Lecture #3

- Proof of mono-operational decidability result
- Review of Take-Grant rules, structures
- Sharing rights in Take-Grant
- Generated systems
- Theft

Safety Question

- Does there exist an algorithm for determining whether a protection system S with initial state s_0 is safe with respect to a generic right r?
  – Here, "safe" = "secure" for an abstract model

Mono-Operational Commands

- An algorithm exists that will determine whether a given mono-operational protection system with initial state s_0 is safe with respect to a generic right r.

Proof (1)

- Consider minimal sequence of commands (of length m) needed to leak r from system with initial state s_0
  – Identify each command by the type of primitive operation it invokes
- Cannot test for *absence* of rights, so **delete**, **destroy** not relevant
  – Ignore them

Proof (2)

- Reorder sequence of commands so all **create**s come first
  - Can be done because **enter**s require subject, object to have been created
- Commands after these check only for *existence* of right

Proof (3)

- It can be shown (see homework!)
  - Suppose s_1, s_2 created and commands test rights in $A[s_1, o_1], A[s_2, o_2]$
  - Doing the same tests on $A[s_1, o_1]$ and $A[s_1, o_2] = A[s_1, o_2] \cup A[s_2, o_2]$ gives same results
  - Thus all **create**s unnecessary
    - Unless s_0 is empty; then you need one **create**

Proof (4)

- $|S_0|$ number of subjects in s_0
- $|O0|$ number of objects in s_0
- n number of (generic) rights
- In worst case, 1 create, so a total of $(|S_0| + 1)(|O_0| + 1)$ elements
- Thus $m \leq n(|S_0| + 1)(|O_0| + 1) + 1$

Take-Grant Protection Model

- A specific (not generic) system
  - Set of rules for state transitions
- Safety decidable, and in time linear with the size of the system
- Goal: find conditions under which rights can be transferred from one entity to another in the system

System

- ○ objects (files, …)
- ● subjects (users, processes, …)
- ⊗ don't care (either a subject or an object)

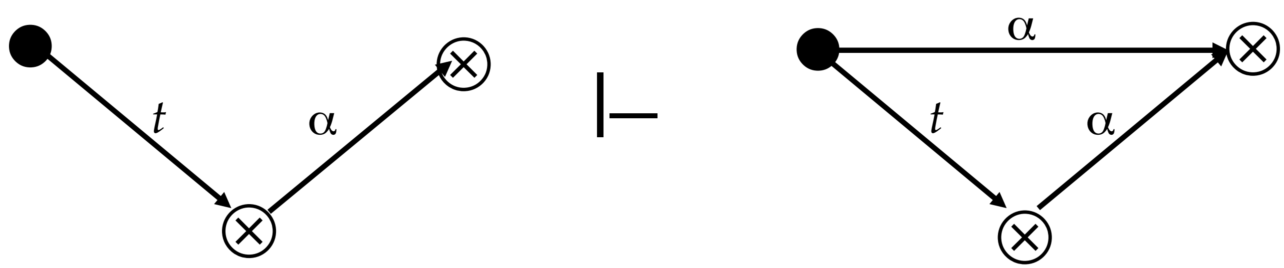
$G \vdash_x G'$ apply a rewriting rule x (witness) to G to get G'

$G \vdash^* G'$ apply a sequence of rewriting rules (witness) to G to get G'

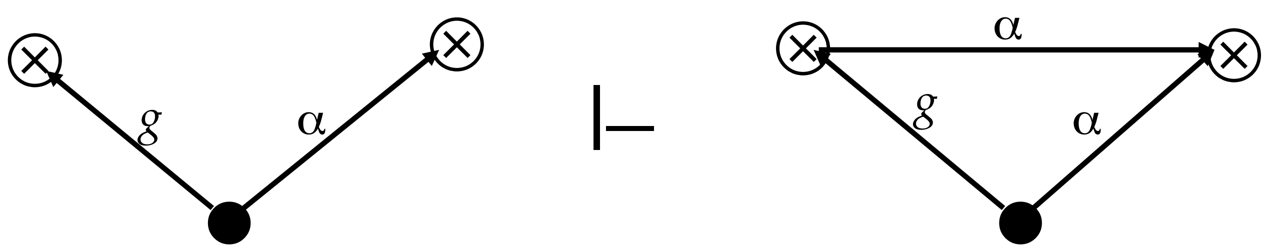
$R = \{ t, g, r, w, \ldots \}$ set of rights

