User Authentication

We cannot enter into alliance with neighboring princes until we are acquainted with their designs.

***—The Art of War*, Sun Tzu**

User Authentication

- fundamental security building block
  - basis of access control & user accountability
- is the process of verifying an identity claimed by or for a system entity
- has two steps:
  - identification - specify identifier
  - verification - bind entity (person) and identifier
- distinct from message authentication

Means of User Authentication

- four means of authenticating user's identity
- based one something the individual
  - knows - e.g. password, PIN
  - possesses - e.g. key, token, smartcard
  - is (static biometrics) - e.g. fingerprint, retina
  - does (dynamic biometrics) - e.g. voice, sign
- can use alone or combined
- all can provide user authentication
- all have issues

Authentication Protocols

- used to convince parties of each others identity and to exchange session keys
- may be one-way or mutual
- key issues are
  - confidentiality – to protect session keys
  - timeliness – to prevent replay attacks

Replay Attacks

- where a valid signed message is copied and later resent
  - simple replay
  - repetition that can be logged
  - repetition that cannot be detected
  - backward replay without modification
- countermeasures include
  - use of sequence numbers (generally impractical)
  - timestamps (needs synchronized clocks)
  - challenge/response (using unique nonce)

One-Way Authentication

- required when sender & receiver are not in communications at same time (eg. email)
- have header in clear so can be delivered by email system
- may want contents of body protected & sender authenticated

Using Symmetric Encryption

- as discussed previously can use a two-level hierarchy of keys
- usually with a trusted Key Distribution Center (KDC)
  - each party shares own master key with KDC
  - KDC generates session keys used for connections between parties
  - master keys used to distribute these to them

Needham-Schroeder Protocol

- original third-party key distribution protocol
- for session between A B mediated by KDC
- protocol overview is:

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: $E(K_s, [f(N_2)])$

Needham-Schroeder Protocol

- used to securely distribute a new session key for communications between A & B
- but is vulnerable to a replay attack if an old session key has been compromised
  - then message 3 can be resent convincing B that is communicating with A
- modifications to address this require:
  - timestamps in steps 2 & 3 (Denning 81)
  - using an extra nonce (Neuman 93)

One-Way Authentication

- use refinement of KDC to secure email
  - since B no online, drop steps 4 & 5
- protocol becomes:
  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: $E(K_b, [K_s||ID_A]) \,||\, E(K_s, M)$
- provides encryption & some authentication
- does not protect from replay attack

Kerberos

- trusted key server system from MIT
- provides centralised private-key third-party authentication in a distributed network
  - allows users access to services distributed through network
  - without needing to trust all workstations
  - rather all trust a central authentication server
- two versions in use: 4 & 5

Kerberos Requirements

- its first report identified requirements as:
  - secure
  - reliable
  - transparent
  - scalable
- implemented using an authentication protocol based on Needham-Schroeder

Kerberos v4 Overview

- a basic third-party authentication scheme
- have an Authentication Server (AS)
  - users initially negotiate with AS to identify self
  - AS provides a non-corruptible authentication credential (ticket granting ticket TGT)
- have a Ticket Granting server (TGS)
  - users subsequently request access to other services from TGS on basis of users TGT
- using a complex protocol using DES

