

# Access Control and Authorisation (1)

Budi Arief

[b.arief@kent.ac.uk](mailto:b.arief@kent.ac.uk)

Based on slides by Shujun Li

- **Basic Concepts**
- The Zoo of Access Control Models
  - ACL
  - DAC
  - RBAC
  - MAC
  - ABAC/PBAC
  - RAC
- Summary

# What is access control?

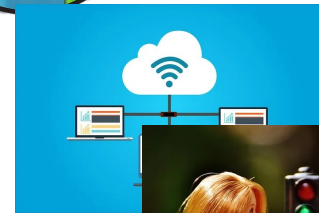


- [Wikipedia](#)
  - *"In the fields of physical security and **information security**, access control (AC) is the **selective restriction of access** to a place or other resource."*
  - *"The act of accessing may mean consuming, entering, or using. Permission to access a resource is called **authorization**."*
- [IETF RFC 4949](#) "Internet Security Glossary, Version 2"
  - *"**Protection** of system resources **against unauthorized access**."*
  - *"A process by which use of system resources is regulated according to a **security policy** and is **permitted only by authorized entities** (users, programs, processes, or other systems) according to that policy."*
  - *"/formal model/ Limitations on **interactions between subjects and objects** in an information system."*

# What is access control?

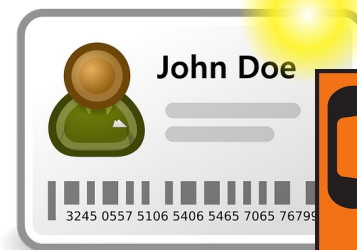
## - Three aspects

1. Who issued the request?
  - Who: a person, a process, a machine, an entity, ...
2. What is requested?
3. Which rules (policies) are applicable when deciding on the request?



## - Two steps

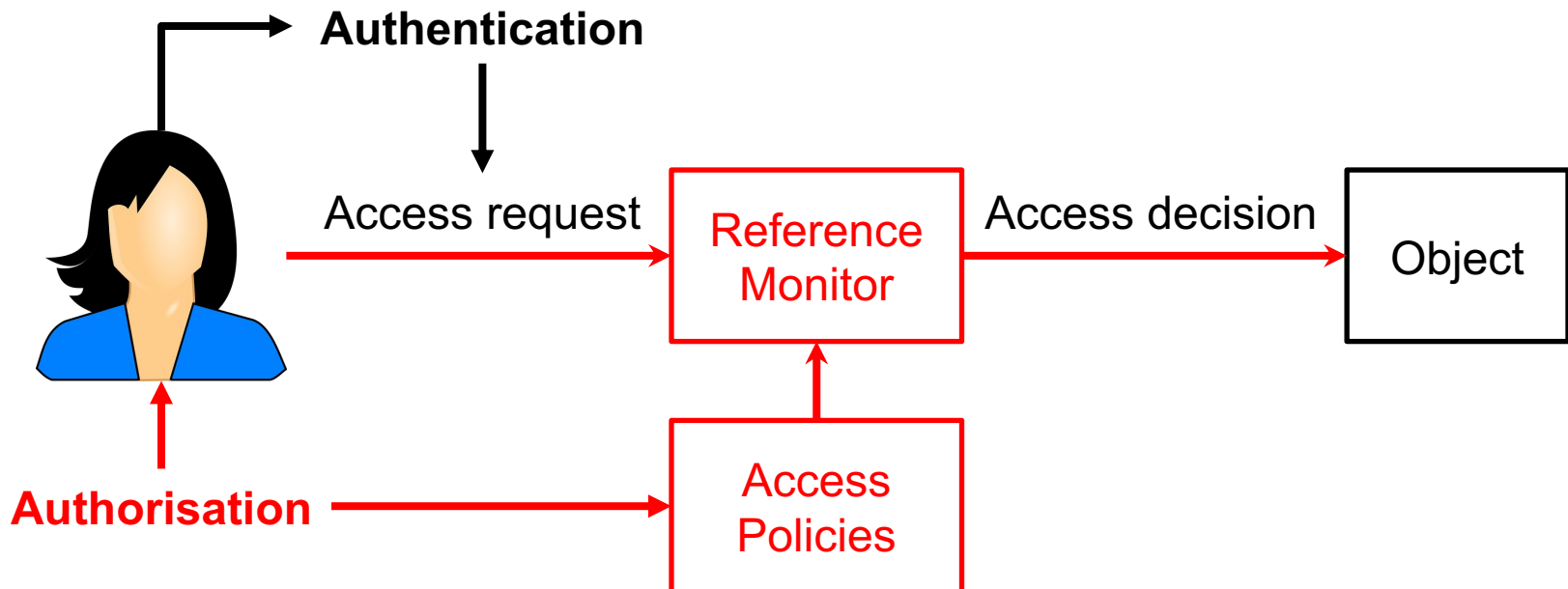
1. (User) Authentication
2. Authorisation



