Extended ReBAC Administrative Models with Cascading Revocation and Provenance Support

Yuan Cheng[1, 2], Khalid Bijon[2], and Ravi Sandhu[1]
Institute for Cyber Security, UTSA[1]
MosaixSoft, Inc.[2]

21st ACM Symposium on Access Control Models and Technologies
June 6-8, 2016, Shanghai, China

ReBAC

- "… a new paradigm of access control needs to be developed that is based on interpersonal relationships …" -- [Gates 2007]

- Relationship-based Access Control (ReBAC) determines access in terms of the relationships among users and resources

- Inspired by the rapid emergence of online social networks

- Exemplary work includes:
  - [Carminati 2009a, 2009b]
  - [Fong 2009]
  - [Fong 2011a, 2011b, Bruns 2012]
  - [Cheng 2012a, 2012b, 2014]
  - [Crampton 2014, Stoller 2015, Rizvi 2015, Crampton 2016]

Administrative ReBAC

- Demand for an appropriate administrative model
  - Dynamic and decentralized nature of OSNs
  - Multiple owners and administrators
  - Proper control on adding and removing of entities, relationships, and policies
- Use ReBAC itself to manage ReBAC
  - Economy of mechanism
  - Prior success of using Role-based Access Control (RBAC) to manage RBAC

RPPM [Crampton 2014]

- J. Crampton and J. Sellwood, *Path Conditions and Principal Matching: A New Approach to Access Control*, **SACMAT 2014**.

- RPPM: relationships, paths, and principal-matching
- Combines UNIX access control model, ReBAC, and RBAC
- Path Condition
  - Bind requests to principals
- Principal Matching
  - Replace a path between entities with a single edge labelled by a principal

RPPM^2 [Stoller 2015]

- S. Stoller, *An Administrative Model for Relationship-Based Access Control*, **DBSec 2015**.

- RPPM^2: RPPM Modified

- Administrative Model
  - Add and Delete Edges/Entities/Authorization Rules
    - The administration of authorization rules is considered the most challenging
  - Economy of mechanism

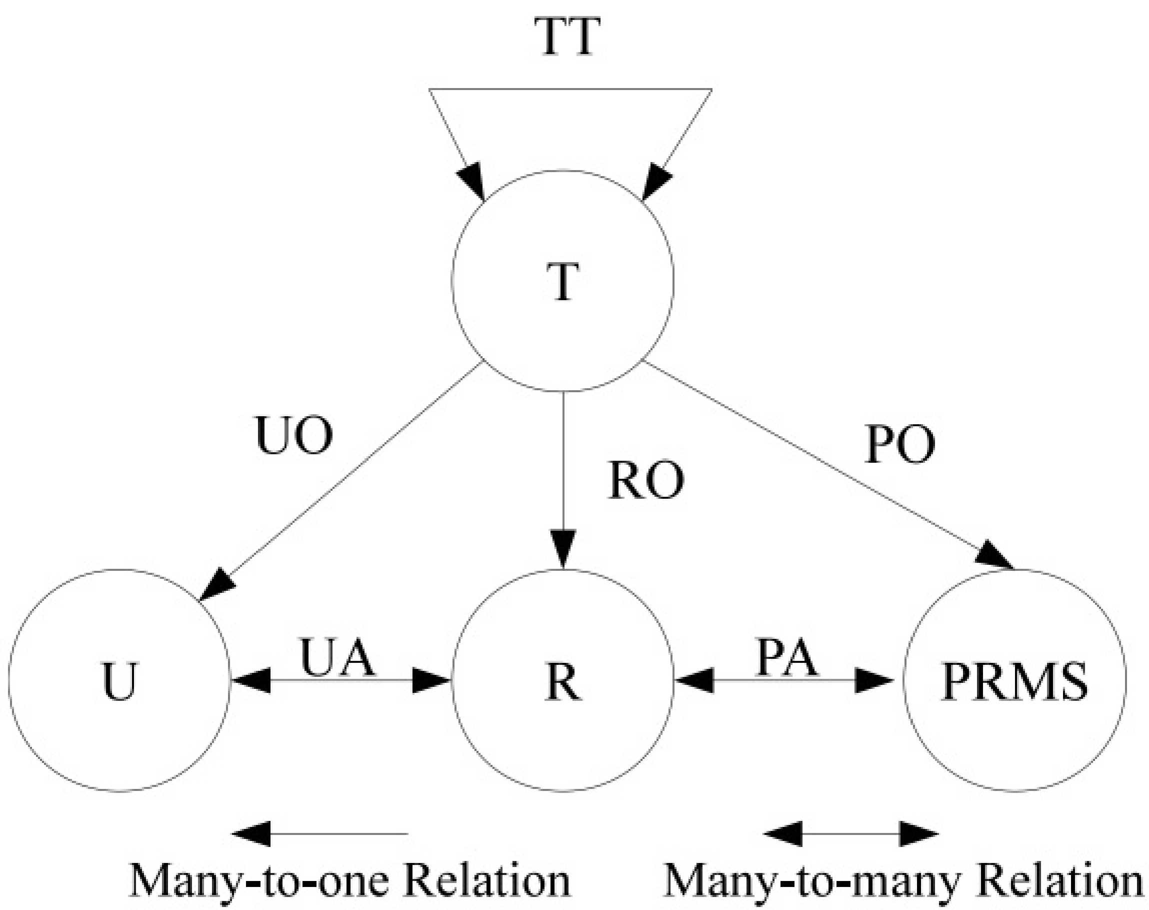
ReBAC for OpenMRS [Rizvi 2015]

