IN2120 Information Security
Spring 2019

Lecture 10

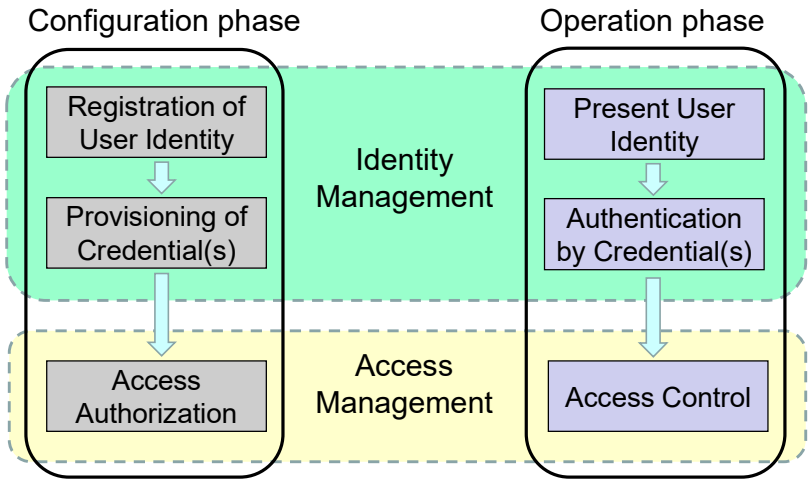
Identity and Access Management

Audun Jøsang
University of Oslo

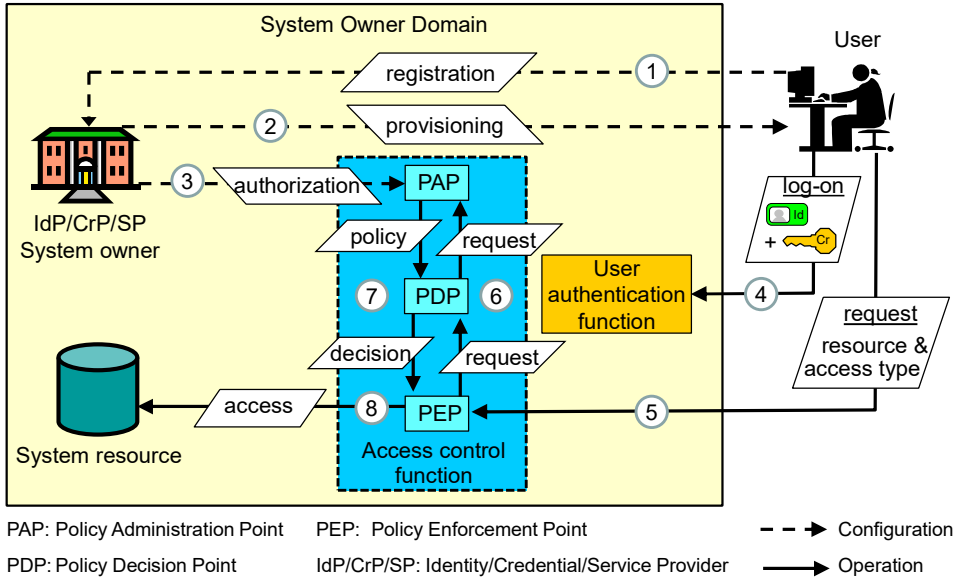
Outline

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

IAM
Identity and Access Management



Identity and Access Management Scenario

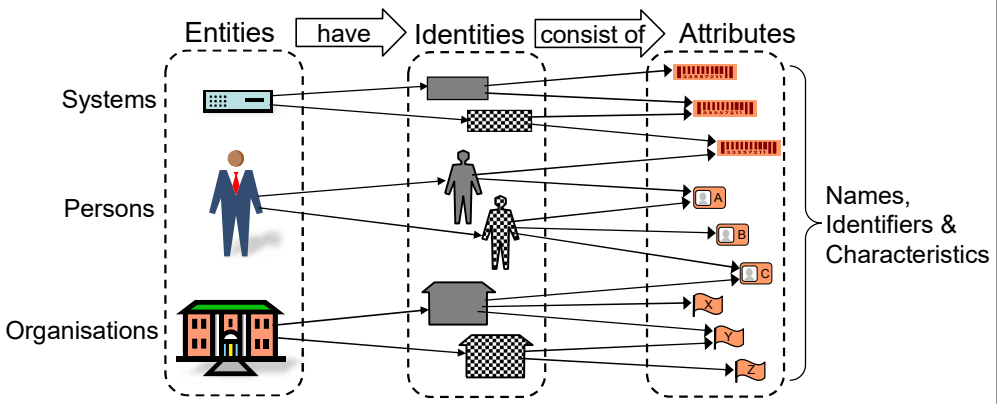


Definition of IAM

- *Identity and access management (IAM) is the security discipline that enables the right individuals to access the right resources at the right times for the right reasons.*
- *IAM addresses the mission-critical need to ensure appropriate access to resources across increasingly heterogeneous technology environments, and to meet increasingly rigorous compliance requirements.*