Rules

take



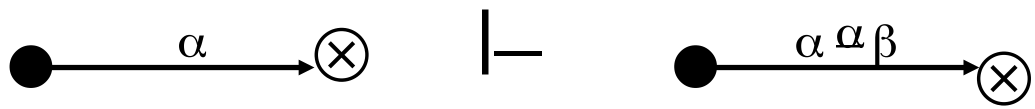
grant



More Rules

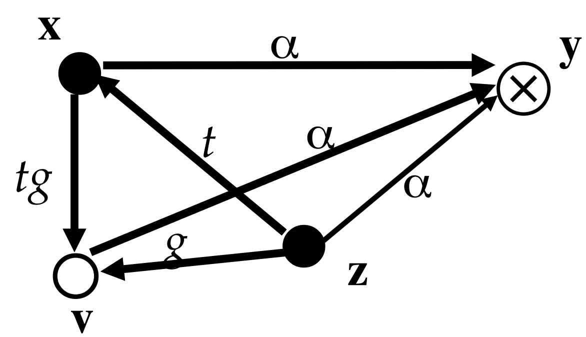
create

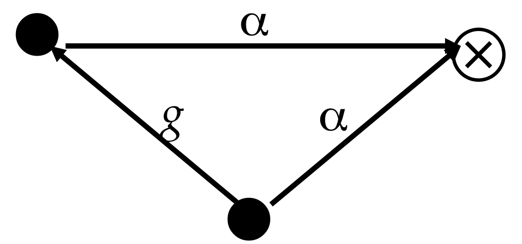
remove



These four rules are called the *de jure* rules

Symmetry





1. **x** creates (*tg* to new) **v**
2. **z** takes (*g* to **v**) from **x**
3. **z** grants (α to **y**) to **v**
4. **x** takes (α to **y**) from **v**

Similar result for grant

Islands

- *tg*-path: path of distinct vertices connected by edges labeled *t* or *g*
  - Call them “tg-connected”
- island: maximal *tg*-connected subject-only subgraph
  - Any right one vertex has can be shared with any other vertex

Initial, Terminal Spans

- *initial span* from **x** to **y**
  - **x** subject
  - *tg*-path between **x**, **y** with word in $\{ \vec{t}^{*}\vec{g} \} \cup \{ \nu \}$
  - Means **x** can give rights it has to **y**
- *terminal span* from **x** to **y**
  - **x** subject
  - *tg*-path between **x**, **y** with word in $\{ \vec{t}^{*} \} \cup \{ \nu \}$
  - Means **x** can acquire any rights **y** has