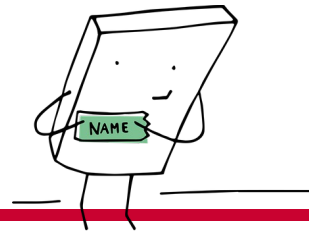
# What is access control?




- The general procedure
  - The access requester is called a subject or a principal, which can be a user or a non-user entity.
  - Reference monitor: the monitor (enforcer) of the access policies



# What is access control?



- More related concepts and terms 
  - Long-term authorisation tends to be called granting of access rights.
    - Who grants access rights to others is a grantor.
  - Delegation: two possible meanings
    - Short-term, temporary granting of access.
    - A grantor (delegator) granting access to other subjects but itself does not have the permission to exercise such rights.
    - Examples: “in-class” exercise!
- Revocation
  - Granted rights can be limited to a defined period of time so will expire at the end of the period, or can be revoked at any time when necessary.
  - Examples: revocation of a card access to a building or a digital certificate.

# “In-class” exercise



- For both types of delegation, find some real-world examples.
- Why are such delegations needed?
- Do some cases involve revocation?
- Hint: Do not just think about computing systems. Think about social, business and political processes, too.

# Outline

---

- Basic Concepts
- The Zoo of Access Control Models
  - ACL
  - DAC
  - RBAC
  - MAC
  - ABAC/PBAC
  - RAC
- Summary



# Access control model: ACL



- **Access control list (ACL)**
  - Used mainly with file systems: a list of permissions associated with a file (or other similar system resources)
  - A real-world example: Permissions of Linux's file system

### Change access rights

You have selected **1 object**.

User may	Group may	Others may
<input checked="" type="checkbox"/> read	<input checked="" type="checkbox"/> read	<input checked="" type="checkbox"/> read
<input checked="" type="checkbox"/> write	<input type="checkbox"/> write	<input type="checkbox"/> write
<input type="checkbox"/> execute	<input type="checkbox"/> execute	<input type="checkbox"/> execute

Octal notation

Text notation

# Access control model: ACL



- **Access control list (ACL)**

- Used mainly with **file systems**: a list of **permissions** associated with a file (or a system resource of another type)
- A real-world example: Permissions of Linux's file system
  - For managing access control to individual files
  - Inherited from UNIX (since the 1960s)
  - **Three types of users**: file owner (u), group (g), and others (o)
  - **Three types of permissions**: read (r), write (w), execute (x); no permission (-)
    - Example: `rwxr-xr--` = owner (rwx, all permissions), group (r-x, read and execute), others (r--, read only)
  - Permissions commonly represented as octal (base-8) notations: each permission of each user group is represented by a bit (0 or 1)
    - Example: `rwxr-xr--` = `111101100` = 754



