A Provenance-based Access Control Model (PBAC)

July 18, 2012
PST’12, Paris, France

Jaehong Park, Dang Nguyen and Ravi Sandhu
Institute for Cyber Security
University of Texas at San Antonio

World-leading research with real-world impact!
Provenance Data

- Information of operations/transactions performed against data objects and versions
  - Actions that were performed against data
  - Agents who performed actions on data
  - Data used for actions
  - Data generated from actions
Provenance-aware Systems

- Capturing/expressing provenance data
- Storing provenance data
- Querying provenance data
- Using provenance data
- Securing provenance data

Access Control
Access control in Provenance-aware Systems

• **Provenance Access Control (PAC)**
  – Controlling access to provenance data which could be more sensitive than the underlying data
  – Needs access control models/mechanisms (e.g., RBAC)
  – (Meaningful) control granularity? Right level of abstraction?

• **Provenance-based Access Control (PBAC)**
  – Using provenance data to control access to the underlying data
  – Provenance-based policy specification

Meaningful granularity of provenance data?
Access Controls in Provenance-aware Systems

PBAC
- Extended PBAC
- Base PBAC

PAC
- Sanitization/Filtering/Redaction/....
- Role-based PAC
- Prov-based PAC

Common Foundations:
Base Provenance Data, DName (named abstraction) and matching DPath (Dependency Path Pattern)
PAC & PBAC in Applications

• Common Foundation
  – Base provenance data
  – Dependency list
    • Dependency Name: meaningful, named abstraction
    • matching regular expression-based causality dependency path pattern

• PAC and PBAC are complementary
  – In PAC, control decision can be based on provenance data (PB-PAC)
  – In PBAC, PAC can be used for added trustworthiness on provenance data
Provenance Data

- Directed Acyclic Graph (DAG)
- Causality dependencies between entities (acting users, action processes and data objects)

- Dependency graph can be traced for extracting Origin, usage, versioning information, etc.
  - PBAC can support origin/usage-based control, Dynamic Separation of Duty (DSOD), workflow control, etc.
From Open Provenance Model (OPM)

• 3 Nodes
  – Artifact (ellipse)
  – Process (Rectangle)
  – Agent (Hexagon)

• 5 Causality dependency edges (not dataflow)

• Provenance data: a set of 2 entities & 1 dependency
  • E.g., (ag,p1,a1,a2): <p1,ag,c>,<p1,a1,u>,<a2,p1,g>

World-leading research with real-world impact!
Direct vs. Indirect Dependencies

- **Direct dependencies**
  - Used (u), wasGeneratedBy (g), wasControlledBy (c)
  - Captured from transactions as base provenance data

- **Indirect dependencies**
  - System-computable dependencies
    - Using pre-defined dependency names and matching dependency path patterns
  - User-declared dependencies
    - Using pre-defined dependency names

World-leading research with real-world impact!
Object Dependency List (DL\textsubscript{O})

• A set of pairs of
  – abstracted dependency names (DNAME) and
  – regular expression-based object dependency path patterns (DPATH)

• Examples
  – \(<\text{wasSubmittedVof}, g_{\text{submit}} \cdot u_{\text{input}} >\>
  – \(<\text{wasAuthoredBy},\text{wasSubmittedVof?\text{.wasReplacedVof}} \cdot g_{\text{upload}} \cdot c >\)
PBAC vs. PAC

PBAC
- DNAME based Policy
- User Authorization
- Action Validation
- System-computable dependency
- User-declared dependency
- Policy Retrieval

Base Provenance Data & Dependency List

PAC
- DNAME as an initial control unit
- Sanitization/filtering on initial trace results
- Needs additional access control system

Provenance-aware System

World-leading research with real-world impact!
PBAC Model Components
A Family of PBAC Models

Combined Models

PBAC\textsubscript{U}  \hspace{2cm}  PBAC\textsubscript{A}  \hspace{2cm}  PBAC\textsubscript{PR}

PBAC\textsubscript{B}
PBAC_\text{B}: A Base Model

- System-captured Base Provenance Data only
  - Using only 3 direct dependencies (u, g, c)
  - No user-declared provenance data
- Object dependency only
- Policy is readily available
  - No policy retrieval required
Example: A Homework Grading System

1. Anyone can upload a homework.
2. A user can replace a homework if she uploaded it (origin-based control) and the homework is not submitted yet.
3. A user can submit a homework if she uploaded it and the homework is not submitted already. (workflow control)
4. A user can review a homework if she is not the author of the homework (DSOD), the user did not review the homework earlier, and the homework is submitted already but not graded yet.
5. A user can grade a homework if the homework is reviewed but not graded yet.
Sample Transactions &
Base Provenance Data

