Integrity Policies

Integrity policies address commercial requirements and differ significantly from government requirements, particularly with their emphasis on data integrity.

Commercial Environment

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 nonproduction 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.

Principles of Operation

1. Separation of Duty

   If two or more steps are required to perform a critical function, at least two different people should perform those steps.

2. Separation of Function

   The production and development environments are two separate systems. One does not do system development on production systems, and one does not do production work on development systems.

3. Auditing

   The process of analyzing systems to determine what actions took place and who performed those actions. Systems emphasize recovery and accountability.
Biba Integrity Model

The Biba security model consists of:

1. a set of subjects $S$
2. a set of objects $O$
3. a set of integrity levels $I$ where the levels are ordered
   The higher the integrity level, the more confidence place in the system
   Data at a higher level is more accurate and/or reliable
   Systems/Data at a higher level is more trustworthy.

Note: Security labels (Confidential Policies) limit the flow of information
Integrity labels (Integrity Policies) inhibit the modification of information.

The Biba Model is a dual of the Bell-La Padula model in that

1. $s \in S$ can read $o \in O$ iff $i(s) \leq i(o)$
2. $s \in S$ can write to $o \in O$ iff $i(o) \leq i(s)$
3. $s_1 \in S$ can execute $s_2 \in S$ iff $i(s_2) \leq i(s_1)$

Rule 1 says that a lower level subject can read higher level objects. This says that one can read up in integrity levels. This makes sense because if the object has a higher integrity level, then it also has a lower level of integrity or that the integrity level of the object has been lower to the integrity level of the reading subject. Thus a subject can read up in integrity levels.

Rule 2 prevents writing from a lower integrity level to a higher integrity level. This prevents a subject from writing to a more highly trusted object. Intuitively, if a subject were to alter a more trusted object, it could implant incorrect or false data. The trustworthiness of the object would drop to that of the subject. Such writing is not allowed.

Rule 3 indicates that processes of higher integrity can execute processes of lower or equal integrity.

Note: By add discretionary controls, we have the full Bell-La Padula dual.

Clark-Wilson Integrity Model

The Clark-Wilson integrity model is radically different from other integrity models. It is commercially oriented and uses transactions as the basic operation. With commercial environments one is concerned with the integrity of the data and the operations on that data.

Def: Data is said to be in a consistent state (or consistent) if it satisfied given properties.

   Banking example of $YB + D - W = TB$

Def: A well-formed transaction is a series of operations that transition the system from one consistent state to another consistent state.

Who determines that a transaction is well-formed? Both the implementer and the certifier must verify that the transaction is well-formed. This applies the principle of separation of duty.
The Clark-Wilson Model identifies the data objects as either constrained data items (CDIs) or unconstrained data items (UDIs). A CDI is subject to integrity controls and a UDI is not subject to integrity controls. Check balances are examples of CDIs and lollipops in the bank lobby are an example of a UDI. The model also has a set of integrity constraints that constrain values of the CDIs.

There are two sets of procedures in the model. Integrity verification procedures (IVPs) verify that the CDIs conform to the integrity constraints at the time that the procedures execute. If the CDIs do conform to the integrity constraints, then the system is said to be in a valid state. The other set of procedures is called transformation procedures (TPs). Transformation procedures change the state of the data in the system from one valid state to another valid state. TPs implement well-formed transactions.

**Certification Rule 1:** When any IVP is executed, it must ensure that all CDIs are in a valid state.

**Certification Rule 2:** For some associated set of CDIs, a TP must transform those CDIs in a valid state into a possibly different valid state.

Certification rule 2 identifies a certified relation that associates a set of CDIs with a particular TP. The system must prevent TPs from operating on CDIs that have not been certified for that TP. Hence, we have the following enforcement rule:

**Enforcement Rule 1:** The system must maintain the certified relations and must ensure that only TPs certified to run on a CDI manipulate that CDI.

Not only must the TP be associated with a set of CDIs, that TP must also be associated with a set of users who can activate that TP. This leads to enforcement rule 2:

**Enforcement Rule 2:** 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. If the user is not associated with a particular TP and CDI, then the TP cannot access that CDI on behalf of the user.

This leads us to the set of triples \((\text{user}, \text{TP}, \{\text{CDI set}\})\) that captures the relationship between the users, TPs, and CDIs. This relation is called allowed A. The certification rule for this relation is Certification rule 3.

**Certification Rule 3:** The allowed relations must meet the requirements imposed by the principle of separation of duty.

This leads to the enforcement rule:

**Enforcement Rule 3:** The system must authenticate each user attempting to execute a TP.

**Note:** Authentication is only required when a user tries to execute a TP, not when the user logs into the system.

Transaction logs are maintained so that auditors can review system transactions. The Clark-Wilson Model treats transaction logs as a CDI. Every TP appends to the transaction log, and no TP can overwrite the log. This leads to
**Certification Rule 4:** All TPs must append enough information to reconstruct the operation to an append-only CDI.

There are situations when a subject will take UDIs and verify that they are correct. This takes a UDI and turns it into a CDI. The certification rule is

**Certification Rule 5:** 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.

One more enforcement rule is needed to maintain the separation of duty principle.

**Enforcement Rule 4:** Only the certifier of a TP may change the list of entities 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.

Two new ideas of integrity models:

1. Captures the way that most commercial banks do business.
   separation of duty
   no multi-level data classification scheme

2. Notion of certification is distinct from the notion of enforcement.
   Each has own set of rules
   Enforcement strong
   Certification weak