Chapter 15

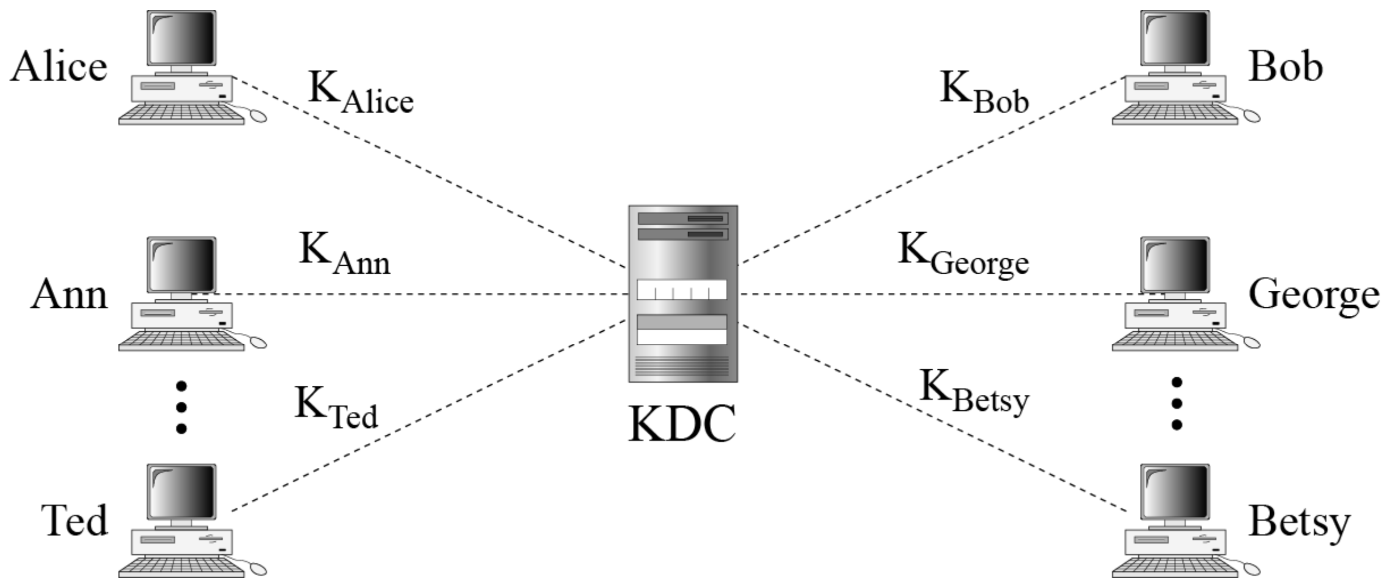
Key Management

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.***
- ***If Alice wants to exchange messages with N people, she needs N different symmetric (secret) keys. If N people need to communicate with each other, a total of N(N-1)/2 keys would be needed assuming a single key is used in both directions of communications between a pair of people. This is normally referred to as the N^2 problem.***
- ***The distribution of keys is another problem. We need an efficient and reliable (trusted) way to maintain and distribute secret keys.***

Key-Distribution Center: KDC

Each person establishes a shared key with the Key-distribution center (KDC).



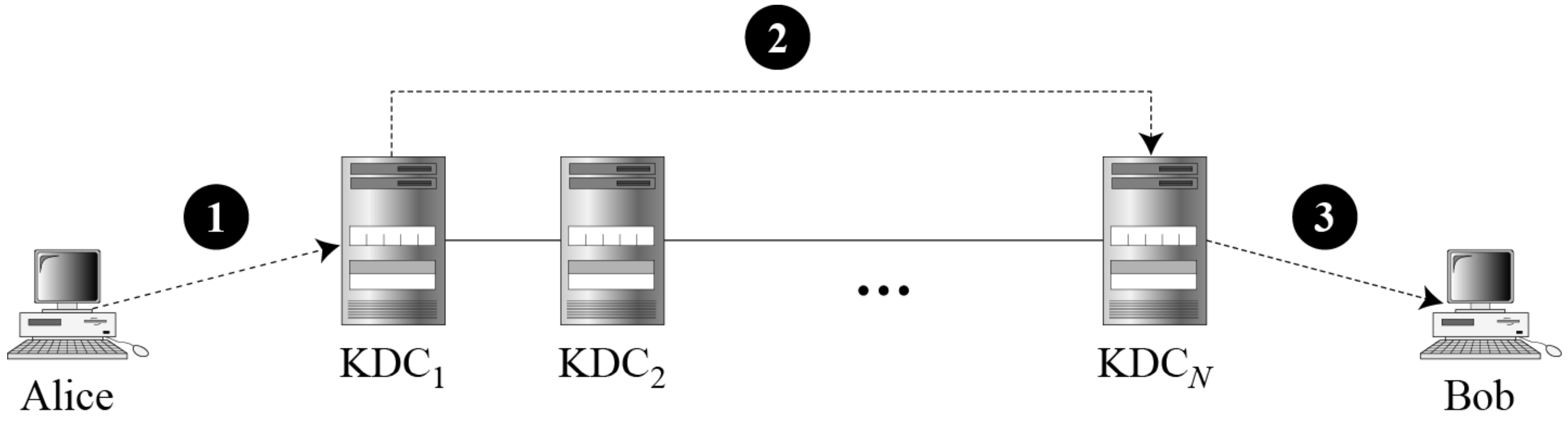
Key-Distribution Center: KDC

The procedure to get a session key between Alice and Bob is as follows

- *Alice sends a request to KDC stating that she needs a session (temporary) secret key between herself and Bob. Alice uses her secret key with the KDC to authenticate her request and herself to the KDC.*
- *The KDC informs Bob about Alice's request.*
- *If Bob agrees and authenticates himself using his secret key with the KDC, a session key is created between the two.*

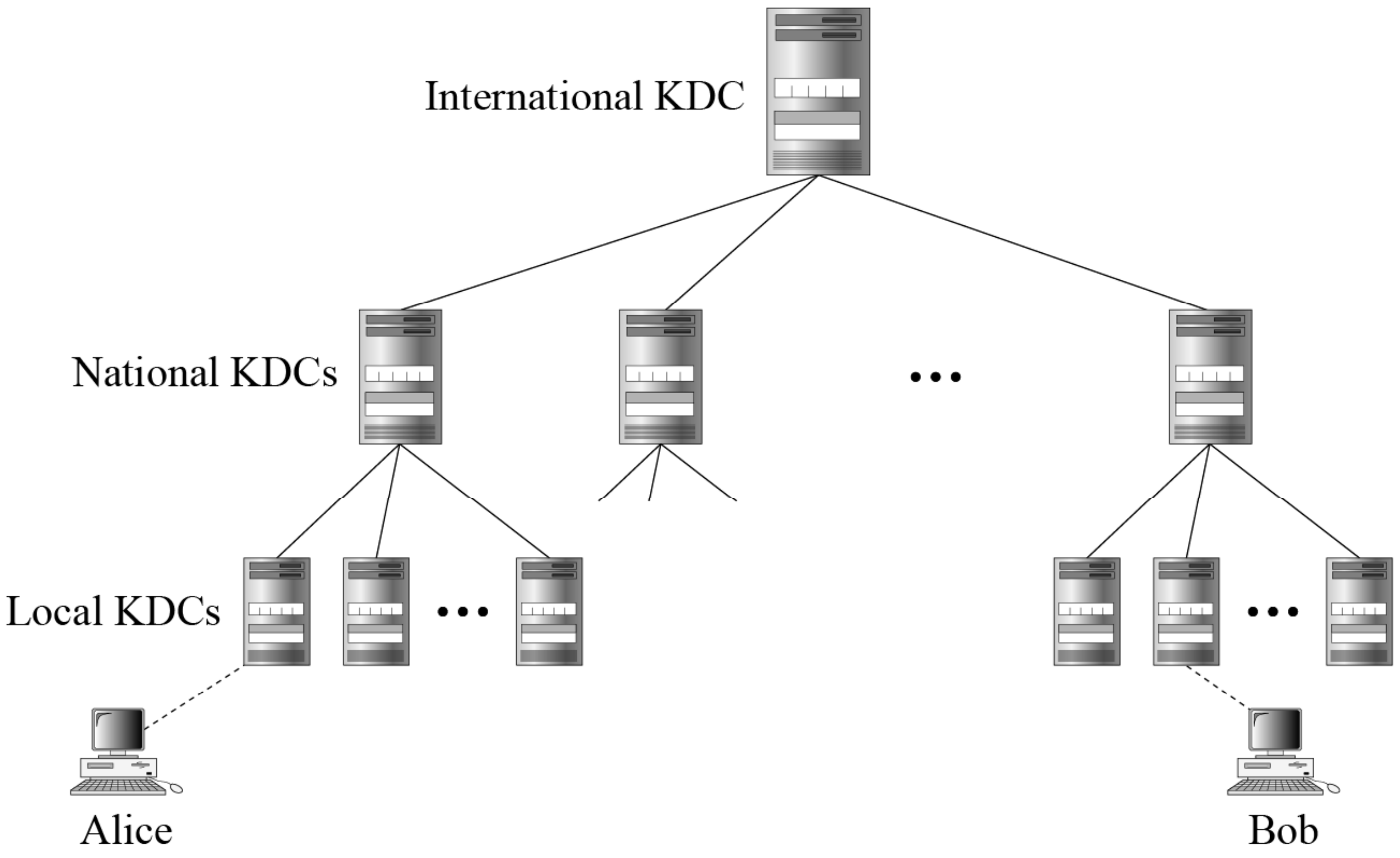
Flat Multiple KDCs.

