Kerberos

V. Balasubramanian

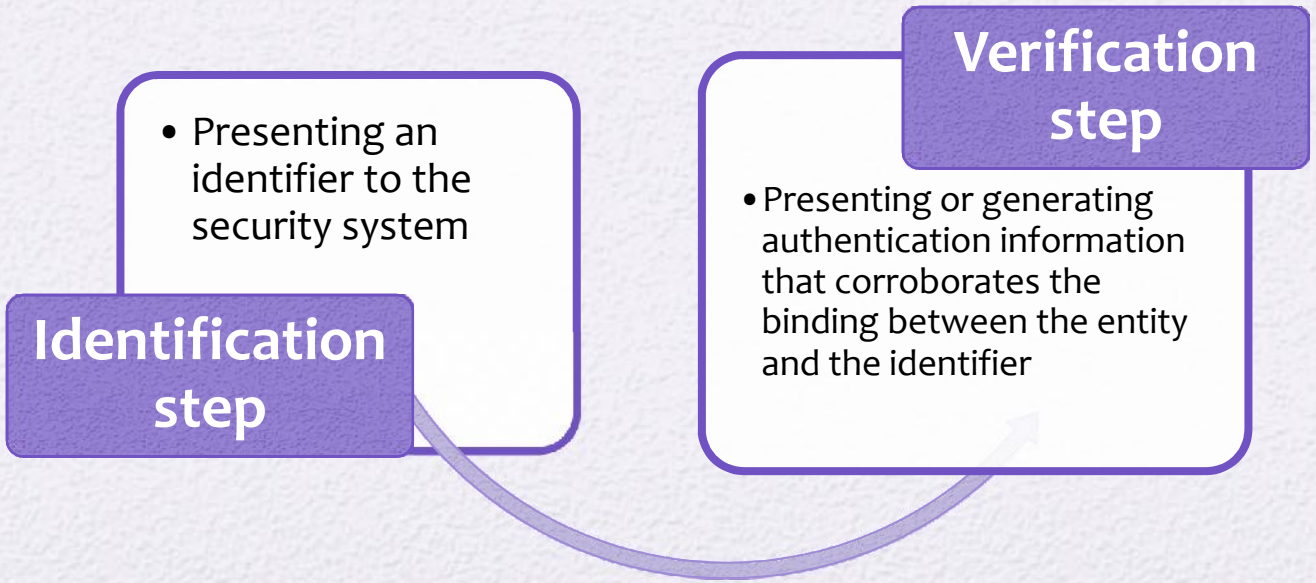
User Authentication

- Authentication functions that have been developed to support network-based user authentication.
- Most widely used authentication services: Kerberos
- user-authentication protocols that rely on asymmetric encryption : X.509 user-authentication protocol

- user authentication is the fundamental building block and the primary line of defense. User authentication is the basis for most types of access control and for user accountability.

Remote User-Authentication Principles

- The process of verifying an identity claimed by or for a system entity
- An authentication process consists of two steps:



- A typical item of authentication information associated with this user ID is a password, which is kept secret (known only to Alice and to the system).
- If no one is able to obtain or guess Alice’s password, then the combination of Alice’s user ID and password enables administrators to set up Alice’s access permissions and audit her activity.

- In essence, identification is the means by which a user provides a claimed identity to the system;
- User authentication is the means of establishing the validity of the claim.
- Note that user authentication is distinct from message authentication.
- message authentication is a procedure that allows communicating parties to verify that the contents of a received message have not been altered and that the source is authentic.

Steps

- The initial requirement for performing user authentication is that the user must be registered with the system.
- Four general means of authenticating a user's identity

Means of User Authentication

Something the individual knows

- Examples include a password, a personal identification number (PIN), or answers to a prearranged set of questions

Something the individual possesses

- Examples include cryptographic keys, electronic keycards, smart cards, and physical keys
  - This is referred to as a token

There are four general means of authenticating a user's identity, which can be used alone or in combination

Something the individual is (static biometrics)

- Examples include recognition by fingerprint, retina, and face

Something the individual does (dynamic biometrics)

- Examples include recognition by voice pattern, handwriting characteristics, and typing rhythm

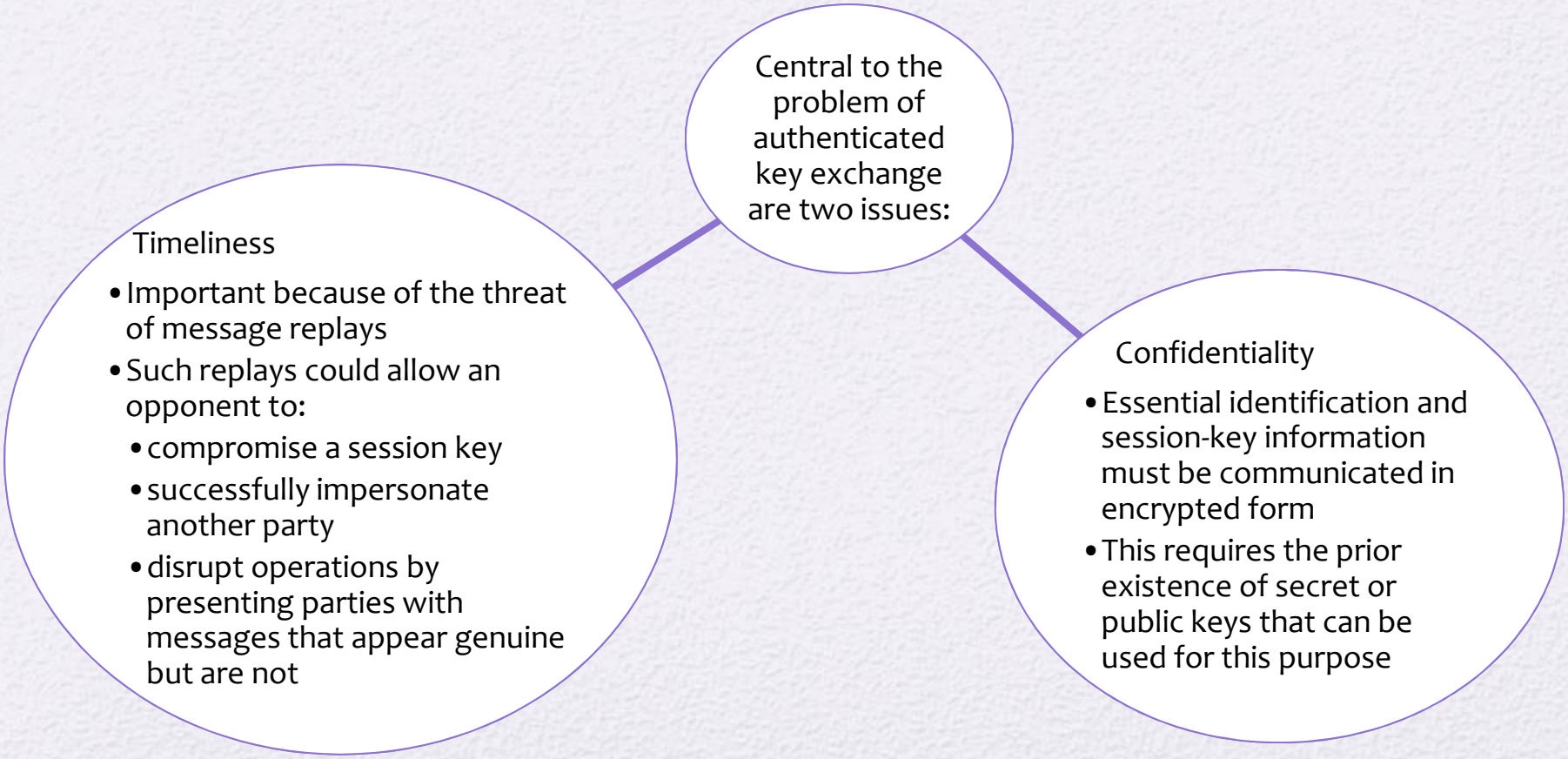
- For network-based user authentication, the most important methods involve cryptographic keys and something the individual knows, such as a password

- All of these methods, properly implemented and used, can provide secure user authentication. However, each method has problems.
- An adversary may be able to guess or steal a password. Similarly, an adversary may be able to forge or steal a token. A user may forget a password or lose a token.

