Provenance-based Access Control Models

July 31, 2014
Dissertation Defense

Dang Nguyen
Institute for Cyber Security
University of Texas at San Antonio

World-leading research with real-world impact!
Presentation Outline

1. Introduction
2. Provenance Data Model
3. Provenance-based Access Control Models
4. PBAC Architecture in Cloud Infrastructure-as-a-Service
5. Conclusion
Background: what is provenance?

Art definition of provenance

- Essential in judging authenticity and evaluating worth.

Data provenance in computing systems

- Is different from log data.
- Contains linkage of information pieces.
- Is utilized in different computing areas.

World-leading research with real-world impact!
Access Control Challenges

- **Usability** of provenance
  - Capturing,
  - Storing,
  - and Querying provenance data.
- **Utility** of provenance
  - Policy specification,
  - Evaluation,
  - and Enforcement.
- **Provenance** in cloud environment
  - Tenant-awareness

**Security** of provenance: provenance access control

Provenance Data Model

Provenance-based Access Control

PBAC in IaaS Architecture
Access Control Approaches

• Traditional access control
  – Based on single units of control: roles, primitive attributes, etc.

• Relationship-based access control
  – Graph-based.
  – Does not make use of history information.

• Based on history information
  – Utilizes log data to extract useful information
    • Mainly looks at users’ history.
  – Cannot specify access control based on linkage information.
  – Assume history information is readily available.

Provenance-based Access Control
Provenance-based Access Control (PBAC)

- So far, no comprehensive and well-defined model in the literature.

- Compared to other access control approaches, PBAC provides richer access control mechanisms
  - Finer-grained policy and control.
  - Provides effective means of history information usage.

- Easily configured to apply in different computing domains and platforms
  - Single system (XACML)
  - Multi-tenant cloud (OpenStack)

World-leading research with real-world impact!
Contributions

• Proposed a provenance data model which enables PBAC configurations in multiple application domains.

• Proposed provenance-based access control models which provides enhanced and finer-grained access control features.
  – Implemented and evaluated an XACML-extended prototype.

• Proposed architecture to enable PBAC in cloud IaaS.
  – Implemented and evaluated an OpenStack-extended prototype.
Thesis Statement

Provenance data forms a directed-acyclic graph where graph edges exhibit the causality dependency relations between graph nodes that represent provenance entities.

A provenance data model that can enable and facilitate the capture, storage and utilization of such information through regular expression based path patterns can provide a foundation for enhancing access control mechanisms.

In essence, provenance-based access control models can provide effective and expressive capabilities in addressing access control issues, including traditional and previously not discussed dynamic separation of duties, in single systems, distributed systems, and within a single tenant and across multiple tenants cloud environment.
Scope and Assumptions

• Assumptions
  – Provenance data is uncompromised and protected.
  – Provenance data is correct.
  – Provenance of provenance is not considered.

• Experimental Scope
  – Does not include provenance capture.
  – Does not include concurrent, dependent access requests.
Presentation Outline

1. Introduction
2. Provenance Data Model
3. Provenance-based Access Control Models
4. PBAC Architecture in Cloud Infrastructure-as-a-Service
5. Conclusion
Characteristics of Provenance Data

- Information of operations/transactions performed against data objects and versions
  - Actions that were performed against data
  - Acting Users/Subjects who performed actions on data
  - Data Objects used for actions
  - Data Objects generated from actions
  - Additional Contextual Information of the above entities

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

- Dependency graph can be traced/traversed for the discovery of Origin, usage, versioning info, etc.
Provenance Data Model
[inspired by OPM]

- **4 Node Types**
  - Object (Artifact)
  - Action (Process)
  - Subject (Agent)
  - Attribute

- **3 Causality dependency edge Types**
  (not a dataflow) and Attribute Edge

Inverse edges are enabled for usage in queries, but **cycle-avoidant**.

