Lecture #5

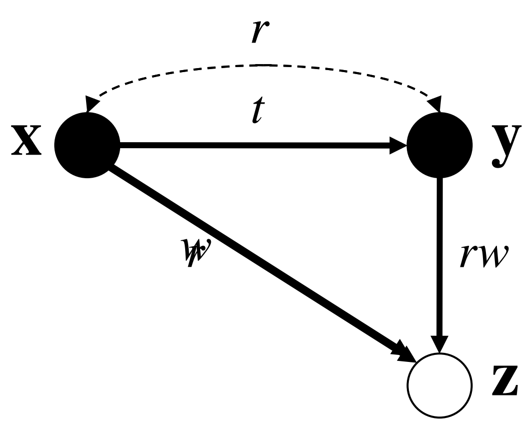
- Knowing in a combined graph
- Basics of Schematic Protection Model

can•know

Definition:

- *can•know*(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 *de jure* or *de facto* rules and in G_n there is an edge from **x** to **y** labeled r.

Example



y creates (*rw* to new) **z**

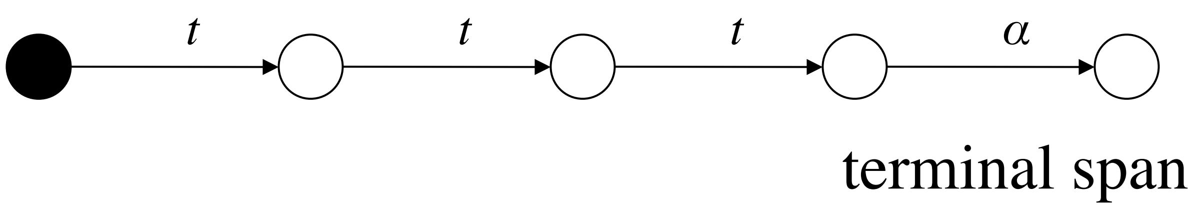
x takes (*r* to **z**) from **y**

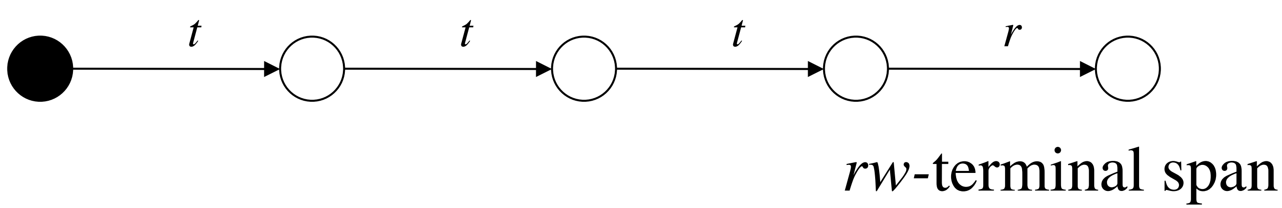
y passes to **x** through **z**

x takes (*w* to **z**) from **y**

y posts to **x** through **z**

Combined Transfers

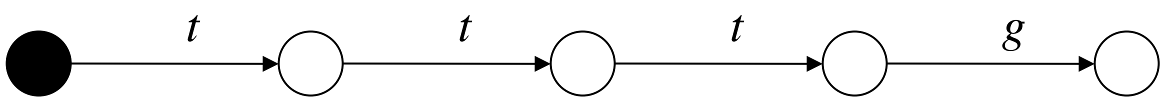




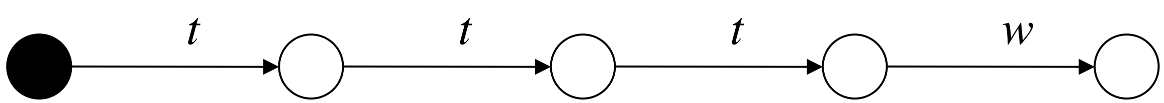
The subject can acquire α rights over the last object