- With respect to biometric authenticators, there are a variety of problems, including dealing with false positives and false negatives, user acceptance, cost, and convenience.
- For network-based user authentication, the most important methods involve cryptographic keys and something the individual knows, such as a password.

Mutual Authentication

- Protocols which enable communicating parties to satisfy themselves mutually about each other's identity and to exchange session keys



Replay Attacks

1. The simplest replay attack is one in which the opponent simply copies a message and replays it later

2. An opponent can replay a timestamped message within the valid time window

3. An opponent can replay a timestamped message within the valid time window, but in addition, the opponent suppresses the original message; thus, the repetition cannot be detected

4. Another attack involves a backward replay without modification and is possible if symmetric encryption is used and the sender cannot easily recognize the difference between messages sent and messages received on the basis of content

Approaches to Coping With Replay Attacks

- Attach a sequence number to each message used in an authentication exchange
  - A new message is accepted only if its sequence number is in the proper order
  - Difficulty with this approach is that it requires each party to keep track of the last sequence number for each claimant it has dealt with
  - Generally not used for authentication and key exchange because of overhead
- Timestamps
  - Requires that clocks among the various participants be synchronized
  - Party A accepts a message as fresh only if the message contains a timestamp that, in A's judgment, is close enough to A's knowledge of current time
- Challenge/response
  - Party A, expecting a fresh message from B, first sends B a nonce (challenge) and requires that the subsequent message (response) received from B contain the correct nonce value

One-Way Authentication

One application for which encryption is growing in popularity is electronic mail (e-mail)

- Header of the e-mail message must be in the clear so that the message can be handled by the store-and-forward e-mail protocol, such as SMTP or X.400
- The e-mail message should be encrypted such that the mail-handling system is not in possession of the decryption key

A second requirement is that of authentication

- The recipient wants some assurance that the message is from the alleged sender

Remote User-Authentication Using Symmetric Encryption

A two-level hierarchy of symmetric keys can be used to provide confidentiality for communication in a distributed environment

- Strategy involves the use of a trusted key distribution center (KDC)
- Each party shares a secret key, known as a master key, with the KDC
- KDC is responsible for generating keys to be used for a short time over a connection between two parties and for distributing those keys using the master keys to protect the distribution

Suppress-Replay Attacks

- The Denning protocol requires reliance on clocks that are synchronized throughout the network
- A risk involved is based on the fact that the distributed clocks can become unsynchronized as a result of sabotage on or faults in the clocks or the synchronization mechanism
- The problem occurs when a sender's clock is ahead of the intended recipient's clock
  - An opponent can intercept a message from the sender and replay it later when the timestamp in the message becomes current at the recipient's site
  - Such attacks are referred to as *suppress-replay attacks*

1. A → KDC: $ID_A \| ID_B \| N_1$
2. KDC → A: $E(K_a, [K_s \| ID_B \| N_1 \| E(K_b, [K_s \| ID_A])])$
3. A → B: $E(K_b, [K_s \| ID_A])$
4. B → A: $E(K_s, N_2)$
5. A → B: $E(K_s, f(N_2))$ where f() is a generic function that modifies the value of the nonce.

With time stamp

1. $A \rightarrow KDC: \quad ID_A \| ID_B$
2. $KDC \rightarrow A: \quad E(K_a, [K_s \| ID_B \| T \| E(K_b, [K_s \| ID_A \| T])])$
3. $A \rightarrow B: \quad E(K_b, [K_s \| ID_A \| T])$
4. $B \rightarrow A: \quad E(K_s, N_1)$
5. $A \rightarrow B: \quad E(K_s, f(N_1))$

Kerberos RFC 1510

- Authentication service developed as part of Project Athena at MIT
- A workstation cannot be trusted to identify its users correctly to network services
  - A user may gain access to a particular workstation and pretend to be another user operating from that workstation
  - 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
  - A user may eavesdrop on exchanges and use a replay attack to gain entrance to a server or to disrupt operations
- Kerberos 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

- Assume an open distributed environment in which users at workstations wish to access services on servers distributed throughout the network.
- Kerberos - In Greek mythology, a many headed dog, commonly three, perhaps with a serpent's tail, the guardian of the entrance of Hades.
- Greek Kerberos has three heads, the modern Kerberos was intended to have three components to guard a network's gate: authentication, accounting, and audit. The last two heads were never implemented.

- Kerberos provides a centralized authentication server whose function is to authenticate users to servers and servers to users.
- Kerberos relies exclusively on symmetric encryption
- Version 4 and 5

Motivation

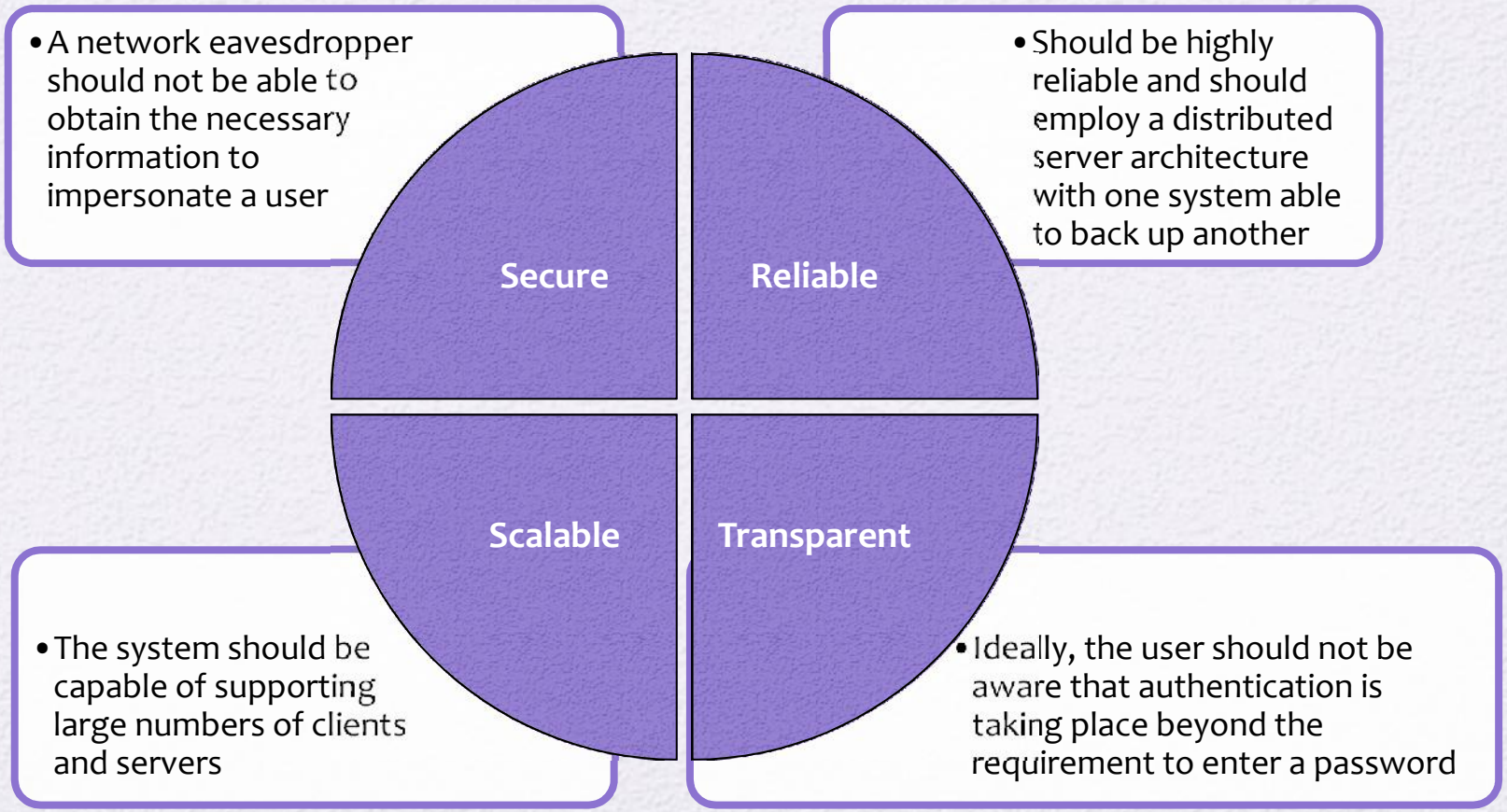
- If a set of users is provided with dedicated personal computers that have no network connections, then a user's resources and files can be protected by physically securing each personal computer. When these users instead are served by a centralized time-sharing system, the time-sharing operating system must provide the security.
- The operating system can enforce access-control policies based on user identity and use the logon procedure to identify users.

