INF3510 Information Security
University of Oslo
Spring 2016

Lecture 9

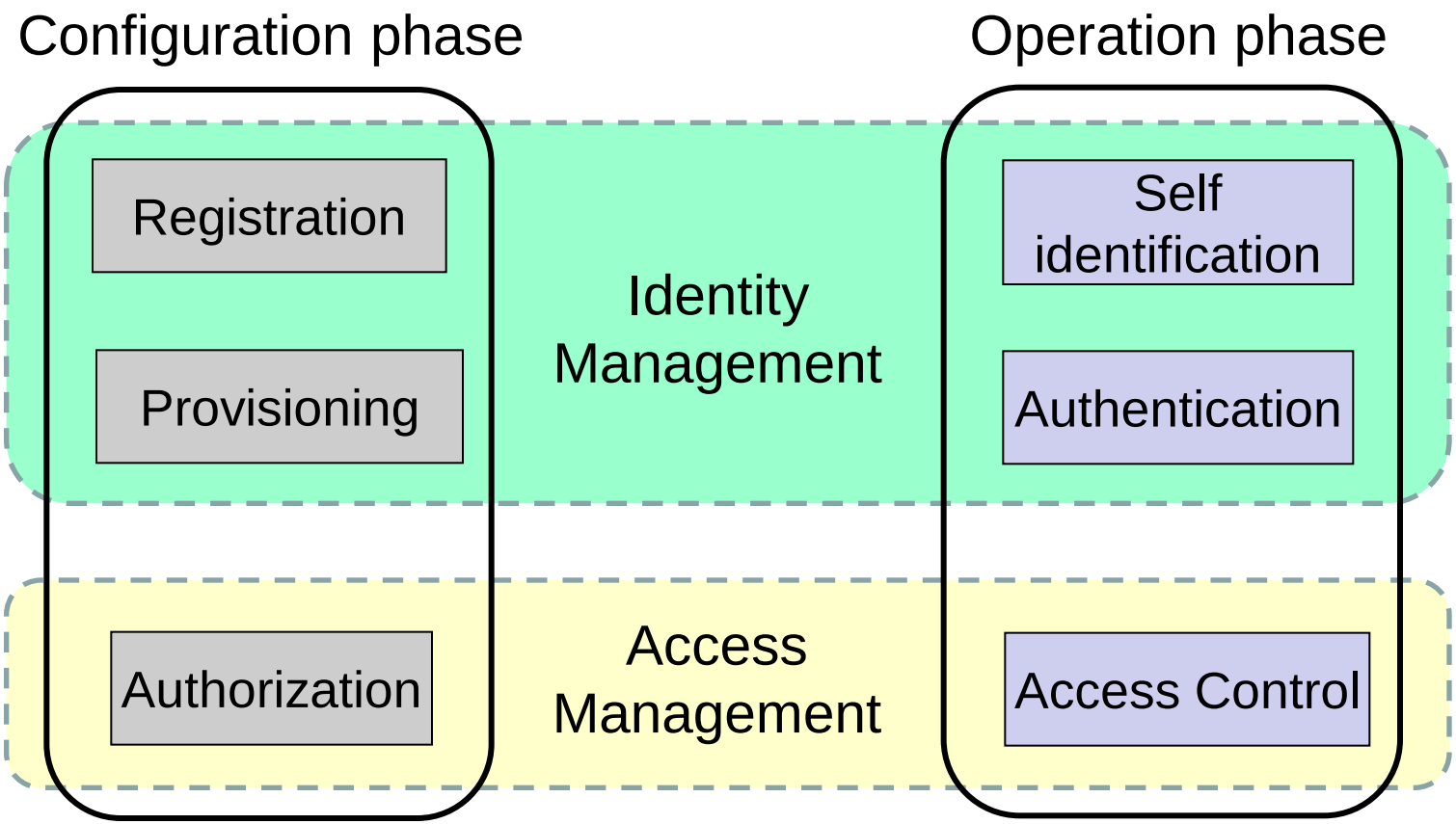
Identity Management and Access Control

University of Oslo
Spring 2016

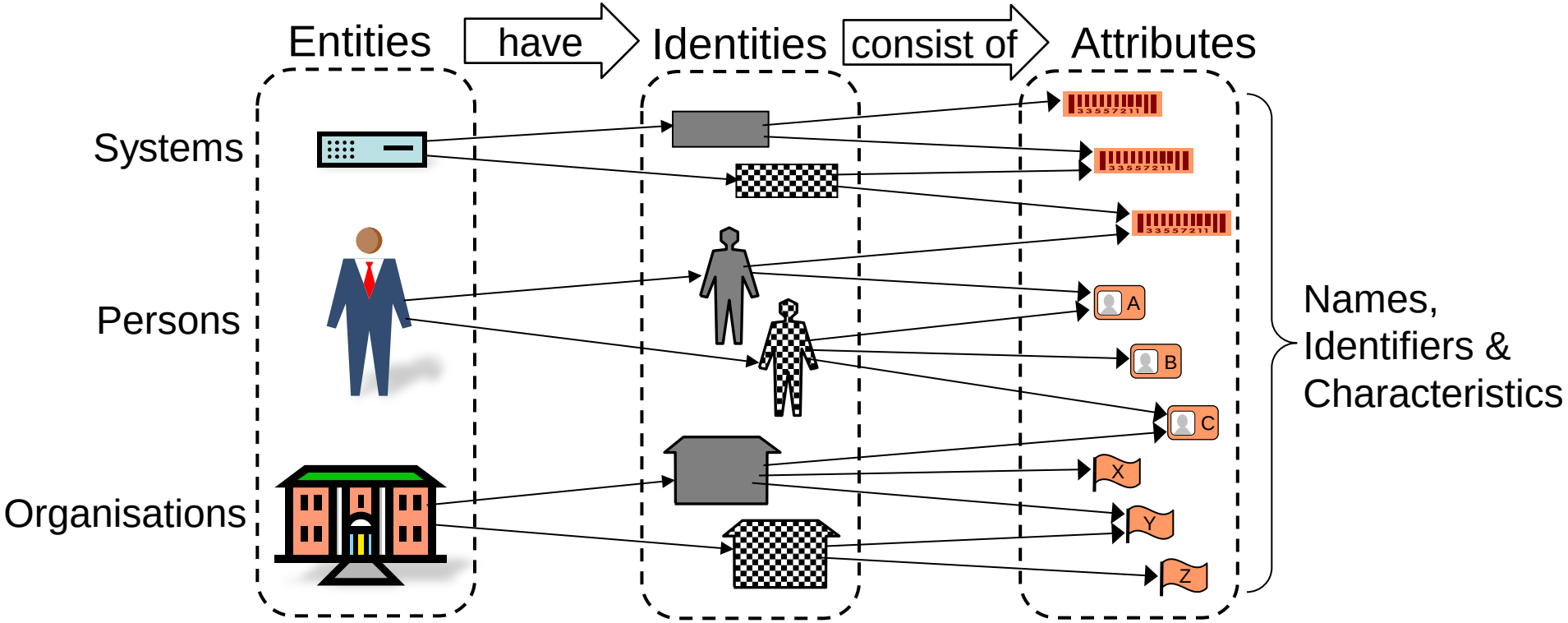
Outline

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

IAM
Identity and Access Management



The concept of identity



Concepts related to identity

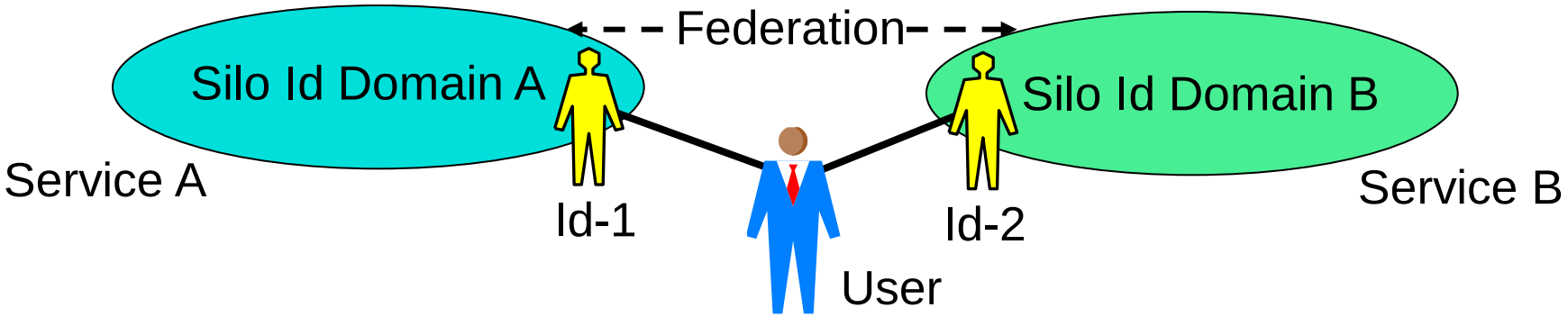
- Entity
  - A person, organisation, agent, system, etc.
- Identity
  - A set of names / attributes of entity in a specific domain
  - An entity may have identities in multiple domains
  - An entity may have multiple identities in one domain
- Digital identity
  - Digital representation of names / attributes in a way that is suitable for processing by computers
- Names and attributes of entity
    - Can be unique or ambiguous within a domain
    - Transient or permanent, self defined or by authority, interpretation by humans and/or computers, etc

Identity

- Etymology (original meaning of words)
  - *“identity” = “same one as last time”.*
- “First-time” authentication is not meaningful
  - because there is no “previous time”
- Authentication requires a first time registration of identity in the form of a name within a domain
- Registration can be take two forms:
  - pre-authentication, from previous identity, e.g. passport
  - creation of new identity, e.g. New born baby

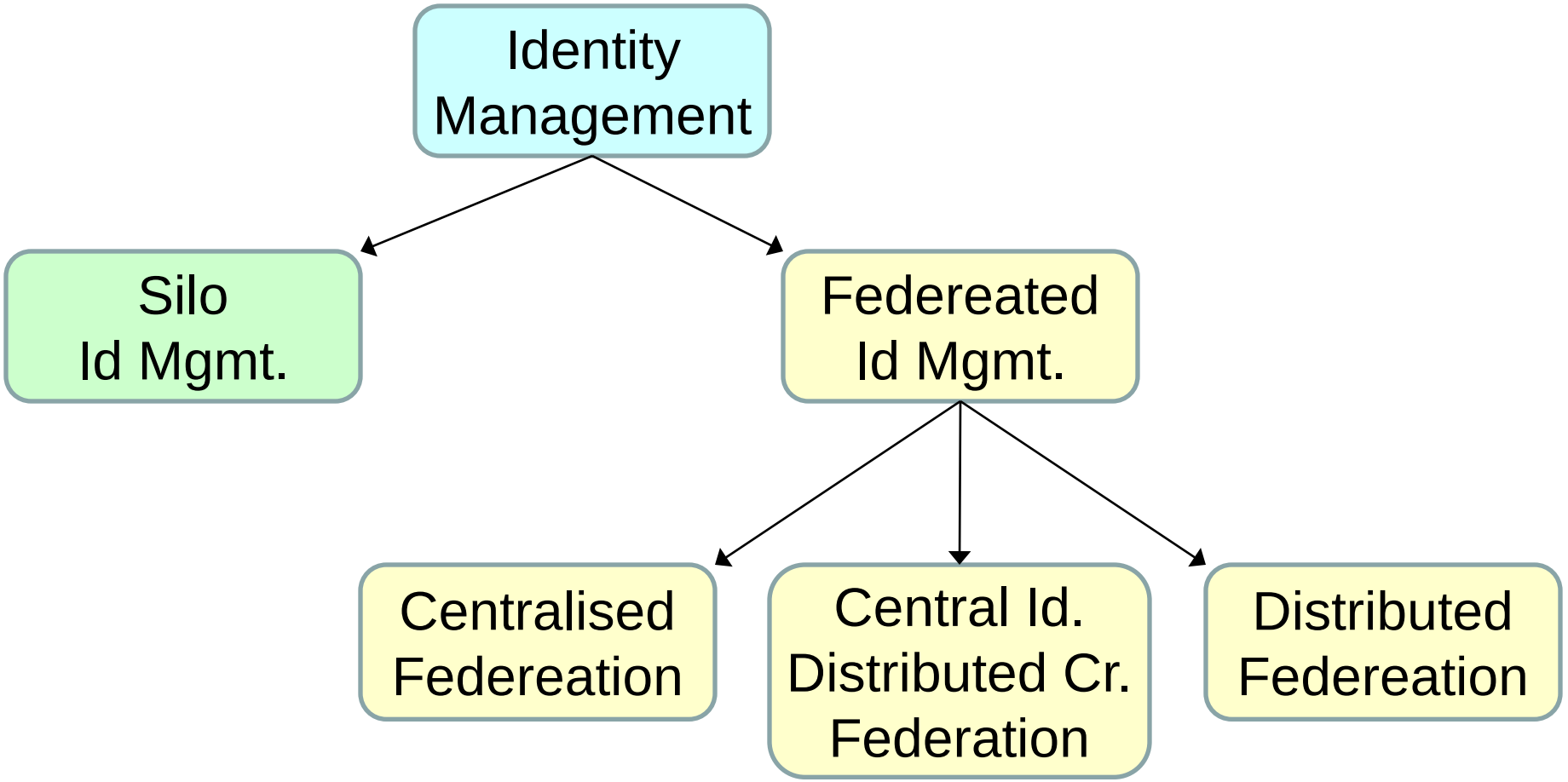
Identity Domains

- An identity domain has a name space of unique names
  - Same user has separate identities in different domains

