April 21: Bell-LaPadula Model

- Bell-LaPadula confidentiality model
- Tranquility
- Declassification
- McLean’s criticism and System Z
Rule

- \( \rho: R \times V \rightarrow D \times V \)
- Takes a state and a request, returns a decision and a (possibly new) state
- Rule \( \rho \) *ssc-preserving* if for all \((r, v) \in R \times V\) and \(v\) satisfying \(ssc \ rel f\), \(\rho(r, v) = (d, v')\) means that \(v'\) satisfies \(ssc \ rel f'\).
  - Similar definitions for \(*\)-property, ds-property
  - If rule meets all 3 conditions, it is *security-preserving*
Unambiguous Rule Selection

- Problem: multiple rules may apply to a request in a state
  - if two rules act on a read request in state $v$ …
- Solution: define relation $W(\omega)$ for a set of rules $\omega = \{ \rho_1, \ldots, \rho_m \}$ such that a state $(r, d, v, v') \in W(\omega)$ iff either
  - $d = i$; or
  - for exactly one integer $j$, $\rho_j(r, v) = (d, v')$
- Either request is illegal, or only one rule applies
Rules Preserving SSC

- Let $\omega$ be set of ssc-preserving rules. Let state $z_0$ satisfy simple security condition. Then $\Sigma(R, D, W(\omega), z_0)$ satisfies simple security condition
  - Proof: by contradiction.
    - Choose $(x, y, z) \in \Sigma(R, D, W(\omega), z_0)$ as state not satisfying simple security condition; then choose $t \in N$ such that $(x_t, y_t, z_t)$ is first appearance not meeting simple security condition
    - As $(x_t, y_t, z_t, z_{t-1}) \in W(\omega)$, there is unique rule $\rho \in \omega$ such that $\rho(x_t, z_{t-1}) = (y_t, z_t)$ and $y_t \neq i$.
    - As $\rho$ ssc-preserving, and $z_{t-1}$ satisfies simple security condition, then $z_t$ meets simple security condition, contradiction.
Adding States Preserving SSC

- Let $v = (b, m, f, h)$ satisfy simple security condition. Let $(s, o, p) \notin b$, $b' = b \cup \{ (s, o, p) \}$, and $v' = (b', m, f, h)$. Then $v'$ satisfies simple security condition iff:

  1. Either $p = e$ or $p = a$; or
  2. Either $p = r$ or $p = w$, and $f_c(s) \text{ dom } f_o(o)$

- Proof

  1. Immediate from definition of simple security condition and $v'$ satisfying $ssc \ rel \ f$
  2. $v'$ satisfies simple security condition means $f_s(s) \text{ dom } f_o(o)$, and for converse, $(s, o, p) \in b'$ satisfies $ssc \ rel \ f$, so $v'$ satisfies simple security condition
Rules, States Preserving $\ast$-Property

- Let $\omega$ be set of $\ast$-property-preserving rules, state $z_0$ satisfies $\ast$-property. Then $\Sigma(R, D, W(\omega), z_0)$ satisfies $\ast$-property.
Rules, States Preserving ds-Property

• Let $\omega$ be set of ds-property-preserving rules, state $z_0$ satisfies ds-property. Then $\Sigma(R, D, W(\omega), z_0)$ satisfies ds-property
Combining

• Let \( \rho \) be a rule and \( \rho(r, v) = (d, v') \), where \( v = (b, m, f, h) \) and \( v' = (b', m', f', h') \). Then:
  1. If \( b' \subseteq b, f' = f \), and \( v \) satisfies the simple security condition, then \( v' \) satisfies the simple security condition
  2. If \( b' \subseteq b, f' = f \), and \( v \) satisfies the *-property, then \( v' \) satisfies the *-property
  3. If \( b' \subseteq b, m[s, o] \subseteq m'[s, o] \) for all \( s \in S \) and \( o \in O \), and \( v \) satisfies the ds-property, then \( v' \) satisfies the ds-property
Proof

1. Suppose $\nu$ satisfies simple security property.
   a) $b' \subseteq b$ and $(s, o, r) \in b'$ implies $(s, o, r) \in b$
   b) $b' \subseteq b$ and $(s, o, w) \in b'$ implies $(s, o, w) \in b$
   c) So $f_c(s) \text{ dom } f_o(o)$
   d) But $f' = f$
   e) Hence $f'_c(s) \text{ dom } f'_o(o)$
   f) So $\nu'$ satisfies simple security condition

2, 3 proved similarly
Example Instantiation: Multics

- 11 rules affect rights:
  - set to request, release access
  - set to give, remove access to different subject
  - set to create, reclassify objects
  - set to remove objects
  - set to change subject security level

- Set of “trusted” subjects $S_T \subseteq S$
  - *-property not enforced; subjects trusted not to violate