The subject can acquire *r* rights over the last object

Combined Transfers



initial span

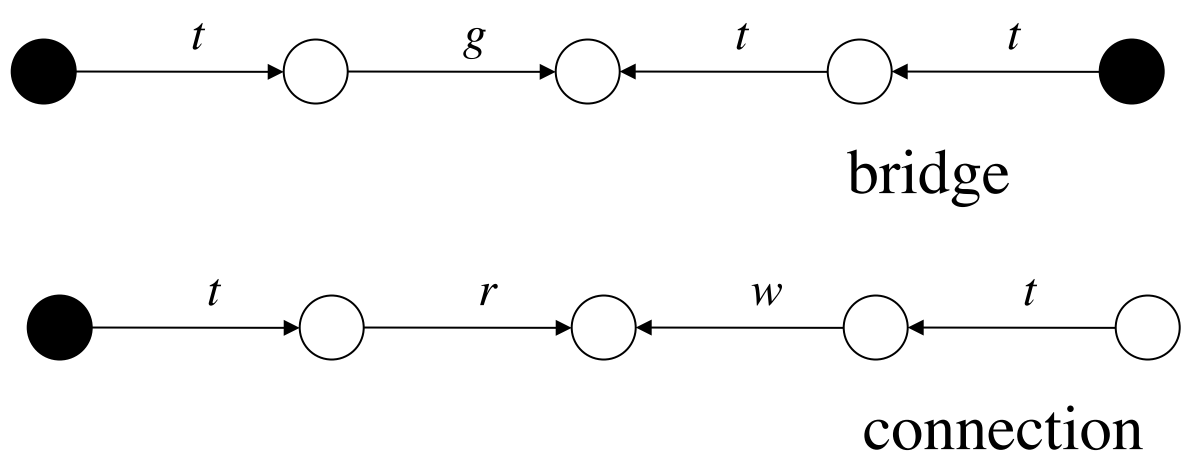


rw-initial span

The subject can acquire *g* rights over the last object

The subject can acquire *w* rights over the last object

Combined Transfers



Just as rights can be transferred over a bridge, information can flow over a connection

Theorem

$can\bullet know(\mathbf{p}, \mathbf{q}, G_0)$ holds if and only if:

(a) $can\bullet share(r, \mathbf{p}, \mathbf{q}, G_0)$ holds, or

(b) there is a sequence of subjects $\mathbf{u_1}, \ldots, \mathbf{u}_n$ such that all of the following are true:

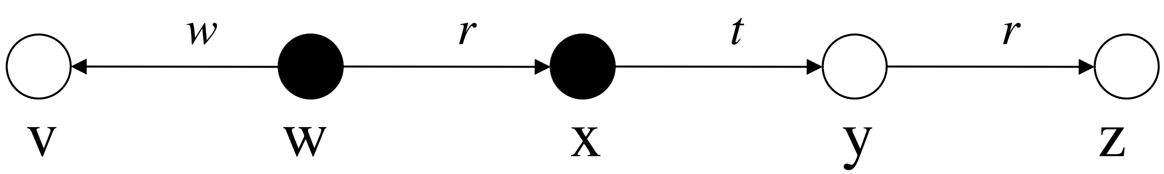
- (i) $\mathbf{p} = \mathbf{u_1}$ or $\mathbf{u_1}$ rw-initially spans to $\mathbf{p}$;
- (ii) $\mathbf{q} = \mathbf{u}_n$ or $\mathbf{u}_n$ rw-terminally spans to $\mathbf{q}$; and
- (iii) for all i, $1 \le i < n$, there is an *rwtg*-path between $\mathbf{u_i}$ and $\mathbf{u_{i+1}}$ with associated word a bridge or connection

Back to Example

can•know($\mathbf{p}$, $\mathbf{q}$, G_0) holds:

- take $n = 2$, $\mathbf{u_1}$ = x, and $\mathbf{u_2}$ = y
  - $\mathbf{p} = \mathbf{u_1}$ or $\mathbf{u_1}$ rw-initially spans to $\mathbf{p}$;
  - $\mathbf{q} = \mathbf{u_2}$ or $\mathbf{u_2}$ rw-terminally spans to $\mathbf{q}$; and
  - there is an *rwtg*-path between $\mathbf{u_1}$ and $\mathbf{u_2}$ with associated word a bridge or connection
    - $\mathbf{u_1}$, $\mathbf{u_2}$ connected with a *t* edge

