Chapter 4

Authentication Applications

Henric Johnson

Blekinge Institute of Technology,Sweden

http://www.its.bth.se/staff/hjo/

henric.johnson@bth.se

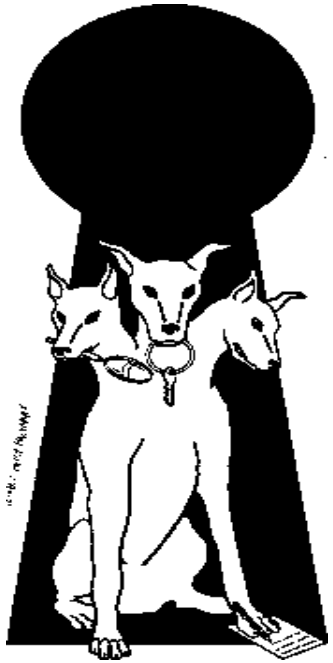
Outline

- Security Concerns
- Kerberos
- X.509 Authentication Service
- Recommended reading and Web Sites

Security Concerns

- Key concerns are **confidentiality** and **timeliness**
  - to provide confidentiality must encrypt identification and session key info
    - which requires the use of previously shared private or public keys
  - need timeliness to prevent **replay attacks**
    - provided by using sequence numbers or timestamps or challenge/response

KERBEROS



In Greek mythology, a many headed dog, the guardian of the entrance of Hades

KERBEROS

- Users wish to access services on servers.
- Three threats exist:
  - User pretend to be another user.
  - User alter the network address of a workstation.
  - User eavesdrop on exchanges and use a replay attack.
- Kerberos is an authentication service.
- **Distributed V.S. Centralized authentication service**

KERBEROS

- Motivation

For a small, closed environment, in which all systems are owned and operated by a single organization.

1. Rely on each individual client workstation to assure the identity of its user or users and rely on each server to enforce a security policy based on user ID.
2. Require that client systems authenticate themselves to servers, but trust the client system concerning the identity of its user.
3. Require the user to prove identity for each service invoked. Also require that servers prove their identity to clients.

- This 3rd approach is supported by Kerberos.
  - Kerberos assumes a distributed client/server architecture and employs one or more Kerberos servers to provide an authentication service.

KERBEROS

- Provides **a centralized authentication server** to authenticate users to servers and servers to users.
- Relies on conventional encryption, making no use of public-key encryption
- Two versions: version 4 and 5
- Version 4 makes use of DES
- A trusted third-party authentication service
  - Clients and Servers trust Kerberos to mediate their mutual authentication.

Kerberos Version 4

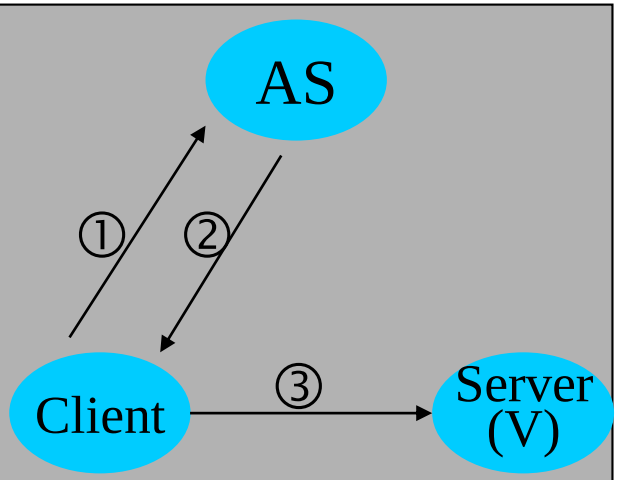
- Terms:
  - C = Client
  - AS = authentication server
  - V = server
  - ID_C = identifier of user on C
  - ID_V = identifier of V
  - P_c = password of user on C
  - ADc = network address of C
  - K_V = secret encryption key shared by AS an V
  - TS = timestamp
  - || = concatenation

A Simple Authentication Dialogue

(1) C → AS: IDc || P_c || IDv

(2) AS → C: Ticket

(3) C → V: IDc || Ticket



Ticket = E_{K_v}[IDc || **AD_c** || IDv]

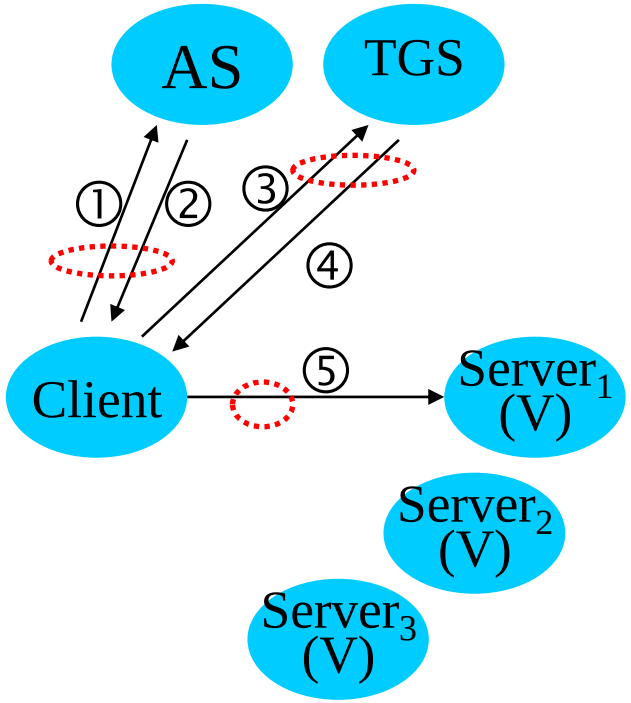
The ticket is valid only if ***it is transmitted from the same workstation that initially requested the ticket***.

A More Secure Authentication Dialogue

- Two problems in the foregoing scenario:
  - Minimize the number of times that a user has to enter a password.
    - Suppose each ticket can be used only once.
    - Each attempt requires reentering the password. ⇐ either requirements of same or different services
    - ➢ Idea: **Reusable Tickets for requirements of the same service**.
      - The similar problems will be appeared again for requirements of the different services.
  - The earlier scenario involved a plaintext transmission of the password. (message #1)
- Solution
  - A new server, the ***Ticket-Granting Server*** (TGS), involved.

