# Access Control

Josh Wilcox (jw14g24@soton.ac.uk)

March 7, 2025

## Contents

# 1 Definition

- Process of granting or denying specific requests to
  - Obtain and use information and relation information processing services
  - Enter specific physcial facilities

# 2 Principles

- Preventing unauthorised users from gaining access to resources
  - External people can not access
- PRevents legitimate users to access resources in an unauthorised manner
  - Dishonest people can not access
- Enables ligitimate users to access resources in an authorised manner
  - Honest people can access
- Specifies who or what has access to each specific resource
  - And the type of access permitted in each instance

# 3 Access Control Context

- Authentication
  - Whether credentials are valid
- Authorisation
  - Granting of a right or permission to access system resources
- Audit
  - Independent review and examination of system record

# 4 Subjects, Objects and Access Rights

## 4.1 Subject

- Entity capable of accessing objects
- Process that represents a user or application
- Thre classes:
  - Owner
  - Group
  - World

## 4.2 Object

- Resource to which access is controlled
- Entity used to contain or receive infromarion

### 4.3 Access Right

- The way the subject accesses an object
- Examples:
  - Read
  - Write
  - Execute
  - Delete
  - Create
  - Search

# 5 Access Control Models

## 5.1 What are policies and models

- Policies define what is allowed
  - Defines acceptable executions in a system
  - Also what executions are unacceptable
  - Analogous to a set of laws
  - Can be enforced locally or in a network
- Security Model
  - Provides representation of a class of systems
  - Descriptions of system behaviours that guide the design of policies

## 5.2 Access Control Models uses

- Define a specific set of authorisation rights
- Define a set of policies for a software system to enforce a set of rights to fulfil security concerns
- Protect for **all** multiuser systems against violation of **CIA**:
  - Confidentiality
  - Integrity
  - Availability

## 5.3 Main models of access control

- Discretionary access control
  - Based on requestor identity
- Mandatory access control
  - Based on comparing security label
- Role-based access control
  - Based on the roles of users
- Attribute-based access control
  - Based on attributes of subject, object, and the environment

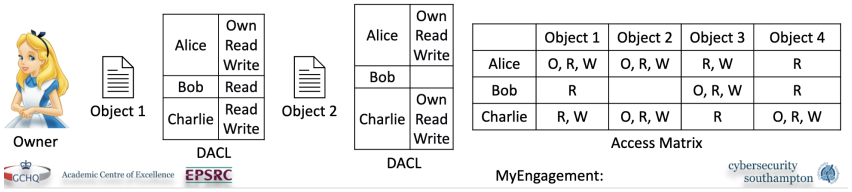
### 5.3.1 Not Mutually Exclusive

- Each model is not mutually exclusive
- An access control mechanism can use attributes from multiple

# 6 Discretionary Access Control - Requestor Identity

## 6.1 Process

- Every object has:
  - An owner
  - A Discretionary Access Control List
    - This contains the permissions of the subjects
- Discretionary Access Control List (A Matrix):
  - Rows - Subjects
  - Columns - Objects
  - Fields - Whether they have Ownership, Read, and Write rights




## 6.2 Principle

- Users own resources and control their access
- Onwer can change object permissions
- Owners can transfer ownership

## 6.3 Issues

- Flexible but open to mistakes
  - Requires all users understand mechanisms and respect security
- Managing policies for a large system is complex
- Objects and subjects change frequently
  - Also their permissions change frequently

# 7 Mandatory Access Control - Security Clearance

## 7.1 Process

- Classifies subjects and objects by **security levels**
  - Each subject has a profile which includes their security **clearance** and need-to-know
    - Clearance - The extent of sensitivity a user can access
    - Need-To-Know - Whether a user is involved in a certain object
  - Every object has a label of two parts

    * Classification - Clearence required
    * Catageory - Enforcement of need-to-know

## 7.2 Properties

- Often identified with multi-level security policies
- Prevents data leakage
- More rigid than DAC, but also more secure
  - Stronger security
  - Less operational **flexibility**
- Mandatory