Final Example



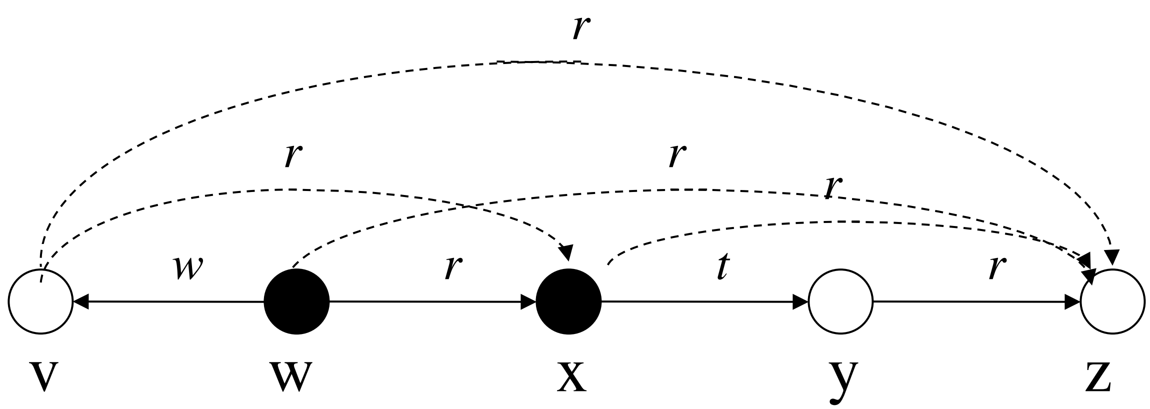
can•share(r, $\mathbf{v}$, $\mathbf{z}$, G_0) is false

- no initial span between **v** and any subject

can•know($\mathbf{v}$, $\mathbf{z}$, G_0) is true

- $\mathbf{u_1} = \mathbf{w}$, $\mathbf{u_2} = \mathbf{x}$
- $\mathbf{u_1}$ *rw*-initially spans to **y**
- $\mathbf{u_2}$ *rw*-terminally spans to **z**
- there is a bridge between $\mathbf{u_1}$ and $\mathbf{u_2}$

Final Example Witness



x takes (*r* to **z**) from **y**

w spies on **z** through **x**

w passes from **z** to **v**

x takes (*r* to **z**) from **y**

w passes from **x** to **v**

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

Examples

- Take-Grant:

  $link(\mathbf{X}, \mathbf{Y}) = \mathbf{Y}/g \in dom(\mathbf{X}) \lor \mathbf{X}/t \in dom(\mathbf{Y})$
- Broadcast:

  $link(\mathbf{X}, \mathbf{Y}) = \mathbf{X}/b \in dom(\mathbf{X})$
- Pull:

  $link(\mathbf{X}, \mathbf{Y}) = \mathbf{Y}/p \in dom(\mathbf{Y})$

Filter Function

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

Example

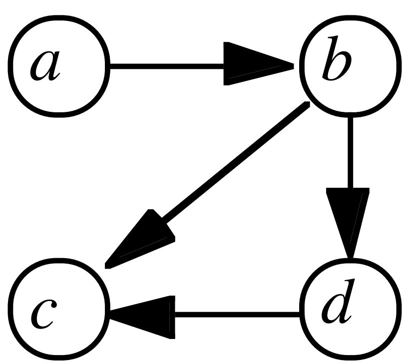
- $f(\tau(\mathbf{Y}), \tau(\mathbf{Z})) = T \times R$
  - Any ticket can be transferred (if other conditions met)
- $f(\tau(\mathbf{Y}), \tau(\mathbf{Z})) = T \times RI$
  - Only tickets with inert rights can be transferred (if other conditions met)
- $f(\tau(\mathbf{Y}), \tau(\mathbf{Z})) = \varnothing$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

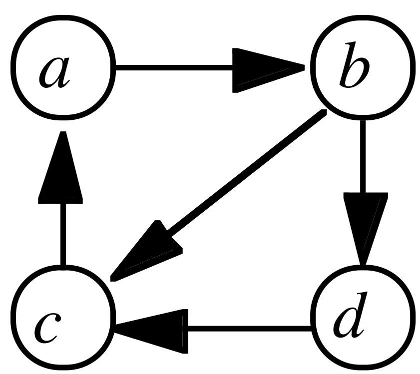
Example

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

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: $cr(a, b) \subseteq \{ b/r{:}c \in RI \}$
  - **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