- S. Rizvi, P. Fong, J. Crampton and J. Sellwood, *Relationship-Based Access Control for OpenMRS*, **SACMAT 2015**.
- Enforce ReBAC in a production-scale system
- Administrative Model
  - Add and remove access control relationships
  - Enabling precondition and applicability precondition

Motivation

- Extend Administrative ReBAC
- Use Case: Configure MT-RBAC
  - RBAC extension with multi-tenancy authorization
- Three motivating problems:
  - Enforce Global Integrity Policy Checks
  - Address Cascading Revocation
  - Resolve Multiple-ownership Issue

MT-RBAC



Multi-tenant RBAC Model Structure

Outline

- Introduction and Motivation
- AReBAC Models 1, 2 and 3
- Experiments
- Conclusion

AReBAC$_1$

- Supports two operations: Add or Remove Edges (a.k.a. Relationships)

- Consistency Policies:
  - The system graph G = (V; E) is always well-formed after allowing admin operation.

- Global Integrity Constraints:
  - Constraints based on certain conditions for participants.

AReBAC$_1$ (cont.)

- Operations

$$\mathcal{A}dd(\mathrm{e}_{admin}, \mathrm{e}_1, \mathrm{e}_2, \mathrm{r}) \lhd$$
$$\mathrm{e}_{admin} \in V \wedge \mathrm{e}_1 \in V \wedge \mathrm{e}_2 \in V \wedge$$
$$\mathrm{r} \in R \wedge (\tau(\mathrm{e}_1), \tau(\mathrm{e}_2), \mathrm{r}) \in E_{PR}$$
$$E' = E \cup \{<\mathrm{e}_1,\mathrm{e}_2, \mathrm{r}>\} \rhd$$

$$\mathcal{RM}(\mathrm{e}_{admin}, \mathrm{e}_1, \mathrm{e}_2, \mathrm{r}) \lhd$$
$$\mathrm{e}_{admin} \in V \wedge (\mathrm{e}_1, \mathrm{e}_2, \mathrm{r}) \in E$$
$$E' = E - \{<\mathrm{e}_1,\mathrm{e}_2, \mathrm{r}>\} \rhd$$