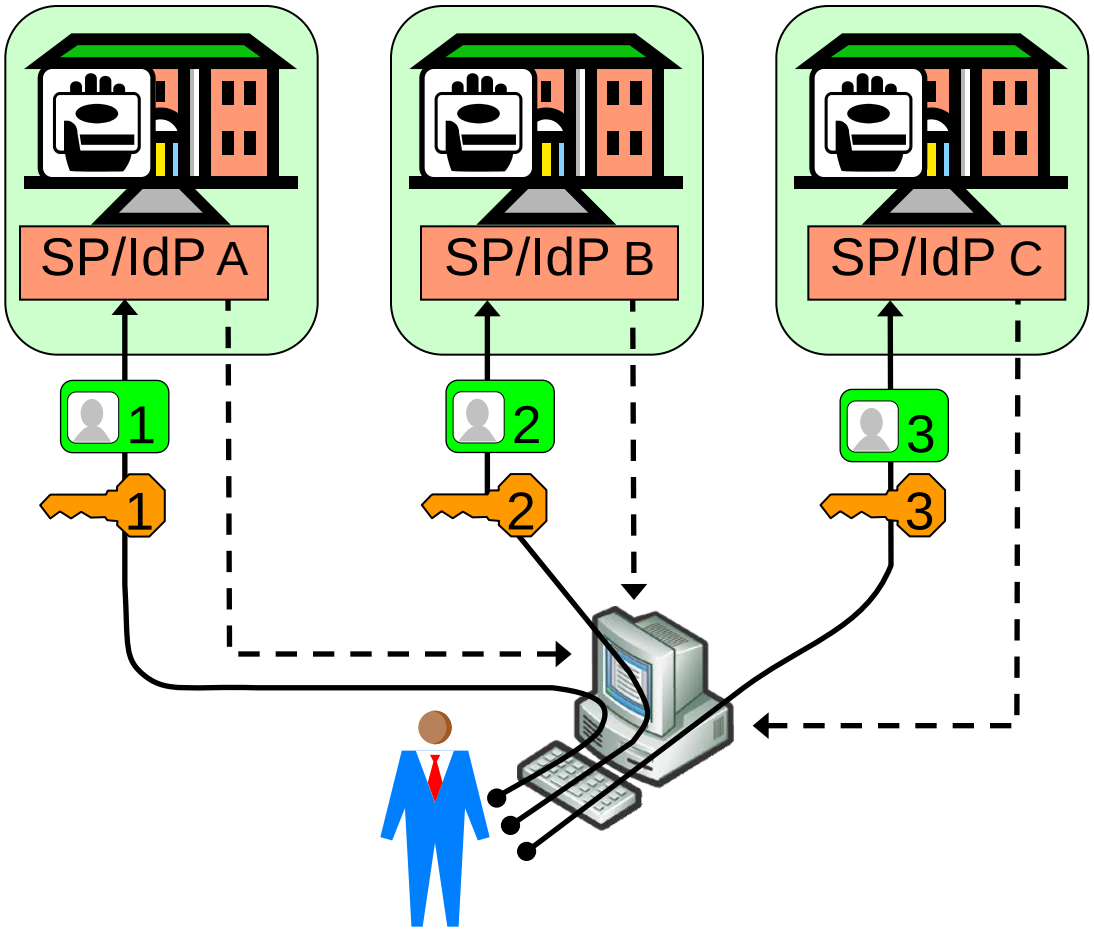


- Identity domain structure options:
  - Silo domain with single authority, e.g. User Ids in company network
  - Distributed hierarchic domain: e.g. DNS (Domain Name System)
- Federation of identity domains
  - Multiple domains have single identity for same user
  - Requires alignment of identity policy between domains

Taxonomy of Identity Management Architectures



Silo identity management model



Legend:

- SP
- IdP
- Identity domain
- User identifier for silo domain (X)
- Authentication token for silo domain (X)
- Service logon
- Service provision

Silo Id domains

- SP (Service Provider) = IdP (Identity Provider):
  SP controls 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, lost business

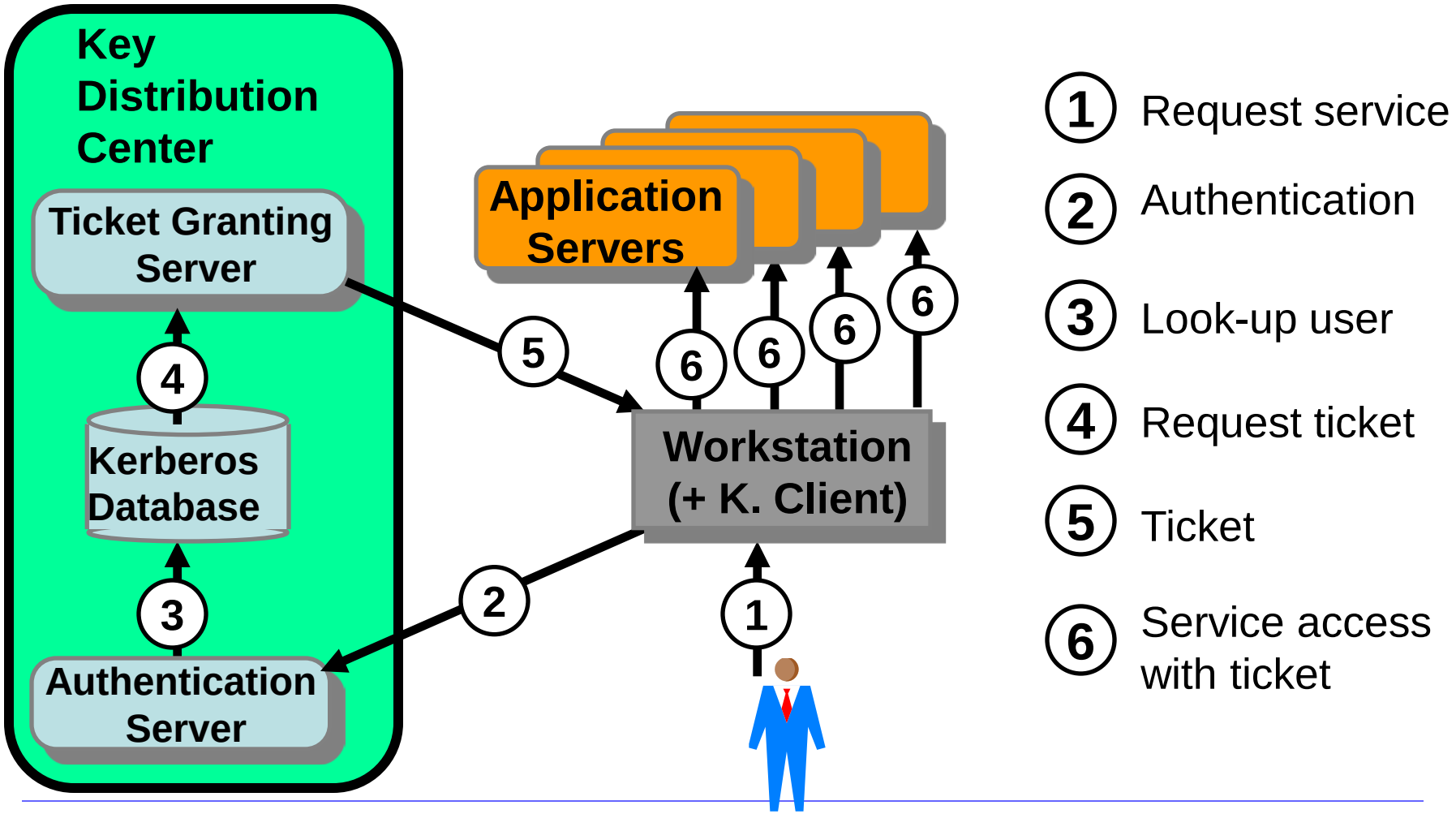
Single Id and SSO (Single Sign-On)

- Users don’t want more identifiers and credentials
- 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 to bundle and collect user information

Kerberos SSO

- Part of project Athena (MIT) in 1983.
- User must authenticate once at the beginning of a workstation session (login session).
- Server then authenticates Kerberos client on user's workstation instead of authenticating the user
  - So user does not need 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

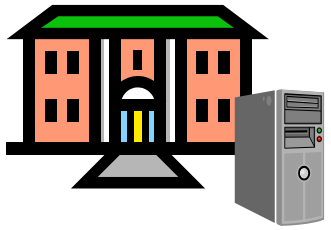


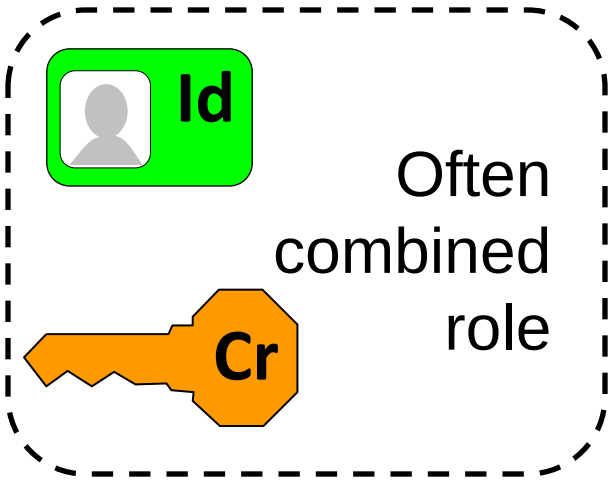
Kerberos – Advantages and limitations

- 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)

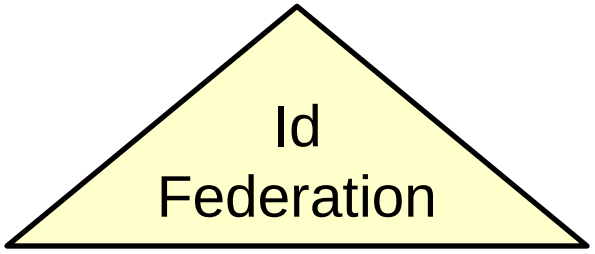
Identity Federation Roles

- User
  - Needs identities and credentials to access multiple SPs.
- Service Provider (SP)
  - Needs to know identity of users, and needs assurance of authenticity.
- Identity Provider (IdP)
  - Controls name space of identities. Issues/registers identities for users.
- Credentials Provider (CrP)
  - Issues/registers credentials for users. Performs authentication of users.