A More Secure Authentication Dialogue

- **Once per user logon session**
  - (1) C → AS: $ID_C||ID_{tgs}$
  - (2) AS → C: $E_{K_C}[Ticket_{tgs}]$
- **Once per type of service**
  - (3) C → TGS: $ID_C||ID_V||Ticket_{tgs}$
  - (4) TGS → C: $Ticket_v$
- **Once per service session**
  - (5) C → V: $ID_C||Ticket_v$



$Ticket_{tgs} = E_{K_{tgs}}[ID_C||AD_C||ID_{tgs}||TS_1||Lifetime_1]$

$Ticket_v = E_{K_v}[ID_C||AD_C||ID_V||TS_2||Lifetime_2]$

A More Secure Authentication Dialogue

- Improvements
  - Only one password query per user session,
  - Protection of the user password.
- Problems:
  - Lifetime associated with the ticket-granting ticket (TGT) and service-granting ticket (SGT)
    - If too short → repeatedly asked for password
    - If too long → greater opportunity to replay
    - The threat is that an opponent will steal the ticket and use it before it expires
  - A requirement for servers to authenticate themselves to users.

Version 4 Authentication Dialogue

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

(1) C → AS: $ID_c \| ID_{tgs} \| TS_1$

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

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

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

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

Use only once and has a very short lifetime

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

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

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

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

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

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

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

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

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

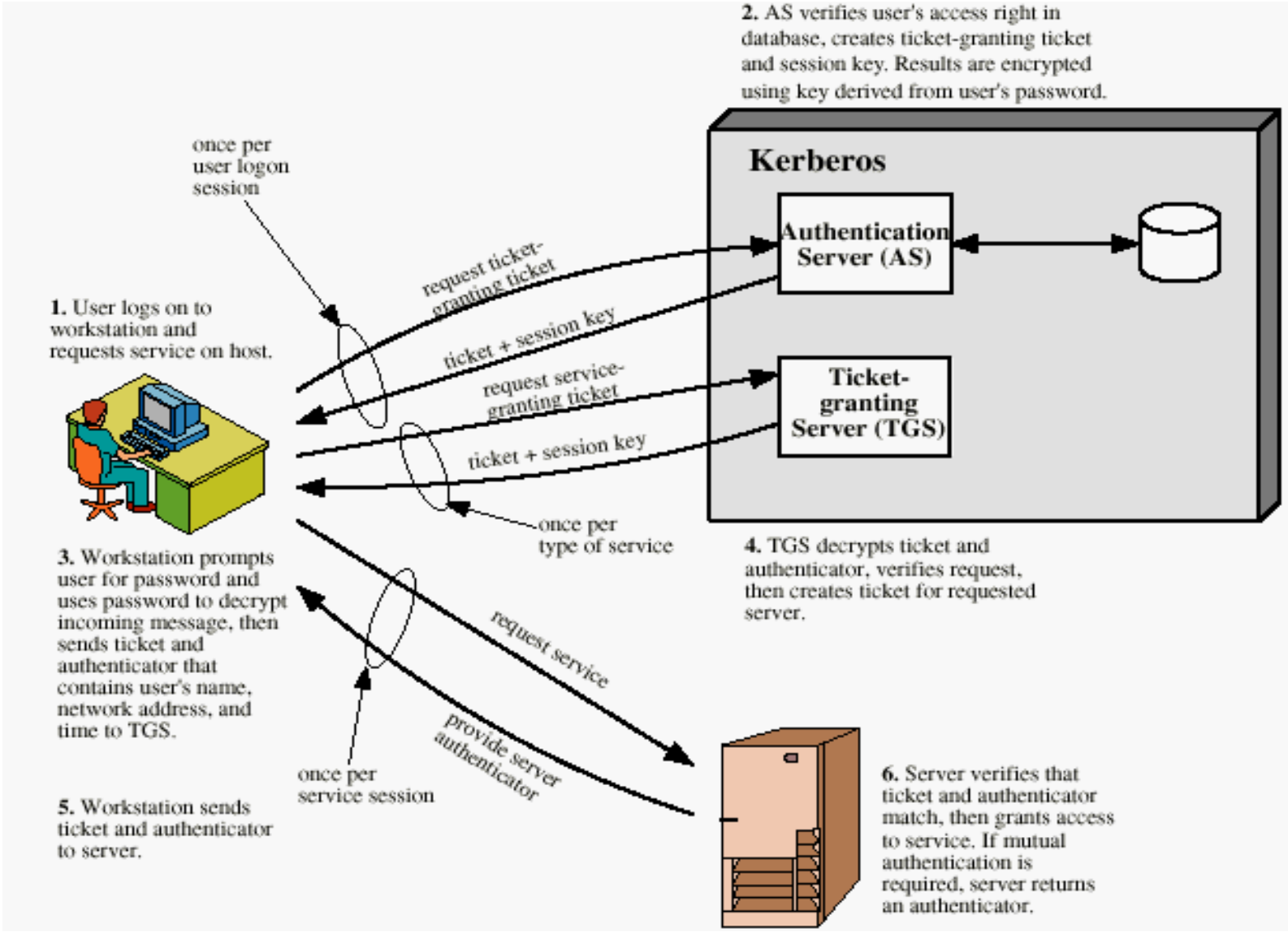
Kerberos Realms and Multiple Kerberi

- A full-service Kerberos environment—a ***realm***
  - A Kerberos server, a number of clients, and a number of application servers.
- Requirements
  1. The Kerberos server must have the user ID and **hashed password** of all participating users in its database.

     ⇒ User register
  2. The Kerberos server must share a secret key with each server.

     ⇒ Servers register

Overview of Kerberos



Kerberos Realms and Multiple Kerberi

- Inter-realm authentication: The third requirement is added:

  3. The Kerberos server in each interoperating realm shares a secret key with the server in the other realm.

  $\Rightarrow$ The two Kerberos servers are registered with each other.

  $\Rightarrow$ The Kerberos server in one realm trust the Kerberos server in the other realm to authenticate its users.

