Security Policy Models

comp620

Bell-LaPadula Model

- The Bell-LaPadula Model is a state machine model used for enforcing access control in government and military applications
- Developed by David Elliott Bell and Leonard J. La Padula in 1973
- Basis for many confidentiality policies

Bell-LaPadula Model, Step 1

- Security levels arranged in linear ordering
  - Top Secret: highest
  - Secret
  - Confidential
  - Unclassified: lowest
- Both people and objects have a *security level*
  - *People or subjects have a clearance level, L*(*s*)
  - Objects have *security classification, L*(*o*)

Example

L	*security level*	*subject*	*object*
4	Top Secret	Tanya	Personnel Files
3	Secret	Sam	E-Mail Files
2	Confidential	Claire	Activity Logs
1	Unclassified	Umoja	Telephone Lists

- Tanya can read all files
- Claire cannot read Personnel or E-Mail Files
- Umoja can only read Telephone Lists

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, $L(o) \le L(s)$ 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 $L(o) \ge L(s)$ 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 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 })

Dominating

- The *dom* relation (“Dominates”) specifies that the levels are greater or equal and the categories includes all items in the dominated categories.
- $(L(a), C)$ *dom* $(L(b), C')$ iff $L(a) \geq L(b)$ and $C' \subseteq C$
- Not a symmetric or asymmetric relation
- Examples
  - (Top Secret, {NUC, ASI}) *dom* (Secret, {NUC})
  - (Secret, {NUC, EUR}) *dom* (Confidential,{NUC, EUR})
  - (Top Secret, {NUC}) ¬*dom* (Confidential, {EUR})
  - (Confidential, {EUR}) ¬*dom* (Top Secret, {NUC})

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*) *dom* *L*(*o*) and *s* has permission to read *o*
  - 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*) *dom* *L*(*s*) and *s* has permission to write *o*
  - Sometimes called “no writes down” rule

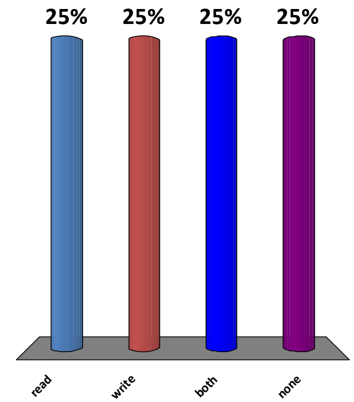
Basic Security Theorem, Step 2

- 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

What type of access is allowed?

Paul, cleared for (TOP SECRET, {A,C}), wants to access a document classified (SECRET, {B,C})

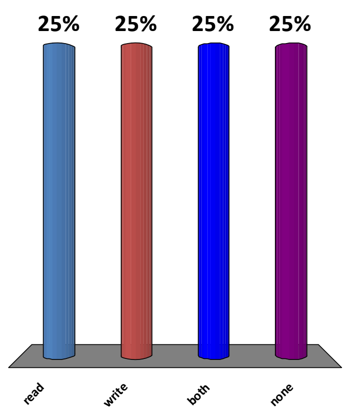
1. read
2. write
3. both
4. none



What type of access is allowed?

Mary, cleared for (CONFIDENTIAL, {C}), wants to access a document classified (CONFIDENTIAL, {B})

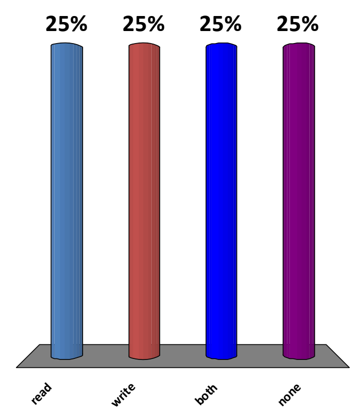
1. read
2. write
3. both
4. none



What type of access is allowed?

Fred, cleared for (SECRET, {C}), wants to access a document classified (CONFIDENTIAL, {C})

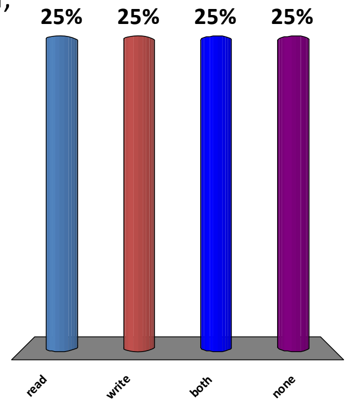
1. read
2. write
3. both
4. none



What type of access is allowed?

Susan, cleared for (TOP SECRET, {A,C}), wants to access a document classified (CONFIDENTIAL, {A})

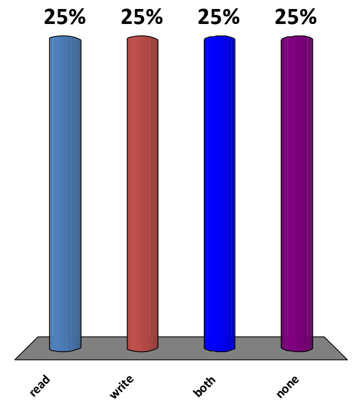
1. read
2. write
3. both
4. none



What type of access is allowed?

Joe has no clearance (UNCLASSIFIED), wants to access a document classified (CONFIDENTIAL, {B})

1. read
2. write
3. both
4. none



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*) *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*(Major) *dom curlevel*(Colonel)
    - Colonel can write to Major without violating "no writes down"
  - Does *L*(*s*) mean *curlevel*(*s*) or *maxlevel*(*s*)?

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 simple security condition or the *-property during the lifetime of the system

Biba Integrity Model

- Formal state transition system that describes a set of access control rules designed to ensure data integrity.
- Developed by Kenneth J. Biba in 1977
- The model is designed so that subjects may not corrupt data in a level ranked higher than the subject, or be corrupted by data from a lower level than the subject.

Up and Down

- This security model is directed toward data integrity (rather than confidentiality) and is characterized by the phrase: "no read down, no write up"
- This is in contrast to the Bell-LaPadula model which is characterized by the phrase "no write down, no read up".

Reading and Writing

- Users can only **create content at or below** their own integrity level (a monk may write a prayer book that can be read by commoners, but not one to be read by a high priest).
- Users can only **view content at or above** their own integrity level (a monk may read a book written by the high priest, but may not read a pamphlet written by a lowly commoner).

Information Transfer Path

- An *information transfer path* is a sequence of objects $o_1, \ldots, o_{n+1}$ and corresponding sequence of subjects $s_1, \ldots, s_n$ such that $s_i \underline{r} o_i$ and $s_i \underline{w} o_{i+1}$ for all i, $1 \le i \le n$.
- Idea: information can flow from o_1 to o_{n+1} along this path by successive reads and writes

Triad Programming Contest

- Saturday morning, April 10
- Teams of 3 students writing programs as fast as possible
- **Prizes**
  - 1st Place $300.00 (**$100 per member**)
  - 2nd Place $225.00 (**$75 per member**)
  - 3rd Place $150.00 (**$50 per member**)
- Contact Prof. Carr (carr@redux.comp.ncat.edu) if you are interested

Teaching Evaluation

- You should have received an email at your ncat account explaining how to do the teaching evaluation
- Teaching evaluations are important
- Answer honestly