Gartner, IT Glossary
http://blogs.gartner.com/it-glossary/identity-and-access-management-iam/

The concept of identity



Concepts related to identity

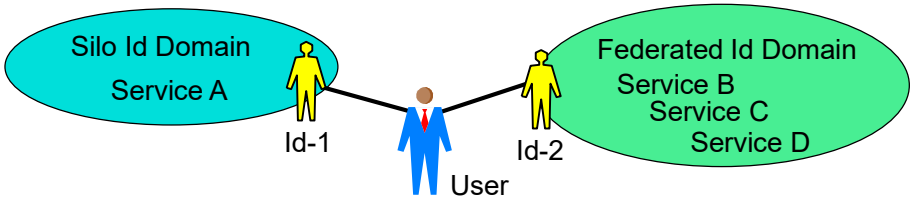
- Entity
  - A person, organisation, agent, system, session, process, etc.
- Identity
  - A set of names / attributes of entity in a specific domain
  - An entity may have identities in multiple domains
  - An entity may have multiple identities in one domain
- Digital identity
  - Digital representation 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 defined by authority, interpretation by humans and/or by 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"
  - because the identity first must be created/registered
- 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

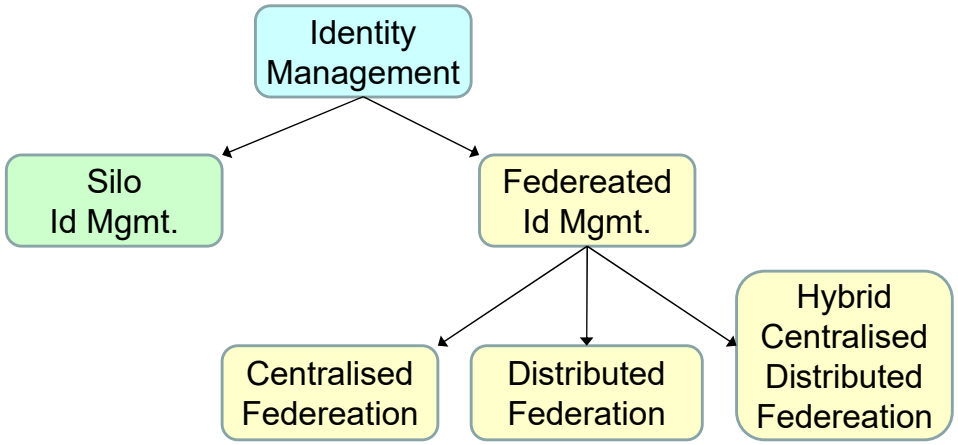
Identity Domains

- An identity domain has a name-space of unique names
  - The same user can have separate identities in different domains

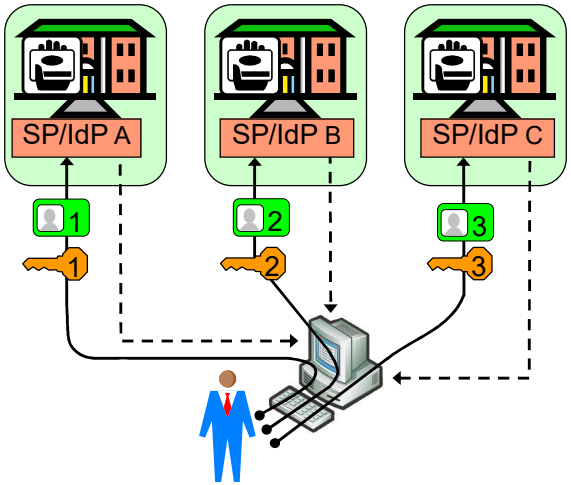


- Identity domain structures:
  - Silo domain with single authority, e.g. User Ids in company network
  - Distributed hierarchic domain: e.g. DNS (Domain Name System)
- Federated identity domains
  - Identity domain can be used by many different Service Providers
  - Requires alignment of identity management between SPs

Taxonomy of Identity Management Architectures



Silo identity management model



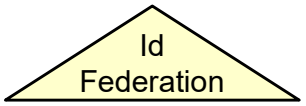
Legend:

- SP
- IdP/CrP
- Identity domain
- User identifier for silo domain
- Authentication token for silo domain
- 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 initial cost for SPs
  - Potentially good privacy
- Disadvantages
  - Identity overload for users, poor usability, no business integration
  - Low acceptance of new services with separate Id & credentials
  - Users must provide same information to different service providers
  - For service providers: Barrier to service bundling and data collection

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.

- Four main types:
  1. **Centralized Federation:** Centralised name space and management of credentials by single IdP/CrP.
  2. **Distributed Identity with Centralised Authentication:** Distributed name spaces managed by multiple IdPs. Centralised credentials authentication by single CrP.
  3. **Centralised Identity with Distributed Authentication:** Centralised name space managed by single IdP. Distributed mgmt. of credentials and authentication by multiple CrPs.
  4. **Distributed Federation:** Distributed name spaces and management of credentials by multiple IdPs and CrPs.