# Outline

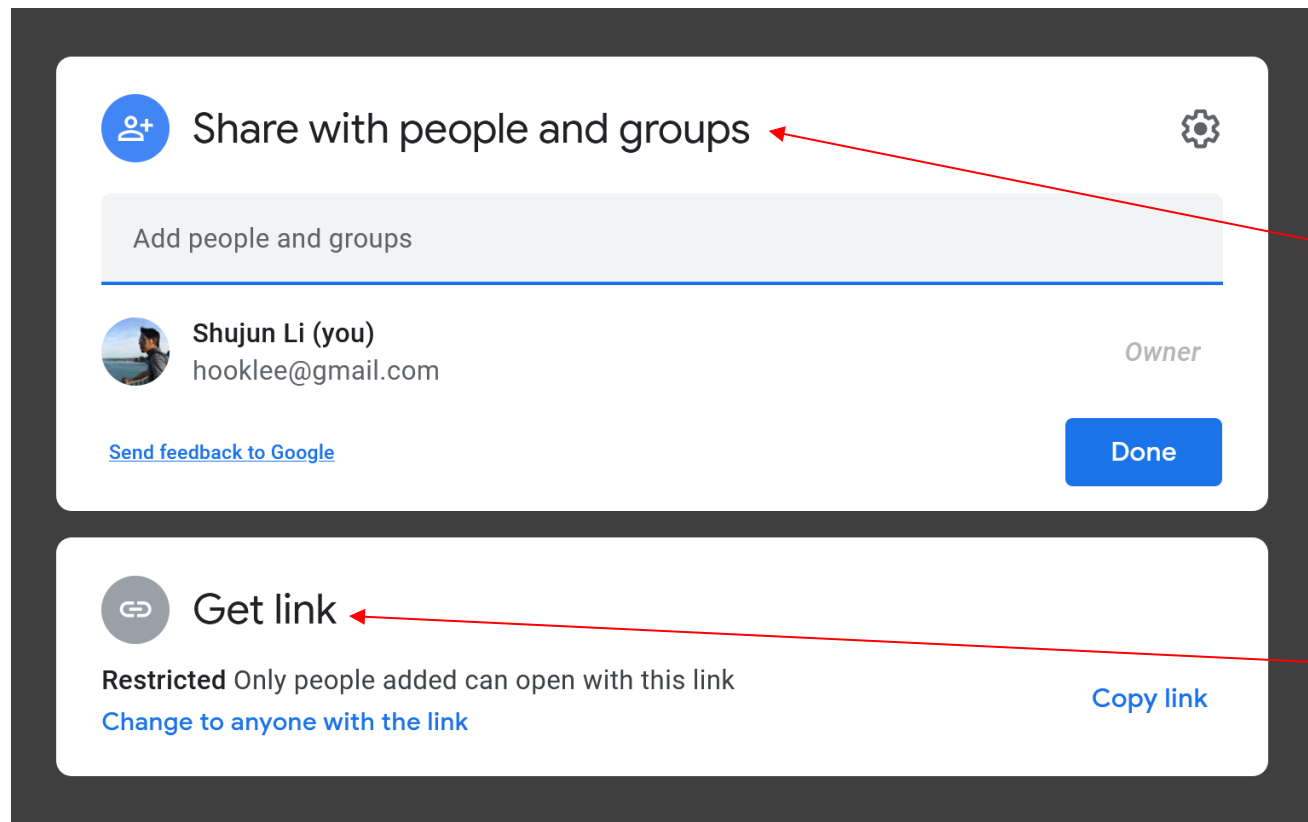
- Basic Concepts
- The Zoo of Access Control Models
  - ACL
  - DAC
  - RBAC
  - MAC
  - ABAC/PBAC
  - RAC
- Summary

# Access control model: DAC



## - Discretionary Access Control (DAC)

- The **owner** of an object defines the access control policies (often based on individual/group **identities**), based on his/her **discretion**.
- A real-world example: Google Drive shared files / folders



ACL

Non-ACL

# Outline

---

- Basic Concepts
- The Zoo of Access Control Models
  - ACL
  - DAC
  - RBAC
  - MAC
  - ABAC/PBAC
  - RAC
- Summary

# Access control model: RBAC



## - Role-Based Access Control (RBAC)

- Entities (users) have **roles**. Roles have a hierarchy and reflects the context of the application. Access control rules are mapped to roles.
- A real world example: WordPress access control system

Classic Edit

Contact us via [contact@abcp.org.uk](mailto:contact@abcp.org.uk) if you have any queries or suggestions.

