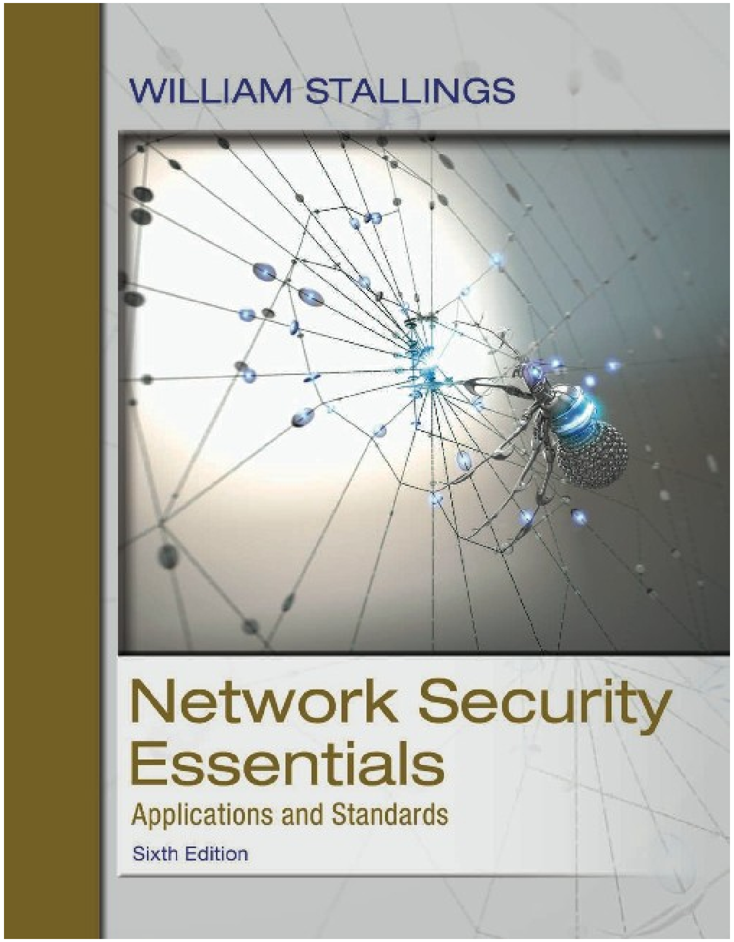
Network Security Essentials: Applications and Standards

Sixth Edition



Chapter 4

Key Distribution and User Authentication

Pearson

Remote User Authentication Principles (1 of 2)

- In most computer security contexts, 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
- RFC 4949 (Internet Security Glossary) defines user authentication as the process of verifying an identity claimed by or for a system entity

Remote User Authentication Principles (2 of 2)

- Identification step

  Presenting an identifier to the security system

- Verification step

  Presenting or generating authentication information that corroborates the binding between the entity and the identifier

Pearson

NIST Model for Electronic User Authentication

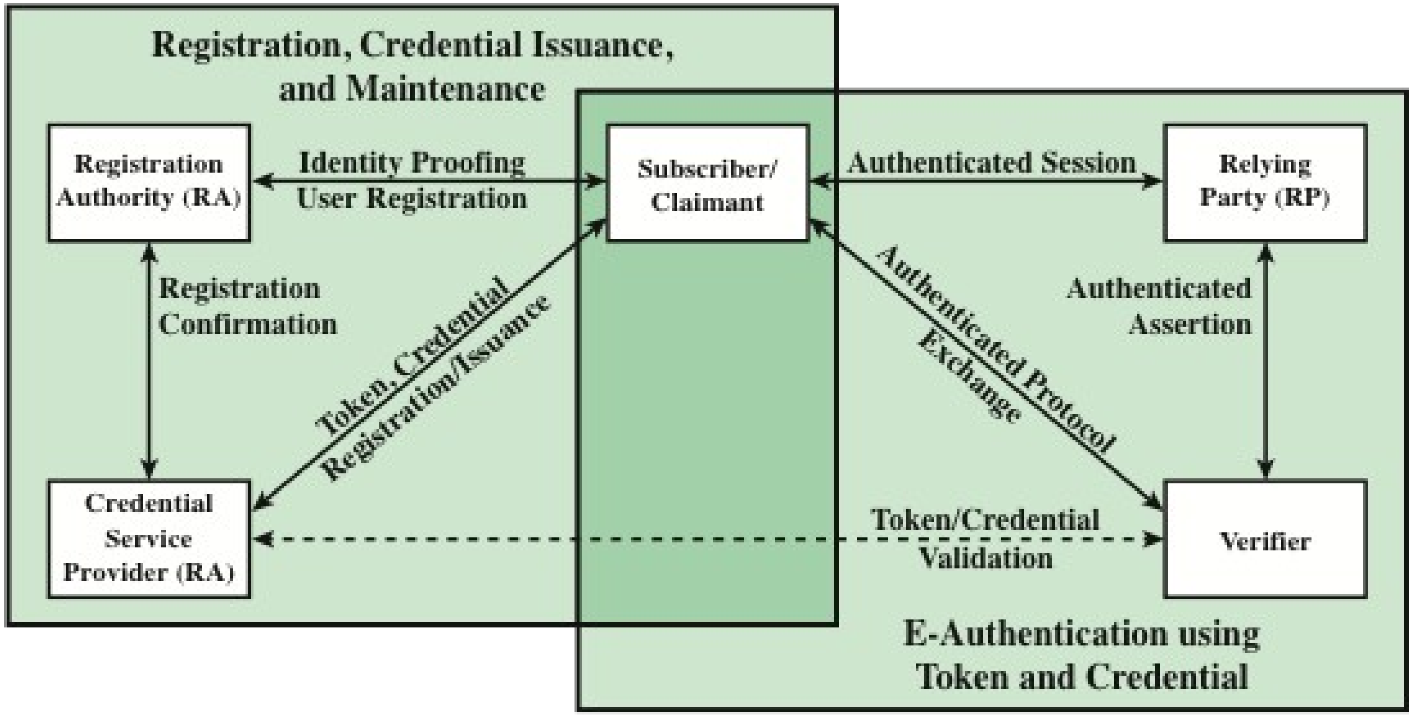
- NIST SP 800-63-2 (Electronic Authentication Guideline, August 2013 defines electronic user authentication as the process of establishing confidence in user identities that are presented electronically to an information system