Identity Federation Types

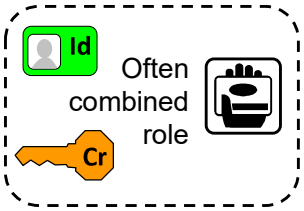
Federation types	Centralised Identity	Distributed Identity
Centralised Authentication	Centralised AADHAAR	Distributed Id Centralised Cr Google + facebook twitter
Distributed Authentication	Centralised Id Distributed Cr ID-porten altinn HelseID	Distributed FEIDE eduroam

Federation model types

- Aadhaar (India) and google+ are centralised because
  - they control and manage the domain's name space of identities,
  - they always verify the authentication credentials in their federations.
- Facebook and Twitter have distributed identities and centralised credentials because
  - they do not manage identities which are ordinary email addresses,
  - they always verify the authentication credentials in their federations.
- The ID-portal Norway has centralised Id and distributed authentication because
  - identities are national id-numbers, managed by the government
  - multiple private credentials providers verify credentials for authentication
- OpenID and eduroam are distributed because
  - multiple Id-providers control and manage name spaces for identities
  - the same Id-providers also verify the credentials for authentication

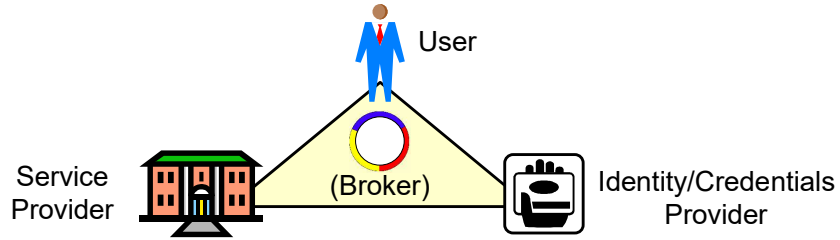
Identity Federation Players

- User
  - Needs identities and credentials to access multiple SPs.
- Service Provider (SP)
  - Needs to know identity of users, and needs assurance of user 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.
- Broker
  - Intermediary between players (not always used)



Federation protocols

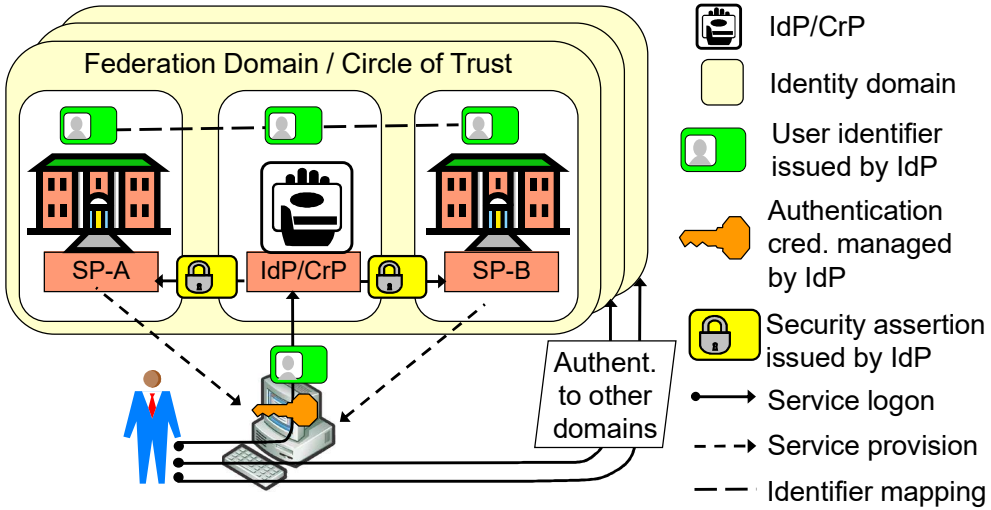
- Authentication by one IdP/CrP/SP is communicated as a security assertions (cryptographic token) to other SPs that trust and accept the assertion of authenticity.
- Usually based on the SAML protocol
  - **S**ecurity **A**ssertions **M**arkup **L**anguage
- Involves multiple players
  - User, IdP, CrP, SP, and sometimes a broker



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 between parties
    - Each federation partner can potentially compromise security
  - Privacy issues,
    - Massive data collection is a threat to data privacy
  - Limited scalability,
    - Limited by political and economical constraints
    - An Identity federation can become a new form of silo

Centralised Federation

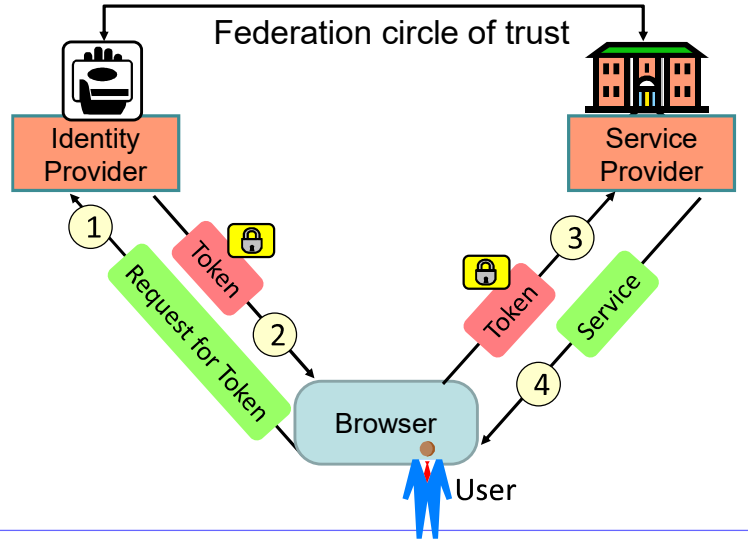


Legend :