Switch to Draft Preview Update

Document Block

Status & Visibility

Visibility [Public](#)

Publish [10 April 2018 8:29 pm](#)

Author [ABCP](#)

[Move to Bin](#)

15 Revisions

Permalink

Featured Image

Excerpt

Discussion

Page Attributes

Languages

**Roles**

[Error Message](#)

**Limit access to the content to users of the selected roles.**

- ☐ Administrator
- ☐ Author
- ☐ Contributor
- ☐ Editor
- ☐ Subscriber

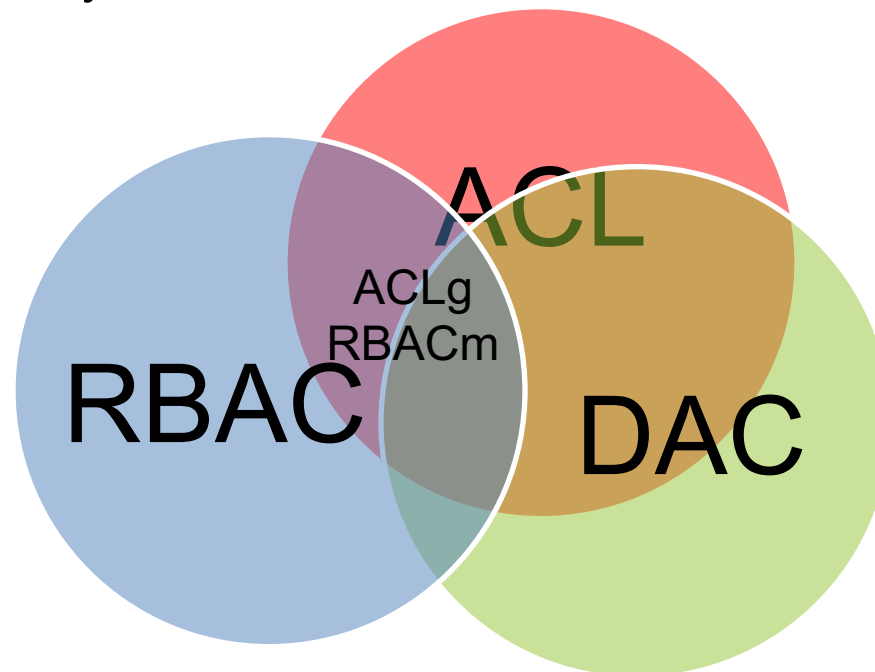
*If no roles are selected, everyone can view the content. The author, any users who can edit the content, and users with the `restrict_content` capability can view the content regardless of role.*

# Access control models: DAC vs RBAC

- DAC can be made a special case of RBAC.
  - ACLg = group-based ACL
    - Use group identities only to assign access control rules.
- RBAC can be made to be comparable with DAC.
  - RBACm = a minimal RBAC model
    - Users, roles, operations, sessions
    - Role-operation associations
    - User-role associations
    - User-session associations
- Groups in ACLg = Roles in RBACm
  - $\Rightarrow$  ACLg = RBACm
  - $\Rightarrow$  The boundary between ACL/DAC and RBAC is not a clear cut.
- John Barkley, "[Comparing Simple Role Based Access Control Models and Access Control Lists](#)," RBAC 1997, ACM

# Access control models: DAC vs RBAC

- An illustrative view of DAC, ACL and RBAC
  - There are different definitions of different AC models.
  - Definitions of those models evolve over time.
    - Historically, DAC often simply referred to ACL-based file permission systems.





- Basic Concepts
- The Zoo of Access Control Models
  - ACL
  - DAC
  - RBAC
  - **MAC**
  - ABAC/PBAC
  - RAC
- Summary

# Access control model: MAC



- **Mandatory Access Control** (MAC)
  - A **central body** sets **mandatory** rules (constraints / restrictions) regarding who can access what (security/sensitivity labels on objects and entities).
  - The central body can be an organisation or a software/hardware system.
  - A real-world example: UK government security classifications + UK national security vetting clearance levels

## OFFICIAL

The majority of information that is created or processed by the public sector. This includes routine business operations and services, some of which could have damaging consequences if lost, stolen or published in the media, but are not subject to a heightened threat profile.

## SECRET

Very sensitive information that justifies heightened protective measures to defend against determined and highly capable threat actors. For example, where compromise could seriously damage military capabilities, international relations or the investigation of serious organised crime.

## TOP SECRET

HMG's most sensitive information requiring the highest levels of protection from the most serious threats. For example, where compromise could cause widespread loss of life or else threaten the security or economic wellbeing of the country or friendly nations.

