Key Distribution and User Authentication

SYMMETRIC KEY DISTRIBUTION USING SYMMETRIC ENCRYPTION

- For symmetric encryption to work, the two parties to an exchange must share the same key, and that key must be protected from access by others
- Frequent key changes are usually desirable to limit the amount of data compromised if an attacker learns the key
- Key distribution technique
  - The means of delivering a key to two parties that wish to exchange data, without allowing others to see the key

KEY DISTRIBUTION

- For two parties A and B, there are the following options:

1. A key can be selected by A and physically delivered to B
2. A third party can select the key and physically deliver it to A and B
3. If A and B have previously and recently used a key, one party could transmit the new key to the other, using the old key to encrypt the new key
4. If A and B each have an encrypted connection to a third party C, C could deliver a key on the encrypted links to A and B

THREATS

- The problem that Kerberos addresses is this:
  - Assume an open distributed environment in which users at workstations wish to access services on servers distributed throughout the network.
  - We would like for servers to be able to restrict access to authorized users and to be able to authenticate requests for service.

- In this environment, the following three threats exist:

1. A user may gain access to a particular workstation and pretend to be another user operating from that workstation.

2. A user may alter the network address of a workstation so that the requests sent from the altered workstation appear to come from the impersonated workstation.

3. A user may eavesdrop on exchanges and use a replay attack to gain entrance to a server or to disrupt operations.

KERBEROS

- Key distribution and user authentication service developed at MIT
- Provides a centralized authentication server whose function is to authenticate users to servers and servers to users
- Relies exclusively on symmetric encryption, making no use of public-key encryption

Two versions are in use

- Version 4 implementations still exist, although this version is being phased out
- Version 5 corrects some of the security deficiencies of version 4 and has been issued as a proposed Internet Standard (RFC 4120)

KERBEROS VERSION 4

- A basic third-party authentication scheme
- Authentication Server (AS)
  - Users initially negotiate with AS to identify self
    - Knows the passwords of all users and stores these in a centralized database
    - These keys have been distributed physically or in some other secure manner.
  - AS provides a non-corruptible authentication credential (ticket granting ticket TGT)
- Ticket Granting Server (TGS)
  - Users subsequently request access to other services from TGS on basis of users TGT
- Complex protocol using DES

TABLE 4.1
SUMMARY OF KERBEROS VERSION 4 MESSAGE EXCHANGES

(1) C → AS $ID_c \| ID_{tgs} \| TS_1$

(2) AS → C $E(K_c, [K_{c,tgs} \| ID_{tgs} \| TS_2 \| Lifetime_2 \| Ticket_{tgs}])$

$$Ticket_{tgs} = E(K_{tgs}, [K_{c,tgs} \| ID_C \| AD_C \| ID_{tgs} \| TS_2 \| Lifetime_2])$$

(a) Authentication Service Exchange to obtain ticket-granting ticket

(3) C → TGS $ID_v \| Ticket_{tgs} \| Authenticator_c$

(4) TGS → C $E(K_{c,tgs}, [K_{c,v} \| ID_v \| TS_4 \| Ticket_v])$

$$Ticket_{tgs} = E(K_{tgs}, [K_{c,tgs} \| ID_C \| AD_C \| ID_{tgs} \| TS_2 \| Lifetime_2])$$

$$Ticket_v = E(K_v, [K_{c,v} \| ID_C \| AD_C \| ID_v \| TS_4 \| Lifetime_4])$$

$$Authenticator_c = E(K_{c,tgs}, [ID_C \| AD_C \| TS_3])$$

(b) Ticket-Granting Service Exchange to obtain service-granting ticket

(5) C → V $Ticket_v \| Authenticator_c$

(6) V → C $E(K_{c,v}, [TS_5 + 1])$ (for mutual authentication)