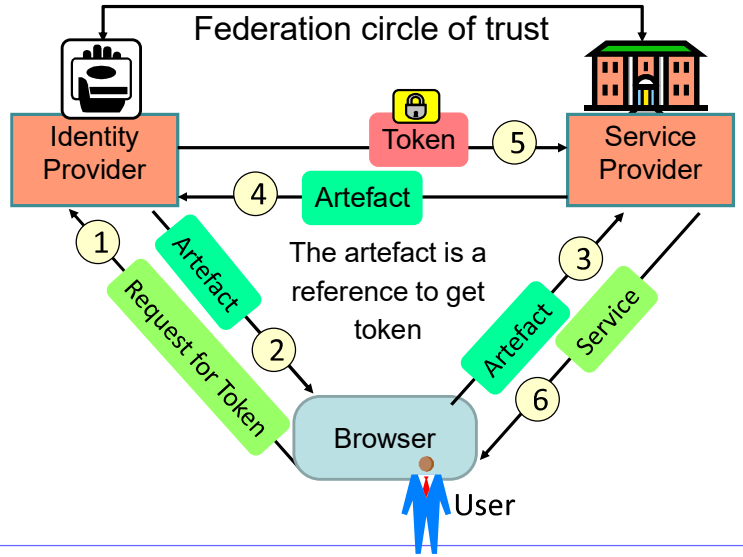
- SP
- IdP/CrP
- Identity domain
- User identifier issued by IdP
- Authentication cred. managed by IdP
- Security assertion issued by IdP
- Service logon
- Service provision
- Identifier mapping

Examples: Facebook connect

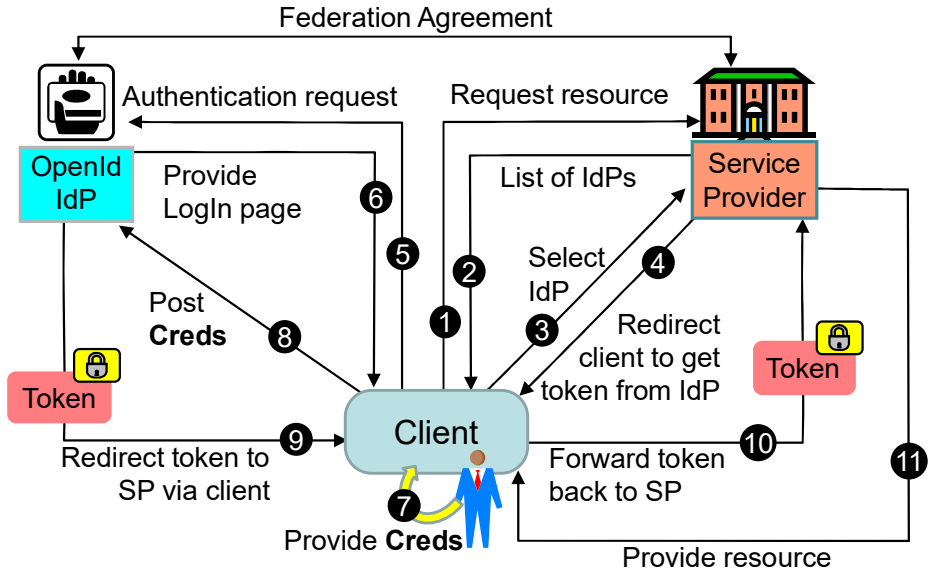
SAML protocol profile: Browser Post Security token via front-channel



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



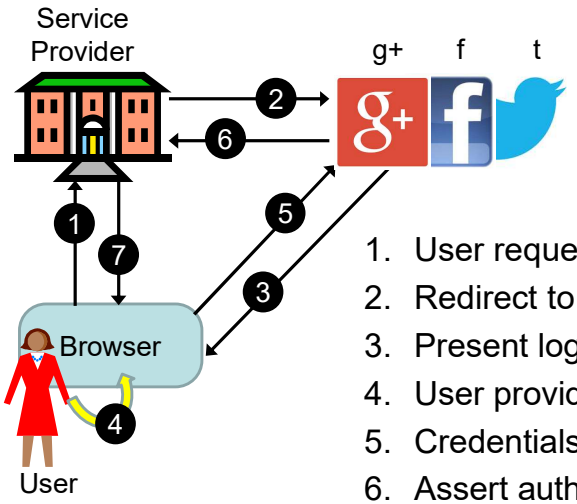
OpenID Connect Protocol



OpenID Connect Characteristics

- Based on OpenID and OAuth 2.0 specifications
- SPs establish federation agreements with IdPs
- Beware of abuse of term “authorization”
  - The OpenId Connect standard uses “authorization” in the meaning of authentication and access control
- OpenID Connect used in the Norwegian HelseID
  - IAM for the Norwegian health sector
  - Health professionals register OpenIds that are independent of their national person numbers
  - Mapping between OpenIds and person number exists but is protected