Request for Service in Another Realm

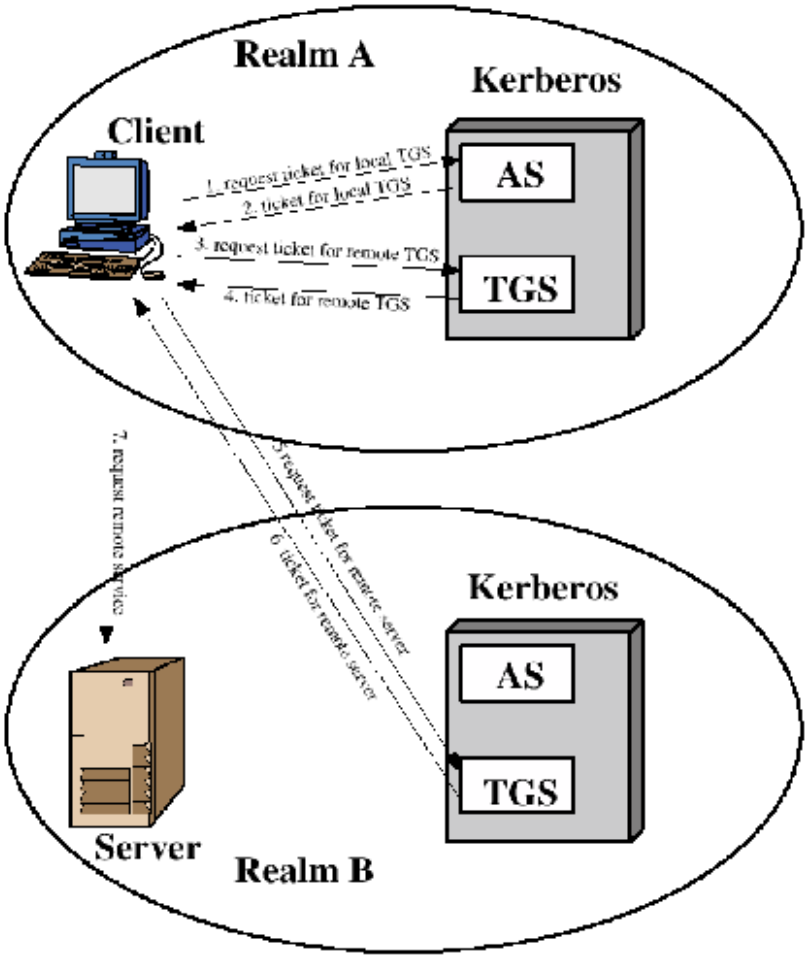


Figure 4.2 Request for Service in Another Realm

Request for Service in Another Realm

(1) C→AS: $ID_c || ID_{tgs} || TS_1$

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

(3) C→TGS: $ID_{tgsrem} || Ticket_{tgs} || Authenticator_c$

(4) TGS→C: $E_{K_{c,tgs}}[K_{c,tgsrem} || ID_{tgsrem} || TS_4 || Ticket_{tgsrem}]$

(5) C→TGS_{rem}: $ID_{vrem} || Ticket_{tgsrem} || Authenticator_c$

(6) TGS_{rem}→C: $E_{K_{c,tgsrem}}[K_{c,vrem} || ID_{vrem} || TS_6 || Ticket_{vrem}]$

(7) C→V_{rem}: $Ticket_{vrem} || Authenticator_c$

Problem: It doesn't scale well to many realms.
If N realms, then N(N-1)/2 secure key exchanges.

Difference Between Version 4 and 5

- Environment shortcomings of version 4
  1. Encryption system dependence (V.4 DES)
  2. Internet protocol dependence
  3. Message byte ordering
  4. Ticket lifetime
  5. Authentication forwarding
  6. Interrealm authentication

- Technical deficiencies of version 4
  1. Double encryption
  2. PCBC encryption
  3. Session keys
  4. Password attacks

The Version 5 Authentication Dialogue

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

(1) C → AS: $\text{Options} \| ID_c \| Realm_c \| ID_{tgs} \| Times \| Nonce_1$

(2) AS → C: $Realm_c \| ID_C \| Ticket_{tgs} \| E_{K_c}[K_{c,tgs} \| Times \| Nonce_1 \| Realm_{tgs} \| ID_{tgs}]$

$$Ticket_{tgs} = E_{K_{tgs}}[Flags \| K_{c,tgs} \| Realm_c \| ID_C \| AD_C \| Times]$$

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

(3) C → TGS: $\text{Options} \| ID_v \| Times \| \| Nonce_2 \| Ticket_{tgs} \| Authenticator_c$

(4) TGS → C: $Realm_c \| ID_C \| Ticket_v \| E_{K_{c,tgs}}[K_{c,v} \| Times \| Nonce_2 \| Realm_v \| ID_V]$

$$Ticket_{tgs} = E_{K_{tgs}}[Flags \| K_{c,tgs} \| Realm_c \| ID_C \| AD_C \| Times]$$

$$Ticket_v = E_{K_v}[Flags \| K_{c,v} \| Realm_c \| ID_C \| AD_C \| Times]$$

$$Authenticator_c = E_{K_{c,tgs}}[ID_C \| Realm_c \| TS_1]$$

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

(5) C → V: $\text{Options} \| \text{Ticket}_v \| \text{Authenticator}_c$

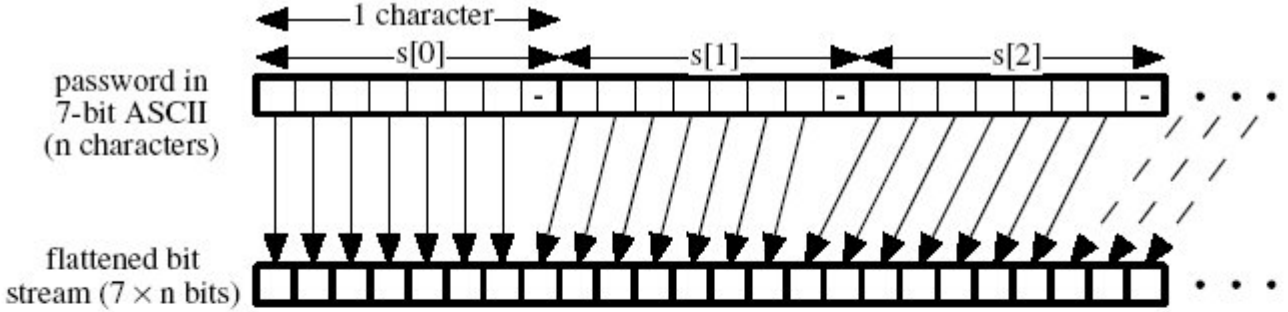
(6) V → C: $E_{K_{c,v}}[TS_2 \| \text{Subkey} \| \text{Seq\#}]$

$$Ticket_v = E_{K_v}[Flags \| K_{c,v} \| Realm_c \| ID_C \| AD_C \| Times]$$

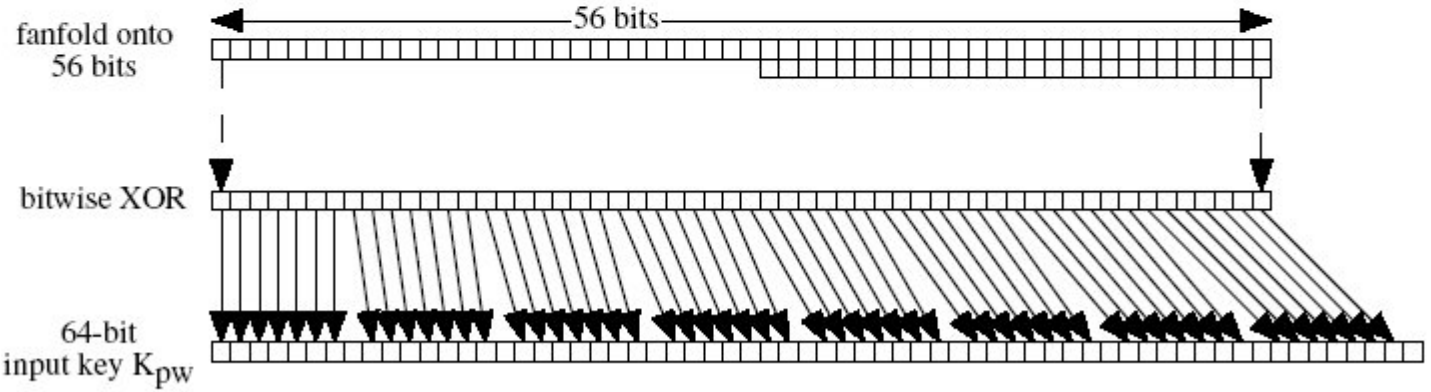
$$Authenticator_c = E_{K_{c,V}}[ID_C \| Realm_c \| TS_2 \| Subkey \| Seq\#]$$