Identity Federation



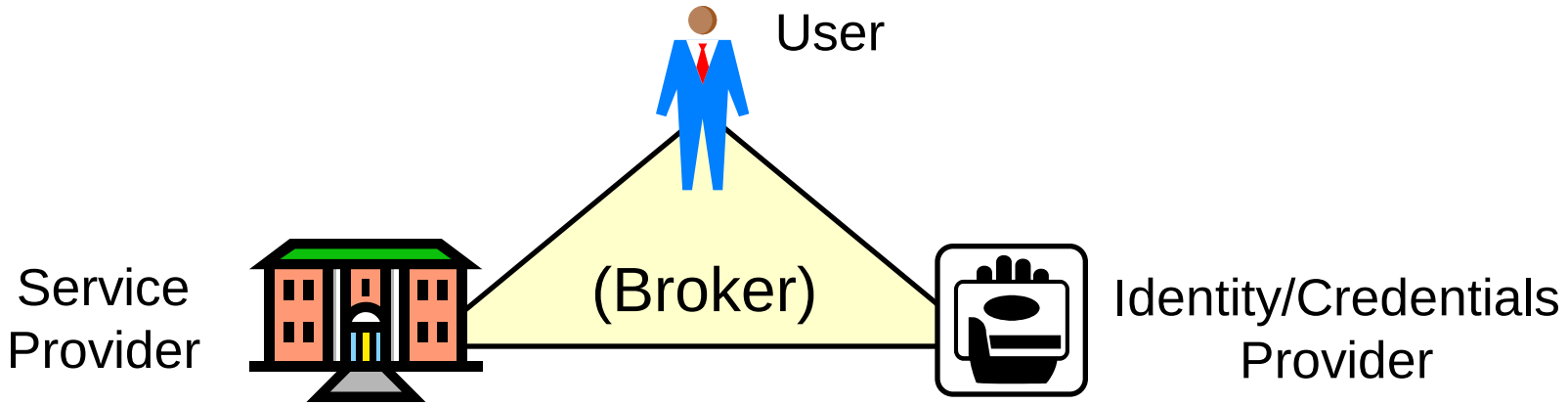
- Identity Federation
  - A set of agreements, standards and technologies that enable a group of SPs to recognise and trust user identities and credentials from different IdPs, CrPs and SPs.
- Three main architectures:
  1. **Centralized Federation:** Centralised mgmt of name space and credentials by single IdP/CrP.
  2. **Distributed Federation:** Distributed mgmt of name space and credentials by multiple IdPs and CrPs. Normally combined IdP/CrP.
  3. **Centralised Identity with Distributed Authentication:** Centralised mgmt of name space by single IdP. Distributed mgmt. of credentials and authentication by multiple CrPs.

Examples of Identity Federations

Federation types	Centralised Identity	Distributed Identity
Centralised Authentication	Centralised 	Distributed Id Central Cr 
Distributed Authentication	Central Id Distributed Cr 	Distributed 

Federation protocols

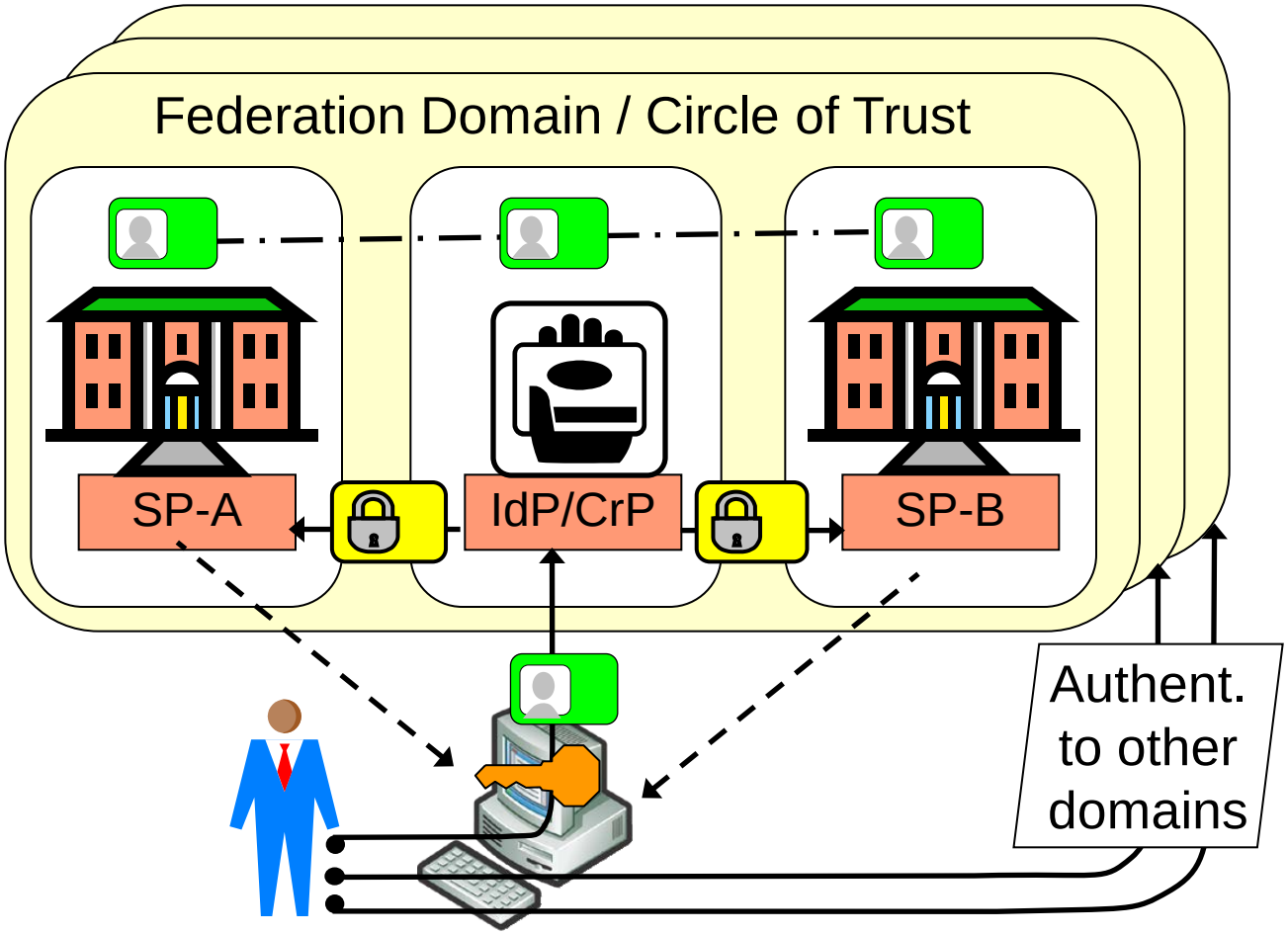
- Authentication by one IdP/CrP/SP is communicated as a security assertions (cryptographic token) to other SPs that trust and accept the assertion of authenticity.
- Usually based on SAML protocol
  - Security Assertions Markup Language
- Involves multiple entities
  - User, IdP, CrP, SP, and sometimes broker entity



Advantage/Disadvantage of Federation

- Advantages
  - Improved usability
  - Allows SPs to bundle services and collect user info
  - Strengthen privacy through pseudonym identities
- Disadvantages
  - High technical and legal complexity
  - High trust requirements
    - E.g. IdP is technically able to access SP on user's behalf
  - Privacy issues,
    - IdP collects info about user habits wrt. which SPs are used
  - Limited scalability,
    - Limited by political and economical constraints
    - An Identity federation becomes a new form of silo

Centralised Federation

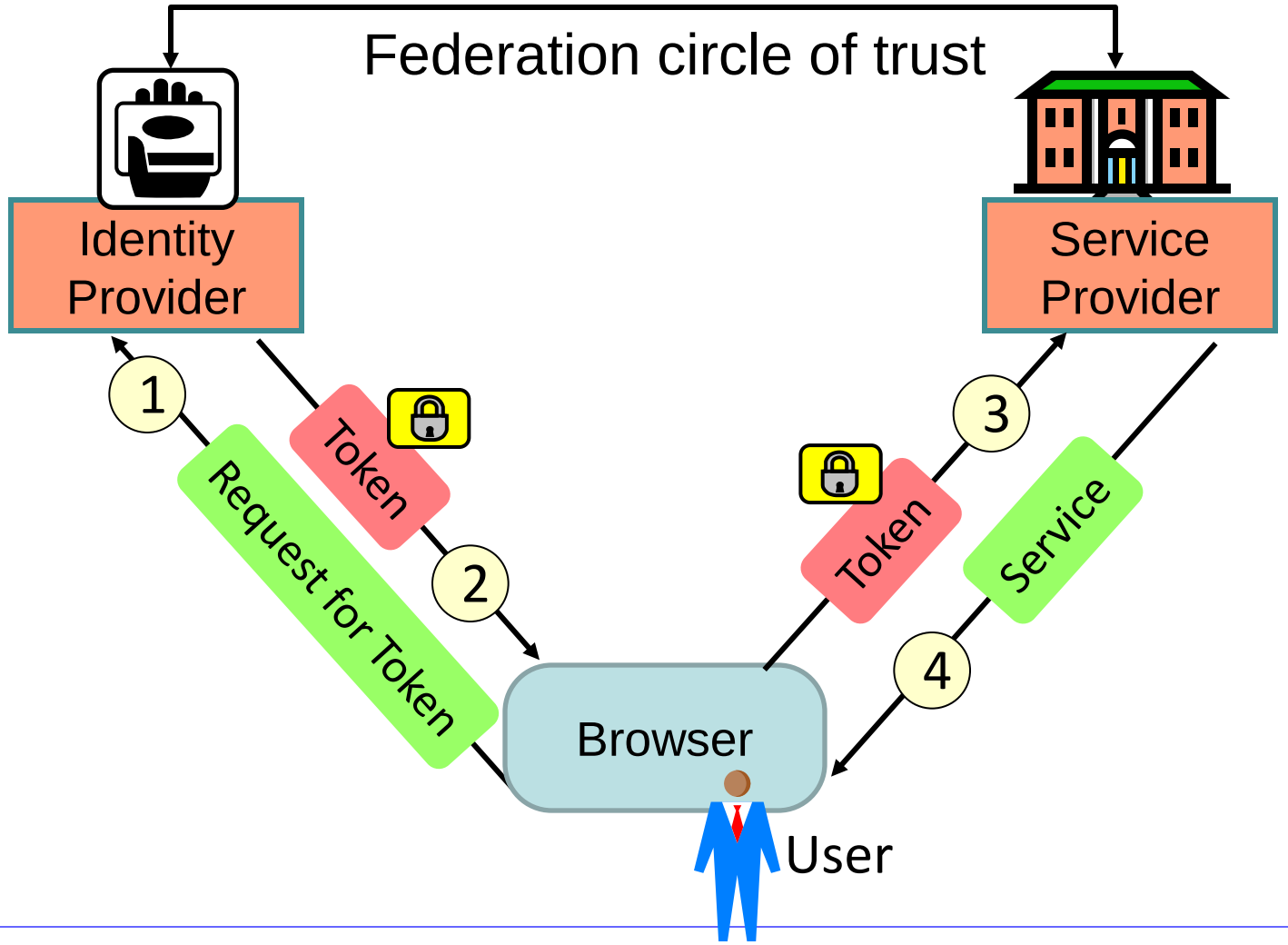


Legend :

