IT Security

Chapter 4: Authentication and Authorization

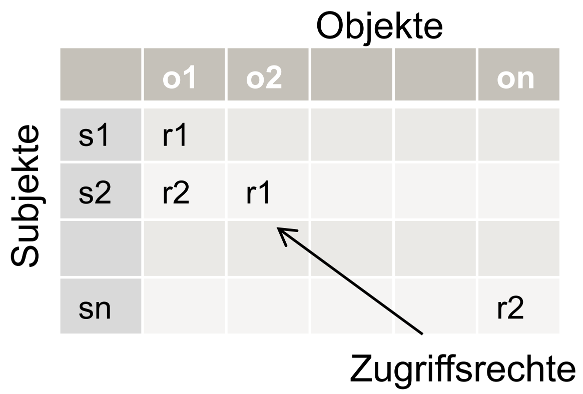
Part 2

Authorization – principles of access control

- **Discretionary-Access-Control (DAC)**
  - user-definable access control
  - each owner can transfer rights to his objects to other users
  - the assignment of rights is controlled decentral
  - restricting access to objects based on the identity of the subject

- **Mandatory Access Control (MAC)**
  - system defines security properties (rule-based)
  - user-defined rights are overridden (dominated) by system-defined ones
  - additional security classes and global rules are introduced
  - limiting access to resources is based on the sensitivity of the information
  - operating systems or applications must provide special measures and services to enforce MAC policies

Realization of access control via access matrix



	o1	o2			on
s1	r1				
s2	r2	r1			
sn					r2

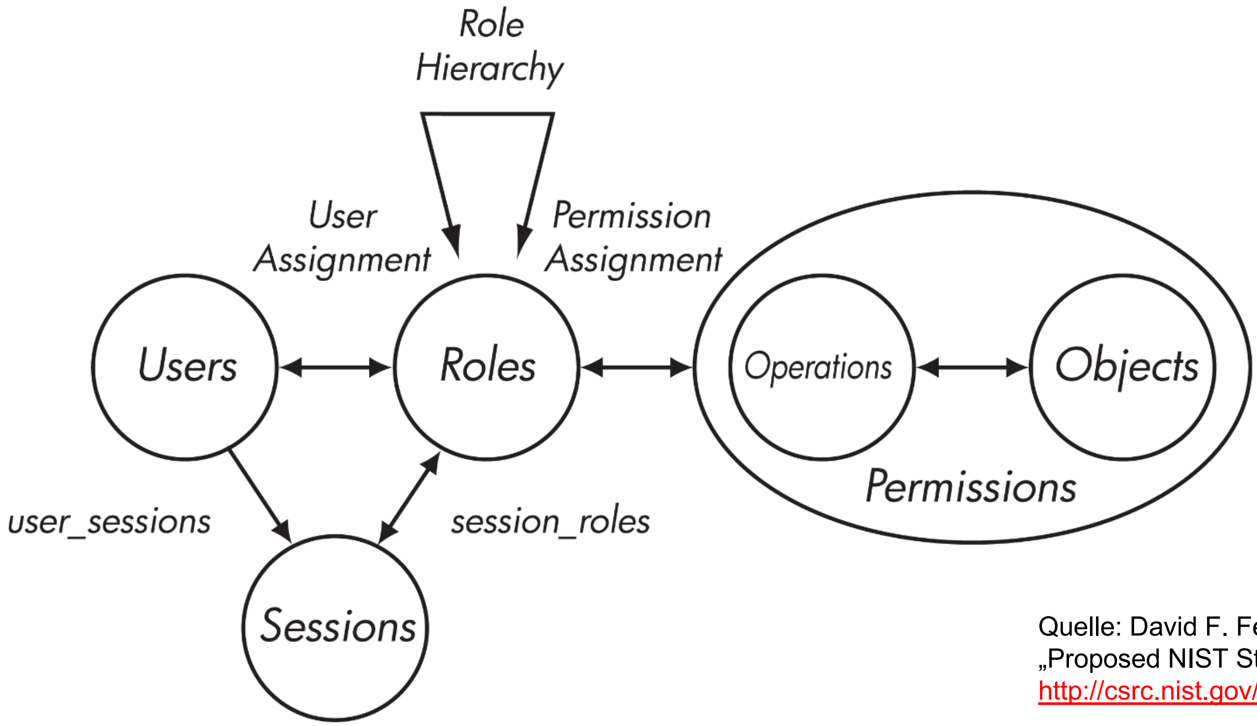
- Access control is often implemented with a (sparse) access matrix that can be viewed in two dimensions