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 identification (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 his or her identity for each service invoked. Also require that servers prove their identity to clients.

- Kerberos supports this third approach. Kerberos assumes a distributed client/server architecture and employs one or more Kerberos servers to provide an authentication service.

Kerberos Requirements

- The first published report on Kerberos listed the following requirements:



Kerberos Version 4

- Makes use of DES to provide the authentication service
- Authentication server (AS)
  - Knows the passwords of all users and stores these in a centralized database
  - Shares a unique secret key with each server
- Ticket
  - Created once the AS accepts the user as authentic; contains the user's ID and network address and the server's ID
  - Encrypted using the secret key shared by the AS and the server
- Ticket-granting server (TGS)
  - Issues tickets to users who have been authenticated to AS
  - Each time the user requires access to a new service the client applies to the TGS using the ticket to authenticate itself
  - The TGS then grants a ticket for the particular service
  - The client saves each service-granting ticket and uses it to authenticate its user to a server each time a particular service is requested

$$\textbf{(1)}\ \mathrm{C} \rightarrow \mathrm{AS}: \quad ID_C \| P_C \| ID_V$$
$$\textbf{(2)}\ \mathrm{AS} \rightarrow \mathrm{C}: \quad Ticket$$
$$\textbf{(3)}\ \mathrm{C} \rightarrow \mathrm{V}: \quad ID_C \| Ticket$$
$$Ticket = E(K_v, [ID_C \| AD_C \| ID_V])$$

where

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
AD_C = network address of C
K_v = secret encryption key shared by AS and V

In this scenario, the user logs on to a workstation and requests access to server V. The client module C in the user's workstation requests the user's password and then sends a message to the AS that includes the user's ID, the server's ID, and the user's password. The AS checks its database to see if the user has supplied the proper password for this user ID and whether this user is permitted access to server V. If both tests are passed, the AS accepts the user as authentic and must now convince the server that this user is authentic. To do so, the AS creates a **ticket** that contains the user's ID and network address and the server's ID. This ticket is encrypted using the secret key shared by the AS and this server. This ticket is then sent back to C. Because the ticket is encrypted, it cannot be altered by C or by an opponent.

With this ticket, C can now apply to V for service. C sends a message to V containing C's ID and the ticket. V decrypts the ticket and verifies that the user ID in the ticket is the same as the unencrypted user ID in the message. If these two match, the server considers the user authenticated and grants the requested service.

The Version 4 Authentication Dialogue

The lifetime associated with the ticket-granting ticket creates a problem:

- If the lifetime is very short (e.g., minutes), the user will be repeatedly asked for a password
- If the lifetime is long (e.g., hours), then an opponent has a greater opportunity for replay

A network service (the TGS or an application service) must be able to prove that the person using a ticket is the same person to whom that ticket was issued

Servers need to authenticate themselves to users

Once per user logon session:

(1) $C \rightarrow AS$: $ID_C \| ID_{tgs}$

(2) $AS \rightarrow C$: $E(K_c, Ticket_{tgs})$

Once per type of service:

(3) $C \rightarrow TGS$: $ID_C \| ID_V \| Ticket_{tgs}$

(4) $TGS \rightarrow C$: $Ticket_v$

Once per service session:

(5) $C \rightarrow V$: $ID_C \| Ticket_v$

$$Ticket_{tgs} = E(K_{tgs}, [ID_C \| AD_C \| ID_{tgs} \| TS_1 \| Lifetime_1])$$

$$Ticket_v = E(K_v, [ID_C \| AD_C \| ID_v \| TS_2 \| Lifetime_2])$$

Table 15.1 (page 464 in textbook)

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

Key Establishment Using a Simplified Version of Kerberos

Alice		**KDC**		**Bob**
KEK: k_A		KEK: k_A, k_B		KEK: k_B
generate nonce r_A				
	$\xrightarrow{\text{RQST}(ID_A, ID_B, r_A)}$			
		generate random k_{ses} generate lifetime T $y_A = e_{k_A}(k_{ses}, r_A, T, ID_B)$ $y_B = e_{k_B}(k_{ses}, ID_A, T)$		
	$\xleftarrow{y_A, y_B}$			
$k_{ses}, r'_A, T, ID_B = e^{-1}_{k_A}(y_A)$ verify $r'_A = r_A$ verify ID_B verify lifetime T generate time stamp T_S $y_{AB} = e_{k_{ses}}(ID_A, T_S)$				
	$\xrightarrow{y_{AB}, y_B}$			
				$k_{ses}, ID_A, T = e^{-1}_{k_B}(y_B)$ $ID_A', T_S = e^{-1}_{k_{ses}}(y_{AB})$ verify $ID_A' = ID_A$ verify lifetime T verify time stamp T_S
$y = e_{k_{ses}}(x)$	$\xrightarrow{y}$			$x = e^{-1}_{k_{ses}}(y)$