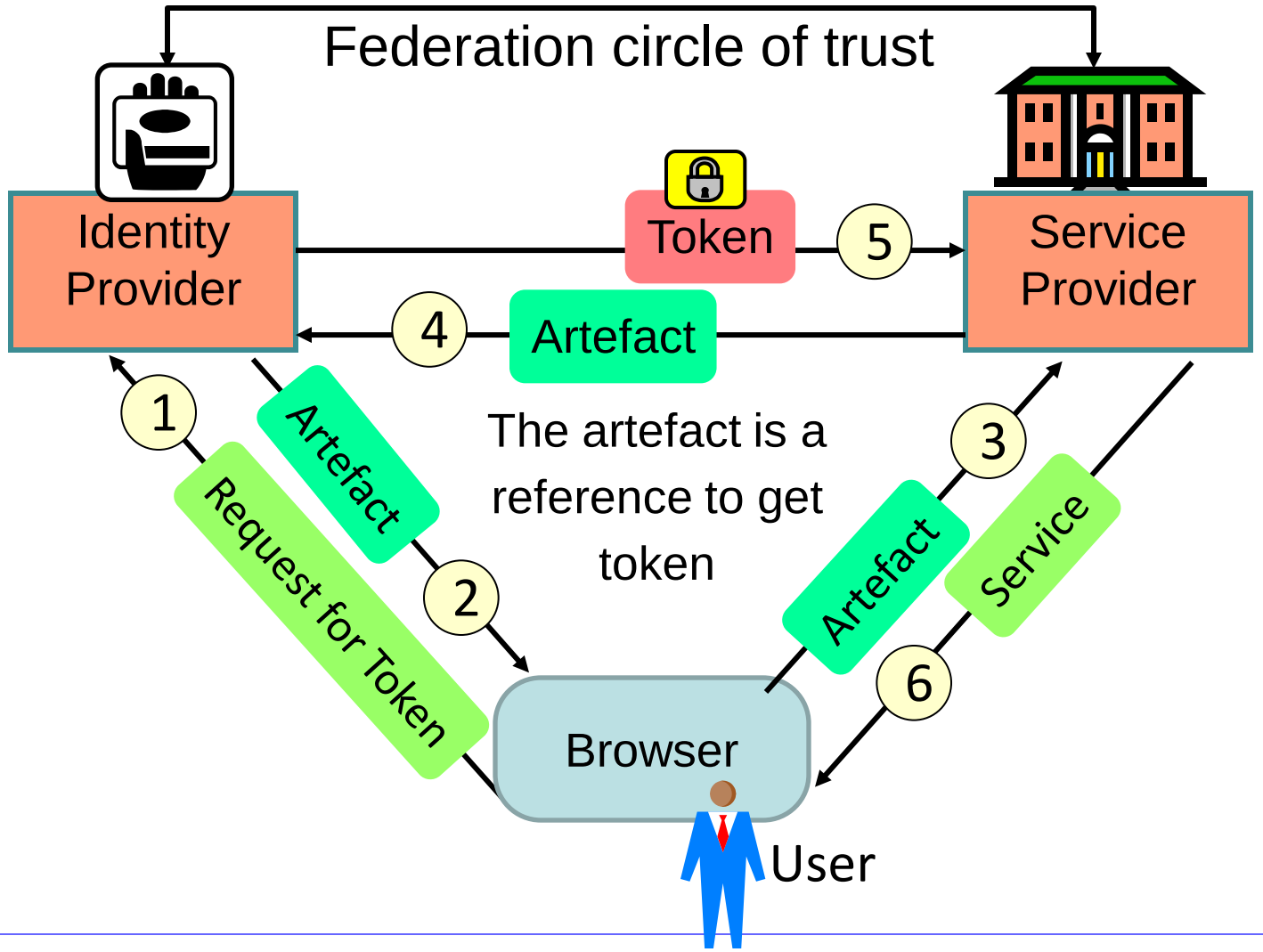
- SP
- IdP/CrP
- Identity domain
- User identifier issued by IdP
- Authentication cred. managed by IdP
- Security assertion issued by IdP
- Service logon
- Service provision
- Identifier mapping

Examples: Facebook connect

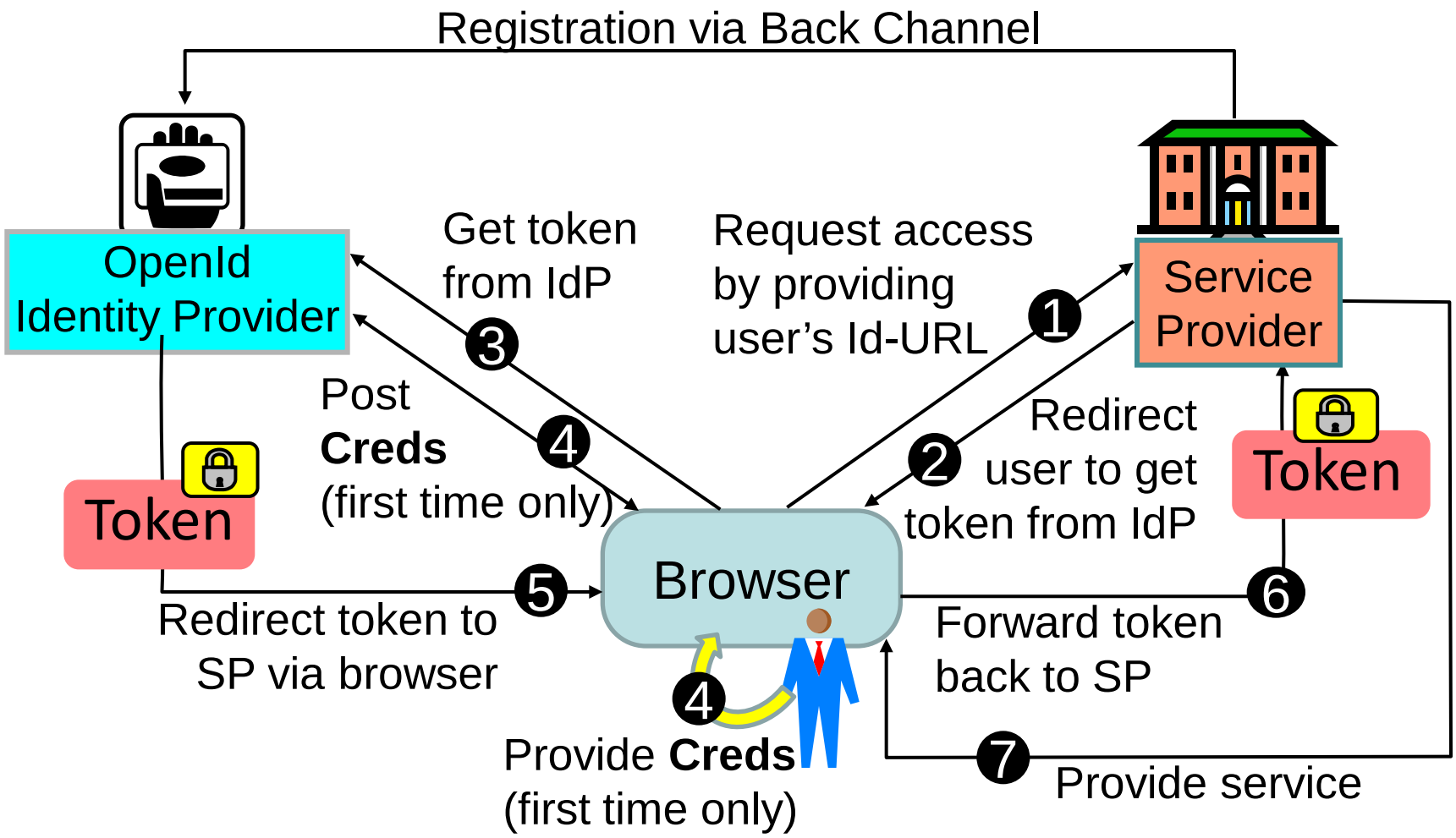
SAML protocol profile: Browser Post Security token via front-channel



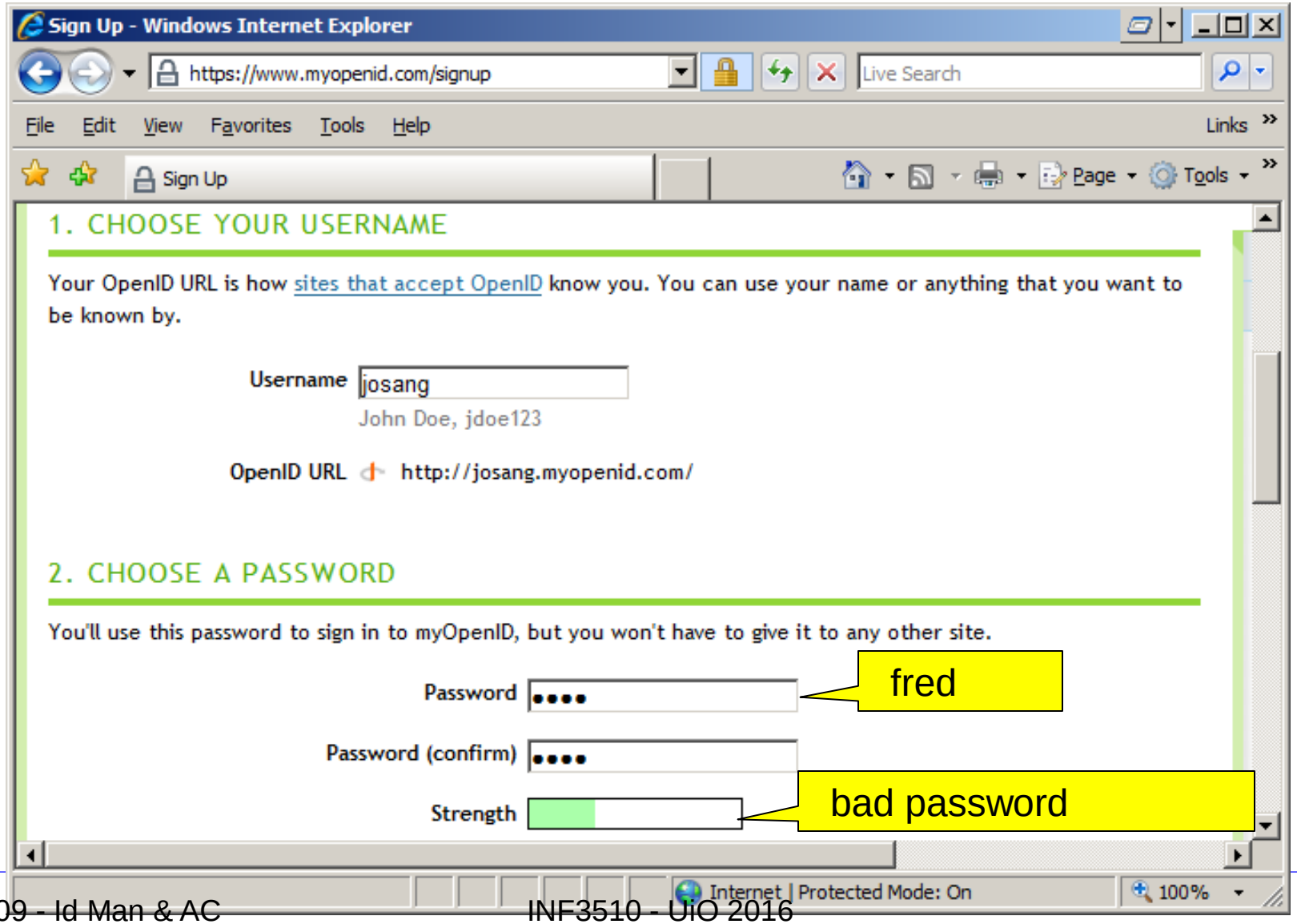
SAML protocol profile: Browser Artefact Security token via back-channel



