

Confidentiality and Integrity

Deepak Puthal

Email: Deepak.Puthal@uts.edu.au

41900 – Fundamentals of Security

Overview

- Confidentiality Policy
- Bell-LaPadula Model
 - Reading Information
 - Writing Information
- Basic Security Theorem
- Integrity Policies
- Biba's model
- Clark-Wilson Integrity Model
- Comparison study

Confidentiality Policy

- Goal: prevent the unauthorized disclosure of information
 - Deals with information flow
 - Integrity incidental
- Multi-level security models are best-known examples
 - Bell-LaPadula Model basis for many, or most, of these

Bell-LaPadula Model, Step 1

- Security levels arranged in linear ordering
 - Top Secret: highest
 - Secret
 - Confidential
 - Unclassified: lowest
- Levels consist of *security clearance* $L(s)$
 - Objects have *security classification* $L(o)$

Reading Information

- Information flows *up*, not *down*
 - “Reads up” disallowed, “reads down” allowed
- Simple Security Condition (Step 1)
 - Subject s can read object o iff $L(o) \leq L(s)$ and s has permission to read o
 - Note: combines mandatory control (relationship of security levels) and discretionary control (the required permission)
 - Sometimes called “no reads up” rule

Writing Information

- Information flows up, not down
 - “Writes up” allowed, “writes down” disallowed
- *-Property (Step 1)
 - Subject s can write object o iff $L(s) \leq L(o)$ and s has permission to write o
 - Note: combines mandatory control (relationship of security levels) and discretionary control (the required permission)
 - Sometimes called “no writes down” rule

Basic Security Theorem, Step 1

- If a system is initially in a secure state, and every transition of the system satisfies the simple security condition, step 1, and the *-property, step 1, then every state of the system is secure
 - Proof: induct on the number of transitions

Bell-LaPadula Model, Step 2

- Expand notion of security level to include categories
- Security level is (*clearance*, *category set*)
- Examples
 - (Top Secret, { NUC, EUR, ASI })
 - (Confidential, { EUR, ASI })
 - (Secret, { NUC, ASI })

Levels and Lattices

- $(A, C) \text{ dom } (A', C')$ iff $A' \leq A$ and $C' \subseteq C$
- Examples
 - $(\text{Top Secret}, \{\text{NUC}, \text{ASI}\}) \text{ dom } (\text{Secret}, \{\text{NUC}\})$
 - $(\text{Secret}, \{\text{NUC}, \text{EUR}\}) \text{ dom } (\text{Confidential}, \{\text{NUC}, \text{EUR}\})$
 - $(\text{Top Secret}, \{\text{NUC}\}) \not\text{dom } (\text{Confidential}, \{\text{EUR}\})$
- Let C be set of classifications, K set of categories. Set of security levels $L = C \times K$, dom form lattice
 - $\text{lub}(L) = (\max(A), C)$
 - $\text{glb}(L) = (\min(A), \emptyset)$

Levels and Ordering

- Security levels partially ordered
 - Any pair of security levels may (or may not) be related by *dom*
- “dominates” serves the role of “greater than” in step 1
 - “greater than” is a total ordering, though

Reading Information

- Information flows *up*, not *down*
 - “Reads up” disallowed, “reads down” allowed
- Simple Security Condition (Step 2)
 - Subject s can read object o iff $L(s) \text{ dom } L(o)$ and s has permission to read o
 - Note: combines mandatory control (relationship of security levels) and discretionary control (the required permission)
 - Sometimes called “no reads up” rule

Writing Information

- Information flows up, not down
 - “Writes up” allowed, “writes down” disallowed
- *-Property (Step 2)
 - Subject s can write object o iff $L(o) \text{ dom } L(s)$ and s has permission to write o
 - Note: combines mandatory control (relationship of security levels) and discretionary control (the required permission)
 - Sometimes called “no writes down” rule

Basic Security Theorem, Step 2

- If a system is initially in a secure state, and every transition of the system satisfies the simple security condition, step 2, and the *-property, step 2, then every state of the system is secure
 - Proof: induct on the number of transitions
 - In actual Basic Security Theorem, discretionary access control treated as third property, and simple security property and *-property phrased to eliminate discretionary part of the definitions — but simpler to express the way done here.

Problem

- Colonel has (Secret, {NUC, EUR}) clearance
- Major has (Secret, {EUR}) clearance
 - Major can talk to colonel (“write up” or “read down”)
 - Colonel cannot talk to major (“read up” or “write down”)
- Clearly absurd!

Solution

- Define maximum, current levels for subjects
 - $maxlevel(s) \text{ dom } curlevel(s)$
- Example
 - Treat Major as an object (Colonel is writing to him/her)
 - Colonel has $maxlevel$ (Secret, { NUC, EUR })
 - Colonel sets $curlevel$ to (Secret, { EUR })
 - Now $L(\text{Major}) \text{ dom } curlevel(\text{Colonel})$
 - Colonel can write to Major without violating “no writes down”
 - Does $L(s)$ mean $curlevel(s)$ or $maxlevel(s)$?
 - Formally, we need a more precise notation

Integrity Policies

Requirements of Policies

- Users will not write their own programs, but will use existing production programs and databases.
- 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.
- A special process must be followed to install a program from the development system onto the production system.
- The special process in requirement 3 must be controlled and audited.
- The managers and auditors must have access to both the system state and the system logs that are generated.

Biba Integrity Model

- Set of subjects S , objects O , integrity levels I , relation $\leq \subseteq I \times I$ holding when second dominates first
- $\text{min}: I \times I \rightarrow I$ 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
- Note relationship between integrity and trustworthiness
- Important point: *integrity levels are **not** security levels*

Biba's Model

- Similar to Bell-LaPadula model
 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 $i(s_2) \leq i(s_1)$
- Add compartments and discretionary controls to get full dual of Bell-LaPadula model
- Information flow result holds
 - Different proof, though
- Actually the “strict integrity model” of Biba’s set of models

LOCUS and Biba

- Goal: prevent untrusted software from altering data or other software
- Approach: make levels of trust explicit
 - *credibility rating* based on estimate of software's trustworthiness (0 untrusted, n highly trusted)
 - *trusted file systems* contain software with a single credibility level
 - Process has *risk level* or highest credibility level at which process can execute
 - Must use *run-untrusted* command to run software at lower credibility level

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: $D + YB - W$
- *Well-formed transaction* move system from one consistent state to another
- Issue: who examines, certifies transactions done correctly?

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: transaction procedures
 - Procedures that take the system from one valid state to another

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. The TP cannot access that CDI on behalf of a user not associated with that TP and CDI.
- System must maintain, enforce certified relation
 - System must also restrict access based on user ID (*allowed relation*)

Users and Rules

- CR3 The allowed relations 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 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

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

- Confidentiality models restrict flow of information
- Bell-LaPadula models multilevel security
 - Cornerstone of much work in computer security
- Integrity policies deal with trust
 - As trust is hard to quantify, these policies are hard to evaluate completely
 - Look for assumptions and trusted users to find possible weak points in their implementation
- Biba based on multilevel integrity
- Clark-Wilson focuses on separation of duty and transactions