Lecture #15

- Noninterference
  - Review notation
  - Definition
  - Security policy in these terms
  - Unwinding theorem
  - Example interpretation
  - Dynamic policies
  - Composition
Security Policy

- Partitions systems into authorized, unauthorized states
- Authorized states have no forbidden interferences
- Hence a *security policy* is a set of noninterference assertions
  - See previous definition
Alternative Development

- System $X$ is a set of protection domains $D = \{ d_1, \ldots, d_n \}$
- When command $c$ executed, it is executed in protection domain $\text{dom}(c)$
- Give alternate versions of definitions shown previously
Output-Consistency

- $c \in C$, $\text{dom}(c) \in D$
- $\sim_{\text{dom}(c)}$ equivalence relation on states of system $X$
- $\sim_{\text{dom}(c)}$ output-consistent if
  \[ \sigma_a \sim_{\text{dom}(c)} \sigma_b \Rightarrow P(c, \sigma_a) = P(c, \sigma_b) \]
- Intuition: states are output-consistent if for subjects in $\text{dom}(c)$, projections of outputs for both states after $c$ are the same
Security Policy

- \( D = \{ d_1, \ldots, d_n \} \), \( d_i \) a protection domain
- \( r: D \times D \) a reflexive relation
- Then \( r \) defines a security policy
- Intuition: defines how information can flow around a system
  - \( d_i rd_j \) means info can flow from \( d_i \) to \( d_j \)
  - \( d_i rd_i \) as info can flow within a domain
Projection Function

- $\pi'$ analogue of $\pi$, earlier
- Commands, subjects absorbed into protection domains
- $d \in D, c \in C, c_s \in C^*$
- $\pi'_d(\nu) = \nu$
- $\pi'_d(c_s c) = \pi'_d(c_s)c$ if $\text{dom}(c)\text{rd}$
- $\pi'_d(c_s c) = \pi'_d(c_s)$ otherwise
- Intuition: if executing $c$ interferes with $d$, then $c$ is visible; otherwise, as if $c$ never executed
Noninterference-Secure

- System has set of protection domains $D$
- System is noninterference-secure with respect to policy $r$ if
  \[ P^*(c, T^*(c_s, \sigma_0)) = P^*(c, T^*(\pi'_d(c_s), \sigma_0)) \]
- Intuition: if executing $c_s$ causes the same transitions for subjects in domain $d$ as does its projection with respect to domain $d$, then no information flows in violation of the policy
Lemma

- Let $T^*(c_s, \sigma_0) \overset{d}{\sim} T^*(\pi'_d(c_s), \sigma_0)$ for $c \in C$
- If $\overset{d}{\sim}$ output-consistent, then system is noninterference-secure with respect to policy $r$
Proof

• $d = \text{dom}(c)$ for $c \in C$

• By definition of output-consistent,

$$T^*(c_s, \sigma_0) \sim^d T^*(\pi'_d(c_s), \sigma_0)$$

implies

$$P^*(c, T^*(c_s, \sigma_0)) = P^*(c, T^*(\pi'_d(c_s), \sigma_0))$$

• This is definition of noninterference-secure with respect to policy $r$
Unwinding Theorem

- Links security of sequences of state transition commands to security of individual state transition commands
- Allows you to show a system design is ML secure by showing it matches specs from which certain lemmata derived
  - Says *nothing* about security of system, because of implementation, operation, *etc.* issues
 Locally Respects

- $r$ is a policy
- System $X$ locally respects $r$ if $\text{dom}(c)$ being noninterfering with $d \in D$ implies $\sigma_a \sim^d T(c, \sigma_a)$
- Intuition: applying $c$ under policy $r$ to system $X$ has no effect on domain $d$ when $X$ locally respects $r$
Transition-Consistent

• \( r \) policy, \( d \in D \)
• If \( \sigma_a \sim^d \sigma_b \) implies \( T(c, \sigma_a) \sim^d T(c, \sigma_b) \), system \( X \) transition-consistent under \( r \)
• Intuition: command \( c \) does not affect equivalence of states under policy \( r \)
Lemma