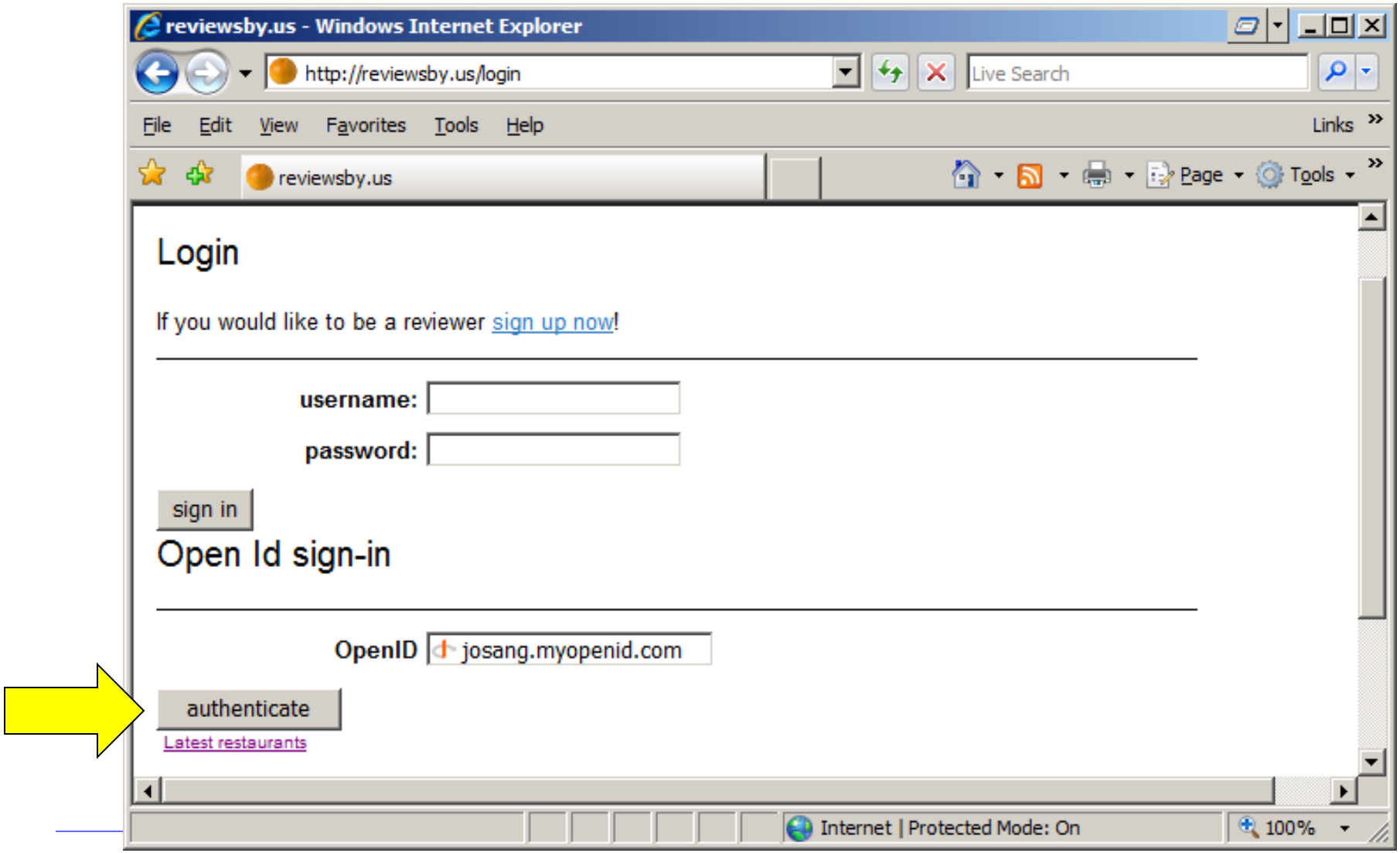
OpenID Distributed Federation



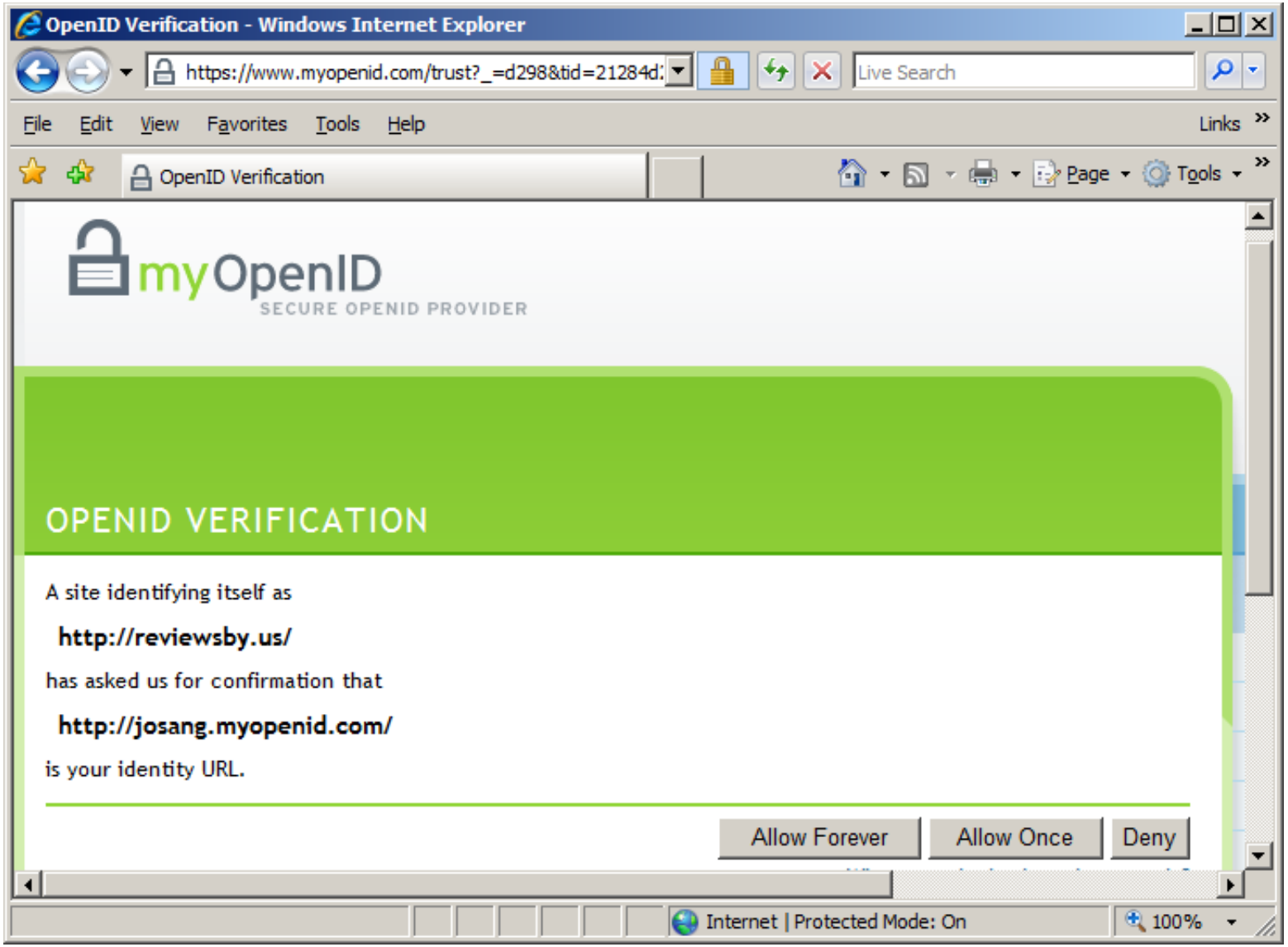
OpenID self registration



Service Access Without Password



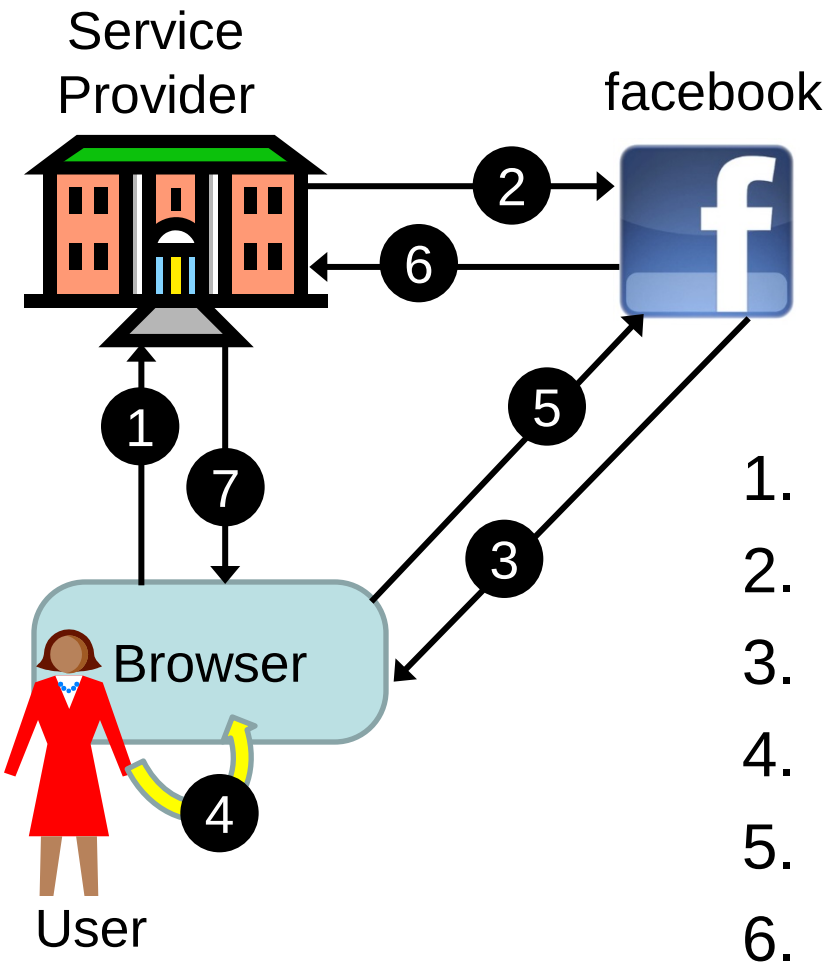
First Time Service Access



OpenID Characteristics

- Self registration
- Anybody can be IdProvider and Server, also you
- Not all IdProviders are recognised as ”authorities”
- A SP can specify which IdPs it accepts
- Not suitable for sensitive services
- Typically for services that only require low authentication assurance
- Vulnerable to multiple forms of abuse

Facebook Centralised Federation



Authentication with Facebook Connect

1. User requests service
2. Redirect to facebook authentication
3. Present facebook login form
4. User provides Id + credential
5. Credentials forwarded to facebook
6. Confirm authenticated user
7. Provide service

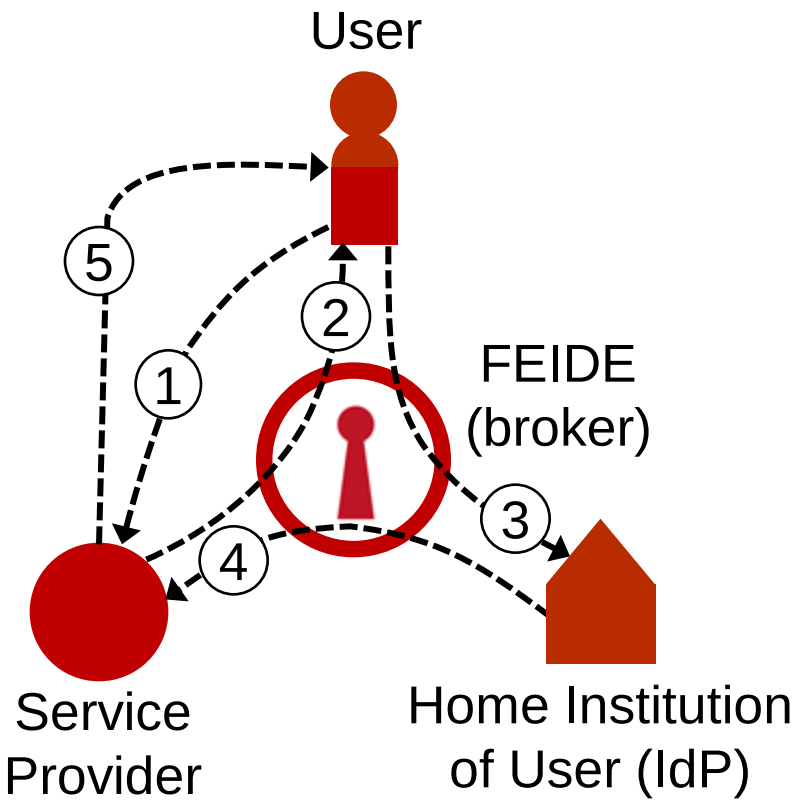
(Felles Elektronisk Identitet)

- FEIDE is a distributed federation with centralised broker for the Norwegian national education sector.
- Users register username and password with own home organisation
- Users authenticate to web-services via FEIDE's centralized login service
- The Service Provider receives user attributes from the user's Home Institution
- The Service Providers never sees the user's password/credential, it only receives user attributes that it need to know in order to provide the service.