- Systems can use the authenticated identity to determine if the authenticated individual is authorized to perform particular functions

- In many cases, the authentication and transaction or other authorized function take place across an open network such as the Internet

- Equally, authentication and subsequent authorization can take place locally, such as across a local area network

Figure 4-1: The NIST SP 800-63-2 E-Authentication Architectural Model


Registration, Credential Issuance, and Maintenance
Registration Authority (RA)
Identity Proofing
User Registration
Subscriber/ Claimant
Authenticated Session
Relying Party (RP)
Registration Confirmation
Token, Credential Registration/Issuance
Authenticated Protocol Exchange
Authenticated Assertion
Credential Service Provider (RA)
Token/Credential Validation
Verifier
E-Authentication using Token and Credential

Means of Authentication (1 of 2)

- There are four general means of authenticating a user’s identity, which can be used alone or in combination
  - 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 type of authenticator is referred to as a token

Means of Authentication (2 of 2)

- 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

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

Kerberos (1 of 2)

- 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

Kerberos (2 of 2)

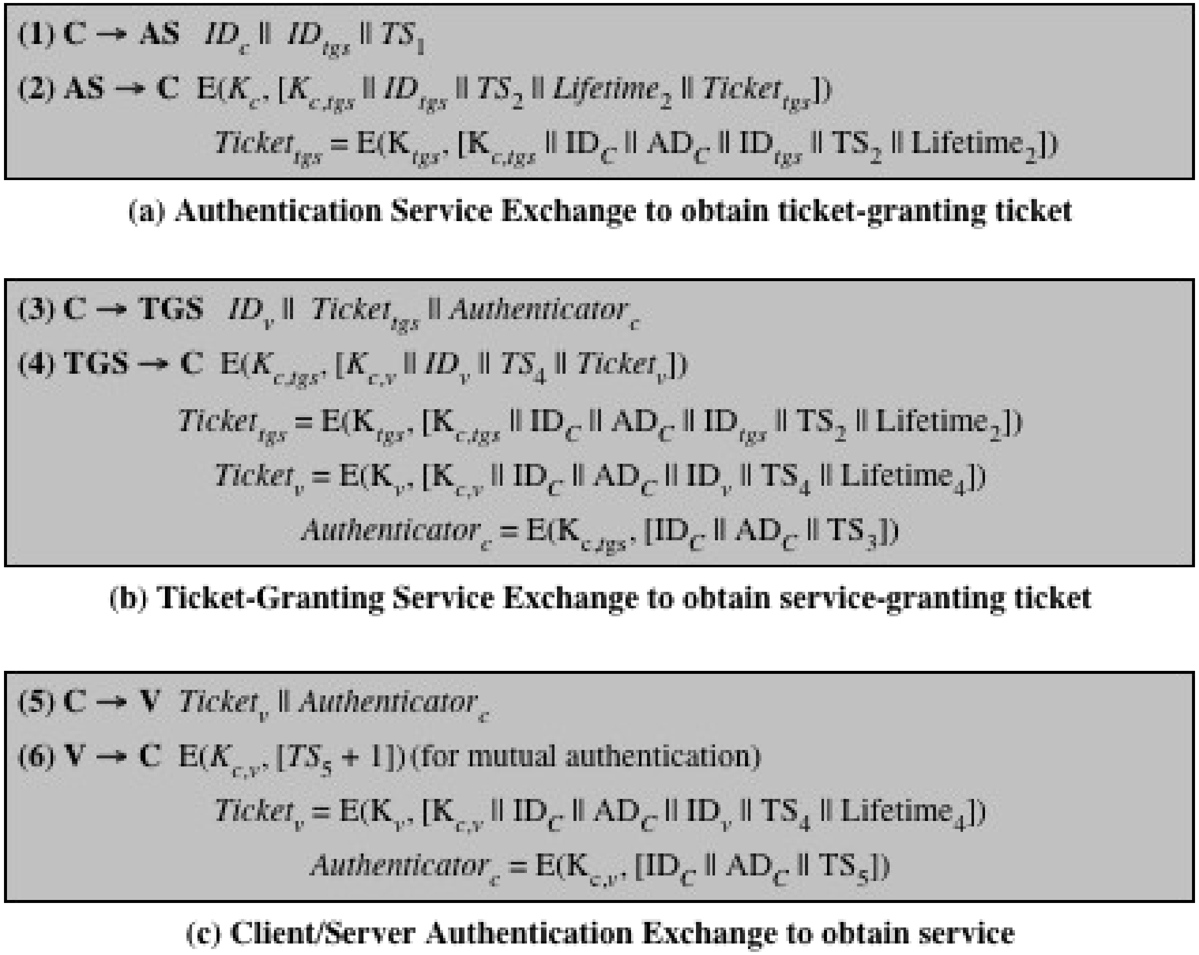
Two versions are in use

- 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
  - 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** $\mathit{ID}_c \| \mathit{ID}_{tgs} \| \mathit{TS}_1$

(2) **AS → C** $E(K_c, [K_{c,tgs} \| \mathit{ID}_{tgs} \| \mathit{TS}_2 \| \mathit{Lifetime}_2 \| \mathit{Ticket}_{tgs}])$

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

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

(3) **C → TGS** $\mathit{ID}_v \| \mathit{Ticket}_{tgs} \| \mathit{Authenticator}_c$

(4) **TGS → C** $E(K_{c,tgs}, [K_{c,v} \| \mathit{ID}_v \| \mathit{TS}_4 \| \mathit{Ticket}_v])$

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

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

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

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

(5) **C → V** $\mathit{Ticket}_v \| \mathit{Authenticator}_c$

(6) **V → C** $E(K_{c,v}, [\mathit{TS}_5 + 1])$ (for mutual authentication)