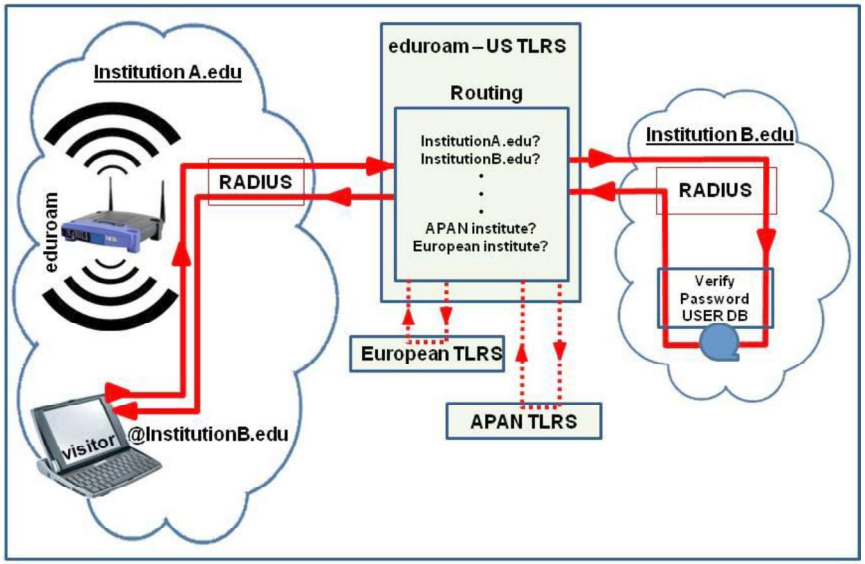
google, facebook and twitter federations



Authentication with google+, FacebookConnect or twitter

1. User requests service
2. Redirect to **g+**, **f** or **t** for authentication
3. Present login form from **g+**, **f** or **t**
4. User provides Id and credentials
5. Credentials forwarded to **g+**, **f** or **t**
6. Assert authenticated user
7. Provide service

Network access



(continued)

- EDUROAM has formal agreements with the public and private locations around Europe for network access
- Home Institutions (universities) are responsible for keeping user data and credentials correct and up-to-date
- Networks provide Internet access.

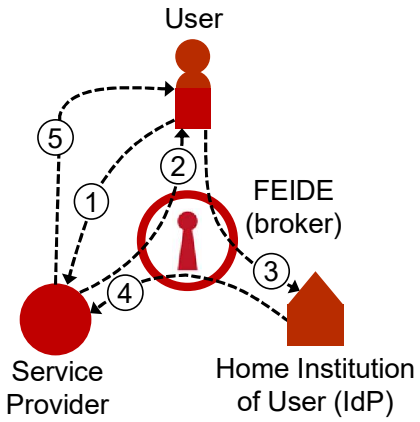
(Felles Elektronisk Identitet)

- 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.

(continued)

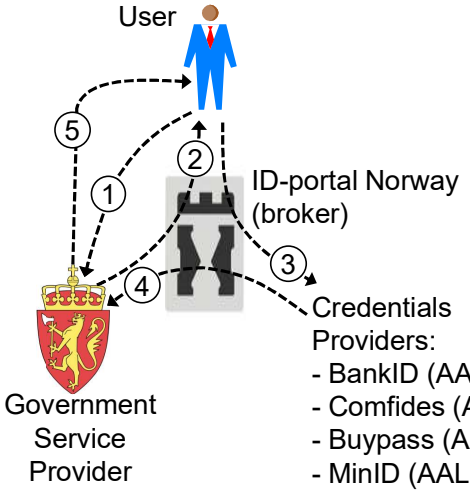
- FEIDE has formal agreements with the universities and schools before they are connected
- Home Institutions (universities and schools) are responsible for keeping user data correct and up-to-date
- Service Providers decide themselves what services their own users and other users should be able to access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

