Towards Automatic Update of Access Control Policy

Jinwei Hu, Yan Zhang, Ruixuan Li

Huazhong University of Science and Technology, Wuhan, China
University of Western Sydney, Sydney, Australia

jwhu@hust.edu.cn
Contents

• Motivations and Background
• Key Questions
• Ideas
• Conclusions
Contents

• Motivations and Background
• Key Questions
• Ideas
• Conclusions
Motivations - Why Update?

• Misconfigurations [SACMAT’08, USENIX SEC’10]

• Permission Assignment
  - A new user joins
  - Task assignment

• Property satisfactions [TISSEC]

• Requirement dynamics [CACM]
Workflow of manual update

specify update constraints

observe the system and update constraints

perform some operations

check system and constraints

constraints violated?

yes

update achieved?

yes

Is the update achievable at all?

no

no

give up?

yes

undo operations

Are all changes necessary?

no

no

yes

end
Background - RBAC Systems

• Role-based access control
Contents

• Motivations and Background
• Key Questions
• Ideas
• Conclusions
Key Questions

• **Q1: What is the update objective?**
  - Assign \{p_5, p_8, p_9\} via \{r_1, r_2, r_3, r_4, r_5, r_6\}
Key Questions

- Q1: What is the update objective?
- Q2: Who is to implement the update?
  - Different administrators come with different power.
  - Interactions/dependencies among administrators.
Key Questions

• Q1: What is the update objective?
• Q2: Who is to implement the update?
• Q3: What is the system behavior after update?
  - Can users still perform their works?
Consideration of Q3

• Users’ permissions vary within range $[\text{lower bound, upper bound}]$

- transparency to users
- maintain access control system functions smoothly
Key Questions

• Q1: What is the update objective?
• Q2: Who is to implement the update?
• Q3: What is the system behavior after update?
• Q4: What are the tolerable changes to roles and role hierarchies?
Consideration of Q4

• Role definitions
  - in terms of permissions, e.g., student = \{use_printer, use_lab, ...\}

• Top-down
  - Business meanings, semantics

• Bottom-up
  - role engineering/mining

• Role definitions change as needed? No change at all?
Key Questions

• Q1: What is the update objective?
• Q2: Who is to implement the update?
• Q3: What is the system behavior after update?
• Q4: What are the tolerable changes to roles and role hierarchies?
• Q5: Is an update optimal (minimal)?
Consideration of Q5

Which update is better, s1 or s2?
Contents

• Motivations and Background
• Key Questions
• Ideas
• Conclusions
Update specification

\[ \text{update} \]
\[ \text{make } P = \{p_5, p_8, p_9\} \text{ available via } T = \{r_1, r_2, r_3, r_4, r_5, r_6\} \]
\[ \text{with} \]
\[ \text{administrators } \text{admin}_1, \text{admin}_2; \]
\[ \text{user-permission constraints} \]
\[ (u_1, \text{no-less-than } \{p_1\}, \text{no-more-than } \{p_1, p_3, p_4\}), \]
\[ (u_2, \text{no-less-than } \{p_1, p_3, p_4, p_5\}, \text{no-more-than } \{p_1, p_3, p_4, p_5\}), \]
\[ (u_3, \text{no-less-than } \{p_3, p_4, p_5\}, \text{no-more-than } \{p_3, p_4, p_5, p_6, p_8\}), \]
\[ (u_4, \text{no-less-than } \{p_7, p_8, p_9\}, \text{no-more-than } \{p_3, p_5, p_6, p_7, p_8, p_9\}); \]
\[ \text{restricted-role constraints} \]
\[ (r_4, \text{no-less-than } \{p_6, p_7\}, \text{no-more-than } \{p_6, p_7, p_8, p_9\}), \]
\[ (r_8, \text{no-less-than } \{p_5, p_6\}, \text{no-more-than } \{p_5, p_6\}); \]
\[ \text{role-hierarchy} = \{(r_2, r_8), (r_3, r_7)\}; \]
\[ \text{minimal}; \]
Model Checking

System Property

Property holds.

Property fails;
A counter-example is generated.
Updating via Model Checking

RBAC System

Property: Requested state is *never* reachable.

Model Checking

- Property holds.
- Property fails; A counter-example is generated.

update achievable?

- No. Requested state is *not* never reachable.
- Yes. Requested state is *not* never reachable, and can be constructed from the counter-example.
Overview

Update request → Transformer → simplified request → Translator

NuSMV Programs

Checking results → Update Constructor ← Reports

← NuSMV
Problems

- State explosion problem
- Memory crash
The Idea of Minimal Update

- original state
- qualified states
- other states

s1 s2 s3

Updating algorithm

difference
The Idea of Minimal Update

original state  qualified states  other states

s1  ...  s2  ...  s3

Updating algorithm

difference
The Idea of Minimal Update

original state  qualified states  other states

s1  s2  s3

Updating algorithm

difference
The Idea of Minimal Update

- original state
- qualified states
- other states

Updating algorithm

No update report
Contents

• Motivations and Background
• Key Questions
• Ideas
• Conclusions
Conclusions

• A tool that accepts and answers high-level update requests.
• Experiments (synthesized data)
• Future work
  - Full administrative model
  - Composition (sequence of update requests)
Thank you!