An Algebraic Approach to the Analysis of Constrained Workflow Systems

Workshop on Foundations of Computer Security, 12 July 2004

Jason Crampton
Information Security Group, Royal Holloway, University of London

What is a workflow system?

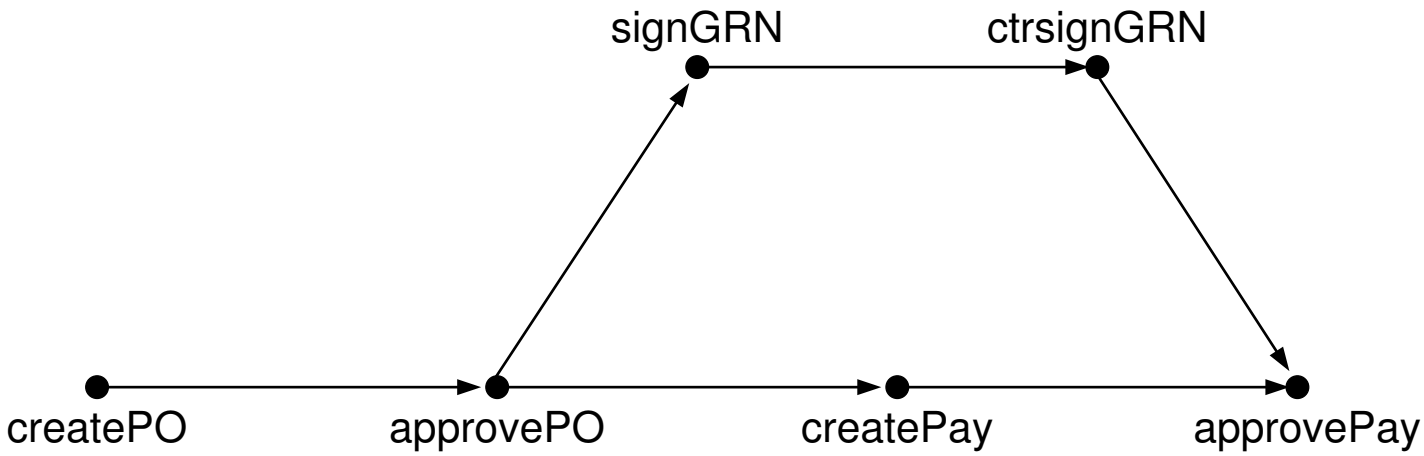
A representation of an organizational or business process

Typically specified as a set of tasks and a set of dependencies between the tasks

Users of the system are authorized to perform certain tasks

Role-based access control is a popular choice for enforcing authorization requirements in workflow systems

Example – A purchase order workflow system



Constrained workflow systems

It may be necessary to impose constraints on who can perform a task given that a prior task has been performed by a particular individual

These constraints may exist to prevent fraud or to implement a particular feature of a business process

- We don't want the same person who creates the purchase order to approve the purchase order
- We want the GRN to be signed by the person that ordered the goods

These constraints supplement the security policy defined by authorization information

The problem defined

Design an abstract machine (reference monitor) that decides whether a user request to execute a task in a workflow instance should be granted

A necessary condition for the request to be granted is that the user has appropriate authorization

Granting the request **must not**

- cause the violation of any constraint
- result in a workflow instance that is unsatisfiable

Motivation

Existing research into constraints in workflow systems

- relies on ad hoc methods
- assumes an underlying computational model
- assumes an underlying access control model

I think the subject can (and should) be approached using

- as few assumptions as possible
- as simple and expressive a framework as possible
- mathematical structures and techniques

Outline of talk

- Basic concepts and definitions
- Combining constraints
- Satisfiability
- Building a reference monitor for workflow systems
- Conclusion

A model for constrained workflow systems (1)

A *workflow specification* is a partially ordered set of tasks T

- If $\mathsf{t} < \mathsf{t}'$ then t must be performed before t' in any instance of the workflow

A *workflow authorization schema* is a pair (T, A), where $A \subseteq \mathsf{T} \times U$

- If $(\mathsf{t}, u) \in A$ then u is authorized to perform t in any instance of the workflow
- Typically A will not be given explicitly and will be derived from other information (role assignments for example)