• $c_1, c_2 \in C$, $d \in D$
• For policy $r$, $\text{dom}(c_1)rd$ and $\text{dom}(c_2)rd$
• Then
  \[ T^*(c_1c_2, \sigma) = T(c_1, T(c_2, \sigma)) = T(c_2, T(c_1, \sigma)) \]
• Intuition: if info can flow from domains of commands into $d$, then order doesn’t affect result of applying commands
Theorem

- $r$ policy, $X$ system that is output consistent, transition consistent, locally respects $r$
- $X$ noninterference-secure with respect to policy $r$
- Significance: basis for analyzing systems claiming to enforce noninterference policy
  - Establish conditions of theorem for particular set of commands, states with respect to some policy, set of protection domains
  - Noninterference security with respect to $r$ follows
Proof

• Must show $\sigma_a \sim^d \sigma_b$ implies

$$T^*(c_s, \sigma_a) \sim^d T^*(\pi'_d(c_s), \sigma_b)$$

• Induct on length of $c_s$

• Basis: $c_s = \nu$, so $T^*(c_s, \sigma) = \sigma$; $\pi'_d(\nu) = \nu$; claim holds

• Hypothesis: $c_s = c_1 \ldots c_n$; then claim holds
Induction Step

• Consider $c_s c_{n+1}$. Assume $\sigma_a \sim^d \sigma_b$ and look at $T^* (\pi'_d (c_s c_{n+1}), \sigma_b)$

• 2 cases:
  – $dom(c_{n+1})rd$ holds
  – $dom(c_{n+1})rd$ does not hold
$\text{dom}(c_{n+1})rd$ Holds

\[ T^*(\pi'_d(c_s c_{n+1}), \sigma_b) = T^*(\pi'_d(c_s) c_{n+1}, \sigma_b) \]

\[ = T(c_{n+1}, T^*(\pi'_d(c_s), \sigma_b)) \]

– by definition of $T^*$ and $\pi'_d$

- $T(c_{n+1}, \sigma_a) \sim^d T(c_{n+1}, \sigma_b)$
  – as $X$ transition-consistent and $\sigma_a \sim^d \sigma_b$

- $T(c_{n+1}, T^*(c_s, \sigma_a)) \sim^d T(c_{n+1}, T^*(\pi'_d(c_s), \sigma_b))$
  – by transition-consistency and IH
\[ \text{dom}(c_{n+1}) \text{rd Holds} \]

\[ T(c_{n+1}, T^*(c_s, \sigma_a)) \sim^d T(c_{n+1}, T^*(\pi'_d(c_s)c_{n+1}, \sigma_b)) \]
- by substitution from earlier equality

\[ T(c_{n+1}, T^*(c_s, \sigma_a)) \sim^d T(c_{n+1}, T^*(\pi'_d(c_s)c_{n+1}, \sigma_b)) \]
- by definition of \( T^* \)

- proving hypothesis
\[ \text{dom}(c_{n+1}) \text{rd} \text{ Does Not Hold} \]

\[ T^*(\pi'_d(c_{s+1}, \sigma_b)) = T^*(\pi'_d(c_s), \sigma_b) \]
  \[- \text{ by definition of } \pi'_d \]

\[ T^*(c_s, \sigma_b) = T^*(\pi'_d(c_{s+1}), \sigma_b) \]
  \[- \text{ by above and IH} \]

\[ T(c_{n+1}, T^*(c_s, \sigma_a)) \sim^d T^*(c_s, \sigma_a) \]
  \[- \text{ as } X \text{ locally respects } r, \text{ so } \sigma \sim^d T(c_{n+1}, \sigma) \text{ for any } \sigma \]

\[ T(c_{n+1}, T^*(c_s, \sigma_a)) \sim^d T(c_{n+1}, T^*(\pi'_d(c_s) c_{n+1}, \sigma_b)) \]
  \[- \text{ substituting back} \]

- proving hypothesis
Finishing Proof

• Take $\sigma_a = \sigma_b = \sigma_0$, so from claim proved by induction,
  
  $$T^*(c_s, \sigma_0) \sim^d T^*(\pi'_d(c_s), \sigma_0)$$

• By previous lemma, as $X$ (and so $\sim^d$) output consistent, then $X$ is noninterference-secure with respect to policy $r$
Access Control Matrix