- UK Cabinet Office, “[Government Security Classifications](#),” published on 18 October 2013, last updated on 21 May 2018 (as of 7 December 2020)

# Access control model: MAC



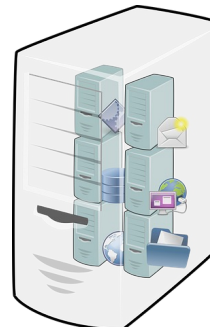
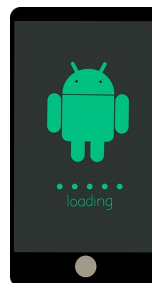
- **Mandatory Access Control (MAC)**
  - A **central body** sets **mandatory** rules (constraints / restrictions) regarding who can access what (security/sensitivity labels on objects and entities).
  - A real-world example: UK government security classifications + UK national security vetting clearance levels
    - Baseline Personnel Security Standard (BPSS): “... access to UK OFFICIAL assets and occasional access to UK SECRET assets.”
    - Security Check (SC): “... long-term, frequent and uncontrolled access to SECRET assets and/or occasional, supervised access to TOP SECRET assets”
    - Enhanced Security Check (eSC): “ ... regular uncontrolled access up to SECRET assets and occasional, controlled access to TOP SECRET assets.”
    - Developed Vetting (DV): “... frequent and uncontrolled access to TOP SECRET assets or ... any access to TOP SECRET codeword material”
    - ...
- United Kingdom Security Vetting, “[National security vetting: clearance levels](#),” published on 12 February 2020

# Access control model: MAC



## - Mandatory Access Control (MAC)

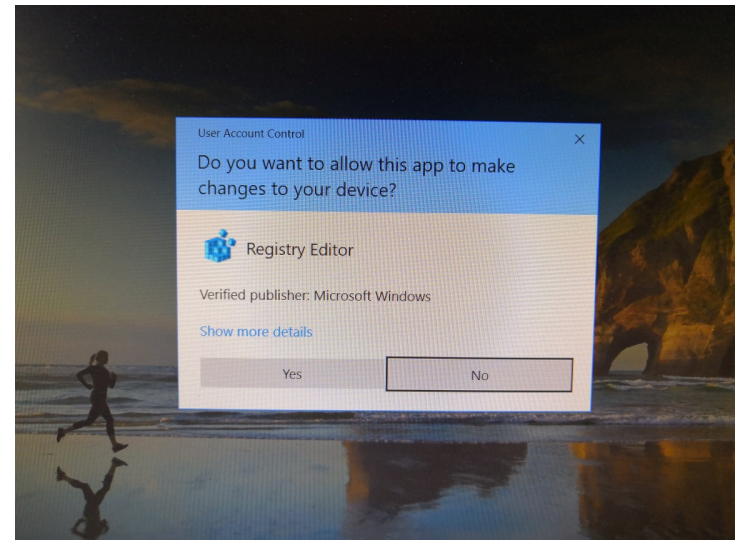
- A **central body** sets **mandatory** rules (constraints / restrictions) regarding who can access what (security/sensitivity labels on objects and entities).
- One more real-world example: Sandboxing
  - Basically, a sandboxed application runs in its own isolated “sandbox”.
    - $\Rightarrow$  If the application is malicious, the harm it can cause is limited.
  - The sandboxing-based isolation is enforced by the OS (or even hardware) and cannot (should not) be changeable by users/applications.
  - Used by many OSs especially mobile OSs for sandboxing mobile apps
  - Virtual machines are running in a sandbox as well.
  - An `<iframe>` element in a web page is also a sandbox.



# Access control model: MAC



- **Mandatory Access Control (MAC)**
  - A **central body** sets **mandatory** rules (constraints / restrictions) regarding who can access what (security/sensitivity labels on objects and entities).
  - A sandboxing example: Windows OS's [User Account Control \(UAC\)](#) / [Mandatory Integrity Control \(MIC\)](#)
    - Isolation of processes with different privileges
    - A number of security-sensitive operations will trigger an UAC prompt for the user to approve.
    - [UAC can however be disabled.](#)
      - What is this?
      - Should you do it?



