Multi-Tenancy Authorization Models for Collaborative Cloud Services

Bo Tang, Ravi Sandhu, and Qi Li

Presented by Bo Tang
OUTLINE

- Introduction
- Background & Motivation
- Formalized Models
  - MTAS
  - AMTAS
  - Enhanced Trust Models
- Policy Specification
- Conclusion and Future Work
OUTLINE

- Introduction
- Background & Motivation
- Formalized Models
  - MTAS
  - AMTAS
  - Enhanced Trust Models
- Policy Specification
- Conclusion and Future Work
Cloud Computing

- **Shared infrastructure**
  - [$$$] -------> [$|$|$]

- **Multi-Tenancy**
  - Virtually dedicated resources

- **Drawbacks:**
  - Data Locked-in
    - Collaborations can only be achieved through desktop.
    - E.g.: open Dropbox files with GoogleDoc.
  - How to collaborate?

Collaborative Access Control

- Centralized Facility
  - Chance for centralized models in distributed systems

- Agility
  - Collaboration and collaborators are temporary

- Homogeneity
  - Handful of popular brands

- Out-Sourcing Trust
  - Built-in collaboration spirit
Industry Solutions

- Microsoft and IBM: Fine-grained data sharing in SaaS using DB schema
  - Only feasible in DB
- NASA: RBAC + OpenStack
  - Lacks ability to support collaborations
- Salesforce (Force.com): SSO + SAML
  - Focus on authentication
  - Heavy management of certificates

Source:
http://nebula.nasa.gov/blog/2010/06/03/nebulas-implementation-role-based-access-control-rbac/
OUTLINE

- Introduction
- Background & Motivation
- Formalized Models
  - MTAS
  - AMTAS
  - Enhanced Trust Models
- Policy Specification
- Conclusion and Future Work
Literature

- **RBAC**
  - CBAC, GB-RBAC, ROBAC
  - Require central authority managing collaborations

- **Delegation Models**
  - dRBAC and PBDM
  - Lacks agility (which the cloud requires)

- **Grids**
  - CAS, VOMS, PERMIS
  - Absence of centralized facility and homogeneous architecture (which the cloud has)

**Problem:**
semantic mismatch
Role-based Trust

- RT, Traust, RMTN AND RAMARS_TM
- Calero et al: towards a multi-tenant authorization system for cloud services
  - Implementation level PoC
  - Open for extensions in trust models
- Suits the cloud (out-sourcing trust)

Challenge: trust relation
OUTLINE

- Introduction
- Background & Motivation
- Formalized Models
  - MTAS
  - AMTAS
  - Enhanced Trust Models
- Policy Specification
- Conclusion and Future Work
Authorization as a Service (AaaS)

AaaS

Multi-Tenant Access Control

Cross-Tenant Access

flickr
Dropbox
Google

Rackspace Hosting

Amazon Web Services
Windows Azure

salesforce
If A trusts B then B (resource owner) can assign

- B’s permissions to A’s roles; and
- B’s roles as junior roles to A’s roles.
### Table I
Administration Functions of AMTAS for Issuer $i$

