NETWORK SECURITY PRACTICES – ATTACK AND DEFENSE

Confidentiality Model

Bell-La Padula

Bell-LaPadula Model

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

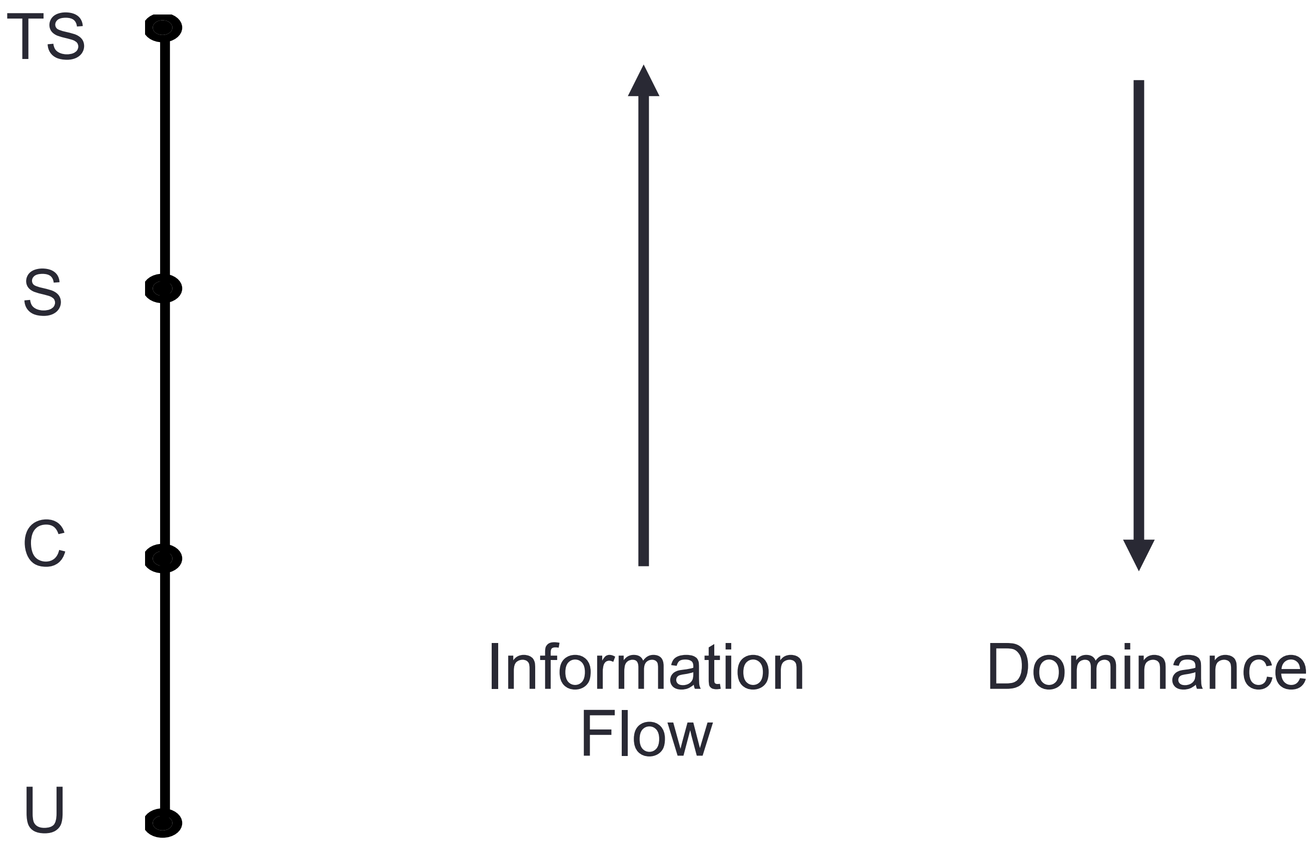
Example

security level	subject	object
Top Secret	Tamara	Personnel Files
Secret	Samuel	E-Mail Files
Confidential	Claire	Activity Logs
Unclassified	Ulaley	Telephone Lists