Kerberos v4 Dialogue

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

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

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

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

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

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

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

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

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

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

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

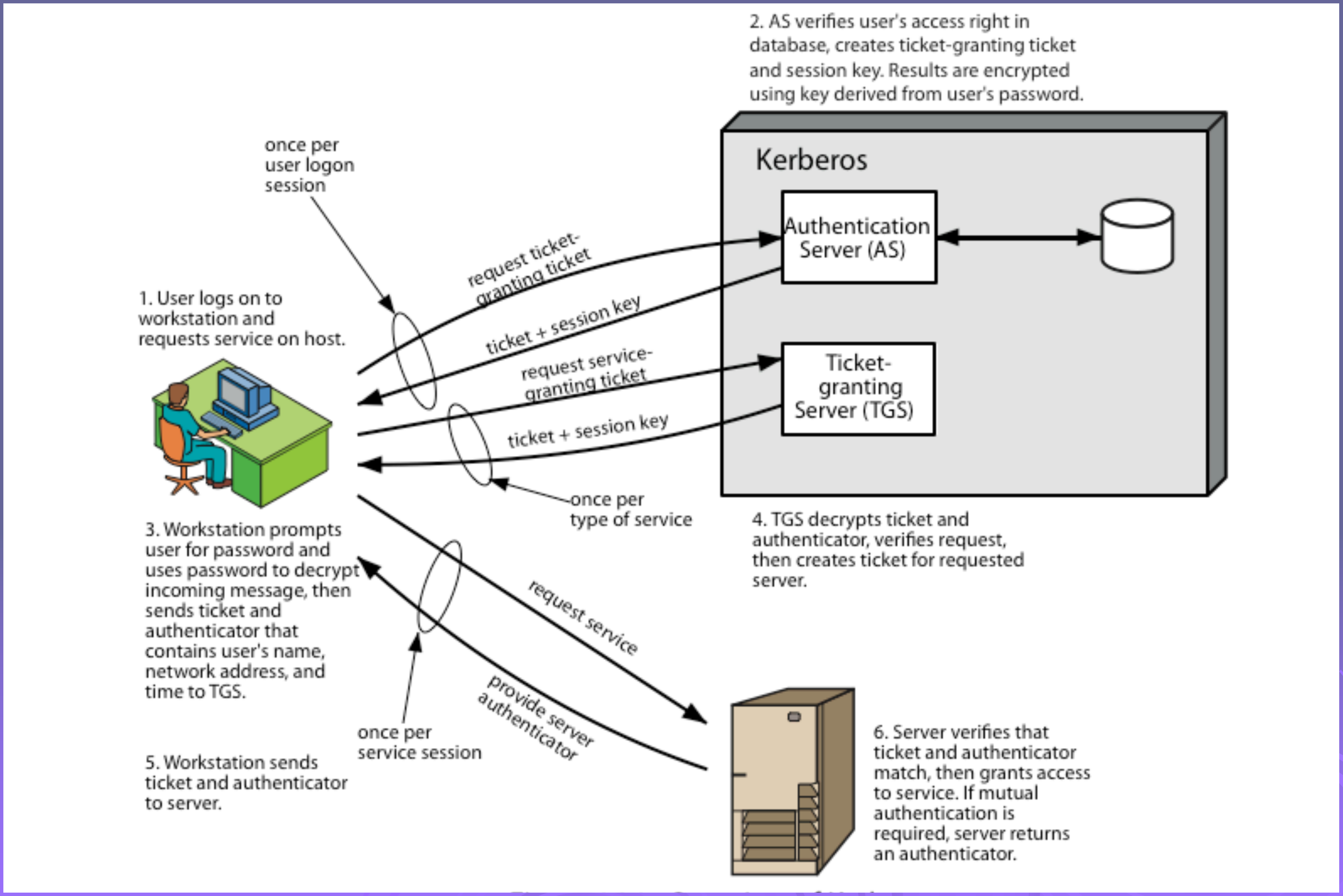
(6) V → C $E(K_{c,v}, [TS_5 + 1])$ (for mutual authentication)

