

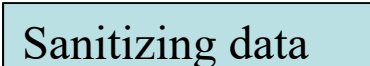
# Chapter 6: Integrity Policies

---

- Requirements
  - Very different than confidentiality policies
- Biba's model
- Clark-Wilson model

# Requirements of Policies for Commercial Applications [Lipner 1982]

---

1. Users will not write their own programs, but will use existing production programs and databases.
2. Programmers will develop and test programs on a non-production system; if they need access to actual data, they will be given production data via a special process, but will use it on their development system. 
3. A special process must be followed to install a program from the development system onto the production system.
4. The special process in requirement 3 must be controlled and audited.
5. The managers and auditors must have access to both the system state and the system logs that are generated.

# Requirements and Principles of Operation

---

- *Separation of duty.* If two or more steps are required to perform a critical function, at least two different subjects should perform them. Moving an application from the development system to the production system is an example of critical function
- *Separation of function.* Different functions are executed on different sets of data. For example, developers do not develop new programs in the production environment. Also they do not process production data in the development environment. If they need data, depending on the sensitivity of data, sanitized versions of these data may be given to them
- *Auditing.* It is the process of analyzing systems to determine what actions took place and who performed them

# Biba Integrity Model

---

- Set of subjects  $S$ , objects  $O$ , integrity levels  $I$ , relation  $\leq \subseteq I \times I$  holding when second *dominates* first
- $\min: I \times I \rightarrow I$  returns lesser of integrity levels
- $i: S \cup O \rightarrow I$  gives integrity level of entity
- $\underline{r}: S \times O$  means  $s \in S$  can read  $o \in O$
- $\underline{w}$ ,  $\underline{x}$  defined similarly

# Intuition for Integrity Levels

---

- The higher the level, the more confidence
  - That a program will execute correctly
  - That data is accurate and/or reliable
- Positive relationship between integrity and trustworthiness

# Biba's Model

---

- Similar to Bell-LaPadula model
  1.  $s \in S$  can read  $o \in O$  iff  $i(s) \leq i(o)$ , no read down
  2.  $s \in S$  can write to  $o \in O$  iff  $i(o) \leq i(s)$ , no write up
  3.  $s_1 \in S$  can execute  $s_2 \in S$  iff  $i(s_2) \leq i(s_1)$
- Add compartments and discretionary controls to get full dual of Bell-LaPadula model
- Information flow result holds

# Example

---

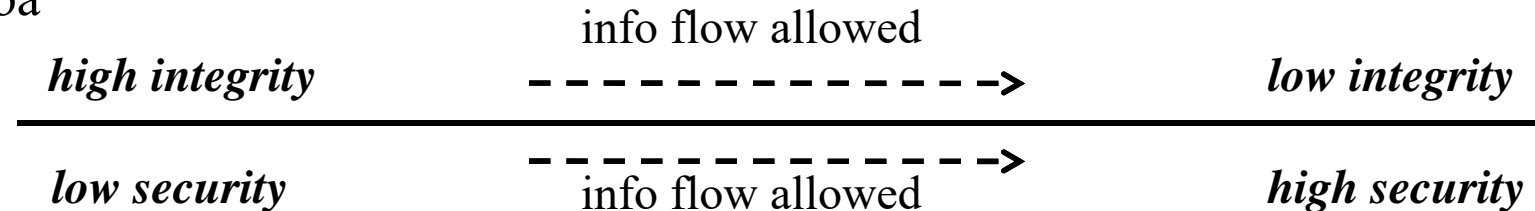
- Peter, with label (Expert, {Chemistry})
- Paul, with label (Freshman, {Chemistry})
- Tutorial has label (Professional, {Chemistry})
- Freshman < Professional < Expert
- Can Peter write tutorial?
  - Yes, write down
- Can Paul write tutorial?
  - No, no write up

# Biba vs. Bell-LaPadula

---

- Important: *integrity levels are **not** security levels*
- Biba: information flows towards *low integrity levels*
- Bell-LaPadula: information flows towards *high security levels*

Biba



Bell-LaPadula



# Clark-Wilson Integrity Model

---

- Integrity defined by a set of constraints
  - Data in a *consistent* or valid state when it satisfies these
- Example: Bank
  - $D$  today's deposits,  $W$  withdrawals,  $YB$  yesterday's balance,  $TB$  today's balance
  - Integrity constraint:  $D + YB - W = TB$
- *Well-formed transactions*: a series of operations that move system from one consistent state to another consistent state. E.g., transfer of money
- Issue: who examines and certifies that transactions are done correctly? E.g., fraud.