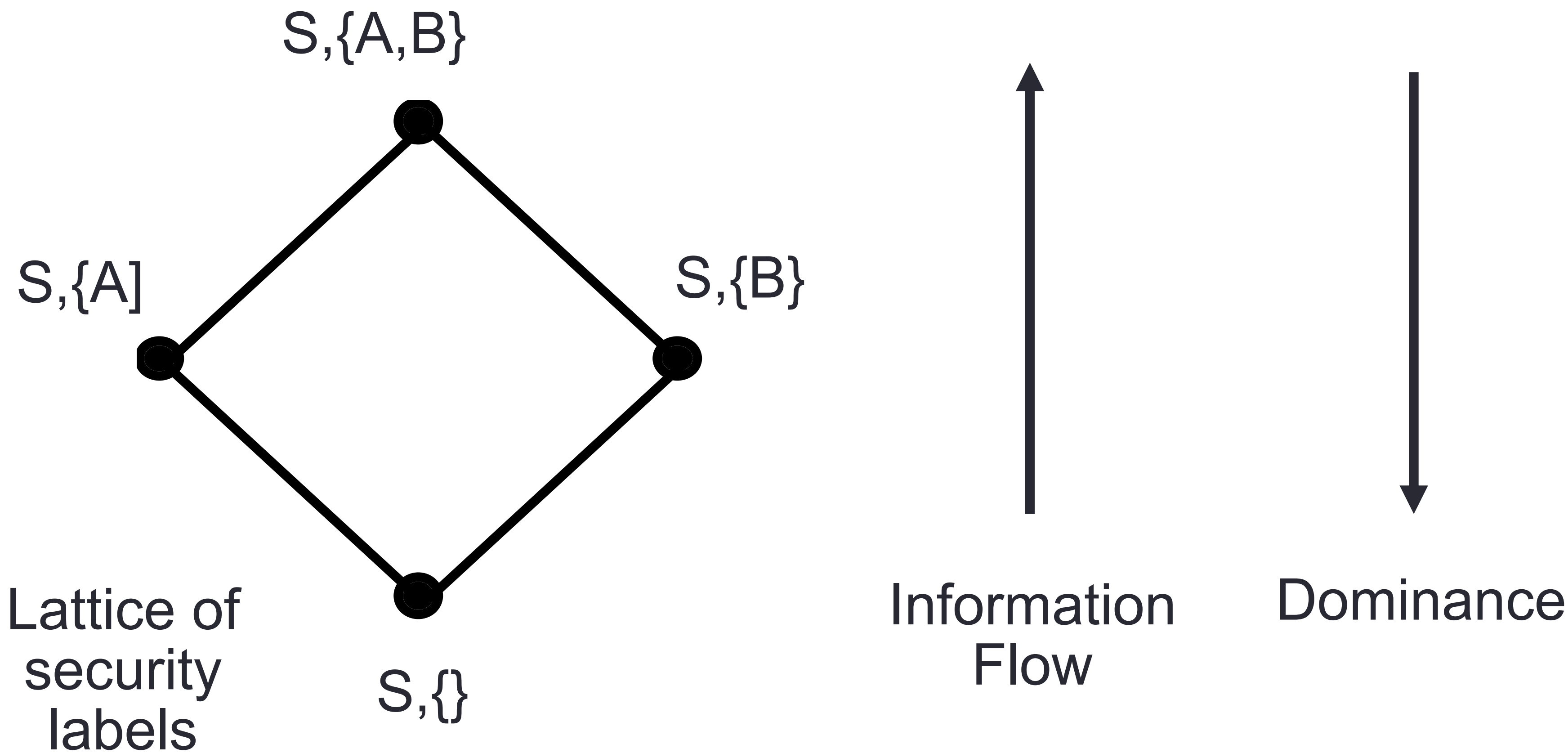
- Tamara can read all files
- Claire cannot read Personnel or E-Mail Files
- Ulaley can only read Telephone Lists