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

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

(c) Client/Server Authentication Exchange to obtain service

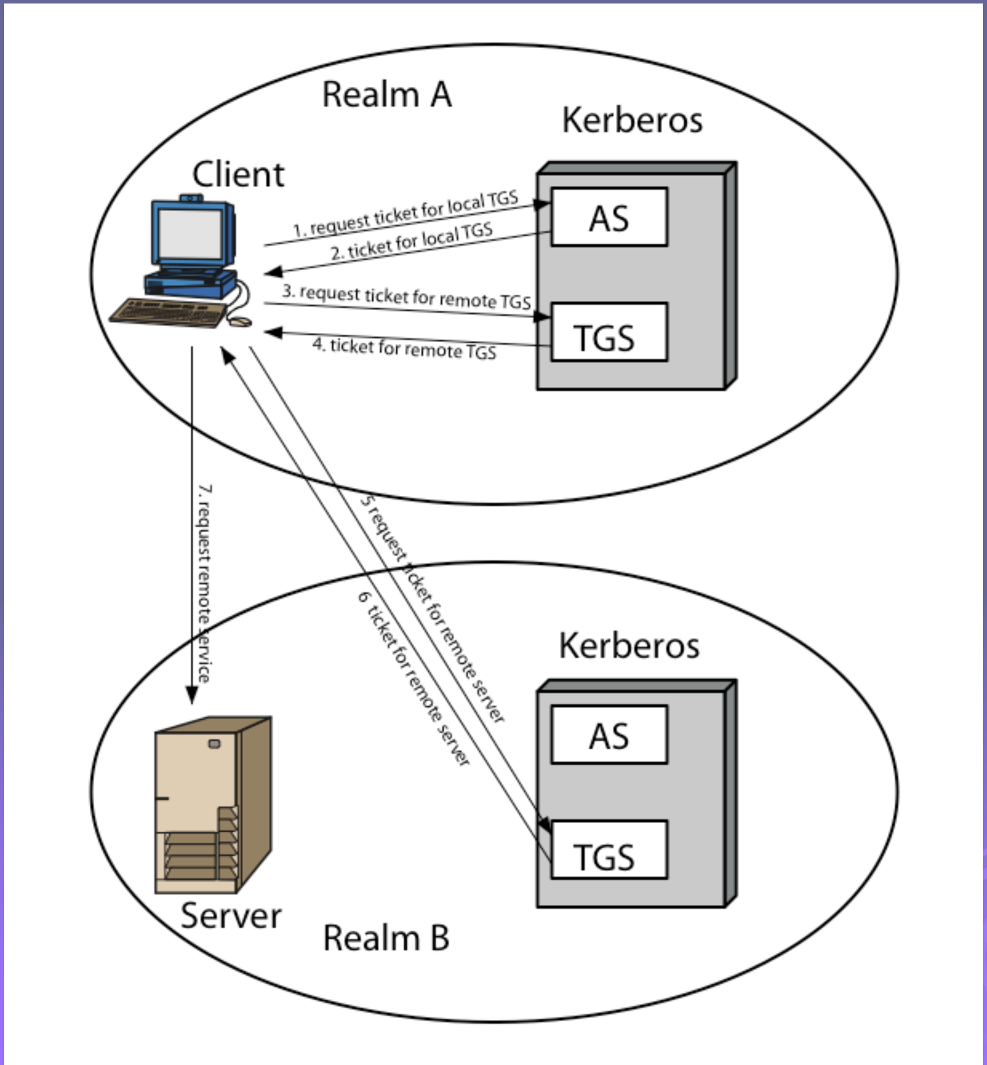
Kerberos 4 Overview



Kerberos Realms

- a Kerberos environment consists of:
  - a Kerberos server
  - a number of clients, all registered with server
  - application servers, sharing keys with server
- this is termed a realm
  - typically a single administrative domain
- if have multiple realms, their Kerberos servers must share keys and trust

Kerberos Realms



Kerberos Version 5

- developed in mid 1990's
- specified as Internet standard RFC 1510
- provides improvements over v4
  - addresses environmental shortcomings
    - encryption alg, network protocol, byte order, ticket lifetime, authentication forwarding, interrealm auth
  - and technical deficiencies
    - double encryption, non-std mode of use, session keys, password attacks

Kerberos v5 Dialogue

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

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

$$Ticket_{tgs} = \mathrm{E}(K_{tgs}, [Flags \| K_{c,tgs} \| Realm_c \| ID_C \| AD_C \| Times])$$

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

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

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

$$Ticket_{tgs} = \mathrm{E}(K_{tgs}, [Flags \| K_{c,tgs} \| Realm_c \| ID_C \| AD_C \| Times])$$

$$\mathrm{Ticket}_v = \mathrm{E}(\mathrm{K}_v, [\mathrm{Flags} \| \mathrm{K}_{c,v} \| \mathrm{Realm}_c \| \mathrm{ID}_C \| \mathrm{AD}_C \| \mathrm{Times}])$$

$$Authenticator_c = \mathrm{E}(K_{c,tgs}, [ID_C \| Realm_c \| TS_1])$$

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

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

(6) V → C $\mathrm{E}_{K_{c,v}} [\, \mathrm{TS}_2 \| \mathrm{Subkey} \| \mathrm{Seq\#} \,]$

$$\mathrm{Ticket}_v = \mathrm{E}(\mathrm{K}_v, [\mathrm{Flags} \| \mathrm{K}_{c,v} \| \mathrm{Realm}_c \| \mathrm{ID}_C \| \mathrm{AD}_C \| \mathrm{Times}])$$

$$Authenticator_c = \mathrm{E}(K_{c,v}, [ID_C \| Realm_c \| TS_2 \| Subkey \| Seq\#])$$

