Consistency of Task-Based Authorization Constraints in Workflow System

Kaijun Tan: UPenn
Jason Crampton: U. of London
Carl Gunter: UPenn
Introduction

- Workflow: consists a sequence of tasks that are organized to facilitate some business process specification
  - Purchase order processing
  - Processing tax refunds
Introduction

- Security Challenge
  - Administration of security-relevant information
  - Specification of authorization policies and constraints
- Solution
  - RBAC
  - Constraints
Introduction

- **Constraints**: restriction on the authorization
  - Separation of Duty
  - Binding of Duties
- **Challenge for constraints in workflow system**
  - Consistency
Goal

- Help workflow designers to define a sound constrained workflow authorization schema such that
  - each workflow instance can complete
  - each user or role authorized to perform a task will have an opportunity to perform that task in some workflow instance
Related Work

- E. Bertino et al: TISS 99
  - Role planning and user planning
    + Constraints are consistent
    - Complicated

- Wainer et al: IJ CIS 03
  - Override constraints when workflow can not be completed
    + Completing workflow instance
    - Weaken security
Our Contribution

- Give explicit definition of constraint workflow authorization schema
- Develop simple specification of authorization constraints
- Define consistency rules which can be checked efficiently
Workflow Example

- Producing payments for tax refunds
  - $t_1$: a clerk prepares a check for a tax refund
  - $t_2$: the check is approved or denied by two different managers
  - $t_3$: A third manager makes a final decision based on the decisions made in $t_2$
  - $t_4$: a clerk issues or voids the check based on the decision of $t_3$
Constraints

- $c_1$: The two instances of $t_2$ should be performed by different users
- $c_2$: $t_2$ and $t_3$ should be performed by different users
- $c_3$: $t_1$ and $t_4$ should be performed by different users
- $c_4$: $t_2$ must be performed by a role that is more senior than the role that performed $t_1$ unless $t_1$ and $t_2$ are performed by GM
- $c_5$: $t_1$ and $t_2$ must be performed by different users
- $c_6$: At least three roles should perform the workflow instance
Constraint Workflow Authorization Scheme

- \((T, \leq, \rho)\), where \(T\) is a (partially ordered) set of tasks and \(\rho\) is a function from \(T\) to the set of natural numbers indicating the number of occurrences of each task in the workflow.
- Role hierarchy: \((R, \leq)\)
- User-role association: \(UA \subseteq U \times R\)
- Task-permission association: \(PA \subseteq T \times R\)
- Constraint-task association: \(PC \subseteq C \times T\)
- Task instance: \((t_i, u_i, r_i)\)
- Workflow instance: \([(t_1, u_1, r_1), \ldots, (t_n, u_n, r_n)]\)
Example

(a) Workflow specification

(b) Role hierarchy

(c) Task-role assignment relation $PA$

<table>
<thead>
<tr>
<th>Task</th>
<th>Role</th>
</tr>
</thead>
<tbody>
<tr>
<td>$t_1$</td>
<td>$RC$</td>
</tr>
<tr>
<td>$t_2$</td>
<td>$RM$</td>
</tr>
<tr>
<td>$t_3$</td>
<td>$RM$</td>
</tr>
<tr>
<td>$t_4$</td>
<td>$RC$</td>
</tr>
</tbody>
</table>

(d) User-role assignment relation $UA$

<table>
<thead>
<tr>
<th>User</th>
<th>Role</th>
</tr>
</thead>
<tbody>
<tr>
<td>Alice</td>
<td>$RC$</td>
</tr>
<tr>
<td>Bob</td>
<td>$RM$</td>
</tr>
<tr>
<td>Carol</td>
<td>$RM$</td>
</tr>
<tr>
<td>Dave</td>
<td>$RC$</td>
</tr>
<tr>
<td>Eve</td>
<td>$GM$</td>
</tr>
<tr>
<td>Fred</td>
<td>$TM$</td>
</tr>
</tbody>
</table>
Constraint-Task assignment PC

- $PC \subseteq C \times T$ where $C = \{c_1, c_2, c_3, c_4, c_5, c_6\}$
  - $c_2$: $t_2$ and $t_3$ should be performed by different users
    - $(c_2, t_3) \in PC$
  - $c_1$: The two instances of $t_2$ should be performed by different users
    - $(c_1, t_2) \in PC$
  - $c_6$: At least three roles should perform the workflow instance
    - $(c_6, t_1) \in PC, (c_6, t_2) \in PC, (c_6, t_3) \in PC, (c_6, t_4) \in PC$
Constraints Specification

