Access Control Fundamentals

Giovanni Russello

g.russello@auckland.ac.nz

http://www.cs.auckland.ac.nz/compsci725s2c/
Access Control

- “The prevention of unauthorized use of a resource, including the prevention of use of a resource in an unauthorized manner”

- central element of computer security

- assume have users
  - authenticate to system
  - assigned access rights to certain resources on system
Access Control Requirements

- reliable input
- least privilege
- separation of duty
- fine and coarse specifications
- open and closed policies
- policy combinations, conflict resolution
- administrative policies
Access Control Elements

- **subject** - entity that can access objects
  - a process representing user/application
- **object** - access controlled resource
  - e.g. files, directories, records, programs etc
- **access right** - way in which subject accesses an object
  - e.g. read, write, execute, delete, create, search
Access Control Models

- Discretionary AC (DAC)
- Mandatory AC (MAC)
- Role-based AC (RBAC)
- Usage Control (UCON)
- Policy-based Access Control
Discretionary Access Control

- Subjects are able to assign on the objects they control access rights to other subjects
- Model used in operating systems and DB management systems
- often provided using an access matrix
## Access Control Matrix

<table>
<thead>
<tr>
<th></th>
<th>File1</th>
<th>File2</th>
<th>File3</th>
<th>File4</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>User A</strong></td>
<td>Own Read Write</td>
<td>Own Read Write</td>
<td>Own Read Write</td>
<td>Read Write</td>
</tr>
<tr>
<td><strong>User B</strong></td>
<td>Read</td>
<td>Own Read Write</td>
<td>Write</td>
<td>Read</td>
</tr>
<tr>
<td><strong>User C</strong></td>
<td>Read Write</td>
<td>Read</td>
<td>Own Read Write</td>
<td>Own Read Write</td>
</tr>
</tbody>
</table>
### Access Control List

<table>
<thead>
<tr>
<th></th>
<th>User A</th>
<th>User B</th>
<th>User C</th>
</tr>
</thead>
<tbody>
<tr>
<td>File1</td>
<td>Own</td>
<td>Read</td>
<td>Read</td>
</tr>
<tr>
<td></td>
<td>Own</td>
<td>Read</td>
<td>Read</td>
</tr>
<tr>
<td></td>
<td>Read</td>
<td>Own</td>
<td>Read</td>
</tr>
<tr>
<td>File2</td>
<td>Own</td>
<td>Read</td>
<td>Read</td>
</tr>
<tr>
<td></td>
<td>Read</td>
<td>Own</td>
<td>Read</td>
</tr>
<tr>
<td>File3</td>
<td>Own</td>
<td>Write</td>
<td>Own</td>
</tr>
<tr>
<td></td>
<td>Read</td>
<td>Write</td>
<td>Read</td>
</tr>
<tr>
<td>File4</td>
<td>Write</td>
<td>Read</td>
<td>Own</td>
</tr>
</tbody>
</table>

**User A**
- Can own, read, and write to File1.
- Can read and write to File2.
- Can read and write to File3.
- Can read and write to File4.

**User B**
- Can read and write to File1.
- Can read and write to File2.
- Can read and write to File3.
- Can read and write to File4.

**User C**
- Can read and write to File1.
- Can read and write to File2.
- Can read and write to File3.
- Can read and write to File4.
## Capability List

<table>
<thead>
<tr>
<th></th>
<th>File1</th>
<th>File2</th>
<th>File3</th>
<th>File4</th>
</tr>
</thead>
<tbody>
<tr>
<td>User A</td>
<td>Own Read Write</td>
<td>Own Read Write</td>
<td>Own Read Write</td>
<td></td>
</tr>
<tr>
<td>User B</td>
<td>Read</td>
<td>Own Read Write</td>
<td>Write</td>
<td>Read</td>
</tr>
<tr>
<td>User C</td>
<td>Read Write</td>
<td>Read</td>
<td>Own Read Write</td>
<td>Own Read Write</td>
</tr>
</tbody>
</table>

Access Matrix Details

S₁, read, F₂

S₂, write, P₂

S₁, grant, S₂, read, F₂
Entities cannot enable other entities to access their resources

It enforces a lattice between labels assigned to subjects and object
- security labels: how sensitive or critical a system resource is
- security clearances: which entities are eligible to access certain resources
The main goal is to control the confidentiality of information.
MAC Confidentiality Rules

Simple Security Property: No Read-Up

- Colonel
- Major
- Sergeant
- Private
- Top Secret
- Secret
- Confidential
- Unclassified