FEIDE Scenario



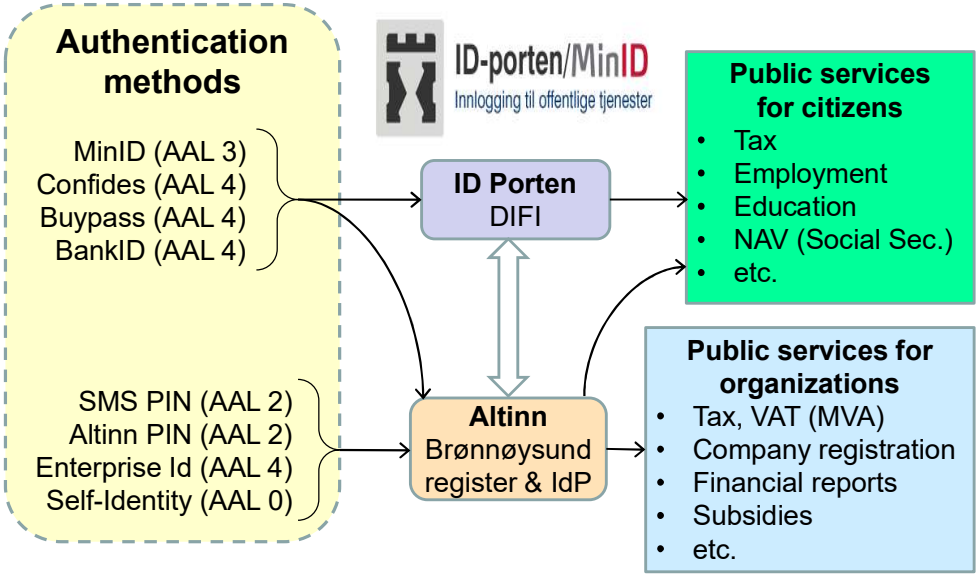
1. User requests access to service
2. Service Provider sends authentication request to FEIDE, and displays FEIDE login form to user.
3. User enters name and password in FEIDE login form, which are sent for validation to Home Institution of user.
4. Home Institution confirms authentic user and provides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

ID-porten/MinID Scenario



1. User requests service access
2. Service Provider sends authentication request to Id-portal, and displays ID-portal login form to user.
3. User selects credentials provider, enters name and password in login form, which are sent for validation to credentials provider of user.
4. Credentials provider confirms authentic user and provides user attributes to ID-portal which forwards these to SP
5. Service Provider analyses user attributes and provides service according to policy

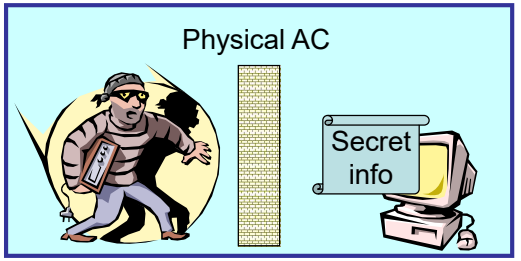
Norw. e-Gov. Distributed Fed. with Broker



Introduction to Logical Access Control

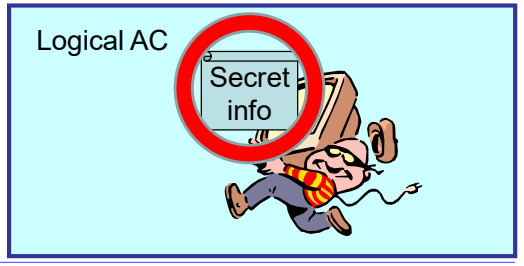
Physical Access Control:

(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

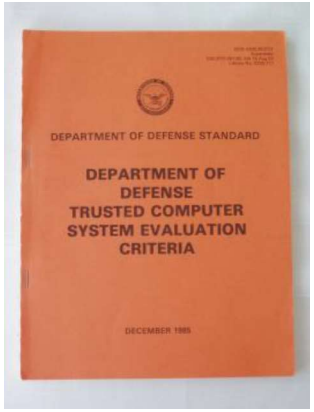
DAC / MAC from the Orange Book (TCSEC)

TCSEC (1985) specifies two AC security models

- Discretionary Access Control (DAC)
  - aka. **Name-Based Access Control**
  - AC policy based on user/group names
  - e.g. *John has* `(r,w)` *- access to HR-files*

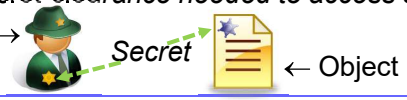
Subjects →	HR	Sales
John	`r,w`	
Mary		`r,w`

← Objects



Orange Book 1985

- Mandatory AC (MAC)
  - aka. **Label-Based Access Control**
  - AC policy based on security labels
  - e.g. *secret-clearance needed to access secret-classified document*



DAC – Discretionary Access Control (Name-Based Access Control)

- Access authorization is specified and enforced based on the name/identity of subjects/objects.
- Typically implemented as 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 for access
- 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		Object names			
		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	-	-

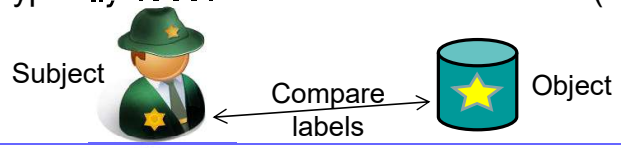
	O1	O2	O3	O4
S4	r,w	x	x	x

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

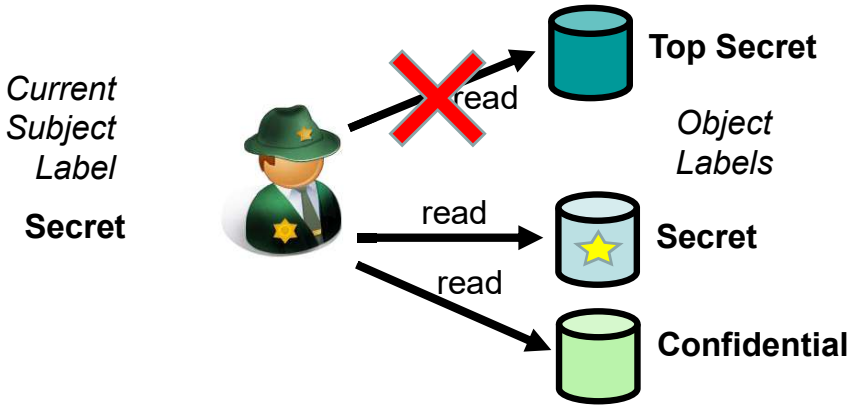
SS-property (Simple Security): No Read Up

