INF3510 Information Security
University of Oslo
Spring 2011

Lecture 8

Identity and Access Management

Audun Jøsang

Outline

- Identity and access management concepts
- Identity management models
- Access control models (security models)

Identity & access management

Identity

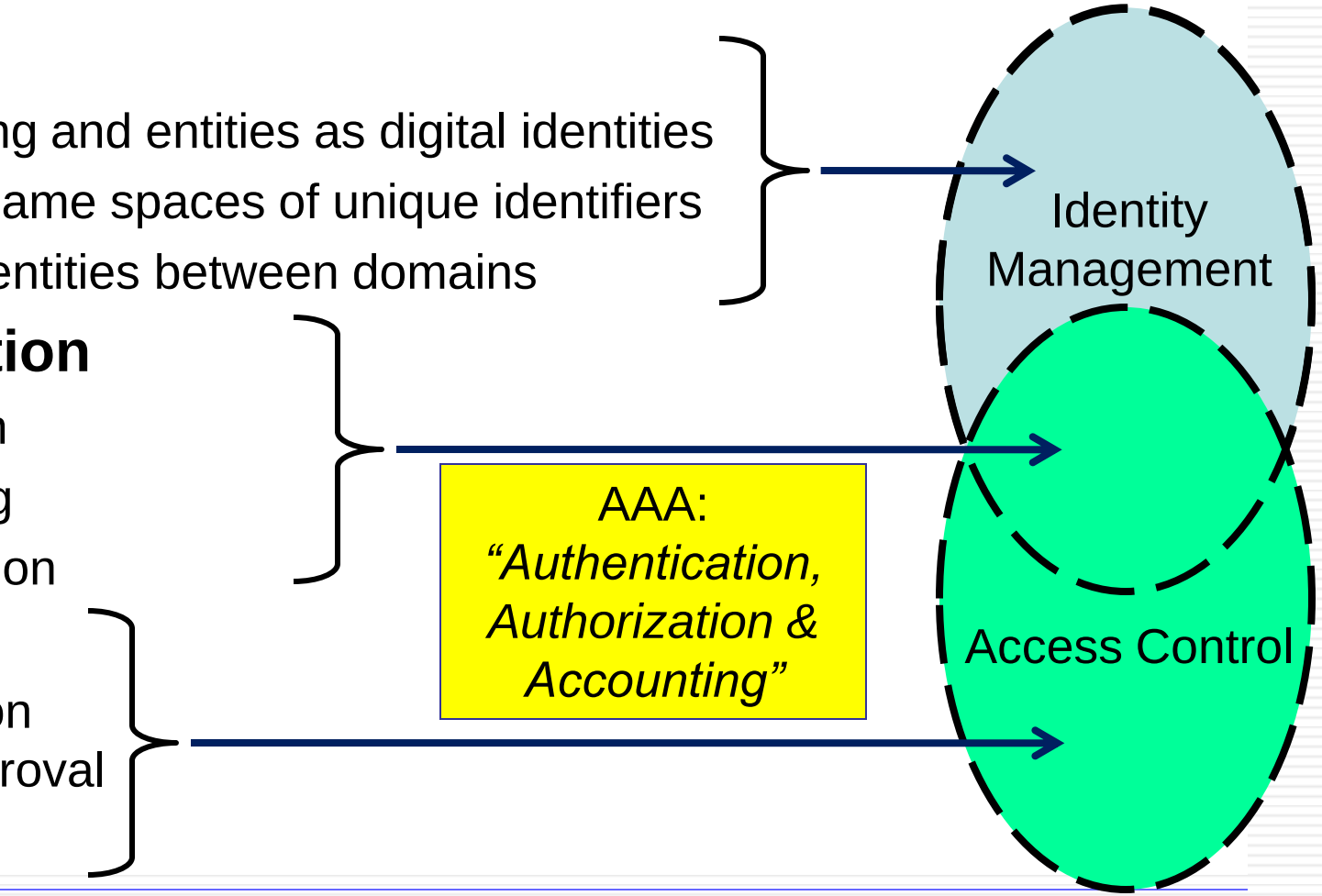
- Representing and entities as digital identities
- Managing name spaces of unique identifiers
- Mapping identities between domains

Authentication

- Registration
- Provisioning
- Authentication

Access

- Authorization
- Access approval
- Accounting



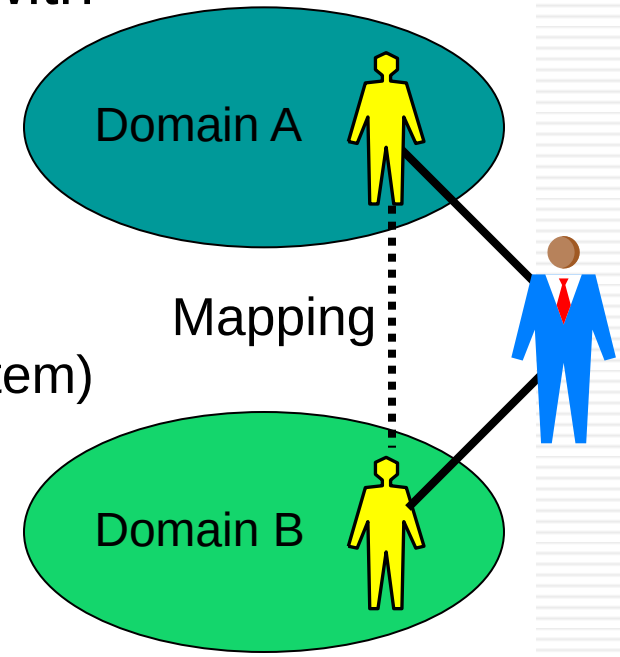
Four types of identity management

(1) Mgmt of user IDs and credentials on SP side	(2) Mgmt of user IDs and credentials on user side
(3) Mgmt of SP IDs and credentials on SP side	(4) Mgmt of SP IDs and credentials on user side

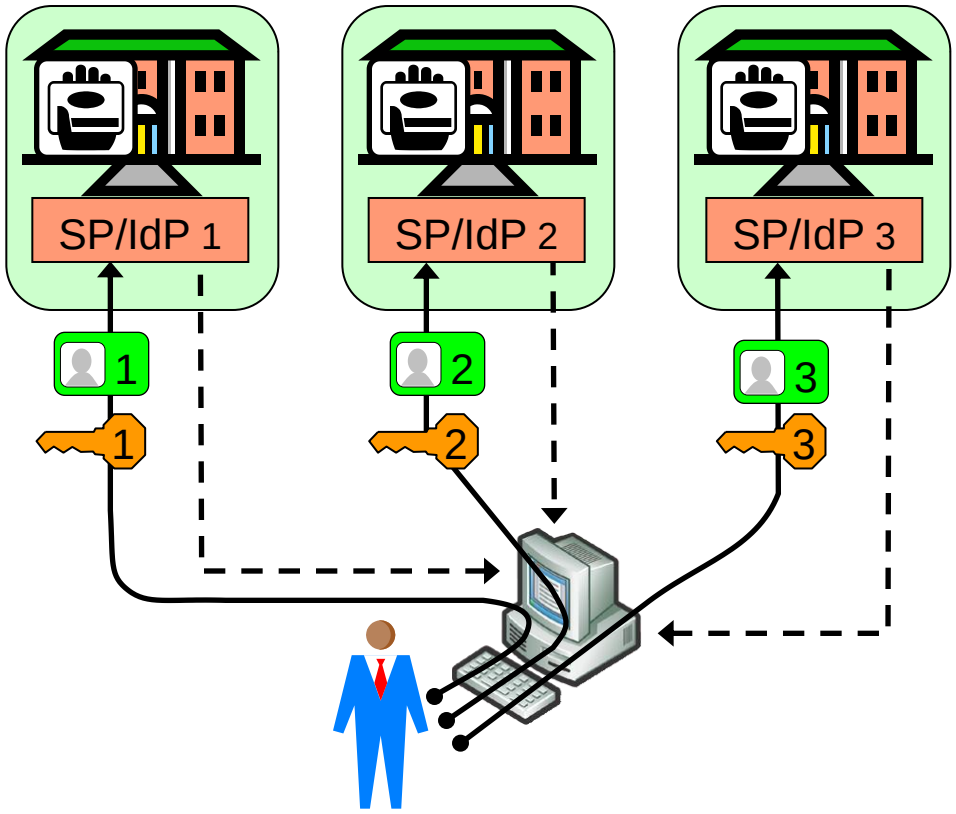
- Only type 1 & 2 are traditionally considered part of IAM
- Types 3,4 are discussed in Lect.9 (Net. & Com. Security

Identity Domains

- An identity domain is a network realm with a name space of unique names
- Management structures:
  - Single authority, e.g. User Ids in company network
  - Hierarchical: e.g. DNS (Domain Name System)
- A single policy is normally applied in a domain
- Integration/federation of domains
  - Requires mapping of identities of same entity
  - Requires alignment of policies



Silo domain model



Legend:

SP

IdP

Identity domain

User identifier managed by IdP #

Authentication token managed by IdP #

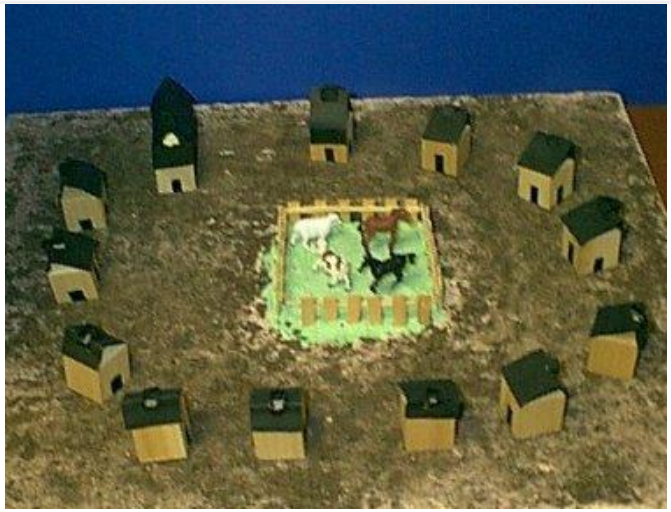
Service logon

Service provision

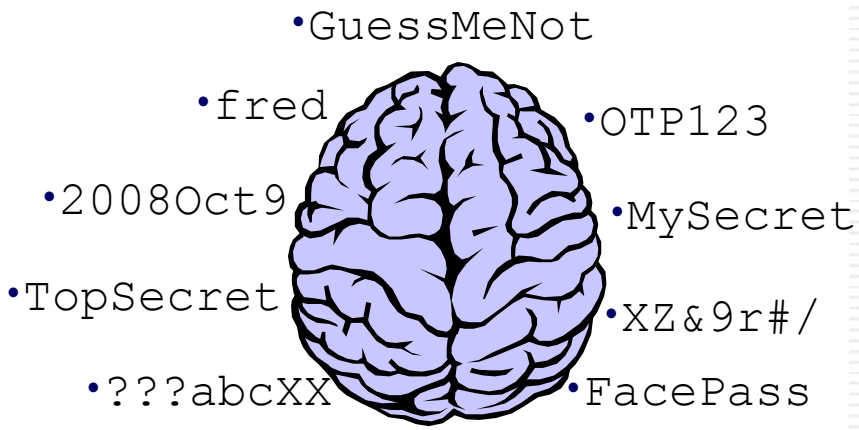
Silo user-identity domains

- SP = IdP: defines name space and provides access credentials
- Unique identifier assigned to each entity
- Advantages
  - Simple to deploy, low cost for SPs
- Disadvantages
  - Identity overload for users, poor usability

Tragedies of the Commons



Common village green



Common brain

Common pockets

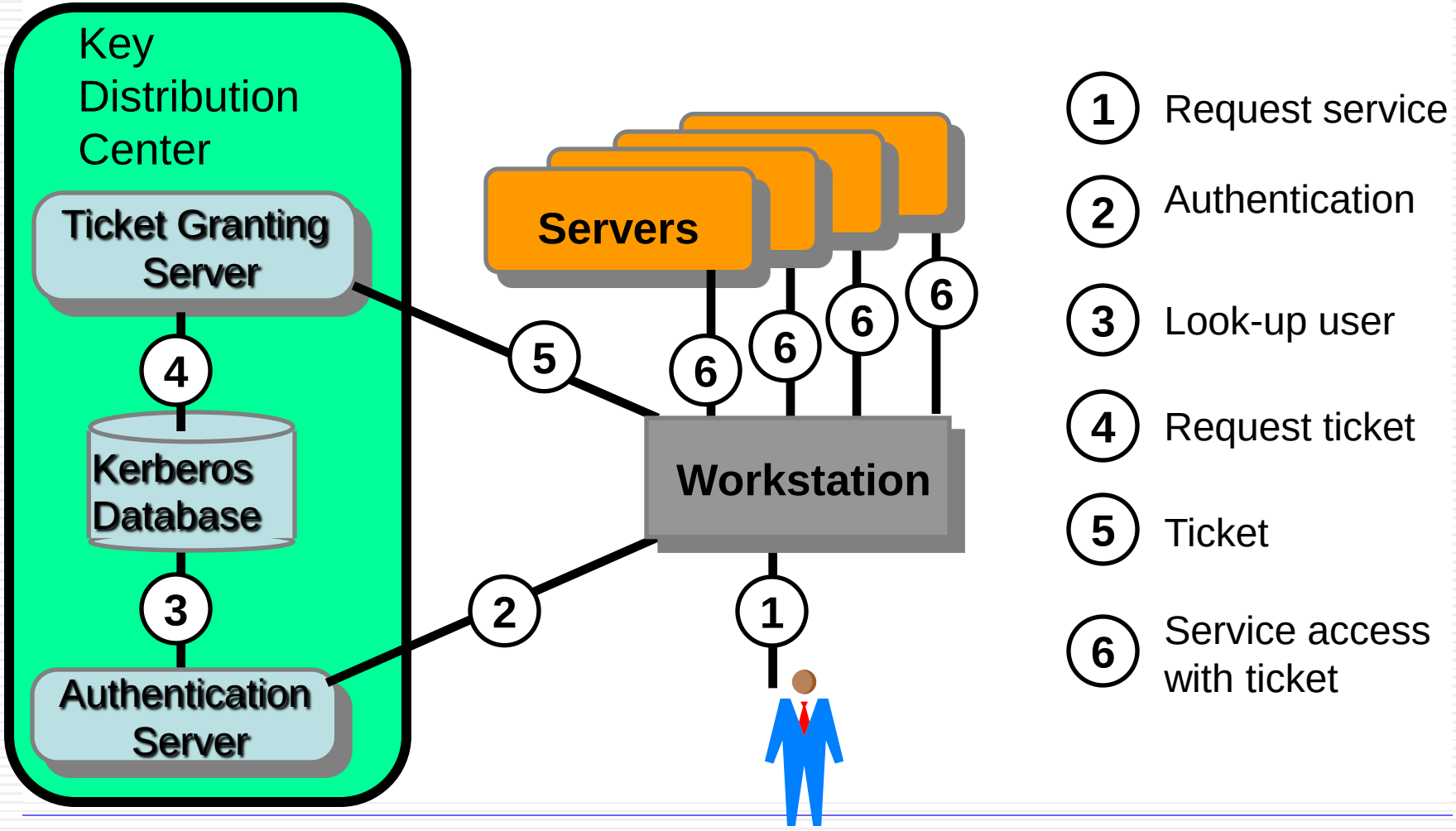
Push towards SSO (Single Sign-On)