$$Ticket_v = E(K_v, [K_{c,v} \| ID_C \| AD_C \| ID_v \| TS_4 \| Lifetime_4])$$

$$Authenticator_c = E(K_{c,v}, [ID_C \| AD_C \| TS_5])$$

(c) Client/Server Authentication Exchange to obtain service

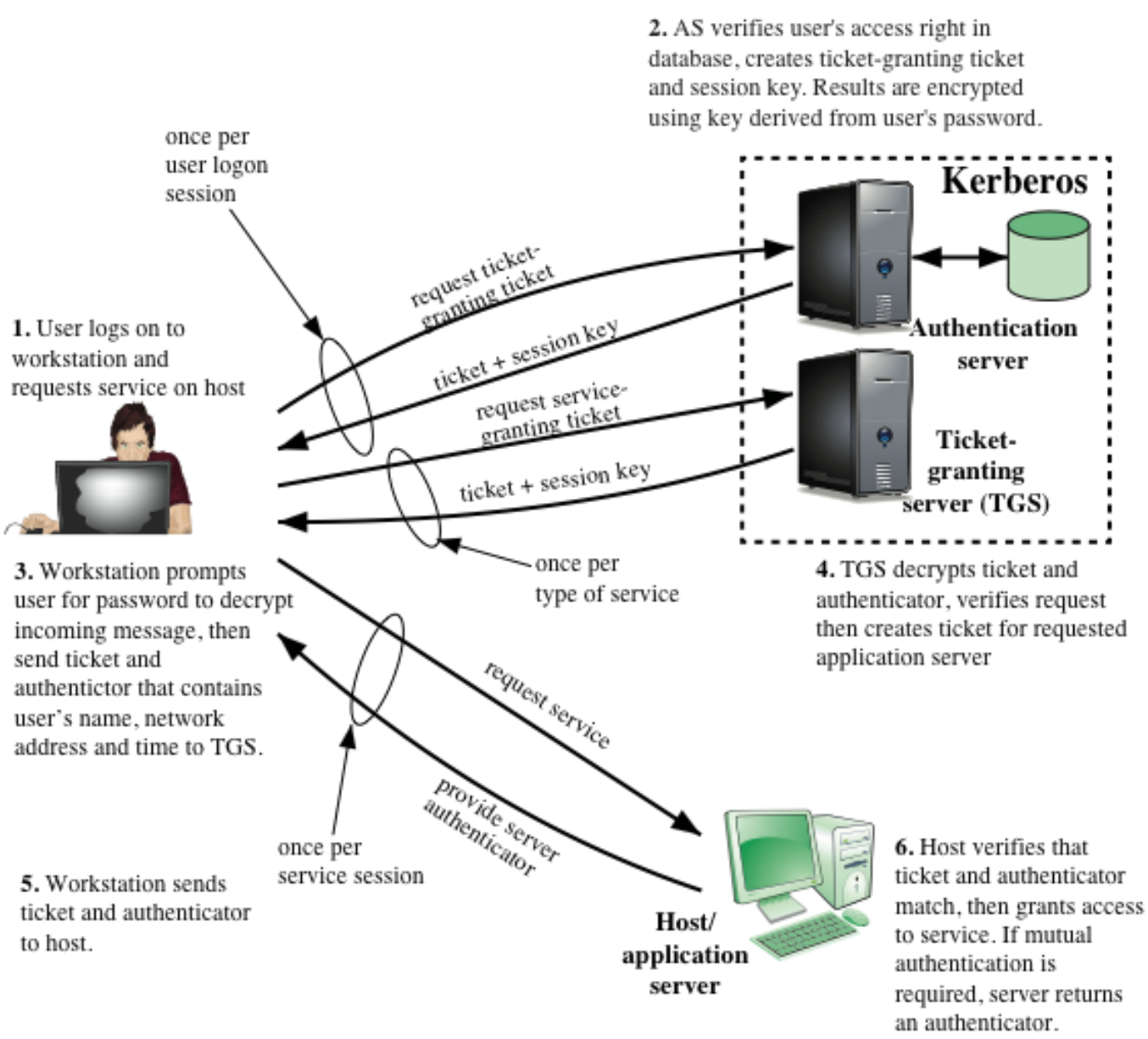


Figure 4.1 Overview of Kerberos

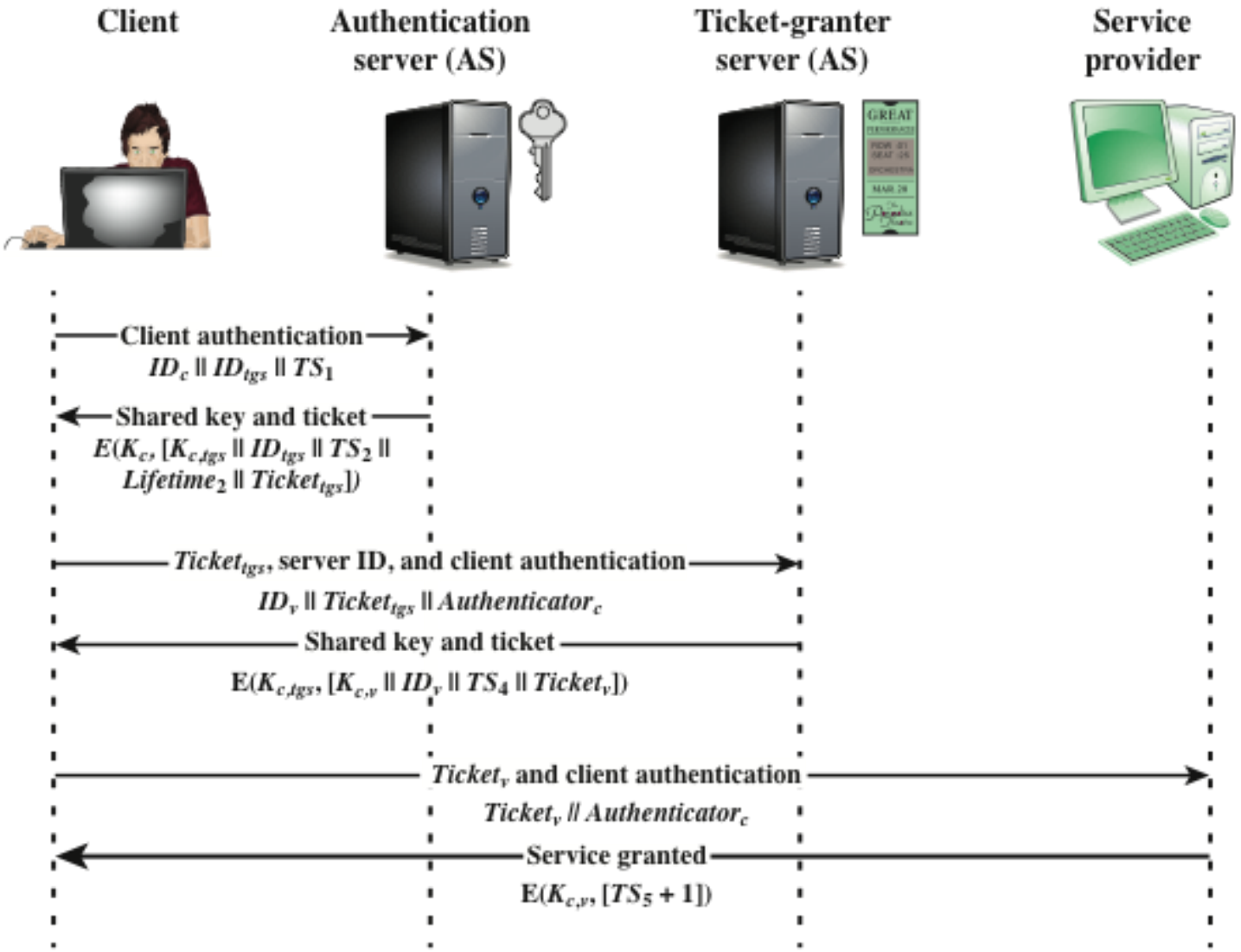


Figure 4.2 Kerberos Exchanges

Table 4.2 Rationale for the Elements of the Kerberos Version 4 Protocol
(page 1 of 3)

Message (1)	Client requests ticket-granting ticket.
ID_C	Tells AS identity of user from this client.
ID_{tgs}	Tells AS that user requests access to TGS.
TS_1	Allows AS to verify that client's clock is synchronized with that of AS.
Message (2)	AS returns ticket-granting ticket.
K_c	Encryption is based on user's password, enabling AS and client to verify password, and protecting contents of message (2).
$K_{c,tgs}$	Copy of session key accessible to client created by AS to permit secure exchange between client and TGS without requiring them to share a permanent key.
ID_{tgs}	Confirms that this ticket is for the TGS.
TS_2	Informs client of time this ticket was issued.
$Lifetime_2$	Informs client of the lifetime of this ticket.
$Ticket_{tgs}$	Ticket to be used by client to access TGS.

(a) Authentication Service Exchange

Table 4.2 Rationale for the Elements of the Kerberos Version 4 Protocol
(page 2 of 3)

Message (3)	Client requests service-granting ticket.
ID_V	Tells TGS that user requests access to server V.
$Ticket_{tgs}$	Assures TGS that this user has been authenticated by AS.
