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

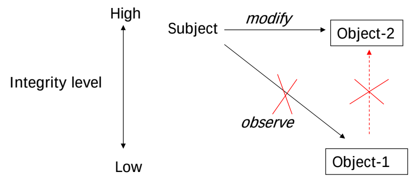
Model elements:
S: set of subjects
O: set of objects
I: set of integrity levels (il) with a partial ordering

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

Biba Model – ”no read down”

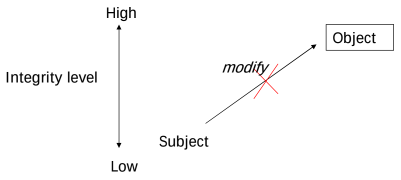
Security properties:

Simple Integrity-Property (“no read down”):
$\forall s \in S, o \in O: \ s\ \underline{o}\ o \ \Rightarrow \ il(s) \leq il(o)$



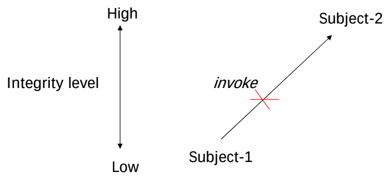
Biba Model – ”no write up”

Integrity *-Property (”no write up”):
$\forall s \in S, o \in O: \ s\ \underline{m}\ o \ \Rightarrow \ il(o) \leq il(s)$



Biba Model – invocation property

Invocation Property:
$\forall s_1, s_2 \in S: \ s_1\ \underline{i}\ s_2 \ \Rightarrow \ il(s_2) \leq il(s_1)$



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:

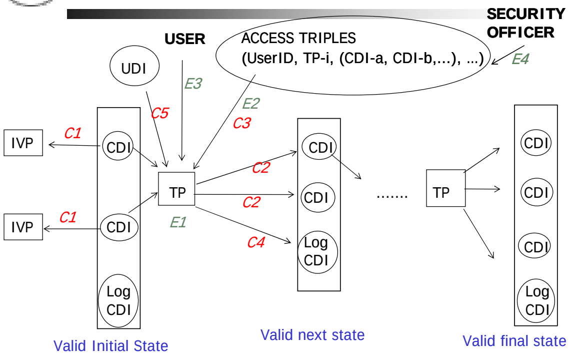
CDIs: Constrained Data Items
IVPs: Integrity Verification Procedures
TPs: Transformation Procedures
(transaction = transformation procedure + permitted CDIs)
UDIs: Unconstrained Data Items (input data)

Clark Wilson - Rules

Two types of rules:

- Certification Rules (C) - performed by the security officer
- Enforcement Rules (E) - done by the system

Clark Wilson - Overview



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

Clark Wilson - Discussion

- 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 to guarantee consistency

Role-Based Access Control (RBAC) - Introduction

- 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/

Role-Based Access Control (RBAC) - Approach

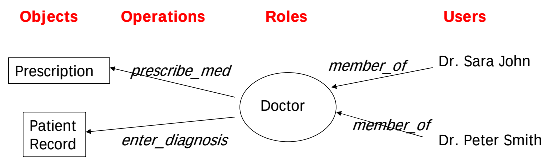
Security administrator assigns:

- Membership of users to roles:
  user --> roles
- Privileges (operations) to a role:
  role --> set of operations

RBAC - Example

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

- enter_diagnosis
- prescribe_medication



RBAC – Access rules

A subject S may access an object O with operation op if

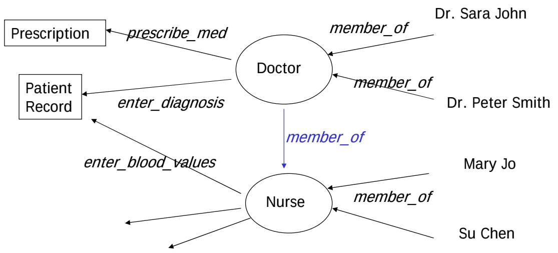
- S performs a role r
- S is authorised to perform r
- r is allowed to perform op
- op is allowed to access O

RBAC – Reduced Administrative Overhead

- 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



RBAC – Cost savings

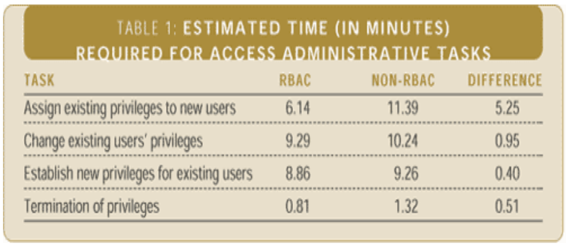


TABLE 1: ESTIMATED TIME (IN MINUTES) REQUIRED FOR ACCESS ADMINISTRATIVE TASKS

TASK	RBAC	NON-RBAC	DIFFERENCE
Assign existing privileges to new users	6.14	11.39	5.25
Change existing users' privileges	9.29	10.24	0.95
Establish new privileges for existing users	8.86	9.26	0.40
Termination of privileges	0.81	1.32	0.51

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 – Futher 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

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

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