MULTILEVEL SECURITY

Lattice of security labels

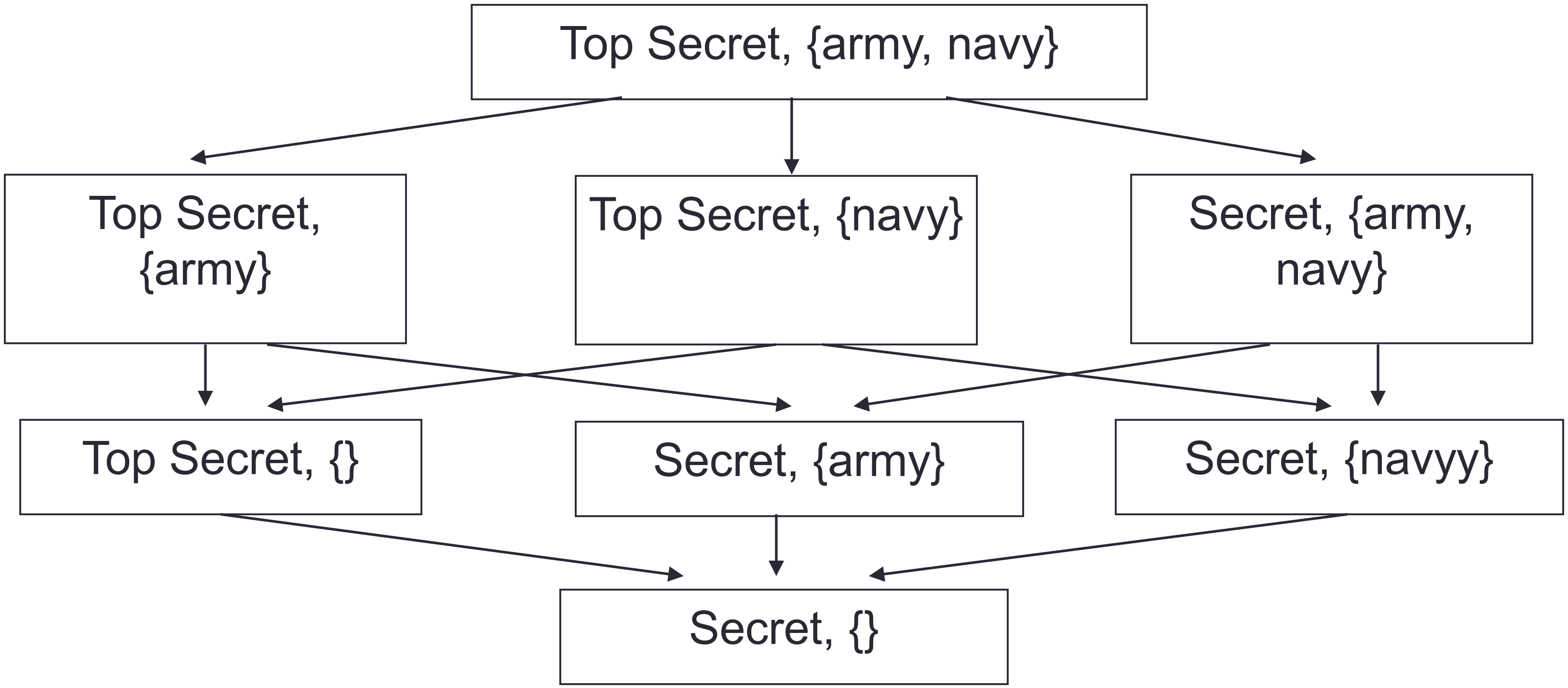


MULTILEVEL SECURITY



An Example Security Lattice

- levels={top secret, secret}
- categories={army,navy}



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 $L(s) \le L(o)$ 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