$Authenticator_c$	Generated by client to validate ticket .
Message (4)	TGS returns service-granting ticket.
$K_{c,tgs}$	Key shared only by C and TGS protects contents of message (4).
$K_{c,v}$	Copy of session key accessible to client created by TGS to permit secure exchange between client and server without requiring them to share a permanent key.
ID_V	Confirms that this ticket is for server V.
TS_4	Informs client of time this ticket was issued.
$Ticket_V$	Ticket to be used by client to access server V.
$Ticket_{tgs}$	Reusable so that user does not have to reenter password.
K_{tgs}	Ticket is encrypted with key known only to AS and TGS, to prevent Tampering.
$K_{c,tgs}$	Copy of session key accessible to TGS used to decrypt authenticator, thereby authenticating ticket.
ID_C	Indicates the rightful owner of this ticket.
AD_C	Prevents use of ticket from workstation other than one that initially requested the ticket.
ID_{tgs}	Assures server that it has decrypted ticket properly.
TS_2	Informs TGS of time this ticket was issued.
$Lifetime_2$	Prevents replay after ticket has expired.
$Authenticator_c$	Assures TGS that the ticket presenter is the same as the client for whom the ticket was issued has very short lifetime to prevent replay.
$K_{c,tgs}$	Authenticator is encrypted with key known only to client and TGS, to prevent tampering.
ID_C	Must match ID in ticket to authenticate ticket.
AD_C	Must match address in ticket to authenticate ticket.
TS_3	Informs TGS of time this authenticator was generated.