Read access:
- Colonel can read Top Secret
- Major can read Secret
- Sergeant cannot read Confidential
- Private cannot read Unclassified
MAC Confidentiality Rules

*(Star)property: No Write-Down

Colonel \rightarrow Top Secret

Major \rightarrow \not\rightarrow Secret

Sergeant \rightarrow Confidential

Private \rightarrow Unclassified
MAC Confidentiality Rules

Strong *(Star)-property: No Write-Down & No Write-up

- Colonel
- Major
- Sergeant
- Private

- Top Secret
- Secret
- Confidential
- Unclassified

write
The main goal is to control the integrity of information.
MAC Integrity Rules

Simple Integrity Axiom: No Read Down

Manager → read → Strategic
Project Leader → Sensitive
Engineer → Confidential
Jr. Engineer → Public
MAC Integrity Rules

*(Star)-Integrity Axiom: No Write Up

Manager → write → Strategic
Project Leader → write → Sensitive
Engineer → write → Confidential
Jr. Engineer → write → Public

No write up for Confidential and Public.
Where is MAC used

- BLP: Implemented the multi-level security policy for US Department of Defense
- BIBA: Implemented in the FreeBSD MAC policy
- A combined versions of BLP and BIBA is used in Android!
Enterprises organise employees in different roles

RBAC maps roles to access rights

Access rights are assigned to roles

After Subjects are authenticated they are assigned to roles
A simple example

Lecturer

Read/Write

Course Material

Read

Student
<table>
<thead>
<tr>
<th></th>
<th>Lecturer</th>
<th>S Lecturer</th>
<th>Ass Prof</th>
<th>Prof</th>
</tr>
</thead>
<tbody>
<tr>
<td>Giovanni</td>
<td>X</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Ulrich</td>
<td></td>
<td></td>
<td>X</td>
<td></td>
</tr>
<tr>
<td>Clark</td>
<td></td>
<td></td>
<td></td>
<td>X</td>
</tr>
</tbody>
</table>
# Role to Access Rights

## Lecturer
- **File1**: Own Read Write
- **File2**: Read
- **File3**: Own Read Write
- **File4**: Read

## S Lecturer
- **File1**: Read
- **File2**: Own Read Write
- **File3**: Write
- **File4**: Read

## Professor
- **File1**: Read Write
- **File2**: Read
- **File3**: Own Read Write
- **File4**: Own Read Write
Extensions to the Model

- A user can be in more than one role
  - Gill Dobbie is both Prof. and HoD

- Roles can be organised in Hierarchies
  - Prof>Ass Prof>Sen Lect>Lect
  - Top Roles inherited access rights of Lower Roles

- Constraints to enforce enterprise-specific requirements
RBAC Constraints

- Separation of Duties (SoD)
  - Protecting the organisation from frauds
- Chinese Wall CW)
  - Conflict of interests between different domains
SoD Details

SoD are used when an activity involves more than one roles:

**Purchase order** needs to be **prepared** by a **clerk** and then **authorised** by a **manager**

To avoid a fraud, the user that prepares the order should not be the same that authorises it.
In Static SoD, the same subject cannot be member of two mutually exclusive roles
- clerk and manager are mutually exclusive

Too restrictive: the user might get assigned to both roles as long as she is not working on the same order!
Dynamic SoD

- In Dynamic SoD, the same subject can be member of two mutually exclusive roles
- **However**, it requires extra checks that need to be done at runtime to avoid undesired behaviour
- Simple DSoD, Object DSoD, Operational DSoD, History DSoD
Simple DSoD

- Users cannot be active in mutually exclusive roles at the same time
- For instance, a user can be assigned to both clerk and manager roles as long as she is not active on both at the same time
Object DSoD

- Users can be active in mutually exclusive roles at the same time as long as she is not operating on the same object instance for the entire business process.

- For instance, a user can act in either clerk or manager role for a purchase order.

- Let's say that there is another operation: sending the order to depot. The user cannot execute this action even if it is not in conflict.
Operational DSoD

- Users can be active in mutually exclusive roles at the same time but cannot perform all the operations of business process.
- For instance, a user can activate both clerk and manager roles but cannot execute both the prepare and authorise operations (even for different objects!!)
History DSoD

- Users can be active in mutually exclusive roles at the same time as long as she is not authorised to execute all the operation for the same object instance.
- For instance, a user can be activate in both clerk role for a purchase order and manager role for another order.
Chinese Wall

- It applies to accesses in multiple domains with conflict of interest
- For instance, a consultant company offering services to both Microsoft and Apple. CW makes sure that an employee of the company will not get access to documents of both companies
Resource