(c) Client/Server Authentication Exchange to obtain service

Remote User Authentication

- in Ch 14 saw use of public-key encryption for session key distribution
  - assumes both parties have other's public keys
  - may not be practical
- have Denning protocol using timestamps
  - uses central authentication server (AS) to provide public-key certificates
  - requires synchronized clocks
- have Woo and Lam protocol using nonces
- care needed to ensure no protocol flaws

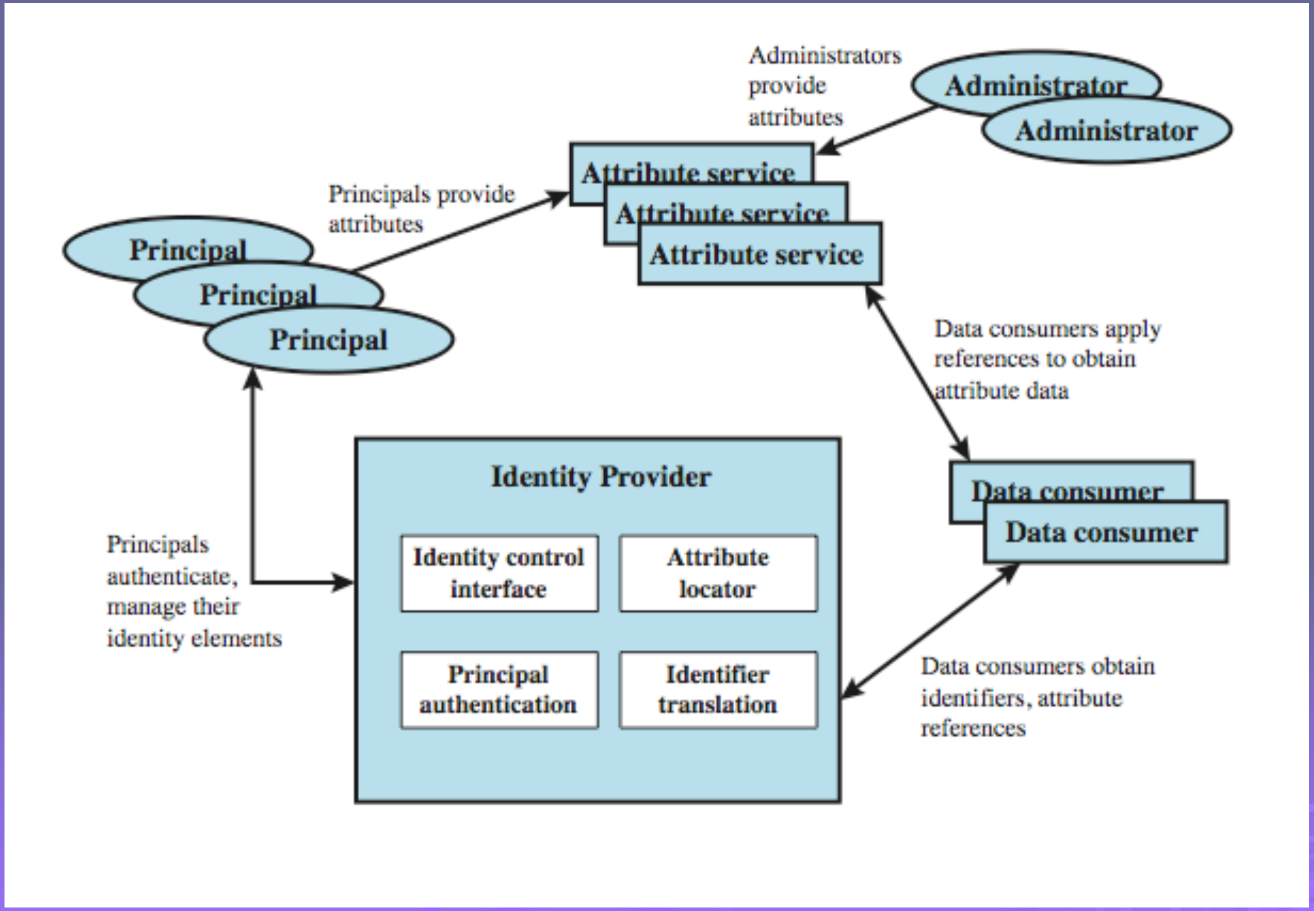
One-Way Authentication

- have public-key approaches for email
  - encryption of message for confidentiality, authentication, or both
  - must now public keys
  - using costly public-key alg on long message
- for confidentiality encrypt message with one-time secret key, public-key encrypted
- for authentication use a digital signature
  - may need to protect by encrypting signature
- use digital certificate to supply public key