FEIDE (continued)

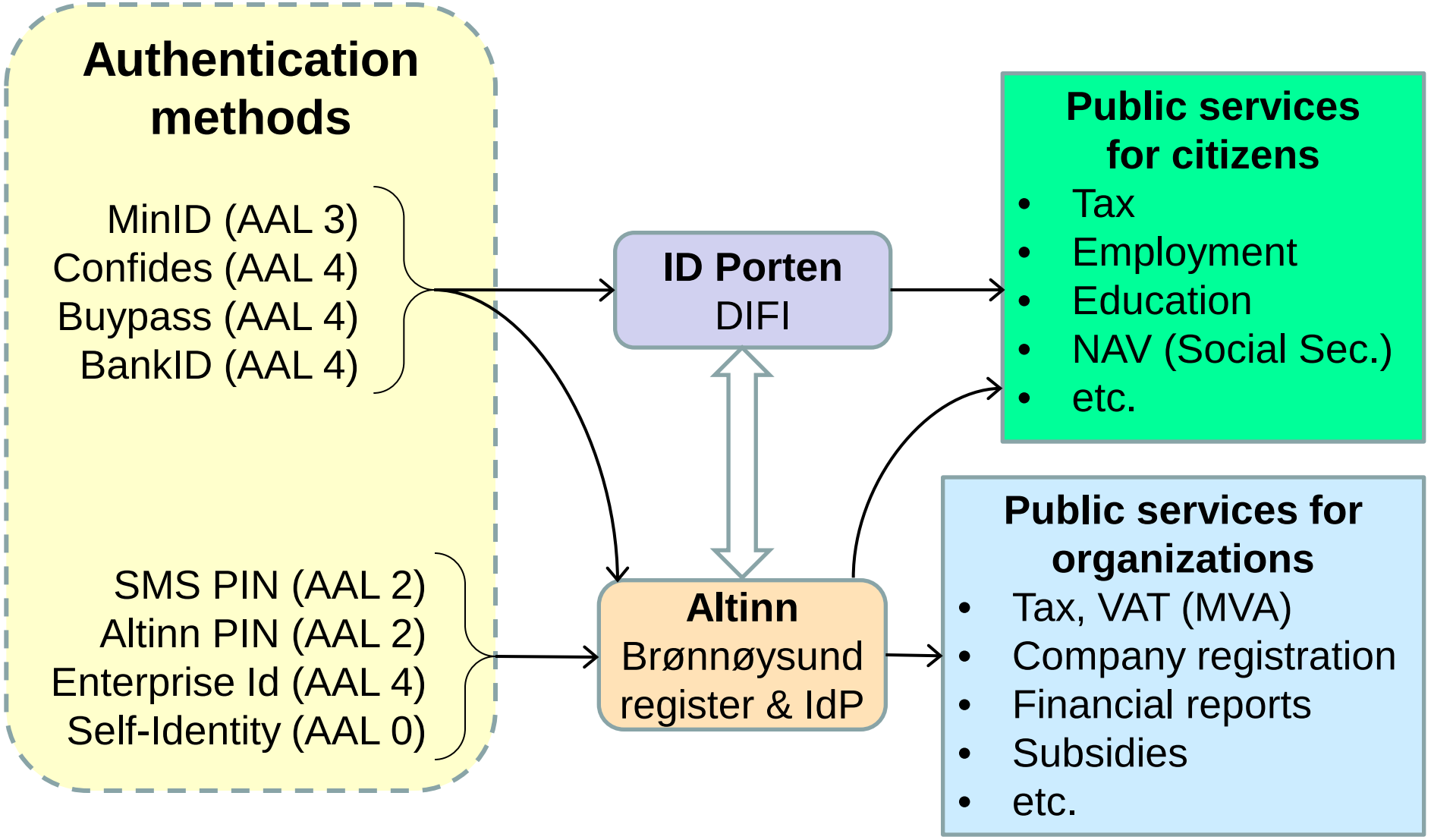
- FEIDE has formal agreements with the universities and schools before they are connected
- Home Institutions (universities and schools) are responsible for keeping user data correct and up-to-date
- Service Providers decide themselves what services their own users and other users should be able to access via FEIDE's central log-in service.

Scenario



1. User requests access to service
2. Service Provider sends authentication request to FEIDE, and displays FEIDE login form to user.
3. User enters name and password in FEIDE login form, which are sent for validation to Home Institution of user.
4. Home Institution confirms authentic user and provides user attributes to FEIDE which forwards these to SP
5. Service Provider analyses user attributes and provides service according to policy

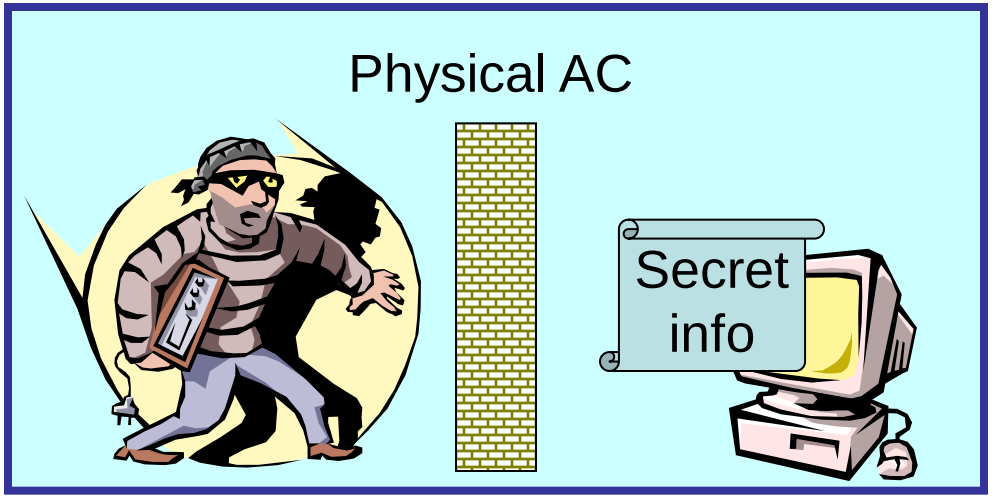
Norw. e-Gov. Distributed Fed. with Broker



Introduction to Logical Access Control

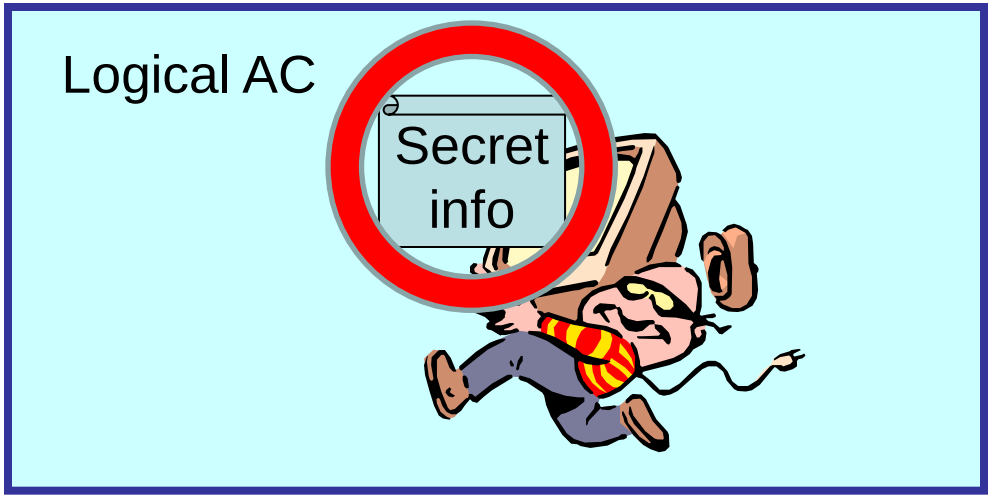
Physical Access Control:

(not the theme today)



Logical Access Control:

(this lecture)



Basic concepts

- Access control security models:
  - *How to define which subjects can access which objects with which access modes?*
- Three classical approaches
  - Discretionary Access Control (DAC)
  - Mandatory access control (MAC)
  - Role-Based Access Control (RBAC)
- Advanced approach for distributed environments:
  - Attribute-Based Access Control (ABAC)
    - Generalisation of DAC, MAC and RBAC

Access modes

- Modes of access:
  - ***Authorizations specify the access permissions of subjects (users) when accessing objects (resources)***
- 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

DAC / MAC
According to the Orange Book (TCSEC)

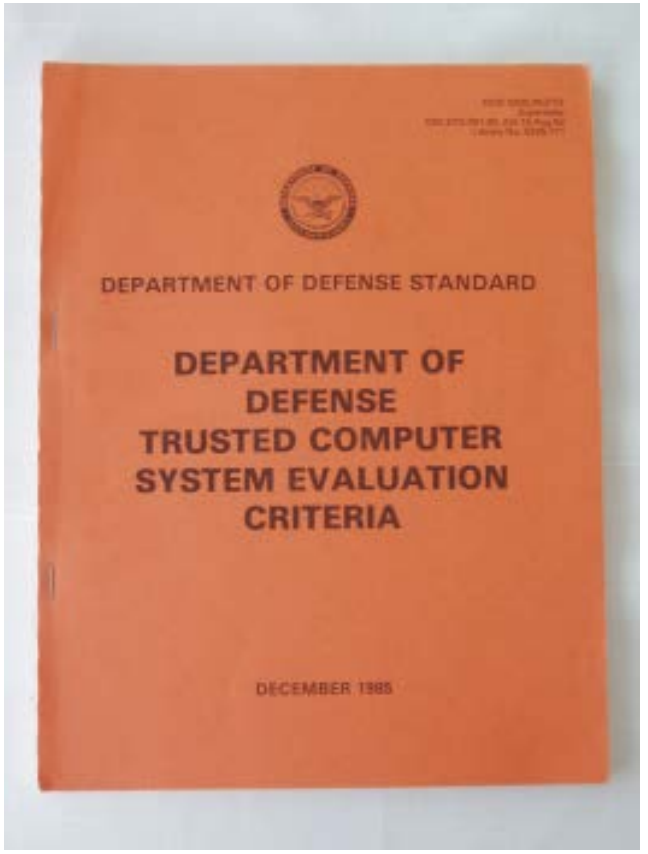
TCSEC (1985) specifies two AC security models

- Discretionary AC (DAC)
  - AC policy based on user identities
  - e.g. *John has* **(r,w)** *- access to HR-files*

	HR	Sales
John	r,w	
Mary		r,w

- Mandatory AC (MAC)
  - AC policy based on security labels
  - e.g. *secret clearance needed for access*

Secret



Orange Book, 1985

DAC – Discretionary Access Control

- Access authorization is specified and enforced based on the identity of the user.
- DAC is typically implemented with ACL (Access Control Lists)
- DAC is discretionary in the sense that the owner of the resource can decide at his/her discretion who is authorized
- Operating systems using DAC:
  - Windows and Linux

DAC principles

- AC Matrix
  - General list of authorizations
  - Impractical, too many empty cells
- Access Control Lists (ACL)
  - Associated with an object
  - Represent columns from AC Matrix
  - Tells who can access the object

Columns→ ↓Rows		Objects			
		O1	O2	O3	O4
Subject names	S1	r,w	-	x	r
	S2	r	-	r	r,w
	S3	-	x	-	-
	S4	r,w	x	x	x

AC Matrix

- AC lists →

	O1
S1	r,w
S2	r
S3	-
S4	r,w

	O2
S1	-
S2	-
S3	x
S4	x

	O3
S1	x
S2	r
S3	-
S4	x

	O4
S1	r
S2	r,w
S3	-
S4	x

ACL in Unix

Each file and directory has an associated ACL

- Three access operations:
  - **read:** from a file
  - **write:** to a file
  - **execute:** a file
- Access applied to a directory:
  - **read:** list contents of dir
  - **write:** create or rename files in dir
  - **execute:** search directory

- Permission bits are grouped in three triples that define read, write, and execute access for owner, group, and others.
- A '-' indicates that the specific access right is not granted.
- **rw-r--r--** means: read and write access for the owner, read access for group, and for others (world).
- **rwx------** means: read, write, and execute access for the owner, no rights for group and no rights for others

Capabilities

- Focus on the subjects:
  - access rights stored with subjects
  - Represents rows of AC Matrix
- Must be impossible for users to create fake capabilities
- Subjects may grant own capabilities to other subjects. Subjects may grant the right to grant rights.
- Challenges:
  - How to check who may access a specific object?
  - How to revoke a capability?
- Similar to SAML security token