<table>
<thead>
<tr>
<th>Function</th>
<th>Condition</th>
<th>Update</th>
</tr>
</thead>
<tbody>
<tr>
<td>assignUser $(i, r, u)$</td>
<td>$i = \text{roleOwner}(r) \land u \in U$</td>
<td>$UA' = UA \cup {u \to r}$</td>
</tr>
<tr>
<td>revokeUser $(i, r, u)$</td>
<td>$i = \text{roleOwner}(r) \land u \in U \land u \to r \in UA$</td>
<td>$UA' = UA \setminus {u \to r}$</td>
</tr>
<tr>
<td>assignPerm $(i, r, p)$</td>
<td>$i = \text{permOwner}(p) \land i \in \text{canUse}(r)$</td>
<td>$PA' = PA \cup {p \to r}$</td>
</tr>
<tr>
<td>revokePerm $(i, r, p)$</td>
<td>$i = \text{permOwner}(p) \land i \in \text{canUse}(r) \land p \to r \in PA$</td>
<td>$PA' = PA \setminus {p \to r}$</td>
</tr>
<tr>
<td>assignRH $(i, r_1, r)$</td>
<td>$i = \text{roleOwner}(r) \land i \in \text{canUse}(r_1) \land \neg (r_1 \gg r) \land \neg (r \geq r_1)$</td>
<td>$\geq' = \geq \cup {r_2, r_3 : R</td>
</tr>
<tr>
<td>revokeRH $(i, r_1, r)$</td>
<td>$i = \text{roleOwner}(r) \land i \in \text{canUse}(r_1) \land r_1 \gg r$</td>
<td>$\geq' = (\gg \setminus {r_1 \to r})^c$</td>
</tr>
<tr>
<td>assignTrust $(i, i_1)$</td>
<td>$i_1 \in I$</td>
<td>$\sim' = \sim \cup {i \to i_1}$</td>
</tr>
<tr>
<td>revokeTrust $(i, i_1)$</td>
<td>$i_1 \in I \land i \preceq i_1 \land i \neq i_1$</td>
<td>$\sim' = \sim \setminus {i \to i_1}$</td>
</tr>
</tbody>
</table>

- **a.** This condition avoids cycle creation in the role hierarchy.
- **b.** It requires $r_1$ to be an immediate ascendant of $r$.
- **c.** Implied relations are preserved after revocation.
- **d.** By revoking the trust relation, the canUse() function of $i$’s roles automatically updates accordingly, same as $PA$ and $RH$. 

**Note:** The conditions for $assignRH$ and $revokeRH$ are complex and involve multiple relationships and conditions, which are not fully detailed here. They are designed to handle specific administrative updates in the AMTAS model.
Problem of MTAS

- Over exposure of truster’s authorization information

Truster-Centric Public Role (TCPR)

- Expose only the truster’s public roles

Relation-Centric Public Role (RCPR)

- Expose public roles in terms of each trust relation
Constraints

- Cyclic Role Hierarchy: lead to implicit role upgrades in the role hierarchy
- SoD: conflict of duties
  - Tenant-level
    - E.g.: SOX compliance companies may not hire the same company for both consulting and auditing.
  - Role-level
    - across tenants
- Chinese Wall: conflict of interests among tenants
OUTLINE

- Introduction
- Background & Motivation
- Formalized Models
  - MTAS
  - AMTAS
  - Enhanced Trust Models
- Policy Specification
- Conclusion and Future Work
Example

- **Role <PolicySet> (RPS-i1)**
  - `<Target>`
    - subject-role = i1
    - subject-tenant = i2
  - `<PolicySetIdReference>`

- **Trust <PolicySet> (TPS-i1)**
  - `<Target>`
    - subject-tenant = i1
  - `<PolicySetIdReference>`

- **Perm. <PolicySet> (PPS-i1)**
  - `<Target>`
    - resource-id = i1
  - `<PolicySetIdReference>`

- **Cross-Issuer**
  - PA / RH

- **Role <PolicySet> (RPS-i2)**
  - `<Target>`
    - subject-role = i2
  - `<PolicySetIdReference>`

- **Trust <PolicySet> (TPS-i2)**
  - `<Target>`
    - subject-tenant = i2
  - `<PolicySetIdReference>`

- **Perm. <PolicySet> (PPS-i2)**
  - `<Target>`
    - resource-id = i2
  - `<PolicySetIdReference>`
OUTLINE

- Introduction
- Background & Motivation
- Formalized Models
  - MTAS
  - AMTAS
  - Enhanced Trust Models
- Policy Specification
- Conclusion and Future Work
Conclusion

- Collaboration needs in the cloud eco-system
- Novel service model: AaaS
- Proposed formal models
  - MTAS, AMTAS, Enhanced Trust Models
  - Constraints
- Policy Specification
Future Work

Accomplished

- Prototype and evaluation
  - Performance overhead ≈ 0.016 seconds
  - Scalable in the cloud
- MT-RBAC (delegation-centric trust model)

On-going Projects

- OpenStack Keystone extensions
- Integrate trust into ABAC: MT-ABAC
- Unified trust framework
Thank You!