- Users don’t want more identifiers
- Low acceptance of new services that require separate user authentication
- Silo model requires users to provide same information to many service providers
- Silo model makes it difficult to offer bundled services, i.e. from different service providers
- Service providers want better quality user information

Kerberos SSO

- Part of project Athena (MIT) in 1983.
- User must identify itself once at the beginning of a workstation session (login session).
- Does not require user to enter password every time a service is requested!
- Every user shares a password with the AS (Authentication Server)
- Every SP (service provider) shares a secret key with the TGS (Ticket Granting Server)
- Tickets are sealed (encrypted) by TGS proves to SPs that the user has been authenticated

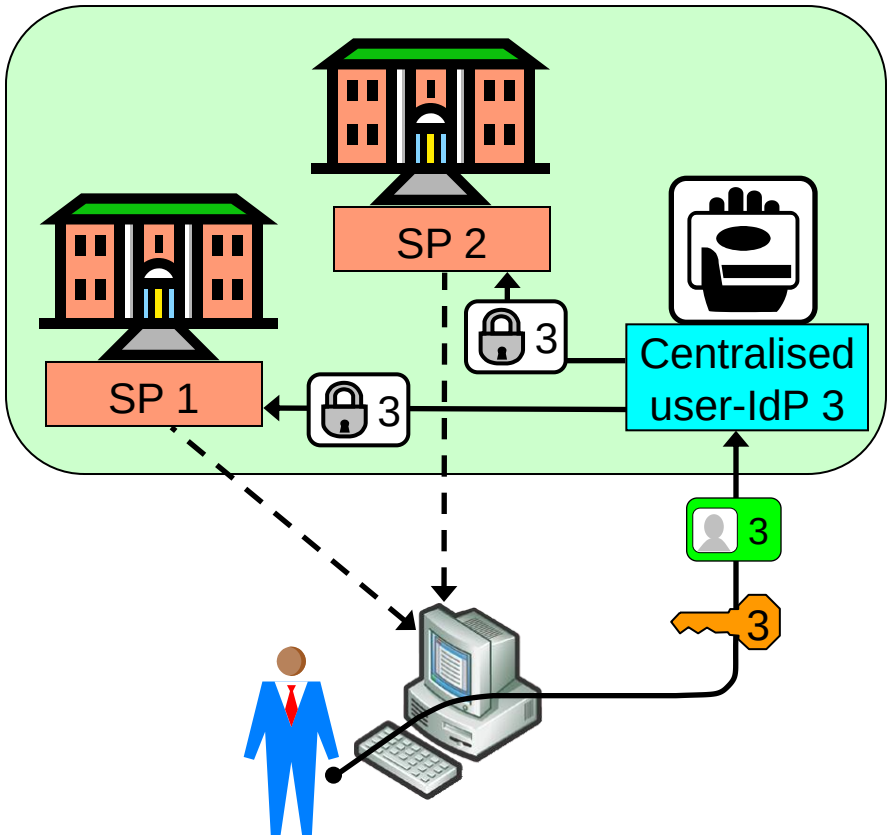
Kerberos – simplified protocol



Kerberos – Advantages and limitations

- First practical SSO solution
- Centralized TTP (Trusted Third Party) model
- Uses only symmetric cryptography
- Requires Kerberos clients and servers + KDC
- Only suitable for organisations under common management (single domain)
- Does not scale to very large domains
- Not suitable for open environments (Internet)

Traditional Single Sign-On (SSO) Model



Examples: Kerberos, .net Passport

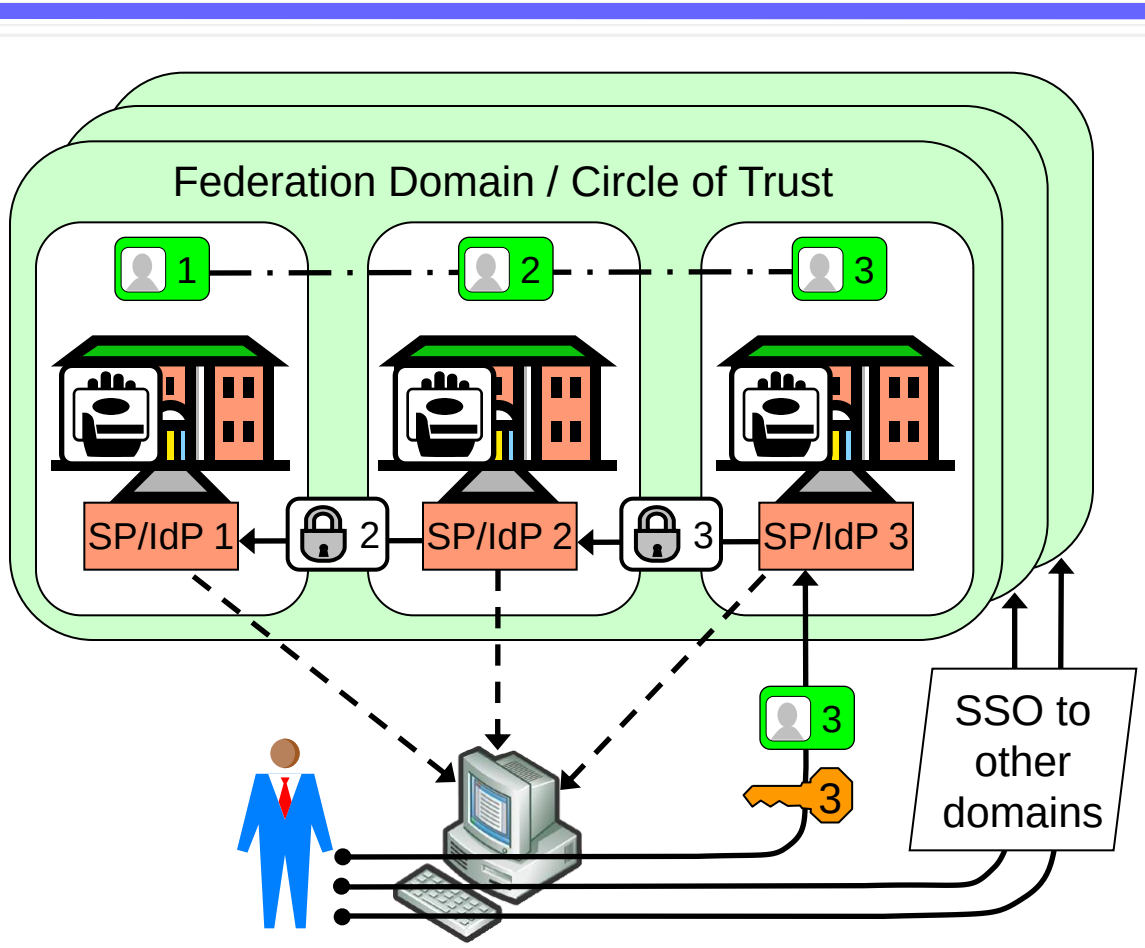
Legend:

- SP
- IdP
- Identity domain
- User identifier issued by IdP #
- Security assertion sent by IdP #
- Authentication token managed by IdP #
- Service logon
- Service provision

Traditional SSO

- Single authority/infrastructure that acts as identifier and credentials provider
- Single authority authenticates users on behalf of all SPs
- Advantages
  - Well suited for SPs under single management, e.g. within large private and government organisations
  - Good usability
- Disadvantages
  - Politically difficult to implement in open environments.
  - Who trusts authentication by other organisations?

Federated SSO model



Legend :

- SP
- IdP
- Identity domain
- # — User identifier issued by IdP #
- # — Authentication token managed by IdP #
- # — Security assertion sent by IdP #
- Service logon
- Service provision
- Identifier mapping

Examples: Liberty Alliance, SAML2.0, WS-Federation, Shibboleth

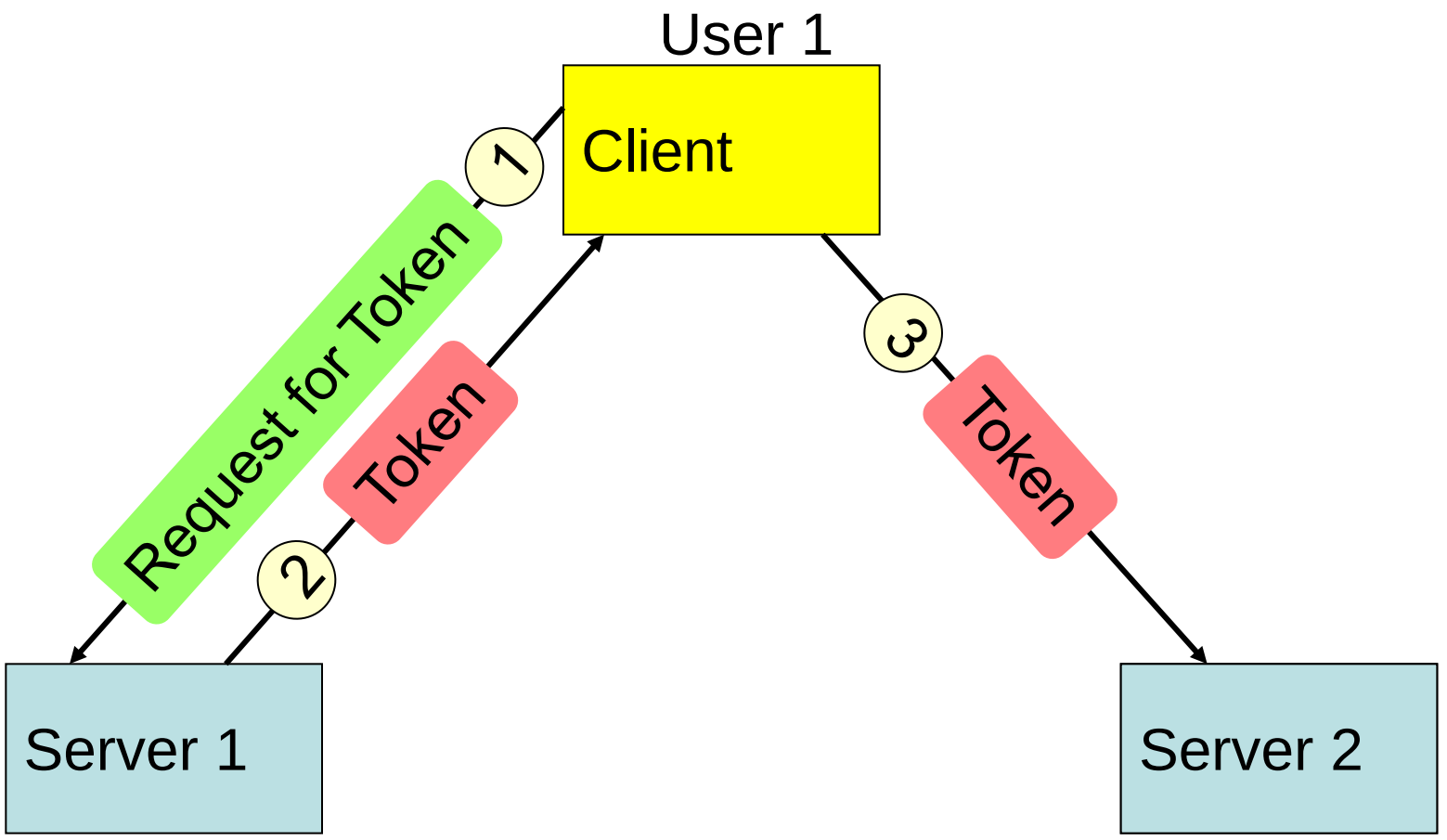
Federated SSO

- Identity Federation
  - A set of agreements, standards and technologies that enable a group of SPs to recognise user identities and entitlements from other SPs
  - Identifier (and credential) issuance as for the silo model
  - **Mapping** between a user's different unique identifiers
  - Authentication by one SP, communicated as security assertions to other SPs
  - Provides SSO in open environments

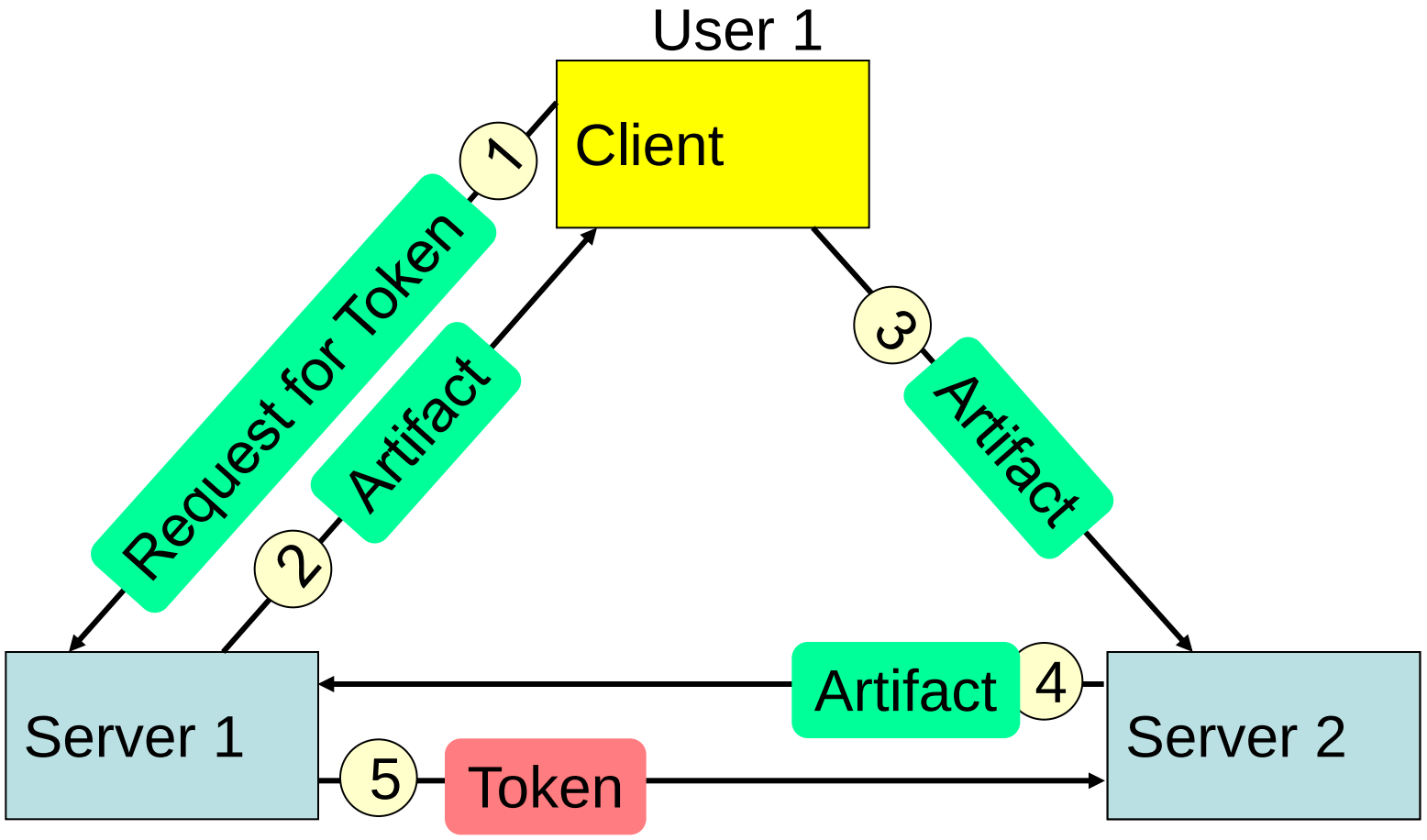
Federated SSO

- Advantages
  - Improved usability (theoretically)
  - Compatible with silo user-identity domains
  - Allows SPs to bundle services and collect user info
- Disadvantages
  - High technical and legal complexity
  - High trust requirements
    - E.g. SP1 is technically able to access SP2 on user's behalf
  - Privacy issues
  - Unimaginable for all SPs to federate,
    - multiple federated SSOs not much better than silo model