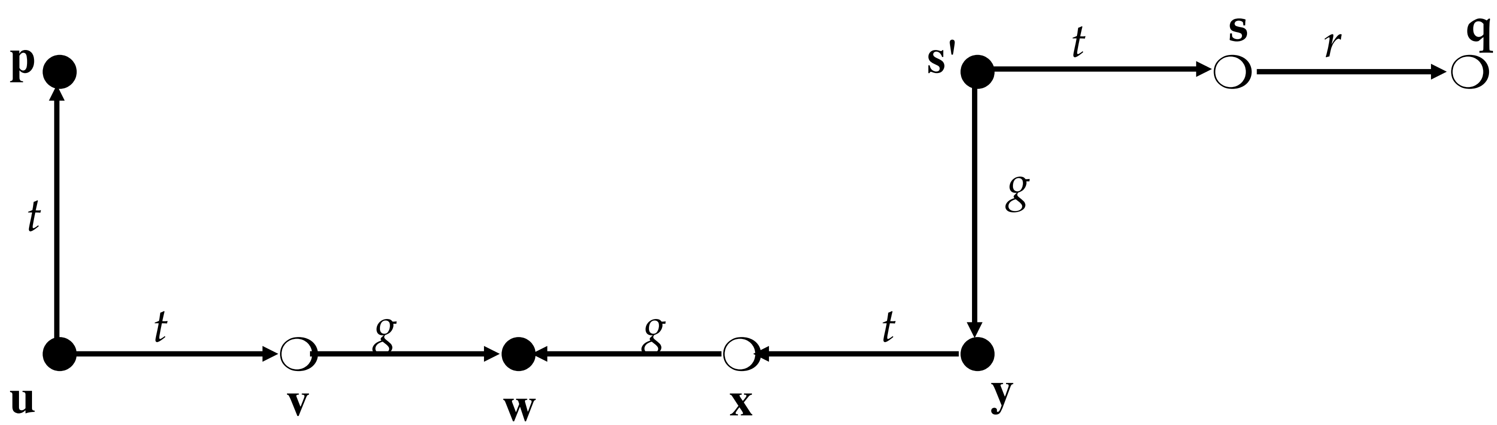
Bridges

- bridge: *tg*-path between subjects **x**, **y**, with associated word in

$$\{\ \vec{t}^*, \vec{t}^*, \vec{t}^*\overleftarrow{g}\,\overleftarrow{t}^*, \vec{t}^*\vec{g}\,\overleftarrow{t}^*\ \}$$

  - rights can be transferred between the two endpoints
  - *not* an island as intermediate vertices are objects

Example



- islands $\{$ **p**, **u** $\}$ $\{$ **w** $\}$ $\{$ **y**, **s'** $\}$
- bridges **u**, **v**, **w**; **w**, **x**, **y**
- initial span **p** (associated word ν)
- terminal span **s's** (associated word $\vec{t}$)

can•share Predicate

Definition:

- *can•share*(r, **x**, **y**, G_0) if, and only if, there is a sequence of protection graphs $G_0, \ldots, G_n$ such that $G_0 \vdash^* G_n$ using only *de jure* rules and in G_n there is an edge from **x** to **y** labeled r.

can•share Theorem

- *can•share*(r, **x**, **y**, G_0) if, and only if, there is an edge from **x** to **y** labeled r in G_0, or the following hold simultaneously:
  - There is an **s** in G_0 with an **s**-to-**y** edge labeled r
  - There is a subject **x′** = **x** or initially spans to **x**
  - There is a subject **s′** = **s** or terminally spans to **s**
  - There are islands $I_1, \ldots, I_k$ connected by bridges, and **x′** in I_1 and **s′** in I_k

Intuition

- **s** has *r* rights over **y**
- **s**′ acquires *r* rights over **y** from **s**
  - Definition of terminal span
- **x**′ acquires *r* rights over **y** from **s**′
  - Repeated application of sharing among vertices in islands, passing rights along bridges
- **x**′ gives *r* rights over **y** to **x**
  - Definition of initial span

Example Interpretation

- ACM is generic
  - Can be applied in any situation
- Take-Grant has specific rules, rights
  - Can be applied in situations matching rules, rights
- Question: what states can evolve from a system that is modeled using the Take-Grant Model?

Take-Grant Generated Systems

- Theorem: G_0 protection graph with 1 vertex, no edges; R set of rights. Then $G_0 \vdash^* G$ iff:
  - G finite directed graph consisting of subjects, objects, edges
  - Edges labeled from nonempty subsets of R
  - At least one vertex in G has no incoming edges

Proof (1)

$\Rightarrow$: By construction; G final graph in theorem

- Let $\mathbf{x}_1, \ldots, \mathbf{x}_n$ be subjects in G
- Let $\mathbf{x}_1$ have no incoming edges

- Now construct G' as follows:
  1. Do "$\mathbf{x}_1$ creates ($\alpha \cup \{ g \}$ to) new subject $\mathbf{x}_i$"
  2. For all $(\mathbf{x}_i, \mathbf{x}_j)$ where $\mathbf{x}_i$ has a rights over $\mathbf{x}_j$, do "$\mathbf{x}_1$ grants (α to $\mathbf{x}_j$) to $\mathbf{x}_i$"
  3. Let β be rights $\mathbf{x}_i$ has over $\mathbf{x}_j$ in G. Do "$\mathbf{x}_1$ removes (($\alpha \cup \{ g \} - \beta$ to) $\mathbf{x}_j$"
- Now G' is desired G

Proof (2)