Kerberos Version 5 Flags

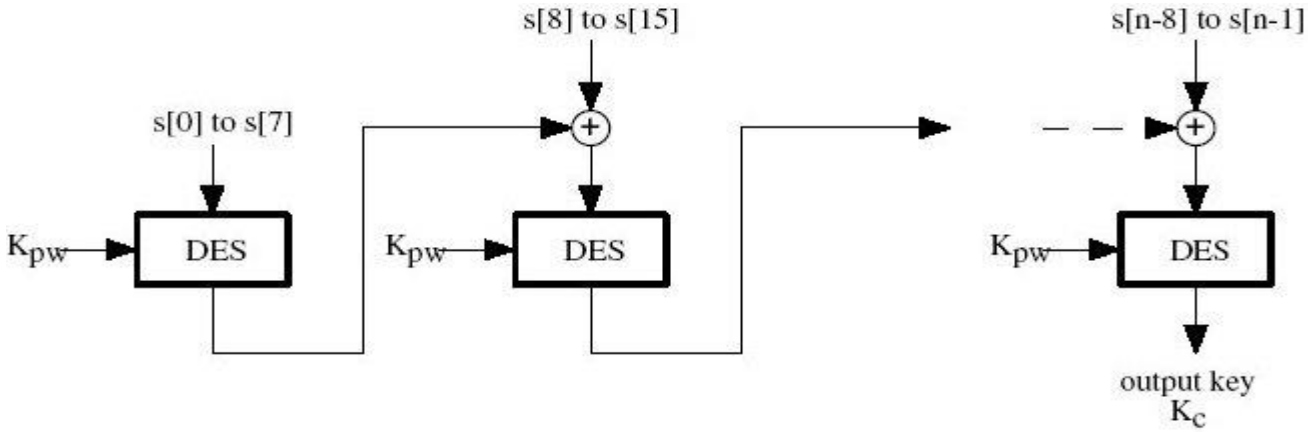
INITIAL	This ticket was issued using the AS protocol and not issued based on a ticket-granting ticket.
PRE-AUTHENT	During initial authentication, the client was authenticated by the KDC before a ticket was issued.
HW-AUTHENT	The protocol employed for initial authentication required the use of hardware expected to be possessed solely by the named client.
RENEWABLE	Tells TGS that this ticket can be used to obtain a replacement ticket that expires at a later date.
MAY-POSTDATE	Tells TGS that a postdated ticket may be issued based on this ticket-granting ticket.
POSTDATED	Indicates that this ticket has been postdated; the end server can check the authtime field to see when the original authentication occurred.
INVALID	This ticket is invalid and must be validated by the KDC before use.
PROXIABLE	Tells TGS that a new service-granting ticket with a different network address may be issued based on the presented ticket.
PROXY	Indicates that this ticket is a proxy.
FORWARDABLE	Tells TGS that a new ticket-granting ticket with a different network address may be issued based on this ticket-granting ticket.
FORWARDED	Indicates that this ticket has either been forwarded or was issued based on authentication involving a forwarded ticket-granting ticket.