- $\Delta(\rho)$ domain
  - determines if components of request are valid
get-read Rule

• Request \( r = (\text{get}, s, o, \_\_\_) \)
  
  – \( s \) gets (requests) the right to read \( o \)

• Rule is \( \rho_1(r, v) \):
  
  if \((r \neq \Delta(\rho_1))\) then \( \rho_1(r, v) = (\_\_, v) \);
  
  else if \((f_s(s) \text{ dom } f_o(o) \text{ and } [s \in S_T \text{ or } f_c(s) \text{ dom } f_o(o)]) \text{ and } r \in m[s, o])\) then \( \rho_1(r, v) = (y, (b \cup \{(s, o, \_\_)\}, m, f, h)) \);
  
  else \( \rho_1(r, v) = (n, v) \);
Security of Rule

• The get-read rule preserves the simple security condition, the *-property, and the ds-property
  – Proof
    • Let \( \nu \) satisfy all conditions. Let \( \rho_1(r, \nu) = (d, \nu') \). If \( \nu' = \nu \), result is trivial. So let \( \nu' = (b \cup \{ (s_2, o, r) \}, m, f, h) \).
• Consider the simple security condition.
  – From the choice of $v'$, either $b' - b = \emptyset$ or $\{(s_2, o, r)\}$
  – If $b' - b = \emptyset$, then $\{(s_2, o, r)\} \in b$, so $v = v'$, proving that $v'$ satisfies the simple security condition.
  – If $b' - b = \{(s_2, o, r)\}$, because the get-read rule requires that $f_s(s) \text{dom} f_o(o)$, an earlier result says that $v'$ satisfies the simple security condition.
Proof

- Consider the \(*\)-property.
  - Either $s_2 \in S_T$ or $f_c(s) \in \text{dom} f_o(o)$ from the definition of get-read
  - If $s_2 \in S_T$, then $s_2$ is trusted, so \(*\)-property holds by definition of trusted and $S_T$.
  - If $f_c(s) \in \text{dom} f_o(o)$, an earlier result says that $v'$ satisfies the simple security condition.
Proof

- Consider the discretionary security property.
  - Conditions in the get-read rule require $r \in m[s, o]$ and either $b' - b = \emptyset$ or $\{ (s_2, o, r) \}$
  - If $b' - b = \emptyset$, then $\{ (s_2, o, r) \} \in b$, so $v = v'$, proving that $v'$ satisfies the simple security condition.
  - If $b' - b = \{ (s_2, o, r) \}$, then $\{ (s_2, o, r) \} \notin b$, an earlier result says that $v'$ satisfies the ds-property.
Rules, States, and Conditions

Let $\rho$ be a rule and $\rho(r, v) = (d, v')$, where $v = (b, m, f, h)$ and $v' = (b', m', f', h')$. Then:

1. If $b \subseteq b', f = f'$, and $v$ satisfies the simple security condition, then $v'$ satisfies the simple security condition.

2. If $b \subseteq b', f = f'$, and $v$ satisfies the *-property, then $v'$ satisfies the *-property.

3. If $b \subseteq b', m[s, o] \subseteq m'[s, o]$ for all $s \in S$ and $o \in O$, and $v$ satisfies the ds-property, then $v'$ satisfies the ds-property.
Example Instantiation: Multics

• 11 rules affect rights:
  – set to request, release access
  – set to give, remove access to different subject
  – set to create, reclassify objects
  – set to remove objects
  – set to change subject security level

• Set of “trusted” subjects $S_T \subseteq S$
  – *-property not enforced; subjects trusted not to violate

• $\Delta(\rho)$ domain
  – determines if components of request are valid
get-read Rule

• Request \( r = (\text{get}, s, o, \_r) \)
  - \( s \) gets (requests) the right to read \( o \)

• Rule is \( \rho_1(r, v) \):
  \[
  \text{if } (r \neq \Delta(\rho_1)) \text{ then } \rho_1(r, v) = (\_i, v); \\
  \text{else if } (f_s(s) \text{ dom } f_o(o) \text{ and } [s \in S_T \text{ or } f_c(s) \text{ dom } f_o(o)]) \\
  \text{and } r \in m[s, o])
  \text{ then } \rho_1(r, v) = (y, (b \cup \{ (s, o, \_r) \}, m, f, h)); \\
  \text{else } \rho_1(r, v) = (\_n, v);
  \]
Security of Rule

- The get-read rule preserves the simple security condition, the \(*\)-property, and the ds-property
  - Proof
    - Let \( v \) satisfy all conditions. Let \( \rho_1(r, v) = (d, v') \). If \( v' = v \), result is trivial. So let \( v' = (b \cup \{ (s_2, o, r) \}, m, f, h) \).
Proof