A model for constrained workflow systems (2)

An *entailment constraint* has the form $(D, (\mathsf{t}, \mathsf{t}'), \rho)$, where $\rho \subseteq U \times U$ and $\mathsf{t} \not\geq \mathsf{t}'$

- $D \subseteq U$ is called the *domain* of the constraint
- ρ is used to determine which users can perform t' once t has been performed by a user in D
  - If $u \in D$ performs t and $u' \in U$ performs t', then $(D, (\mathsf{t}, \mathsf{t}'), \rho)$ is *satisfied* iff $(u, u') \in \rho$

A *constrained workflow authorization schema* is a triple (T, A, C), where C is a set of entailment constraints

Examples

Define

$$\mathsf{0}' = \{(u, v) : u, v \in U, u \neq v\}$$

$(D, (\mathsf{t}, \mathsf{t}'), \mathsf{0}')$ is a separation of duty constraint

- if $u \in D$ performs t, then u cannot perform t'
- $(U, (\texttt{createPO}, \texttt{apprPO}), \mathsf{0}')$

Examples

Define

$$0' = \{(u, v) : u, v \in U, u \neq v\} \qquad 1' = \{(u, u) : u \in U\}$$

$(D, (\mathsf{t}, \mathsf{t}'), 0')$ is a separation of duty constraint

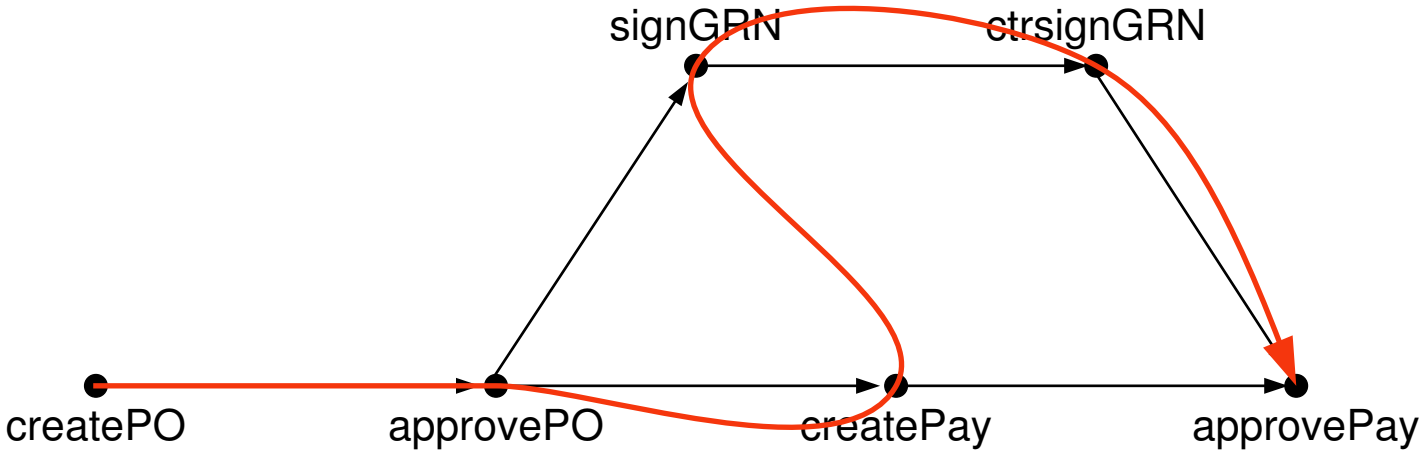
- if $u \in D$ performs t, then u cannot perform t′
- $(U, (\texttt{createPO}, \texttt{apprPO}), 0')$