- 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**

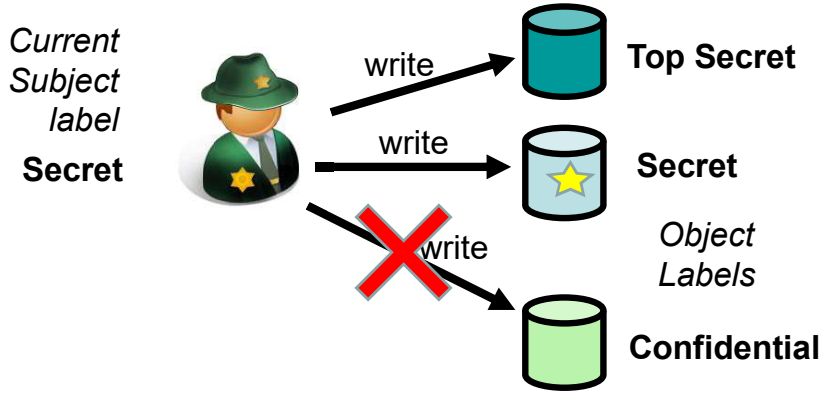
- Subjects working on information/tasks at a given level should not be allowed to write to a lower level, because otherwise it could create unauthorized information flow.
- So you should only be able write to files with an equal or higher label as your security clearance level.

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



Diagram

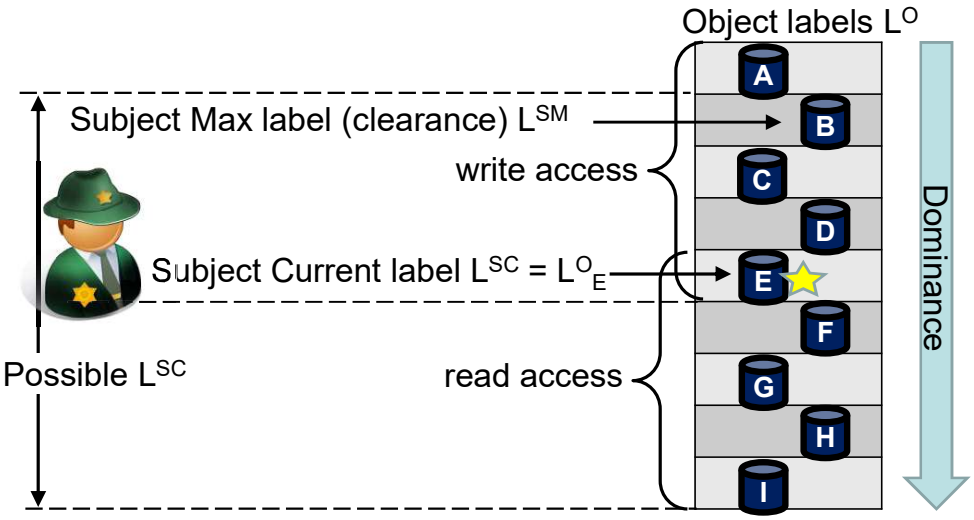
Bell-LaPadula (MAC model) *-Property: No Write Down



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 $L^{SC} \geq L^{O}$
  - Label L^{SC} dominates label L^{O}
- *-property allows write-access when $L^{SC} \leq L^{O}$
  - Label L^{O} dominates label L^{SC}
- Simultaneous read- and write-access requires $L^{SC} = L^{O}$

Bell-LaPadula label relationships



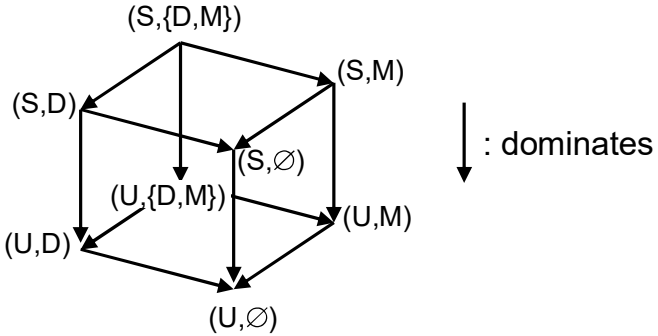
Partial Ordering of MAC Labels

- Example: Define a label L = (h, c) where h and c are label-parameters which take values from sets H and C

  h $\in$ hierarchical set H = {Secret, Unclassified} = {S, U}

  c $\subseteq$ category set C = {Development, Marketing, $\varnothing$} = {D, M, $\varnothing$}