# Entities

---

- CDIs: constrained data items
  - Data subject to integrity controls
- UDIs: unconstrained data items
  - Data not subject to integrity controls
- IVPs: integrity verification procedures
  - Procedures that test the CDIs conform to the integrity constraints
- TPs: transformation procedures
  - Procedures that take the system from one valid state to another

# Bank Account Example

---

- The balances in the accounts are CDIs
- Gifts selected by customers when their accounts are opened are UDIs
- Checking that the accounts are balanced is an IVP
- Depositing money, withdrawing money, and transferring money between accounts are TPs
- To ensure that the accounts are managed correctly, a bank examiner must certify that the bank is using proper procedures
- Also, those procedures may apply only to deposit and checking accounts (not other types of accounts)

# Certification Rules 1 and 2

---

- CR1 When any IVP is run, it must ensure all CDIs are in a valid state
- CR2 For some associated set of CDIs, a TP must transform those CDIs in a valid state into a (possibly different) valid state
- Defines relation *certified* that associates a set of CDIs with a particular TP
  - Example: TP balance, CDIs accounts, in bank example

# Enforcement Rules 1 and 2

---

- ER1 The system must maintain the certified relations and must ensure that only TPs certified to run on a CDI manipulate that CDI.
- ER2 The system must associate a user with each TP and set of CDIs. The TP may access those CDIs on behalf of the associated user.
- E.g., a janitor is not allowed to balance accounts
  - System must maintain, enforce *certified* relation
  - System must also restrict access based on user ID (*allowed* relation, next slide)

# Users and Rules

---

- CR3 The allowed relations (*user*, TP, {CDI *set*}) must meet the requirements imposed by the principle of *separation of duty*.
- ER3 The system must authenticate each user attempting to execute a TP
- Type of authentication undefined, and depends on the instantiation
  - Authentication *not* required before use of the system, but *is* required before manipulation of CDIs (requires using TPs)

# Logging

---

- CR4 All TPs must append enough information to reconstruct the operation to an append-only CDI.
- This CDI is the log
  - Auditor needs to be able to determine what happened during reviews of transactions

# Handling Untrusted Input

---

CR5 Any TP that takes as input a UDI may perform only valid transformations, or no transformations, for all possible values of the UDI. The transformation either rejects the UDI or transforms it into a CDI.

- In bank, numbers entered at keyboard are UDIs, so cannot be input to TPs. TPs must validate numbers (to make them a CDI) before using them; if validation fails, TP rejects UDI



# Separation of Duty In Model

---

- ER4 Only the certifier of a TP may change the list of entities (CDIs) associated with that TP. No certifier of a TP, or of an entity associated with that TP, may ever have execute permission with respect to that entity.
- Enforces separation of duty with respect to certified and allowed relations

# List of Rules

---

- CR1      When any IVP is run, it must ensure all CDIs are in a valid state
- CR2      For some associated set of CDIs, a TP must transform those CDIs in a valid state into a (possibly different) valid state
- CR3      The allowed relations (*user*, TP, {CDI set}) must meet the requirements imposed by the principle of *separation of duty*.
- CR4      All TPs must append enough information to reconstruct the operation to an append-only CDI.
- CR5      Any TP that takes as input a UDI may perform only valid transformations, or no transformations, for all possible values of the UDI. The transformation either rejects the UDI or transforms it into a CDI.
- ER1      The system must maintain the certified relations and must ensure that only TPs certified to run on a CDI manipulate that CDI.
- ER2      The system must associate a user with each TP and set of CDIs. The TP may access those CDIs on behalf of the associated user.
- ER3      The system must authenticate each user attempting to execute a TP
- ER4      Only the certifier of a TP may change the list of entities (CDIs) associated with that TP. No certifier of a TP, or of an entity associated with that TP, may ever have execute permission with respect to that entity.

# Comparison With Requirements

---

1. Users can't certify TPs, only “trusted personnel” can
  - ❑ Users will not write their own programs, but will use existing production programs and databases
2. Procedural, so model doesn't directly cover it; but special process corresponds to using TP
  - No technical controls can prevent programmer from developing program on production system; usual control is to delete software tools
  - ❑ Programmers will develop and test programs on a non-production system; if they need access to actual data, they will be given production data via a special process, but will use it on their development system

# Comparison With Requirements

---

3. TP does the installation, trusted personnel do certification

- ☐ A special process must be followed to install a program from the development system onto the production system

4. CR4 provides logging; ER3 authenticates trusted personnel doing installation; CR5, ER4 control installation procedure

- New program UDI before certification, CDI (and TP) after
- ☐ The special process in requirement 3 must be controlled and audited.

# Comparison With Requirements

---

5. Log is CDI, so appropriate TP can provide managers, auditors access
  - Access to state handled similarly
  - The managers and auditors must have access to both the system state and the system logs that are generated

# Comparison to Biba

---

- Biba
  - No notion of certification rules; *trusted subjects* ensure actions obey rules
  - Untrusted data examined before being made trusted
- Clark-Wilson
  - Explicit requirements that *actions* must meet
  - Trusted entity must certify *method* to upgrade untrusted data (and not certify the data itself)

# Key Points

---

- Integrity policies deal with trust
- Biba based on multilevel integrity
- Clark-Wilson focuses on separation of duty and transactions