Lecture 4
October 4, 2023

Lattices

- Lattices used to analyze several models
  - Bell-LaPadula confidentiality model
  - Biba integrity model
- A lattice consists of a set and a relation
- Relation must partially order set
  - Relation orders some, but not all, elements of set

Sets and Relations

- S set, R: $S \times S$ relation
  - If $a, b \in S$, and $(a, b) \in R$, write aRb
- Example
  - $I = \{ 1, 2, 3 \}$; R is $\leq$
  - $R = \{ (1, 1), (1, 2), (1, 3), (2, 2), (2, 3), (3, 3) \}$
  - So we write $1 \leq 2$ and $3 \leq 3$ but not $3 \leq 2$

Relation Properties

- Reflexive
  - For all $a \in S$, aRa
  - On I, $\leq$ is reflexive as $1 \leq 1$, $2 \leq 2$, $3 \leq 3$
- Antisymmetric
  - For all $a, b \in S$, $aRb \wedge bRa \Rightarrow a = b$
  - On I, $\leq$ is antisymmetric as $1 \leq x$ and $x \leq 1$ means $x = 1$
- Transitive
  - For all $a, b, c \in S$, $aRb \wedge bRc \Rightarrow aRc$
  - On I, $\leq$ is transitive as $1 \leq 2$ and $2 \leq 3$ means $1 \leq 3$

Example

- $\mathbb{C}$ set of complex numbers
- $a \in \mathbb{C} \Rightarrow a = a_R + a_I i$, where a_R, a_I integers
- $a \leq_C b$ if, and only if, $a_R \leq b_R$ and $a_I \leq b_I$
- $a \leq_C b$ is reflexive, antisymmetric, transitive
  - As ≤ is over integers, and a_R , a_I are integers

Partial Ordering

- Relation *R* orders some members of set *S*
  - If all ordered, it's a total ordering
- Example
  - ≤ on integers is total ordering
  - $\leq_{\mathbb{C}}$ is partial ordering on $\mathbb{C}$
    - Neither $3+5i \leq_{\mathbb{C}} 4+2i$ nor $4+2i \leq_{\mathbb{C}} 3+5i$ holds

Upper Bounds

- For a, $b \in S$, if u in S with aRu, bRu exists, then u is an *upper bound*
  - A *least upper bound* if there is no $t \in S$ such that aRt, bRt, and tRu
- Example
  - For $1 + 5i$, $2 + 4i \in \mathbb{C}$
    - Some upper bounds are $2 + 5i$, $3 + 8i$, and $9 + 100i$
    - Least upper bound is $2 + 5i$

Lower Bounds

- For $a, b \in S$, if l in S with lRa, lRb exists, then l is a *lower bound*
  - A *greatest lower bound* if there is no $t \in S$ such that tRa, tRb, and lRt
- Example
  - For $1 + 5i, 2 + 4i \in \mathbb{C}$
    - Some lower bounds are 0, $-1 + 2i$, $1 + 1i$, and $1+4i$
    - Greatest lower bound is $1 + 4i$

Lattices

- Set S, relation R
  - R is reflexive, antisymmetric, transitive on elements of S
  - For every s, $t \in S$, there exists a greatest lower bound under R
  - For every s, $t \in S$, there exists a least upper bound under R

Example

- S = { 0, 1, 2 }; R = ≤ is a lattice
  - R is clearly reflexive, antisymmetric, transitive on elements of S
  - Least upper bound of any two elements of S is the greater of the elements
  - Greatest lower bound of any two elements of S is the lesser of the elements