• Consider the simple security condition.
  – From the choice of $v'$, either $b' - b = \emptyset$ or $\{ (s_2, o, r) \}$
  – If $b' - b = \emptyset$, then $\{ (s_2, o, r) \} \in b$, so $v = v'$, proving that $v'$ satisfies the simple security condition.
  – If $b' - b = \{ (s_2, o, r) \}$, because the get-read rule requires that $f_c(s) \text{dom} f_o(o)$, an earlier result says that $v'$ satisfies the simple security condition.
Proof

• Consider the *-property.
  – Either $s_2 \in S_T$ or $f_c(s) \text{ dom } f_o(o)$ from the definition of get-read
  – If $s_2 \in S_T$, then $s_2$ is trusted, so *-property holds by definition of trusted and $S_T$.
  – If $f_c(s) \text{ dom } f_o(o)$, an earlier result says that $v'$ satisfies the simple security condition.
Proof

• Consider the discretionary security property.
  – Conditions in the get-read rule require \( r \in m[s, o] \) and either \( b' - b = \emptyset \) or \( \{ (s_2, o, r) \} \)
  – If \( b' - b = \emptyset \), then \( \{ (s_2, o, r) \} \in b \), so \( v = v' \), proving that \( v' \) satisfies the simple security condition.
  – If \( b' - b = \{ (s_2, o, r) \} \), then \( \{ (s_2, o, r) \} \not\in b \), an earlier result says that \( v' \) satisfies the ds-property.
give-read Rule

- Request $r = (s_1, \text{give}, s_2, o, \underline{r})$
  - $s_1$ gives (request to give) $s_2$ the (discretionary) right to read $o$
  - Rule: can be done if giver can alter parent of object
    - If object or parent is root of hierarchy, special authorization required
- Useful definitions
  - $\text{root}(o)$: root object of hierarchy $h$ containing $o$
  - $\text{parent}(o)$: parent of $o$ in $h$ (so $o \in h(\text{parent}(o)))$
  - $\text{canallow}(s, o, v)$: $s$ specially authorized to grant access when object or parent of object is root of hierarchy
  - $m \land m[s, o] \leftarrow \underline{r}$: access control matrix $m$ with $\underline{r}$ added to $m[s, o]$
give-read Rule

- Rule is $\rho_6(r, v)$:
  
  \[
  \text{if} \ (r \neq \Delta(\rho_6)) \ \text{then} \ \rho_6(r, v) = (i, v);
  \]
  
  \[
  \text{else if} \ ([o \neq \text{root}(o) \ \text{and} \ \text{parent}(o) \neq \text{root}(o) \ \text{and} \ \text{parent}(o) \in b(s_1:w)] \ \text{or} \\
  [\text{parent}(o) = \text{root}(o) \ \text{and} \ \text{canallow}(s_1, o, v) ] \ \text{or} \\
  [o = \text{root}(o) \ \text{and} \ \text{canallow}(s_1, o, v) ]) \\
  \text{then} \ \rho_6(r, v) = (y, (b, m \land m[s_2, o] \leftarrow r, f, h));
  \]
  
  \[
  \text{else} \ \rho_1(r, v) = (n, v);
  \]
The *-property, and the ds-property

- Proof: Let \( \nu \) satisfy all conditions. Let \( \rho_1(r, \nu) = (d, \nu') \).
  If \( \nu' = \nu \), result is trivial. So let \( \nu' = (b, m[s_2, o] \leftarrow \_f, h) \).
  So \( b' = b, f' = f, m'[x, y] = m[x, y] \) for all \( x \in S \) and \( y \in O \) such that \( x \neq s \) and \( y \neq o \), and \( m[s, o] \subseteq m'[s, o] \).
  Then by earlier result, \( \nu' \) satisfies the simple security condition, the *-property, and the ds-property.
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 of Weak Tranquility

- Only one subject at TOP SECRET
- Document at CONFIDENTIAL
- New CONFIDENTIAL user to be added
  - User should not see document
- Raise document to SECRET
  - Subject still cannot write document
  - All security relationships unchanged
Declassification

- Lowering the security level of a document
  - Direct violation of the “no writes down” rule
  - May be necessary for legal or other purposes

- Declassification policy
  - Part of security policy covering this
  - Here, “secure” means classification changes to a lower level in accordance with declassification policy
Principles

- Principle of Semantic Consistency
- Principle of Occlusion
- Principle of Conservativity
- Principle of Monotonicity of Release
Principle of Semantic Consistency

• As long as the semantics of the parts of the system not involved in the declassification do not change, those parts may be changed without affecting system security
  – No leaking due to semantic incompatibilities
  – *Delimited release*: allow declassification, release of information only through specific channels (“escape hatches”)