When the number of people using a KDC increases, the system becomes unmanageable. To solve this problem, we divide the community into domains. Each domain has one KDC (or more if redundancy is desired for fault tolerance). If Alice is in one domain and Bob is in another domain, Alice contacts her KDC which in turn contacts the KDC in Bob's domain. The two KDC's can create a secret key between Alice and Bob. This system is called Flat multiple KDCs.



Hierarchical Multiple KDCs

The hierarchical multiple KDC system has one (or more) KDC at the top of the hierarchy. For example, if Alice and Bob are in two different countries. Alice sends the request to her local KDC, which relays the request to the national KDC, which forwards it to the international KDC. The request is then relayed all the way down to the local KDC where Bob lives.



Session Keys

- ***The secret key established between the KDC and a member can be used only between that member and the KDC, not between two members.***
- ***The KDC can help two members (after authenticating their secret key with the KDC) establish a temporary key that can be used by the two members for a single session. After communication is terminated, the session key becomes invalid.***

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

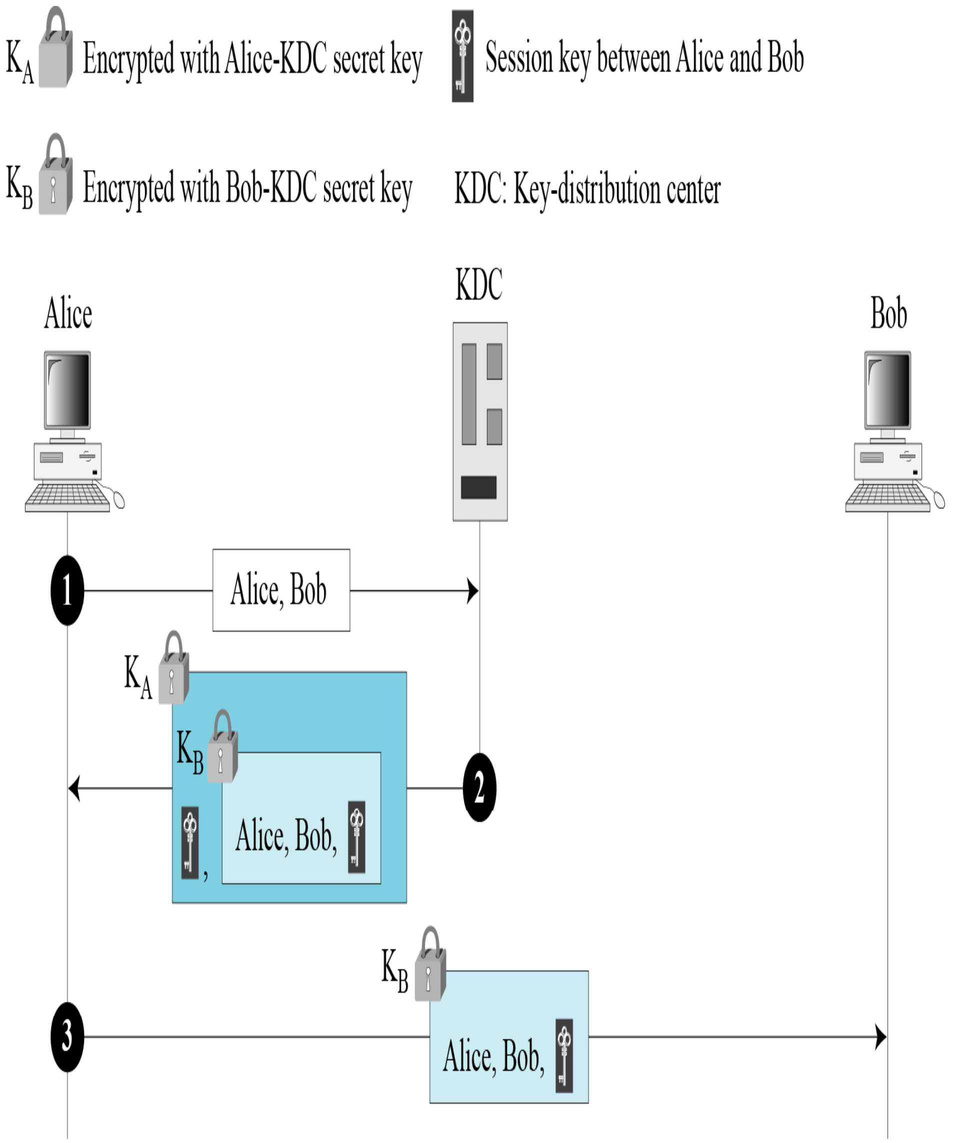
A Simple Protocol Using a KDC

1 *Alice sends a plaintext message to KDC to request a symmetric session key between herself and Bob.*

2 *The KDC creates a ticket encrypted using Bob's key K_B containing the session key. The ticket and the session key are sent to Alice in a message encrypted using Alice's key K_A. Alice decrypts the message and retrieves the session key and Bob's ticket.*

3 *Alice sends the ticket to Bob who opens (decrypts) the ticket and obtains the value of the session key.*

This simple protocol is prone to replay attacks. An adversary can save the message (ticket) in step 3 and replay it later.



K_A Encrypted with Alice-KDC secret key
Session key between Alice and Bob
K_B Encrypted with Bob-KDC secret key
KDC: Key-distribution center
Alice
KDC
Bob
1
Alice, Bob
K_A
K_B
Alice, Bob,
2
K_B
3
Alice, Bob,

Needham-Schroeder Protocol

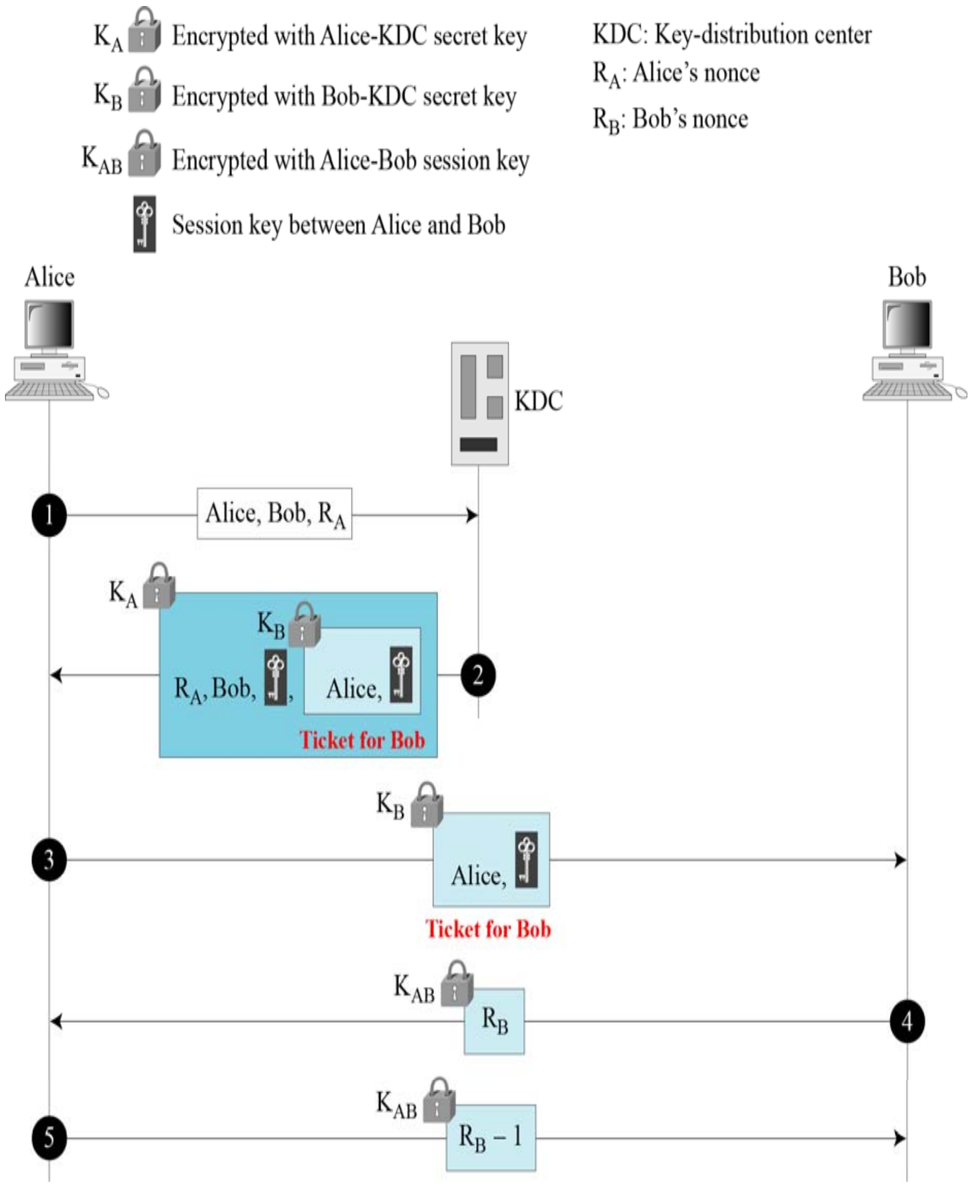
1 *Alice sends a message to KDC that includes her nonce R_A.*

2 *The KDC sends an encrypted message to Alice that includes Alice's nonce, the session key, and an encrypted ticket to B that includes the session key. The ticket is encrypted using Bob's key and the whole message is encrypted using Alice's key.*