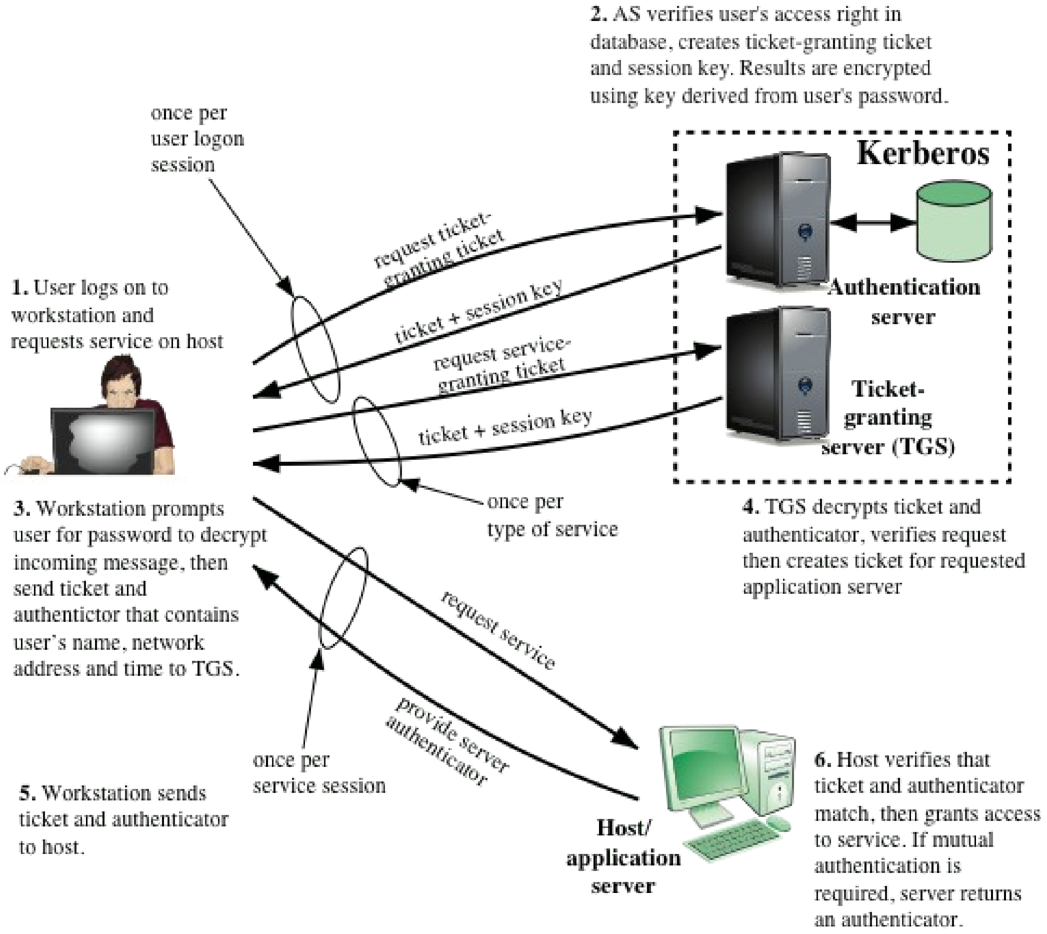
$\mathit{Ticket}_v = E(K_v, [K_{c,v} \| ID_C \| AD_C \| ID_v \| TS_4 \| Lifetime_4])$

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

(c) Client/Server Authentication Exchange to obtain service

Pearson

Figure 4-2: Overview of Kerberos



1. User logs on to workstation and requests service on host

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.

3. Workstation prompts user for password to decrypt incoming message, then send ticket and authentictor that contains user's name, network address and time to TGS.

4. TGS decrypts ticket and authenticator, verifies request then creates ticket for requested application server

5. Workstation sends ticket and authenticator to host.

6. Host verifies that ticket and authenticator match, then grants access to service. If mutual authentication is required, server returns an authenticator.

Figure 4-3: Kerberos Exchanges

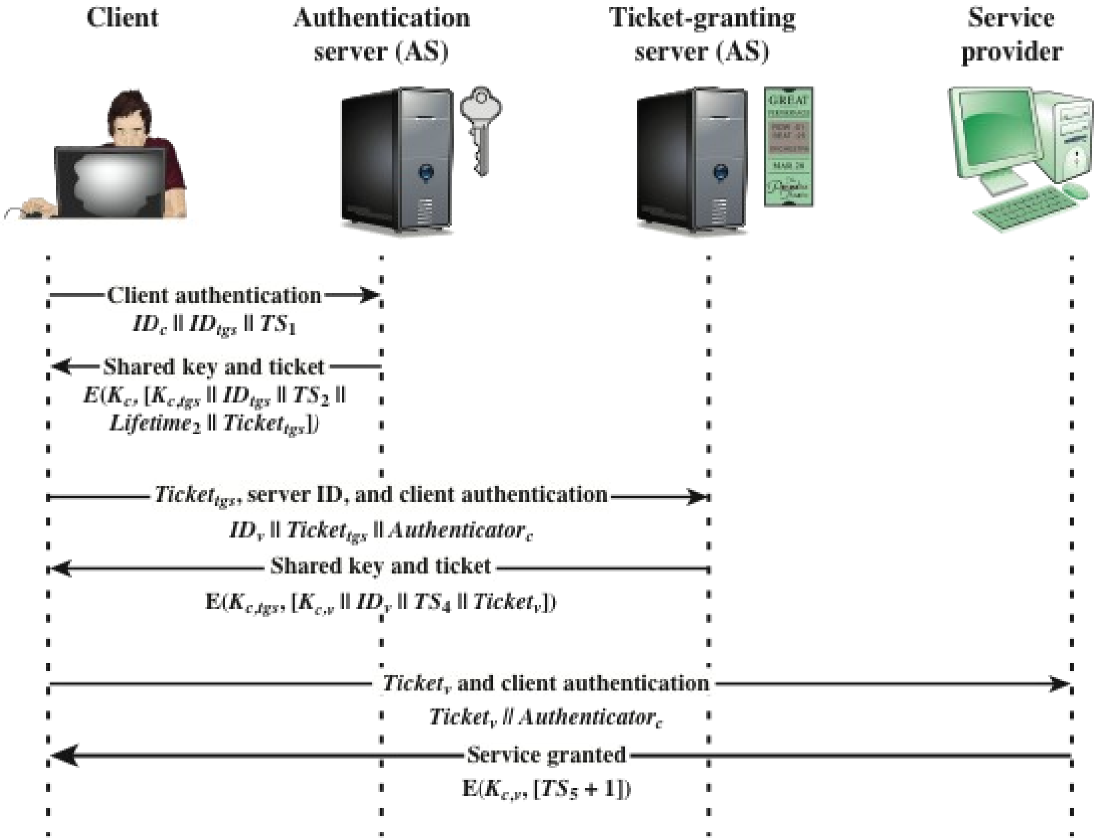

Client
Authentication server (AS)
Ticket-granting server (AS)
Service provider
GREAT
Client authentication
$ID_c \parallel ID_{tgs} \parallel TS_1$
Shared key and ticket
$E(K_c, [K_{c,tgs} \parallel ID_{tgs} \parallel TS_2 \parallel Lifetime_2 \parallel Ticket_{tgs}])$
$Ticket_{tgs}$, server ID, and client authentication
$ID_v \parallel Ticket_{tgs} \parallel Authenticator_c$
Shared key and ticket
$\mathrm{E}(K_{c,tgs}, [K_{c,v} \parallel ID_v \parallel TS_4 \parallel Ticket_v])$
$Ticket_v$ and client authentication
$Ticket_v \parallel Authenticator_c$
Service granted
$\mathrm{E}(K_{c,v}, [TS_5 + 1])$

Table 4-2 Rationale for the Elements of the Kerberos Version 4 Protocol (1 of 7)

(a) Authentication Service Exchange

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).