$(D, (\mathsf{t}, \mathsf{t}'), 1')$ is a "binding of duty" constraint

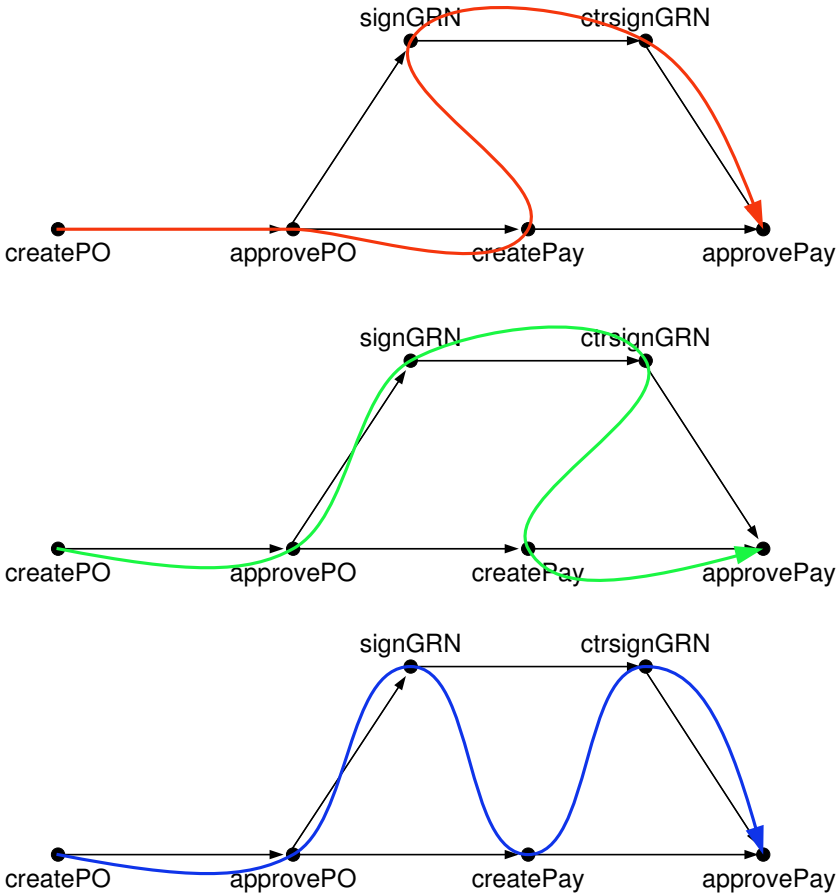
- if $u \in D$ performs t, then u must perform t′
- $(U, (\texttt{createPO}, \texttt{signGRN}), 1')$

Linear extensions

A linear extension of a workflow specification T represents a possible sequence of execution of the tasks in T



Linear extensions



Execution schedules

An execution schedule for (T, A, C) is a pair (L, α), where

- L is a linear extension of T
- $\alpha : \mathsf{T} \to U$ assigns users to tasks

such that

- each user that performs a task in the schedule appropriately authorized

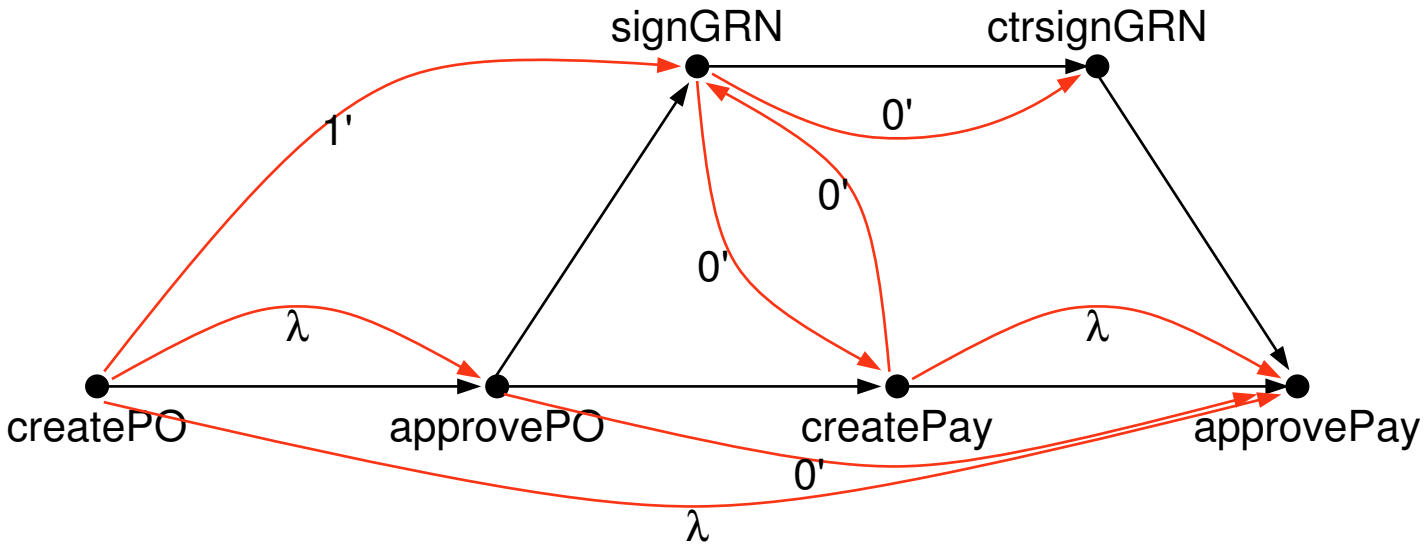
$$\forall \mathsf{t} \in \mathsf{T},\ (\mathsf{t}, \alpha(\mathsf{t})) \in A$$