(b) Ticket-Granting Service Exchange

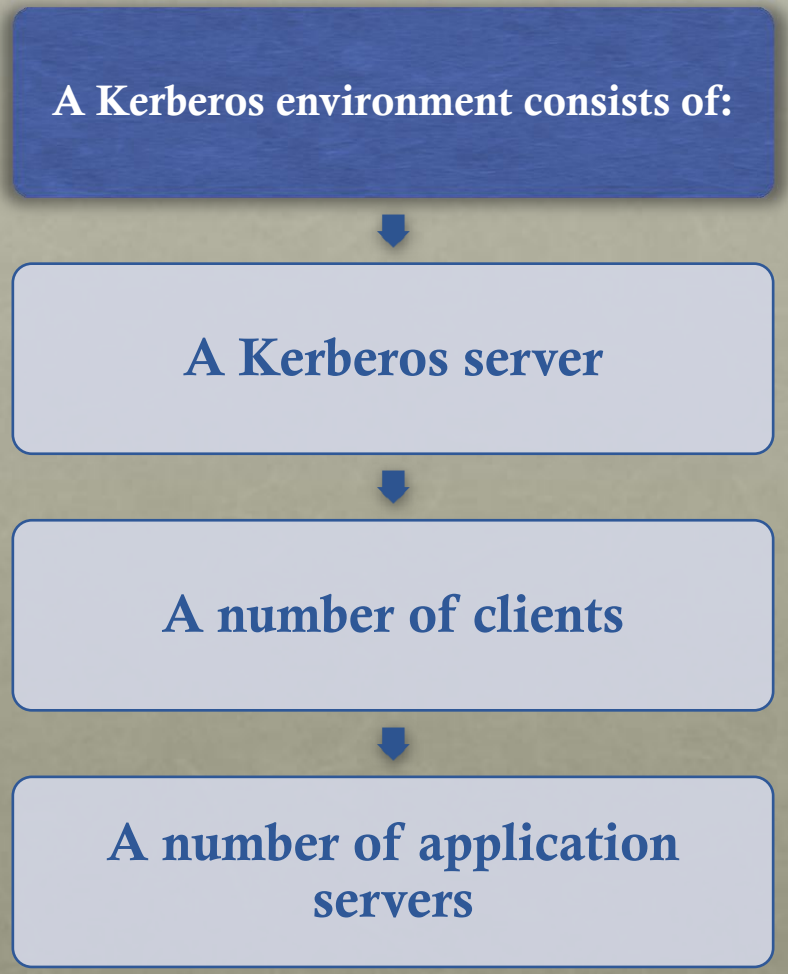
Table 4.2 Rationale for the Elements of the Kerberos Version 4 Protocol
(page 3 of 3)

Message (5)	Client requests service.
$Ticket_V$	Assures server that this user has been authenticated by AS.
$Authenticator_c$	Generated by client to validate ticket.
Message (6)	Optional authentication of server to client.
$K_{c,v}$	Assures C that this message is from V.
$TS_5 + 1$	Assures C that this is not a replay of an old reply.
$Ticket_v$	Reusable so that client does not need to request a new ticket from TGS for each access to the same server.
K_v	Ticket is encrypted with key known only to TGS and server, to prevent Tampering.
$K_{c,v}$	Copy of session key accessible to client; used to decrypt authenticator, thereby authenticating ticket.
ID_C	Indicates the rightful owner of this ticket.
AD_C	Prevents use of ticket from workstation other than one that initially requested the ticket.
ID_V	Assures server that it has decrypted ticket properly.
TS_4	Informs server of time this ticket was issued.
$Lifetime_4$	Prevents replay after ticket has expired.
$Authenticator_c$	Assures server that the ticket presenter is the same as the client for whom the ticket was issued; has very short lifetime to prevent replay.
$K_{c,v}$	Authenticator is encrypted with key known only to client and server, to prevent tampering.
ID_C	Must match ID in ticket to authenticate ticket.
AD_c	Must match address in ticket to authenticate ticket.
TS_5	Informs server of time this authenticator was generated.

(c) Client/Server Authentication Exchange

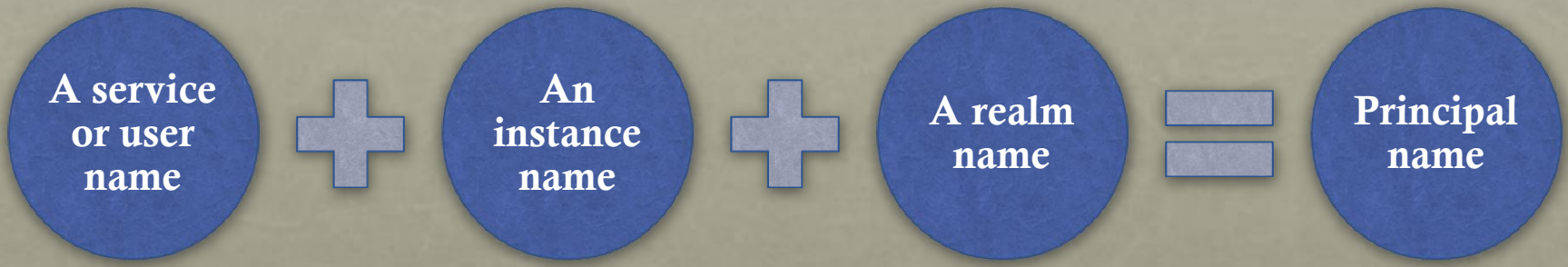
KERBEROS REALMS

- Kerberos realm
  - A set of managed nodes that share the same Kerberos database
  - The Kerberos database resides on the Kerberos master computer system, which should be kept in a physically secure room
  - A read-only copy of the Kerberos database might also reside on other Kerberos computer systems
  - All changes to the database must be made on the master computer system
  - Changing or accessing the contents of a Kerberos database requires the Kerberos master password



KERBEROS PRINCIPAL

- A service or user that is known to the Kerberos system
- Each Kerberos principal is identified by its principal name



Principal names consist of three parts

Table 4.3 Summary of Kerberos Version 5 Message Exchanges

(1) C → AS $Options \| IDc \| Realmc \| IDtgs \| Times \| Nonce1$

(2) AS → C $Realmc \| IDC \| Ticket tgs \| E(Kc, [Kc,tgs \| Times \| Nonce1 \| Realmtgs \| IDtgs])$