Table 4-2 Rationale for the Elements of the Kerberos Version 4 Protocol (2 of 7)

K_{ctgs}	Copy of session key accessible to client created by AS to permit secure exchange between client and T G S 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.

Pearson

Table 4-2 Rationale for the Elements of the Kerberos Version 4 Protocol (3 of 7)

(b) Ticket-Granting Service Exchange

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_{ctgs}	Key shared only by C and TGS protects contents of message(4)
$K_{c,v}$	Copy or 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.

Table 4-2 Rationale for the Elements of the Kerberos Version 4 Protocol (4 of 7)

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_{ctgs}	Copy or session key accessible to TGS used to decrypt authenticator, thereby authenticating ticket.
ID_c	Indicates the rightful owner or 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 or time this ticket was issued.

Table 4-2 Rationale for the Elements of the Kerberos Version 4 Protocol (5 of 7)

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_{ctgs}	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 or time this authenticator was generated.

Table 4-2 Rationale for the Elements of the Kerberos Version 4 Protocol (6 of 7)

(c) Client/Server Authentication

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.

Table 4-2 Rationale for the Elements of the Kerberos Version 4 Protocol (7 of 7)

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 (1 of 2)

- 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 Realms (2 of 2)

A Kerberos environment consists of:

- A Kerberos server
- A number of clients
- A number of application servers

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

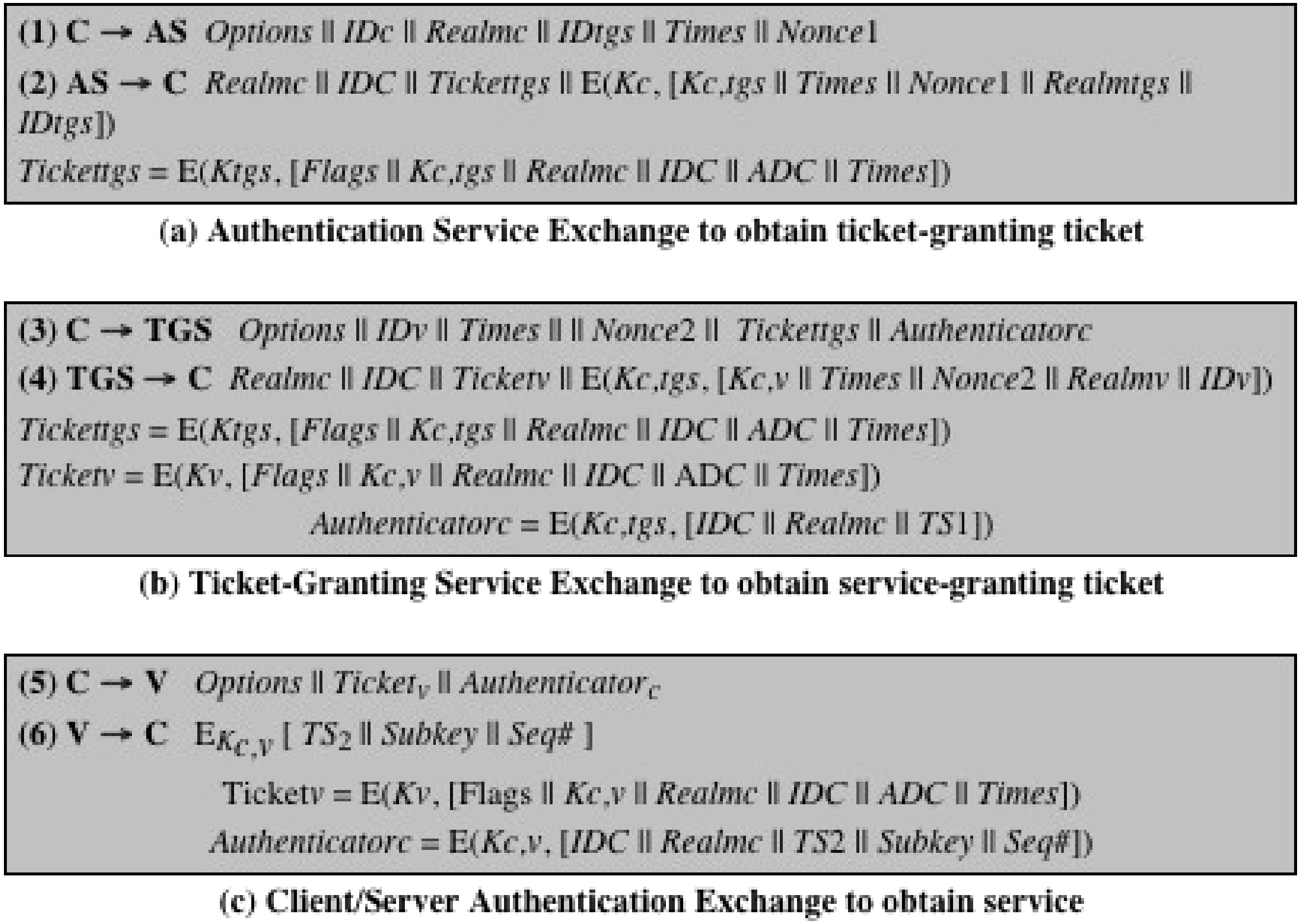
- A service or user name + An instance name + A realm name = Principal name

Differences between Versions 4 and 5

Environmental Shortcomings	**Technical Deficiencies**
• Encryption system dependence • Internet protocol dependence • Message byte ordering • Ticket lifetime • Authentication forwarding • Interrealm authentication	• Double encryption • PCBC encryption • Session keys • Password attacks

