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

- Subjects are cleared $L(s)$ to security level
  1. Unclassified: lowest
  2. Confidential
  3. Secret
  4. Top Secret: highest
- Objects are classified $L(o)$ to the same security level
### Example

<table>
<thead>
<tr>
<th>security level</th>
<th>subject</th>
<th>object</th>
</tr>
</thead>
<tbody>
<tr>
<td>Top Secret</td>
<td>Tamara</td>
<td>Personnel Files</td>
</tr>
<tr>
<td>Secret</td>
<td>Samuel</td>
<td>E-Mail Files</td>
</tr>
<tr>
<td>Confidential</td>
<td>Claire</td>
<td>Activity Logs</td>
</tr>
<tr>
<td>Unclassified</td>
<td>Ulaley</td>
<td>Telephone Lists</td>
</tr>
</tbody>
</table>

- Tamara can read all files
- Claire cannot read Personnel or E-Mail Files
- Ulaley can only read Telephone Lists

### BLP<sub>mac</sub> – More Formally

- Set of Subjects (s<sub>i</sub> ∈ S)
- Set of Objects (o<sub>j</sub> ∈ O)
- Set of Levels (l<sub>k</sub> ∈ L)
- An access request is a tuple of cross products (L×S, L×O)
- Access is granted iff L(s) ≥ L(o)

### Reading Information

- Information flows *up*, not *down*
  - “Reads up” disallowed,
  - “Reads down” allowed
**BLP\textsubscript{dac}**

- Need to know
- Loosely defined
- Permission
- Discretionary

**BLP - Reading Information**

- Simple Security Condition (Step 1)
  - Subject \( s \) can read object \( o \) iff:
    - \( L(o) \leq L(s) \) and
    - \( s \) has the need to read \( o \)
- Combines MAC (relationship of security levels) and DAC (the required permission)
- Sometimes called “no reads up” rule

**Writing Information**

- Information flows up, not down
  - “ Writes up” allowed
  - “ Writes down” disallowed
- \textasteriskcentered-Property (Step 1)
  - Subject \( s \) can write object \( o \) iff \( L(s) \leq L(o) \) and \( s \) has permission to write \( o \)
  - “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.

Bell-LaPadula Model, Step 2

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

Levels and Lattices

- \((A, C) \ dom (A', C')\) iff \(A' \leq A\) and \(C' \subseteq C\)
- Examples
  - (Top Secret, \{NUC, ASI\}) \ dom (Secret, \{NUC\})
  - (Secret,\{NUC, EUR\}) \ dom (Confidential,\{NUC, EUR\})
  - (Top Secret, \{NUC\}) \ dom (Confidential, \{EUR\})
- Let \(C\) be set of classifications, \(K\) set of categories. Set of security levels \(L = C \times K\), \textit{dom} form lattice
  - \textit{lub}(L) = (\textit{max}(A), C)\) (max clearance, all categories)
  - \textit{glb}(L) = (\textit{min}(A), \emptyset)\) (min clearance, no categories)
Levels and Lattices

- $(A, C) \text{ dom } (A', C')$ iff $A' \leq A$ and $C' \subseteq C$

Examples
- $(\text{Top Secret, \{NUC, ASI\}}) \text{ dom } (\text{Secret, \{NUC\}})$
- $(\text{Secret,\{NUC\},EUR}) \text{ dom } (\text{Confidential,\{NUC, EUR\}})$
- $(\text{Top Secret, \{NUC\}}) \neg \text{ dom } (\text{Confidential, \{EUR\}})$

Levels and Ordering

- Security levels partially ordered
  - Any pair of security levels may (or may not) be related by $\text{dom}$
- “dominates” serves the role of “greater than” in step 1

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 $o$
Writing Information

- Information flows up, not down
  - "Writes up" allowed, "writes down" disallowed

- *-Property (Step 2)
  - Subject \( s \) can write object \( o \) iff \( L(o) \) is in \( dom L(s) \) and \( s \) has permission to write \( o \)

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

Object Labels

- To implement file system MAC, each object must have MAC label
  1. Roots of file systems have explicit MAC labels
     - If mounted file system has no label, it gets label of mount point
  2. Object with implicit MAC label inherits label of parent
Formal Model Definitions