# 8 Role-Based Access Control - Roles

## 8.1 Process

- Access is based on a user's **role** in an organisation
  - Each role has **permissions** associated with them

| Role |
|---|
| User |
| Superuser |

| Object |
|---|
| Object 1 |
| Object 2 |
| Object 3 |

| Role | Permission |
|---|---|
| User | Read, Write – Object 1 |
| User | Read – Object 2 |
| User | Own, Read, Write – Object 3 |
| Superuser | Own, Read, Write – Object 1 |
| Superuser | Own, Read, Write – Object 2 |
| Superuser | Read – Object 3 |

| User | Role |
|---|---|
| Alice | User |
| Alice | Superuser |
| Bob | User |
| Charlie | User |

## 8.2 Intuition

- Many objects and subjects can have identical attributes based on their role
  - Policy is based on these attributes
- Allows streamlining for companies that use organisational hierarchy
- "Roles" are the central authorisation mechanism
  - **Role** - Representation of a group of subjects that are allowed to perform the same operations on the same objects based on their role

## 8.3 Advantages

- Roles are an **abstraction** of jobs or functions
  - Different from user groups - which are just collections
  - Emphasis on **responsibility** and associated permissions
- Policies become more manageable
- Reduces user admin
- Easy to audit
- Higher flexibility and scalability

## 8.4 RBAC Family

### 8.4.1 Role Hierarchies

- Role Hierarchies allow one role to **inherit** permissions from another role
- Simplifies expression of policies to different roles

### 8.4.2 Constraints

- A constraint is a relationship among roles or a condition related to roles
- They restrict the ways in which the components of an RBAC may be configured
- **Mutually Exclusive Roles**
  - Constraint to define that a user can only be assigned exactly one role from a set of mutually exlcusive roles
    - Separates duties and capabilities
  - *Example: A user cannot be both an auditor and a cashier to prevent fraud.*
- **Cardinality**
  - Constraint to set the maximum number of roles in the system
  - *Example: Limiting the number of system administrators to maintain better control.*
- **Prerequisite Roles**
  - A constraint to only allow the addition of a role to someone if they already have a prerequisite roles
  - *Example: A user must be a trained nurse before they can become a head nurse.*

# 9 ABAC - Attribute Based Access Control

## 9.1 Attributes

- Characteristics that define specific aspects of a subject, object, environment, or operation
- Subject attributes define the identity and characteristics of the subject
  - ID, name, organisation, job title etc.
- Object attributes are often extracted from the object metadata
- Environment attributes describe operational and situational context which the information access occurs

## 9.2 Methodology of ABAC

- Access control decisions are made by evaluating specified rules against the attributes of entities, operations, and the environment.
- Instead of roles, ABAC uses attributes to define access rights.
- Attributes can be associated with:
  - **Subject**: User attributes like age, department, security clearance, job title, etc.
  - **Object**: Resource attributes like file type, creation date, sensitivity level, data owner, etc.
  - **Environment**: Contextual attributes like time of day, location, network connection type, etc.
  - **Action**: The specific operation being attempted, such as read, write, execute, delete, etc.
- Access is granted if the attributes satisfy the defined policies.

- Policies are expressed as rules that combine these attributes.
- Example Policy:
  - "A user in the HR department (subject attribute) can read (action) a document (object) if the document's sensitivity level is 'internal' (object attribute) and the access is attempted during business hours (environment attribute)."

## 9.3 Advantages and Dista

- Dynamic - Evaluated **at time of request**
- Contextual - Environmental conditions are more easily considered
- Fine-Grained - Rules can be **extremely granular** and situational
  - A larger, and more definitive, set of rules can be defined
  - Can enforce DAC, MAC, and RBAC