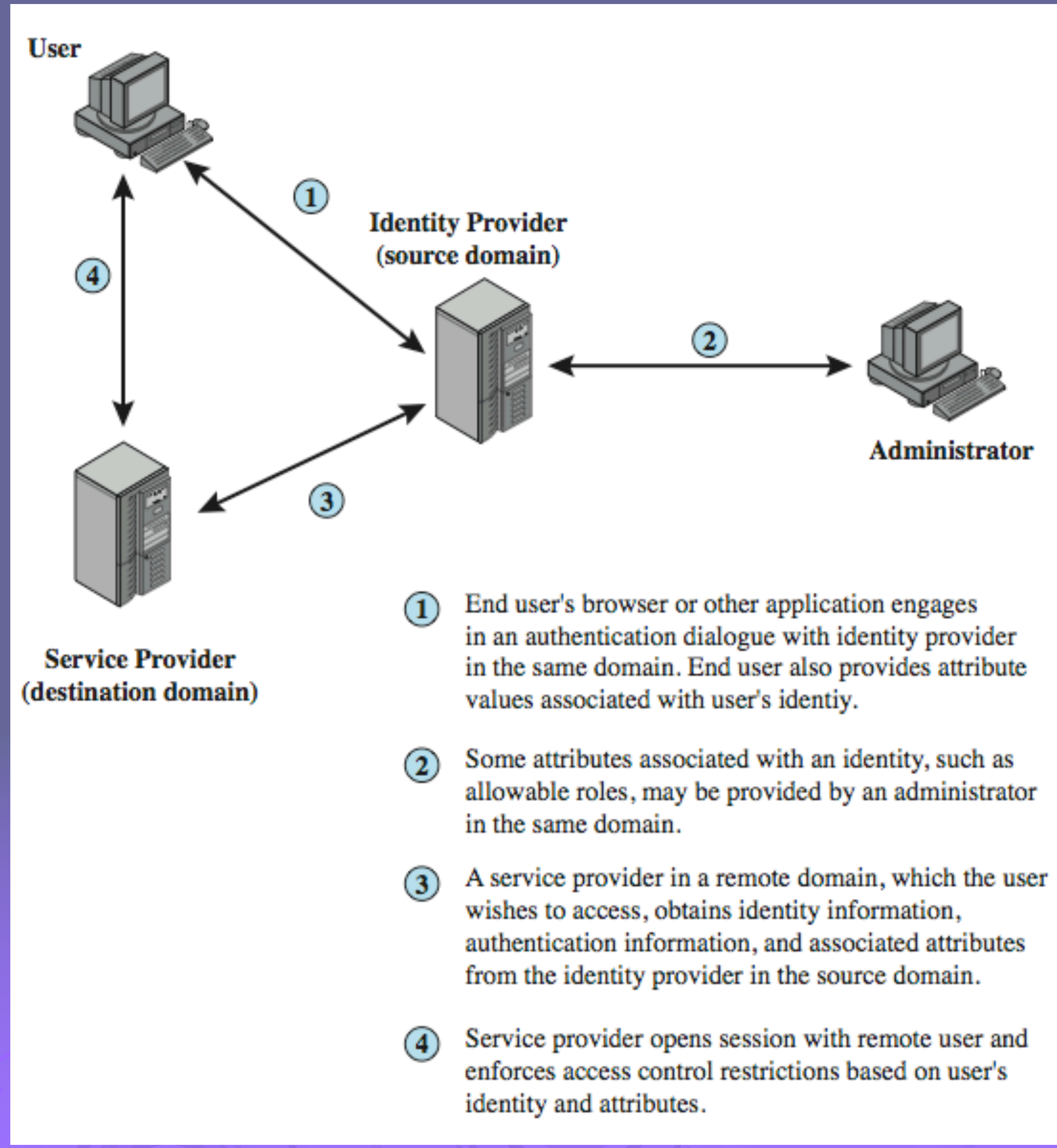
Federated Identity Management

- use of common identity management scheme
  - across multiple enterprises & numerous applications
  - supporting many thousands, even millions of users
- principal elements are:
  - authentication, authorization, accounting, provisioning, workflow automation, delegated administration, password synchronization, self-service password reset, federation
- Kerberos contains many of these elements