# Access control model: MAC



## - Mandatory Access Control (MAC)

- A **central body** sets **mandatory** rules (constraints / restrictions) regarding who can access what (security/sensitivity labels on objects and entities).
- More real-world examples: **Origin-based security policies**

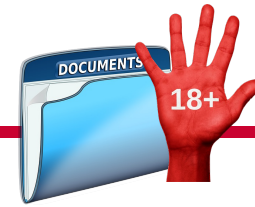


- In web applications, origin-based security policies are often used by the web browser / web server to specify which resources a script in a web page is allowed to access for security purposes.
- The most well-known example is the same-origin policy (SOP) : a web browser allows scripts contained in one web page to access data in a another web page, but only if both are from the same origin.
  - Cross-site Scripting (XSS) attacks aim at bypassing SOP.
- Content Security Policy (CSP) extends SOP to allow the web server to inform the web browser about more fine-tuned policies.
- Cross-origin resource sharing (CORS): A WHATWG protocol established to allow restricted access of resources in one web page by another web page from a different domain.



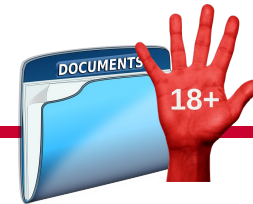
- Basic Concepts
- The Zoo of Access Control Models
  - ACL
  - DAC
  - RBAC
  - MAC
  - ABAC/PBAC
  - RAC
- Summary

# Access control model: ABAC



- **Attribute-Based Access Control (ABAC)**
  - Entities have **attributes**. Access control rules are mapped to attributes.
  - Also called **Policy-Based Access Control (PBAC)**
  - RBAC can be seen as a special case of ABAC: **role as an attribute!**
  - A real-world example: UK law and regulations on alcohol control
    - Those under 18 are not allowed to the following: “drinking alcohol in public”; “for someone to sell you alcohol”; “to buy or try to buy alcohol”; “for an adult to buy or try to buy alcohol for you”
    - “You’re 16 or 17 and accompanied by an adult, you can drink (but not buy) beer, wine or cider with a meal.”
    - “If you’re 16 or under, you may be able to go to a pub (or premises primarily used to sell alcohol) if you’re accompanied by an adult. However, this isn’t always the case. ...”
    - “You can serve alcohol in a restaurant if you’re 16 or 17 ...”
    - “It’s illegal to give alcohol to children under 5.”
- UK government (Gov.uk), “[Alcohol and young people](#),” accessed on 7 December 2020





- **Attribute-based Access Control (ABAC)**
  - Entities have **attributes**. Access control rules are mapped to attributes.
  - A second example: Attribute-based encryption (ABE)
    - General idea: A set of attributes must be met to decrypt a ciphertext.
    - Ciphertext-policy attribute-based encryption (CP-ABE): Each user's private key is associated with a set of attributes; a ciphertext is decryptable only if an access control policy based on a defined sets of subsets of attributes are met.
      - Example: Alice's key has attributes (A, B) and Bob's key has an attribute C; a ciphertext has an access control policy of (A AND D) OR C  $\Rightarrow$  Alice cannot decrypt the ciphertext, but Bob can.
    - Key-policy attribute-based encryption (KP-ABE): Each user's private key is generated by embedding an access control policy; a ciphertext is generated w.r.t. a set of attributes.
      - Example: Alice's key has an access control policy "A OR C" and Bob's key has an attribute "C AND D"; a ciphertext is associated with (C, D)  $\Rightarrow$  Alice cannot decrypt the ciphertext, but Bob can.