AC Capabilities ↓

	O1	O2	O3	O4
S1	r,w	-	x	r

	O1	O2	O3	O4
S2	r	-	r	r,w

	O1	O2	O3	O4
S3	-	x	-	-

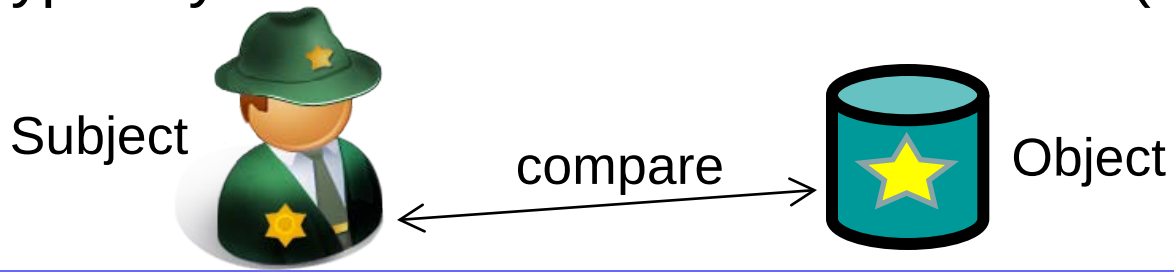
	O1	O2	O3	O4
S4	r,w	x	x	x

MAC – Mandatory Access Control

- Access authorization is specified and enforced with security labels
  - Security clearance for subjects
  - Classification levels for objects
- MAC compares subject and object labels
- MAC is mandatory in the sense that users do not control access to the resources they create.
- A system-wide set of **AC policy rules** for subjects and objects determine modes of access
- OS with MAC:
  - SE Linux supports MAC

MAC principles: Labels

- Security Labels can be assigned to subjects and objects
  - Can be strictly ordered security levels, e.g. “Confidential” or “Secret”
  - Can also be partially ordered categories, e.g. {Sales-dep, HR-dep}
- Dominance relationship between labels
  - ($L_A \geq L_B$) means that label L_A dominates label L_B
- Object labels are assigned according to sensitivity
- Subject labels are determined by security clearance
- Access control decisions are made by comparing the subject label with the object label according to specific model
- MAC is typically based on Bell-LaPadula model (see later)



Bell-LaPadula: The classical MAC model

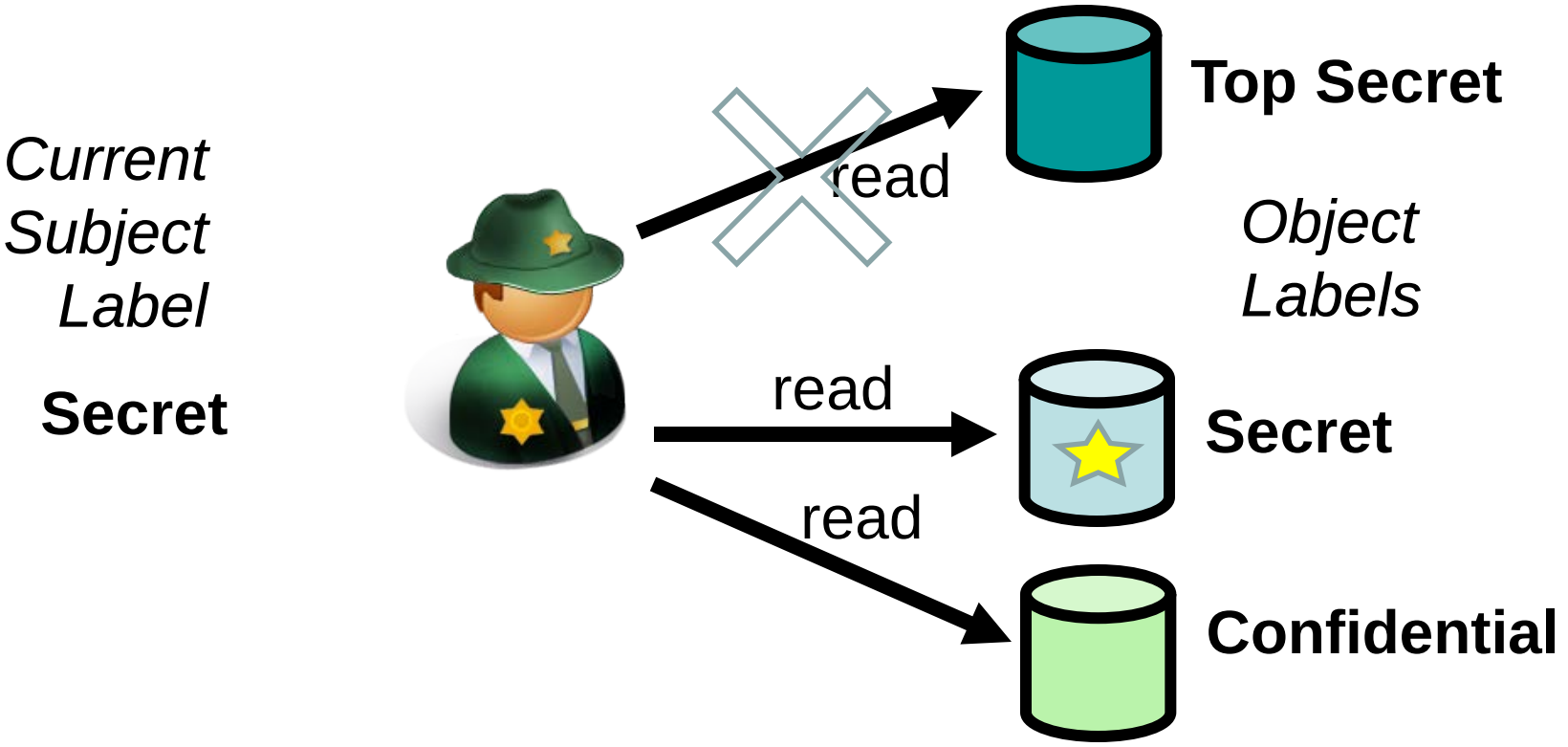
SS-property (Simple Security): No Read Up

- A subject should not be able to read files with a higher label than its own label, because otherwise it could cause unauthorized disclosure of sensitive information.
- So you should only be able to read documents with an equal or lower label as your security clearance level.

***-Property (Star Property): No Write Down**

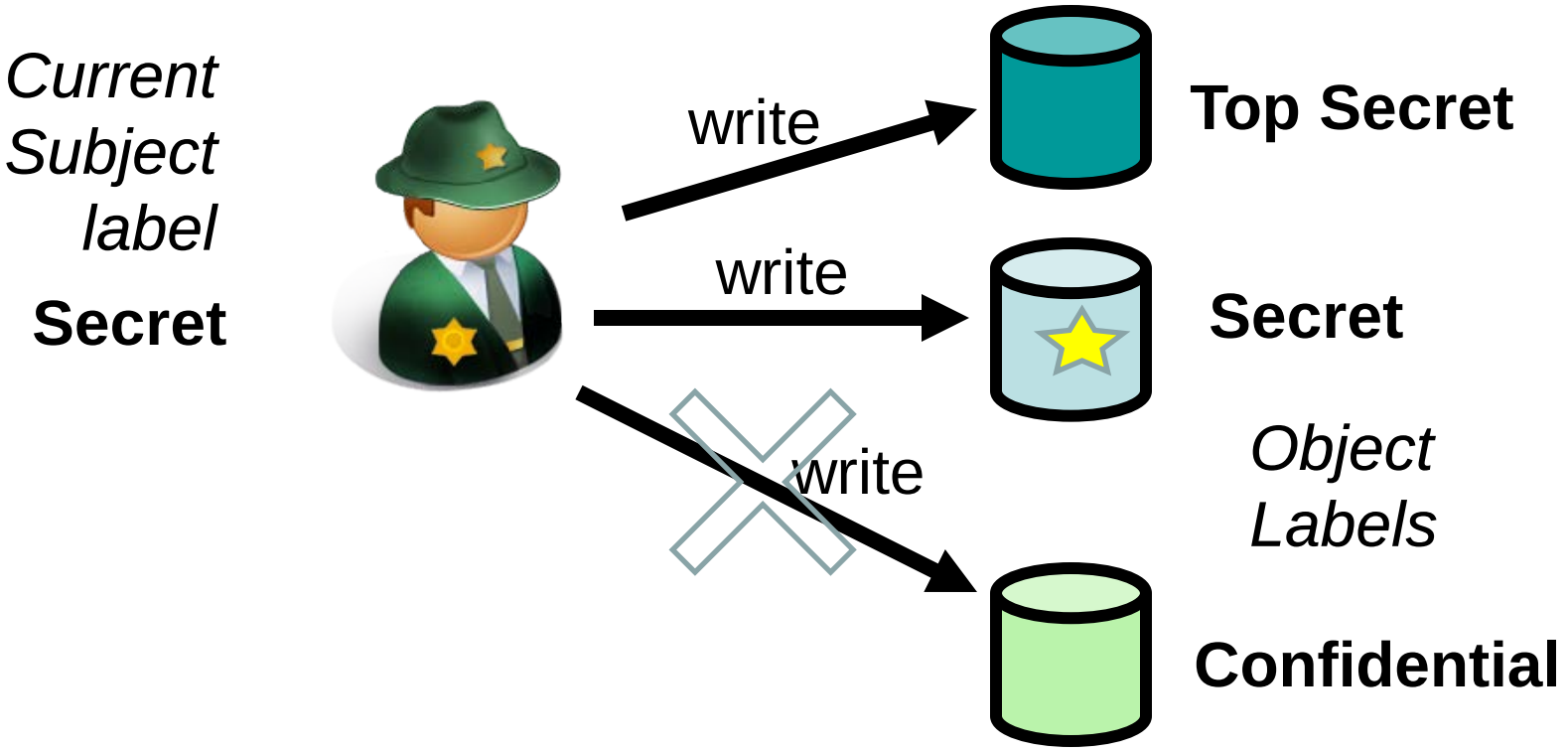
- Subjects working on information/tasks at a given level should not be allowed to write to a lower level, because otherwise it could create unauthorized information flow.
- So you should only be able write to files with an equal or higher label as your security clearance level.