Pearson

Table 4-3: Summary of Kerberos Version 5 Message Exchanges



(1) C → AS *Options* || *IDc* || *Realmc* || *IDtgs* || *Times* || *Nonce*1

(2) AS → C *Realmc* || *IDC* || *Tickettgs* || E(*Kc*, [*Kc,tgs* || *Times* || *Nonce*1 || *Realmtgs* || *IDtgs*])

Tickettgs = E(*Ktgs*, [*Flags* || *Kc,tgs* || *Realmc* || *IDC* || *ADC* || *Times*])

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

(3) C → TGS *Options* || *IDv* || *Times* || || *Nonce*2 || *Tickettgs* || *Authenticatorc*

(4) TGS → C *Realmc* || *IDC* || *Ticketv* || E(*Kc,tgs*, [*Kc,v* || *Times* || *Nonce*2 || *Realmv* || *IDv*])

Tickettgs = E(*Ktgs*, [*Flags* || *Kc,tgs* || *Realmc* || *IDC* || *ADC* || *Times*])

Ticketv = E(*Kv*, [*Flags* || *Kc,v* || *Realmc* || *IDC* || ADC || *Times*])

Authenticatorc = E(*Kc,tgs*, [*IDC* || *Realmc* || *TS*1])

(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#*]

Ticketv = E(*Kv*, [Flags || *Kc,v* || *Realmc* || *IDC* || *ADC* || *Times*])

Authenticatorc = E(*Kc,v*, [*IDC* || *Realmc* || *TS*2 || *Subkey* || *Seq#*])

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

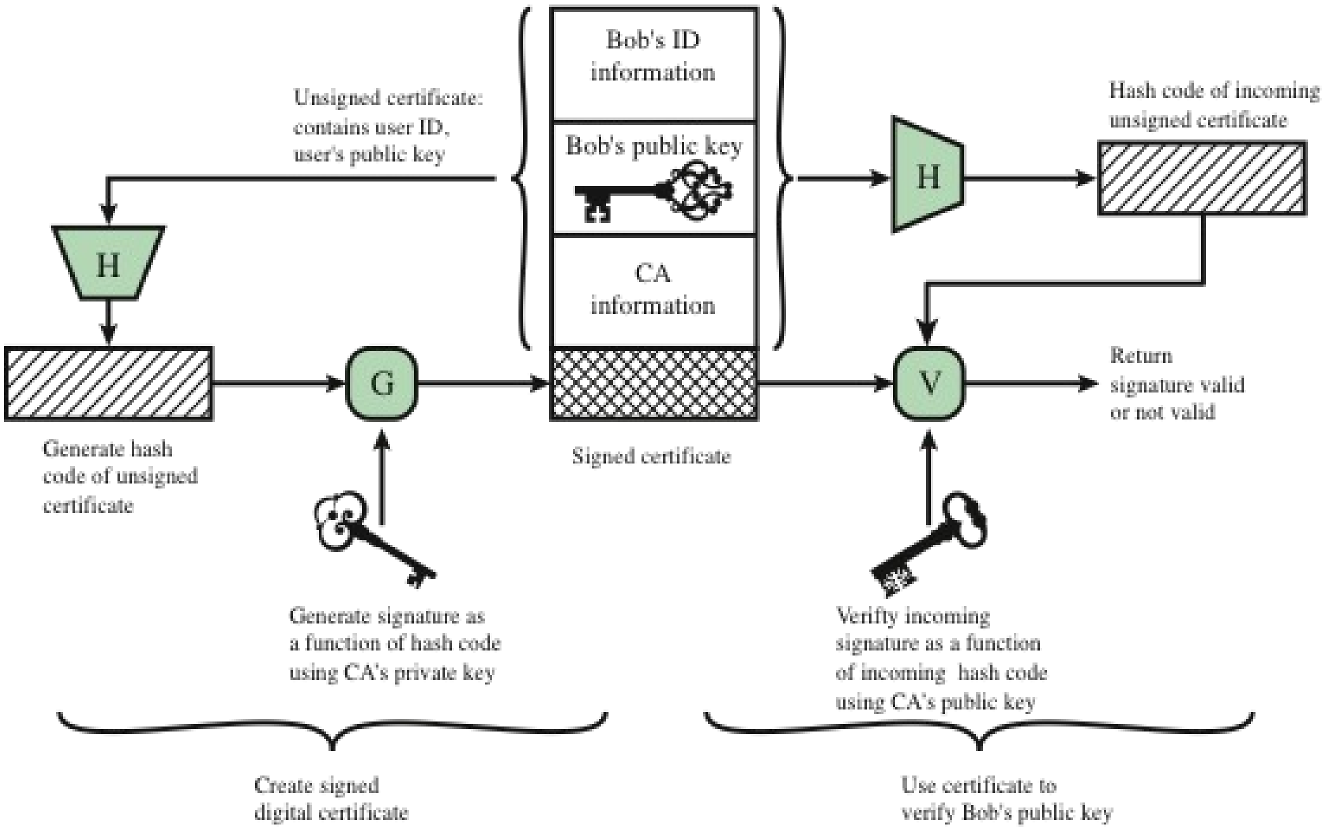
Key Distribution using Asymmetric Encryption (1 of 2)

- 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

Key Distribution using Asymmetric Encryption (2 of 2)

