**Integrity Models**

Integrity Goals:
- Data consistency (validity of data)
- Preventing unauthorised users from making modifications
- Preventing authorised users from making improper modifications

**Biba Integrity Model (1977)**

First Integrity Model, dual to Bell LaPadula

Model elements:
- S: set of subjects
- O: set of objects
- I: set of integrity levels (il) with a partial ordering

Access modes: g (observe), m (modify), i (invoke)

**Biba Model – “no read down”**

Security properties:

Simple Integrity-Property (“no read down”):
\[ \forall s \in S, o \in O: s \circ o \Rightarrow il(s) \leq il(o) \]

**Biba Model – “no write up”**

Integrity *-Property (“no write up”):
\[ \forall s \in S, o \in O: s \hspace{1mm} m \hspace{1mm} o \Rightarrow il(o) \leq il(s) \]

**Biba Model - Invocation property**

Invocation Property:
\[ \forall s_1, s_2 \in S: s_1 \downarrow s_2 \Rightarrow il(s_2) \leq il(s_1) \]

**Biba Model - Discussion**

- Does not address data consistency
- Only prevention of modifications by unauthorized users
- Authorized users can still make improper modifications
- Problem to assign appropriate integrity levels
- Only implemented in few systems
**Clark Wilson Integrity Model (1988):**

**Policy is based on two key concepts:**
- Well-formed transactions
- Separation of duties

**Clark Wilson - Model Description**

**State variables:**
- CDIs: Constrained Data Items
- IVPs: Integrity Verification Procedures
- TPs: Transformation Procedures
  - (transaction = transformation procedure + permitted CDIs)
- UDIs: Unconstrained Data Items (input data)

**Clark Wilson - Rules**

**Two types of rules:**
- Certification Rules (C) - performed by the security officer
- Enforcement Rules (E) - done by the system

**Clark Wilson - Certification Rules**

- C1: IVPs certify CDI are valid
- C2: TPs preserve valid state
- C3: Suitable separation of duties
- C4: TPs write log
- C5: TPs validate UDI

**Clark Wilson - Enforcement Rules**

- E1: CDIs changed only by authorized TPs
- E2: Users are authorized for TPs
- E3: Users are authenticated
- E4: Authorisation lists (Access Triples) changed only by security officer
Access triples can:
- prevent modifications by unauthorized users
- enforce separation of duties
- Separation of Duties Principle can prevent authorized users from making improper modifications
- IVPs and TPs can help guarantee consistency

Developed at NIST/NSA since early nineties
- American National Standard - ANSI INCITS 359-2004
- Implemented in SUN Solaris, SeLinux, Oracle, Sybase,...

See also: http://csrc.nist.gov/rbac/

Security administrator assigns:
- Membership of users to roles: 
  \[ \text{user} \rightarrow \text{roles} \]
- Privileges (operations) to a role:
  \[ \text{role} \rightarrow \text{set of operations} \]

A user in the role “doctor” may perform the following transactions:
- enter_diagnosis
- prescribe_medication

A subject S may access an object O with operation \( \text{op} \) if:
- S performs a role \( \text{r} \)
- S is authorised to perform \( \text{r} \)
- \( \text{r} \) is allowed to perform \( \text{op} \)
- \( \text{op} \) is allowed to access \( \text{O} \)

Allocation of operations to a role remains relatively constant or changes slowly over time.
- The security administrator’s task consists simply of granting and revoking memberships to the set of specified roles.
- Roles can be composed of roles with an inheritance of privileges
RBAC – Role Hierarchies

Example: “Doctor” role inherits privileges of “Nurse” role

- Prescription: prescribe_med
- Dr. Sara John
- Patient Record: enter_diagnosis
- Dr. Peter Smith
- Nurse: enter_blood_values
- Mary Jo
- Su Chen

RBAC – Cost savings

<table>
<thead>
<tr>
<th>TASK</th>
<th>RBAC</th>
<th>NON-RBAC</th>
<th>DIFFERENCE</th>
</tr>
</thead>
<tbody>
<tr>
<td>Add new user privilege</td>
<td>6.14</td>
<td>11.30</td>
<td>5.16</td>
</tr>
<tr>
<td>Change existing privileges</td>
<td>9.29</td>
<td>10.24</td>
<td>0.95</td>
</tr>
<tr>
<td>Establish new privileges</td>
<td>6.86</td>
<td>9.26</td>
<td>2.40</td>
</tr>
<tr>
<td>Terminate privileges</td>
<td>0.81</td>
<td>1.52</td>
<td>0.71</td>
</tr>
</tbody>
</table>

Annual cost savings: $6,924 a year for 1,000 employees

RBAC – Further features (1)

- Static Separation of Duties
  - by defining mutually exclusive roles
  - e.g. In a bank the roles “teller”, “auditor” cannot be authorised for the same user

- Dynamic Separation of Duties
  - By defining mutually exclusive activation of roles
  - e.g. Roles “Payment Initiator”, “Payment Authoriser” can be performed by same user, but not at the same time

RBAC – Further features (2)

- Role Cardinality
  - by defining a capacity for a role
  - e.g. only one user is “Manager”

RBAC – Features in Commercial Database Management Systems

<table>
<thead>
<tr>
<th>Feature</th>
<th>Informix</th>
<th>Sybase</th>
<th>Oracle</th>
</tr>
</thead>
<tbody>
<tr>
<td>Ability to role grantee to grant that role to other users</td>
<td>YES</td>
<td>NO</td>
<td>YES</td>
</tr>
<tr>
<td>Multiple active roles for a user session</td>
<td>NO</td>
<td>YES</td>
<td>YES</td>
</tr>
<tr>
<td>Specify a default active role set for a user session</td>
<td>NO</td>
<td>YES</td>
<td>YES</td>
</tr>
<tr>
<td>Build a role hierarchy</td>
<td>YES</td>
<td>YES</td>
<td>YES</td>
</tr>
<tr>
<td>Specify static separation of duty constraints on roles</td>
<td>NO</td>
<td>YES</td>
<td>NO</td>
</tr>
<tr>
<td>Specify dynamic separation of duty constraints on roles</td>
<td>(YES)</td>
<td>YES</td>
<td>NO</td>
</tr>
<tr>
<td>Specify maximum or minimum cardinality for role membership</td>
<td>NO</td>
<td>NO</td>
<td>NO</td>
</tr>
<tr>
<td>Grant DBMS System Privileges to a role</td>
<td>NO</td>
<td>YES</td>
<td>YES</td>
</tr>
<tr>
<td>GRANT DBMS Object Privileges to a role</td>
<td>YES</td>
<td>YES</td>
<td>YES</td>
</tr>
</tbody>
</table>

Exercise 2

Consider the following commercial security requirements:

1. Users of applications (e.g. accounting) will use application programs and databases; they will not write their own programs to operate on the databases

2. Application program developer will do their development and testing in a test environment, and have no access to the application (source or object) programs or databases. They have access to programming tools and test data.

→ Discuss if/how this requirement can be enforced by Bell LaPadula, Clark Wilson and RBAC