(a) Convert password to bit stream



(b) Convert bit stream to input key



(c) Generate DES CBC checksum of password

DES PCBC Mode

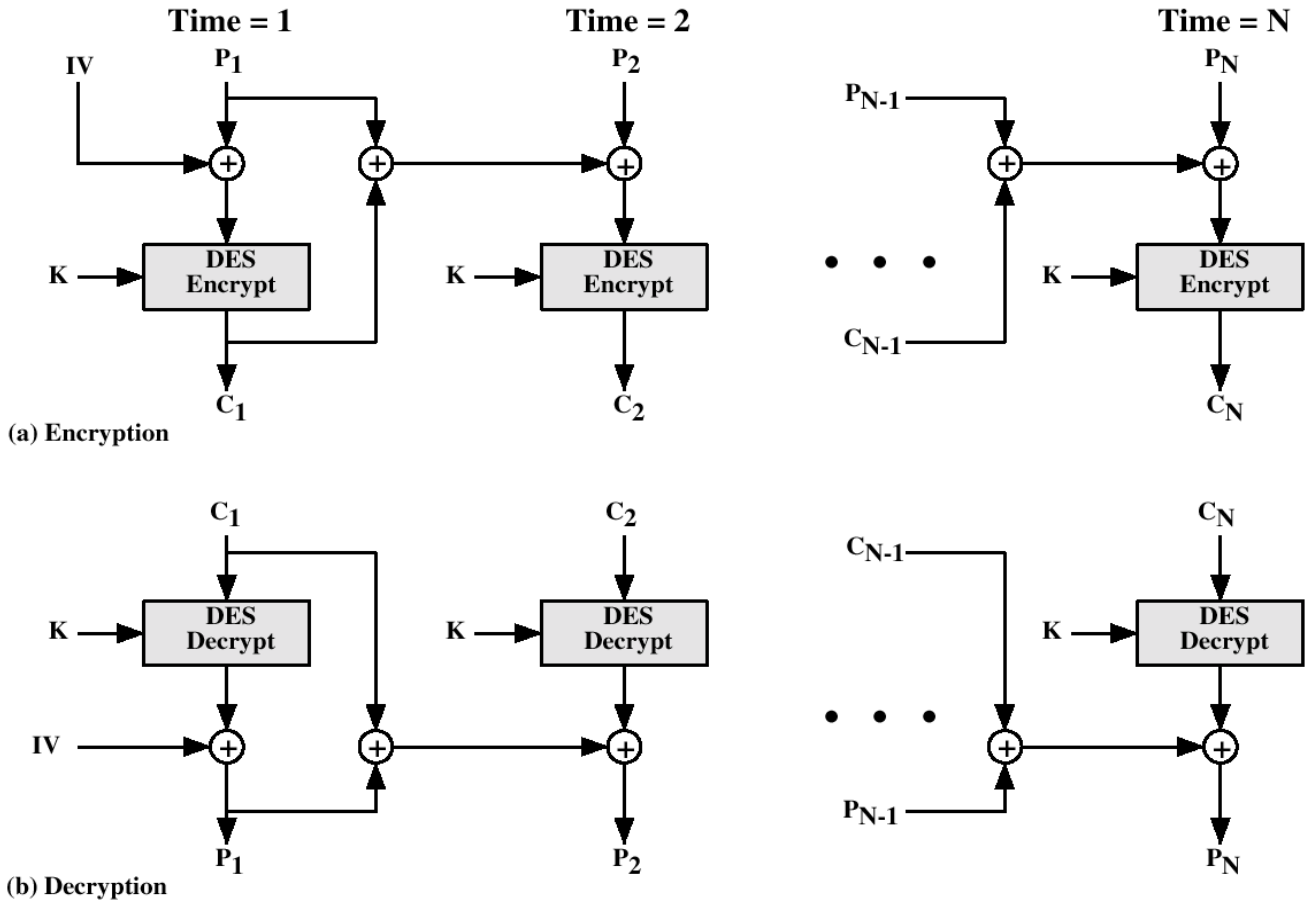
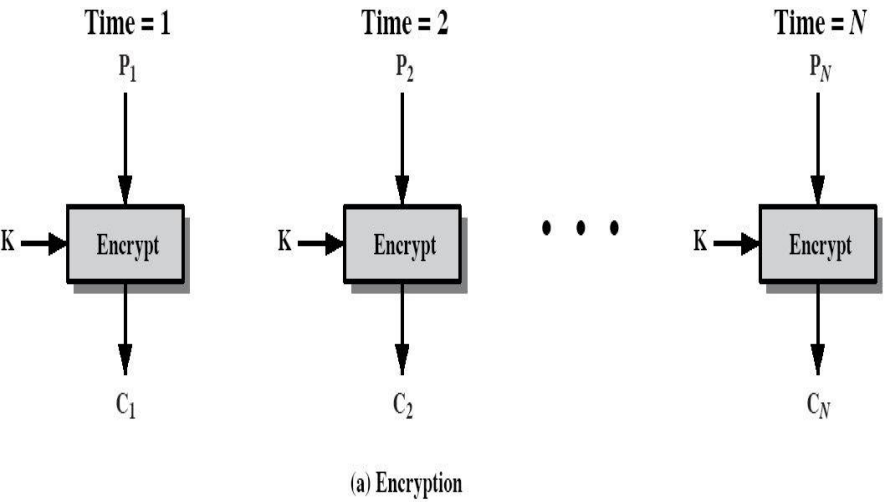
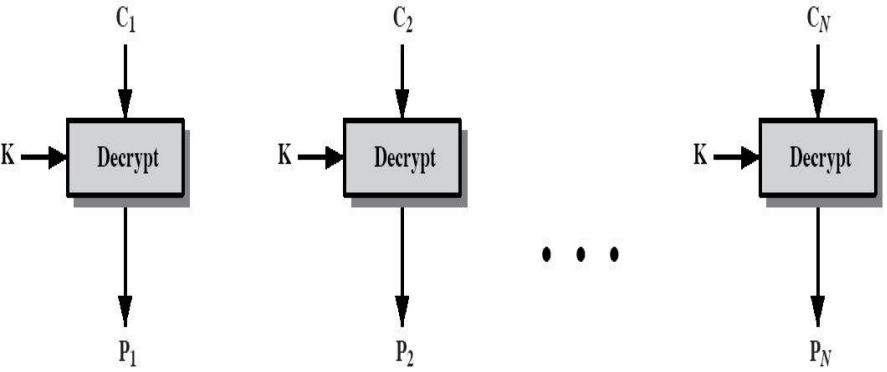


Figure 4.7 Propagating Cipher Block Chaining (PCBC) Mode

DES ECB Mode

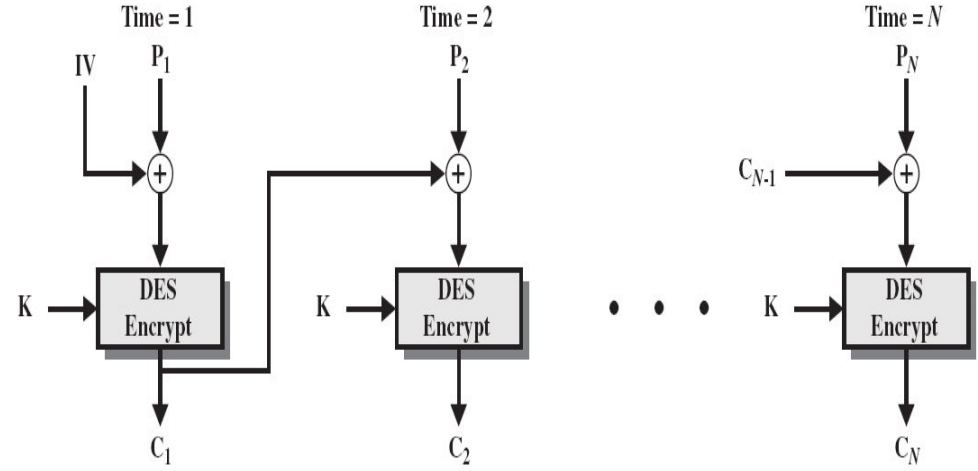


(a) Encryption

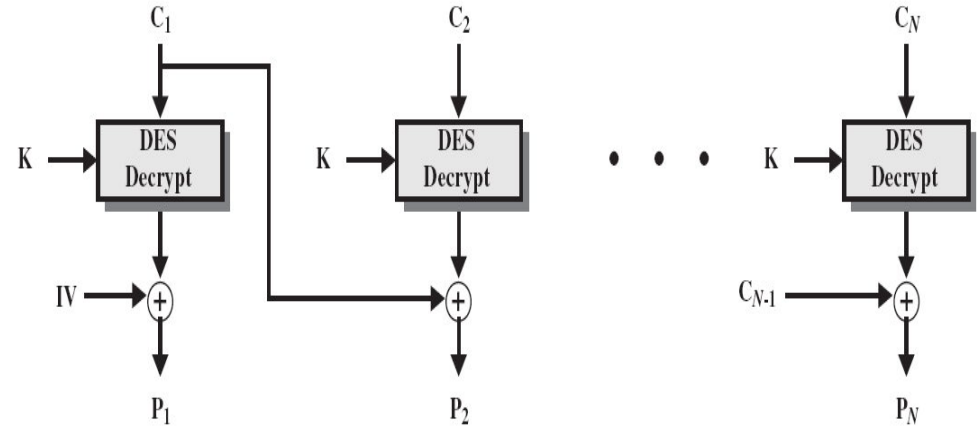


(b) Decryption

DES CBC Mode



(a) Encryption



(b) Decryption

ECB V.S. CBC V.S. PCBC

- ECB (Electronic Codebook) mode
  - Each plaintext block is independently encrypted.
- CBC (Cipher Block Chaining) mode
  - **The same plaintext block, if repeated, produces different ciphertext blocks.**
  - If an error occurs in transmission of ciphertext block C_I, then this error propagates to the recovered plaintext blocks P_I and P_{I+1}.
- PCBC (Propagating Cipher Block Chaining) Mode
  - **An error in one ciphertext block is propagated to all subsequent decrypted blocks of the message, rendering each block useless.**
  - Data encryption and integrity are combined in one operation.