# Outline

---

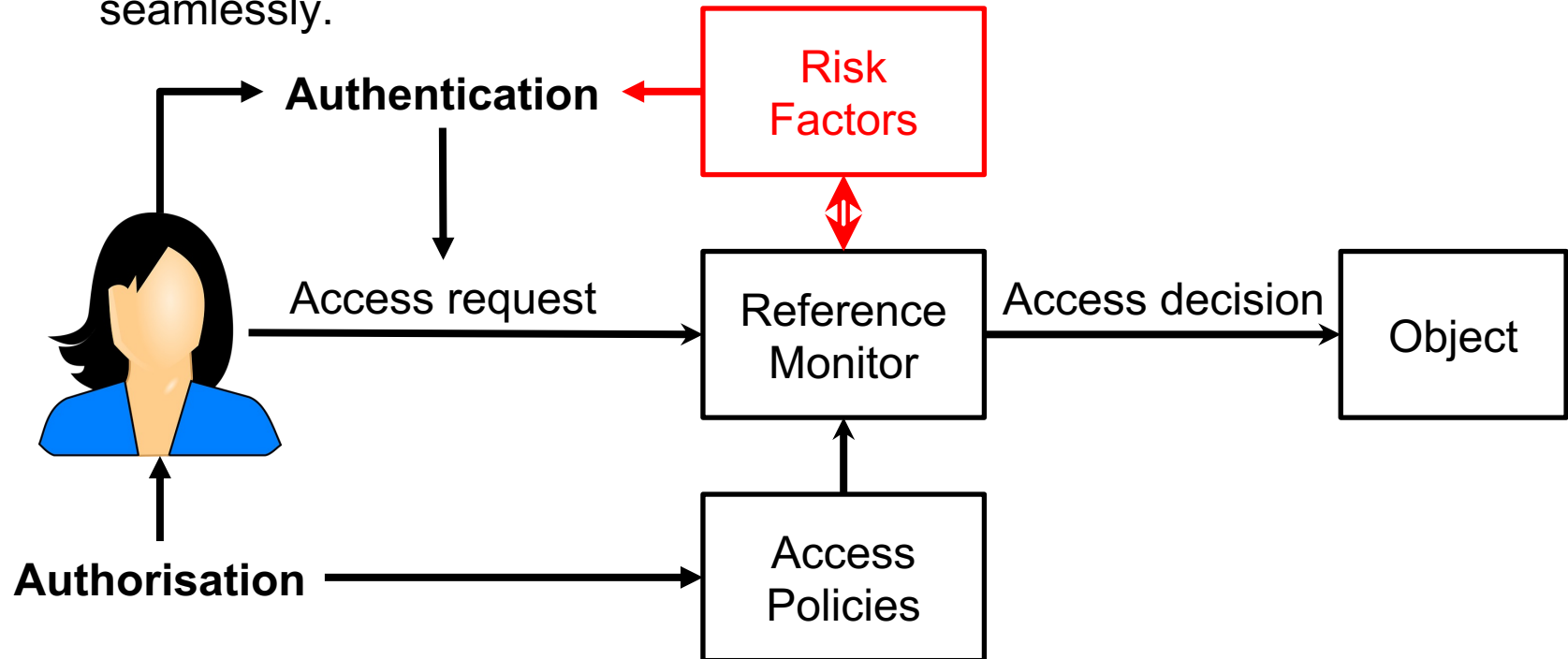
- Basic Concepts
- The Zoo of Access Control Models
  - ACL
  - DAC
  - RBAC
  - MAC
  - ABAC/PBAC
  - RAC
- Summary

# Access control model: RAC



## - Risk-based Access Control (RAC)

- Each access request is **risk** evaluated. Access control rules are mapped to different **risk** levels.  $\Rightarrow$  Can work with **risk-based authentication** seamlessly.



- Hany F. Atlam, Muhammad Ajmal Azad, Madini O. Alassafi, Abdulrahman A. Alshdadi and Ahmed Alenezi, "[Risk-Based Access Control Model: A Systematic Literature Review](#)," *Future Internet*, 12(6):103, 2020, MDPI

# Outline

---

- Basic Concepts
- The Zoo of Access Control Models
  - ACL
  - DAC
  - RBAC
  - MAC
  - ABAC/PBAC
  - RAC
- Summary

# The zoo of access control models



- [Discretionary Access Control](#) (DAC)
  - The **owner** of an object defines the access control policies (based on individual/group identities), based on his/her **discretion**.
- [Mandatory Access Control](#) (MAC)
  - A **central body** sets **mandatory** rules (constraints / restrictions) regarding who can access what (security/sensitivity labels on objects and entities).
- [Role-Based Access Control](#) (RBAC)
  - Entities have **roles**. Access control rules are mapped to roles.
- [Attribute-Based Access Control](#) (ABAC)
  - Entities have **attributes**. Access control rules are mapped to attributes.
- [Risk-based Access Control](#) (RAC)
  - Each access request is **risk** evaluated. Access control rules are mapped to different **risk** levels.  $\Rightarrow$  Can work seamlessly with **risk-based authentication**.
- ..., **hybrid models** (mixing two or more models)