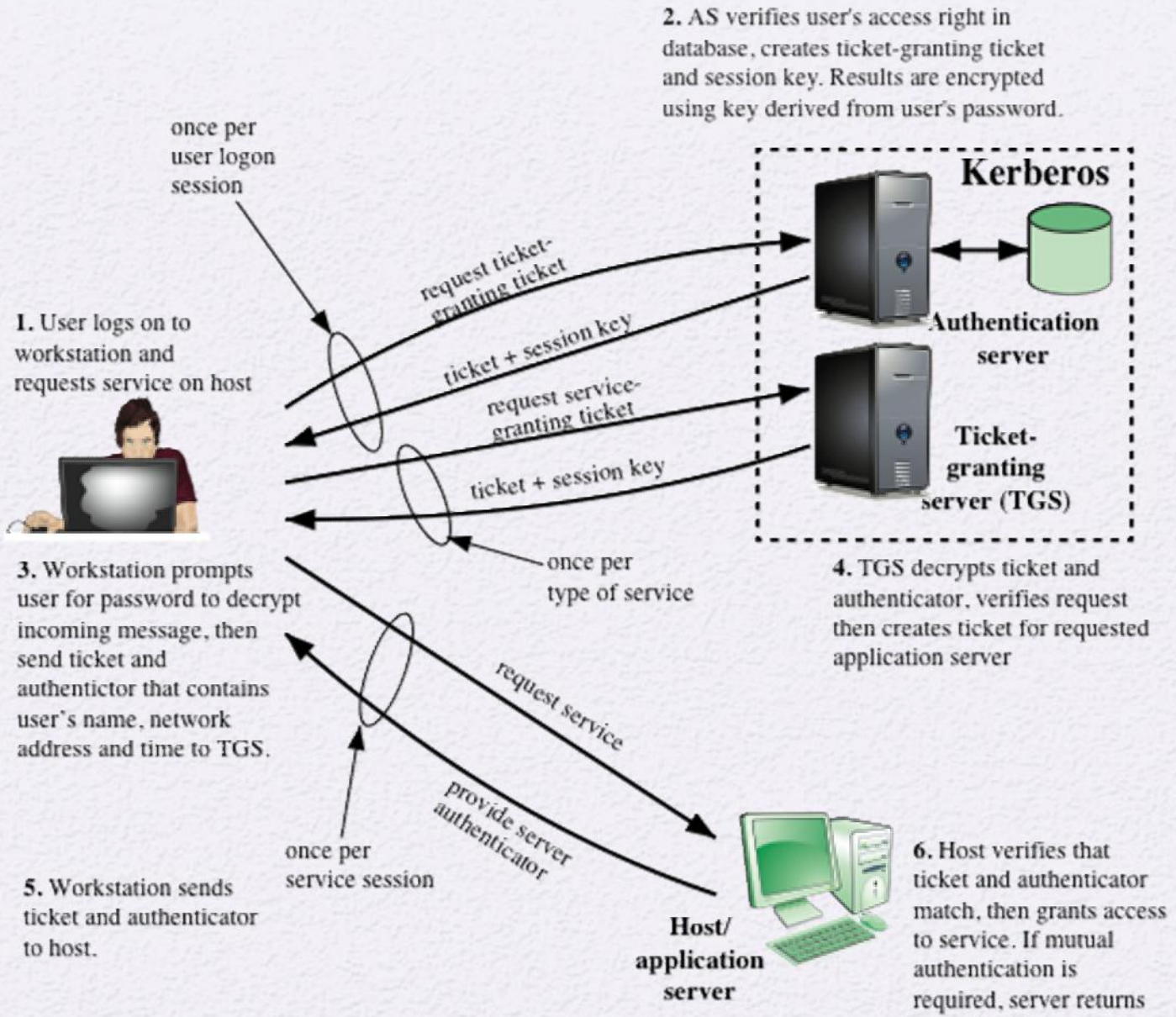


Figure 15.1 Overview of Kerberos

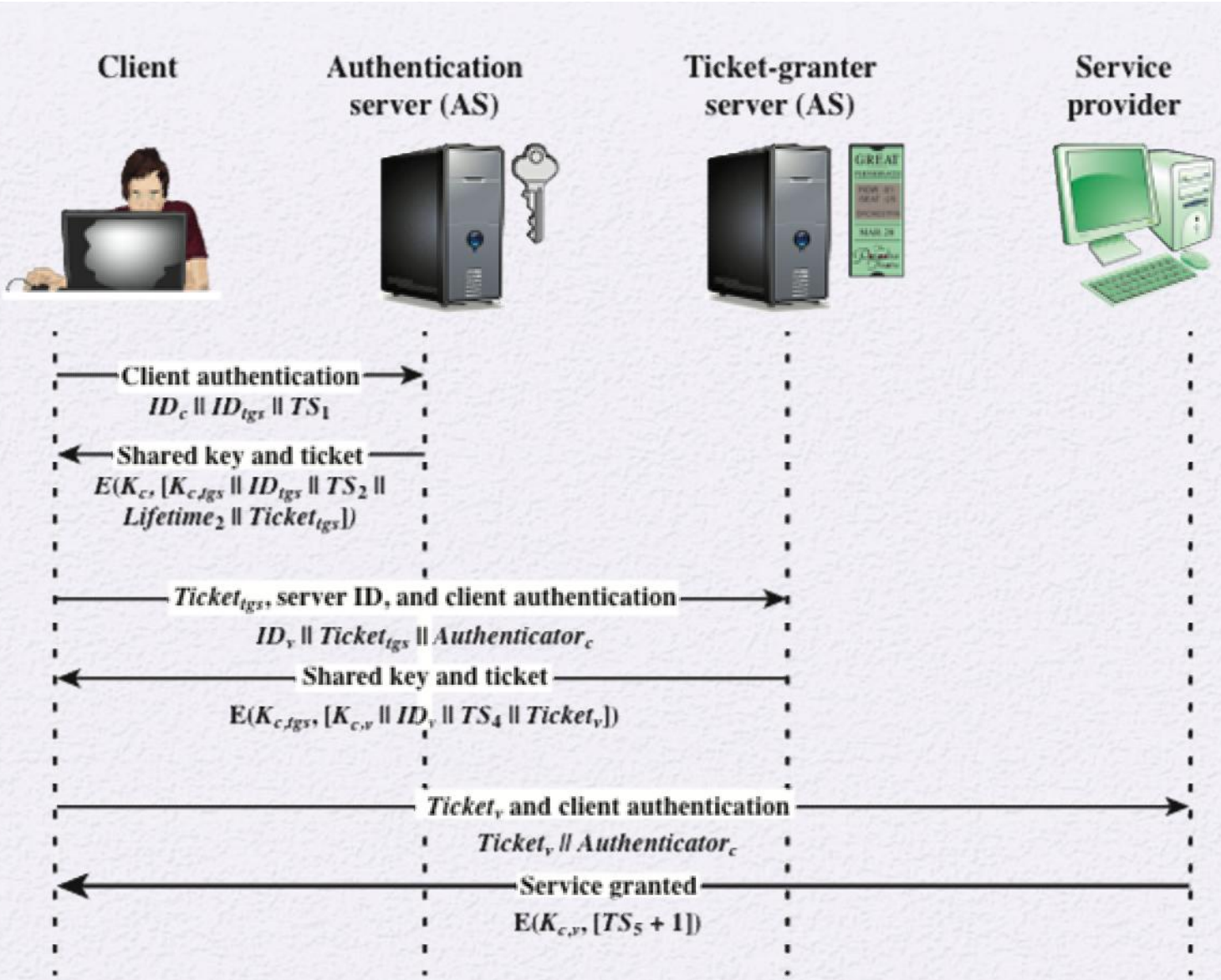


Figure 15.2 Kerberos Exchanges

Table 15.2 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.

(This table can be found on pages 467 – 468 in the textbook)

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.

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.

Kerberos Realms and Multiple Kerberi

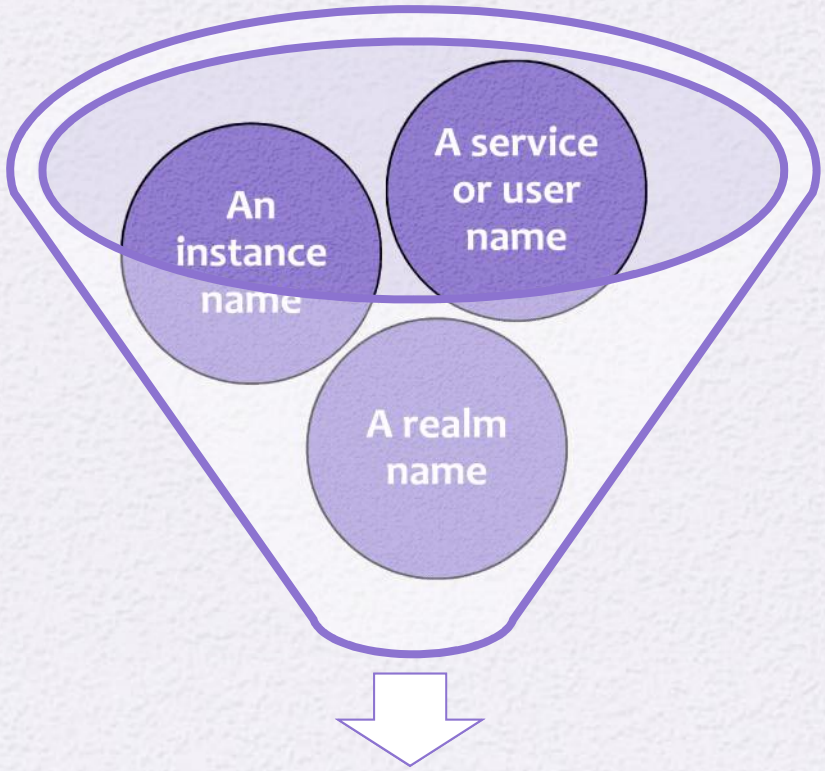
- A full-service Kerberos environment consisting of a Kerberos server, a number of clients, and a number of application servers requires that:
  - The Kerberos server must have the user ID and hashed passwords of all participating users in its database; all users are registered with the Kerberos server
  - The Kerberos server must share a secret key with each server; all servers are registered with the Kerberos server
  - The Kerberos server in each interoperating realm shares a secret key with the server in the other realm; the two Kerberos servers are registered with each other