Bell-LaPadula (MAC model)
SS-Property: No Read Up



Diagram

Bell-LaPadula (MAC model)
*-Property: No Write Down

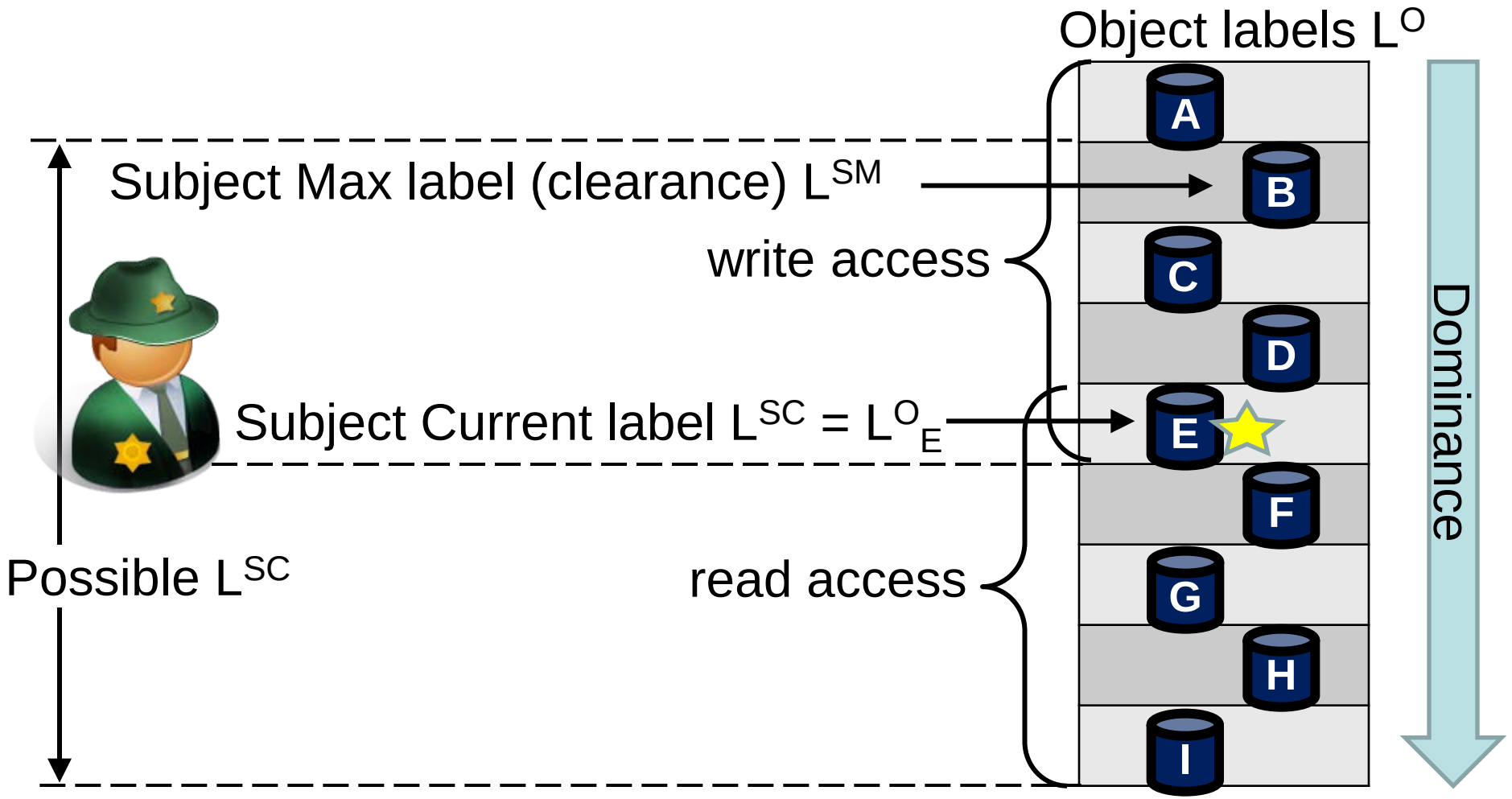


Labels in Bell La Padula

- Users have a clearance level L^{SM} (Subject Max level)
- Users log on with a current clearance level L^{SC} (Subject Current level) where $L^{SC} \leq L^{SM}$
- Objects have a sensitivity level L^{O} (Object)

- SS-property allows read access when $L^{SC} \geq L^{O}$
- *-property allows write access when $L^{SC} \leq L^{O}$

Bell-LaPadula label relationships



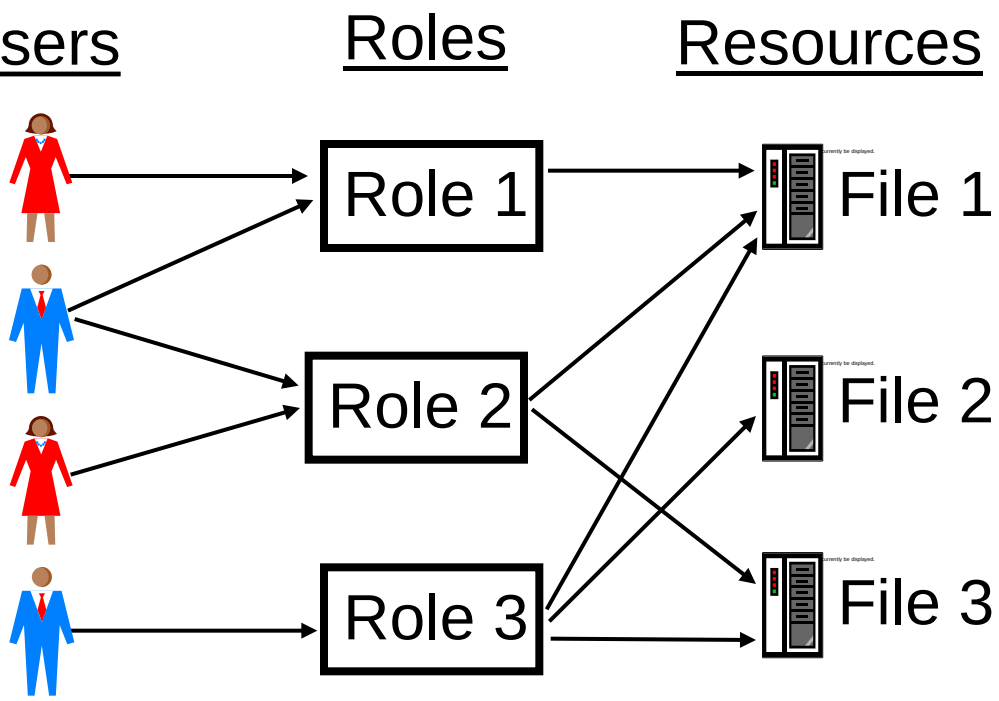
Combined MAC & DAC

- Combining access control approaches:
  - A combination of mandatory and discretionary access control approaches is often used
    - MAC is applied first,
    - DAC applied second after positive MAC
    - Access granted only if both MAC and DAC positive
  - Combined MAC/DAC 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

RBAC: Role Based Access Control

- 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 and/or 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 her to perform only what's required for that role.
- All users with the same role have the same permissions.

ABAC and XACML

ABAC = Attribute Based Access Control

- ABAC specifies access authorizations and approves access through policies combined with attributes. The policy rules can apply to any type of attributes (user attributes, resource attribute, context attributed etc.).
- XACML used to express ABAC attributes and policies.

XACML = eXtensible Access Control Markup Language

- The XACML standard defines a language for expressing access control attributes and policies implemented in XML, and a processing model describing how to evaluate access requests according to the rules defined in policies.
- XACML attributes are typically structured in ontologies

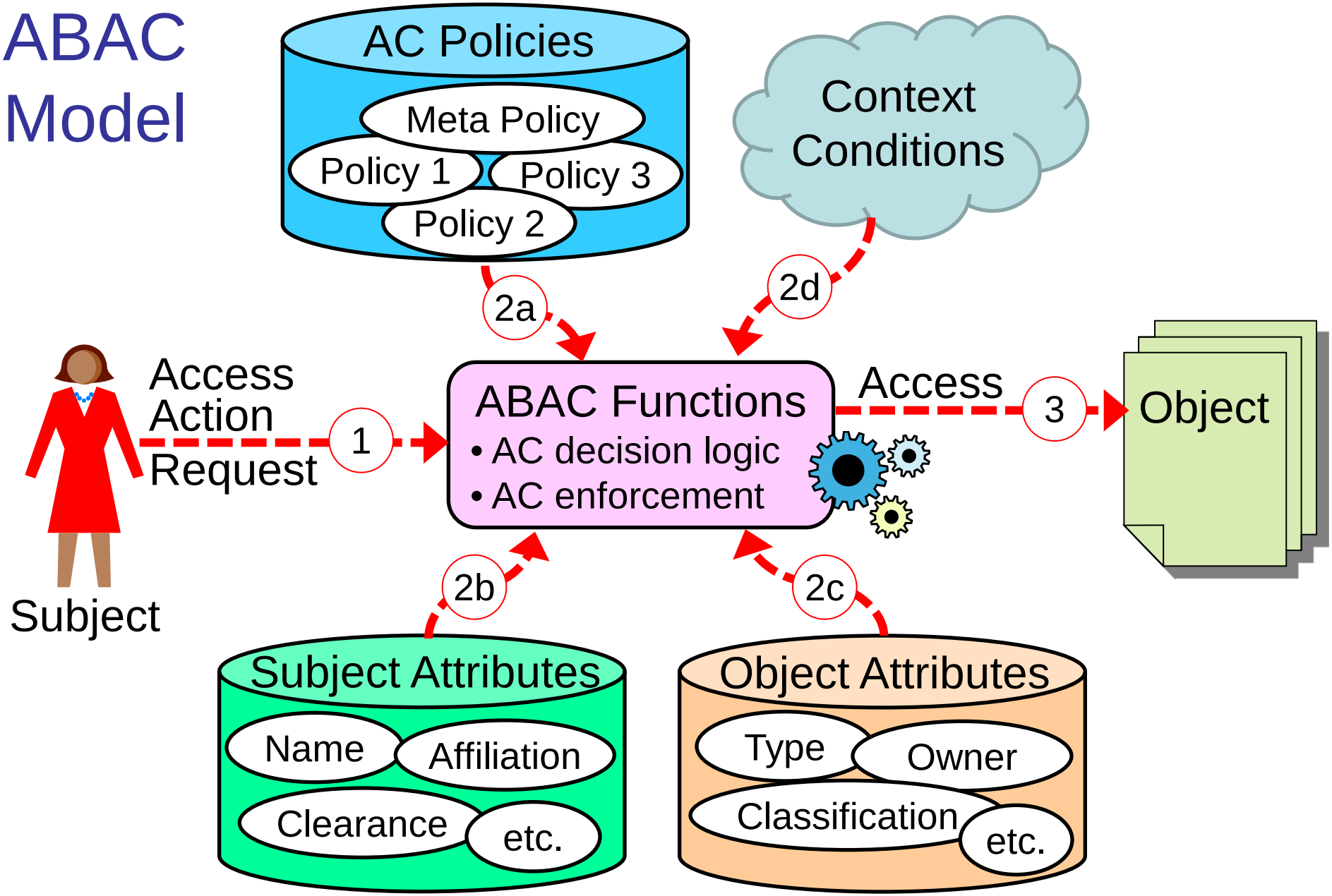
Attribute Based Access Control

- ABAC makes AC decisions based on Boolean conditions on attribute values.
- **Subject**, **Object**, **Context**, and **Action** consist of attributes
  - Subject attributes could be: Name, Sex, DOB, Role, etc.
  - Each attributes has a value, e.g.:
  - (Name (subject) = Alice), (Sex(subject) = F), (Role(subject) = HR-staff), (AccessType(action) = {read, write}),

    (Owner(object) = HR), (Type(object) = salary)
- The AC logic analyses all (attribute = value) tuples that are required by the relevant policy.
  - E.g. permit if:

    [Role(subject) = HR-staff) and (AccessType(action) = read) and (Owner(object) = HR)] and (Time(query) = office-hours)]

ABAC Model


AC Policies
Meta Policy
Policy 1
Policy 3
Policy 2
Context Conditions
2a
2d
Access Action Request
1
ABAC Functions
• AC decision logic
• AC enforcement
Access
3
Object
Subject
2b
2c
Subject Attributes
Name
Affiliation
Clearance
etc.
Object Attributes
Type
Owner
Classification
etc.

Global Consistence

- ABAC systems require an XML terminology to express all possible attributes and their values,
- Must be consistent across the entire domain,
  - e.g. the attribute Role and all its possible values, e.g. (Role(subject) = HR-staff), must be known and interpreted by all systems in the AC security domain.
- Requires standardization:
  - e.g. for access to medical journals, medical terms must be interpreted in a consistent way by all systems
  - current international work on XML of medical terms
- Consistent interpretation of attributes and values is a major challenge for implementing ABAC.

ABAC: + and –

On the positive side:

- ABAC is much more flexible than DAC, MAC or RBAC
  - DAC, MAC and RBAC can be implemented with ABAC
- Can use any type of access policies combined with an unlimited number of attributes
- Suitable for access control in distributed environments
  - e.g. national e-health networks

On the negative side:

- Requires defining business concepts in terms of XML and ontologies which is much more complex than what is required in traditional DAC, MAC or RBAC systems.
- Political alignment and legal agreements required for ABAC in distributed environments

Meta-policies i.c.o. inconsistent policies

- Sub-domain authorities defined their own policies
- Potential for conflicting policies
  - E.g. two policies dictate different access decisions
- Meta-policy rules needed in case the ABAC logic detects policy rules that lead to opposite decisions
- Meta-policy takes priority over all other policies, e.g.
  - Meta-Policy Deny Overrides: If one policy denies access, but another policy approves access, then access is denied.
    This is a conservative meta-policy.
  - Meta-Policy Approve Overrides: If one policy denies access, but another policy approves access, then access is approved.
  - This is a lenient meta-policy.

End of lecture