can\textbullet steal Predicate

Definition:

\textbullet can\textbullet steal(r, x, y, G_0) if, and only if, there is no edge from x to y labeled r in G_0, and the following hold simultaneously:

-- There is edge from x to y labeled r in G_n
-- There is a sequence of rule applications \( \rho_1, \ldots, \rho_n \) such that \( G_{i-1} \vdash G_i \) using \( \rho_i \)
-- For all vertices v, w in \( G_{i-1} \), if there is an edge from v to y in \( G_0 \) labeled r, then \( \rho_i \) is not of the form “v grants (r to y) to w”
Example

- *can-steal(α, s, w, G₀)*:
  1. u grants (t to v) to s
  2. s takes (t to u) from v
  3. s takes (α to w) from u

(can-steal) Theorem

- *can-steal(r, x, y, G₀)* if, and only if, the following hold simultaneously:
  a) There is no edge from x to y labeled r in G₀
  b) There exists a subject x' such that x' = x or x' initially spans to x
  c) There exists a vertex s with an edge labelled α to y in G₀
  d) *can-share(t, x', s, G₀)* holds
Outline of Proof

$\Rightarrow$: Assume conditions hold

- $x$ subject
  - $x$ gets $t$ rights to $s$, then takes $\alpha$ to $y$ from $s$

- $x$ object
  - $can\cdot share(t, x', s, G_0)$ holds
  - If $x'$ has no $\alpha$ edge to $y$ in $G_0$, $x'$ takes ($\alpha$ to $y$) from $s$ and grants it to $x$
  - If $x'$ has a edge to $y$ in $G_0$, $x'$ creates surrogate $x''$, gives it ($t$ to $s$) and ($g$ to $x''$); then $x''$ takes ($\alpha$ to $y$) and grants it to $x$

Outline of Proof

$\Leftarrow$: Assume $can\cdot steal(\alpha, x, y, G_0)$ holds

- First two conditions immediate from definition of $can\cdot steal$, $can\cdot share$
- Third condition immediate from theorem of conditions for $can\cdot share$
- Fourth condition: $\rho$ minimal length sequence of rule applications deriving $G_n$ from $G_0$; $i$ smallest index such that $G_{i-1} \models G_i$ by rule $\rho_i$ and adding $\alpha$ from some $p$ to $y$ in $G_i$;
  - What is $\rho_i$?
Outline of Proof

- Not remove or create rule
  - y exists already
- Not grant rule
  - G, first graph in which edge labeled α to y is added, so by definition of can•share, cannot be grant
- take rule: so can•share(t, p, s, G_0) holds
  - So is subject s' such that s' = s or terminally spans to s
  - Sequence of islands with x' ∈ l_i and s' ∈ l_n
- Derive witness to can•share(t, x', s, G_0) that does not use “s grants (α to y) to” anyone

Conspiracy

- Minimum number of actors to generate a witness for can•share(α, x, y, G_0)
- Access set describes the “reach” of a subject
- Deletion set is set of vertices that cannot be involved in a transfer of rights
- Build conspiracy graph to capture how rights flow, and derive actors from it
Example

Access Set

- *Access set $A(y)$ with focus $y$*: set of vertices:
  - $\{ y \}$
  - $\{ x | y$ initially spans to $x \}$
  - $\{ x' | y$ terminally spans to $x \}$
- Idea is that focus can give rights to, or acquire rights from, a vertex in this set
Example

- $A(x) = \{ x, a \}$
- $A(b) = \{ b, a \}$
- $A(c) = \{ c, b, d \}$
- $A(d) = \{ d \}$
- $A(e) = \{ e, d, i, j \}$
- $A(y) = \{ y \}$
- $A(f) = \{ f, y \}$
- $A(h) = \{ h, f, i \}$

Deletion Set

- Deletion set $\delta(y, y')$: contains those vertices in $A(y) \cap A(y')$ such that:
  - $y$ initially spans to $z$ and $y'$ terminally spans to $z$;
  - $y$ terminally spans to $z$ and $y'$ initially spans to $z$;
  - $z = y$
  - $z = y'$
- Idea is that rights can be transferred between $y$ and $y'$ if this set non-empty
Example

\[ (x, b) = \{ a \} \]
\[ (d, e) = \{ d \} \]
\[ (b, c) = \{ b \} \]
\[ (y, f) = \{ y \} \]
\[ (c, d) = \{ d \} \]
\[ (h, f) = \{ f \} \]
\[ (c, e) = \{ d \} \]

Conspiracy Graph

- Abstracted graph \( H \) from \( G_0 \):
  - Each subject \( x \in G_0 \) corresponds to a vertex \( h(x) \in H \)
  - If \( \delta(x, y) \neq \emptyset \), there is an edge between \( h(x) \) and \( h(y) \) in \( H \)
- Idea is that if \( h(x), h(y) \) are connected in \( H \), then rights can be transferred between \( x \) and \( y \) in \( G_0 \)
Example

Results

- \( l(x) \): \( h(x) \), all vertices \( h(y) \) such that \( y \) initially spans to \( x \)
- \( T(x) \): \( h(x) \), all vertices \( h(y) \) such that \( y \) terminally spans to \( x \)
- Theorem: \textit{can\-share}(\( \alpha, x, y, G_0 \)) iff there exists a path from some \( h(p) \) in \( l(x) \) to some \( h(q) \) in \( T(y) \)
- Theorem: \( l \) vertices on shortest path between \( h(p) \), \( h(q) \) in above theorem; \( l \) conspirators necessary and sufficient to witness
Example: Conspirators

- \( I(x) = \{ h(x) \} \), \( T(z) = \{ h(e) \} \)
- Path between \( h(x) \), \( h(e) \) so can share \( r, x, z, G_0 \)
- Shortest path between \( h(x) \), \( h(e) \) has 4 vertices
  \( \Rightarrow \) Conspirators are \( e, c, b, x \)

Example: Witness

- \( e \) grants \( r \) to \( z \) to \( d \)
- \( c \) takes \( r \) to \( z \) from \( d \)
- \( c \) grants \( r \) to \( z \) to \( b \)
- \( b \) grants \( r \) to \( z \) to \( a \)
- \( x \) takes \( r \) to \( z \) from \( a \)
Key Question

- Characterize class of models for which safety is decidable
  - Existence: Take-Grant Protection Model is a member of such a class
  - Universality: In general, question undecidable, so for some models it is not decidable
- What is the dividing line?