- all constraints are satisfied

$$\forall (D, (\mathsf{t}, \mathsf{t}'), \rho),\ \alpha(\mathsf{t}) \in D \text{ implies } (\alpha(\mathsf{t}), \alpha(\mathsf{t}')) \in \rho$$

The entailment graph

Each constraint is visualized as a labelled edge in a directed graph whose nodes are tasks



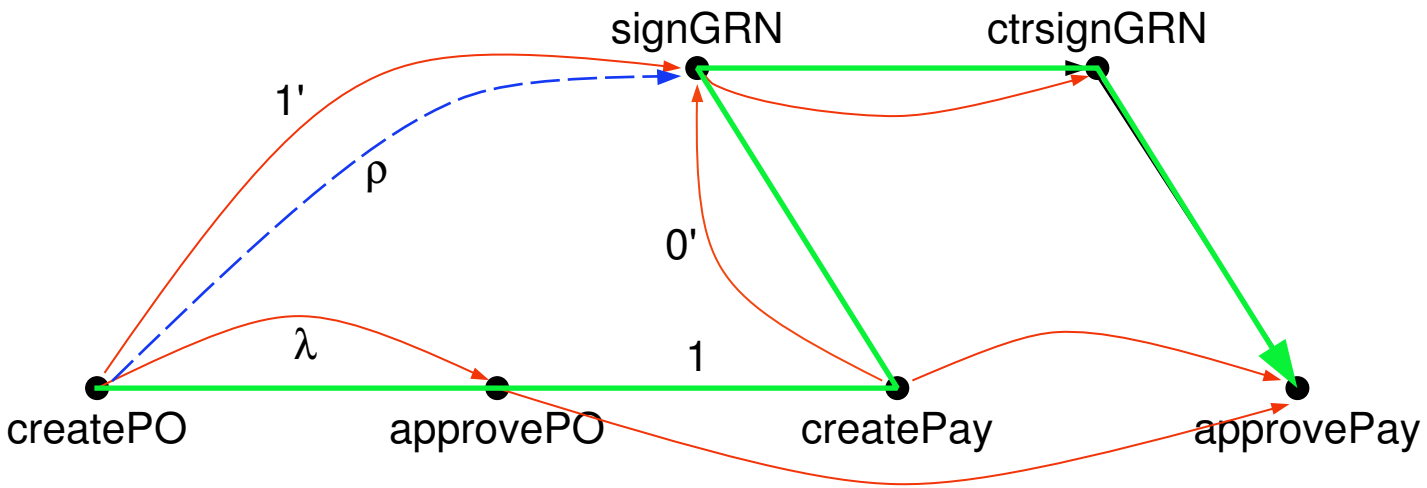
$(u, u') \in \lambda$ if u is less senior than u'

Paths in the entailment graph

The ordering on the tasks can be modelled using the relation $1 = U \times U$

Paths represent composition of constraints

Multiple paths between two nodes can be reduced to a single constraint; $\rho = \lambda 1 0' \cap 1'$