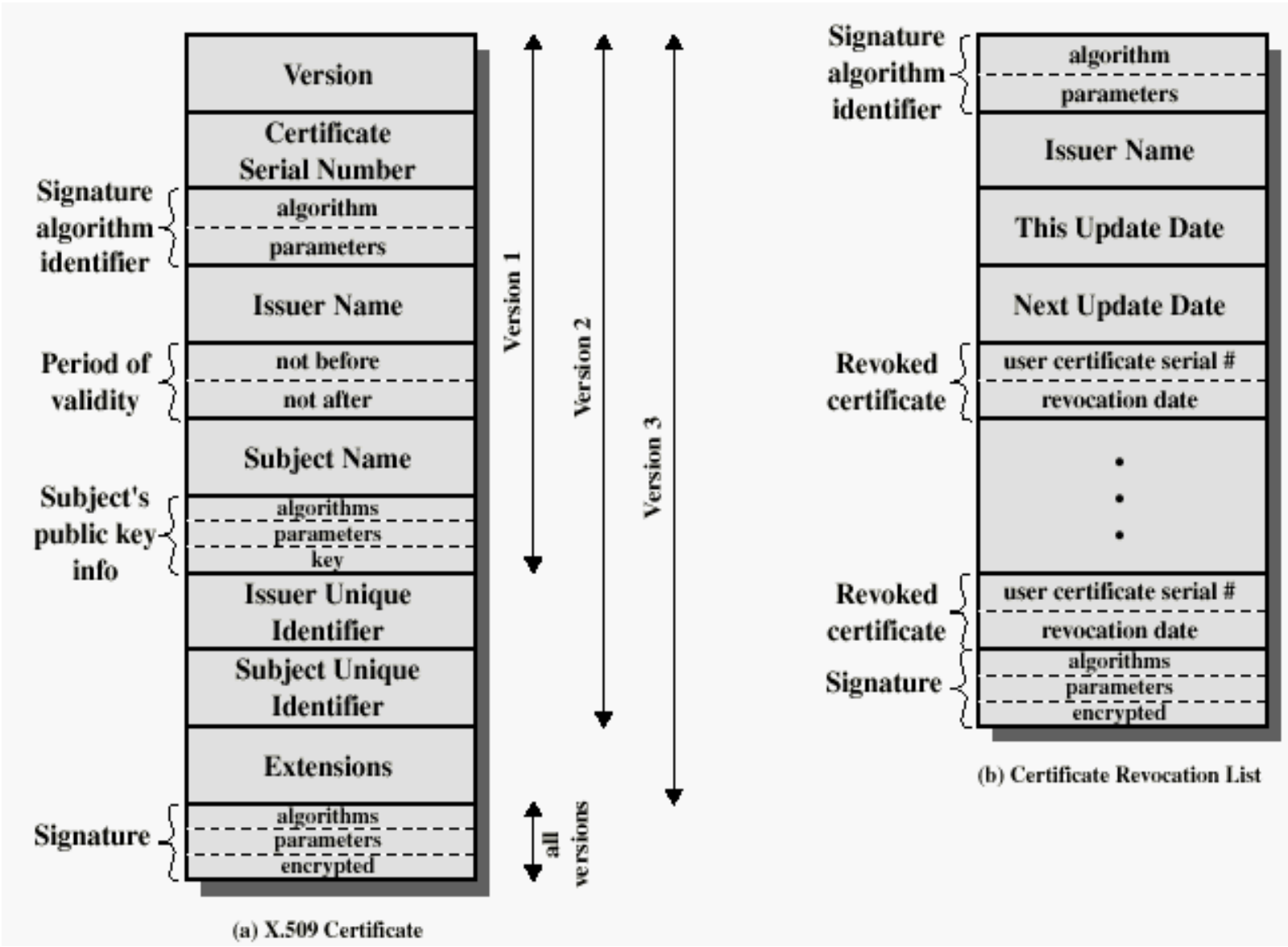
Kerberos - in practise

- **Currently have two Kerberos versions:**
- 4 : restricted to a single realm
- 5 : allows inter-realm authentication, in beta test
- Kerberos v5 is an Internet standard
- specified in RFC1510, and used by many utilities
- **To use Kerberos:**
- need to have a KDC on your network
- need to have Kerberised applications running on all participating systems
- major problem - US export restrictions
- Kerberos cannot be directly distributed outside the US in source format (& binary versions must obscure crypto routine entry points and have no encryption)
- else crypto libraries must be reimplemented locally

X.509 Authentication Service

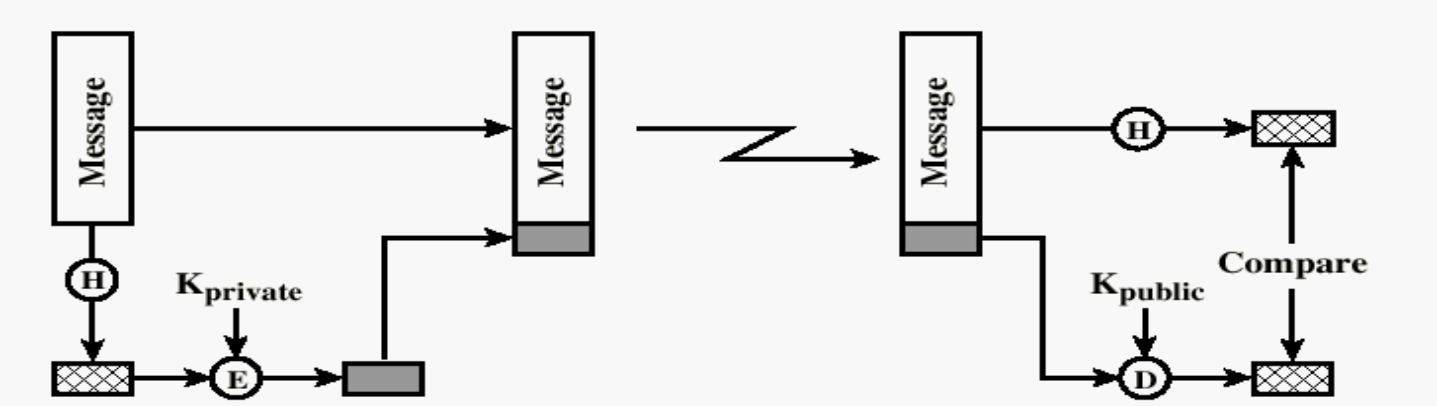
- X.509
  - A directory service
    - Distributed set of servers that maintains a database about users.
  - A framework for the provision of authentication services
    - Defined alternative authentication protocols based on the use of public-key certificates.
  - The heart of the X.509 scheme
    - Public-key certificate
    - Each certificate contains the public key of a user and is signed with the private key of a CA.
  - Based on the use of public-key cryptography and digital signatures
    - RSA is recommended to use.
    - Digital signature scheme is assumed to require the use of a hash function.
- Applications
  - S/MIME, IP Security, SSL/TLS and SET