Kerberos Realm

- A set of managed nodes that share the same Kerberos database
- The 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
- Identified by its principal name



Three parts of a principal name

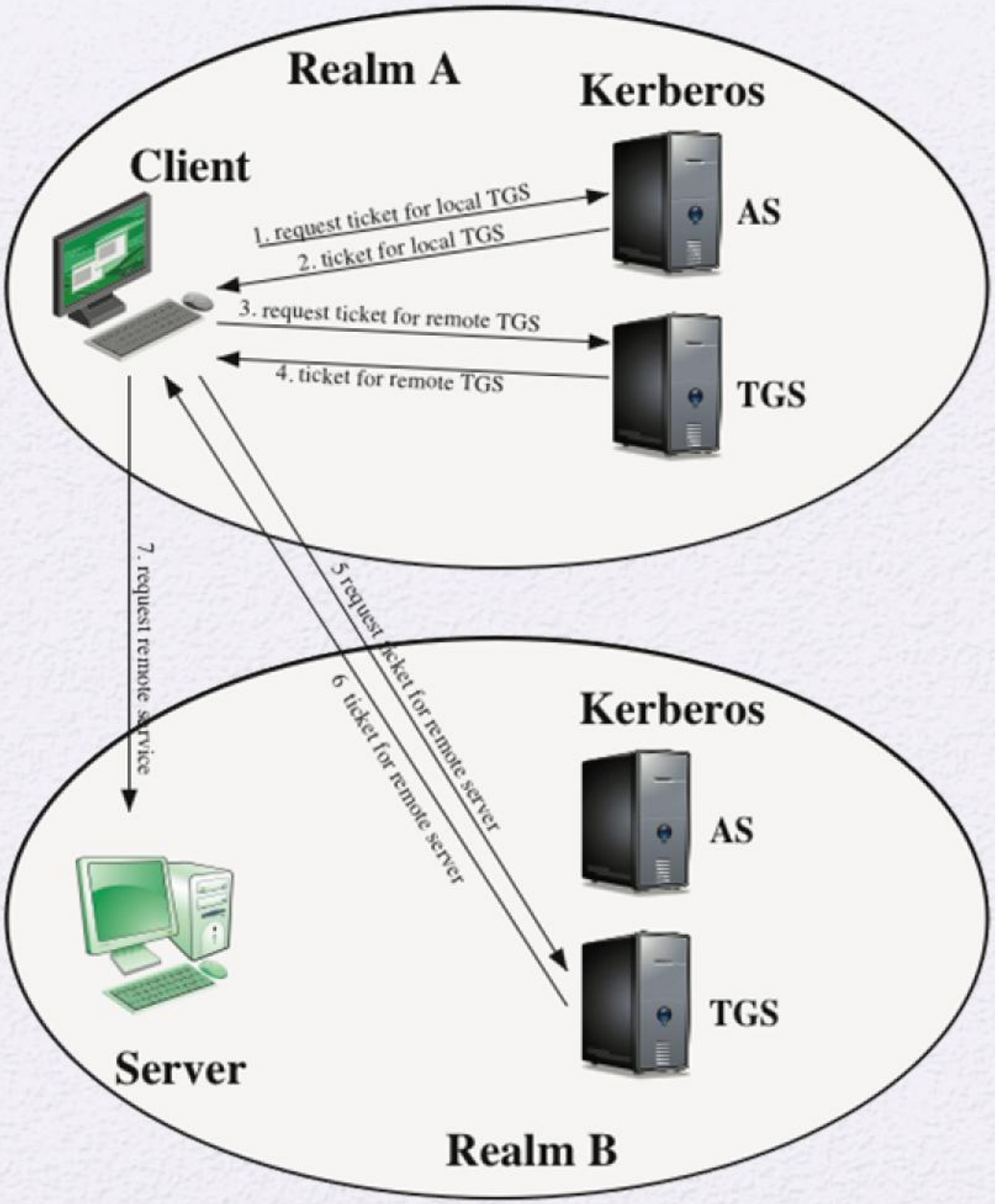


Figure 15.3 Request for Service in Another Realm

Differences Between Versions 4 and 5

Version 5 is intended to address the limitations of version 4 in two areas:

Environmental shortcomings

- Encryption system dependence
- Internet protocol dependence
- Message byte ordering
- Ticket lifetime
- Authentication forwarding
- Interrealm authentication

Technical deficiencies

- Double encryption
- PCBC encryption
- Session keys
- Password attacks

Table 15.3
Summary of Kerberos Version 5 Message Exchanges

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

(2) AS → C $Realmc \| IDC \| Ticket_{tgs} \| E(Kc, [Kc,tgs \| Times \| Nonce1 \| Realm_{tgs} \| IDtgs])$

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

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

(3) C → TGS $Options \| IDv \| Times \| \| Nonce2 \| Ticket_{tgs} \| Authenticator_c$

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

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

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

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

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

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

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

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

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

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

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.

Table 15.4

Kerberos Version 5 Flags

(Table can be found on page 474 in textbook)

Mutual Authentication

- Public-key encryption for session key distribution
  - Assumes each of the two parties is in possession of the current public key of the other
  - May not be practical to require this assumption
- Denning protocol using timestamps
  - Uses an authentication server (AS) to provide public-key certificates
  - Requires the synchronization of clocks
- Woo and Lam makes use of nonces
  - Care needed to ensure no protocol flaws

One-Way Authentication

- Have public-key approaches for e-mail
  - Encryption of message for confidentiality, authentication, or both
  - The public-key algorithm must be applied once or twice to what may be a long message
- For confidentiality encrypt message with one-time secret key, public-key encrypted
- If authentication is the primary concern, a digital signature may suffice

Federated Identity Management

- Relatively new concept dealing with the use of a common identity management scheme across multiple enterprise and numerous applications and supporting many users

- Services provided include:
  - Point of contact
  - SSO protocol services
  - Trust services
  - Key services
  - Identity services
  - Authorization
  - Provisioning
  - Management