3 *Alice sends the ticket to Bob.*

4 *Bob decrypts the ticket and sends his challenge R_B to Alice encrypted with the session key.*

5 *Alice responds by sending to Bob the encrypted value R_B-1 (rather than R_B to prevent replay attacks).*



K_A Encrypted with Alice-KDC secret key
K_B Encrypted with Bob-KDC secret key
K_{AB} Encrypted with Alice-Bob session key
Session key between Alice and Bob
KDC: Key-distribution center
R_A: Alice's nonce
R_B: Bob's nonce
Alice
Bob
KDC
1
Alice, Bob, R_A
K_A
K_B
R_A, Bob,
Alice,
2
Ticket for Bob
K_B
3
Alice,
Ticket for Bob
K_{AB}
R_B
4
K_{AB}
$R_B - 1$
5

Otway-Rees Protocol

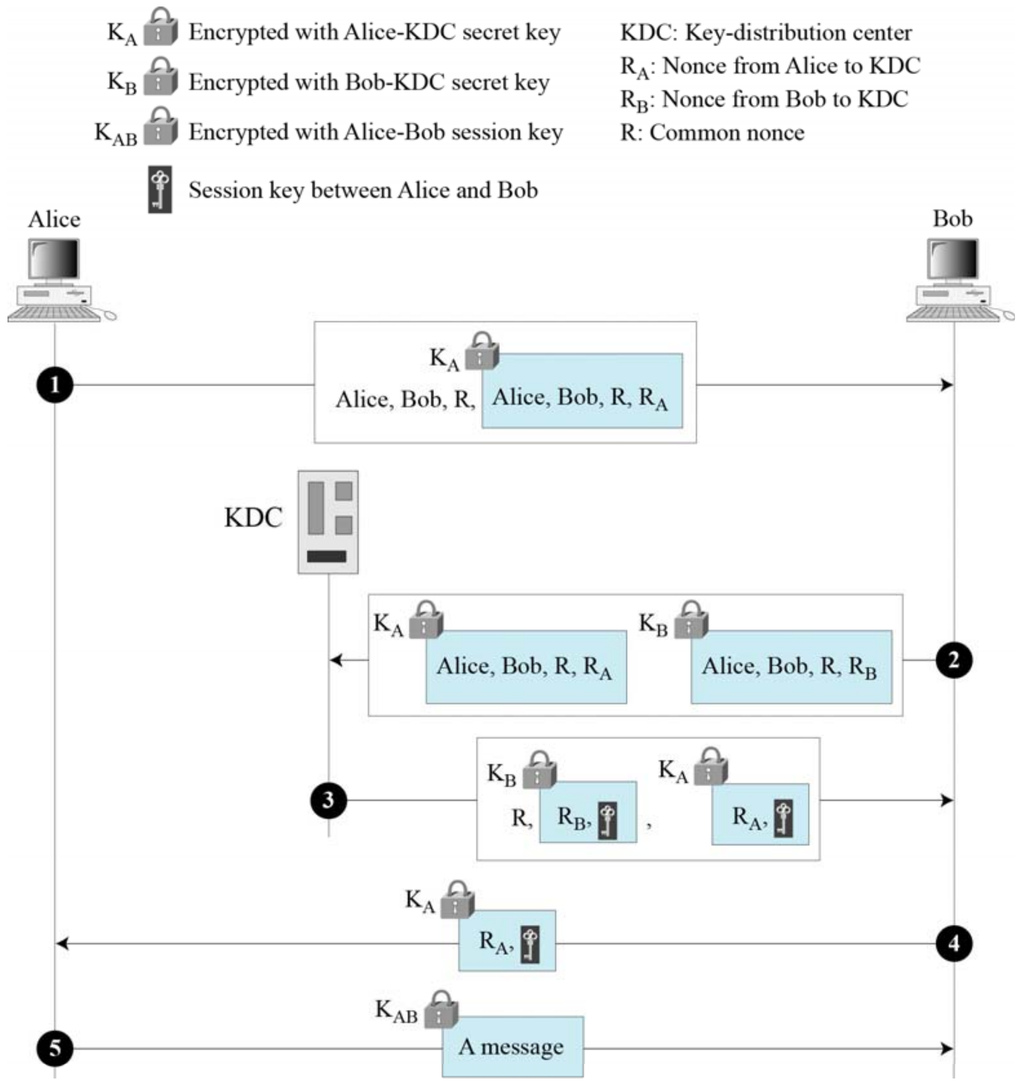
1 *Alice sends a message to Bob that includes a common nonce R and her challenge R_A and a ticket to the KDC containing both R and R_A. The ticket is encrypted with Alice's secret key.*

2 *Bob creates a similar ticket but with his own nonce R_B. Bob sends both tickets to KDC.*

3 *The KDC creates a message that contains R, a ticket for Alice with nonce R_A and a ticket for Bob with nonce R_B. The tickets contain the session key. The KDC sends the message to Bob.*

4 *Bob sends Alice her ticket.*

5 *Alice sends a short (hello) message encrypted with the session key to Bob.*



K_A Encrypted with Alice-KDC secret key
K_B Encrypted with Bob-KDC secret key
K_{AB} Encrypted with Alice-Bob session key
Session key between Alice and Bob
KDC: Key-distribution center
R_A: Nonce from Alice to KDC
R_B: Nonce from Bob to KDC
R: Common nonce
Alice
Bob
KDC
1
K_A
Alice, Bob, R, Alice, Bob, R, R_A
2
K_A Alice, Bob, R, R_A
K_B Alice, Bob, R, R_B
3
K_B R, R_B,
K_A R_A,
4
K_A R_A,
5
K_{AB} A message

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.

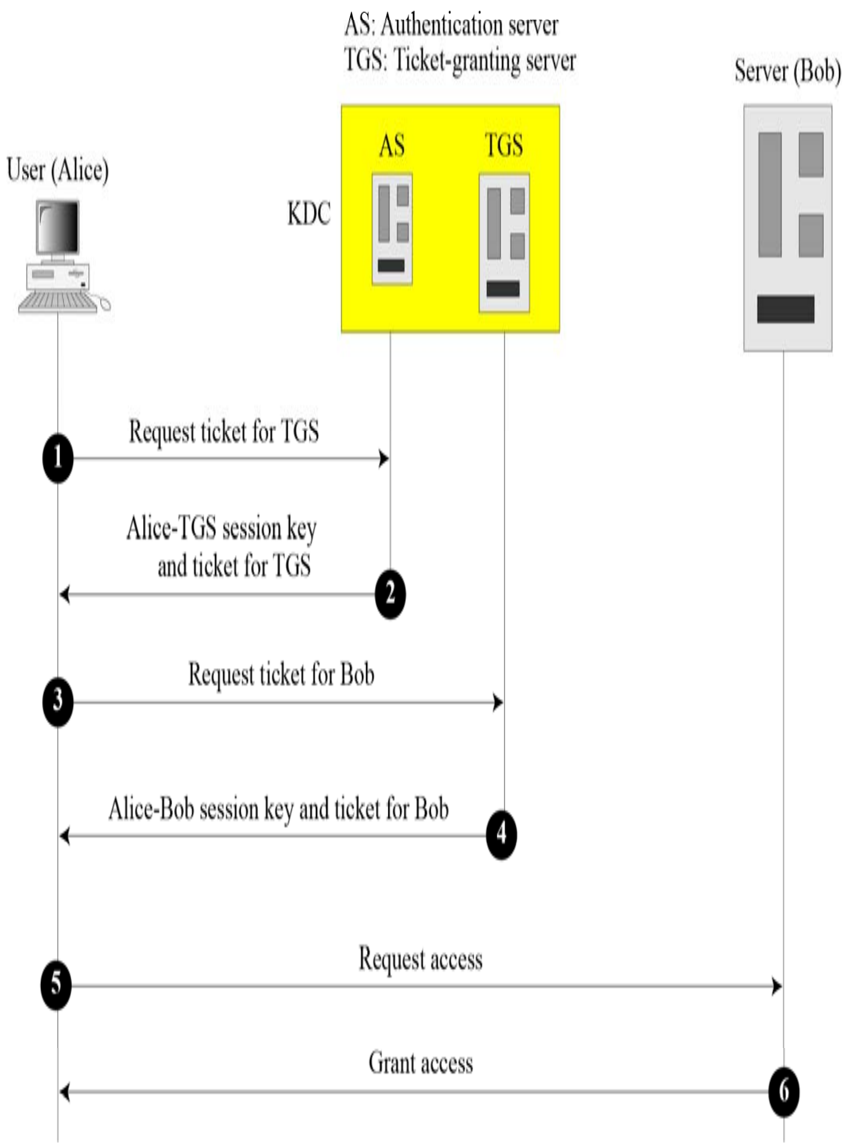
Kerberos has separated user verification from the process of issuing tickets that allow the user to access different servers. Kerberos is designed to support client-server applications, such as FTP, in which the client process at the user site communicates with the server process at the server site.