Why no write-down

ACL

File X

A:r

A:w

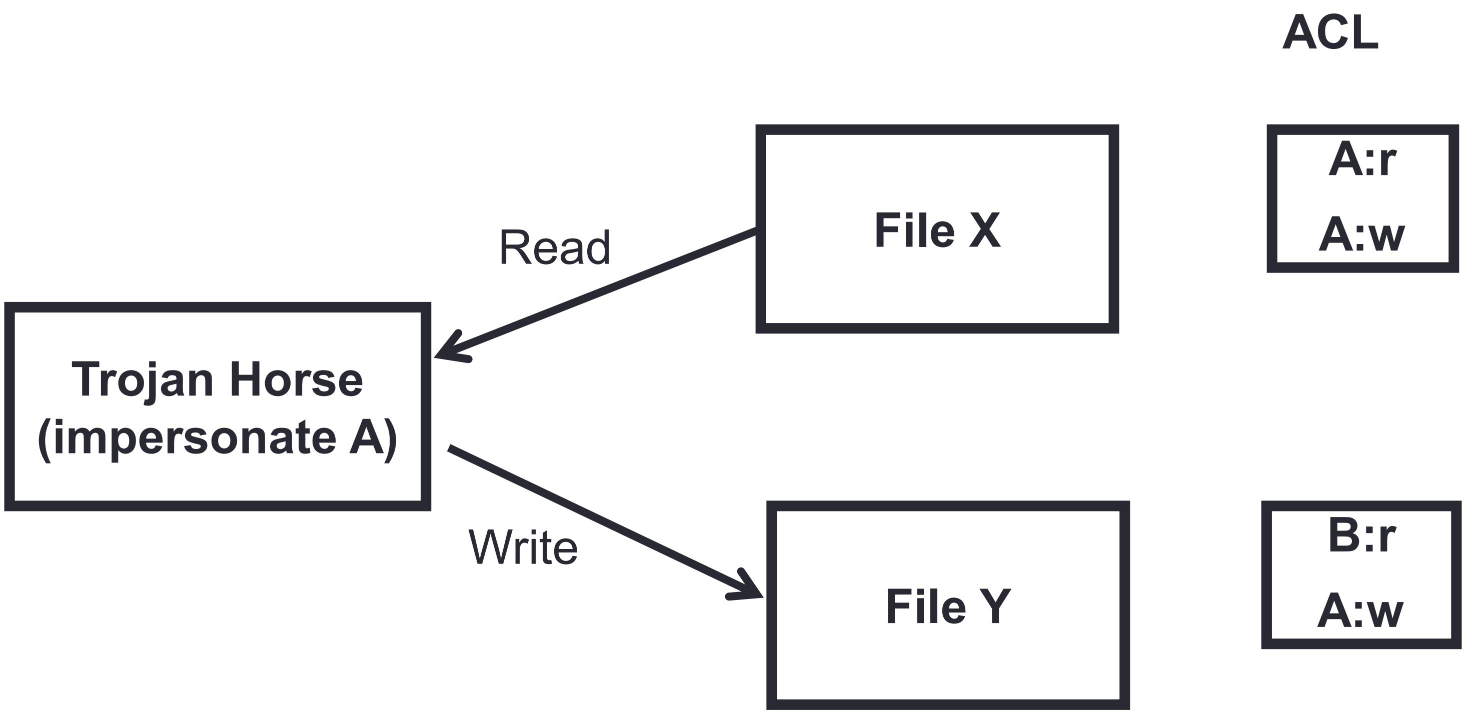
File Y

B:r

A:w

Principal B cannot read file X

Why no write-down



Principal B can read contents of file X copied to file Y

More Details in BLP

- Trusted subjects
  - some subjects are identified as trusted subjects, the star property does not apply to trusted subjects
  - why having trusted subjects?

- In the actual model, each subject has two levels: the maximum level and the current level
  - the simple security condition uses the maximum level
  - the *-property uses the current level

*-property

- Applies to subjects (principals) not to users
- Users are trusted (must be trusted) not to disclose secret information outside of the computer system
- Subjects are not trusted because they may have Trojan Horses embedded in the code they execute
- *-property prevents overt leakage of information and does not address the covert channel problem

BLP Formal Definitions

- S subjects, O objects, P rights
  - Defined rights: r read, a write, w read/write, e empty
- 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
  - $\underline{y}$ allowed, $\underline{n}$ not allowed, $\underline{i}$ illegal, $\underline{o}$ error
- W set of actions of the system
  - $W \subseteq R \times D \times V \times V$

Example

- $S = \{ s \}$, $O = \{ o \}$, $P = \{ \underline{\text{r}}, \underline{\text{w}} \}$
- $C = \{$ High, Low $\}$, $K = \{$ All $\}$
- For every $f \in F$, either $f_c(s) = ($ High, $\{$ All $\})$ or $f_c(s) = ($ Low, $\{$ All $\})$
- Initial State:
  - $b_1 = \{ (s, o, \underline{\text{r}}) \}$, $m_1 \in M$ gives s read access over o, and for $f_1 \in F$, $f_{c,1}(s) = (\text{High}, \{\text{All}\})$, $f_{o,1}(o) = (\text{Low}, \{\text{All}\})$
  - Call this state $v_0 = (b_1, m_1, f_1, h_1) \in V$.