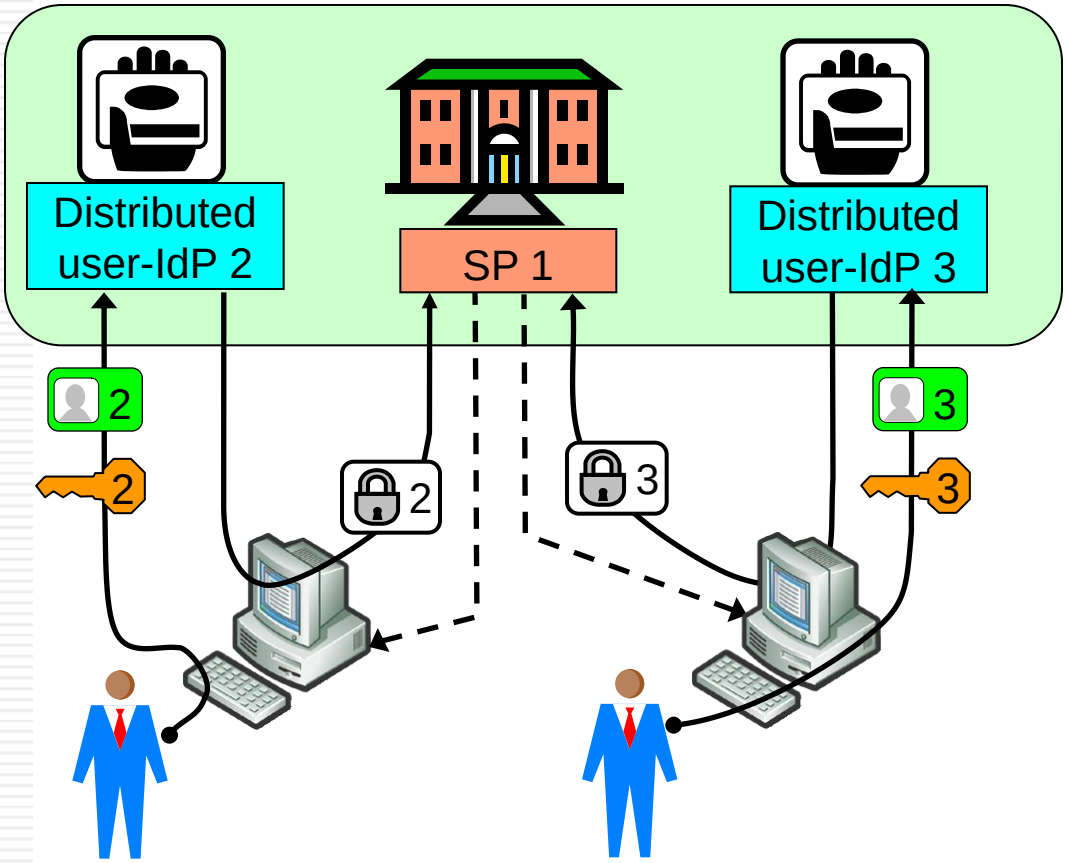
SAML identity federation protocol with security token sent vie browser



SAML identity federation protocol with security token sent through back channel



Common SSO identity model



Legend :

- SP
- IdP
- Common identity domain
- User identifier managed by IdP #
- Authentication token managed by IdP #
- Security assertion issued by IdP #
- Service logon
- Service provision

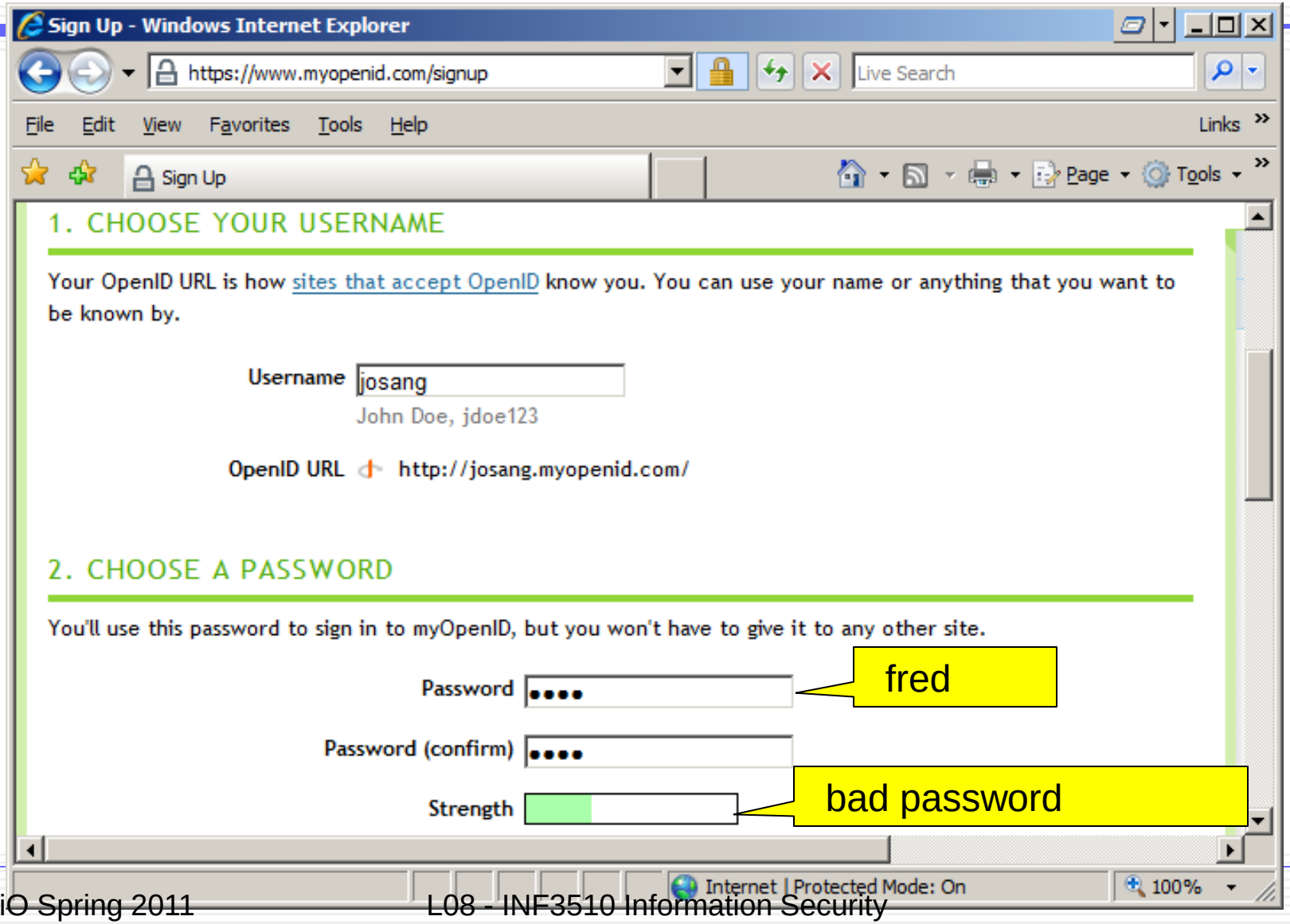
Example: OpenID

Common SSO identity model

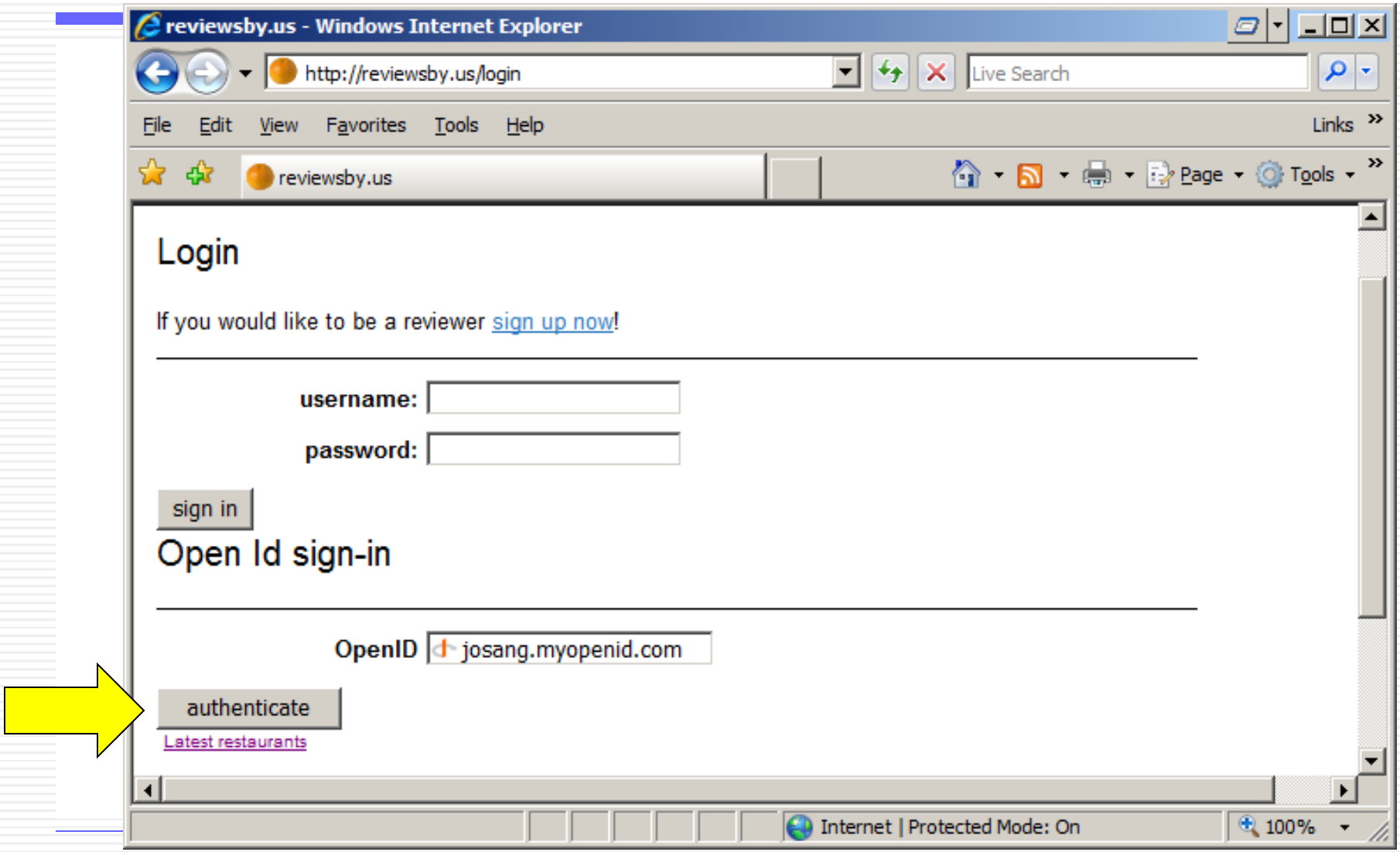
- Single common identifier name space
  - E.g. based on URIs or XRis
- Distributed assignment of identifiers
  - Each IdP controls its own domain name
  - Registers users under domain name
- Whoever controls a domain name can be IdP

- IdPs are involved for every service access
  - Collect info about service access