**Disadvantange** - Rather complex!

# 10 Comparison of Access Control Methods

| Method | Pros | Cons |
|---|---|---|
| **Discretionary Access Control (DAC)** | • Flexibility for resource owners<br>• Simple and intuitive for users<br>• Facilitates easy collaboration<br>• Users can quickly adjust permissions<br>• Owners can delegate access control<br>• Well-suited for user-managed environments | • Vulnerable to user errors and poor security practices<br>• Susceptible to malware exploiting user privileges<br>• No central administrative control<br>• Difficult to maintain consistent security policies<br>• Can lead to information leakage<br>• Complex administration in large systems<br>• No control over information flow after access is granted |
| **Mandatory Access Control (MAC)** | • Strong security enforcement through centralized control<br>• Effective protection against data leakage<br>• Consistent policy enforcement<br>• Protection against malware and Trojan horses<br>• Systematic implementation of security levels<br>• Controls information flow effectively<br>• Policies cannot be altered by users | • Rigid and inflexible structure<br>• Significant administrative overhead<br>• Reduced operational flexibility<br>• Can impede legitimate work processes<br>• Complex to implement correctly<br>• Requires careful classification of all subjects and objects<br>• Potential for over-classification limiting data access |
| **Role-Based Access Control (RBAC)** | • Simplifies administration through role abstractions<br>• Aligns well with organizational structures<br>• Reduces administrative overhead<br>• Supports principle of least privilege<br>• Scalable for large organizations<br>• Easier to audit access rights<br>• Supports separation of duties through constraints<br>• Simplifies user onboarding and offboarding | • Role explosion in complex organizations<br>• Difficult to handle exceptions or temporary access<br>• Less flexible for dynamic/context-based requirements<br>• Can become complex with many roles and hierarchies<br>• Initial setup requires careful role planning<br>• No inherent support for context-based decisions<br>• Role maintenance requires regular reviews |
| **Attribute-Based Access Control (ABAC)** | • Highly flexible and dynamic access decisions<br>• Context-aware by incorporating environmental factors<br>• Implements fine-grained access control<br>• Adaptable to complex organizational needs<br>• Reduces administrative overhead in large complex systems<br>• Can enforce DAC, MAC, and RBAC policies<br>• Better handles temporal and spatial constraints<br>• Easier to adapt to changing requirements | • Complex to implement and maintain<br>• Performance overhead from attribute evaluation<br>• Difficult to audit and validate policies<br>• Potential for conflicting rules<br>• Requires comprehensive attribute management<br>• More difficult for users to understand access decisions<br>• Can lead to unexpected access denials if attributes change<br>• Requires sophisticated policy engines |

## 10.1 Recommended Use Cases

### 10.1.1 Discretionary Access Control (DAC)

- **Personal computing environments** where users manage their own files and resources
- **Small collaborative teams** where trust levels are high and sharing is frequent
- **Research environments** where researchers need flexibility to share data with collaborators
- **Creative industries** where work products are frequently shared for feedback
- **Development environments** where developers need to control access to their code repositories
- **File sharing systems** where users upload and manage their own content

### 10.1.2 Mandatory Access Control (MAC)

- **Military and intelligence systems** handling classified information
- **Government agencies** with strict information security requirements
- **Financial institutions** requiring strict data segregation and leak prevention
- **Healthcare systems** storing highly sensitive patient information
- **Critical infrastructure control systems** where security breaches could cause catastrophic damage
- **Systems processing trade secrets** or intellectual property requiring utmost protection
- **Environments with strong regulatory compliance** requirements like GDPR or HIPAA

### 10.1.3 Role-Based Access Control (RBAC)

- **Enterprise environments** with clear organizational structures and job functions
- **Healthcare systems** with well-defined roles (doctors, nurses, administrators, patients)
- **Banking and finance organizations** with hierarchical structures and regulatory requirements
- **Educational institutions** separating access for faculty, staff, and students
- **E-commerce platforms** with different permissions for customers, sellers, and administrators
- **ERP systems** where access maps to business functions and departments
- **Government agencies** with formal organizational hierarchies

### 10.1.4 Attribute-Based Access Control (ABAC)

- **Complex organizations** with dynamic access requirements that change based on context
- **Cloud computing environments** serving diverse users with varying access needs
- **IoT systems** managing access among diverse devices and users
- **Healthcare systems** requiring context-aware access (emergency situations, patient relationships)
- **Multi-national organizations** dealing with varying privacy regulations by region
- **Inter-organizational information sharing** where organizational boundaries must be respected
- **Zero-trust security architectures** requiring constant verification of multiple attributes
- **Systems with temporal access requirements** (time-limited access, time-of-day restrictions)