$cr(a, a) = \{\ a/r{:}c,\ self/r{:}c \mid r{:}c \in R\}$

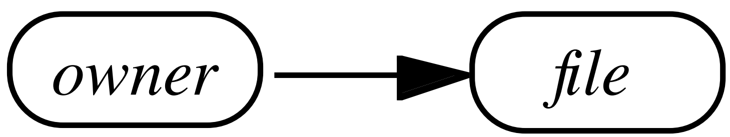
Attenuating Create Rule

$cr(a, b)$ attenuating if:

1. $cr_C(a, b) \subseteq cr_P(a, b)$ and
2. $a/r{:}c \in cr_P(a, b) \Rightarrow self/r{:}c \in cr_P(a, b)$

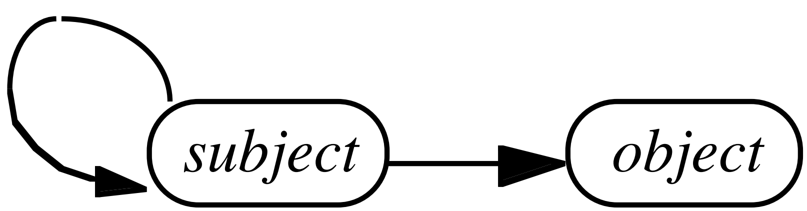
Example: Owner-Based Policy

- Users can create files, creator can give itself any inert rights over file
  - $cc = \{ \ (\ user\ ,\ file\ )\ \}$
  - $cr(user, file) = \{\ file/r{:}c \mid r \in RI\ \}$
- 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
  - $cc = \{ (s, s), (s, o) \}$
  - $cr_C(a, b) = \varnothing$
  - $cr_P(s, s) = \{s/tc, s/gc, s/rc, s/wc \}$
  - $cr_P(s, o) = \{s/rc, s/wc \}$
- 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*

Definitions

- System begins at initial sate
- Authorized operation causes *legal transition*
- Sequence of legal transitions moves system into final state
  - This sequence is a *history*
  - Final state is *derivable* from history, initial state

More Definitions

- States represented by h
- Set of subjects SUB^h, entities ENT^h
- Link relation in context of state h $link^h$
- Dom relation in context of state h dom^h

$path^h(\mathbf{X},\mathbf{Y})$

- $\mathbf{X}$, $\mathbf{Y}$ connected by one link or a sequence of links
- Formally, either of these hold:
  - for some i, $link_i^h(\mathbf{X}, \mathbf{Y})$; or
  - there is a sequence of subjects $\mathbf{X}_0, \ldots, \mathbf{X}_n$ such that $link_i^h(\mathbf{X}, \mathbf{X}_0)$, $link_i^h(\mathbf{X}_n,\mathbf{Y})$, and for $k = 1, \ldots, n$, $link_i^h(\mathbf{X}_{k-1}, \mathbf{X}_k)$
- If multiple such paths, refer to $path_j^h(\mathbf{X}, \mathbf{Y})$

Capacity $cap(path^h(\mathbf{X},\mathbf{Y}))$

- Set of tickets that can flow over $path^h(\mathbf{X},\mathbf{Y})$
  - If $link_i^h(\mathbf{X},\mathbf{Y})$: set of tickets that can be copied over the link (i.e., $f_i(\tau(\mathbf{X}), \tau(\mathbf{Y}))$)
  - Otherwise, set of tickets that can be copied over *all* links in the sequence of links making up the $path^h(\mathbf{X},\mathbf{Y})$
- Note: all tickets (except those for the final link) *must* be copyable

Flow Function

- Idea: capture flow of tickets around a given state of the system
- Let there be m $path^h$s between subjects **X** and **Y** in state h. Then *flow function*

$$flow^h: SUB^h \times SUB^h \rightarrow 2^{T \times R}$$

is:

$$flow^h(\mathbf{X},\mathbf{Y}) = \bigcup_{i=1,\dots,m} cap(path_i^h(\mathbf{X},\mathbf{Y}))$$

Properties of Maximal State

- Maximizes flow between all pairs of subjects
  - State is called *
  - Ticket in *flow**(**X**,**Y**) means there exists a sequence of operations that can copy the ticket from **X** to **Y**
- Questions
  - Is maximal state unique?
  - Does every system have one?