Kerberos Servers

Authentication Server (AS): ***is the KDC in the Kerberos protocol. Each user registers with AS and is granted a user ID and password.***

Ticket-Granting Server (TGS) ***issues a ticket for the real server (Bob). It also provides the session key (K_{AB}) between the user and the real server.***

Real Server ***(Bob) provides services for the user (Alice).***


AS: Authentication server
TGS: Ticket-granting server
Server (Bob)
User (Alice)
KDC
AS
TGS
Request ticket for TGS
1
Alice-TGS session key and ticket for TGS
2
Request ticket for Bob
3
Alice-Bob session key and ticket for Bob
4
Request access
5
Grant access
6

Kerberos Operation

Diagram is on next slide

The client process (Alice) can access the real server process (Bob) in six steps

1. ***Alice sends her request to AS in plaintext.***

2. ***The AS sends a message encrypted with Alice's key $K_{A\text{-}AS}$. The message contains a session key $K_{A\text{-}TGS}$ that will be used by Alice to contact TGS and a ticket for TGS encrypted using TGS's key $K_{AS\text{-}TGS}$. When the message arrives, Alice types her password which is used by the client process to create $K_{A\text{-}AS}$, then decrypt the message to extract the session key and the ticket.***

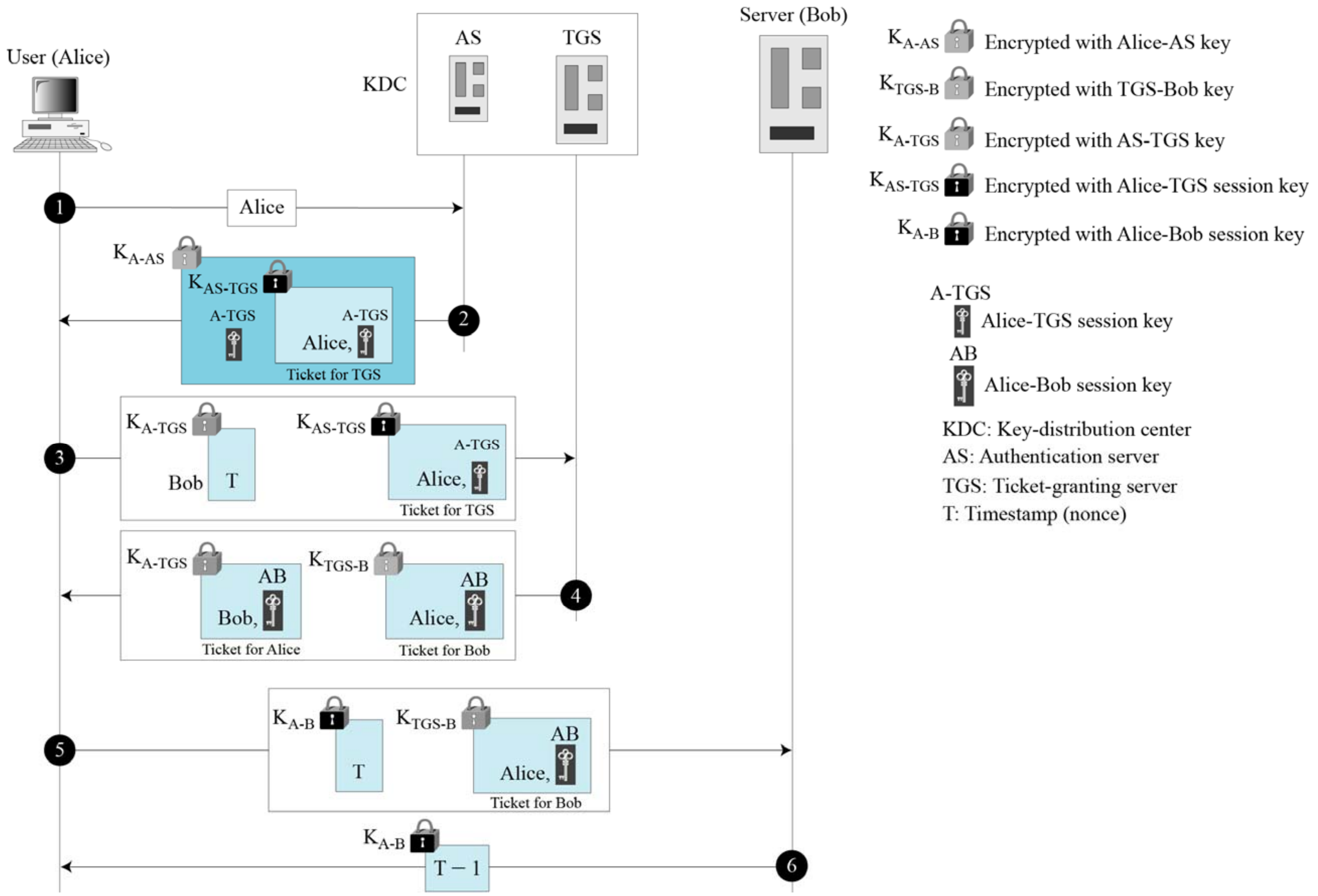
3. ***Alice sends three items to TGS: the ticket from AS, the name of the real server (Bob), and a timestamp encrypted with $K_{A\text{-}TGS}$.***

4. ***TGS sends to Alice two tickets both containing the session key $K_{A\text{-}B}$ between Alice and Bob. Alice's ticket is encrypted with the session key $K_{A\text{-}TGS}$ and Bob's ticket is encrypted with Bob's password/key K_{TGS_B}.***

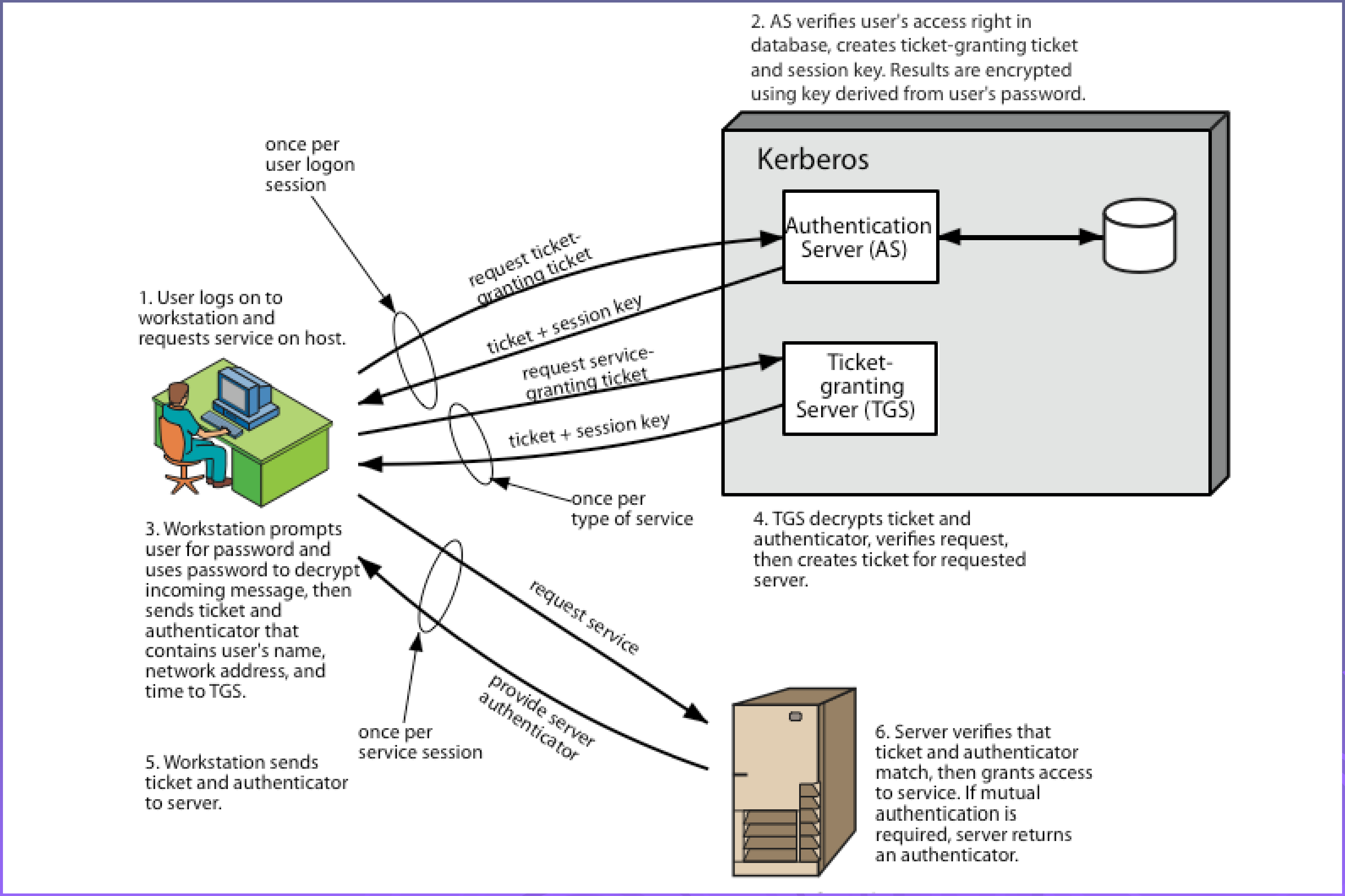
5. ***Alice sends Bob's ticket with the timestamp encrypted with $K_{A\text{-}B}$.***

6. ***Bob responds by subtracting 1 from the timestamp and encrypts the response with $K_{A\text{-}B}$.***