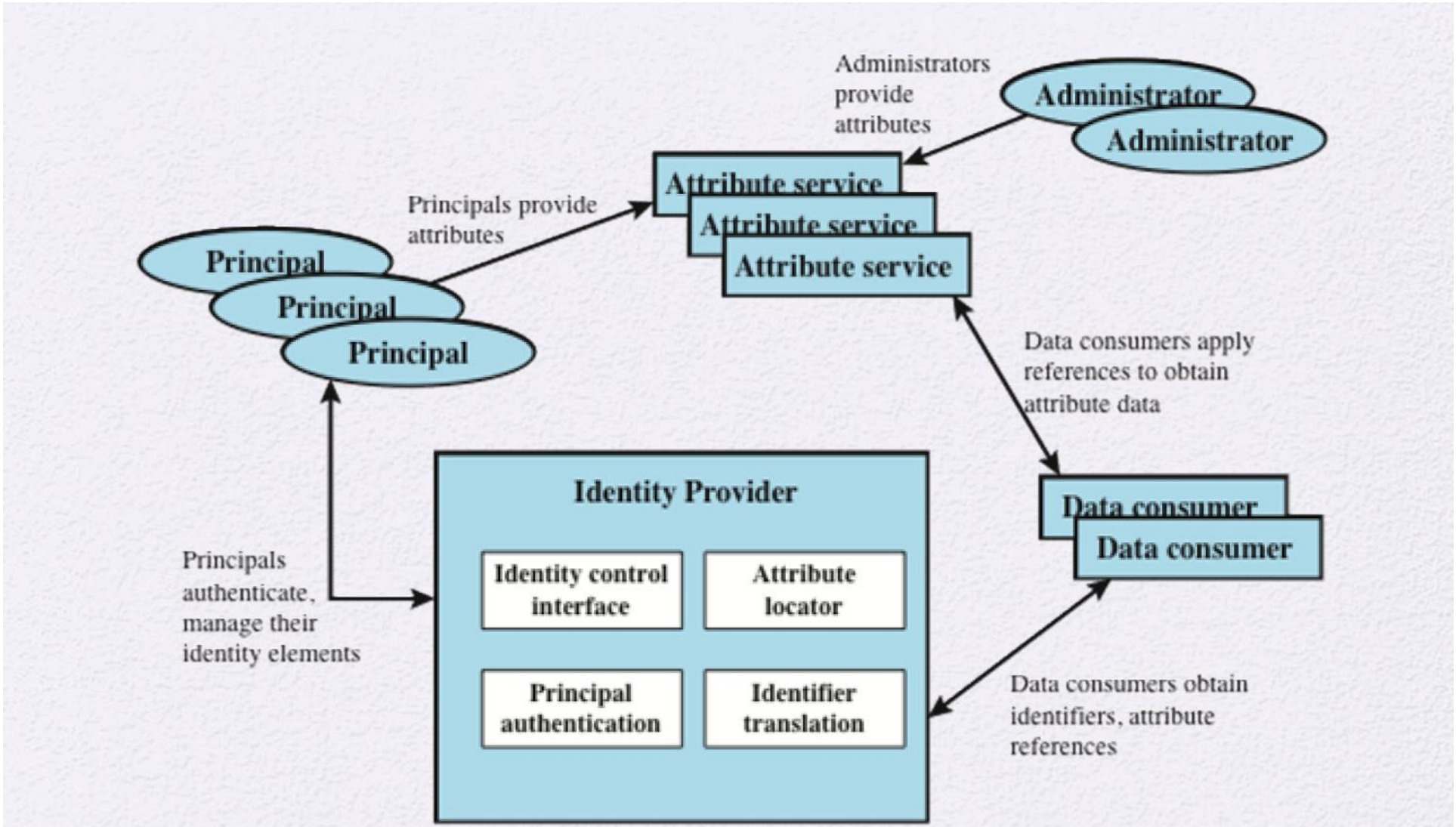
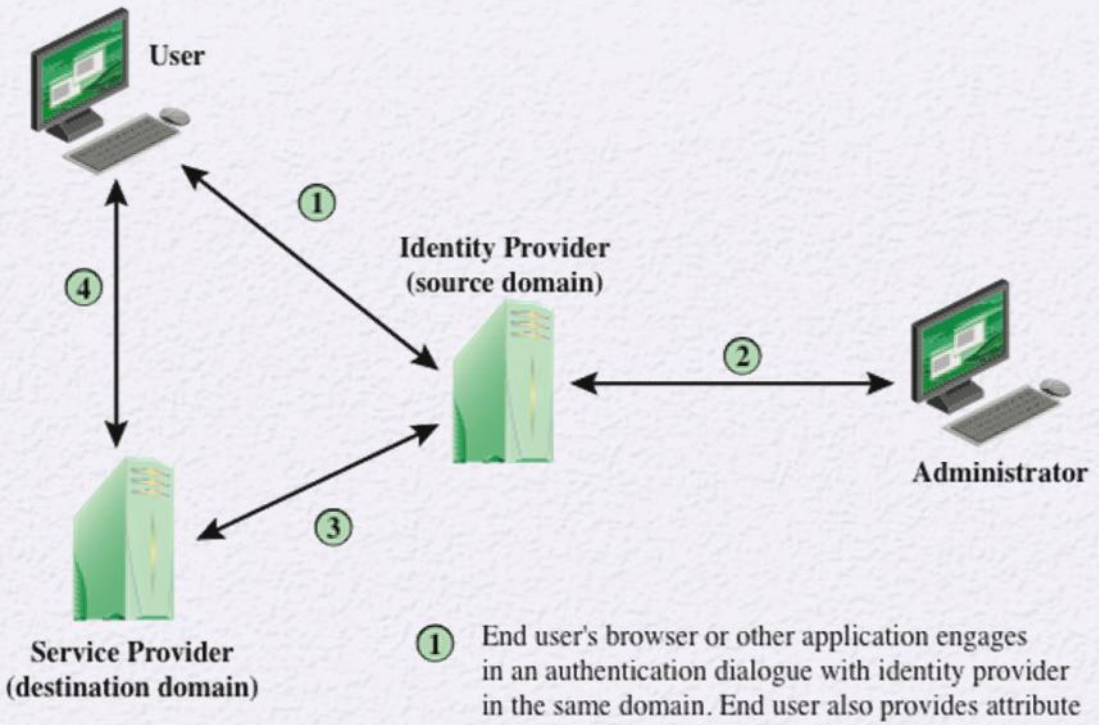


Figure 15.4 Generic Identity Management Architecture



1. End user's browser or other application engages in an authentication dialogue with identity provider in the same domain. End user also provides attribute values associated with user's identity.

2. Some attributes associated with an identity, such as allowable roles, may be provided by an administrator in the same domain.

3. A service provider in a remote domain, which the user wishes to access, obtains identity information, authentication information, and associated attributes from the identity provider in the source domain.

4. Service provider opens session with remote user and enforces access control restrictions based on user's identity and attributes.

Figure 15.5 Federated Identity Operation

Key Standards

The Extensible Markup Language (XML)	The Simple Object Access Protocol (SOAP)	WS-Security	Security Assertion Markup Language (SAML)
A markup language that uses sets of embedded tags or labels to characterize text elements within a document so as to indicate their appearance, function, meaning, or context	Enables applications to request services from one another with XML-based requests and receive responses as data formatted with XML	A set of SOAP extensions for implementing message integrity and confidentiality in Web services	An XML-based language for the exchange of security information between online business partners