- Policy

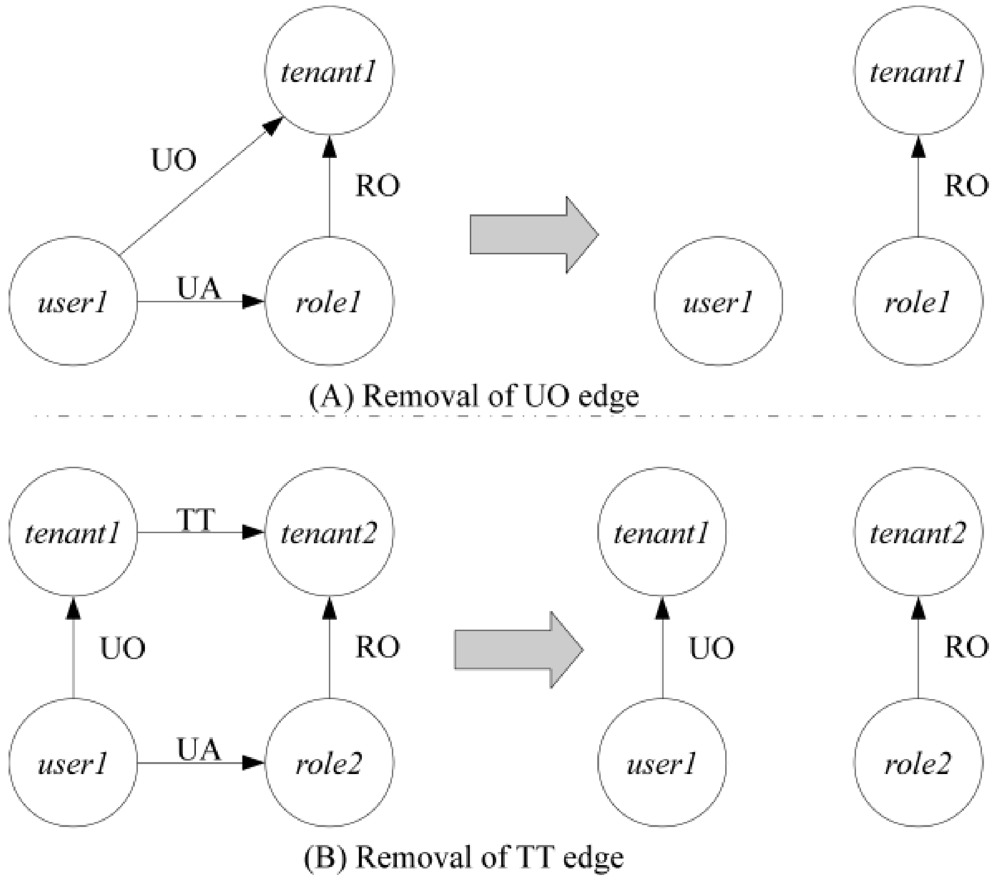
$$p = \mathsf{OP}(\mathrm{e}_{admin}, \mathrm{e}_1, \mathrm{e}_2, \mathrm{r}) \leftarrow \mathsf{enableC}(\mathrm{e}_{admin}, \mathrm{e}_1, \mathrm{e}_2) \wedge \mathsf{preC}(\mathrm{e}_1, \mathrm{e}_2)$$

- Examples

Operation	Enabling Pre-Condition	Applicability Pre-Condition
$\mathcal{A}dd(\mathrm{tenant}_1, \mathrm{tenant}_1, \mathrm{tenant}_2, \mathrm{TT})$	True	True
$\mathcal{RM}(\mathrm{tenant}_1, \mathrm{user}_1, \mathrm{role}_1, \mathrm{UA})$	$user \cdot UO \cdot tenant \wedge role \cdot RO \cdot tenant$	True
$\mathcal{A}dd(\mathrm{tenant}_2, \mathrm{tenant}_2, \mathrm{user}_2, \mathrm{UO})$	True	$(—, \mathrm{user}_2, \mathrm{UO}) \notin \mathrm{E}$

Cascading Revocation

- The operation will trigger a series of recursive removal of edges on the graph in addition to the direct consequence of the operation.