- Example of interpretation
- Given: access control information
- Question: are given conditions enough to provide noninterference security?
- Assume: system in a particular state
  - Encapsulates values in ACM
ACM Model

- Objects $L = \{ l_1, \ldots, l_m \}$
  - Locations in memory
- Values $V = \{ v_1, \ldots, v_n \}$
  - Values that $L$ can assume
- Set of states $\Sigma = \{ \sigma_1, \ldots, \sigma_k \}$
- Set of protection domains $D = \{ d_1, \ldots, d_j \}$
Functions

- **value**: \( L \times \Sigma \rightarrow V \)
  - returns value \( v \) stored in location \( l \) when system in state \( \sigma \)
- **read**: \( D \rightarrow 2^V \)
  - returns set of objects observable from domain \( d \)
- **write**: \( D \rightarrow 2^V \)
  - returns set of objects observable from domain \( d \)
Interpretation of ACM

• Functions represent ACM
  – Subject $s$ in domain $d$, object $o$
  – $r \in A[s,o]$ if $o \in \text{read}(d)$
  – $w \in A[s,o]$ if $o \in \text{write}(d)$

• Equivalence relation:

$$\left[ \sigma_a \sim^{\text{dom}(c)} \sigma_b \right] \iff \left[ \forall l_i \in \text{read}(d) \left[ \text{value}(l_i, \sigma_a) = \text{value}(l_i, \sigma_b) \right] \right]$$

  – You can read the \textit{exactly} the same locations in both states
Enforcing Policy

- 5 requirements
  - 3 general ones describing dependence of commands on rights over input and output
    - Hold for all ACMs and policies
  - 2 that are specific to some security policies
    - Hold for most policies
Enforcing Policy $r$: First

- Output of command $c$ executed in domain $\text{dom}(c)$ depends only on values for which subjects in $\text{dom}(c)$ have read access

$$\sigma_a \sim_{\text{dom}(c)} \sigma_b \Rightarrow P(c, \sigma_a) = P(c, \sigma_b)$$
Enforcing Policy $r$: Second

- If $c$ changes $l_i$, then $c$ can only use values of objects in $\text{read}(\text{dom}(c))$ to determine new value

\[
[ \sigma_a \sim^{\text{dom}(c)} \sigma_b \text{ and } \\
(value(l_i, T(c, \sigma_a)) \neq value(l_i, \sigma_a) \text{ or } \\
value(l_i, T(c, \sigma_b)) \neq value(l_i, \sigma_b)) ] \Rightarrow \\
value(l_i, T(c, \sigma_a)) = value(l_i, T(c, \sigma_b))
\]
Enforcing Policy $r$: Third

- If $c$ changes $l_i$, then $\text{dom}(c)$ provides subject executing $c$ with write access to $l_i$

$$\text{value}(l_i, T(c, \sigma_a)) \neq \text{value}(l_i, \sigma_a) \implies l_i \in \text{write}(\text{dom}(c))$$
Enforcing Policies $r$: Fourth

- If domain $u$ can interfere with domain $v$, then every object that can be read in $u$ can also be read in $v$
- So if object $o$ cannot be read in $u$, but can be read in $v$; and object $o'$ in $u$ can be read in $v$, then info flows from $o$ to $o'$, then to $v$

Let $u, v \in D$; then $urv \Rightarrow \text{read}(u) \subseteq \text{read}(v)$
Enforcing Policies r: Fifth

• Subject $s$ can write object $o$ in $v$, subject $s'$ can read $o$ in $u$, then domain $v$ can interfere with domain $u$

$$l_i \in \text{read}(u) \text{ and } l_i \in \text{write}(v) \Rightarrow vru$$
Theorem

• Let $X$ be a system satisfying the five conditions. The $X$ is noninterference-secure with respect to $r$

• Proof: must show $X$ output-consistent, locally respects $r$, transition-consistent
  – Then by unwinding theorem, theorem holds
Output-Consistent