- Access control list - **Access Control Lists (ACL)**
  - object-based view, one list per object to be protected
  - ACL define the access rights of subjects to objects
  - advantage: easy administration and revocation of rights
  - disadvantage: sometimes inefficient with many subjects

- Access tickets - **Capabilities (permissions)**
  - subject-based view
  - Tamper-proof tickets that authorize the holder to access an object
  - advantage: flexible, decentralized, suitable for delegation
  - disadvantage: withdrawal of rights is time-consuming

Role Based Access Control (RBAC)

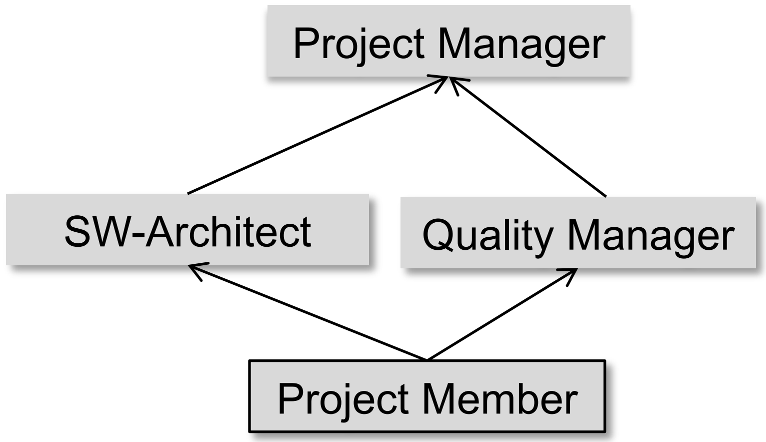
- **Rolle-Based-Access**-Control-Pattern
  - the permissions for objects are assigned to roles (**pr = permission to role**)
  - subjects are assigned to roles (**sr = subject to role**)



Quelle: David F. Ferraiolo, Ravi Sandhu, Serban Gavrila, R. Richard Kuhn, „Proposed NIST Standard for Role-Based Access Control“, 2001
http://csrc.nist.gov/rbac/rbacSTD-ACM.pdf

Components of a RBAC model

- **Sessions**
  - a session means a subject is active in a role
  - a subject may only be active in roles of which he is a member
  - a subject has only the rights of his active role
- **Role hierarchy**
  - goal : replication of organizational structures
  - definition of a partial order on roles
    $R_i, R_j \in Role, if\ R_i \leq R_j$
    then R_i has all rights of R_j



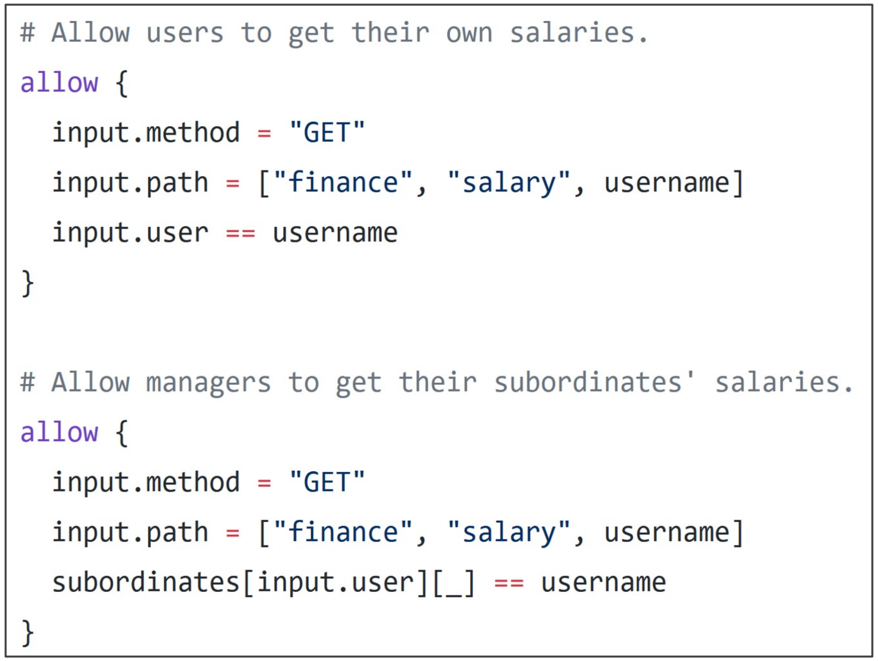
- **Static separation of duties**: mutual exclusion of role memberships.