Schematic Protection Model

- Type-based model
  - Protection type: entity label determining how control rights affect the entity
    - Set at creation and cannot be changed
  - Ticket: description of a single right over an entity
    - Entity has sets of tickets (called a domain)
    - Ticket is X/r, where X is entity and r right
  - Functions determine rights transfer
    - Link: are source, target "connected"?
    - Filter: is transfer of ticket authorized?
Link Predicate

- Idea: $link_i(X, Y)$ if $X$ can assert some control right over $Y$
- Conjunction of disjunction of:
  - $X/z \in dom(X)$
  - $X/z \in dom(Y)$
  - $Y/z \in dom(X)$
  - $Y/z \in dom(Y)$
  - true

Examples

- Take-Grant:
  \[ link(X, Y) = Y/g \in dom(X) \lor X/t \in dom(Y) \]
- Broadcast:
  \[ link(X, Y) = X/b \in dom(X) \]
- Pull:
  \[ link(X, Y) = Y/p \in dom(Y) \]
Filter Function

- Range is set of copyable tickets
  - Entity type, right
- Domain is subject pairs
- Copy a ticket $\mathbf{X}/r.c$ from $\text{dom} \mathbf{Y}$ to $\text{dom} \mathbf{Z}$
  - $\mathbf{X}/r.c \in \text{dom} \mathbf{Y}$
  - $\text{link} \mathbf{Y}, \mathbf{Z}$
  - $\tau(\mathbf{Y})/r.c \in f_\tau(\tau(\mathbf{Y}), \tau(\mathbf{Z}))$
- One filter function per link function

Example

- $f(\tau(\mathbf{Y}), \tau(\mathbf{Z})) = T \times R$
  - Any ticket can be transferred (if other conditions met)
- $f(\tau(\mathbf{Y}), \tau(\mathbf{Z})) = T \times RI$
  - Only tickets with inert rights can be transferred (if other conditions met)
- $f(\tau(\mathbf{Y}), \tau(\mathbf{Z})) = \emptyset$
  - No tickets can be transferred
Example

- Take-Grant Protection Model
  - $TS = \{ \text{subjects} \}$, $TO = \{ \text{objects} \}$
  - $RC = \{ tc, gc \}$, $RI = \{ rc, wc \}$
  - $link(p, q) = p/t \in dom(q) \lor q/g \in dom(p)$
  - $f(subject, subject) = \{ subject, object \} \times \{ tc, gc, rc, wc \}$

Create Operation

- Must handle type, tickets of new entity
- Relation $cc(a, b)$ [$cc$ for can-create]
  - Subject of type $a$ can create entity of type $b$
- Rule of acyclic creates:
Types

- \( cr(a, b) \): tickets created when subject of type \( a \) creates entity of type \( b \) [\( cr \) for \( create \)-\( rule \)]
- \( B \) object: \( cr(a, b) \subseteq \{ b/r.c \in R | \} \)
  - \( A \) gets \( B/r.c \) iff \( b/r.c \in cr(a, b) \)
- \( B \) subject: \( cr(a, b) \) has two subsets
  - \( cr_P(a, b) \) added to \( A \), \( cr_C(a, b) \) added to \( B \)
  - \( A \) gets \( B/r.c \) if \( b/r.c \in cr_P(a, b) \)
  - \( B \) gets \( A/r.c \) if \( a/r.c \in cr_C(a, b) \)

Non-Distinct Types

\( cr(a, a) \): who gets what?

- \( self/r.c \) are tickets for creator
- \( a/r.c \) tickets for created

\( cr(a, a) = \{ a/r.c, self/r.c | r.c \in R | \} \)
Attenuating Create Rule

\( cr(a, b) \) attenuating if:
1. \( cr_C(a, b) \subseteq cr_P(a, b) \) and
2. \( a/r: c \in cr_P(a, b) \Rightarrow self/r: c \in cr_P(a, b) \)

Example: Owner-Based Policy

- Users can create files, creator can give itself any inert rights over file
  - \( cc = \{ (\text{user}, \text{file}) \} \)
  - \( cr(\text{user}, \text{file}) = \{ \text{file}/r: c \mid r \in RI \} \)
- Attenuating, as graph is acyclic, loop free
Example: Take-Grant

- Say subjects create subjects (type $s$), objects (type $o$), but get only inert rights over latter
  - $cc = \{(s, s), (s, o)\}$
  - $cr_c(a, b) = \emptyset$
  - $cr_P(s, s) = \{s/tc, s/gc, s/rc, s/wc\}$
  - $cr_P(s, o) = \{s/rc, s/wc\}$

- Not attenuating, as no self tickets provided; subject creates subject

Safety Analysis

- Goal: identify types of policies with tractable safety analyses
- Approach: derive a state in which additional entries, rights do not affect the analysis; then analyze this state
  - Called a maximal state