X.509 Formats



X.509 Formats

- Format of X.509
  - CA<<A>> = CA {V, SN, AI, CA, TA, A, AP}
  - Y<<X>>
    - The certificate of user X issued by certification authority Y.
  - Y{I}
    - The signing of I by Y.
- Typical digital signature approach



(b) Using public-key encryption

Obtaining a User's Certificate

- Characteristics of certificates generated by CA
  - Any user with access to the public key of the CA can recover the user public key that was certified.
  - No part other than the CA 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.

Obtaining a User's Certificate

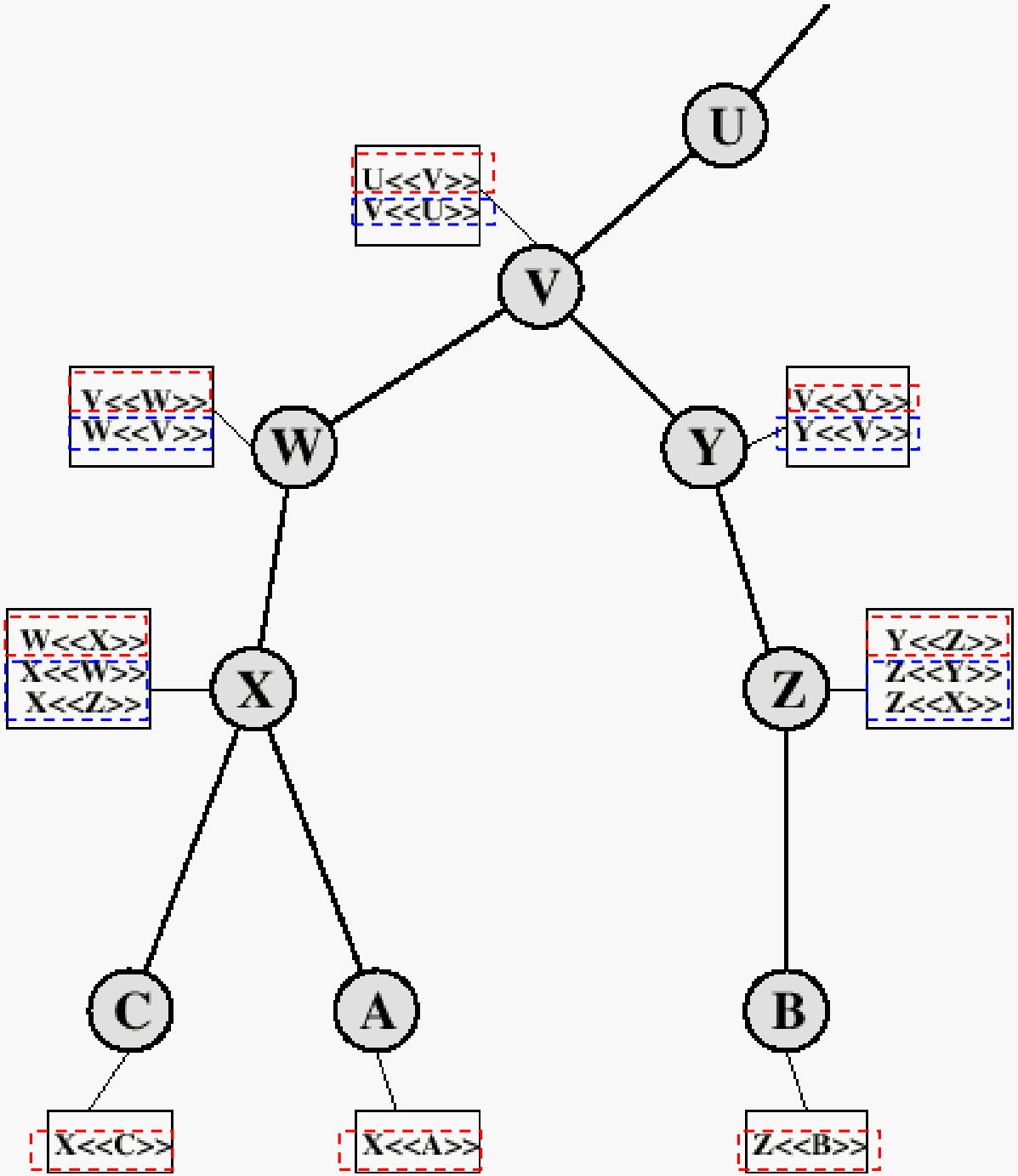
- If all users subscribe to the same CA, you can get somebody's certificate by
  - The directory service of CA
  - Exchange of certificates each other
- For a large of community of users
  - It may not be practical for all users to subscribe to the same CA.
  - X.509 suggests that CAs be arranged in a hierarchy.

Obtaining a User's Certificate

- Scenario
  - The certificate of user A: X_1<<A>>
  - The certificate of user B: X_2<<B>>
  - Two CAs have securely exchange their own public keys.
    - X_1<< X_2 >> , X_2<< X_1 >>
  - How does A conduct a secured and signed message to B, and vice versa?
- ***A chain of certificates***
  - A's chain of certificates
    - X_1<< X_2 >> X_2<<B>>
  - B's chain of certificates
    - X_2<< X_1 >> X_1<<A>>

Each pair of CAs in the chain (X_i,X_{i+1}) must have created certificates for each other.