- Take equivalence relation to be $\sim^d$, first condition is definition of output-consistent
Locally Respects $r$

- Proof by contradiction: assume $(\text{dom}(c), d) \notin r$ but $\sigma_a \sim^d T(c, \sigma_a)$ does not hold
- Some object has value changed by $c$:
  \[ \exists l_i \in \text{read}(d) \ [ \text{value}(l_i, \sigma_a) \neq \text{value}(l_i, T(c, \sigma_a)) ] \]
- Condition 3: $l_i \in \text{write}(d)$
- Condition 5: $\text{dom}(c)rd$, contradiction
- So $\sigma_a \sim^d T(c, \sigma_a)$ holds, meaning $X$ locally respects $r$
Transition Consistency

- Assume $\sigma_a \sim^d \sigma_b$
- Must show
  \[\text{value}(l_i, T(c, \sigma_a)) = \text{value}(l_i, T(c, \sigma_b))\]
  for $l_i \in \text{read}(d)$
- 3 cases dealing with change that $c$ makes in $l_i$ in states $\sigma_a, \sigma_b$
Case 1

- \( \text{value}(l_i, T(c, \sigma_a)) \neq \text{value}(l_i, \sigma_a) \)
- Condition 3: \( l_i \in \text{write}(\text{dom}(c)) \)
- As \( l_i \in \text{read}(d) \), condition 5 says \( \text{dom}(c) \text{rd} \)
- Condition 4 says \( \text{read}(\text{dom}(c)) \subseteq \text{read}(d) \)
- As \( \sigma_a \sim^d \sigma_b, \sigma_a \sim^{\text{dom}(c)} \sigma_b \)
- Condition 2:
  \[
  \text{value}(l_i, T(c, \sigma_a)) = \text{value}(l_i, T(c, \sigma_b))
  \]
- So \( T(c, \sigma_a) \sim^{\text{dom}(c)} T(c, \sigma_b) \), as desired
Case 2

• \( \text{value}(l_i, T(c, \sigma_b)) \neq \text{value}(l_i, \sigma_b) \)
• Condition 3: \( l_i \in \text{write}(\text{dom}(c)) \)
• As \( l_i \in \text{read}(d) \), condition 5 says \( \text{dom}(c)rd \)
• Condition 4 says \( \text{read}(\text{dom}(c)) \subseteq \text{read}(d) \)
• As \( \sigma_a \sim^d \sigma_b \), \( \sigma_a \sim^{\text{dom}(c)} \sigma_b \)
• Condition 2:

\[
\text{value}(l_i, T(c, \sigma_a)) = \text{value}(l_i, T(c, \sigma_b))
\]
• So \( T(c, \sigma_a) \sim^{\text{dom}(c)} T(c, \sigma_b) \), as desired
Case 3

• Neither of the previous two
  – \( \text{value}(l_i, T(c, \sigma_a)) = \text{value}(l_i, \sigma_a) \)
  – \( \text{value}(l_i, T(c, \sigma_b)) = \text{value}(l_i, \sigma_b) \)

• Interpretation of \( \sigma_a \sim_d \sigma_b \) is:
  for \( l_i \in \text{read}(d) \), \( \text{value}(l_i, \sigma_a) = \text{value}(l_i, \sigma_b) \)

• So \( T(c, \sigma_a) \sim_d T(c, \sigma_b) \), as desired

• In all 3 cases, \( X \) transition-consistent
Policies Changing Over Time

- Problem: previous analysis assumes static system
  - In real life, ACM changes as system commands issued
- Example: $w \in C^*$ leads to current state
  - $cando(w, s, z)$ holds if $s$ can execute $z$ in current state
  - Condition noninterference on $cando$
  - If $\neg cando(w, \text{Lara}, \text{“write } f\text{”})$, Lara can’t interfere with any other user by writing file $f$
Generalize Noninterference