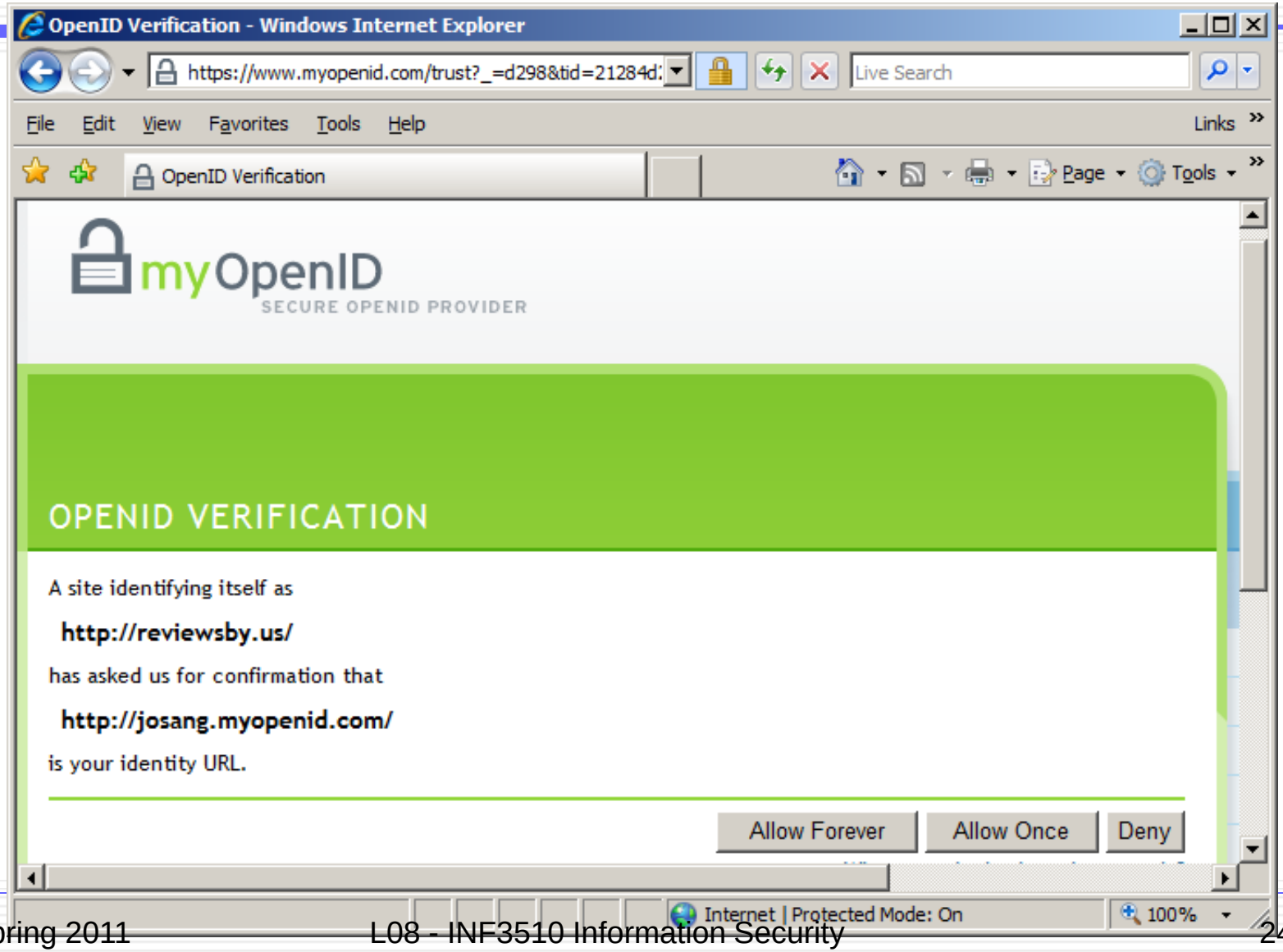
OpenID self registration



Service Access Without Password



First Time Sevice Access



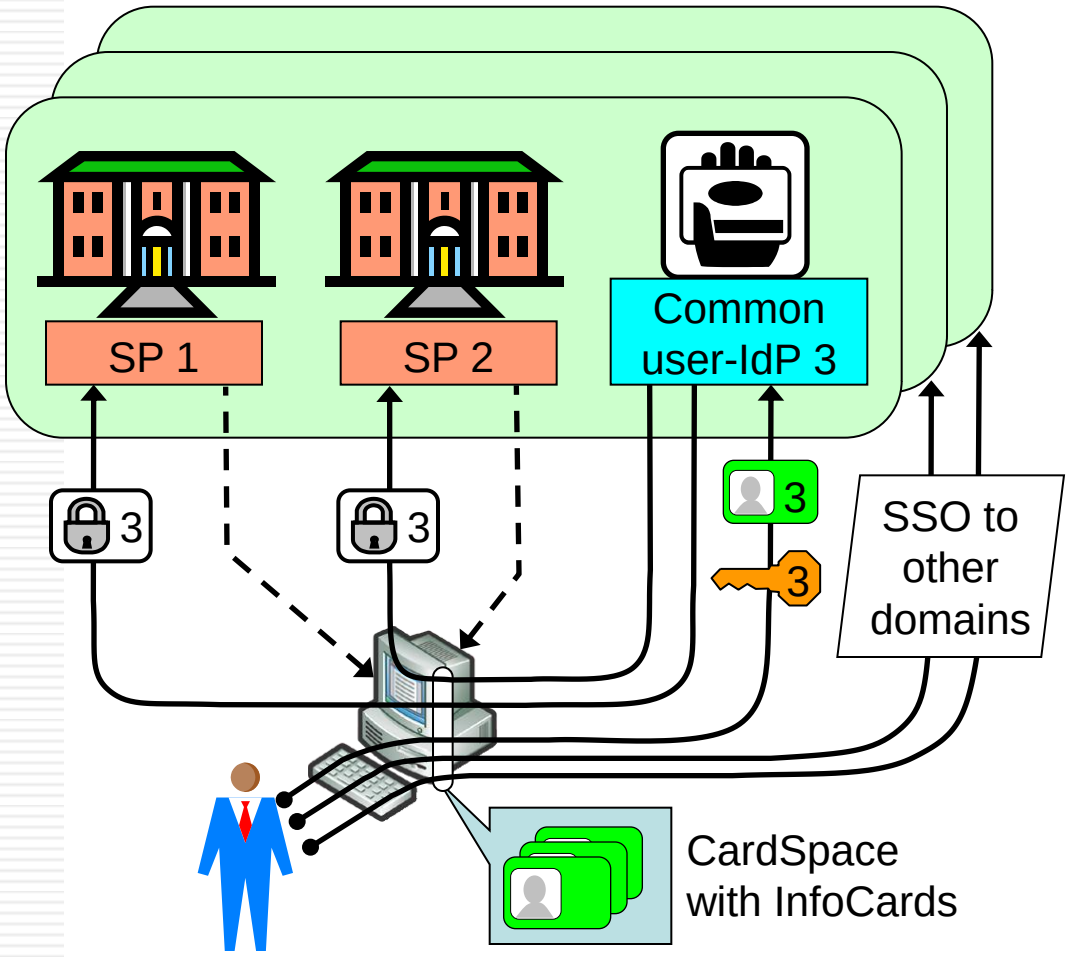
OpenID Characteristics

- Self registration
- ID Providers are not ”authorities”
- You can be your own ID Provider and Server
- Only supports AAL-1
- Not suitable for sensitive services
- Targets online services with AAL-1
- Open to multiple forms of abuse

OpenID Business Model

- For ID Providers
  - Collection of market data
  - Knows who uses which service
  - Fragmentation of ID Provider market is a threat
- For Service Providers (Relying Party)
  - Potentially more traffic and business
- For users
  - Avoid multiple identities
  - Avoids typing passwords
  - (Must still type OpenID identifier)

Microsoft's InfoCard model



Legend :

- SP
- IdP
- Identity domain
- User identifier managed by IdP #
- Authentication token managed by IdP #
- Security assertion issued by IdP #
- Service logon
- Service provision

InfoCard Model

- Requires intelligent browser
- Identities called ”InfoCard” stored in the browser’s ”CardSpace”
- Browser automatically relays security assertions
- SignOn to IdP subject to phising
- Supports multiple IdPs
- ”MS.Net Passport” renamed ”MS Live Space”
- CardSpace is compatible with dstributed common identity models, e.g. OpenID

SSO automation on server or client side

- SSO offers single **manual** authentication
- But involves repeated **automated** authentications
- SSO is therefore a logon-automation mechanism
- Where to put the automation system?
  - Both on server and client side: **Traditional SSO**
    - Kerberos, InfoCard
  - On server side only: **Federated SSO**
  - On client side only: **User Centric SSO**
    - Password wallets in software or hardware

User-Centric SSO

- Advantages
  - Improved usability
  - Compatible with silo identity domains
  - Low trust requirements
  - Good privacy protection
- Disadvantages
  - Does not allows SPs to control service bundling
  - Does not allow SPs to collect user information
  - Requires user-side software or hardware
  - Requires user education

SSO model suitability

- Federated SSO, well suited for
  - Large organisations
  - Government organisations
  - Closely associated organisations
  - Related Web service providers
- User-centric SSO, well suited for
  - Open networks
  - e-commerce
  - Unrelated Web services

The European IDA → IDABC → ISA

- IDA: *Interchange of Data between Administrations*
  - EU Work Programme 2000 – 2004
- IBAC: *Interoperable Delivery of European eGovernment Services to public Administrations, Business and Citizens*
  - EU Work Programme 2005 – 2009
- ISA: *Interoperable Solutions for European Public Administrations*
  - EU Work Programme 2010 – 2015

- Assurance Levels 1-4 defined in IDA auth. policy of 2004.
- Should include Level 0 to cover non-authenticating services and anonymous authentication

The STORK Project 2009 - 2011

- **S**ecure iden**T**ity acr**O**ss bo**R**ders lin**K**ed
- Cross-border recognition of eID
- Supports mobility of citizens
- Pilots:
  - Cross-border authentication platform
  - Safe use of the Internet for children using eID
  - Cross-border student mobility
  - Cross-border online delivery of documents
  - Change of address with eID

Four national identity federations

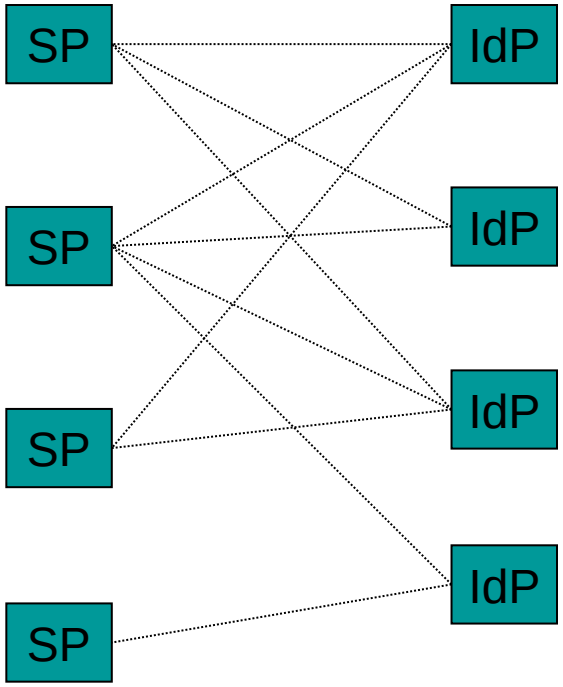
Haka (Finland): Operational (Shibboleth)

FEIDE (Norway): Operational (Moria, SAML2.0)

DK-AAI (Denmark): Piloting (A-Select)

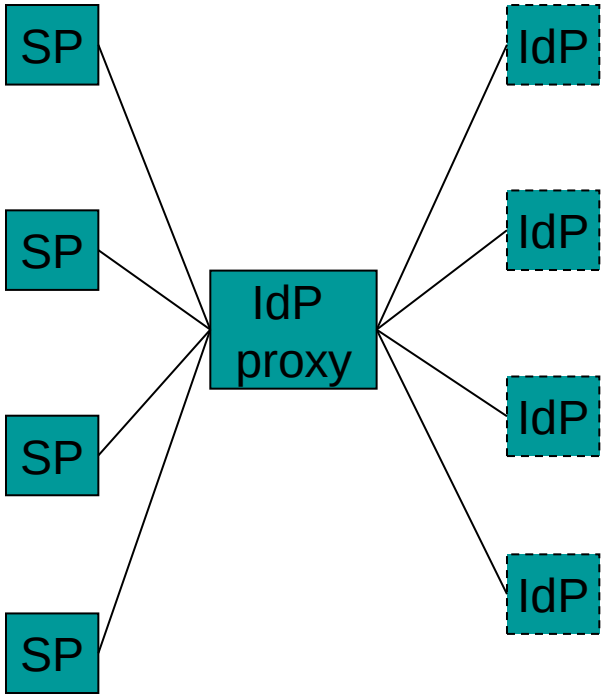
SWAMID (Sweden): Piloting (Shibboleth)

Technical shape of a federation: Distributed



- Model deployed by Haka (.fi), SWAMID (.se) and several other federations
- Pros
  - No single point of failure in the message flow
  - Costs of federation management low
- Cons
  - Hard to track errors and
  - Not well supported by commercial products

Technical shape of a federation: Centralized



- Model deployed by FEIDE (.no) and WAYF (.dk)
- Pros
  - A single point where to locate problems and introduce new features
  - Economics of scale
- Cons
  - A single point of failure
  - Everyone needs to trust the IdP in the middle

FEIDE (Felles Elektronisk Identitet)

- FEIDE is a system for Id management within the Norwegian national education sector.
- Users have only one username and password
- Users access web-services via a central log-in service
- Services are given what they need to know about the user
- Services are not given the users password/credential, only information about the user

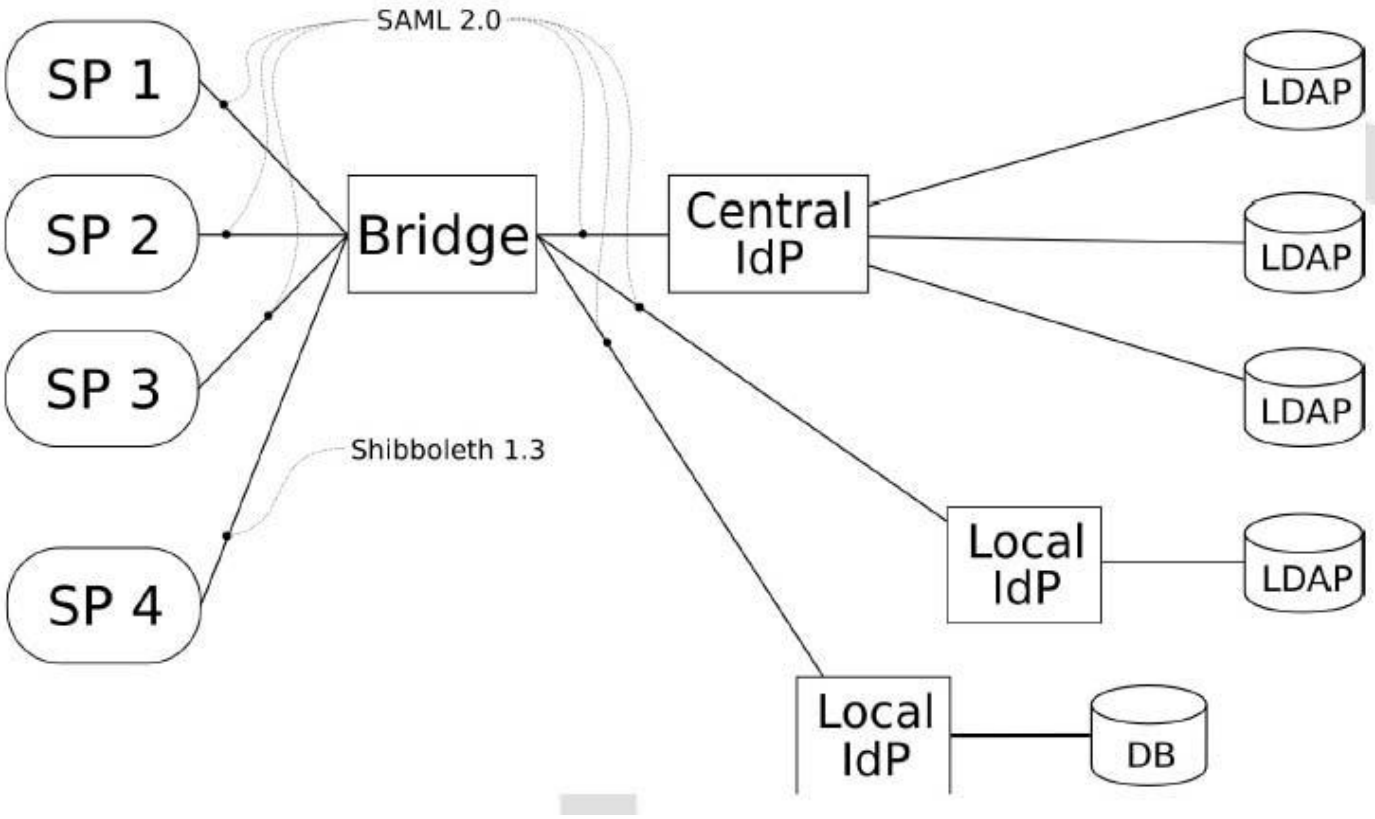
FEIDE (continued)

