ECS 235B Module 18
Bell-LaPadula Model

Formal Model Definitions

- S subjects, O objects, P rights
  - Defined rights: $\underline{r}$ read, $\underline{a}$ write, $\underline{w}$ read/write, $\underline{e}$ 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

More Definitions

- Hierarchy functions H: $O \rightarrow \mathbb{P}(O)$
- Requirements
  1. $o_i \neq o_j \Rightarrow h(o_i) \cap h(o_j) = \varnothing$
  2. There is no set $\{ o_1, \ldots, o_k \} \subseteq O$ such that for $i = 1, \ldots, k$, $o_{i+1} \in h(o_i)$ and $o_{k+1} = o_1$.
- Example
  - Tree hierarchy; take $h(o)$ to be the set of children of o
  - No two objects have any common children (#1)
  - There are no loops in the tree (#2)

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
  - $W \subseteq R \times D \times V \times V$

History

- $X = R^N$ set of sequences of requests
- $Y = D^N$ set of sequences of decisions
- $Z = V^N$ set of sequences of states
- Interpretation
  - At time $t \in N$, system is in state $z_{t-1} \in V$; request $x_t \in R$ causes system to make decision $y_t \in D$, transitioning the system into a (possibly new) state $z_t \in V$
- System representation: $\Sigma(R, D, W, z_0) \in X \times Y \times Z$
  - $(x, y, z) \in \Sigma(R, D, W, z_0)$ iff $(x_t, y_t, z_{t-1}, z_t) \in W$ for all t
  - (x, y, z) called an *appearance* of $\Sigma(R, D, W, z_0)$

Example

- $S = \{ s \}$, $O = \{ o \}$, $P = \{ \underline{\mathrm{r}}, \underline{\mathrm{w}} \}$
- $C = \{ \text{High}, \text{Low} \}$, $K = \{ \text{All} \}$
- For every $f \in F$, either $f_c(s) = (\text{High}, \{ \text{All} \})$ or $f_c(s) = (\text{Low}, \{ \text{All} \})$
- Initial State:
  - $b_1 = \{ (s, o, \underline{\mathrm{r}}) \}$, $m_1 \in M$ 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 $f_{s,1}(s')$ = (Low, {All}), $m_1 \in M$ gives s read access over o and s' write access to o
- As s' not written to o, $b_1 = \{ (s, o, \underline{\text{r}}) \}$
- r_1: s' requests to write to o:
    - System decides $d_1 = \underline{\text{y}}$ (as m_1 gives it that right, and $f_{s,1}(s') = f_o(o)$)
    - 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 $v_1 = (b_2, m_1, f_1, h_1) \in V$
  - $b_2 = \{ (s, o, \underline{\mathrm{r}}), (s', o, \underline{\mathrm{w}}) \}$
  - $f_{c,1}(s) = (\text{High}, \{ \text{All} \})$, $f_{o,1}(o) = (\text{Low}, \{ \text{All} \})$
- r_2: s requests to write to o:
  - System decides $d_2 = \underline{\mathrm{n}}$ (as $f_{c,1}(s)$ *dom* $f_{o,1}(o)$)
  - New state $v_2 = (b_2, m_1, f_1, h_1) \in V$
  - $b_2 = \{ (s, o, \underline{\mathrm{r}}), (s', o, \underline{\mathrm{w}}) \}$
  - So, $x = (r_1, r_2)$, $y = (\underline{\mathrm{y}}, \underline{\mathrm{n}})$, $z = (v_0, v_1, v_2)$, 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. $p = \underline{\text{e}}$ or $p = \underline{\text{a}}$
  2. $p = \underline{\text{r}}$ or $p = \underline{\text{w}}$ 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. $b(s: \underline{\mathrm{a}}) \neq \varnothing \Rightarrow [\forall o \in b(s: \underline{\mathrm{a}})\ [\ f_o(o)\ dom\ f_c(s)\ ]\ ]$
  2. $b(s: \underline{\mathrm{w}}) \neq \varnothing \Rightarrow [\forall o \in b(s: \underline{\mathrm{w}})\ [\ f_o(o) = f_c(s)\ ]\ ]$
  3. $b(s: \underline{\mathrm{r}}) \neq \varnothing \Rightarrow [\forall o \in b(s: \underline{\mathrm{r}})\ [\ f_c(s)\ dom\ f_o(o)\ ]\ ]$
- Idea: for writing, object dominates subject; for reading, subject dominates object

*-Property

- 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
  - Note simple security condition uses f_s; *-property uses f_c

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 ds-property is deleted

Secure

- A state is secure iff it satisfies:
  - Simple security condition
  - *-property
  - Discretionary security property
- A system is secure if the only states it can enter satisfy the above 3 properties

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"