What are the benefits of RBAC-Model?

- Role concepts are very flexible to use, task-oriented, administrable and scale well
- They allow direct replication of known organizational and rights structures in companies and are a good basis for ID management
- Intuitive and relatively simple mapping of roles to business processes (workflows) enables **need-to-know** rights assignment
- Changes to ***pr*** are rare;
  but changes to role memberships ***sr*** are frequent;
- Simple and efficient rights management, automatic rights revocation at end of membership.
- Danger: Roles are misused to represent permissions, which can lead to an exploding number of roles

Rule-Based Access Control (RuBAC)

- Access control based on rules
- Typical applications:
  Firewalls, routers
- With user rights it can be used with MAC
  - rules describe situations in which a subject can access an object
  - RuBAC quickly become very complex
- Rules can be described with policies:
  Policy-Based-Access Control PBAC



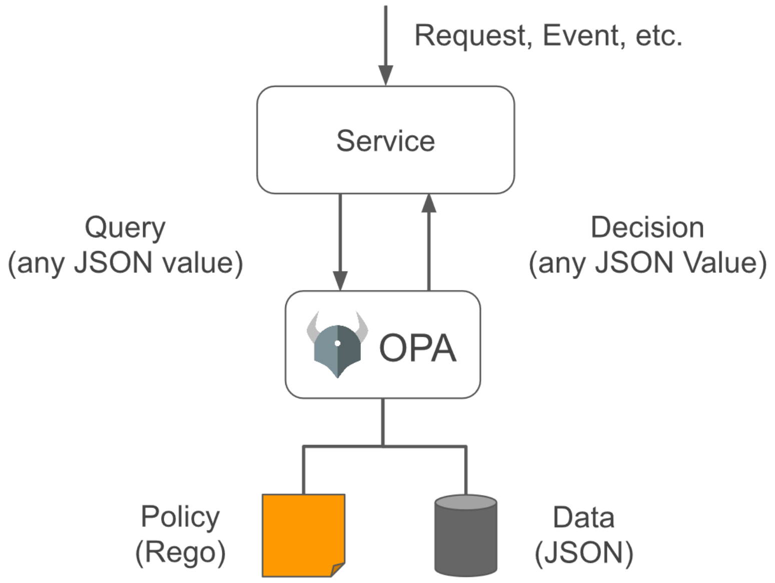
```
# Allow users to get their own salaries.
allow {
  input.method = "GET"
  input.path = ["finance", "salary", username]
  input.user == username
}

# Allow managers to get their subordinates' salaries.
allow {
  input.method = "GET"
  input.path = ["finance", "salary", username]
  subordinates[input.user][_] == username
}
```

Example of an Open Policy Agent in Policy language Rego („ray-go“)