- **c** wasControlledBy
- **u** used
- **g** wasGeneratedBy
- **t** hasAttribute

World-leading research with real-world impact!
Capturing, Storing, and Querying Provenance Data

Transaction:

RDF Triples:

querying

SPARQL:

SELECT ?agent WHERE { HW1_G [g:c] ?agent }
SELECT ?user WHERE { HW1_G [g:t[actUser]] ?user }

World-leading research with real-world impact!
Provenance Graph Example

SELECT ?user
WHERE
{ HW1_G' [g\:u\:g\:c] ?user }

{ HW1_G' [[g\:u]\:*\:g\:c ] ?user }

World-leading research with real-world impact!
Students can upload a homework to the system, after which they can replace it multiple times before they submit the homework. Once it is submitted, the homework can be reviewed by other students or designated graders until it is graded by the teaching assistant (TA).
A Base Provenance Data Graph
Dependency List

• Dependency List (DL): A set of identified dependencies that consists of pairs of
  – Dependency Name: abstracted dependency names (DNAME) and
  – regular expression-based dependency path pattern (DPATH)

• Examples
  – < wasReplacedVof, g_replace u_input >
  – < wasAuthoredBy, wasSubmittedVof?.wasReplacedVof *.g_upload.c >
A Base Provenance Data Graph

wasReviewedOby

wasReplacedVof
$DL_O: < wasReplacedVof, g_{replace}.u_{input} >$

wasSubmittedVof

wasReviewedOof

wasGradedOof
Presentation Outline

1. Introduction
2. Provenance Data Model
3. Provenance-based Access Control Models
4. PBAC Architecture in Cloud Infrastructure-as-a-Service
5. Conclusion
PBAC Models

- **PBAC_B**: utilizes base data model
  - Does not capture contextual information

- **PBAC_C**: extending the base model
  - Incorporate *contextual information* associated with the main entities (*Subjects*, etc.)
  - Extend base data model with attributes
PBAC$_B$ Components

- **Subjects**
- **Actions**
- **Objects**

User authorization

**Access Evaluation**

- **Policies**
- **Dependency Lists**
- **Base Provenance Data**

World-leading research with real-world impact!
Sample Policies

1. Anyone can upload a homework.
2. A user can replace a homework if she uploaded it (usr. authz) and the homework is not submitted yet (act. valid).

1. allow(au, upload, o) ⇒ true
2. allow(au, replace, o) ⇒ au∈(o, wasAuthoredBy) ∧ |(o,wasSubmittedVof)| = 0.
PBAC \textsubscript{C} Components

Contextual Info.

Subjects \rightarrow Actions \rightarrow Objects

Access Evaluation

- Policies
- Dependency Lists
- Base Provenance Data
- Attribute Provenance Data

World-leading research with real-world impact!
DSOD Examples in HGS

• Sample English policies:
  – A student cannot **review** the homework he **submitted** – Object-based DSOD
  – A student cannot **grade** a homework before it is **submitted** – History-based DSOD
  – A student cannot **grade** a homework unless **reviews’ combined weights** exceeds 3 – Transaction Control Expression

• An informal policy:
  
  allow(sub,grade,o) =>
  
  sum(o,previousReviewProcesses.hasAttributeOf(Weight)) <= 3

• Compatible to **XACML** policy language
  – Extending **OASIS XACML** architecture and implementation.
Extended XACML Architecture

- PEP: policy enforcement point
- PDP: policy decision point
- PAP: policy administration point
- PIP: policy information point
Experiment and Performance

• **System**
  – Ubuntu 12.10 image with 4GB Memory and 2.5 GHz quad-core CPU running on a Joyent SmartData center (ICS Private Cloud).

• **Mock Data simulating HGS scenario**
  – Extreme depth and width settings for graph traversal queries.