First Transition

- Now suppose in state v_0: $S = \{ s, s' \}$
- Suppose $f_{c,1}(s') = (\text{Low}, \{\text{All}\})$
- $m_1 \in M$ gives s and s' read access over o
- As s' not written to o, $b_1 = \{ (s, o, \underline{\text{r}}) \}$
- $z_0 = v_0$; if s' requests r_1 to write to o:
  - System decides $d_1 = \underline{\text{y}}$
  - New state $v_1 = (b_2, m_1, f_1, h_1) \in V$
  - $b_2 = \{ (s, o, \underline{\text{r}}), (s', o, \underline{\text{w}}) \}$
  - Here, $x = (r_1)$, $y = (\underline{\text{y}})$, $z = (v_0, v_1)$

Second Transition

- Current state $v_1 = (b_2, m_1, f_1, h_1) \in V$
  - $b_2 = \{ (s, o, \underline{\text{r}}), (s', o, \underline{\text{w}}) \}$
  - $f_{c,1}(s) = (\text{High}, \{ \text{All} \}), f_{o,1}(o) = (\text{Low}, \{ \text{All} \})$
- s requests r_2 to write to o:
  - System decides $d_2 = \underline{\text{n}}$ (as $f_{c,1}(s)$ *dom* $f_{o,1}(o)$)
  - New state $v_2 = (b_2, m_1, f_1, h_1) \in V$
  - $b_2 = \{ (s, o, \underline{\text{r}}), (s', o, \underline{\text{w}}) \}$
  - So, $x = (r_1, r_2)$, $y = (\underline{\text{y}}, \underline{\text{n}})$, $z = (v_0, v_1, v_2)$, where $v_2 = v_1$

Basic Security Theorem

- Define action, secure formally
  - Using a bit of foreshadowing for “secure”
- 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 a $t \in N$ 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. $p = \underline{e}$ or $p = \underline{a}$
    2. $p = \underline{r}$ or $p = \underline{w}$ and $f_s(s)$ *dom* $f_o(o)$
- 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

Necessary and Sufficient

- $\Sigma(R, D, W, z_0)$ satisfies the simple security condition for any secure state z_0 iff for every action $(r, d, (b, m, f, h), (b', m', f', h'))$, W satisfies
  - Every $(s, o, p) \in b - b'$ satisfies *ssc rel f*
  - Every $(s, o, p) \in b'$ that does not satisfy *ssc rel f* is not in b
- Note: "secure" means z_0 satisfies *ssc rel f*
- First says every (s, o, p) added satisfies *ssc rel f*; second says any (s, o, p) in b' that does not satisfy *ssc rel f* is deleted

*-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. $b(s: \underline{a}) \neq \varnothing \Rightarrow [\forall o \in b(s: \underline{a}) \, [\, f_o(o) \; dom \; f_c(s) \,]\,]$
    2. $b(s: \underline{w}) \neq \varnothing \Rightarrow [\forall o \in b(s: \underline{w}) \, [\, f_o(o) = f_c(s) \,]\,]$
    3. $b(s: \underline{r}) \neq \varnothing \Rightarrow [\forall o \in b(s: \underline{r}) \, [\, f_c(s) \; dom \; f_o(o) \,]\,]$
- Idea: for writing, object dominates subject; for reading, subject dominates object

*-Property

- If all states satisfy simple security condition, system satisfies simple security condition
- If a subset S' of subjects satisfy *-property, then *-property satisfied relative to $S' \subseteq S$
- Note: tempting to conclude that *-property includes simple security condition, but this is false
  - See condition placed on <u>w</u> right for each