- 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

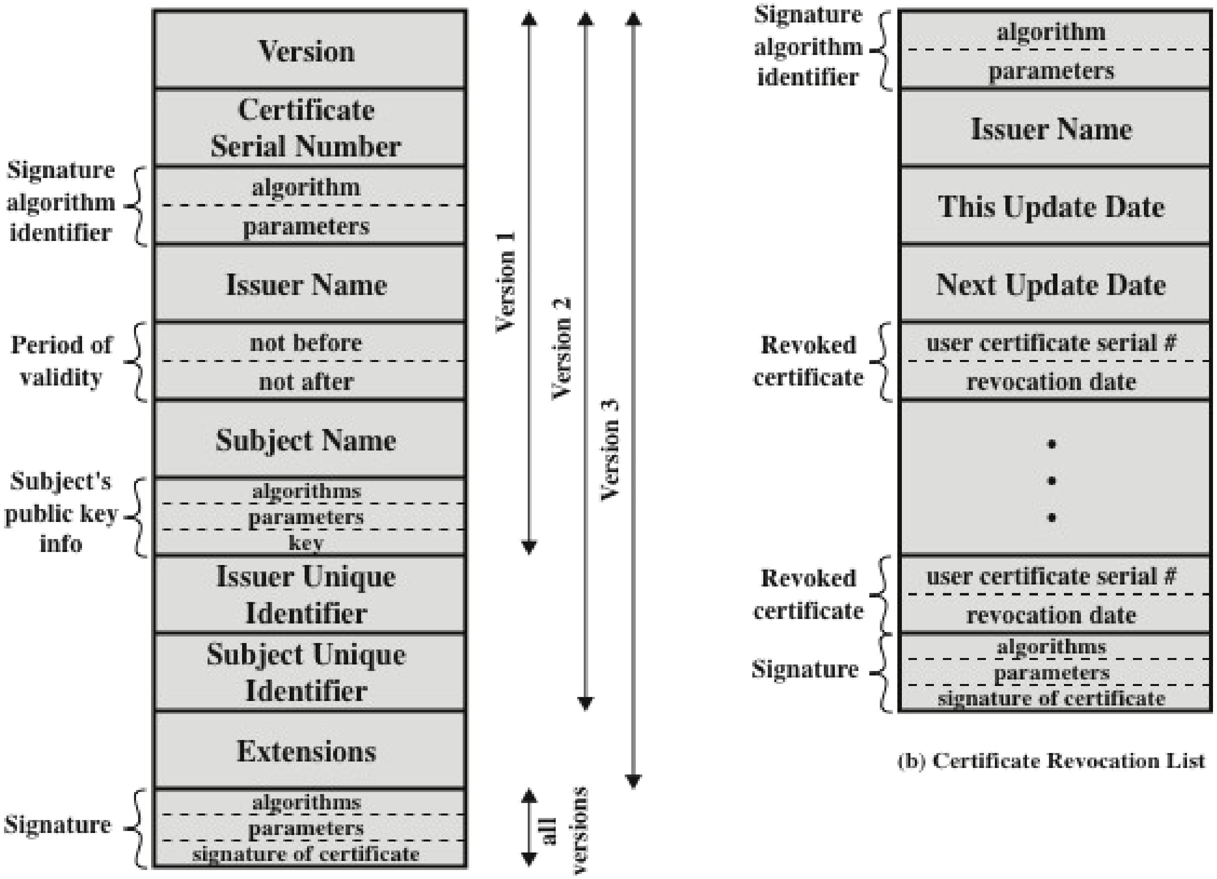
Figure 4-4: Public-Key Certificate Use


Bob's ID information
Bob's public key
CA information
Unsigned certificate: contains user ID, user's public key
H
Generate hash code of unsigned certificate
G
Generate signature as a function of hash code using CA's private key
Signed certificate
H
Hash code of incoming unsigned certificate
V
Return signature valid or not valid
Verifty incoming signature as a function of incoming hash code using CA's public key
Create signed digital certificate
Use certificate to verify Bob's public key

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

Figure 4-5: X.509 Formats

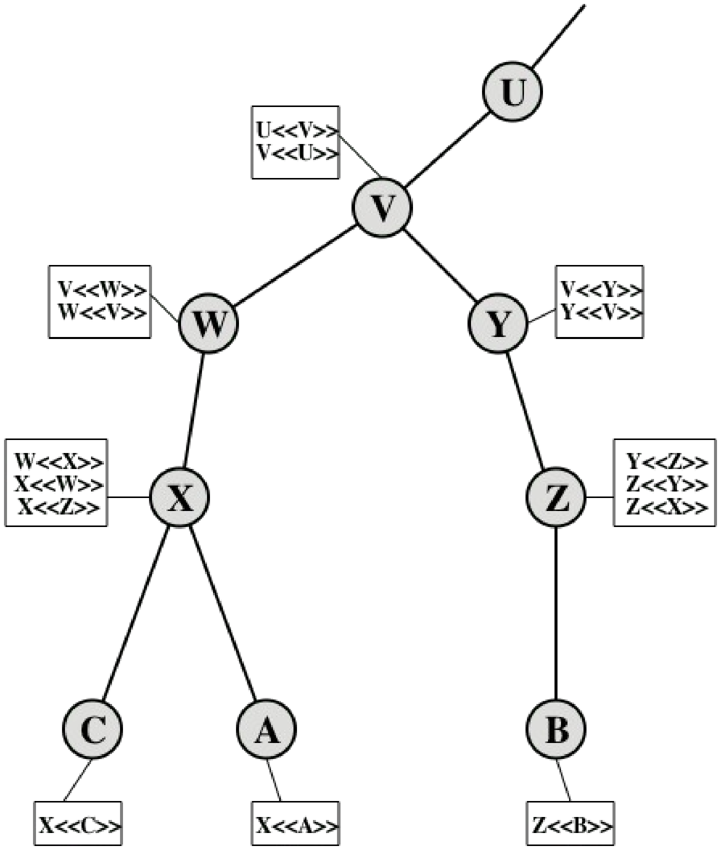


(b) Certificate Revocation List

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

Figure 4-6: X.509 CA Hierarchy-A Hypothetical Example



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

Pearson

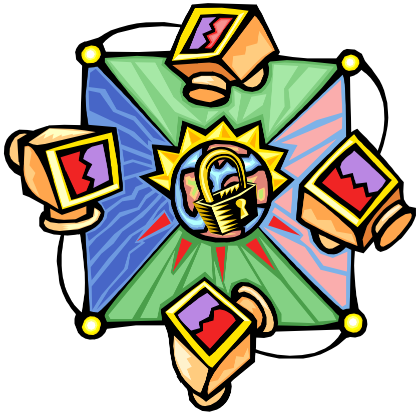
Certificate Subject and Issuer Attributes

- 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
- Includes:
  - Subject alternative name
  - Issuer alternative name
  - Subject directory attributes