- FEIDE have formal agreements with the schools before they are connected
- The home organizations (schools) are responsible for the data about the users (correct and up-to-date)
- Home organizations decide themselves what services their users should be able to access via the central log-in service

FEIDE Technical Aspects

- Based on SAML 2.0
- Backend authenticate users by using LDAP
- One central identity provider (IdP) where service providers (SPs) are connected
- Single Sign On when going between services
- Single Log Out when logging out from a service

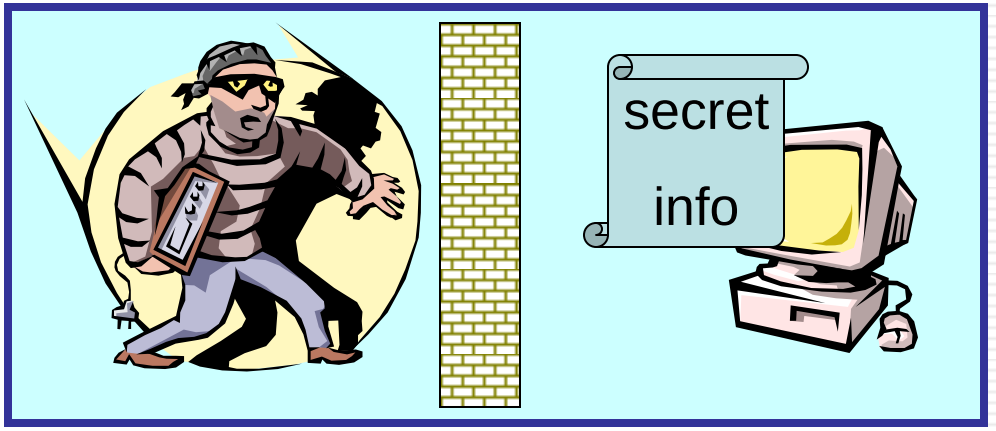
FEIDE Architecture



Introduction to access control

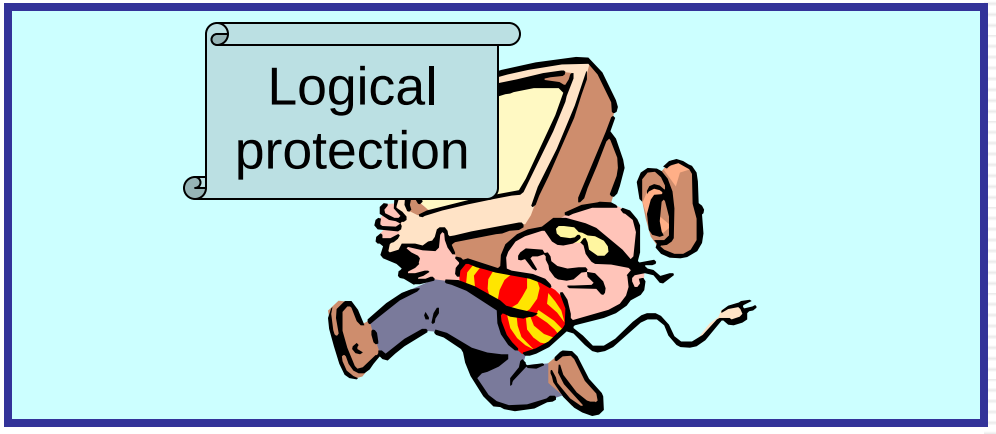
Physical Access Control:

(theme for lecture on physical security)



Logical Access Control:

(theme for this lecture)



Introduction to access control

- Access Control
  - controls how users and systems access other systems and resources
  - prevents unauthorized users access to resources
  - prevents authorized users from misusing resources
- Some information assets may be accessible to all, but access to some information assets should be restricted.
- Unauthorized access could compromise
  - Confidentiality
  - Integrity
  - Availability

  of information assets

Basic concepts

- Access control terminology:
  - Subjects
    - Entities requesting access to a resource
    - Examples: Person (User), Process, Device
    - Active
    - Initiate the request and is the user of information
  - Objects
    - Resources or entities which contains information
    - Examples: Disks, files, records, directories
    - Passive
    - Repository for information, the resources that a subject tries to access

Access modes

- Modes of access:
  - ***What access permissions (authorizations) should subjects have?***
- If you are authorized to access a resource, what are you allowed to do to the resource?
  - Example: possible access permissions include
    - read - observe
    - write – observe and alter
    - execute – neither observe nor alter
    - append - alter

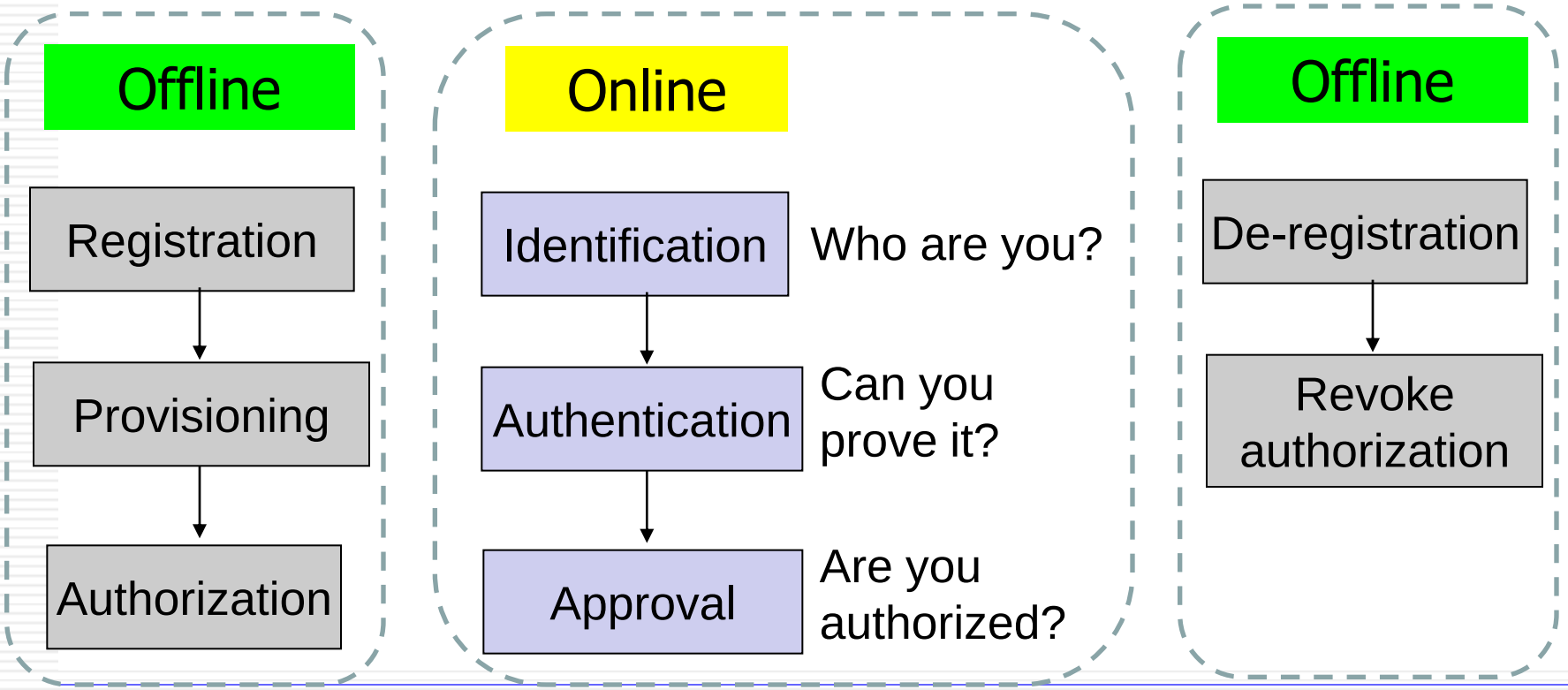
Access control phases

- Three phases of access control
  1. Registration and authorization phase
     a. Authorise subject by defining the AC policy
     b. Distribute access credentials/token to subject (provisioning)
     c. Change authorization whenever necessary
  2. Authentication and approval phase
     a. Authenticate subject
     b. Approve access as authorised by policy
     c. Monitor access
  3. Termination phase

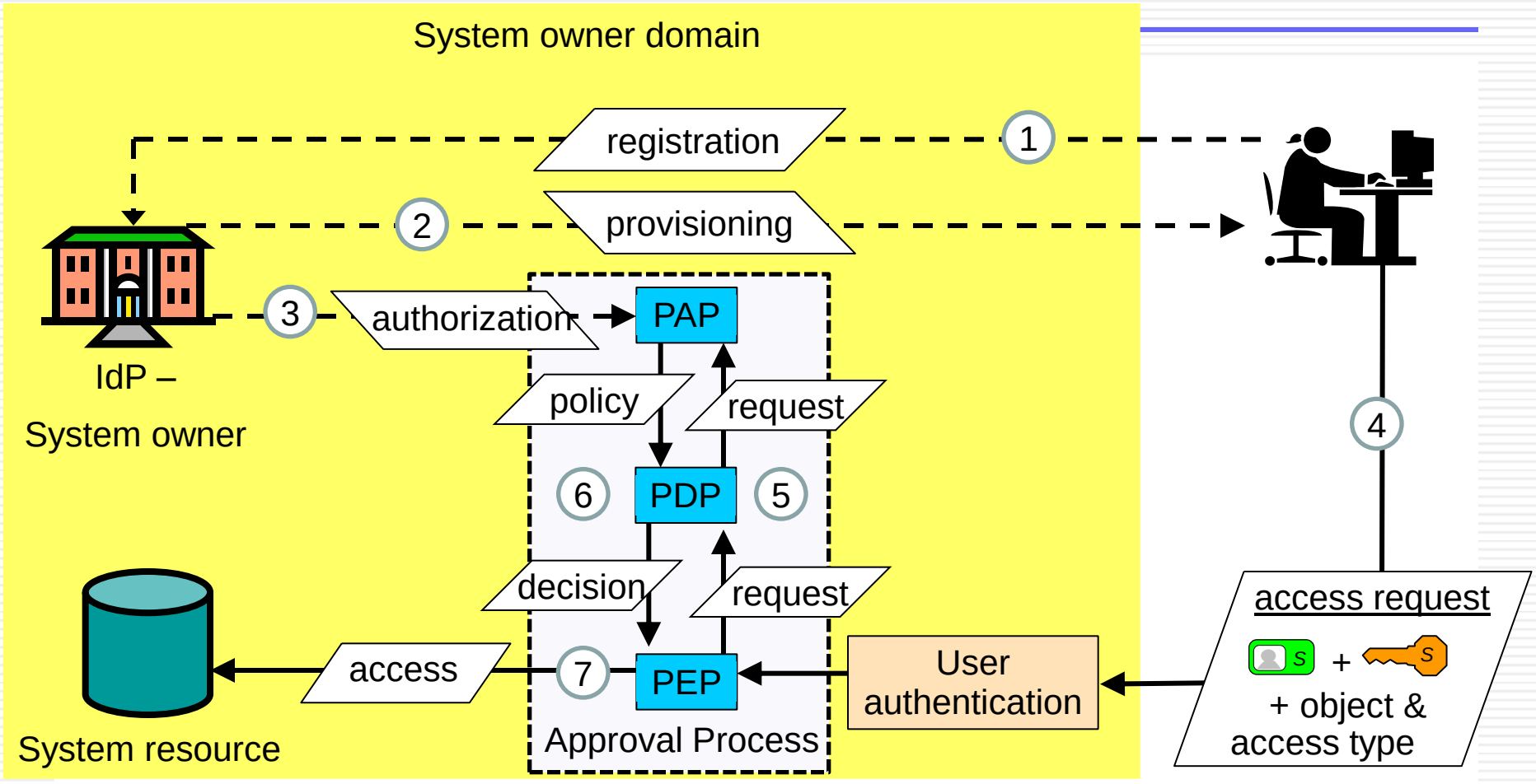
     De-register identity / Revoke authorization

Access Control Phases

- Access control has a procedure component in:
  - Offline: registration, provisioning, authorization (only once)
  - Online: controlling access during operations (repeatedly)



Access control and WS-Security concepts
(http://www.oasis-open.org/specs/index.php)



Basic concepts

- Access control approaches:
  - *How do you define which subjects can access which objects?*
- Three main approaches
  - Discretionary access control (DAC)
  - Mandatory access control (MAC)
  - Role-based access control (RBAC)

Basic concepts: DAC

- Discretionary access control: 2 interpretations:
  1. Access rights to an object or resource are granted at the discretion of the owner
     - e.g. security administrator, the owner of the resource, or the person who created the asset
     - DAC is discretionary in the sense that a subject with a certain access authorization is capable of passing that authorization (directly or indirectly) to any other subject.
  2. According to the Orange Book (TCSEC) DAC is implemented as a an ACL (Access Control List)
- Popular operating systems use DAC, both according to interpretation 1) and 2)

Basic concepts: ACL

- Access Control Lists (ACL)
  - Attached to an object
  - Provides an access rule for a list of subjects
  - Simple means of enforcing policy
  - Does not scale well
- ACLs can be combined into an access control matrix covering access rules for a set of objects

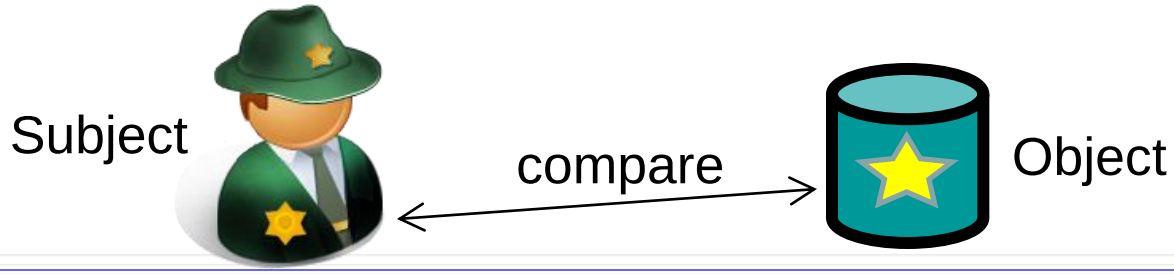
		Objects			
		O1	O2	O3	O4
Subjects	S1	rw	-	x	r
	S2	r	-	r	rw
	S3	-	x	-	-
	S4	rw	x	x	x