• **Results for tracing 2k/12k edges**
  – 0.017/0.718 second per deep request
  – 0.014/0.069 second per wide request
Throughput Evaluation

- 500 concurrent, nondependent requests
- Results for tracing 2k/12k edges
  - 0.014/0.16 second per deep request
  - 0.014/0.04 second per wide request
Presentation Outline

1. Introduction
2. Provenance Data Model and Access Control
3. Provenance-based Access Control Models
4. PBAC Architecture in Cloud Infrastructure-as-a-Service
5. Conclusion
Cloud Computing

- Cloud computing has been the “next big thing.”
- Has 3 primary service models:
  - Software-as-a-Service (SaaS)
  - Platform-as-a-Service (PaaS)
  - Infrastructure-as-a-Service (IaaS)
- We focus on PBAC for IaaS
  - Specifically, multi-tenant single-cloud systems.
Access Control Aspects

• DSOD concerns for virtual resources management and protection
  – Ex: Only virtual images up-loaders are allowed to delete.

• Multi-tenant concerns
  – A virtual image may be created in one tenant, copied to another tenant and modified, and used to launch a virtual machine instance in another.
Tenant-aware PBAC

Tenants as contextual information.
Architecture Overview

CloudService

Provides system data

ProvService (PS)

Provides Provenance Data

ProvAuthzService (PBAS)

Provides authorization
Deployment Architecture

Variations:

- Integrated Deployment
- Stand-alone Deployment
- Hybrid Deployment

Design pros & cons:
- Ease of integration
- Communication latency
- Provenance data sharing
Logical Architecture

PROV-SERVICE Dataflow

PROVAUTHZ-SERVICE Dataflow

World-leading research with real-world impact!
OpenStack Conceptual Architecture

World-leading research with real-world impact!
Nova PBAS
Implementation

World-leading research with real-world impact!
Experiments

• Measure the time an authorization process takes from the time of request until decision is returned.
  – nova list
  – glance image-list

• 4 experimental configurations:
  – E1: normal Nova and Glance authorization.
  – E2: integrated PBAS/PS services with Nova and Glance.
  – E3: integrated PBAS/PS service, stand-alone from Nova and Glance.
  – E4: separate PBAS and PS services, stand-alone from Nova and Glance.

• Deployment Configurations:
  – 4GB RAM, 2.5 GHz quad-core CPU.
  – OpenStack Devstack (Grizzly) on 12.04 Ubuntu.

• Mainly test deep-shaped provenance graphs.
  – Generate mock data for virtual images and machines scenario.
## Results and Evaluation

<table>
<thead>
<tr>
<th>Traversal Distance</th>
<th>Glance (e1)</th>
<th>Glance (e2)</th>
<th>Glance (e3)</th>
<th>Glance (e4)</th>
</tr>
</thead>
<tbody>
<tr>
<td>No PBAC</td>
<td>0.55</td>
<td>-</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>20 Edges</td>
<td>-</td>
<td>0.575</td>
<td>0.607</td>
<td>.642</td>
</tr>
<tr>
<td>1000 edges</td>
<td>-</td>
<td>.612</td>
<td>.788</td>
<td>.852</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Traversal Distance</th>
<th>Nova (e1)</th>
<th>Nova (e2)</th>
<th>Nova (e3)</th>
<th>Nova (e4)</th>
</tr>
</thead>
<tbody>
<tr>
<td>No PBAC</td>
<td>0.75</td>
<td>-</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>20 Edges</td>
<td>-</td>
<td>0.84</td>
<td>0.902</td>
<td>1.062</td>
</tr>
<tr>
<td>1000 edges</td>
<td>-</td>
<td>2.292</td>
<td>.362</td>
<td>4.102</td>
</tr>
</tbody>
</table>
Conclusion

✓ Proposed a framework of **provenance data and PBAC models** for enhanced access control.

✓ Proposed an **architecture** that enables PBAC and PS in cloud IaaS.

✓ **Proof-of-concept prototypes**
  1. XACML architecture extension and evaluation.
  2. OpenStack architecture extension and evaluation.

➢ **An access control foundation for secure provenance-centric computing!**
Future Work and Directions

- Expanding provenance data model to include user-declared provenance data.

- Collaborated PBAC usage
  - Multi-cloud.
  - Distributed systems.

- Full-cycle implementation and evaluation
  - including provenance capturing service.

- Provenance Access Control models and mechanisms.
  - Utilizing PBAC foundations.
Publications


Thank you!!!

Questions and Comments?