X.509 CA Hierarchy



Forward certificate

Reverse certificate

Question:

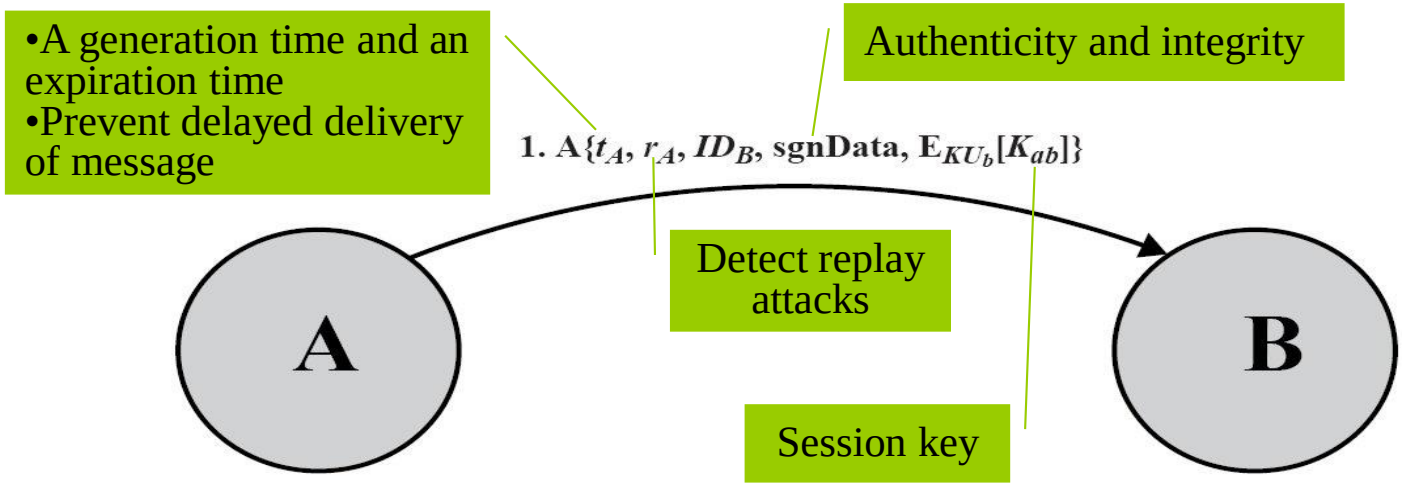
If user A wants to make a secure and signed communication, how does A acquire certificates from the directory to establish a certification path to B? And vice versa.

Revocation of Certificates

- Reasons for revocation
  - The users secret key is assumed to be compromised.
  - The user is no longer certified by this CA.
  - The CA's certificate is assumed to be compromised.
- ***Certification revocation list (CRL)***
  - CRL is signed by the issuer and includes:
    - Issuer's name, the date the list was created, the date the next CRL is scheduled to be issued, and an entry for each revoked certificate.
    - Each entry consists of:
      - Serial number of a certificate and revocation date for that certificate.

Authentication Procedures

- One-way authentication
  - A single transfer of information from one user (A) to another (B), and establishes the following:
    - The identity of A and that the message was generated by A
    - That the message was intended for B
    - The integrity and originality (it has not been sent multiple times) of the message

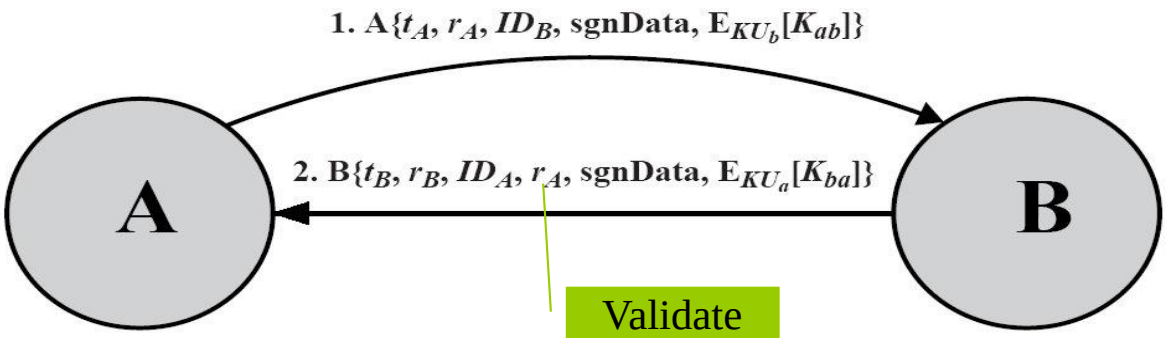


t_A: timestamp
r_A: nonce
ID_B: identity of B

(a) One-way authentication

Authentication Procedures

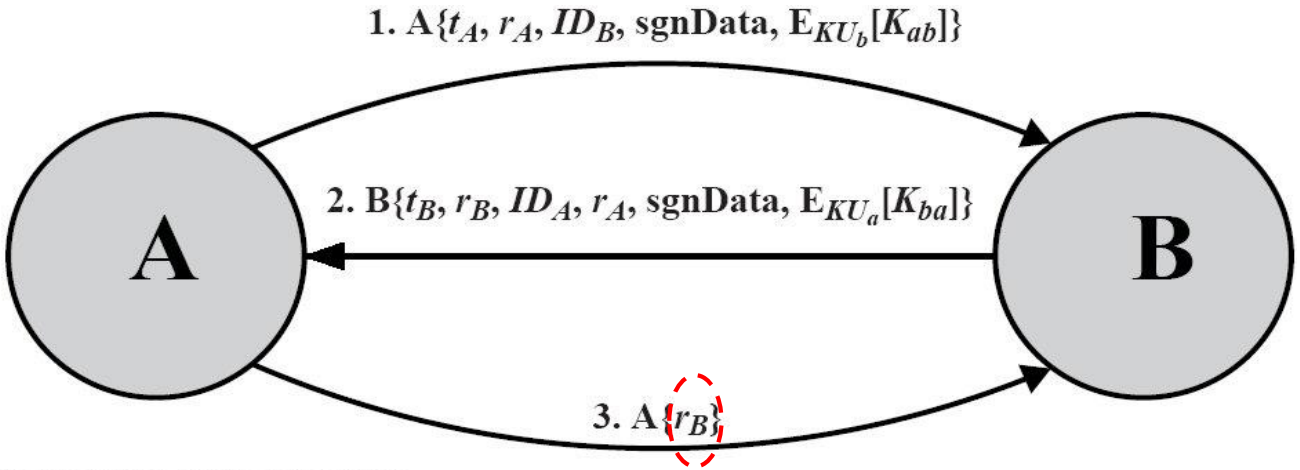
- Two-way authentication
  - The identity of B and that the reply message was generated by B
  - That the message was intended for A
  - The integrity and originality of the reply



(b) Two-way authentication

Authentication Procedures

- Three-way authentication: The intent of this design:
  - **Timestamps need not be checked.**
    - Because both nonces are echoed back by the other side, each side can check the returned nonce to detect replay attacks.
  - This approach is needed when **synchronized clocks are not available**.



(c) Three-way authentication

Recommended Reading and WEB Sites

- www.whatis.com (search for kerberos)
- Bryant, W. Designing an Authentication System: A Dialogue in Four Scenes. http://web.mit.edu/kerberos/www/dialogue.html
- Kohl, J.; Neuman, B. “The Evolotion of the Kerberos Authentication Service” http://web.mit.edu/kerberos/www/papers.html
- http://www.isi.edu/gost/info/kerberos/