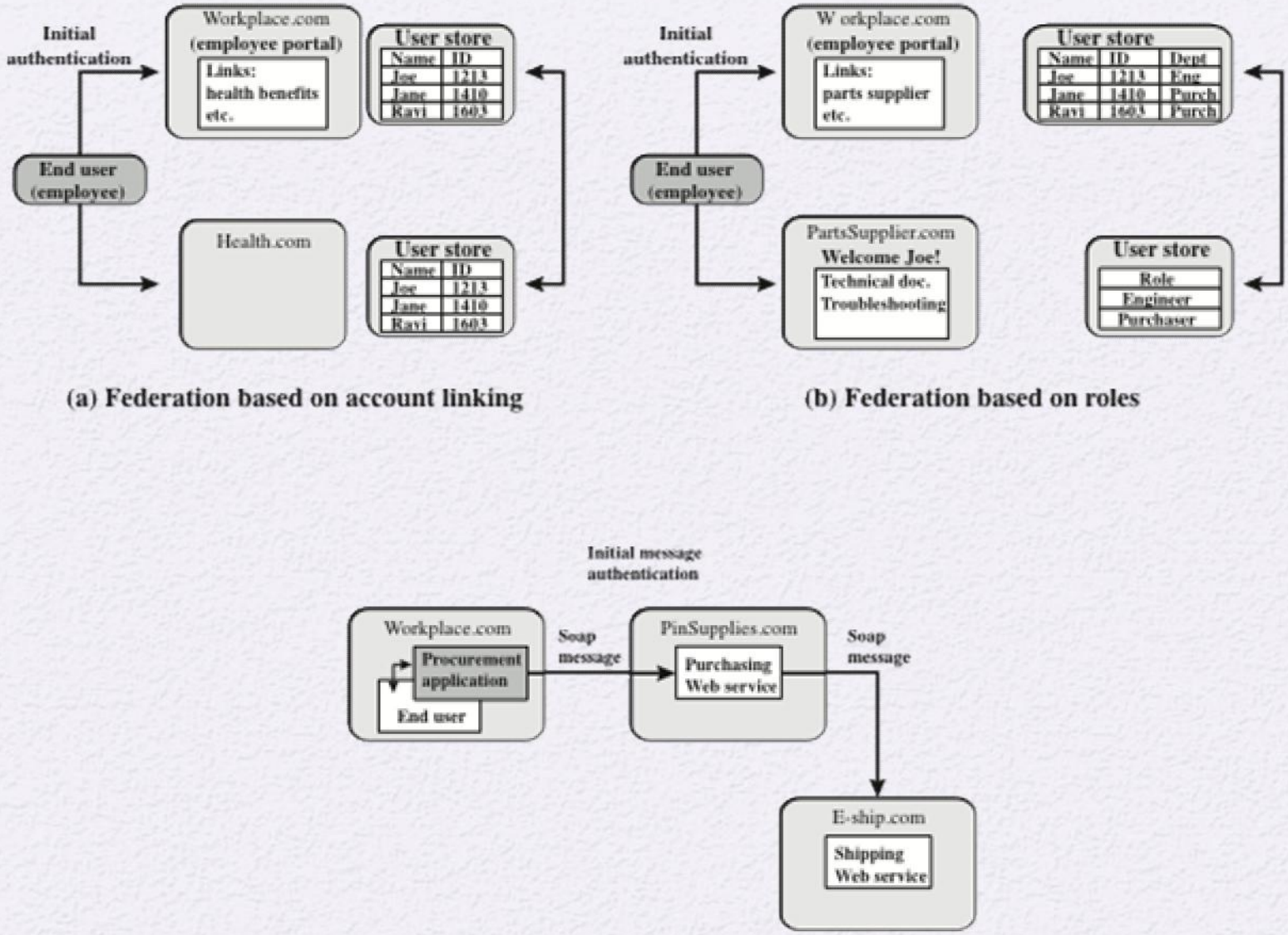


Figure 15.6 Federated Identity Scenarios

Personal Identity Verification

- User authentication based on the possession of a smart card is becoming more widespread
  - Has the appearance of a credit card
  - Has an electronic interface
  - May use a variety of authentication protocols
- A smart card contains within it an entire microprocessor, including processor, memory, and I/O ports
- A smart card includes three types of memory:
  - Read-only memory (ROM) stores data that does not change during the card's life
  - Electronically erasable programmable ROM (EEPROM) holds application data and programs; also holds data that may vary with time
  - Random access memory (RAM) holds temporary data generated when applications are executed

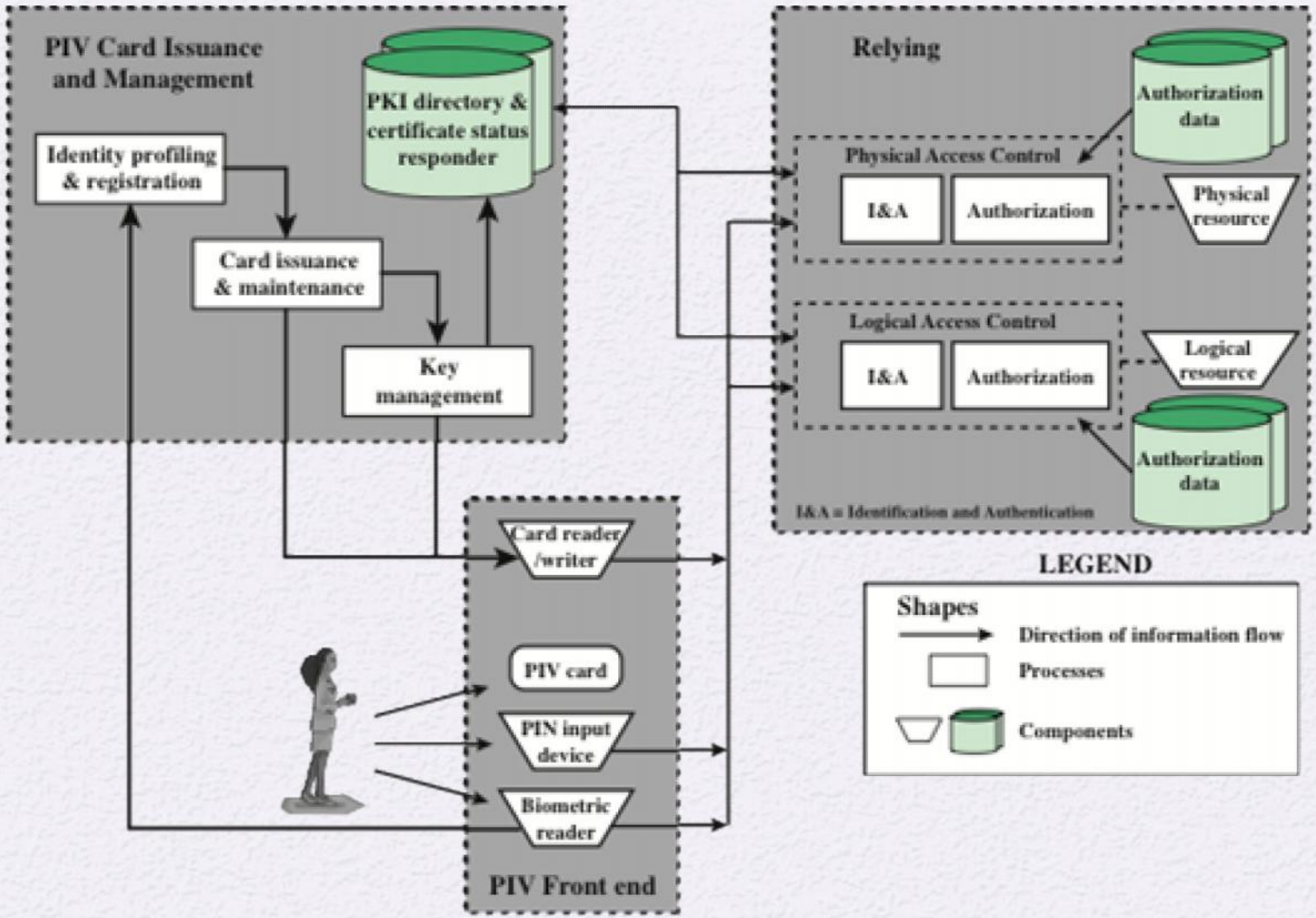


Figure 15.7 FIPS 201 PIV System Model

PIV Documentation

- **FIPS 201-2—Personal Identity Verification (PIV) of Federal Employees and Contractors**
  - Specifies the physical card characteristics, storage media, and data elements that make up the identity credentials resident on the PIV card
- **SP 800-73-3—Interfaces for Personal Identity Verification**
  - Specifies the interfaces and card architecture for storing and retrieving identity credentials from a smart card, and provides guidelines for the use of authentication mechanisms and protocols
- **SP 800-76-2—Biometric Data Specification for Personal Identity Verification**
  - Describes technical acquisition and formatting specifications for the biometric credentials of the PIV system
- **SP 800-78-3—Cryptographic Algorithms and Key Sizes for Personal Identity Verification**
  - Identifies acceptable symmetric and asymmetric encryption algorithms, digital signature algorithms, and message digest algorithms, and specifies mechanisms to identify the algorithms associated with PIV keys or digital signatures
- **SP 800-104—A Scheme for PIV Visual Card Topography**
  - Provides additional recommendations on the PIV card color-coding for designating employee affiliation
- **SP 800-116—A Recommendation for the Use of PIV Credentials in Physical Access Control Systems (PACS)**
  - Describes a risk-based approach for selecting appropriate PIV authentication mechanisms to manage physical access to Federal government facilities and assets
- **SP 800-79-1—Guidelines for the Accreditation of Personal Identity Verification Card Issuers**
  - Provides guidelines for accrediting the reliability of issuers of PIV cards that collect, store, and disseminate personal identity credentials and issue smart cards
- **SP 800-96—PIV Card to Reader Interoperability Guidelines**
  - Provides requirements that facilitate interoperability between any card and any reader