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
  - Subject current label: $L^{SC} = (h^{SC}, c^{SC})$,
  - Object label: $L^{O} = (h^{O}, c^{O})$
- Dominance: $L^{SC} \geq L^{O}$ iff $(h^{O} \leq h^{SC}) \wedge (c^{O} \subseteq c^{SC})$
  - In case $L^{SC} = L^{O}$ then also $L^{SC} \geq L^{O}$ and $L^{O} \geq L^{SC}$
- Non-dominance cases: $L^{SC} \not\geq L^{O}$
  - $(h^{O} > h^{SC}) \wedge (c^{O} \subseteq c^{SC})$; insufficient hierarchic level
  - $(h^{O} \leq h^{SC}) \wedge (c^{O} \not\subset c^{SC})$; insufficient category set
  - $(h^{O} > h^{SC}) \wedge (c^{O} \not\subset c^{SC})$; insufficient level and category

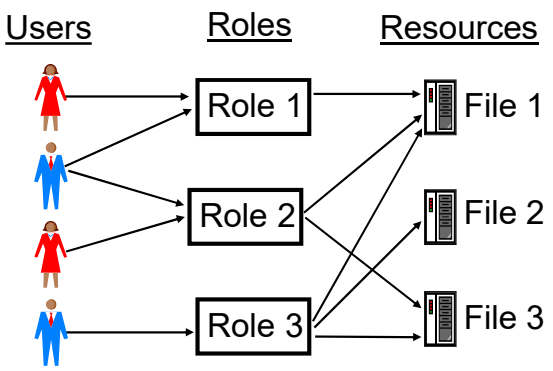
Combined MAC & DAC

- Combining MAC and DAC access control:
  - It can be useful to combine MAC and DAC access control
    - MAC policy is applied first,
    - DAC policy applied subsequently in case of positive MAC
    - Access granted only if both MAC and DAC decisions are positive
  - Advantage:
    - MAC ensures that users with insufficient clearance label in terms of level and category can not access resources with a dominant classification label
    - DAC makes it possible to enforce 'need to know' to limit access that would otherwise be granted under the MAC policy

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

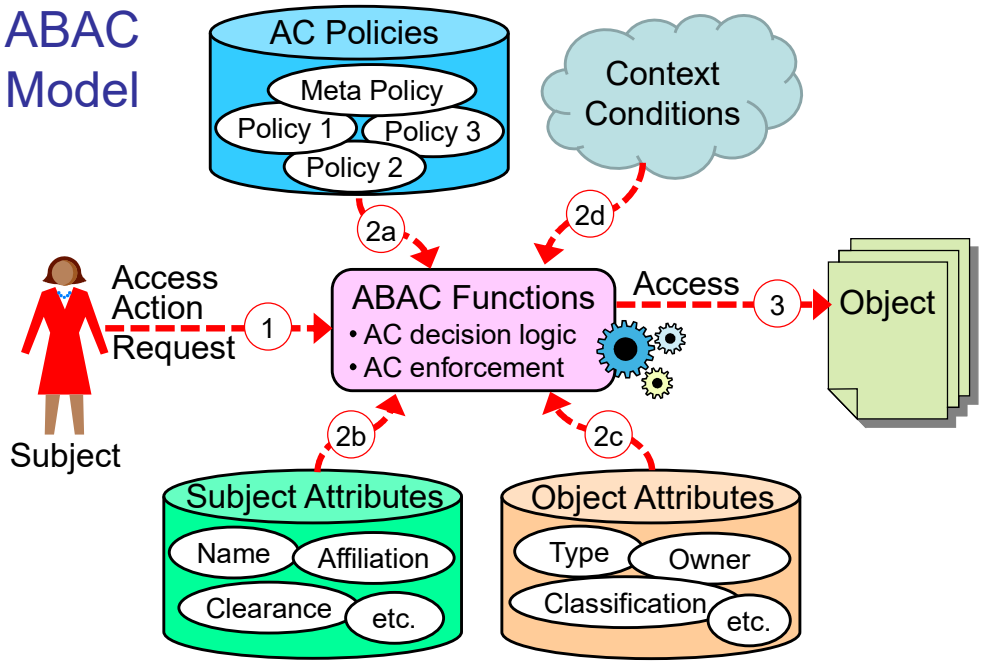
• The XACML standard defines a language for expressing access control attributes and policies implemented in XML, and a processing model describing how to evaluate access requests according to the rules defined in policies.

• 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



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 authorization 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 AC 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 are 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 different policies could dictate different access decisions
- Meta-policy rules needed in case the ABAC logic detects policy rules that lead to conflicting decisions
- Meta-policy takes priority over all other policies, e.g.
  - Meta-Policy Deny Override: If one policy denies access, but another policy approves access, then access is denied. This is a conservative meta-policy.
  - Meta-Policy Approve Override: If one policy denies access, but another policy approves access, then access is approved.
  - This is a lenient meta-policy.

End of lecture