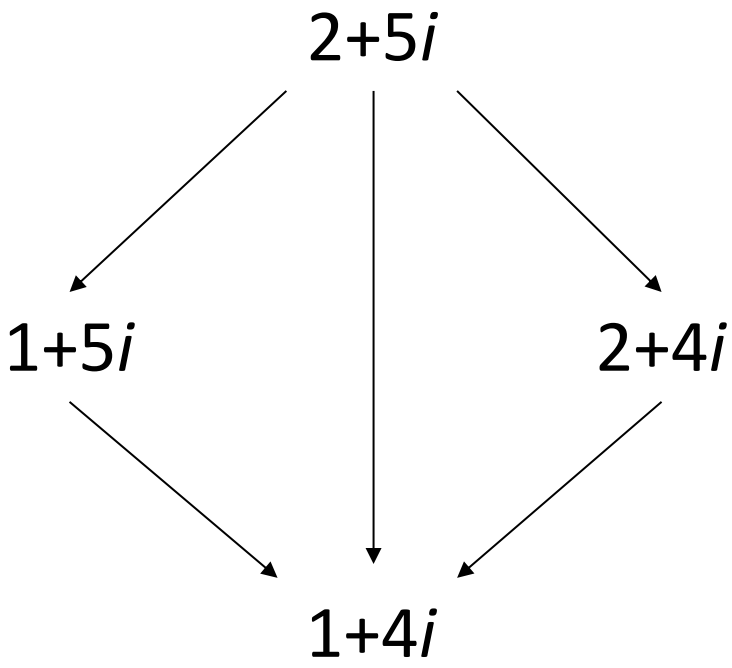
Picture

Arrows represent ≤; this forms a total ordering

Example

- $\mathbb{C}, \leq_{\mathbb{C}}$ form a lattice
  - $\leq_{\mathbb{C}}$ is reflexive, antisymmetric, and transitive
    - Shown earlier
  - Least upper bound for *a* and *b*:
    - $c_R = \max(a_R, b_R)$, $c_I = \max(a_I, b_I)$; then $c = c_R + c_I i$
  - Greatest lower bound for *a* and *b*:
    - $c_R = \min(a_R, b_R)$, $c_I = \min(a_I, b_I)$; then $c = c_R + c_I i$

Picture



Arrows represent $\leq_{\mathbb{C}}$

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) *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}) $\neg$*dom* (Confidential, {EUR})
- Let C be set of classifications, K set of categories. Set of security levels $L = C \times K$, *dom* form lattice
  - $lub(L) = (max(A), C)$
  - $glb(L) = (min(A), \varnothing)$

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*
    - 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*) *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*) *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*)?
    - Formally, we need a more precise notation

Example: Trusted Solaris

- Provides mandatory access controls
  - Security level represented by *sensitivity label*
  - Least upper bound of all sensitivity labels of a subject called *clearance*
  - Default labels ADMIN_HIGH (dominates any other label) and ADMIN_LOW (dominated by any other label)
- *S* has controlling user U_S
  - S_L sensitivity label of subject
  - *privileged*(*S*, *P*) true if *S* can override or bypass part of security policy *P*
  - *asserted* (*S*, *P*) true if *S* is doing so

Rules

C_L clearance of S, S_L sensitivity label of S, U_S controlling user of S, and O_L sensitivity label of O

1. If ¬*privileged*(S, "change S_L"), then no sequence of operations can change S_L to a value that it has not previously assumed
2. If ¬*privileged*(S, "change S_L"), then ¬ *asserted*(S, "change S_L")
3. If ¬*privileged*(S, "change S_L"), then no value of S_L can be outside the clearance of U_S
4. For all subjects S, named objects O, if ¬*privileged*(S, "change O_L"), then no sequence of operations can change O_L to a value that it has not previously assumed

Rules (*con't*)

C_L clearance of S, S_L sensitivity label of S, U_S controlling user of S, and O_L sensitivity label of O

5. For all subjects S, named objects O, if ¬*privileged*(S, "override O's mandatory read access control"), then read access to O is granted only if S_L *dom* O_L
   - Instantiation of simple security condition
6. For all subjects S, named objects O, if ¬*privileged*(S, "override O's mandatory write access control"), then write access to O is granted only if O_L *dom* S_L and C_L *dom* O_L
   - Instantiation of *-property

Initial Assignment of Labels

- Each account is assigned a label range [clearance, minimum]
- On login, Trusted Solaris determines if the session is single-level
  - If clearance = minimum, single level and session gets that label
  - If not, multi-level; user asked to specify clearance for session; must be in the label range
  - In multi-level session, can change to any label in the range of the session clearance to the minimum

Writing

- Allowed when subject, object labels are the same or file is in downgraded directory *D* with sensitivity label D_L and all the following hold:
  - S_L *dom* D_L
  - *S* has discretionary read, search access to *D*
  - O_L *dom* S_L and $O_L \neq S_L$
  - *S* has discretionary write access to *O*
  - C_L *dom* O_L
- Note: subject cannot read object

Directory Problem

- Process *p* at MAC_A tries to create file */tmp/x*
- */tmp/x* exists but has MAC label MAC_B
  - Assume MAC_B dom MAC_A
- Create fails
  - Now *p* knows a file named *x* with a higher label exists
- Fix: only programs with same MAC label as directory can create files in the directory
  - Now compilation won't work, mail can't be delivered

Multilevel Directory

- Directory with a set of subdirectories, one per label
  - Not normally visible to user
  - p creating */tmp/x* actually creates */tmp/d/x* where *d* is directory corresponding to MAC_A
  - All *p*'s references to */tmp* go to */tmp/d*
- *p* cd's to */tmp*
  - System call stat("/", &buf) returns information about */tmp/d*
  - System call mldstat("/", &buf) returns information about*/tmp*