- $S$ subjects, $O$ objects, $P$ rights
  - Defined rights: $r$ read, $a$ write, $w$ read/write, $e$ empty (delete contents)
- $M$ set of possible access control matrices
- $C$ set of clearances/classifications, $K$ set of categories, $L=C\times K$ set of security levels
- $F = \{(f_s, f_o, f_c)\}$
  - $f_s(s)$ maximum security level of subject $s$
  - $f_c(s)$ current security level of subject $s$
  - $f_o(o)$ security level of object $o$

States and Requests

- $V$ set of states
  - Each state is $(b, m, f, h)$
    - $b$ is like $m$, but excludes rights not allowed by $f$
- $R$ set of requests for access
- $D$ set of outcomes
  - $y$ allowed, $n$ not allowed, $i$ illegal, $o$ error
- $W$ set of actions of the system
  - $W \subseteq R \times D \times V \times V$

Basic Security Theorem

- Define action, secure formally
- Restate properties formally
  - Simple security condition
  - "-property
  - Discretionary security property
- State conditions for properties to hold
- State Basic Security Theorem
Action

- A request and decision that causes the system to move from one state to another
  - Final state may be the same as initial state
- \((r, d, v, v') \in R \times D \times V \times V\) is an action of \(\Sigma(R, D, W, z_0)\) iff there is an \((x, y, z) \in \Sigma(R, D, W, z_0)\) and an \(t \in N\) such that \((r, d, v, v') = (x_t, y_t, z_t, z_{t-1})\)
  - Request \(r\) made when system in state \(v\); decision \(d\) moves system into (possibly the same) state \(v'\)
  - Correspondence with \((x_t, y_t, z_t, z_{t-1})\) makes states, requests, part of a sequence

Simple Security Condition

- \((s, o, p) \in S \times O \times P\) satisfies the simple security condition relative to \(f\) (written \(ssc \ rel \ f\)) iff one of the following holds:
  1. \(p = e\) or \(p = a\)
  2. \(p = r\) or \(p = w\) and \(f(s) \ dom \ f(o)\)
- Holds vacuously if rights do not involve reading
- If all elements of \(b\) satisfy \(ssc \ rel \ f\), then state satisfies simple security condition
- If all states satisfy simple security condition, system satisfies simple security condition

*-Property

- \(b(s; p_1, \ldots, p_n)\) set of all objects that \(s\) has \(p_1, \ldots, p_n\) access to
- State \((b, m, f, h)\) satisfies the *-property iff for each \(s \in S\) the following hold:
  1. \(b(s; a) \neq \emptyset \Rightarrow [\forall o \in b(s; a)\ [f(o) \ dom \ f(s)]]\)
  2. \(b(s; w) \neq \emptyset \Rightarrow [\forall o \in b(s; w)\ [f(o) = f(s)]]\)
  3. \(b(s; r) \neq \emptyset \Rightarrow [\forall o \in b(s; r)\ [f(s) \ dom \ f(o)]]\)
- Idea: for writing, object dominates subject; for reading, subject dominates object
Discretionary Security Property

- State \((b, m, f, h)\) satisfies the discretionary security property iff, for each \((s, o, p) \in b\), then \(p \in m[s, o]\)
- If \(s\) can read \(o\), then it must have rights in the access control matrix \(m\)
- This is the discretionary access control part of the model
  - The other two properties are the mandatory access control parts of the model

Secure

- A system is secure iff it satisfies:
  - Simple security condition
  - \(*\)-property
  - Discretionary security property
- A state meeting these three properties is also said to be secure

Basic Security Theorem

- \(\Sigma(R, D, W, z_0)\) is a secure system if \(z_0\) is a secure state and \(W\) satisfies the conditions for the preceding three theorems
Principle of Tranquility

- Raising object’s security level
  - Information once available to some subjects is no longer available
  - Usually assume information has already been accessed, so this does nothing
- Lowering object’s security level
  - The declassification problem
  - Essentially, a “write down” violating *-property
  - Solution: define set of trusted subjects that sanitize or remove sensitive information before security level lowered

Types of Tranquility

- Strong Tranquility
  - The clearances of subjects, and the classifications of objects, do not change during the lifetime of the system
- Weak Tranquility
  - The clearances of subjects, and the classifications of objects, do not change in a way that violates the SSC or the *-property during the lifetime of the system