AReBAC_2

- Policy:

$$p = \mathcal{R}M(\mathrm{e}_{admin},\ \mathrm{e}_1,\ \mathrm{e}_2,\ \mathrm{r}) \leftarrow \mathsf{enableC}(\mathrm{e}_{admin},\ \mathrm{e}_1,\ \mathrm{e}_2) \wedge \mathsf{preC}(\mathrm{e}_1,\ \mathrm{e}_2) : \mathcal{C}_{revoke}(\mathrm{e}_1,\ \mathrm{e}_2,\ \mathrm{r}).$$

- $C_{revoke}(e_1, e_2, r)$ returns a set of edges that needs to be removed (possibly empty) when the policy p is used to authorize the edge removal operation.

Dependent-Edge Discovery

- Identification of dependent edges is non-trivial
  - Maintaining dependency relations could be costly

- Dependent-edge Discovery Algorithm
  - Depth-first search (O(V+E))
  - Dependency mapping function (O(1))
    - Maps the dependency edge (e_1, e_2, *label*) to an *ordered* set of relationship labels *Path*, and a set of dependent relationship labels R_d
  - Overall complexity is O(V+E)

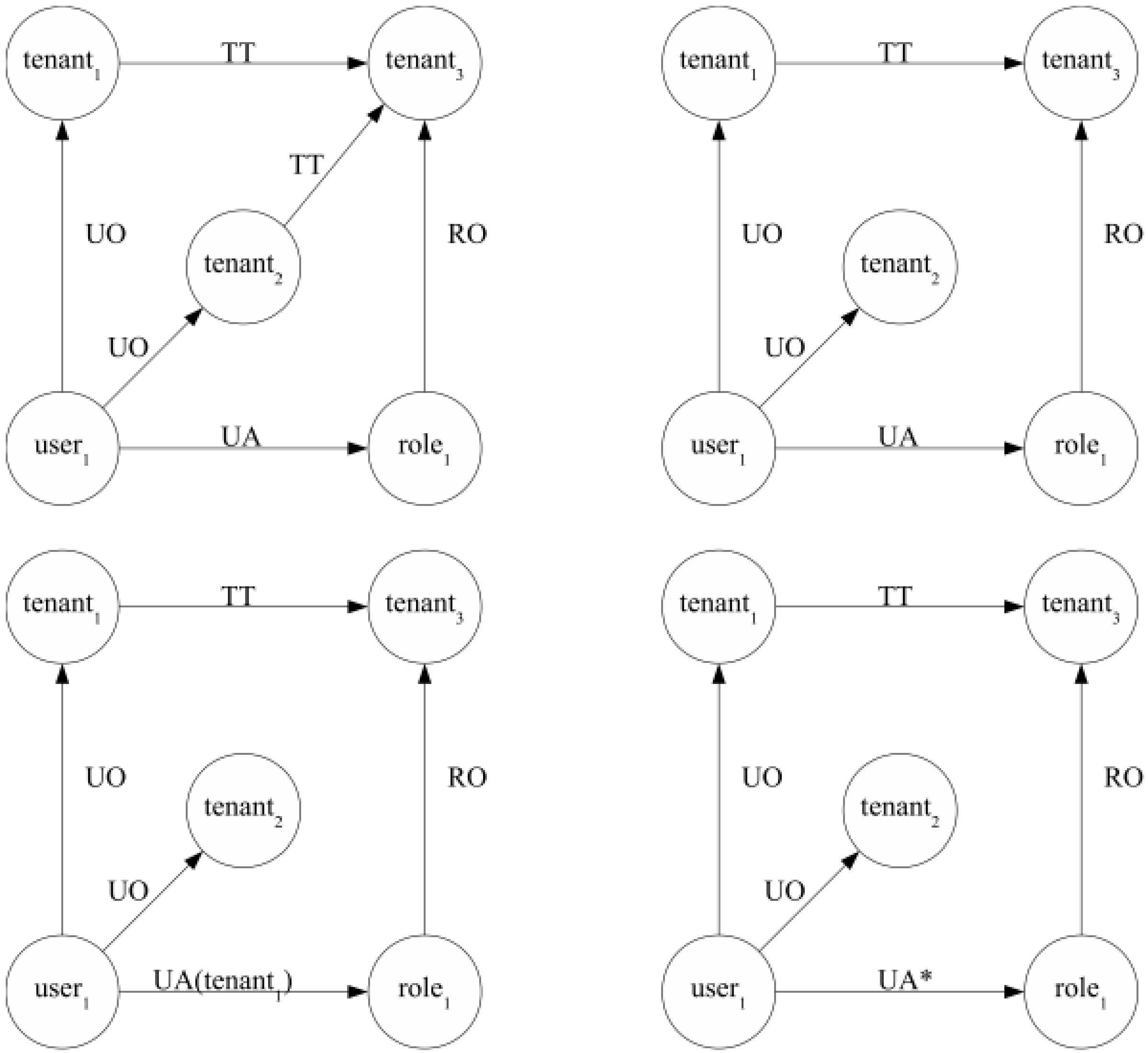
Provenance

- The provenance of a piece of data is the process that led to that piece of data
- Causality dependencies record the flow of transactions in the system
- The Open Provenance Model (OPM) captures such causality dependencies and expresses them in the provenance graph
- We can use provenance to address the multi-ownership issue