Manipulating constraints

- Make $D = U$ for all constraints

$$\rho' = \rho \cup \{(u, u') : u \notin D, u' \in U\}$$

- Compute single relation for each pair of tasks $(\mathrm{t}, \mathrm{t}')$

$$\rho'' = \rho \cap \rho'$$

- Compose constraint $(U, (\mathrm{t}, \mathrm{t}'), \rho)$ with constraint $(U, (\mathrm{t}', \mathrm{t}''), \rho')$

$$\rho'' = \{(u, u'') : \exists u' \in U, (u, u') \in \rho, (u', u'') \in \rho'\}$$

- Overlay authorization information with constraints

$$\rho' = \rho \cap \{(u, u') : (\mathrm{t}, u), (\mathrm{t}', u') \in A\}$$

Consequences

For any constrained workflow authorization schema (T, A, C), there exists a schema $(\mathsf{T}, \mathsf{T} \times U, C^*)$ such that

- For all $\mathsf{t}, \mathsf{t}' \in \mathsf{T}$ such that $\mathsf{t} \not\geqslant \mathsf{t}'$ there exists a constraint $c \in C^*$ of the form $(U, (\mathsf{t}, \mathsf{t}'), \rho)$
- Every execution schedule of (T, A, C) is an execution schedule of $(\mathsf{T}, \mathsf{T} \times U, C^*)$ and vice versa

Interesting questions

Is a constrained workflow authorization schema satisfiable?

Is a particular instance of a workflow schema satisfiable?

Given a satisfiable workflow authorization schema, is it possible to design a reference monitor so that every instance of that schema is satisfiable?

An observation

Suppose that (T, A, C) is a constrained workflow authorization schema and that a minimal element $\mathsf{t}_0 \in \mathsf{T}$ is performed by u

This gives rise to a new CWAS (T, A', C), where

$$A' = \{(\mathsf{t}_0, u)\} \cup \{(\mathsf{t}, u) \in A : \mathsf{t} \neq \mathsf{t}_0\}$$

More generally, let I be an instance of (T, A, C), where I is an order ideal in T, and let $I(\mathsf{t})$ denote the user who performed task t

Then $(\mathsf{T}, A|I, C)$ is a CWAS, where

$$A|I = \{(\mathsf{t}, I(\mathsf{t})) : \mathsf{t} \in I\} \cup \{(t, u) \in A : \mathsf{t} \notin T \setminus I\}$$

Consequences

The same decision procedure can be used to answer the questions

- Is a CWAS satisfiable?
- Is an instance of a CWAS satisfiable?

A reference monitor that guarantees every workflow instance completes must check that

- if the request were to be granted, the resulting CWAS is satisfiable
- (the request is authorized and does not violate any constraint)

Two strategies

Compute the "closure" of the entailment graph

- Not acyclic
- Difficult to distinguish between walks and paths
- Problem is NP-complete

Enumerate every linear extension

- Treat each linear extension as a workflow schema
- Polynomial in the number of tasks
- Form single workflow schema (taking "intersection" of schemata for linear extensions)

A reference monitor

Let I be an instance of W = (T, A, C) and let $I \cup \{\mathrm{t}\}$ be an order ideal in T

A request by u to execute t in this instance of W is granted by a completion compliant reference monitor if

- there exists an execution schedule for W|I such that u executes t and
- there exists an execution schedule for W|$(I \cup \{(\mathrm{t}, u)\})$

In other words, we simply run the algorithm for the workflow W|$(I \cup \{(\mathrm{t}, u)\})$ *before* granting the request (t, i, u)

If the request is granted, the next request must be evaluated for the workflow W|$(I \cup \{\mathrm{t}, u\})$

Advantages of this approach

Independent of underlying computational model and access control model

- Has generality that other approaches lack
- Can be implemented in a variety of different ways
- Uniform treatment of constraints (many existing approaches are ad hoc)

Simple and rigorous

- Design of reference monitor is simpler
- Overall understanding of mechanisms is improved
- Computational complexity of reference monitor is reduced

Future work

Main priority is to develop model to incorporate multiple instances of tasks

- The specification is a poset T and a function $f : \mathsf{T} \to \mathbb{N}$ indicating the number occurrences of each task
- A constraint still has the form $((D, (\mathsf{t}, \mathsf{t}'), \rho)$, but t may equal t'
  - A task t may be repeated $f(\mathsf{t})$ times but must be performed by the same (different, etc.) user