$Ticket tgs = E(Ktgs, [Flags \| Kc,tgs \| Realmc \| IDC \| ADC \| Times])$

(a) **Authentication Service Exchange to obtain ticket-granting ticket**

(3) C → TGS $Options \| IDv \| Times \| \| Nonce2 \| Ticket tgs \| Authenticatorc$

(4) TGS → C $Realmc \| IDC \| Ticket v \| E(Kc,tgs, [Kc,v \| Times \| Nonce2 \| Realmv \| IDv])$

$Ticket tgs = E(Ktgs, [Flags \| Kc,tgs \| Realmc \| IDC \| ADC \| Times])$

$Ticket v = E(Kv, [Flags \| Kc,v \| Realmc \| IDC \| ADC \| Times])$

$Authenticatorc = E(Kc,tgs, [IDC \| Realmc \| TS1])$

(b) **Ticket-Granting Service Exchange to obtain service-granting ticket**

(5) C → V $Options \| Ticket_v \| Authenticator_c$

(6) V → C $E_{Kc,v} [TS_2 \| Subkey \| Seq\#]$

$Ticket v = E(Kv, [Flags \| Kc,v \| Realmc \| IDC \| ADC \| Times])$

$Authenticatorc = E(Kc,v, [IDC \| Realmc \| TS2 \| Subkey \| Seq\#])$

(c) **Client/Server Authentication Exchange to obtain service**

KEY DISTRIBUTION USING ASYMMETRIC ENCRYPTION

- One of the major roles of public-key encryption is to address the problem of key distribution
- There are two distinct aspects to the use of public-key encryption in this regard:
  - The distribution of public keys
  - The use of public-key encryption to distribute secret keys
- Public-key certificate
  - Consists of a public key plus a user ID of the key owner, with the whole block signed by a trusted third party
  - Typically, the third party is a certificate authority (CA) that is trusted by the user community, such as a government agency or a financial institution
  - A user can present his or her public key to the authority in a secure manner and obtain a certificate
  - The user can then publish the certificate
  - Anyone needing this user's public key can obtain the certificate and verify that it is valid by way of the attached trusted signature