Necessary and Sufficient

- $\Sigma(R, D, W, z_0)$ satisfies the *-property relative to $S' \subseteq S$ for any secure state z_0 iff for every action $(r, d, (b, m, f, h), (b', m', f', h'))$, W satisfies the following for every $s \in S'$
  - Every $(s, o, p) \in b - b'$ satisfies the *-property relative to S'
  - Every $(s, o, p) \in b'$ that does not satisfy the *-property relative to S' is not in b
- Note: "secure" means z_0 satisfies *-property relative to S'
- First says every (s, o, p) added satisfies the *-property relative to S'; second says any (s, o, p) in b' that does not satisfy the *-property relative to S' is deleted

Discretionary Security Property

- State (b, m, f, h) satisfies the discretionary security property iff, for each $(s, o, p) \in b$, then $p \in m[s, o]$
- Idea: if s can read o, then it must have rights to do so 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

Necessary and Sufficient

- $\Sigma(R, D, W, z_0)$ satisfies the ds-property for any secure state z_0 iff, for every action $(r, d, (b, m, f, h), (b', m', f', h'))$, W satisfies:
  - Every $(s, o, p) \in b - b'$ satisfies the ds-property
  - Every $(s, o, p) \in b'$ that does not satisfy the ds-property is not in b
- Note: "secure" means z_0 satisfies ds-property
- First says every (s, o, p) added satisfies the ds-property; second says any (s, o, p) in b' that does not satisfy the *-property is deleted

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
  - The theorems are on the slides titled "Necessary and Sufficient"

Is BLP Notion of Security Good?

- The objective of BLP security is to ensure
  - a subject cleared at a low level should never read information classified high
- The ss-property and the *-property are sufficient to stop such information flow at any given state.
- What about information flow across states?

BLP Security Is Not Sufficient!

- Consider a system with s_1,s_2,o_1,o_2
  - $f_S(s_1)=f_C(s_1)=f_O(o_1)=\text{high}$
  - $f_S(s_2)=f_C(s_2)=f_O(o_2) =\text{low}$
- And the following execution
  - s_1 gets access to o_1, read something, release access, then change current level to low, get write access to o_2, write to o_2
- Every state is secure, yet illegal information exists

How to Deal With This?

- The following have been proposed:
  - subject cannot change current levels
  - require a subject to “forget” everything when changing levels
- But the original BLP security is wrong!
- And all the fixes limit the applicability of the model
- It is not the model that is wrong, it is the definition of security that is wrong.

BLP Security Is Not Necessary!

- Consider a system with only s_1,s_2,o_1,o_2
  - $f_S(s_1)=f_C(s_1)=f_O(o_1)$=high
  - $f_S(s_2)=f_C(s_2)=f_O(o_2)$ =low
- And an access matrix s.t. s_2 cannot access o_2
- And the following execution
  - s_1 gets access to o_1, and get write access to o_2, then the state violates *-property
- Why is this system bad?

Summary of Issues with BLP Notion of Security

- BLP notion of security is neither sufficient nor necessary to stop illegal information flow (through overt channels)
- The state based approach is too low level and limited in expressive power

How to Fix The BLP Notion of Security?

- May need to differentiate externally visible objects from other objects
  - e.g., a printer is different from a memory object
- State-sequence based property
  - e.g., exists no sequence of states so that there is an information path from a high object to a low externally visible object or to a low subject

Basic Security Theorem

- Restatement of The Basic Security Theorem: A system (z_0,W) is a secure system if and only if z_0 is a secure state and each action of the system leads the system into a secure state.
- Given a system (z_0,W), $\sigma \in W$ is an action of the system iff. there is an appearance of the system that uses σ

Observations of the BST

- The BST is a result of defining security as a state-based property.
- The BST cannot be used to justify the BLP notion of security
  - This is McLean's main point in his papers
    - "A Comment on the Basic Security Theorem of Bell and LaPadula" [1985]
    - "Reasoning About Security Models" [1987]
    - "The Specification and Modeling of Computer Security" [1990]

Observations of the BST

- The BST intends to provide a necessary and sufficient condition for verifying that a system is secure without running the system
  - [McLean 90]: “The most notable theorem known about BLP-security is called the `Basic Security Theorem (BST), which gives necessary and sufficient conditions for a system starting in a secure state to never reach a non-secure state.”