Basic concepts: MAC

- Mandatory access control: 2 interpretations:
  1. A central authority assigns access privileges
     - MAC is mandatory in the sense that users do not control access to the resources they create.
  2. According to the Orange Book (TCSEC) MAC is implemented with security labels
     - Example: Security clearance and classification levels
- A system-wide **set of rules** for objects and subjects specify permitted modes of access
  - (SE)Linux includes MAC

Basic concepts: Labels

- Security Labels can be assigned to subjects and objects
  - Represents a specific security level, e.g. “Confidential” or “Secret”
- Object labels are assigned according to sensitivity
- Subject labels are determined by the authorization policy
- Access control decisions are made by comparing the subject label with the object label according to rules
- The set of decision rules is a security model
  - Used e.g. in the Bell-LaPadula and Biba models (see later)



Basic concepts: Combined MAC & DAC

- Combining access control approaches:
  - A combination of mandatory and discretionary access control approaches is often used
    - Mandatory access control is applied first:
    - If access is granted by the mandatory access control rules,
      - then the discretionary system is invoked
    - Access granted only if both approaches permit
  - This combination ensures that
    - no owner can make sensitive information available to unauthorized users, and
    - ‘need to know’ can be applied to limit access that would otherwise be granted under mandatory rules

Basic concepts: RBAC

- Role based access control
  - Access rights are based on the role of the subject, rather than the identity
    - A role is a collection of procedures or jobs that the subject performs
    - Examples: user, administrator, student, etc
    - A subject could have more than one role, and more than one subject could have the same role
  - RBAC can be combined with DAC and MAC

Security Models Introduction

- In order to describe an access policy, it is necessary to describe the entities that the access policy applies to and the rules that govern their behaviour.
- A security model provides this type of description.
- Security models have been developed to describe access policies concerned with:
  - Confidentiality (Bell-LaPadula,Clark-Wilson,Brewer-Nash,RBAC)
  - Integrity (Biba, Clark-Wilson, RBAC)
  - Prevent conflict of interest (Brewer-Nash, RBAC)

The Bell-LaPadula Model

Important Point:

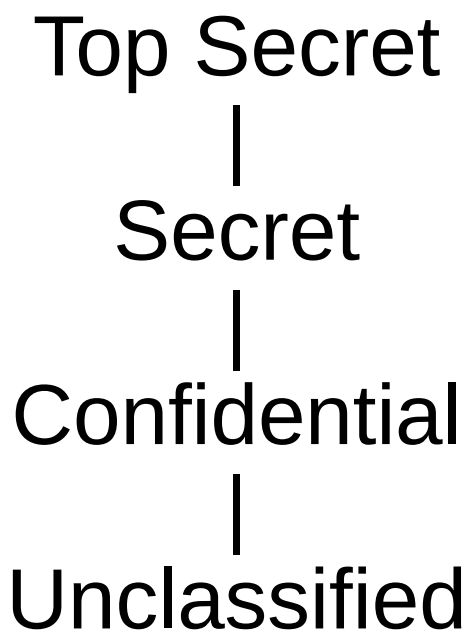
The Bell-LaPadula model has its origins in the military's need to maintain the confidentiality of classified information.

Bell-LaPadula Model: Overview

- While working for the Mitre Corporation in the 1970s, David E. Bell and Leonard J. LaPadula developed the Bell-La Padula Model in response to US Air Force concerns over the security of time-sharing mainframe systems.
- The Bell-LaPadula model focuses on the confidentiality of classified information – a Confidentiality-focussed Security Policy.
- The model is a formal state transition model of computer security policy that describes a set of access control rules enforced through the use of security labels on objects and clearances for subjects.

Diagram

Bell-LaPadula Model: Hierarchical Security Levels



- Security levels are typically used in military and national security domains
- Provide coarse-grained access control

Access Categories

- To implement the 'need to know' principle, define a set of non-ordered categories.
  - Subject and objects can have a set of categories in addition to their hierarchical security level;
- Example categories could be
  - Names of departments, such as:

Development – Production – Marketing – HR

- Not originally part of the Bell-LaPadula model

Bell-LaPadula Model: Security Labels

- Each subject and object has a Security Label
  - Subjects have a Maximum Security Label L^{SM}.
  - Subjects can use a Current Security Label $L^{SC} \leq L^{SM}$
  - Objects have a fixed Security Label L^{O}.
- The aim is to prevent subjects from accessing an object with a security label that is incompatible with the subject's security label.
- Subjects can chose to use a lower "current" label than their maximum label when accessing objects.

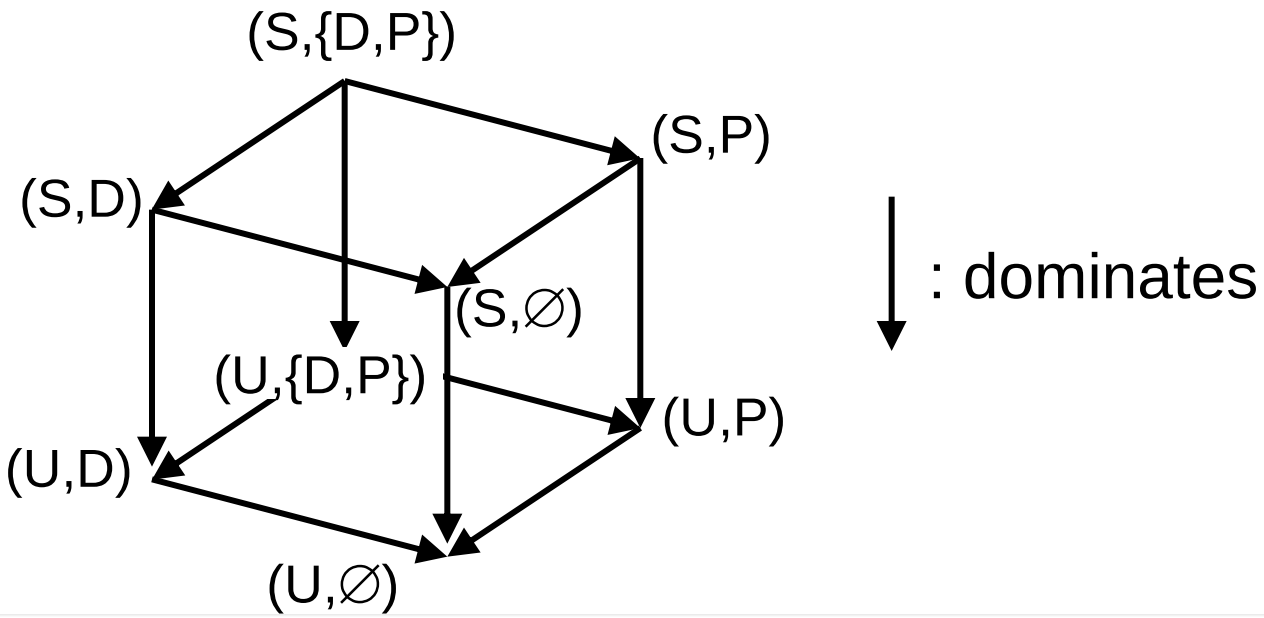
Bell La Padula Model: Security Labels and Domination

- Security labels that are assigned to subjects and objects can consist of two components
  - a hierarchical level, and
  - a set of categories (not originally part of Bell-LaPadula)
- Label dominance
  - Let label L_A = (h-level$_A$, category-set$_A$)
  - Let label L_B = (h-level$_B$, category-set$_B$).
  - Then Label$_A$ dominates Label$_B$ iff
    - h-level$_B$ is less than or equal to h-level$_A$ and
    - category-set$_B$ is a subset of category-set$_A$.

Partial Ordering of Labels

- Example: Define a label L = (h, c) where
  - h $\in$ hierarchical set H = {Unclassified, Secret} = {U, S}
  - c $\subseteq$ category set C = {Development, Production} = {D, P}

Partial ordering lattice



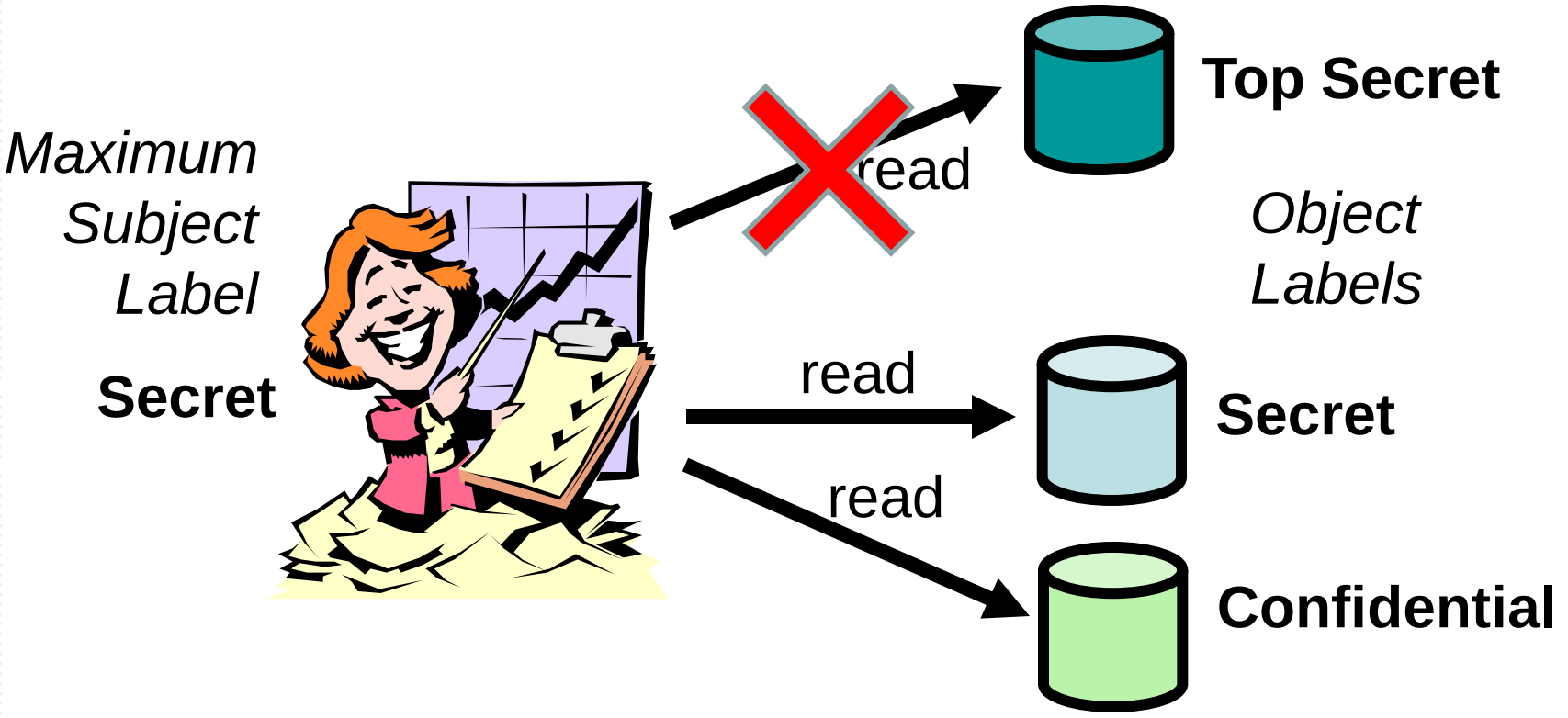
Definition of label dominance

- Labels defined as: $L = (h, c)$, $h \in H$ and $c \subseteq C$
  H: set of hierarchical levels, C: set of categories
- Example labels: $L_A = (h_A, c_A)$, $L_B = (h_B, c_B)$,
- Dominance: $L_A \geq L_B$ iff $(h_B \leq h_A) \wedge (c_B \subseteq c_A)$
  - In case $L_A = L_B$ then also $L_A \geq L_B$ and $L_B \geq L_A$
- Non-dominance cases: $L_A \not\geq L_B$
  - $(h_B > h_A) \wedge (c_B \subseteq c_A)$; insufficient security level
  - $(h_B \leq h_A) \wedge (c_B \not\subset c_A)$; insufficient category set
  - $(h_B > h_A) \wedge (c_B \not\subset c_A)$; insufficient level and category

Bell-LaPadula Model: Security Properties

- In each state of a system the Bell-LaPadula model maintains three security properties:
  - ss-property (simple security)
  - * -property (star)
  - ds-property (discretionary security)