I·C·S The Institute for Cyber Security

Edges with Multiple Ownership

UTSA

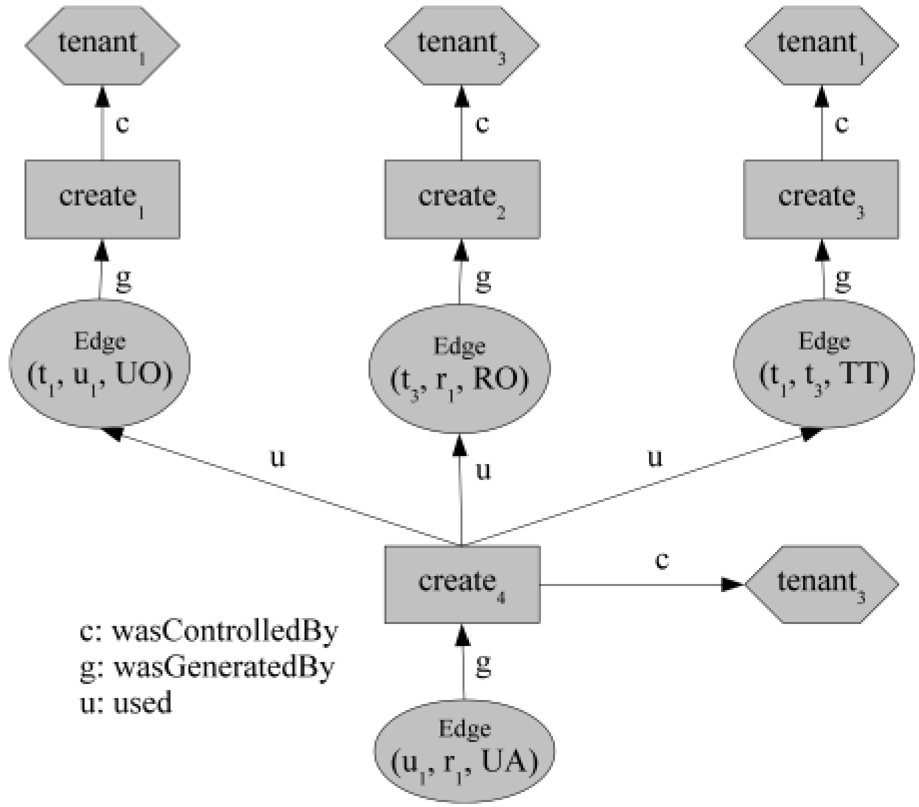


*: provenance data updated

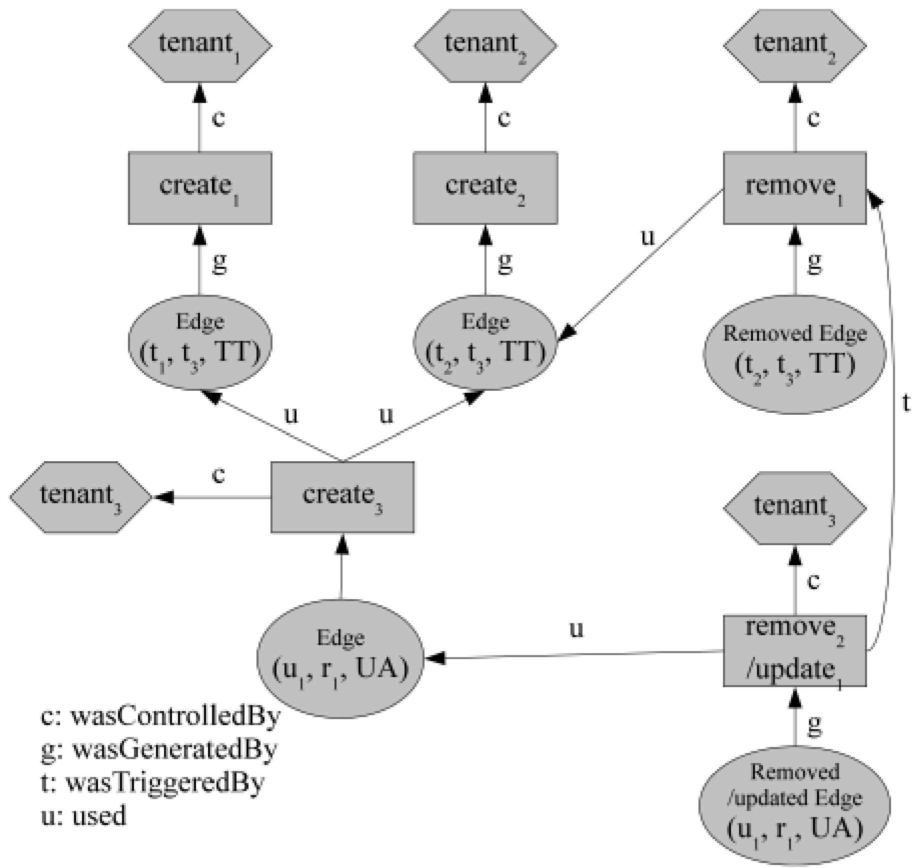
AReBAC$_3$

- Use provenance information to capture and express causality dependencies for assisting authorization.
  - Independent from ReBAC formalization
  - Extensible to enable Provenance-based Access Control (PBAC)
  - Potentially facilitate *multi-level* cascading revocation
  - Provenance vs typed parameters
    - More complicated and costly
    - More expressive power and richer information