Open Policy Agent

Open Policy Agent OPA enables Rule Based Access Control in cloud environments

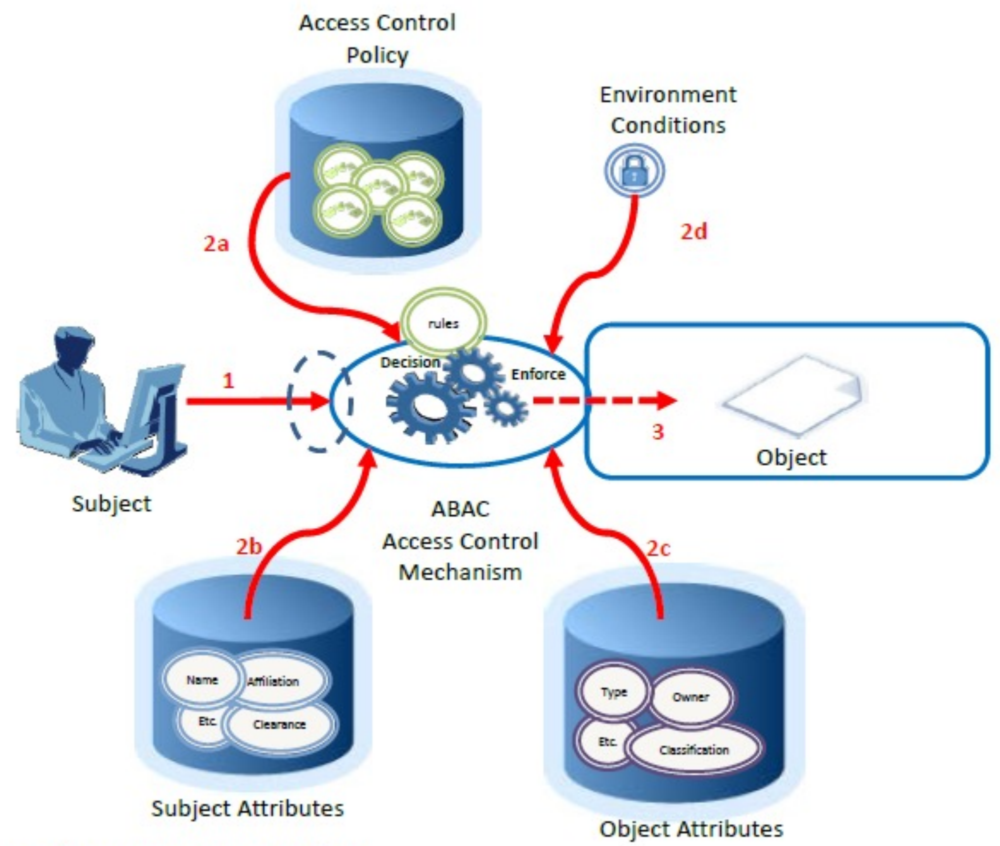


OPA generates policy decisions by evaluating the query input and against policies and data

OPA enables fine-grained policy-based control in cloud native environments

Quelle: https://www.openpolicyagent.org/

Attibute Based Access Control (ABAC)



1. Subject requests access to object
2. Access Control Mechanism evaluates a) Rules, b) Subject Attributes, c) Object Attributes, and d) Environment Conditions to compute a decision
3. Subject is given access to object if authorized

Quelle: NIST http://nvlpubs.nist.gov/nistpubs/specialpublications/NIST.sp.800-162.pdf

- OpenID Connect enables access control based on attributes (claims)
- ABAC is attribute based
- XACML eXtensible Access Control Markup Language is an attribute-based access control policy language.
- Application area:
  - API gateway for micro services
  - Access to Big Data systems

Access Control Patterns

- **Least privilege**: a subject should be given only those privileges needed for it to complete its tasks, raises system stability and security
- **Need to Know**: user gets access only if it‘s necessary to conduct its duties
- **Separation of Duty**: more than one user is required to complete a task, increases protection from fraud and errors, control against insider attacks
- **Separation of Concerns**: separate a computer program into distinct sections
- **Open Policy**: everything is allowed which is not forbidden
- **Closed Policy**: only explicit authorized access is allowed
- **Dual Control:** Four eyes principle, two or more separate entities are necessary to access sensitive functions or information

Summary Authentication and Authorization

- There are many different variants for authentication
- They differ in security, mobility, cost and convenience
- There are different standards and technologies for SSO (Kerberos, OAuth, OpenID, SAML)
- RBAC is a very flexible model for managing access rights
- In modern cloud environments, RuBAC and ABAC also play an important role