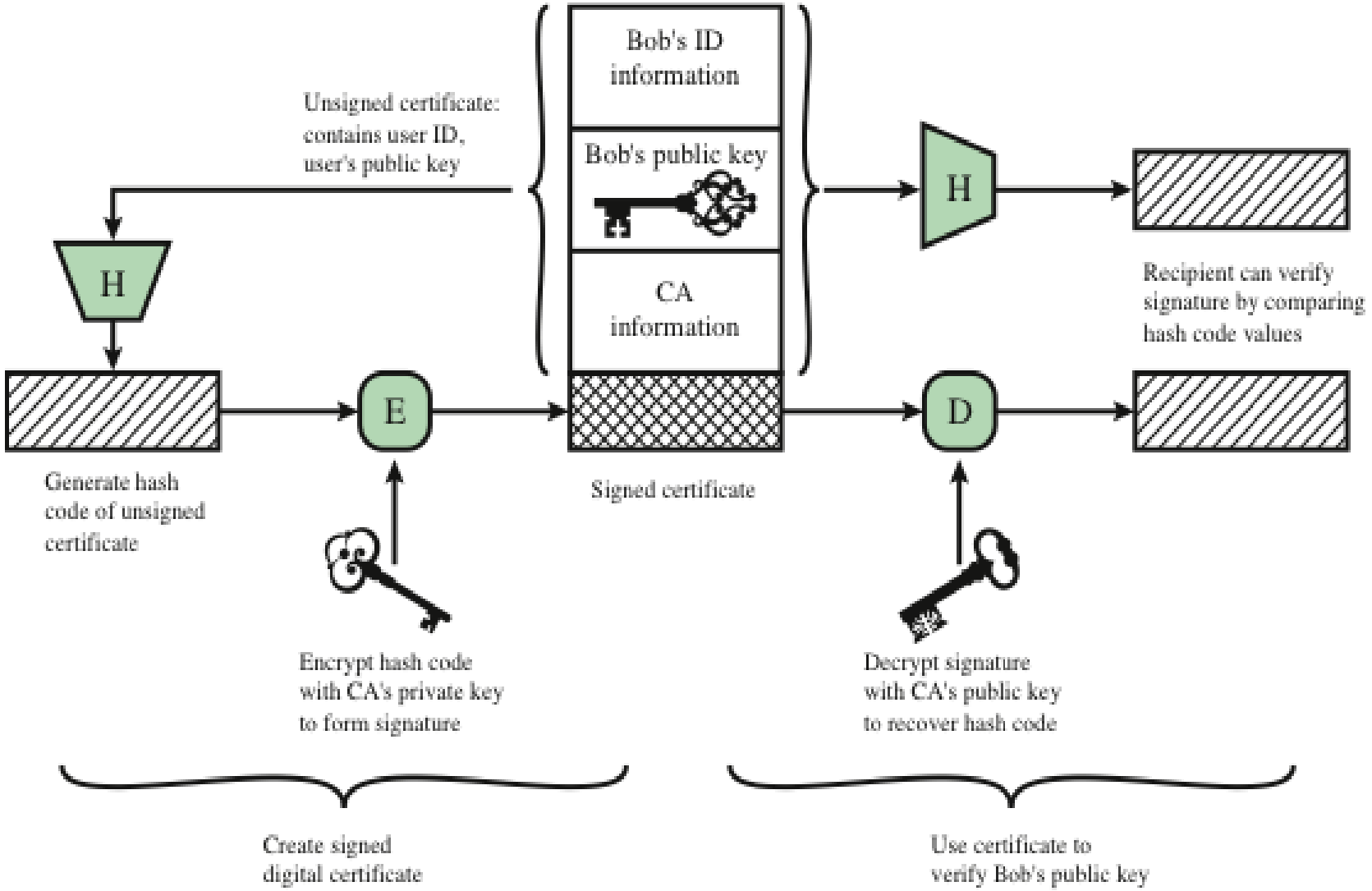
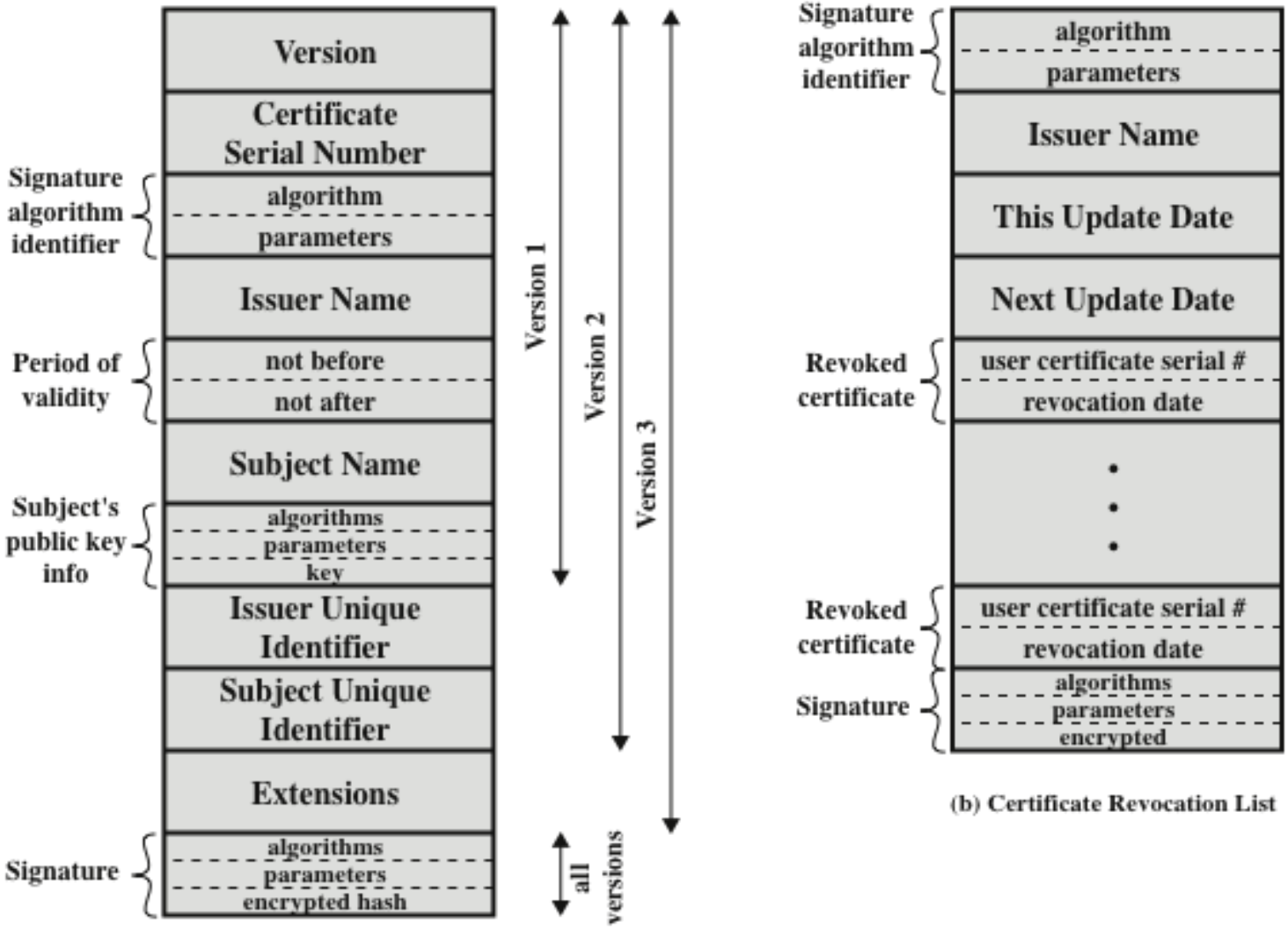


Figure 4.3 Public-Key Certificate Use

X.509 CERTIFICATES

- ITU-T recommendation X.509 is part of the X.500 series of recommendations that define a directory service
- Defines a framework for the provision of authentication services by the X.500 directory to its users
- The directory may serve as a repository of public-key certificates
- Defines alternative authentication protocols based on the use of public-key certificates
  - Was initially issued in 1988
  - Based on the use of public-key cryptography and digital signatures
- The standard does not dictate the use of a specific algorithm but recommends RSA