Kerberos Operation



Kerberos 4 Overview



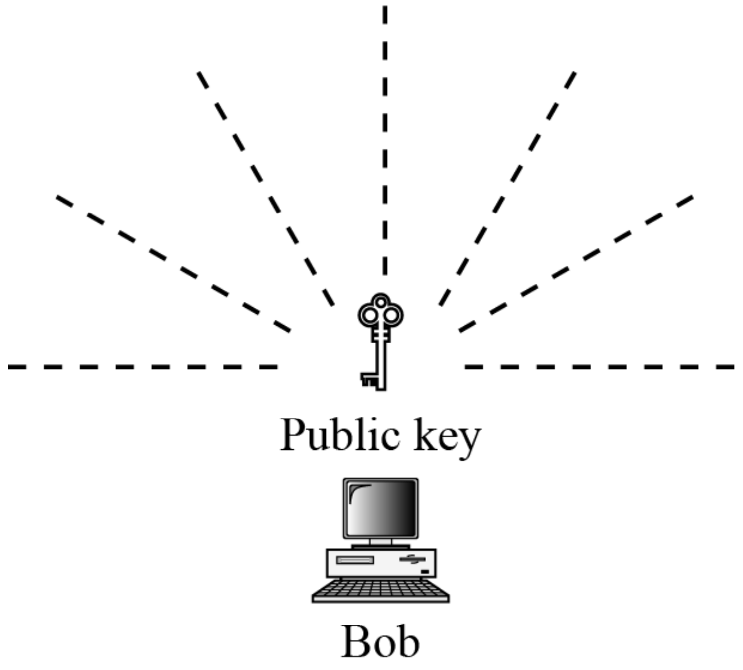
2. AS verifies user's access right in database, creates ticket-granting ticket and session key. Results are encrypted using key derived from user's password.
Kerberos
Authentication Server (AS)
Ticket-granting Server (TGS)
once per user logon session
request ticket-granting ticket
ticket + session key
request service-granting ticket
ticket + session key
once per type of service
1. User logs on to workstation and requests service on host.
3. Workstation prompts user for password and uses password to decrypt incoming message, then sends ticket and authenticator that contains user's name, network address, and time to TGS.
4. TGS decrypts ticket and authenticator, verifies request, then creates ticket for requested server.
request service
provide server authenticator
once per service session
5. Workstation sends ticket and authenticator to server.
6. Server verifies that ticket and authenticator match, then grants access to service. If mutual authentication is required, server returns an authenticator.

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.

Public Announcement

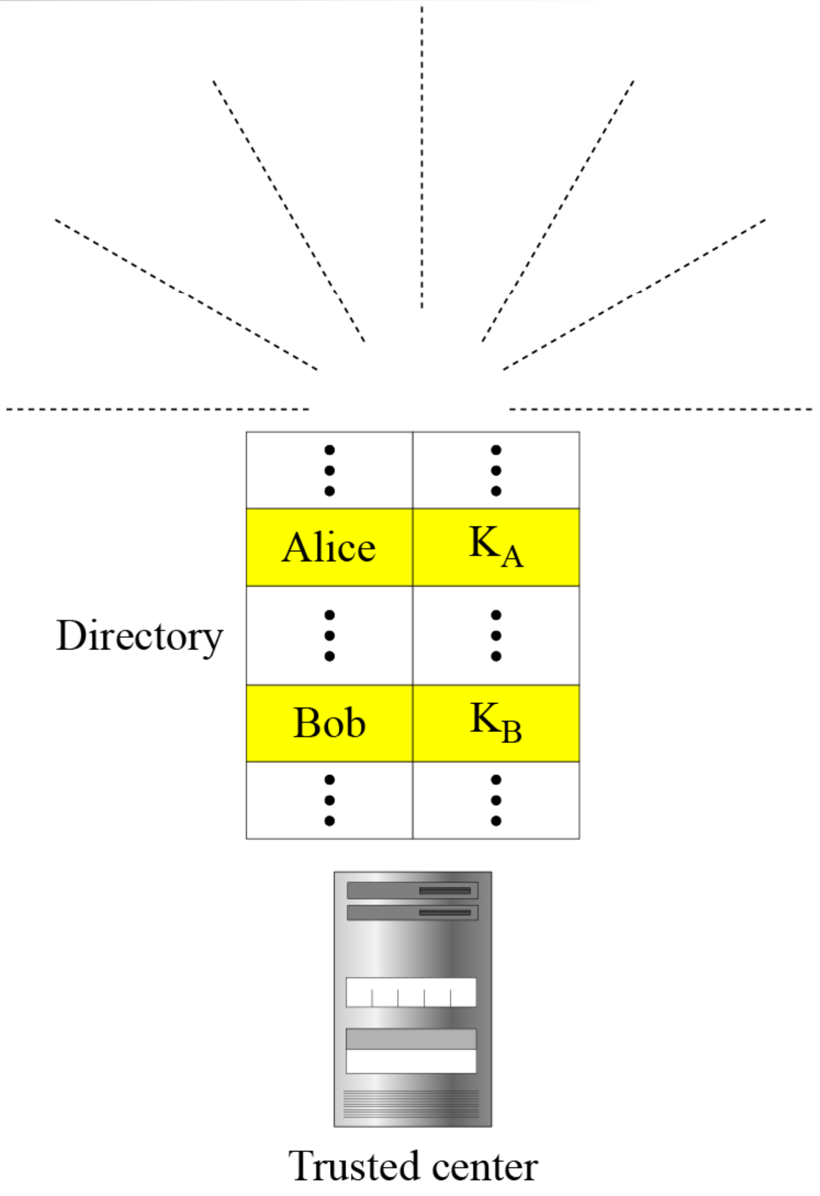
Bob makes his public key available on his web site. Alice can get Bob's public key by accessing Bob's site or sending email to him. This method is simple but is not secure and is subject to forgery.



Trusted Center

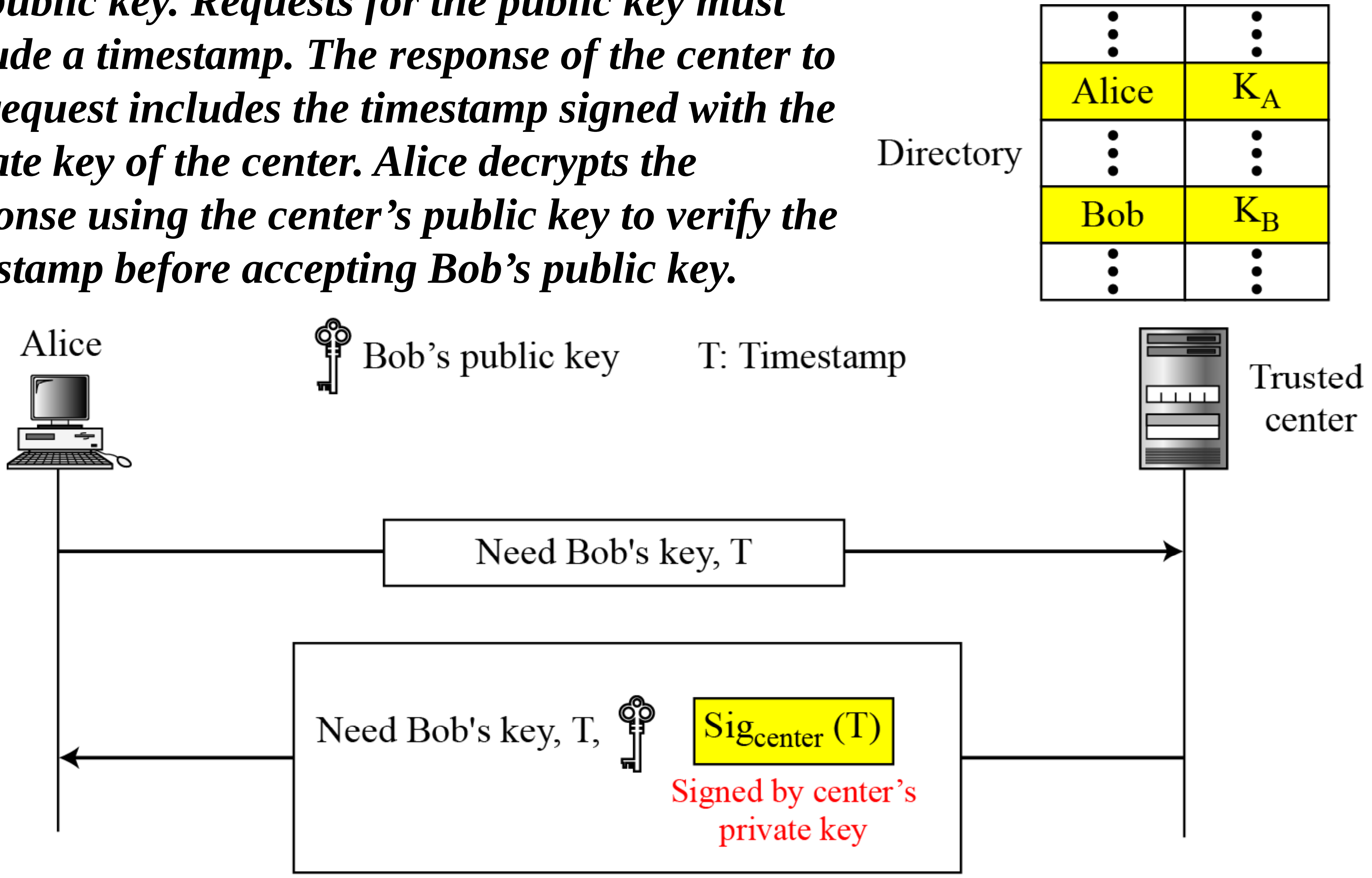
The trusted center retains and updates a directory of public keys. Each user must register with the trusted center and establish a user ID and password. The user can then deliver his/her public key for insertion into the directory.

The center can publicly advertise the directory and respond to inquiries about public keys.