OPM Graph for Adding UA Edge

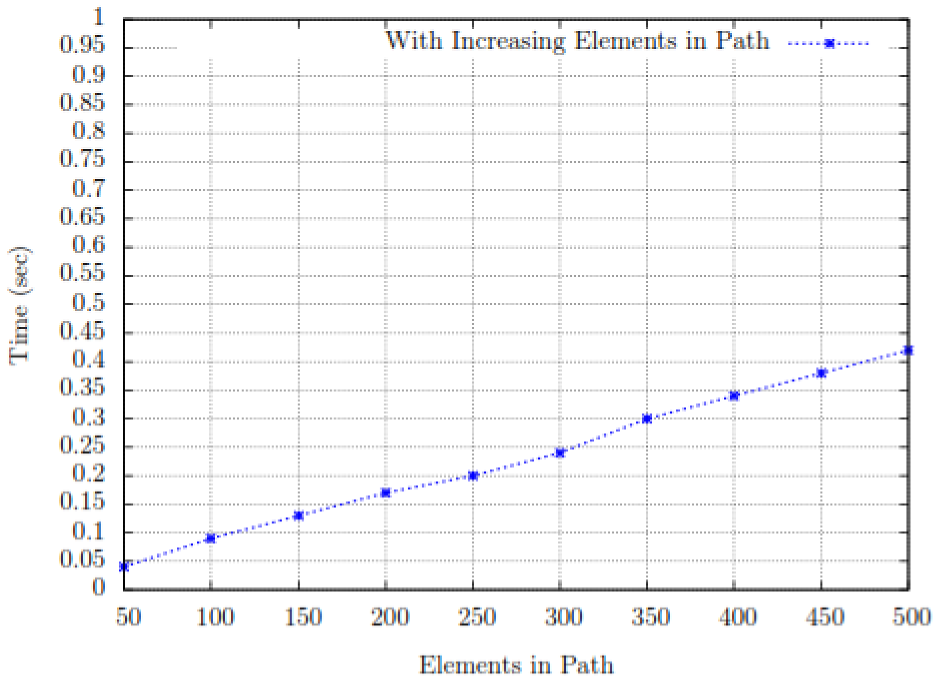


OPM Graph for Removing UA Edge



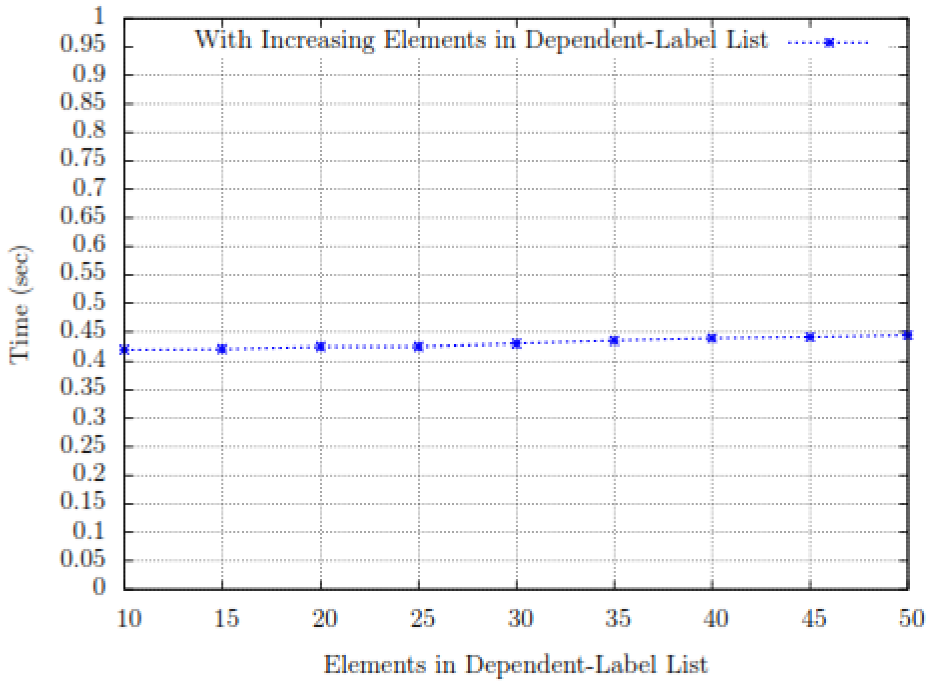
Experiments

- Experiment 1: Varied size of *Path*, fixed size of *rSet*



Experiments (cont.)

- Experiment 2: Fixed size of *Path*, varied size of *rSet*



Conclusion

- Proposed a family of three administrative ReBAC models based on RPPM^2 policy language
- Identified and addressed three problems
  - Integrity constraints
  - Cascading revocation
  - Multi-ownership of edges
- Provided a dependent-edge discovery algorithm
- Used the proposed models to capture MT-RBAC

- Next:
  - Investigate new problems about ReBAC administration
    - Policy administration
    - Synthesize ReBAC and PBAC, etc.

Questions? Comments?