⇐: Let **v** be initial subject, and $G_0 \vdash^* G$

- Inspection of rules gives:
  - *G* is finite
  - *G* is a directed graph
  - Subjects and objects only
  - All edges labeled with nonempty subsets of *R*
- Limits of rules:
  - None allow vertices to be deleted so **v** in *G*
  - None add incoming edges to vertices without incoming edges, so **v** has no incoming edges

Example: Shared Buffer

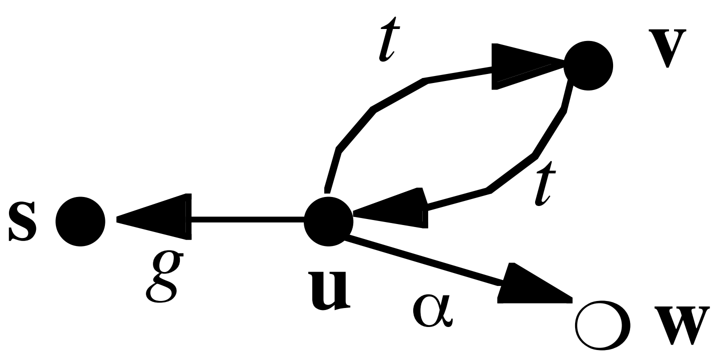
- Goal: **p**, **q** to communicate through shared buffer **b** controlled by trusted entity **s**
  1. **s** creates ({r, w} to new object) **b**
  2. **s** grants ({r, w} to **b**) to **p**
  3. **s** grants ({r, w} to **b**) to **q**

can•steal Predicate

Definition:

- *can•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 ρ_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 ρ_i is ***not*** of the form "**v** grants (r to **y**) to **w**"

Example



- *can•steal*(α, **s**, **w**, G_0):

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*(α, $\mathbf{x}$, $\mathbf{y}$, G_0) if, and only if, the following hold simultaneously:
  a) There is no edge from $\mathbf{x}$ to $\mathbf{y}$ labeled α in G_0
  b) There exists a subject $\mathbf{x}'$ such that $\mathbf{x}' = \mathbf{x}$ or $\mathbf{x}'$ initially spans to $\mathbf{x}$
  c) There exists a vertex $\mathbf{s}$ with an edge labeled α to $\mathbf{y}$ in G_0
  d)can•share(t, $\mathbf{x}'$, $\mathbf{s}$, G_0) holds

Proof (1)

$\Rightarrow$: Assume conditions hold

- **x** subject
  - **x** gets t rights to **s**, then takes α to **y** from **s**
- **x** object
  - *can•share*(t, **x**′, **s**, G_0) holds
  - If **x**′ has no α edge to **y** in G_0, **x**′ takes (α 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 (α to **y**) and grants it to **x**

Proof (2)

$\Leftarrow$: Assume *can•steal*(α, **x**, **y**, G_0) holds

- First two conditions immediate from definition of *can•steal*, *can•share*
- Third condition immediate from theorem of conditions for *can•share*
- Fourth condition: ρ minimal length sequence of rule applications deriving G_n from G_0; i smallest index such that $G_{i-1} \vdash G_i$ by rule ρ_i and adding α from some **p** to **y** in G_i
  – What is ρ_i?

Proof (3)

- Not remove or create rule
  - **y** exists already
- Not grant rule
  - G_i 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 by earlier theorem, there is subject **s′** such that **s′** = **s** or terminally spans to **s**
  - Also, sequence of islands with **x′** $\in I_1$ and **s′** $\in I_n$
- If **s** object, **s′** ≠ **s**. If **s′**, **p** in same island, **p** = **s′**. If not, sequence not minimal; so *can•share*(t, **x**, **s**, G_0) holds
  - Can choose **s′** in same island as **p**

Proof (4)

- If **s** subject, $\mathbf{p} \in I_n$. If $\mathbf{p} \notin G_0$, there is subject **q** for which *can•share*(t, **q**, **s**, G_0) holds
  - $\mathbf{s} \in G_0$ and none of the rules add new labels to incoming edges on existing vertices

As **s** owns a rights to **y** in G_0, two cases. If $\mathbf{s} \neq \mathbf{q}$, replace

s grants (α to **y**) to **q**

with

p takes (α to **y**) from **s**

p takes (g to **q**) from **s**

p grants (α to **y**) to **q**

If $\mathbf{s} = \mathbf{q}$, you only need the first.