Controlled Trusted Center

A controlled trusted center achieves higher level of security by adding control on the distribution of the public key. Requests for the public key must include a timestamp. The response of the center to the request includes the timestamp signed with the private key of the center. Alice decrypts the response using the center's public key to verify the timestamp before accepting Bob's public key.



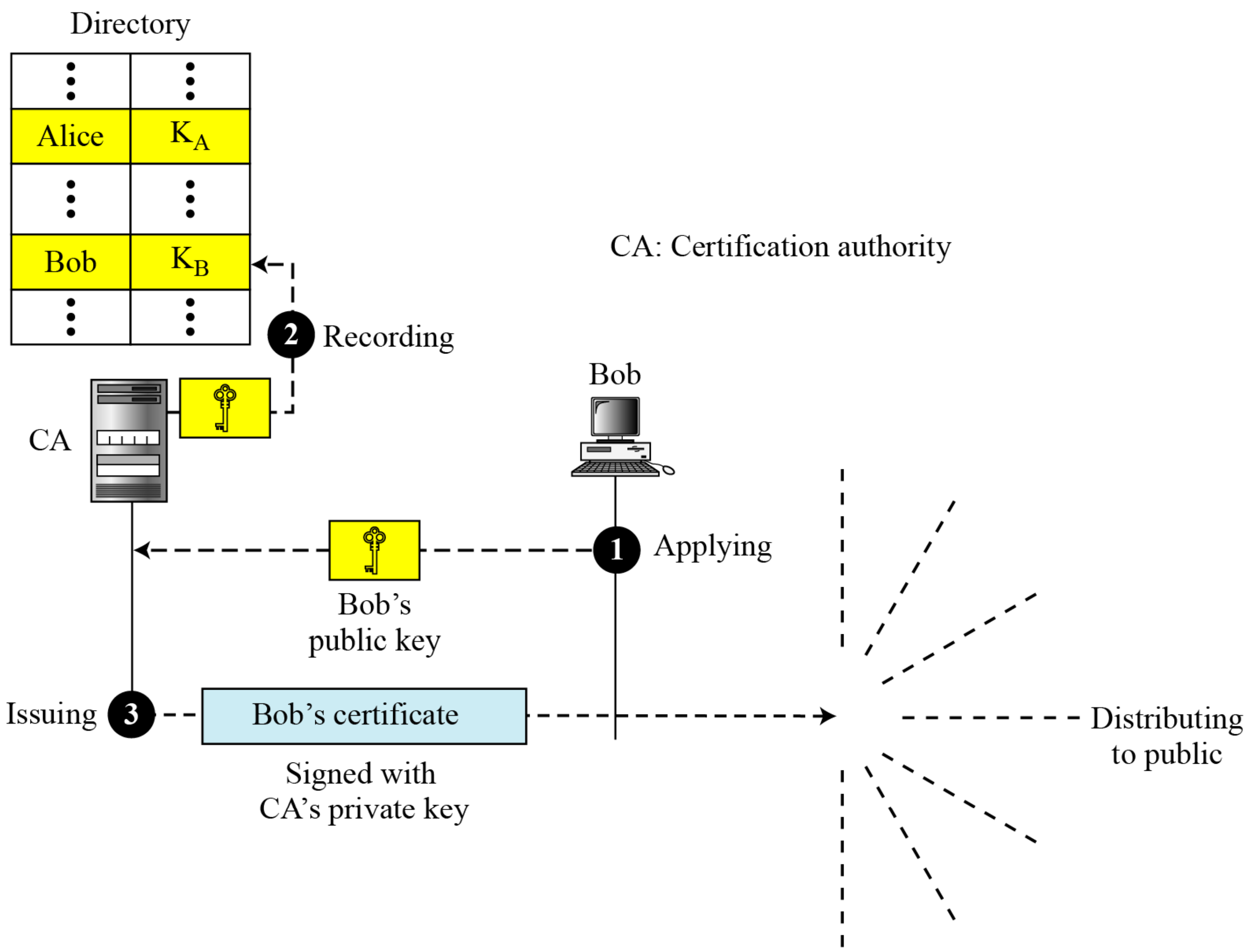
Certification Authority

Security certificates are used to reduce the load on trusted centers.

- ***A server (Bob) can request a certificate from a certification authority (CA), which could be a cross-certified* company or state or federal organization. Bob's request contains his identification and his public key.***
- ***The CA checks the identification of Bob. If verified, the CA writes Bob's public key on the certificate and signs it with its own private key.***
- ***Bob can now upload the signed certificate and store it on his site or Bob may send the certificate to users upon request.***
- ***Any user who wants Bob's public key can download the certificate and decrypts it using the CA's public key to extract Bob's public key.***

*** Cross-certification will be explained at the end of this chapter**

Certification Authority



X.509

The Internet community has accepted the ITU-T* recommendation X.509 as a way to unify certificate formats. In X.509, the certificate has the following important fields:

***Version number:** this field is the version of X.509 (current version is 3).*

***Serial number:** this field is the serial number assigned to each certificate and is unique for each certificate issuer.*

***Signature algorithm ID:** this field identifies the signature algorithm used in the certificate. This field is repeated in the signature field.*

***Issuer name:** this field identifies the CA that issued the certificate.*

***Validity Period:** this field defines the earliest (not before) time and the latest (not after) time during which the certificate is valid.*

***Subject name:** this field defines the entity that owns the public key stored in this certificate.*

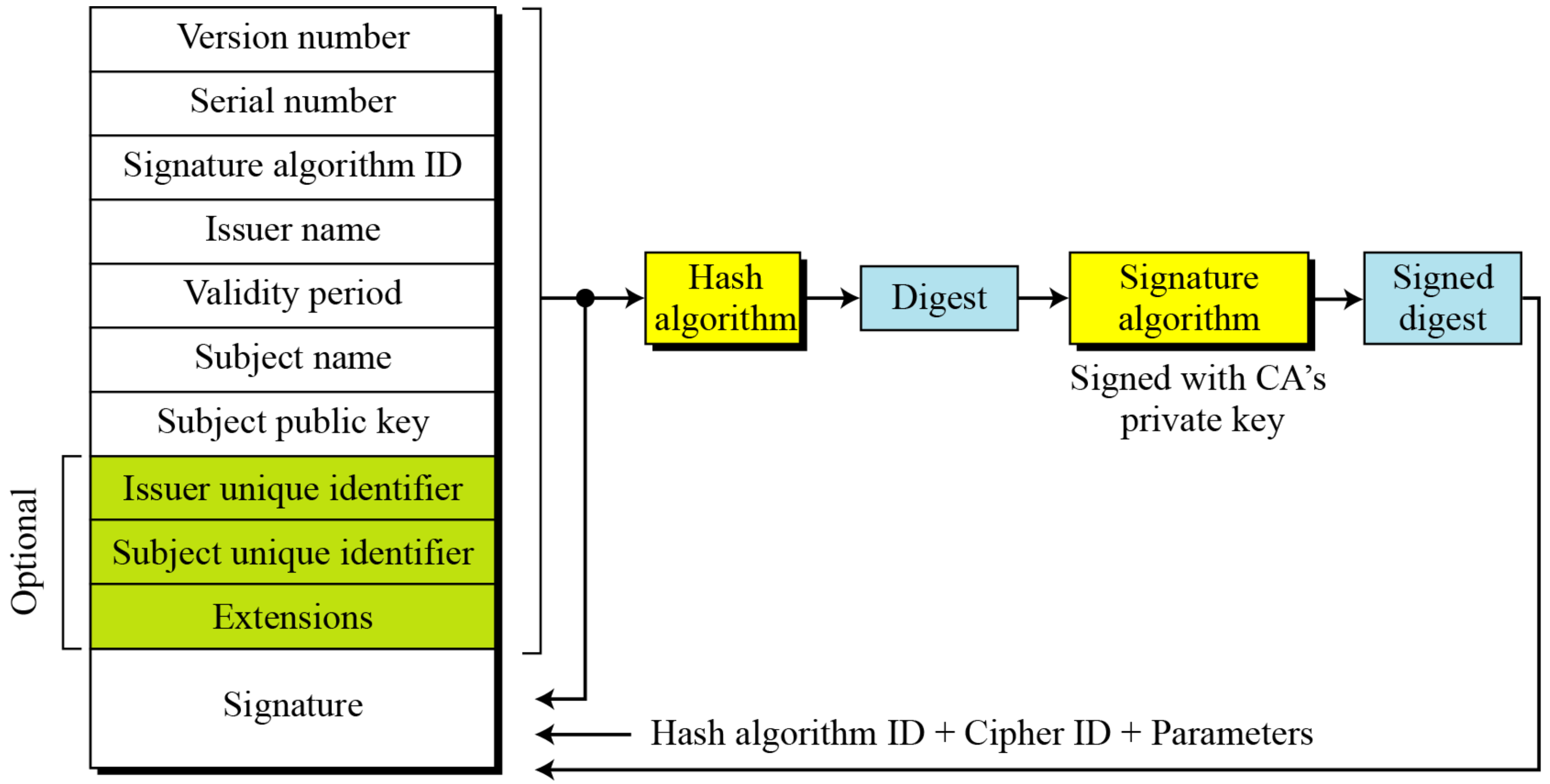
***Subject public key:** this field gives the value of the public key of the owner of the certificate and defines the public key algorithm.*

***Signature:** this field contains the digest of all other fields in the certificate encrypted by the CA's private key, and also contains the ID of the signature algorithm.*

* ITU-T = International Telecommunication Union- Telecommunication Standardization Sector

X.509

In X.509, the certificate has the following fields:



The optional Issuer or Subject unique identifier allows two issuers or two subjects to have the same value in the Issuer or Subject name field, provided their unique identifiers are different.