(a) X.509 Certificate

(b) Certificate Revocation List

Figure 4.4 X.509 Formats

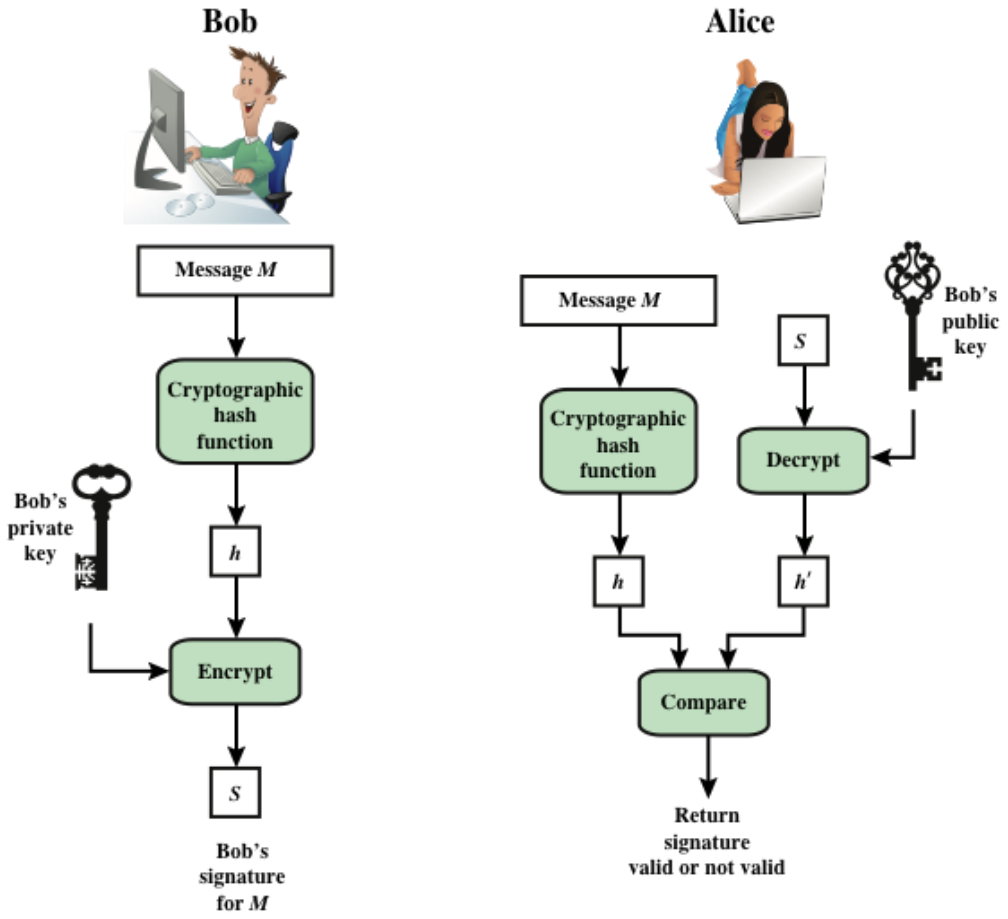


Figure 4.5 Simplified Depiction of Essential Elements of Digital Signature Process

The CA signs the certificate with its private key. If the corresponding public key is known to a user, then that user can verify that a certificate signed by the CA is valid.

Obtaining a User's Certificate

- User certificates generated by a CA have the following characteristics:
  - Any user with access to the public key of the CA can verify the user public key that was certified
  - No party other than the certification authority can modify the certificate without this being detected
- Because certificates are unforgeable, they can be placed in a directory without the need for the directory to make special efforts to protect them

- If A has obtained a certificate from CA X1 and B has obtained a certificate from CA X2. If A does not securely know the public key of X2, then B's certificate, issued by X2, is useless to A. A can read B's certificate, but A cannot verify the signature.

- If the two CAs have securely exchanged their own public keys, the following procedure will **enable A to obtain B's public key.**
  - 1. A obtains (from the directory) the certificate of X2 signed by X1. Because A securely knows X1's public key, A can obtain X2's public key from its certificate and verify it by means of X1's signature on the certificate.
  - 2. A then goes back to the directory and obtains the certificate of B signed by X2. Because A now has a trusted copy of X2's public key, A can verify the signature and securely obtain B's public key.
  - X1<<X2>>X2<<B>>