• (au1, upload1, o_{1v1}): < upload1, au1, c >, <o_{1v1}, upload1, g_{upload} >
• (au1, replace1, o_{1v1}, o_{1v2}): < replace1, au1, c >, <replace1, o_{1v1}, uinput>, <o_{1v2}, replace1, g_{replace} >
• (au1, submit1, o_{1v2}, o_{1v3}): < submit1, au1, c >, <submit1, o_{1v2}, uinput>, <o_{1v3}, submit1, g_{submit} >
• (au2, review1, o_{1v3}, o_{2v1}): < review1, au2, c >, <review1, o_{1v3}, uinput>, <o_{2v1}, review1, g_{review} >
• (au3, grade1, o_{1v3}, o_{3v1}): < grade1, au3, c >, <grade1, o_{1v3}, uinput>, <o_{3v1}, grade1, g_{grade} >
A Sample Base Provenance Data
A Sample Base Provenance Data

\[ \text{wasReviewedOby} \]

\[ \text{wasReplacedVof} \]
\[ D_{\text{L0}}: < \text{wasReplacedVof}, g_{\text{replace}} \cdot u_{\text{input}} > \]

\[ \text{wasSubmittedVof} \]

\[ \text{wasReviewedOof} \]

\[ \text{wasGradedOof} \]
A Sample Base Provenance Data

wasAuthoredBy
DL₀: <wasAuthoredBy, wasSubmittedVof?, wasReplacedVof * .g_{\text{upload}} \cdot c >
A Sample Base Provenance Data wasReviewedBy DL₀: < wasReviewedBy, wasReviewedOof⁻¹. wasReviewedOby >
Sample Object Dependency List (DL\textsubscript{O})

1. < wasReplacedVof, g\textsubscript{replace}.u\textsubscript{input} >
2. < wasSubmittedVof, g\textsubscript{submit}.u\textsubscript{input} >
3. < wasReviewedOof, g\textsubscript{review}.u\textsubscript{input} >
4. < wasReviewedOby, g\textsubscript{review}.c >
5. < wasGradedOof, g\textsubscript{grade}.u\textsubscript{input} >
6. < wasAuthoredBy, wasSubmittedVof\textsubscript{-1}.wasReplacedVof \cdot g\textsubscript{upload}.c >
7. < wasReviewedBy, wasReviewedOof\textsuperscript{-1}.wasReviewedOby >
Sample Policies

1. Anyone can upload a homework.
2. A user can replace a homework if she uploaded it (origin-based control) and the homework is not submitted yet.
3. A user can submit a homework if she uploaded it and the homework is not submitted already. (workflow control)

1. allow(au, upload, o) ⇒ true
2. allow(au, replace, o) ⇒ au∈(o, wasAuthoredBy) ∧ |(o,wasSubmittedVof)| = 0.
3. allow(au, submit, o) ⇒ au∈(o, wasAuthoredBy) ∧ |(o,wasSubmittedVof)|=0.

World-leading research with real-world impact!
Sample Policies (cont.)

4. A user can **review** a homework if she is not the author of the homework *(DSOD)*, the user did not review the homework earlier, and the homework is submitted already but not graded yet.

5. A user can **grade** a homework if the homework is reviewed but not graded yet.

4. \( \text{allow}(\text{au}, \text{review}, o) \Rightarrow \text{au} \notin (o, \text{wasAuthoredBy}) \land \text{au} \notin (o, \text{wasReviewedBy}) \land |(o, \text{wasSubmittedV of})| \neq 0 \land |(o, \text{wasGradedOof}^{-1})| = 0. \)

5. \( \text{allow}(\text{au}, \text{grade}, o) \Rightarrow |(o, \text{wasReviewedOof})| \neq 0 \land |(o, \text{wasGradedOof}^{-1})| = 0. \)
Access Evaluation Procedure

- Rule collecting phase
- User authorization (UAuth) phase
- Action validation (AVal) phase
- Conjunctive decision of UAuth and AVal
Access Evaluation Example

- Policy: user can submit a homework if she uploaded it (origin-based control) and the homework is not submitted already. (workflow control)

\[(au1, submit2, o_{1v3})\]
Rule Collecting Phase

• Request: \((au_1, submit_2, o_{1v3})\)
• Action type: \textit{submit}
• Policy for \textit{submit}
  \[ \text{allow}(au, submit, o) \Rightarrow au \in (o, \text{wasAuthoredBy}) \land |(o, \text{wasSubmittedVof})| = 0. \]
• User authorization rule
  \[ au \in (o, \text{wasAuthoredBy}) \]
• Action Validation rule
  \[ |(o, \text{wasSubmittedVof})| = 0 \]
User Authorization Phase

- **User Authorization Rule:** \( au \in (o, \text{wasAuthoredBy}) \)
- **Dependency List (DL)**
  - \( \langle \text{wasReplacedVof}, g_{\text{replace}}.u_{\text{input}} \rangle, \langle \text{wasSubmittedVof}, g_{\text{submit}}.u_{\text{input}} \rangle \)
  - \( \langle \text{wasAuthoredBy}, \text{wasSubmittedVof}\?\text{.wasReplacedVof} \* g_{\text{upload}}.c \rangle \)
- \( au1 \in (o1v3, [g_{\text{submit}}.u_{\text{input}}]?[g_{\text{replace}}.u_{\text{input}}]\*g_{\text{upload}}.c) = \{au1\} \)
Action Validation Phase

- **Action Validation Rule**: $|(o, wasSubmittedVof)| = 0$
- **Dependency List (DL)**: $< wasSubmittedVof, g_{submit \cdot u_{input}} >$
- $|(o1v3, g_{submit \cdot u_{input}})| \neq 0$
Summary

• Proposed a foundation for PBAC and PAC
  – the notion of named abstractions of causality dependency path patterns
  – Regular expression-based dependency path pattern

• Identified a Family of PBAC models

• Developed a Base model for PBAC
  – Supports Simple and effective policy specification and access control management
  – Supports DSOD, workflow control, origin-based control, usage-based control, object versioning, etc.
What’s next?

- Enhancing/extending PBAC model
- Provenance Access Control Models
- Provenance data sharing in multiple systems
Thank you!

• Questions and Comments?