Bell-LaPadula Model: SS-Property: No Read Up

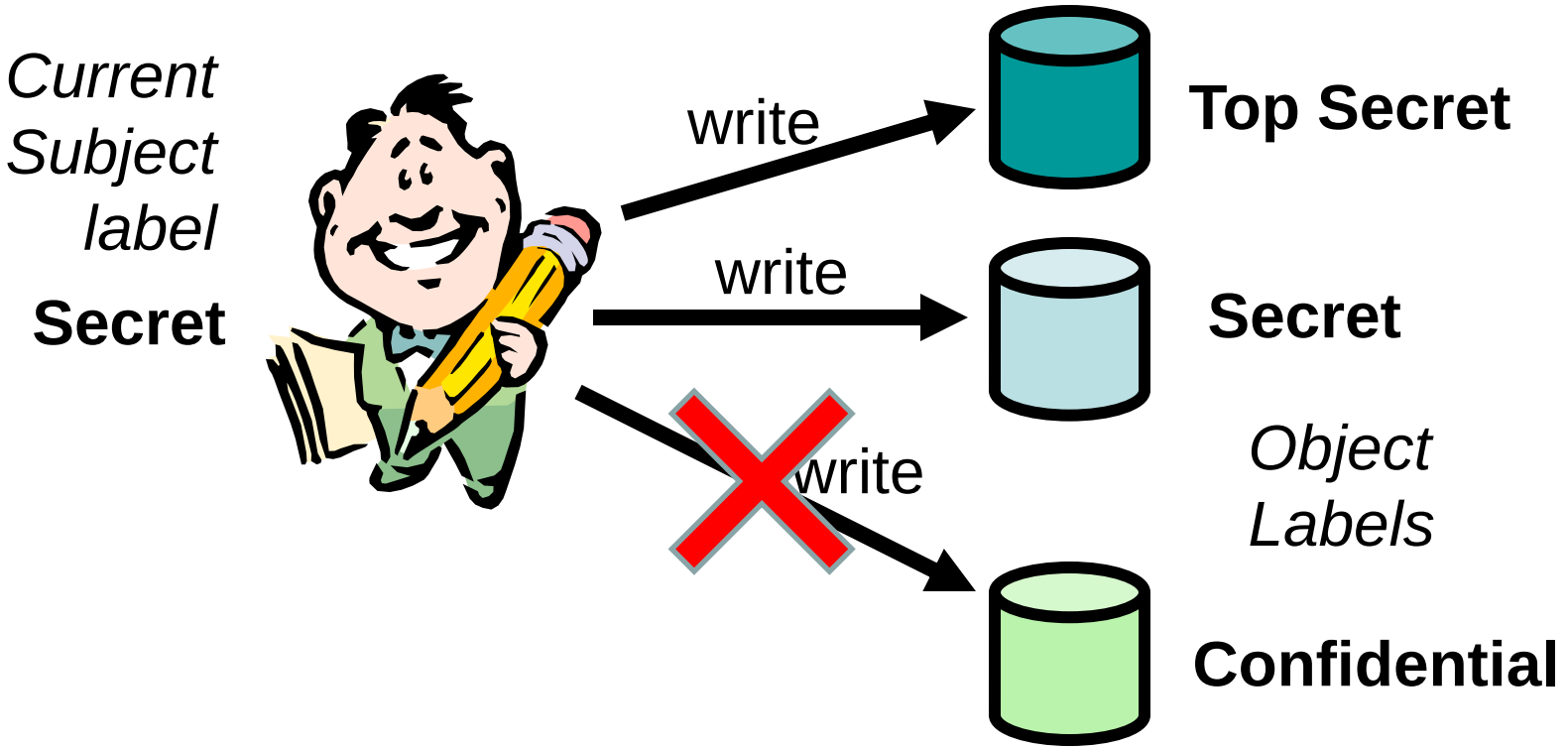


Bell-LaPadula Model: *-Property: No Write Down

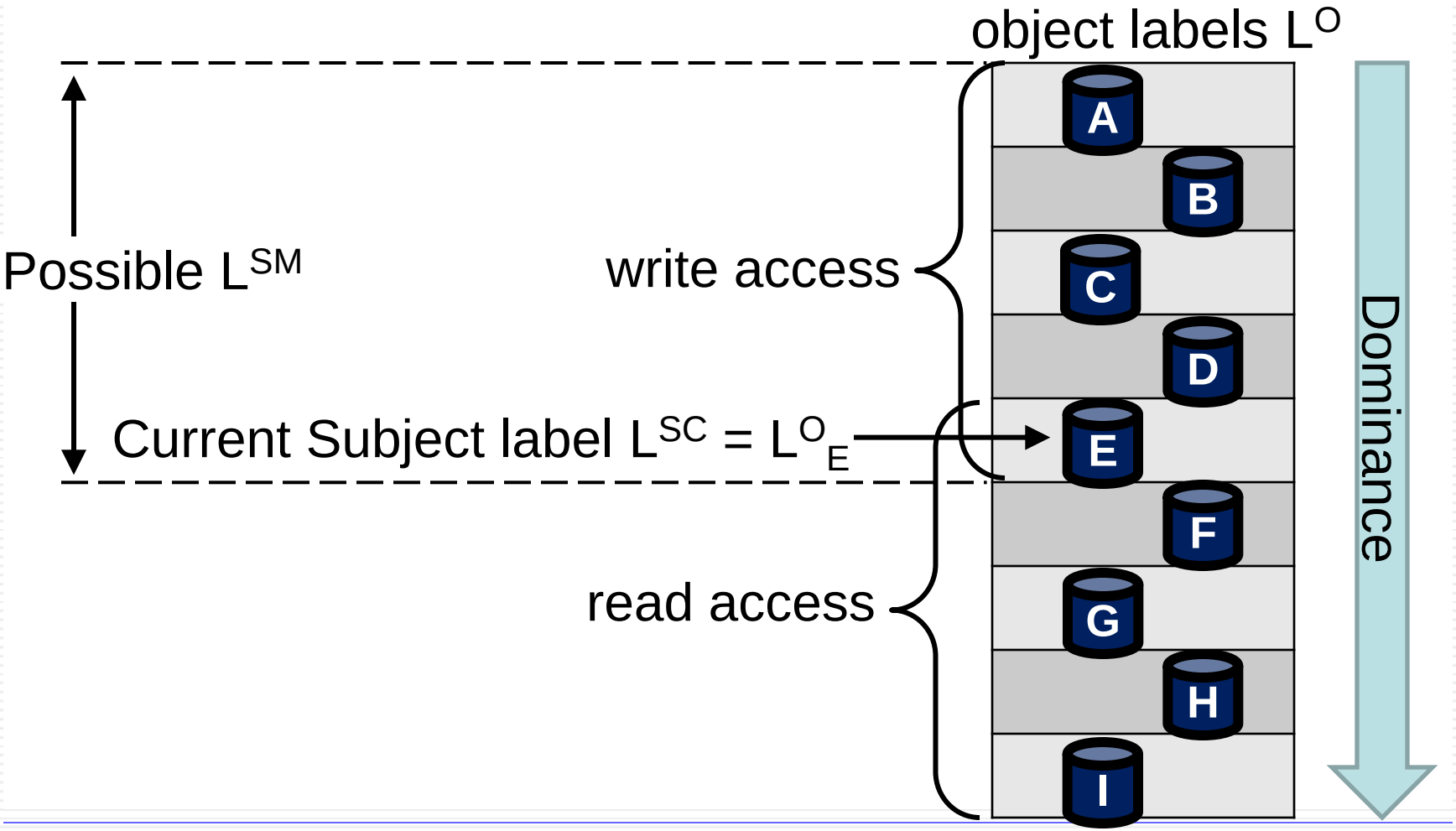
- Subjects working on information/tasks at a given label should not be allowed to write to a lower level because this could leak sensitive information.
- For example, you should only be able write to files with the same label as your label, or
- you could also write to files with a higher label than your label, but you should not be able to read those files.

Diagram

Bell-LaPadula Model: *-Property: No Write Down



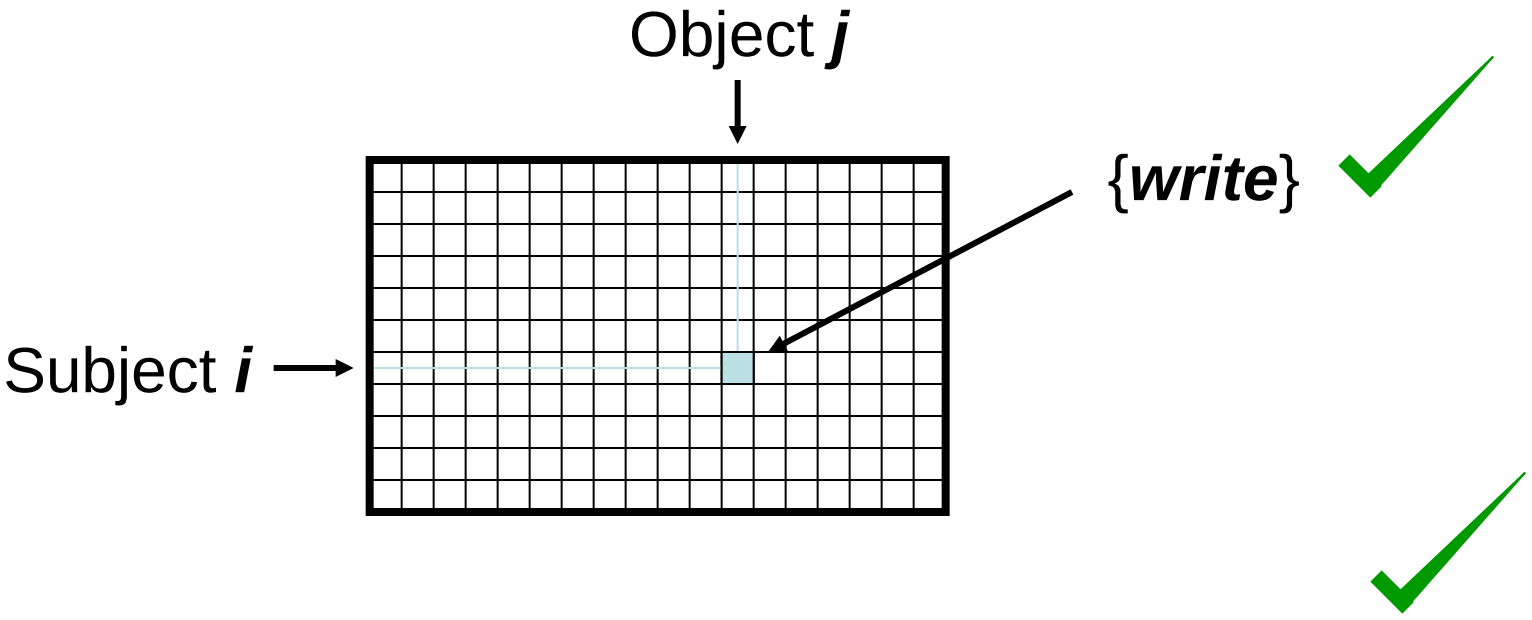
Bell-LaPadula label relationships



Diagram

Bell-LaPadula Model: DS Property: Matrix Entry

- M(i,j) satisfies current access request



? Access request : {subject = ***i***, object = ***j***, access = ***write***}

Bell-LaPadula Model: DS Property: Matrix Entry

- The ds-property (discretionary security property) is a Bell-LaPadula security model rule that demands that the current access by subject S to object O is permitted by the current access permission matrix M.
- This was the original method to enforce need-to-know in Bell-LaPadula.

The Biba Model for Integrity

- In Biba, subjects and objects have integrity labels
- *The Biba Simple Integrity Axiom* states that a subject at a given level of integrity must not read an object at a lower integrity level (**no read down**).
- *The Biba * (star) Integrity Axiom* states that a subject at a given level of integrity must not write to any object at a higher level of integrity (**no write up**).
- Opposite to Bell-LaPadula
- Combining Biba and Bell-LaPadula results in a model where subjects can only read and write at their own level

The Brewer-Nash Chinese Wall Model

Brewer-Nash model: Overview

- The Brewer-Nash model is a confidentiality model for the commercial world.
  - In a consultancy-based business, analysts have to ensure that no conflicts of interest arise in respect to dealings with different clients.
  - A conflict of interest is a situation where someone in a position of trust has competing professional and/or personal interests and their ability to carry out their duties and responsibilities objectively is compromised or may be seen to be compromised.
- Rule: There must be no information flow that causes a 'conflict of interest'.

Brewer-Nash model: Sanitized and Unsanitized Information

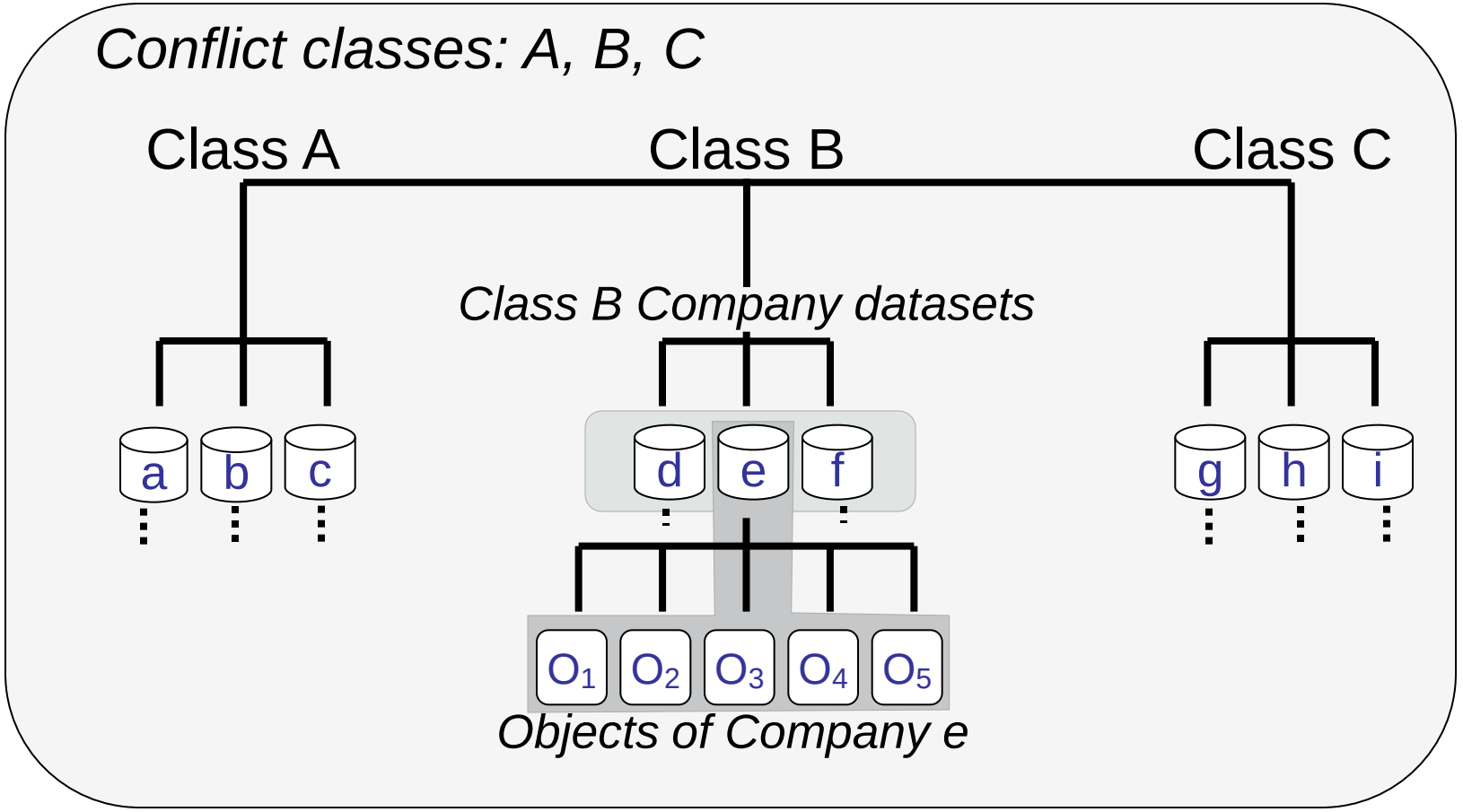
- Assume that a consultancy-based business has confidential information pertaining to individual companies that are its clients.
  - Information that can be identified as belonging to a particular company is deemed to be unsanitized.
  - Information that cannot be identified as belonging to a particular company is deemed to be sanitized.
  - Also, where information is held regarding a company but it is not confidential (already public knowledge say), this information is not subject to the policy implemented by this model.
- The Brewer-Nash model is concerned with the flow of unsanitized information.
  - Sanitized information flow is not of concern in this model.