X.509 HIERARCHY

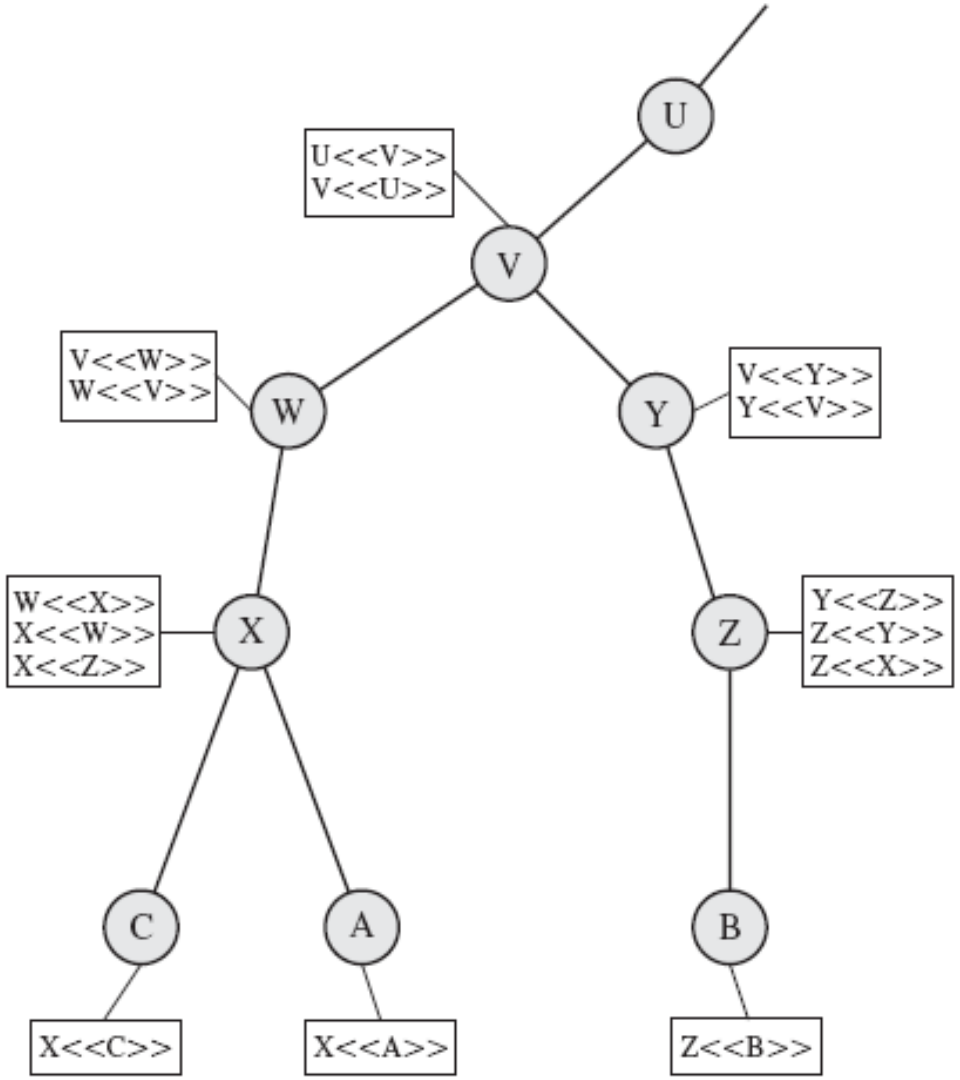


Figure 4.6 X.509 Hierarchy: A Hypothetical Example

- **Forward certificates:** Certificates of X generated by other CAs
- **Reverse certificates:** Certificates generated by X that are the certificates of other CAs

- User A can access the certificate of B using

X<<W>> W<<V>> V<<Y>> Y<<Z>> Z<<B>>

- B can access A's public key using

Z<<Y>> Y<<V>> V<<W>> W<<X>> X<<A>>

REVOCATION OF CERTIFICATES

- Each certificate includes a period of validity
- Typically a new certificate is issued just before the expiration of the old one
- It may be desirable on occasion to revoke a certificate before it expires for one of the following reasons:
  - The user's private key is assumed to be compromised
  - The user is no longer certified by this CA; reasons for this include subject's name has changed, the certificate is superseded, or the certificate was not issued in conformance with the CA's policies
  - The CA's certificate is assumed to be compromised

X.509 VERSION 3

Includes a number of optional extensions that may be added to the version 2 format

Each extension consists of:

- An extension identifier
- A criticality indicator
- An extension value

The certificate extensions fall into three main categories:

- Key and policy information
- Subject and issuer attributes
- Certification path constraints

KEY AND POLICY INFORMATION

- These extensions convey additional information about the subject and issuer keys, plus indicators of certificate policy
- A certificate policy is a named set of rules that indicates the applicability of a certificate to a particular community and/or class of application with common security requirements

Includes:

- Authority key identifier
- Subject key identifier
- Key usage
- Private-key usage period
- Certificate policies
- Policy mappings

CERTIFICATE SUBJECT AND ISSUER ATTRIBUTES

- These extensions support alternative names, in alternative formats, for a certificate subject or certificate issuer and can convey additional information about the certificate subject to increase a certificate user's confidence that the certificate subject is a particular person or entity

- For example, information such as postal address, position within a corporation, or picture image may be required.

Includes

- Subject alternative name: Contains one or more alternative names, using any of a variety of forms. This field is important for supporting certain applications, such as electronic mail, EDI, and IPSec, which may employ their own name forms.
- Issuer alternative name: Contains one or more alternative names, using any of a variety of forms.
- Subject directory attributes: Conveys any desired X.500 directory attribute values for the subject of this certificate.

CERTIFICATION PATH CONSTRAINTS

- These extensions allow constraint specifications to be included in certificates issued for CAs by other CAs
- The constraints may restrict the types of certificates that can be issued by the subject CA or that may occur subsequently in a certification chain

Includes:

- Basic constraints: Indicates if the subject may act as a CA. If so, a certification path length constraint may be specified.
- Name constraints: Indicates a name space within which all subject names in subsequent certificates in a certification path must be located.
- Policy constraints: Specifies constraints that may require explicit certificate policy identification or inhibit policy mapping for the remainder of the certification path.