PIV Credentials and Keys

- **Personal Identification Number (PIN)**
  - Required to activate the card for privileged operation
- **Cardholder Unique Identifier (CHUID)**
  - Includes the Federal Agency Smart Credential Number (FASC-N) and the Global Unique Identification Number (GUID), which uniquely identify the card and the cardholder
- **PIV Authentication Key**
  - Asymmetric key pair and corresponding certificate for user authentication
- **Two fingerprint templates**
  - For biometric authentication
- **Electronic facial image**
  - For biometric authentication
- **Asymmetric Card Authentication Key**
  - Asymmetric key pair and corresponding certificate used for card authentication
- **Optional elements include the following:**
- **Digital Signature Key**
  - Asymmetric key pair and corresponding certificate that supports document signing and signing of data elements such as the CHUID
- **Key Management Key**
  - Asymmetric key pair and corresponding certificate supporting key establishment and transport
- **Symmetric Card Authentication Key**
  - For supporting physical access applications
- **PIV Card Application Administration Key**
  - Symmetric key associated with the card management system
- **One or two iris images**
  - For biometric authentication

Table 15.5
PIV Algorithms and Key Sizes

PIV Key Type	Algorithms	Key Sizes (bits)	Application
PIV Authentication Key	RSA	2048	Supports card and cardholder authentication for an interoperable environment.
	ECDSA	256	
Card Authentication Key	3TDEA	168	Supports card authentication for physical access
	AES	128, 192, or 256	
	RSA	2048	Supports card authentication for an interoperable environment.
	ECDSA	256	
Digital Signature Key	RSA	2048 or 3072	Supports document signing and nonce signing
	ECDSA	256 or 384	
Key Management Key	RSA	2048	Supports key establishment and transport.
	ECDH	256 or 384	

Authentication

- **Using the electronic credentials resident on a PIV card, the card supports the following authentication mechanisms:**

  - **CHUID**

    The cardholder is authenticated using the signed CHUID data element on the card. The PIN is not required. This mechanism is useful in environments where a low level of assurance is acceptable and rapid contactless authentication is necessary

  - **Card Authentication Key**

    The PIV card is authenticated using the Card Authentication Key in a challenge response protocol. The PIN is not required. This mechanism allows contact (via card reader) or contactless (via radio waves) authentication of the PIV card without the holder's active participation, and provides a low level of assurance

  - **BIO**

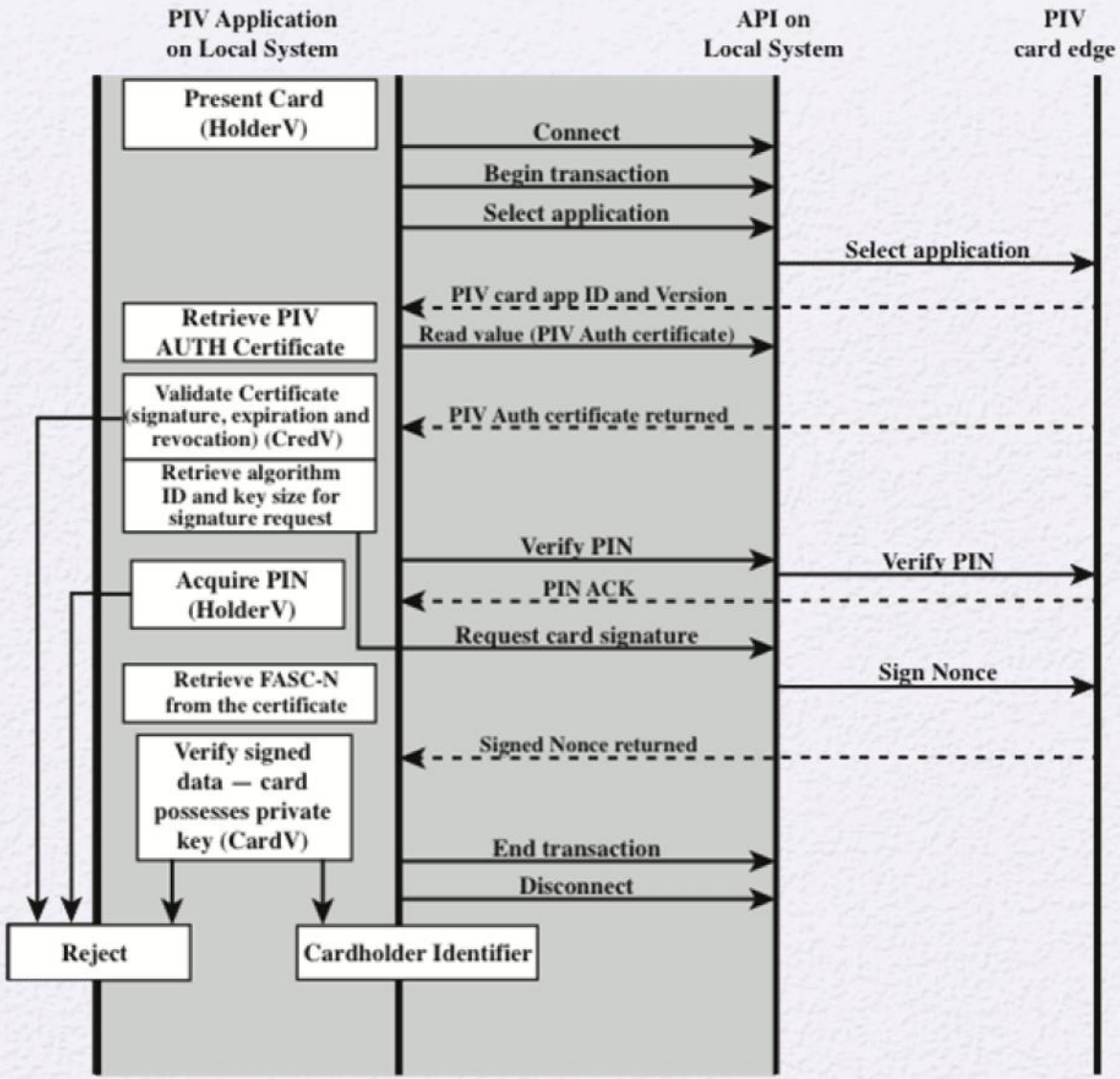
    The cardholder is authenticated by matching his or her fingerprint sample(s) to the signed biometric data element in an environment without a human attendant in view. The PIN is required to activate the card. This mechanism achieves a high level of assurance and requires the cardholder's active participation is submitting the PIN as well as the biometric sample

  - **BIO-A**

    The cardholder is authenticated by matching his or her fingerprint sample(s) to the signed biometric data element in an environment with a human attendant in view. The PIN is required to activate the card. This mechanism achieves a very high level of assurance when coupled with full trust validation of the biometric template retrieved from the card, and requires the cardholder's active participation is submitting the PIN as well as the biometric sample

  - **PKI**

    The cardholder is authenticated by demonstrating control of the PIV authentication private key in a challenge response protocol that can be validated using the PIV authentication certificate. The PIN is required to activate the card. This mechanism achieves a very high level of identity assurance and requires the cardholder's knowledge of the PIN



CardV = Card validation
CredV = Credential validation
HolderV = Cardholder validation
FASC-N = Federal Agency Smart Credential Number

Figure 15.8 Authentication using PIV Authentication Key

Summary

- Remote user-authentication principles
  - Mutual authentication
  - One-way authentication
- Remote user-authentication using symmetric encryption
  - Mutual authentication
  - One-way authentication
- Kerberos
  - Motivation
  - Kerberos V4 and V5

- Remote user-authentication using asymmetric encryption
  - Mutual authentication
  - One-way authentication
- Federated identity management
  - Identity management
  - Identity federation
- Personal identity verification
  - PIV system model
  - PIV documentation
  - PIV credentials and keys
  - authentication