Pearson

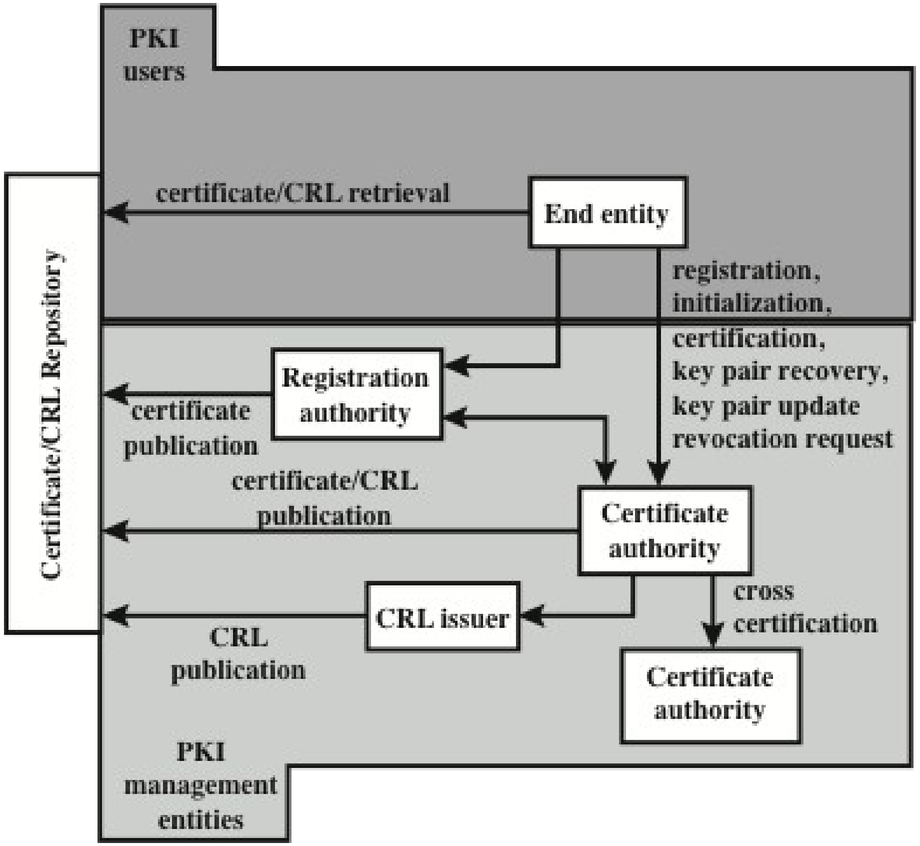
Certification Path Constraints

- 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
  - Name constraints
  - Policy constraints



Pearson

PKIX Architectural Model


PKI users
certificate/CRL retrieval
End entity
registration, initialization, certification, key pair recovery, key pair update revocation request
Certificate/CRL Repository
Registration authority
certificate publication
certificate/CRL publication
Certificate authority
CRL issuer
cross certification
Certificate authority
CRL publication
PKI management entities

PKIX Management Functions (1 of 2)

- Functions that potentially need to be supported by management protocols:
  - Registration
  - Initialization
  - Certification
  - Key pair recovery
  - Key pair update
  - Revocation request
  - Cross certification

PKIX Management Functions (2 of 2)

- Alternative management protocols:
  - Certificate management protocols (CMP)

    Designed to be a flexible protocol able to accommodate a variety of technical, operational, and business models
  - Certificate management messages over CMS (CMC)

    Is built on earlier work and is intended to leverage existing implementations

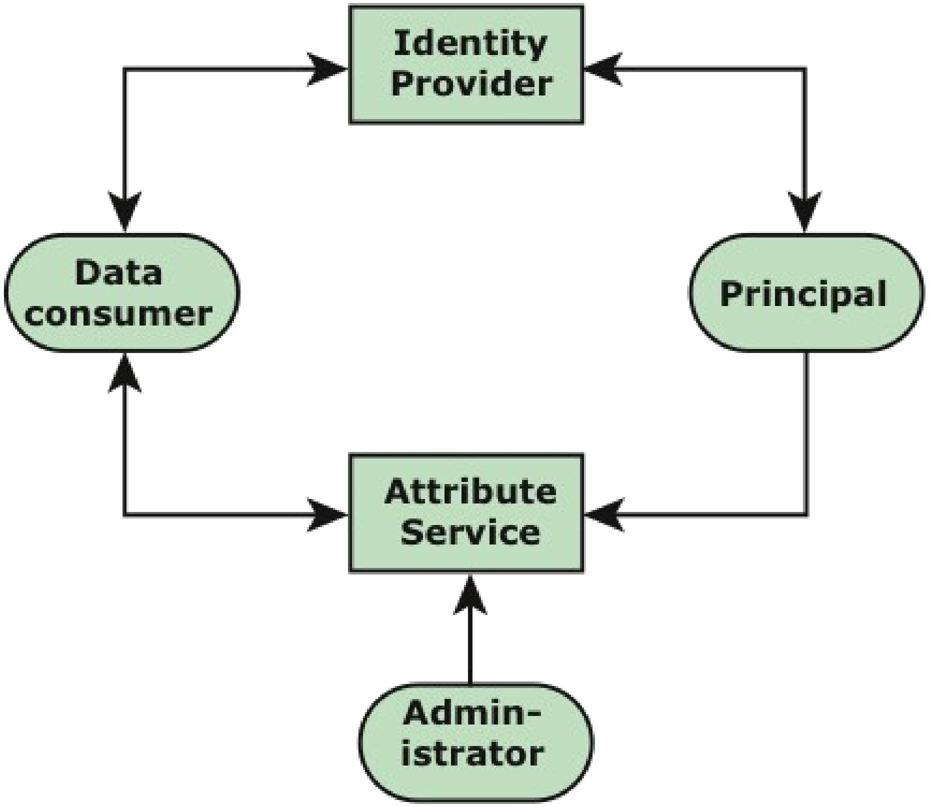
Identity Management (1 of 2)

- A centralized, automated approach to provide enterprise wide access to resources by employees and other authorized individuals
  - Focus is defining an identity for each user (human or process), associating attributes with the identity, and enforcing a means by which a user can verify identity
  - Central concept is the use of single sign-on (SSO) which enables a user to access all network resources after a single authentication

Identity Management (2 of 2)

- Principal elements of an identity management system:
  - Authentication
  - Authorization
  - Accounting
  - Provisioning
  - Workflow automation
  - Delegated administration
  - Password synchronization
  - Self-service password reset
  - Federation

