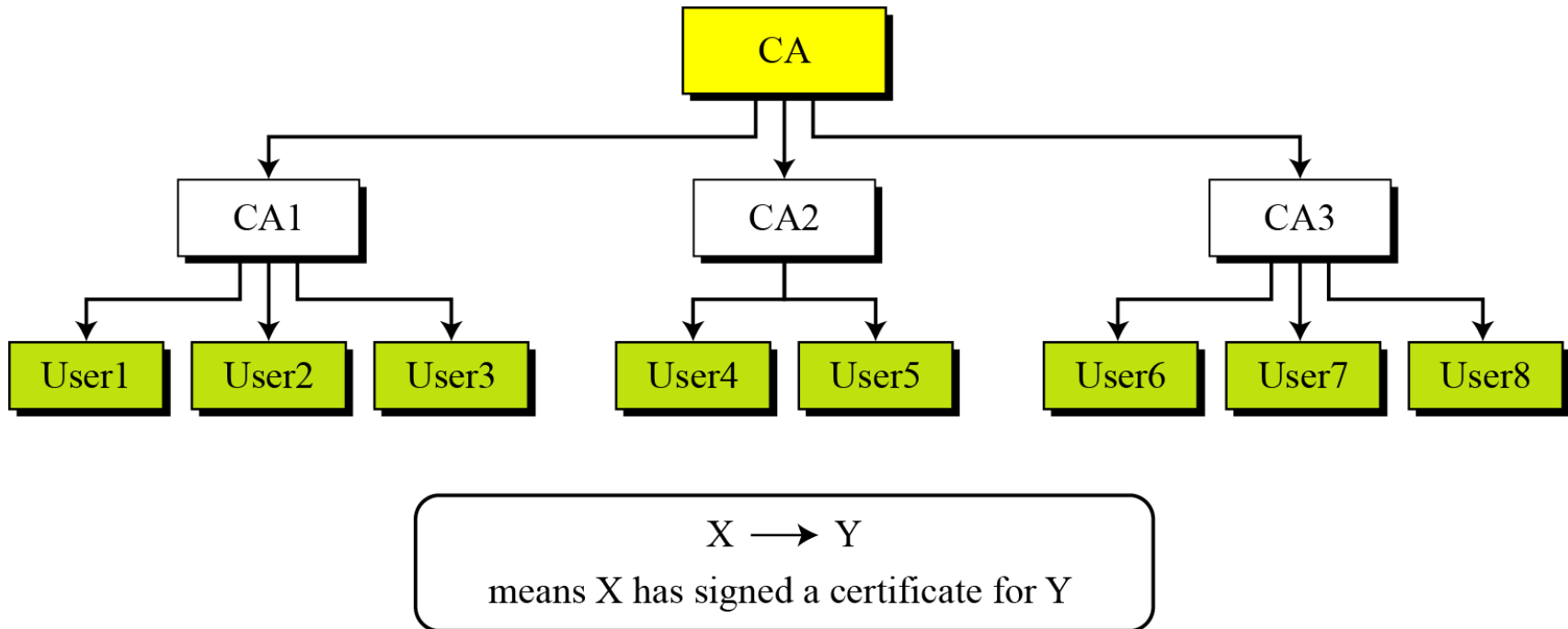
Figure 15.19 *Some duties of a PKI*



15.4.6 Continued

Trust Model

Figure 15.20 *PKI hierarchical model*



15.4.6 Continued

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, $CA\langle\langle CA1\rangle\rangle$ and $CA1\langle\langle User3\rangle\rangle$, to User1.

- a. User1 validates $CA\langle\langle CA1\rangle\rangle$ using the public key of CA.**
- b. User1 extracts the public key of CA1 from $CA\langle\langle CA1\rangle\rangle$.**
- c. User1 validates $CA1\langle\langle User3\rangle\rangle$ using the public key of CA1.**
- d. User1 extracts the public key of User 3 from $CA1\langle\langle User3\rangle\rangle$.**

15.4.6 Continued

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.