Brewer-Nash model: Objects, Datasets & Conflict Classes

- Objects:
  - Individual items of information belonging to a single corporation are stored as objects;
  - Each object has a security label
  - Security labels contain information about which company the object belongs to, and the ‘conflict of interest’ class the object belongs to.
- Company datasets:
  - All objects which concern the same corporation are grouped together into a company dataset;
- Conflict classes:
  - All company datasets whose corporations are in competition are grouped together into the same conflict of interest class.

Diagram

Brewer-Nash model: Example conflict classes and companies



Brewer-Nash ss-property (simple security)

- An analyst who has already accessed an object of company *e* in conflict class *B* cannot now access any objects of other companies in conflict class *B*
  - Because companies in the same conflict class are in competition with each other. Accessing information of multiple companies in the same conflict class would lead to a conflict of interest.
- The analyst can access an object of any company in conflict class *A* or *C*
  - Insider information about companies in class A or C does not represent a conflict of interest with companies in class B because they are not in direct competition with each other

Brewer-Nash model: Star (*) Security Property

- Suppose two analysts, user 1 and user 2, have the following access:
  - User 1 has access to information about company **e** in class B and company **a** in class A
  - User 2 has access to information about company **d** in class B and company **a** in class A
- If user 1 reads information from company **e** and writes it to a company **a** object, then user 2 has access to company **e** information.
- This should not be permitted because of the conflict of interest between company **e** and company **d**.

Brewer-Nash model: Star (*) Security Property

- Write access is only permitted if:
  - access is permitted by the **ss** rule, and
  - no object can be read which is in a different company dataset from the one for which write access is requested and contains unsanitized information.

- In other words, write access is granted only if no other object (with unsanitized data) can be currently read which is in a different company dataset (in any conflict class)

The Clark-Wilson Model

Clark Wilson model: Overview

- The Clark-Wilson Security model is an *integrity* model for the commercial environment.
- There is an emphasis on controlling transaction processing.
- The Clark-Wilson Security model provides a formal model for commercial integrity
  - The model attempts to prevent unauthorised modification of data, fraud and errors.

Clark Wilson model: Overview

- The Clark-Wilson Security model attempts to follow the conventional controls used in bookkeeping and auditing through certification and enforcement.
- Data is divided into two types
  - Unconstrained data items (UDI)
  - Constrained data items (CDI)
- CDIs cannot be accessed directly by users - they must be accessed through a transformation procecure (TP)
- In certain circumstances UDI may become CDI

Clark-Wilson model: System Integrity

- Internal consistency:
  - Is the internal state of the system consistent at all times?
  - This can be enforced by integrity verification procedures (IVPs)
  - The IVPs certify that the CDIs are in a valid state
  - The TPs must preserve state validity

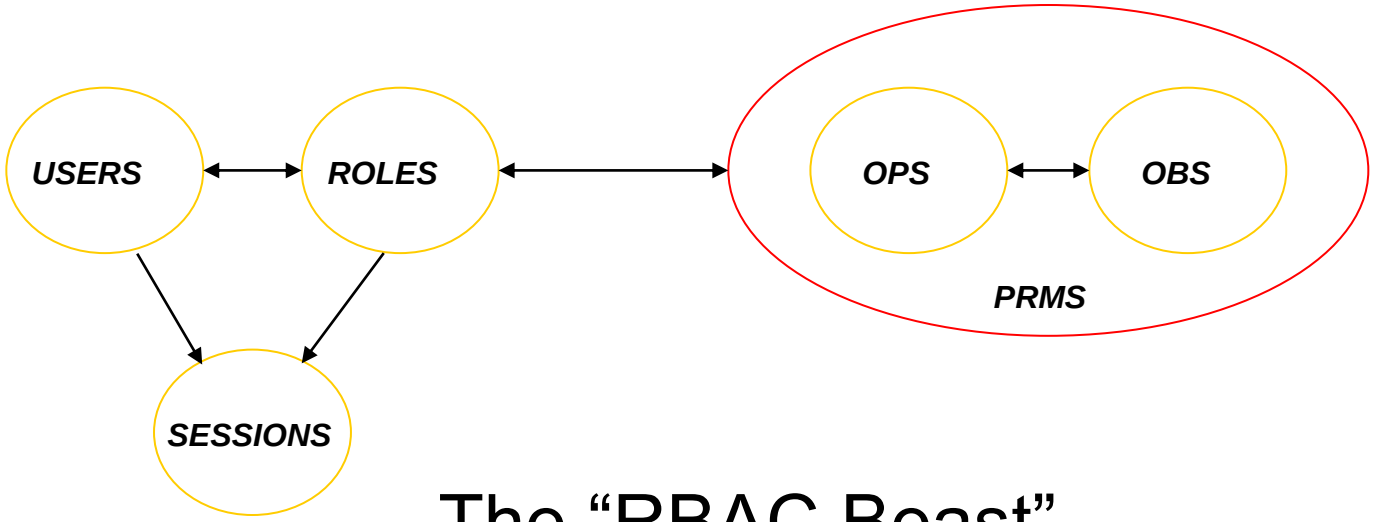
Clark Wilson model: Security Requirements Overview

- Every user must be identified and authenticated.
- Each data item can only be manipulated by a particular set of allowed programs.
- Each user can run only a particular set of programs.
- Separation of duty and well-formed transaction rules must be enforced by the system.
- Auditing log must be maintained.

The RBAC Model
Role Based Access Control

Role-Based Access Control

- A brief introduction
  - Based on Proposed NIST Standard for Role-Based Access Control
  - http://csrc.nist.gov/rbac/sandhu-ferraiolo-kuhn-00.pdf

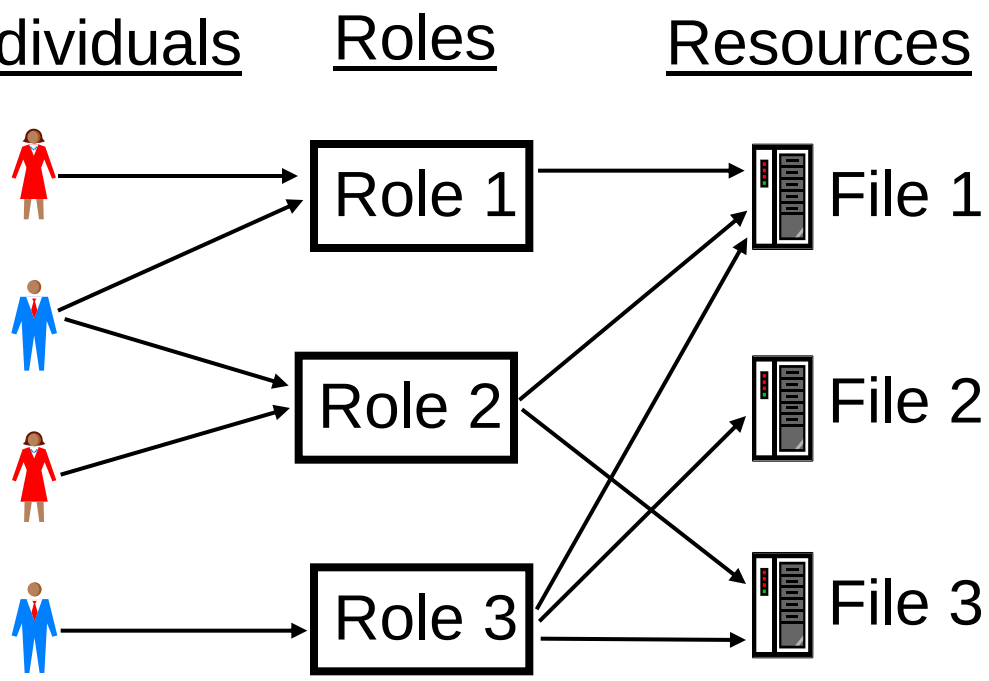


The “RBAC Beast”

RBAC rationale

- A user has access to an object based on the assigned role.
- Roles are defined based on job functions.
- Permissions are defined based on job authority and responsibilities within a job function.
- Operations on an object are invocated based on the permissions.
- The object is concerned with the user's role and not the user.

RBAC Flexibility



User's change frequently, roles don't

- RBAC can be configured to do MAC
- RBAC can be configured to do DAC

RBAC Privilege Principles

- Roles are engineered based on the principle of least privilege .
- A role contains the minimum amount of permissions to instantiate an object.
- A user is assigned to a role that allows him or her to perform only what's required for that role.
- All user with the same role have the same permissions.

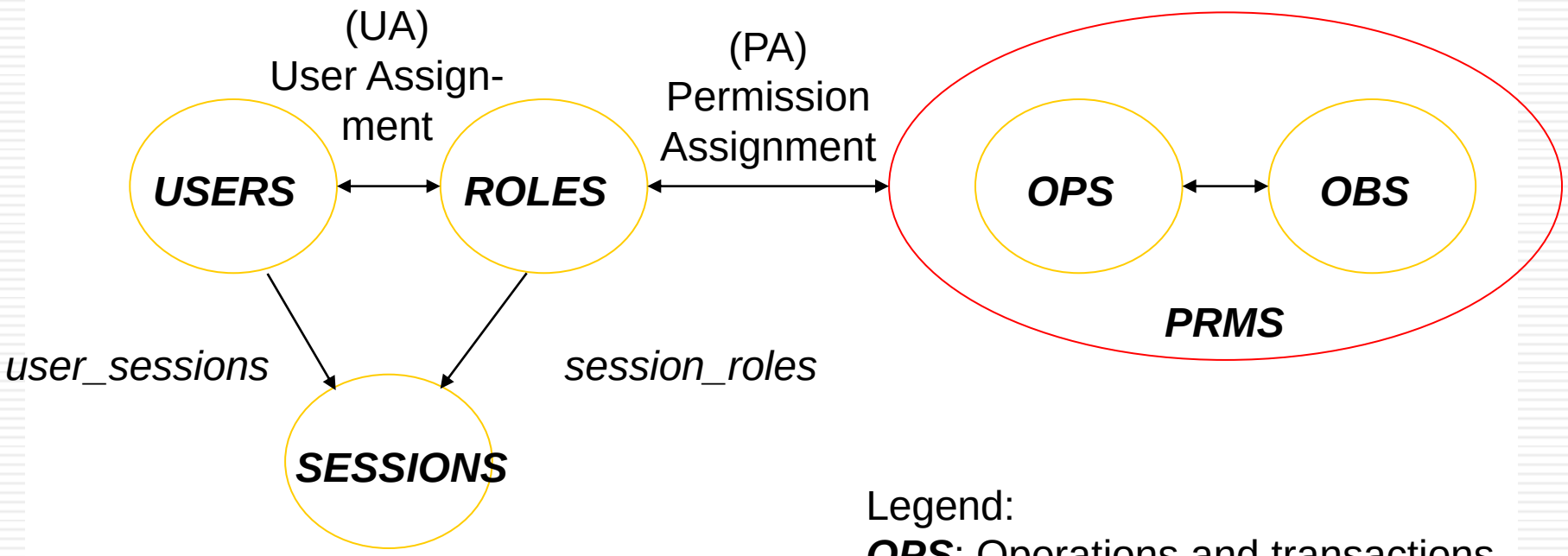
RBAC Framework

- Core Components
- Constraining Components
  - Hierarchical RBAC
    - Allows roles to be defined in a hierarchy, and role inheritance
  - Constrained RBAC
    - Can prevent conflict of interest in two ways
    - SSD (Static Separation of Duties) prevents assignment of conflicting roles
    - DSD (Dynamic Separation of Duties) allows assignment of conflicting roles, but prevents their simultaneous invocation

RBAC Core Components

- Defines:
  - USERS
  - ROLES
  - OPERATIONS (*ops*)
  - OBJECTS (*obs*)
  - User Assignments (*ua*)
    - assigned_users
  - Permissions (*prms*)
    - Assigned Permissions
    - Object Permissions
    - Operation Permissions
  - Sessions
    - User Sessions
    - Available Session Permissions
    - Session Roles

Core RBAC



Legend:
OPS: Operations and transactions
OBS: Objects, databases, files
PRMS: Permissions

UA (user assignment) and PA (permission assignment)

USERS set

Alice

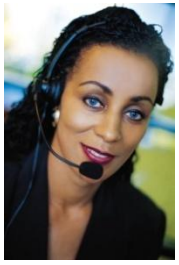
Bob

A user can be assigned to one or more roles

A role can be assigned to one or more users

ROLES set

Developer



Help Desk Rep

A role can be assigned to one or more prms

A prms can be assigned to one or more roles

PRMS set

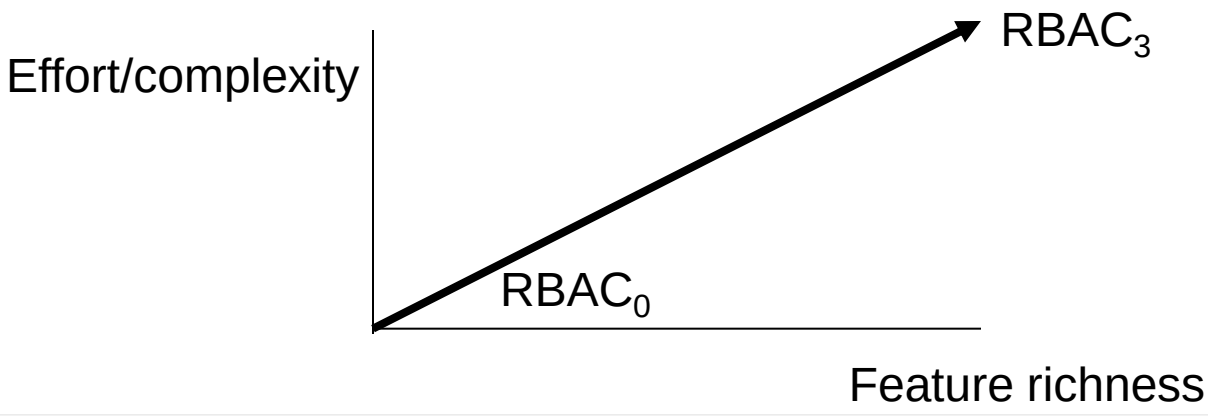
Create
Delete
Drop

View
Update
Append

RBAC Models

Core RBAC

Models	Hierarchical	Constrained
$RBAC_0$	No	No
$RBAC_1$	Yes	No
$RBAC_2$	No	Yes
$RBAC_3$	Yes	Yes



RBAC Operational Aspects

- System Level Functions
  - Creation of user sessions
  - Role activation/deactivation
  - Constraint enforcement
  - Access Decision Calculation
- Administrative Operations
  - Create, Delete, Maintain elements and relations
- Implementation challenge
  - Large number of different roles can become a problem in practical implementations

End of lecture