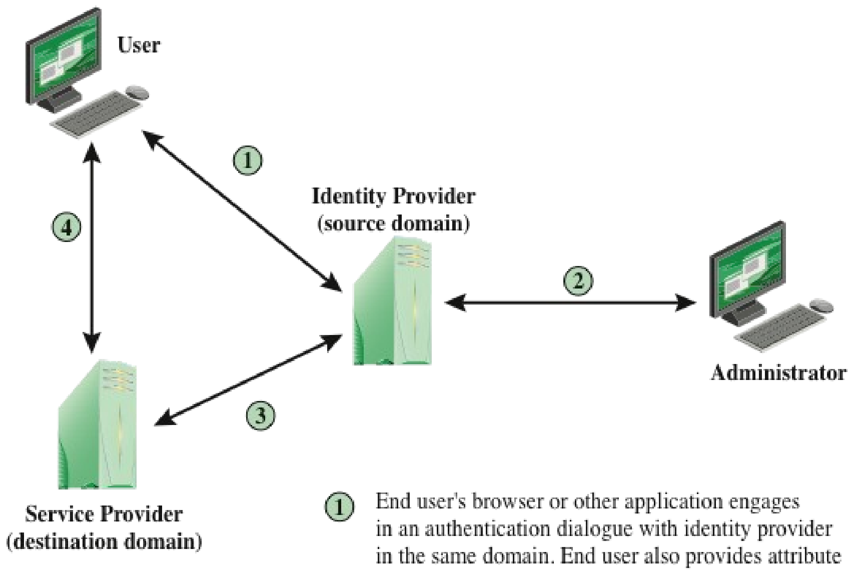
Figure 4-8: Generic Identity Management System



Identity Federation

- Identity federation is, in essence, an extension of identity management to multiple security domains
- Federated identity management refers to the agreements, standards, and technologies that enable the portability of identities, identity attributes, and entitlements across multiple enterprises and numerous applications and supports many thousands, even millions, of users
- Another key function of federated identity management is identity mapping
  - The federated identity management protocols map identities and attributes of a user in one domain to the requirements of another domain

Federated Identity Operation



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.

Standards (1 of 2)

The Extensible Markup Language (XML)

- Appear similar to HTML documents that are visible as Web pages, but provide greater functionality
- Includes strict definitions of the data type of each field
- Provides encoding rules for commands that are used to transfer and update data objects

The Simple Object Access Protocol (SOAP)

- Minimal set of conventions for invoking code using XML over HTTP
- Enables applications to request services from one another with X ML-based requests and receive responses as data formatted with X ML

Standards (2 of 2)

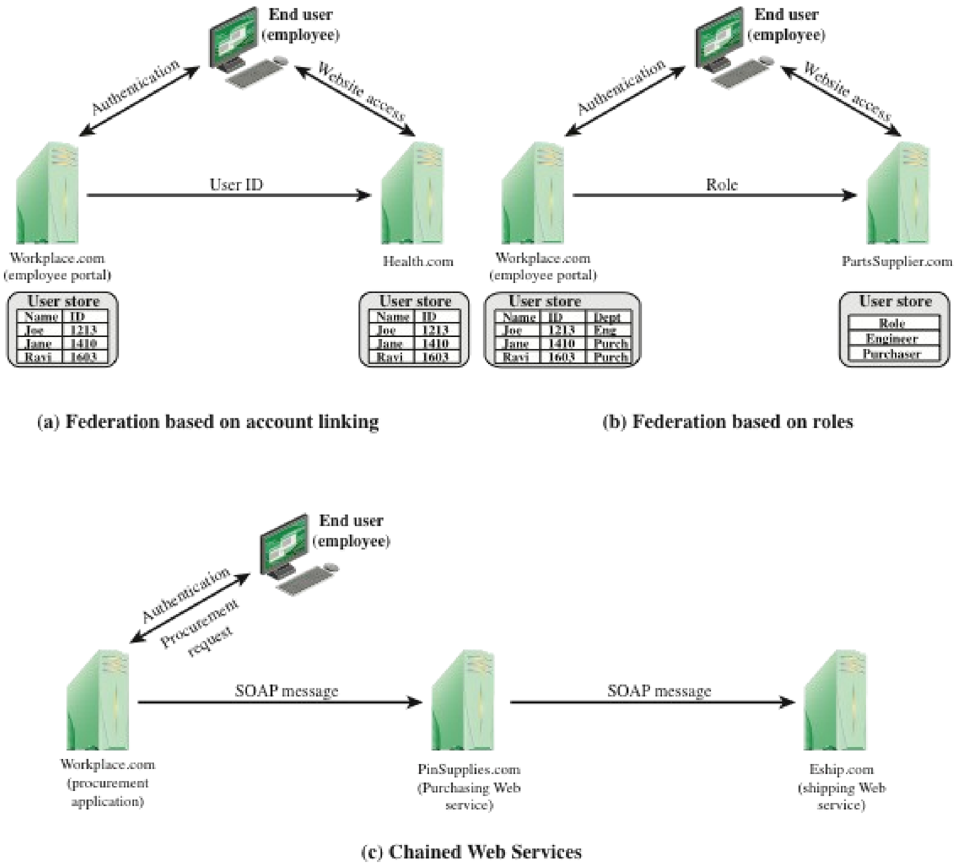
WS-Security

- A set of SOAP extensions for implementing message integrity and confidentiality in Web services
- Assigns security tokens to each message for use in authentication

Security Assertion Markup Language (SAML)

- An XML-based language for the exchange of security information between online business partners
- Conveys authentication information in the form of assertions about subjects

Figure 4-10: Federated Identity Scenarios



User store

Name	ID
Joe	1213
Jane	1410
Ravi	1603

User store

Name	ID
Joe	1213
Jane	1410
Ravi	1603

(a) Federation based on account linking

User store

Name	ID	Dept
Joe	1213	Eng
Jane	1410	Purch
Ravi	1603	Purch

User store

Role
Engineer
Purchaser

(b) Federation based on roles

(c) Chained Web Services

Summary (1 of 2)

- Remote user authentication principles
  - The NIST model for electronic user authentication
  - Means of authentication
- Symmetric key distribution using symmetric encryption
- Kerberos
  - Version 4
  - Version 5
- Key distribution using asymmetric encryption
  - Public-key certificates
  - Public-key distribution of secret keys

Summary (2 of 2)

- X.509 certificates
  - Certificates
  - X.509 Version 3
- Public-key infrastructure
  - PKIX management functions
  - PKIX management protocols
- Federated identity management
  - Identity management
  - Identity federation

Copyright