Identity Management



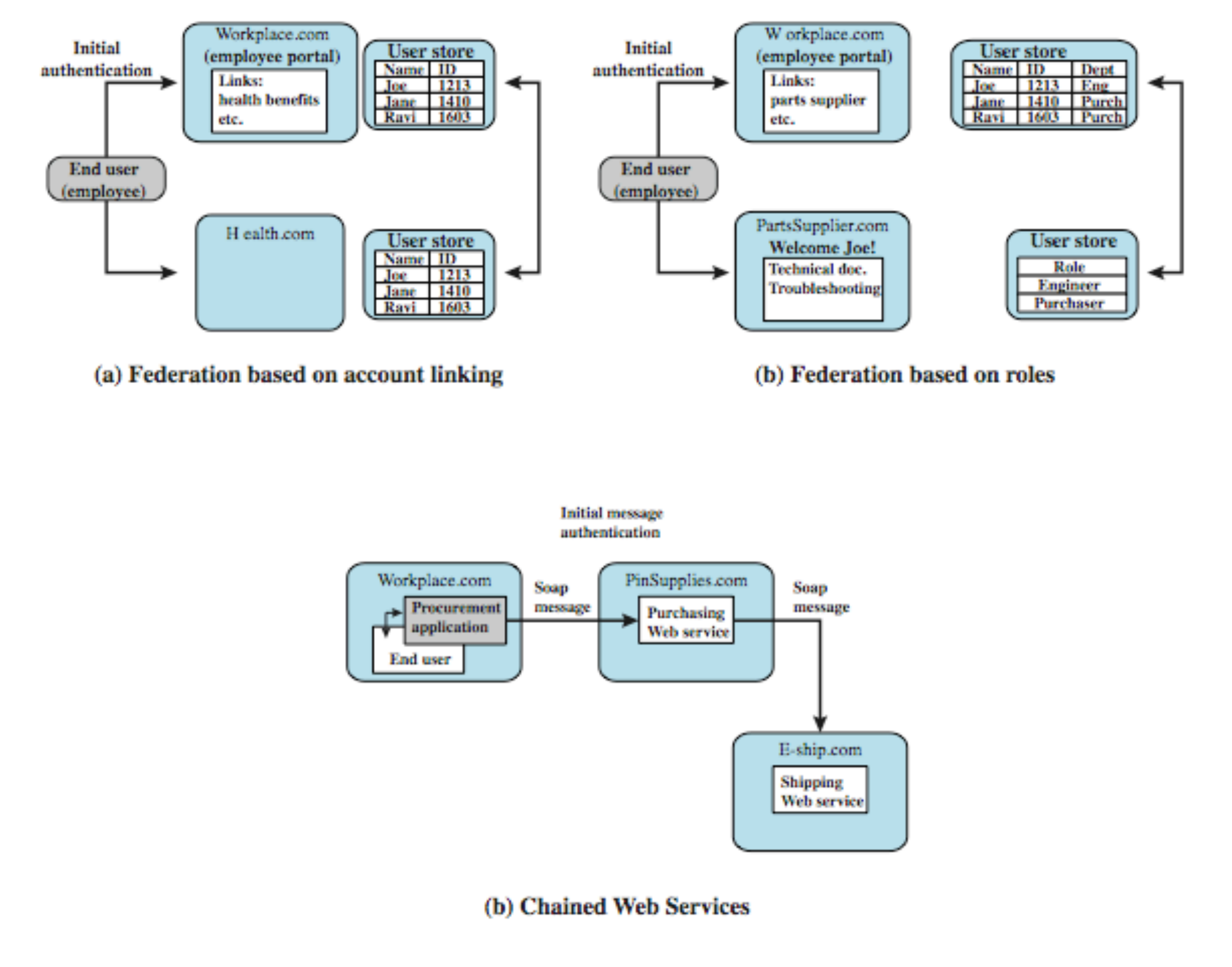
Identity Federation



Standards Used

- Security Assertion Markup Language (SAML)
  - XML-based language for exchange of security information between online business partners
- part of OASIS (Organization for the Advancement of Structured Information Standards) standards for federated identity management
  - e.g. WS-Federation for browser-based federation
- need a few mature industry standards

Federated Identity Examples



Summary

- have considered:
  - remote user authentication issues
  - authentication using symmetric encryption
  - the Kerberos trusted key server system
  - authentication using asymmetric encryption
  - federated identity management