X.509

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 Revocation

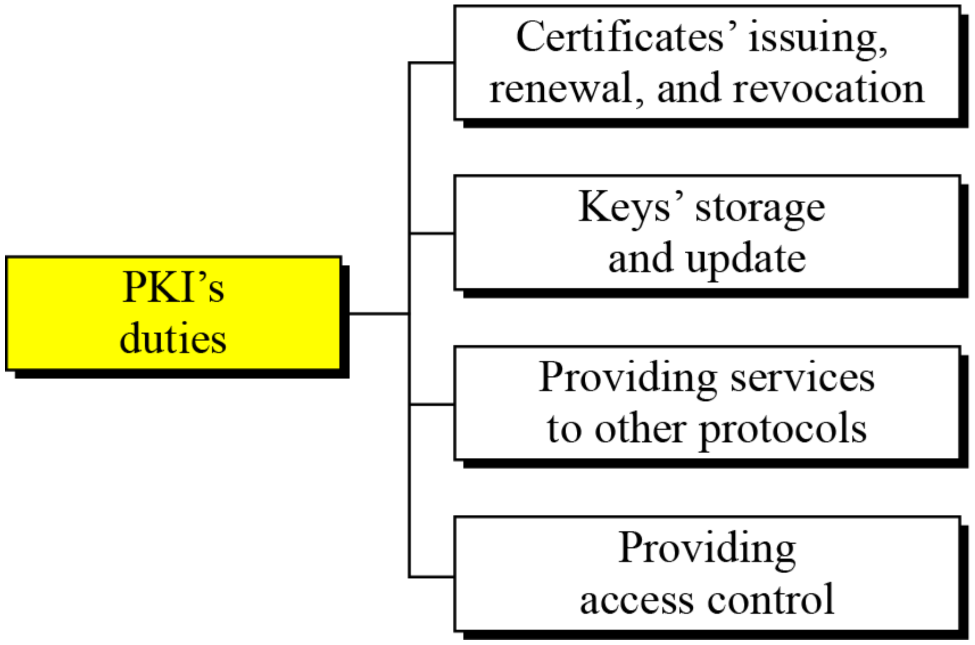
In some cases a certificate must be revoked before its expiration (e.g., the private key of the subject or of the CA has been compromised). The revocation is done by periodically issuing a certificate revocation list (CRL) that contains all revoked certificates that have not expired on the date the CRL is issued. To ensure the validity of a certificate, the user must check the latest CRL published by the CA that issued the certificate.

Public-Key Infrastructure (PKI)

PKI is a model for creating, distributing and revoking certificates based on the X.509. IETF (Internet Engineering Task Force) has created the public-key infrastructure X.509 (PKIX).

Some duties of PKI

- Issue, renew and revoke certificates.
- Store and update private keys for members who wish to hold their private keys at a safe place.
- Provide services to other Internet security protocols that need public key info such as IPSec and TLS.
- Provide access control, i.e., provide different levels of access to the information stored in its database.


PKI's duties
Certificates' issuing, renewal, and revocation
Keys' storage and update
Providing services to other protocols
Providing access control

Public-Key Infrastructures (PKI)

PKI Trust Model

For scalability, there should be many certification authorities in the world; each CA handles a specified number of certificates. The PKI trust model defines rules that specify how a user can verify a certificate received from a CA.

As an example, the PKI hierarchical trust model defines hierarchical rules that specify how a user can verify a certificate received from a CA.

PKI uses the following notation to denote the certificate issued and signed by certification authority X for entity Y

$$X \ll Y \gg$$

Public-Key Infrastructures (PKI)

Example 15.3

User1 knows only the public key of the root CA. Show how can User1 obtain a verified copy of User3's public key.

Solution

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

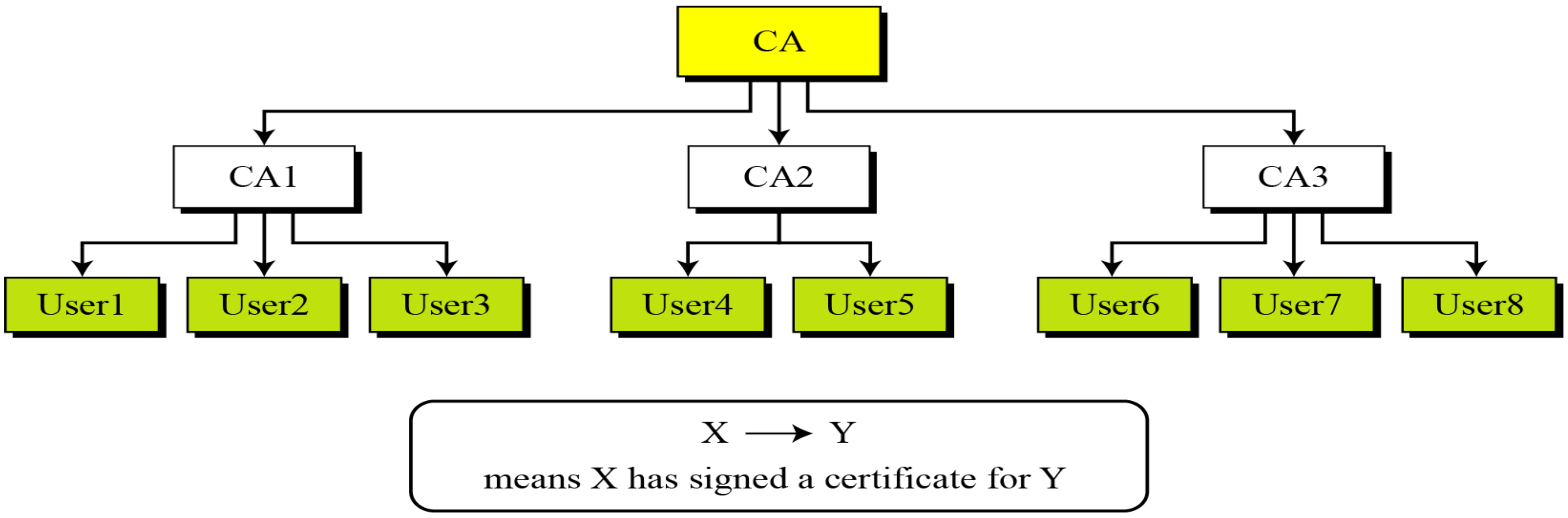
a. User1 validates CA<<CA1>> using the public key of CA.

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>>.

Users1 has used the following chain **CA<<CA1>> CA1<<User3>>**



CA Hierarchy-Certificate Validation Path (Chain)

A only knows the public key of X and B only knows the public key of Z.
A acquires B's certificate using the chain: X<<W>> W<<V>> V<<Y>> Y<<Z>> Z<<B>>
B acquires A's certificate using the chain: Z<<Y>> Y<<V>> V<<W>> W<<X>> X<<A>>

Because X signed a certificate for the public key of Z, a shorter chain for A to acquire B's certificate is as follows:
X<<Z>> Z<<B>>

Because Z signed a certificate for the public key of X, a shorter chain for B to acquire A's certificate is as follows:
Z<<X>> X<<A>>

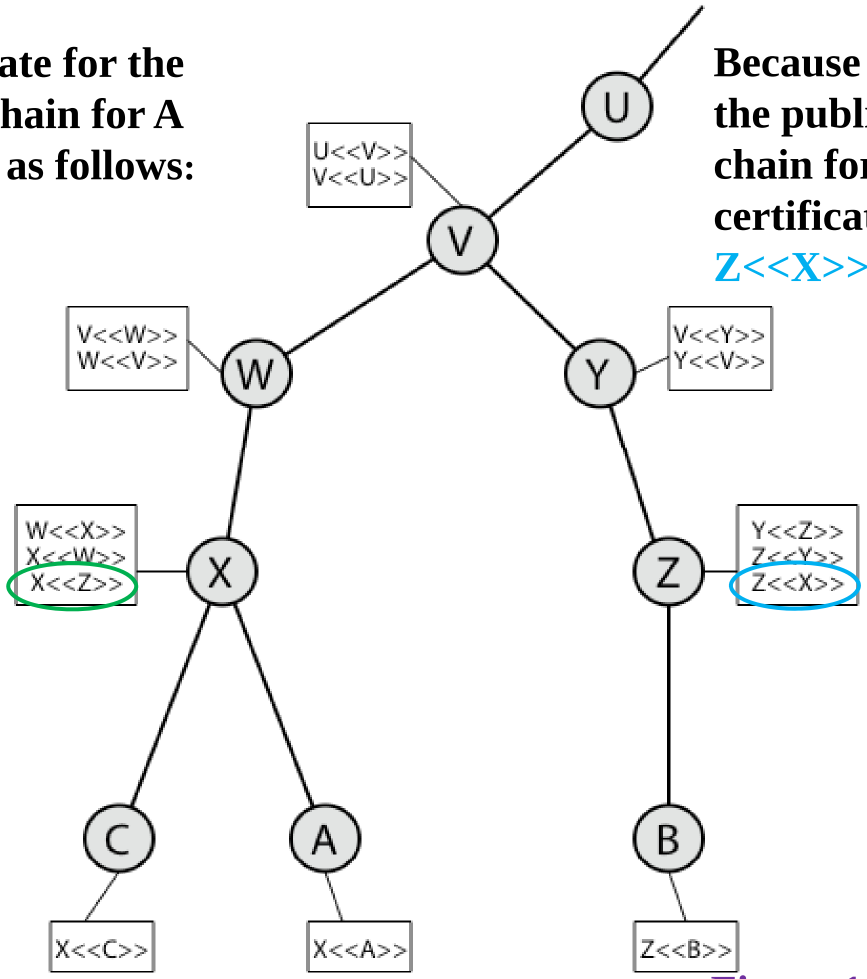


Figure 14.5 (Stallings Book)