BST and Static Verification of Security

- Can one use BST to verify whether a system is secure or not without running the system?
  - Repeat of BST: A system (z_0, W) is a secure system if and only if z_0 is a secure state and each action of the system leads the system into a secure state.

BST and Static Verification of Security

- Yes and No.
  - if every $\sigma \in W$ leads the system into a secure state, then the system is secure
  - if some $\sigma \in W$ leads the system into an insecure state, then we don’t know whether the system is secure
    - as we don’t know whether σ is an action or not

- BST provides effectively only sufficient (but not necessary) conditions.

McLean’s Criticism of BLP

- BST cannot be used to justify BLP security
  - [McLean 1985] If one define security to be any other state-based property, BST still holds
    - Defense [Bell 1988]: exactly what is security is outside the model
  - [McLean 1987] System Z, defines a state change that downgrade everything
    - Defense 1: Tranquility principle disallows that
    - Defense 2: If such state change is desired, then fine.
- Tranquility principle
  - the security levels of subjections & objects will not change during the normal operation.

Main Contributions of BLP

- The overall methodology to show that a system is secure
  - adopted in many later works
- The state-transition model
  - which includes an access matrix, subject security levels, object levels, etc.
- The introduction of *-property
  - ss-property is not enough to stop illegal information flow

Main Technical Flaws of BLP

- The BLP notion of security is neither necessary nor sufficient to stop illegal information flows
- That BLP defines security as a state-based property is too low level and limited in expressive power
- The BST fails to provide *necessary* conditions for verifying a system is BLP-secure

Other Issues with BLP

- Deal only with confidentiality,
  - does not deal with integrity at all
- Does not deal with information flow through covert channels

Overt (Explicit) Channels vs. Covert Channels

- Security objective of MLS in general, BLP in particular
  - high-classified information cannot flow to low-cleared users
- Overt channels of information flow
  - read/write an object
- Covert channels of information flow
  - communication channel based on the use of system resources not normally intended for communication between the subjects (processes) in the system

Examples of Covert Channels

- Using file lock as a shared boolean variable
- By varying its ratio of computing to input/output or its paging rate, the service can transmit information to a concurrently running process
- Covert channels are often noisy
- However, information theory and coding theory can be used to encode and decode information through noisy channels

More on Covert Channels

- Covert channels cannot be blocked by *-property
- It is generally very difficult, if not impossible, to block all covert channels
- One can try to limit the bandwidth of covert channels
- Military requires cryptographic components be implemented in hardware
  - to avoid Trojan horse leaking keys through covert channels

More on MLS: Security Levels

- Used as attributes of both subjects & objects
  - clearance & classification
- Typical military security levels:
  - top secret ≥ secret ≥ confidential ≥ unclassified
- Typical commercial security levels
  - restricted ≥ proprietary ≥ sensitive ≥ public

Security Categories

- Also known as compartments
- Typical military security categories
  - army, navy, air force
  - nato, nasa, noforn
- Typical commercial security categories
  - Sales, R&D, HR
  - Dept A, Dept B, Dept C

Security Labels

- Labels = Levels $\times$ P (Categories)
- Define an ordering relationship among Labels
  - (e1, C1) $\leq$ (e2, C2) iff. e1 $\leq$e2 and C1 $\subseteq$ C2
- This ordering relation is a partial order
  - reflexive, transitive, anti-symmetric
  - e.g., $\subseteq$
- All security labels form a lattice

Key Points

- Confidentiality models restrict flow of information
- Bell-LaPadula models multilevel security
  - Cornerstone of much work in computer security
- Controversy over meaning of security
  - Different definitions produce different results

Readings

- You can take a look at Chapter 5 of the book 'Computer Security: art and science' by Matt Bishop (available at NCTU Library)
- Secure Computer System by Bell and La Padula
  - http://csrc.nist.gov/publications/history/bell76.pdf
- 'A lattice model for secure information flow' by Dorothy E. Denning
  - http://faculty.nps.edu/dedennin/publications/lattice76.pdf
- Role-based Access Control
  - http://en.wikipedia.org/wiki/Rbac