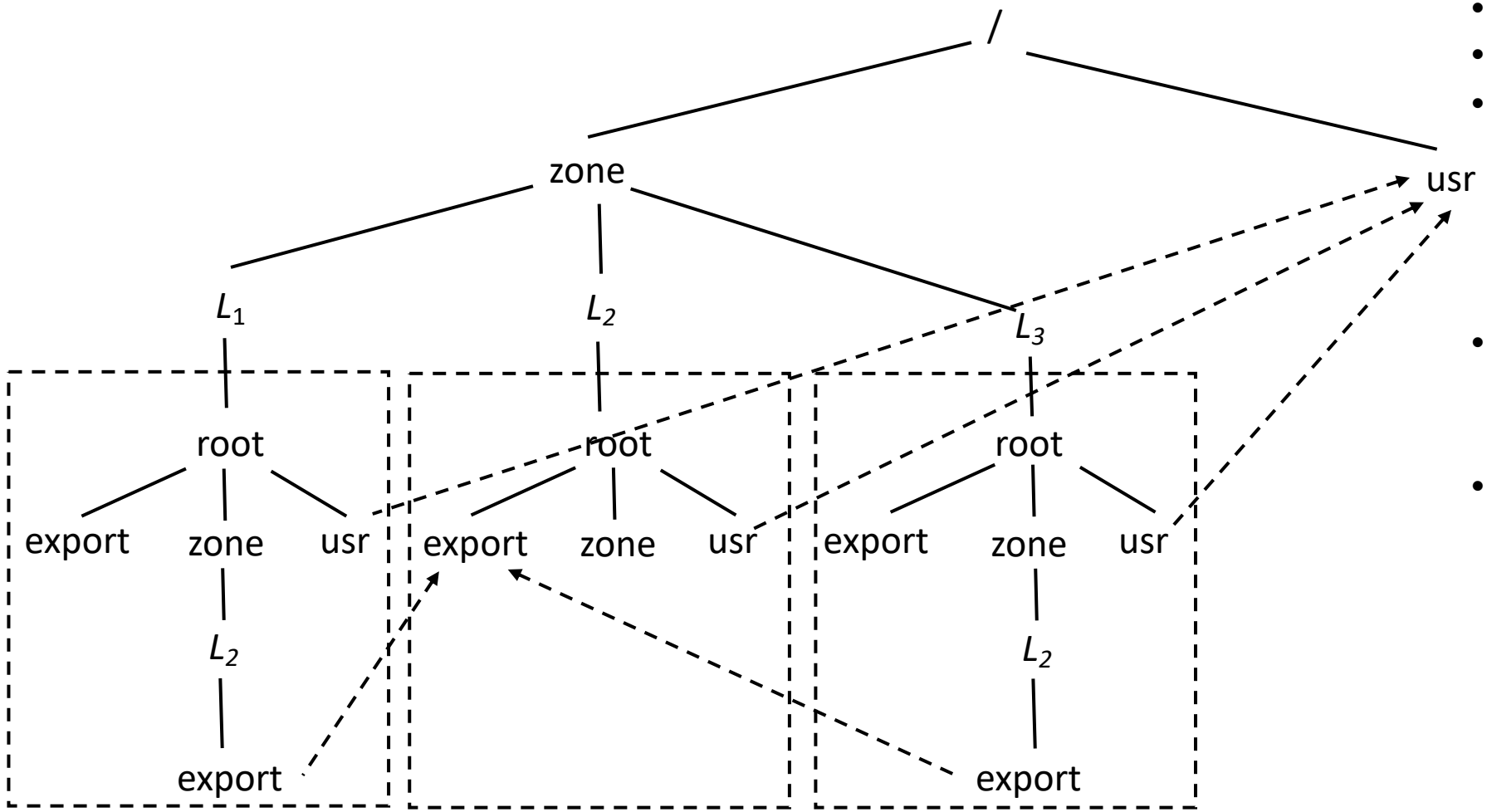
Labeled Zones

- Used in Trusted Solaris Extensions, various flavors of Linux
- *Zone*: virtual environment tied to a unique label
  - Each process can only access objects in its zone
- *Global zone* encompasses everything on system
  - Its label is ADMIN_HIGH
  - Only system administrators can access this zone
- Each zone has a unique root directory
  - All objects within the zone have that zone's label
  - Each zone has a unique label

More about Zones

- Can import (mount) file systems from other zones provided:
  - If importing *read-only*, importing zone's label must dominate imported zone's label
  - If importing *read-write*, importing zone's label must equal imported zone's label
    - So the zones are the same; import unnecessary
  - Labels checked at time of import
- Objects in imported file system retain their labels

Example



- L_1 *dom* L_2
- L_3 *dom* L_2
- Process in L_1 can read any file in the export directory of L_2 (assuming discretionary permissions allow it)
- L_1, L_3 disjoint
  - Do not share any files
- System directories imported from global zone, at ADMIN_LOW
  - So can only be read

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

Example: Trusted Solaris

- Security administrator can provide specific authorization for a user to change the MAC label of a file
  - “downgrade file label” authorization
  - “upgrade file label” authorization
- User requires additional authorization if not the owner of the file
  - “act as file owner” authorization

Principles of Declassification

- Principle of Semantic Consistency
  - As long as semantics of components that do not do declassification do not change, the components can be altered without affecting security
- Principle of Occlusion
  - A declassification operation cannot conceal an *improper* declassification
- Principle of Conservativity
  - Absent any declassification, the system is secure
- Principle of Monotonicity of Release
  - When declassification is performed in an authorized manner by authorized subjects, the system remains secure