- $G \subseteq S$ group of subjects, $A \subseteq Z$ set of commands, $p$ predicate over elements of $C^*$
- $c_s = (c_1, \ldots, c_n) \in C^*$
- $\pi''(\nu) = \nu$
- $\pi''((c_1, \ldots, c_n)) = (c_1', \ldots, c_n')$
  - $c_i' = \nu$ if $p(c_1', \ldots, c_{i-1}')$ and $c_i = (s, z)$ with $s \in G$ and $z \in A$
  - $c_i' = c_i$ otherwise
Intuition

- $\pi''(c_s) = c_s$

- But if $p$ holds, and element of $c_s$ involves both command in $A$ and subject in $G$, replace corresponding element of $c_s$ with empty command $\nu$
  - Just like deleting entries from $c_s$ as $\pi_{A,G}$ does earlier
Noninterference

- $G, G' \subseteq S$ groups of subjects, $A \subseteq Z$ set of commands, $p$ predicate over $C^*$
- Users in $G$ executing commands in $A$ are noninterfering with users in $G'$ under condition $p$ iff, for all $c_s \in C^*$, all $s \in G'$, $\text{proj}(s, c_s, \sigma_i) = \text{proj}(s, p''(c_s), \sigma_i)$
- Written $A, G :| G'$ if $p$
Example

- From earlier one, simple security policy based on noninterference:

\[ \forall (s \in S) \forall (z \in Z) \]

\[ \{ \{z\}, \{s\} :| S \textbf{ if } \neg \textit{cando}(w, s, z) \}\]

- If subject can’t execute command (the \(\neg \textit{cando}\) part), subject can’t use that command to interfere with another subject
Another Example

• Consider system in which rights can be passed
  – \( \text{pass}(s, z) \) gives \( s \) right to execute \( z \)
  – \( w_n = v_1, \ldots, v_n \) sequence of \( v_i \in C^* \)
  – \( \text{prev}(w_n) = w_{n-1} \); \( \text{last}(wn) = v_n \)
Policy

• No subject \( s \) can use \( z \) to interfere if, in previous state, \( s \) did not have right to \( z \), and no subject gave it to \( s \)

\[
\{ z \}, \{ s \} \models S \text{ if } \\
[ \neg \text{cando}(\text{prev}(w), s, z) \land \\
[ \text{cando}(\text{prev}(w), s', \text{pass}(s, z)) \Rightarrow \\
\neg \text{last}(w) = (s', \text{pass}(s, z)) ] ]
\]
Effect

- Suppose $s_1 \in S$ can execute $\text{pass}(s_2, z)$
- For all $w \in C^*$, $\text{cando}(w, s_1, \text{pass}(s_2, z))$ true
- Initially, $\text{cando}(v, s_2, z)$ false
- Let $z' \in Z$ be such that $(s_3, z')$ noninterfering with $(s_2, z)$
  - So for each $w_n$ with $v_n = (s_3, z')$,
    \[
    \text{cando}(w_n, s_2, z) = \text{cando}(w_{n-1}, s_2, z)
    \]
Effect

• Then policy says for all \( s \in S \)
  \[
  \text{proj}(s, ((s_2, z), (s_1, \text{pass}(s_2, z))),
  (s_3, z'), (s_2, z)), \sigma_i) =
  \text{proj}(s, ((s_1, \text{pass}(s_2, z)), (s_3, z'), (s_2, z)), \sigma_i)
  \]
• So \( s_2 \)'s first execution of \( z \) does not affect any subject’s observation of system
Policy Composition I

• Assumed: Output function of input
  – Means deterministic (else not function)
  – Means uninterruptability (differences in timings can cause differences in states, hence in outputs)

• This result for deterministic, noninterference-secure systems
Compose Systems

- Louie, Dewey LOW
- Hughie HIGH
- $b_L$ output buffer
  - Anyone can read it
- $b_H$ input buffer
  - From HIGH source
- Hughie reads from:
  - $b_{LH}$ (Louie writes)
  - $b_{LDH}$ (Louie, Dewey write)
  - $b_{DH}$ (Dewey writes)
Systems Secure

- All noninterference-secure
  - Hughie has no output
    - So inputs don’t interfere with it
  - Louie, Dewey have no input
    - So (nonexistent) inputs don’t interfere with outputs