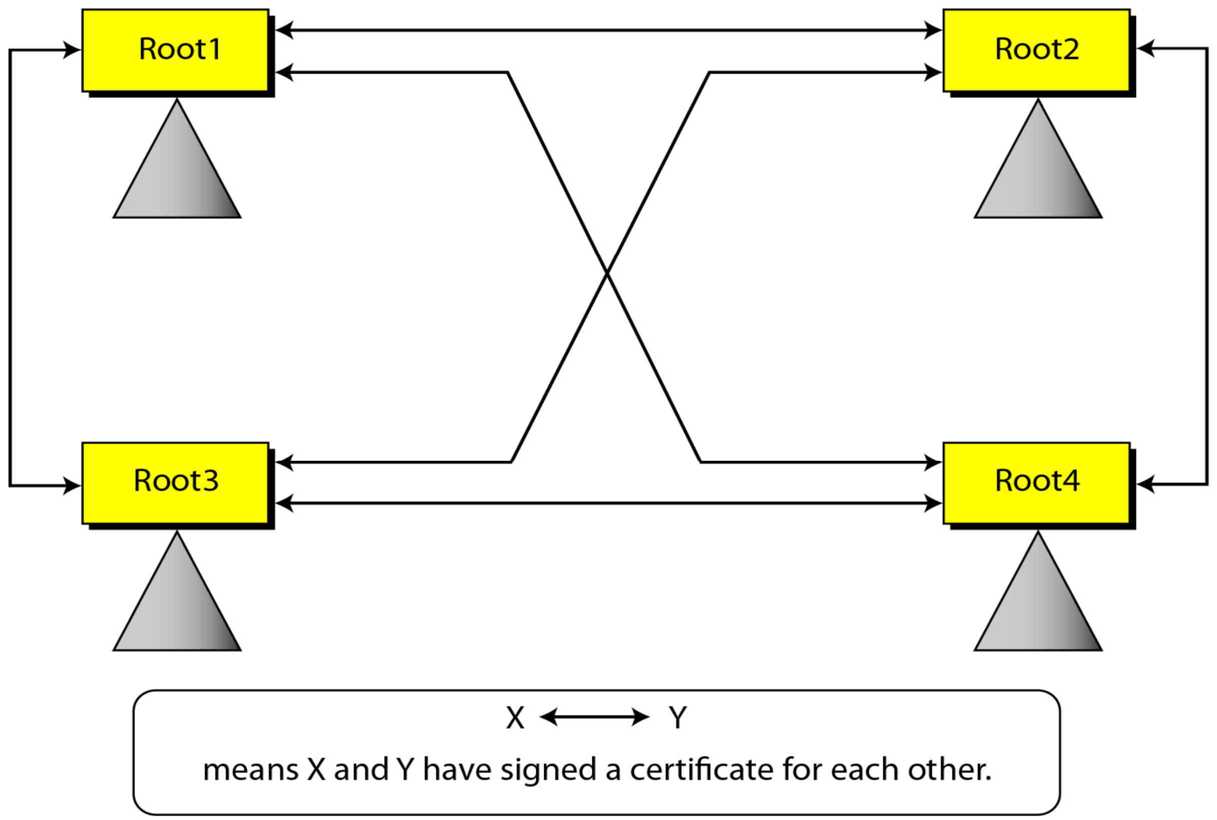
Public-Key Infrastructures (PKI)

Example 15.4

Some Web browsers, such 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 these roots and view the certificate.

The Mesh Model

The hierarchical model with a single root is not suitable for a very large community. The mesh model, that connects several roots together, is more useful. Roots are connected by the mesh but each root has its own hierarchical structure. Certificates in the mesh are cross-certificates, i.e., each root certifies each other root. For a fully connected mesh with four roots, we need 4*3 = 12 certificates. Each double-arrow represents 2 certificates.

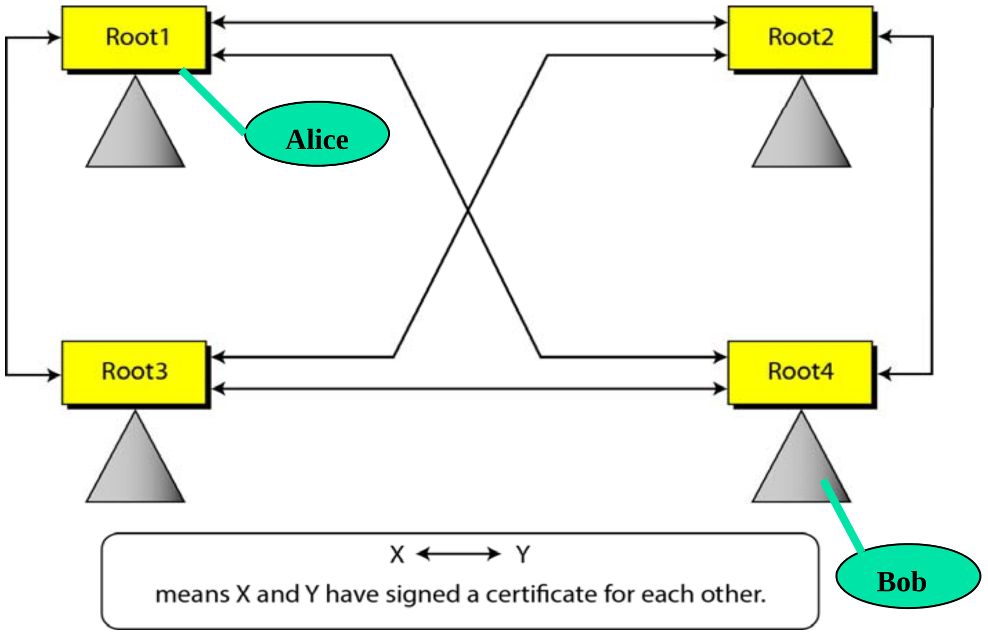


Example 15.5

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

Solution

Alice looks at the directory of Root1 to find Root1<< Root4>>. Using the public key of Root4, Alice can follow the chain of certificates from Root4 to Bob. Alice can then extract and verify Bob’s public key.



The Federal Public Key Infrastructure (FPKI) Authority

http://www.idmanagement.gov/fpkipa/ *An Official Website of the United States Government*

The Federal Public Key Infrastructure (FPKI) Policy Authority is an interagency body set up to enforce digital certificate standards for trusted identity authentication across the federal agencies and between federal agencies and outside bodies, such as universities, state and local governments and commercial entities.

Why A U.S. Federal PKI?

- Statutory mandates for e-government and implementing electronic signature technology
- Demands for improved services at lower cost
- International Competition
- International Collaboration

The U.S. Federal Bridge Certification Authority (FBCA)

Within FPKI, the **FBCA** performs the following functions:

- Create trust paths among individual Agency PKIs
- Develops cross-certificates for Participating Commercial CA's
- Propagate policy information to certificate users in different Agencies
- Support the Agency use of approved cryptographic algorithms

Some Certified PKI Shared Service Providers (SSP)

- VeriSign, Inc. (A Symantec Company)
- Verizon / Cybertrust
- Operational Research Consultants, Inc.
- Department of the Treasury
- Entrust Managed Services

Some Entities that are cross-certified with the FBCA

Department of the Treasury, Department of State, State of Illinois, ACES/IdenTrust, DoD External CA, US Patent & Trademark Office, Government Printing Office, CertiPath Bridge, VeriSign, Verizon Business, Entrust, Inc.

Examples of Agency, State and University PKIs:

NASA PKI
http://nasaca.nasa.gov/

State of Illinois PKI
http://www2.illinois.gov/pki/Pages/default.aspx

University of Wisconsin PKI
http://www.educause.edu/Resources/PKIImplementationattheUniversi/158806

Commercial Cross-Certified Company

VeriSign, Inc

http://www.verisign.com/

VeriSign has been acquired by Symantec.

VeriSign offers SSL certificates**, VeriSign Trusted Seals, and other solutions for web site security.

VeriSign Trust™ Seal is intended to assure customers they can trust the link, trust the site, and trust the transaction.

Symantec Managed PKI Service issues and manages X.509 certificates that interoperate with a wide variety of operating systems, devices, VPN, mail, and web browser software.

Symantec Digital IDs for Secure Email allow users to digitally sign and encrypt their digital communications using a certificate.

***** SSL certificate is an X.509 certificate; SSL will be covered later in CNT 4403***

InCommon Certificate Service for U.S. Higher Education

- The InCommon Certificate Service was launched in August 2010.
- InCommon, operated by Internet2, provides trust services for the U.S. education and research communities.   It includes an identity management Federation Service and a Certificate Service.
- The InCommon Certificate Service provides the U.S. higher education community with unlimited SSL, client and code signing digital certificates at one fixed annual fee. An institution can acquire unlimited certificates for all of its domains, including those hosted for professional societies, athletics, or other campus-based organizations.
- The InCommon Certificate Service provides a cost-effective means of deploying and operating a more secure and authenticated online environment. In addition to the already low fee, Internet2 members receive a 25 percent discount. Details are available at

  www.incommon.org/cert
- InCommon is offering this service through a partnership with Comodo CA Ltd., a major certification authority and Internet2 industry member.