- Entailment constraints
  - Execution of t is constrained by the execution of t’ where t’ < t
  - *Entailed task* of t’ : t
  - \((ur, t', E, pred)\)
    - \(ur \in \{U, R\}\)
    - \(E \subseteq U(t')\) if \(ur = u\) or \(E \subseteq R(t')\) if \(ur = r\)
    - \(pred \in \{=, \neq, <, >, \leq, \geq\}\)
    - \(pred(t_u, t_{t'}) = true\) if \(ur = u\) or \(pred(r_u, r_{t'}) = true\) if \(ur = r\)

- For example, \(c_2\): t_2 and t_3 should be performed by different users
  - \(c_2 = (u, t_2, U(t_2), \neq)\)
Cardinality Constraints

- **Local cardinality constraints**
  - Impose restrictions on the number of users to execute a task
    - \( c_1 \): The two instances of t2 should be performed by different users
    - \( c_1 = (2,2) \)

- **Global cardinality constraints**
  - Impose restrictions on the number of roles to execute a set of tasks
    - \( c_6 \): At least three roles should perform the workflow instance
    - \( c_6 = (T,3) \)
Consistency

- A constrained authorization schema 
  \(((T,\leq,\rho),PA,PC)\) is \textit{sound} if
  - All \((c,t) \in PC\) is well-formed
    - A constraint-task pair is \textit{well-formed if}
  - For all \(u \in U(t)\) and all \(r \in R(t)\), there is a successful workflow instance in which \(t\) completes with \(u\) or \(r\)
Constraint Interaction

- Two constraint-task pairs *interplay* if
  - The constraints are assigned to the same task
    - E.g., (((2,2), t₁), ((r, t₁, R(t₁), '), t₂)
  - The constraints are assigned to a task and its entailed task
    - E.g., (((u, t', U(t'), '), t) and (((2, 2), t') where t is an entailed task of t'
  - The constraints are assigned to a set of restricted tasks in T'
    - E.g., for global cardinality constraint (T', nₜ), ((T', nₜ), t) and ((T', nₜ), t') where t, t' ∈ T'
Interplaying Sets

- **Affected tasks in CT**
  - $(((2,2), t_2), ((r, t_1, R(t_1), <), t_2))$: \{t_2\}
  - $(((u, t', U(t'), \neq), t), ((2, 2), t'))$: \{t\}

- **Non-interplay set SC**
  - any $(c, t)$ in SC does not belong to any CT or they are in some CT but $t$ is not an affected task of that CT

- Any $(c, t) \in SC$ should be *self-consistent*

- Any CT should be *inter-play-consistent*
Interplay consistency

- Self-consistent or inter-play consistent
  - Consistency rules for different constraint-task pairs
    - E.g., for \( c=(u, t', E, \text{pred}) \in SC \), then for any \( \alpha \in E \), \( U(t|t', \alpha, c) \neq \emptyset \) and \( \bigcup_{\alpha \in U(t')} U(t|t', \alpha, c) = U(t) \)
  - Computations required
    - User set processing
Main Result

- **Theorem:** a constrained workflow authorization schema \(((T, \leq, \rho), PA, PC)\) is sound if and only if all constraint-task pairs in SC are self-consistent and all the inter-play sets are inter-play-consistent
Example

- Constraint Inconsistency of our example
- E.g. $CT = \{(c_4, t_2), ((2, 2), t_2)\}$ inter-play
  - $c_4$: $t_2$ must be performed by a role that is more senior than the role that performed $t_1$ unless $t_1$ and $t_2$ are performed by GM
  - $c'_4 = (r, t_1, R(t_1), <), c''_4 = (r, t_1, GM, =)$
  - If Bob or Carol executes $t_1$, then the workflow instance cannot complete at $t_2$
Example

(a) Workflow specification

(b) Role hierarchy

<table>
<thead>
<tr>
<th>Task</th>
<th>Role</th>
</tr>
</thead>
<tbody>
<tr>
<td>t₁</td>
<td>RC</td>
</tr>
<tr>
<td>t₂</td>
<td>RM</td>
</tr>
<tr>
<td>t₃</td>
<td>RM</td>
</tr>
<tr>
<td>t₄</td>
<td>RC</td>
</tr>
</tbody>
</table>

(c) Task-role assignment relation $PA$

<table>
<thead>
<tr>
<th>User</th>
<th>Role</th>
</tr>
</thead>
<tbody>
<tr>
<td>Alice</td>
<td>RC</td>
</tr>
<tr>
<td>Bob</td>
<td>RM</td>
</tr>
<tr>
<td>Carol</td>
<td>RM</td>
</tr>
<tr>
<td>Dave</td>
<td>RC</td>
</tr>
<tr>
<td>Eve</td>
<td>GM</td>
</tr>
<tr>
<td>Fred</td>
<td>TM</td>
</tr>
</tbody>
</table>

(d) User-role assignment relation $UA$
Conclusion

- Give a solution to constraints consistency problem in workflow system
  - Simple constraints specification
  - Efficient consistency rules checking