April 21, 2017

ECS 235B Spring Quarter 2017
Principle of Occlusion

• Declassification mechanism cannot conceal *improper* lowering of security levels
  – Robust declassification property: attacker cannot use escape hatches to obtain information unless it is properly declassified
Other Principles

- Principle of Conservativity
  - Absent declassification, system is secure

- Principle of Monotonicity of Release
  - When declassification is performed in an authorized manner by authorized subjects, the system remains secure

Idea: declassifying information in accordance with declassification policy does not affect security
Controversy

• McLean:
  – “value of the BST is much overrated since there is a great deal more to security than it captures. Further, what is captured by the BST is so trivial that it is hard to imagine a realistic security model for which it does not hold.”
  – Basis: given assumptions known to be non-secure, BST can prove a non-secure system to be secure
†-Property

- State \((b, m, f, h)\) satisfies the †-property iff for each \(s \in S\) the following hold:
  1. \(b(s: \text{a}) \neq \emptyset \Rightarrow \forall o \in b(s: \text{a}) \left[ f_c(s) \text{ dom } f_o(o) \right] \)
  2. \(b(s: \text{w}) \neq \emptyset \Rightarrow \forall o \in b(s: \text{w}) \left[ f_o(o) = f_c(s) \right] \)
  3. \(b(s: \text{r}) \neq \emptyset \Rightarrow \forall o \in b(s: \text{r}) \left[ f_c(s) \text{ dom } f_o(o) \right] \)
- Idea: for reading, subject dominates object; for writing, subject also dominates object
- Differs from *-property in that the mandatory condition for writing is reversed
  - For *-property, it’s “object dominates subject”
Analogues

The following two theorems can be proved

• $\Sigma(R, D, W, z_0)$ 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 $b$

• $\Sigma(R, D, W, z_0)$ is a secure system if $z_0$ is a secure state and $W$ satisfies the conditions for the simple security condition, the †-property, and the ds-property.
Problem

• This system is clearly non-secure!
  – Information flows from higher to lower because of the †-property
Discussion

• Role of Basic Security Theorem is to demonstrate that rules preserve security

• Key question: what is security?
  – Bell-LaPadula defines it in terms of 3 properties (simple security condition, *-property, discretionary security property)
  – Theorems are assertions about these properties
  – Rules describe changes to a particular system instantiating the model
  – Showing system is secure requires proving rules preserve these 3 properties
Rules and Model

- Nature of rules is irrelevant to model
- Model treats “security” as axiomatic
- Policy defines “security”
  - This instantiates the model
  - Policy reflects the requirements of the systems
- McLean’s definition differs from Bell-LaPadula
  - … and is not suitable for a confidentiality policy
- Analysts cannot prove “security” definition is appropriate through the model
System Z

- System supporting weak tranquility
- On *any* request, system downgrades *all* subjects and objects to lowest level and adds the requested access permission
  - Let initial state satisfy all 3 properties
  - Successive states also satisfy all 3 properties
- Clearly not secure
  - On first request, everyone can read everything
Reformulation of Secure Action

- Given state that satisfies the 3 properties, the action transforms the system into a state that satisfies these properties and eliminates any accesses present in the transformed state that would violate the property in the initial state, then the action is secure.
- BST holds with these modified versions of the 3 properties.
Reconsider System Z

- Initial state:
  - subject $s$, object $o$
  - $C = \{\text{High, Low}\}$, $K = \{\text{All}\}$
- Take:
  - $f_c(s) = (\text{Low, } \{\text{All}\})$, $f_o(o) = (\text{High, } \{\text{All}\})$
  - $m[s, o] = \{w\}$, and $b = \{(s, o, w)\}$.
- $s$ requests $r$ access to $o$
- Now:
  - $f'_o(o) = (\text{Low, } \{\text{All}\})$
  - $(s, o, r) \in b'$, $m'[s, o] = \{r, w\}$
Non-Secure System Z

• As \((s, o, r) \in b' - b\) and \(f_o(o) \text{ dom } f_c(s)\), access added that was illegal in previous state
  
  – Under the new version of the Basic Security Theorem, the current state of System Z is not secure

  – But, as \(f'_c(s) = f'_o(o)\) under the old version of the Basic Security Theorem, the current state of System Z is secure
Response: What Is Modeling?

- Two types of models
  1. Abstract physical phenomenon to fundamental properties
  2. Begin with axioms and construct a structure to examine the effects of those axioms
- Bell-LaPadula Model developed as a model in the first sense
  - McLean assumes it was developed as a model in the second sense
Reconciling System Z

• Different definitions of security create different results
  – Under one (original definition in Bell-LaPadula Model), System Z is secure
  – Under other (McLean’s definition), System Z is not secure