Multi-Tenant Access Control for Cloud Services

PhD Dissertation Defense
Bo Tang

Committee Members:
Dr. Ravi Sandhu, Chair
Dr. Kay Robbins
Dr. Gregory White
Dr. Weining Zhang
Dr. Jaehong Park

07/31/2014
The Cloud

Anytime
Anywhere

World-Leading Research with Real-World Impact!
Really? But where is my data?
Really? But where is my data?

Multi-Tenancy
Cloud & Multi-Tenancy

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

- **Multi-Tenancy**
  - Isolated workspace for customers
  - Virtually temporarily dedicated resources

- **Problem:**
  - How to collaborate across tenants?
    - Even if across my own tenants?
Define Tenant

- All deployment models are multi-tenant
  - E.g.: public cloud, private cloud and community cloud.

- From Cloud Service Provider (CSP) perspective
  - A billing customer
  - Manages its own users and cloud resources

- The owner of a tenant can be
  - An individual, an organization or a department in an organization, etc.
Characteristics of Cloud

➢ Centralized Facility
  ❖ Resource pooling

➢ Self-Service Agility
  ❖ Each tenant manages its own authorization
  ❖ Tenants, users and resources are temporary

➢ Homogeneity
  ❖ Identical or similar architecture and system settings

➢ Out-Sourcing Trust
  ❖ Built-in collaboration spirit
Multi-Tenant Access Control (MTAC)

CTTM

MTAS

MT-RBAC

MT-ABAC

Policy (P)

Enforcement (E)

MTAaaS Framework

Implementation (I)

Domain Trust in OpenStack

Chapter 3

Chapter 4

Chapter 5

Top-Down Approach
Motivation

World-Leading Research with Real-World Impact!
Problem Statement

The fact that contemporary cloud services are intrinsically not designed to cultivate collaboration between tenants limits the development of the cloud. Fine-grained access control models in traditional distributed environments are not directly applicable.

Thesis Statement

The problem of multi-tenant access control in the cloud can be partially solved by integrating various types of unidirectional and unilateral trust relations between tenants into role-based and attribute-based access control models.
Chapter 2: Related Work

Centralized Approaches
- RBAC extensions: ROBAC, GB-RBAC
- Multi-domain role mapping

Decentralized Approaches
- RT, dRBAC: credential-based delegation
- Delegation models: PBDM, RBDM

Attribute-Based Approaches
- NIST ABAC: application framework for collaboration
- ABAC models: ABURA, RBAC-A, $\text{ABAC}_\alpha$, $\text{ABAC}_\beta$

Enforcement and Implementation
- Grid: PERMIS, VOMS, CAS
- Web: ABAC for SOA systems
- Cloud: centralized authorization service with trust models
Scope and Assumptions

- Standardized APIs
  - Cross-tenant accesses are functionally available
- Properly authenticated users
- One Cloud Service
  - Of a kind: IaaS, PaaS or SaaS.
- Two-Tenant Trust (rather than community trust)
- Unidirectional Trust Relations
  - “I trust you” does not mean “you trust me”
- Unilateral Trust Relations (trustor trusts trustee)
  - Trustee cannot control the trust relation
Multi-Tenant Access Control (MTAC)

World-Leading Research with Real-World Impact!

Chapter 3

Chapter 4

Chapter 5

Policy (P)

MTAS

MT-RBAC

MT-ABAC

Enforcement (E)

MTAaaS Framework

Implementation (I)

Domain Trust in OpenStack

Top-Down Approach
MTAS

Formalizing Calero et al work

Trust

Issuer Trust (IT)

Tenants (T)

Role Ownership (RO)

Permission Ownership (PO)

User Ownership (UO)

User Assignment (UA)

Permission Assignment (PA)

Role Hierarchy (RH)

Users (U)

Roles (R)

Permissions (P)

World-Leading Research with Real-World Impact!
Tenant Trust (TT) relation is not partial order

- Reflexive: $A \sqsubseteq A$
- But not transitive: $A \sqsubseteq B \land B \sqsubseteq C \not\Rightarrow A \sqsubseteq C$
- Neither symmetric: $A \sqsubseteq B \not\Rightarrow B \sqsubseteq A$
- Nor anti-symmetric: $A \sqsubseteq B \land B \sqsubseteq A \not\Rightarrow A \equiv B$
Tenants are managed by CSP

- on self-service basis

Each tenant administer:

- Trust relations with other tenants
- Entity components:
  - users, roles and permissions
- UA, PA and RH assignments
  - Cross-tenant assignments are issued by the trustee (t1)
    - UA: trustor (t2) users to trustee (t1) roles
    - PA: trustee (t1) permissions to trustor (t2) roles
    - RH: trustee (t1) roles junior to trustor (t2) roles
Fine-grained Trust Extensions

- Problem of MTAS trust model
  - Over exposure of trustor’s authorization information

- Trustor-Centric Public Role (TCPR)
  - Expose only the trustor’s public roles
    - E.g.: OS expose only the dev.OS role to all the trustees

- Relation-Centric Public Role (RCPR)
  - Expose public roles specific for each trust relation
    - E.g.: OS expose only the dev.OS role to E when OS trusts E
Trust Types Between Tenants

- **Intuitive Trust (Type-α)**
  - Delegations: RT, PBDM, etc.
  - Trustor gives access to trustee
    - Trustor has full control

- **MTAS trust (Type-β)**
  - Trustee gives access to trustor

- **Other Types?**
  - Trustee takes access from trustor (Type-γ)
  - Trustor takes access from trustee (Type-δ)
  - And more?
Example of Cross-Tenant Trust

[$]$: grant the access

**Example:**

- **Type-α**: E trusts OS so that E can say [$].
- **Type-β**: OS trusts E so that E can say [$].
- **Type-γ**: E trusts OS so that OS can say [$].
- **Type-δ**: OS trusts E so that OS can say [$].
Example of Cross-Tenant Trust

[§]: grant the access

Example:

- **Type-α**: E trusts OS so that E can say [§].
- **Type-β**: OS trusts E so that E can say [§].
- **Type-γ**: E trusts OS so that OS can say [§].
- **Type-δ**: OS trusts E so that OS can say [§].
MT-RBAC

Issuers: Real-world Owners e.g. E and OS

Type-γ Trust
Issuers administer tenants

Each issuer administer:

- Trust relations from owned tenants
- Entity components:
  - tenants, users, roles and permissions
- UA, PA and RH assignments
  - Cross-tenant assignments are issued by the trustee’s (t2’s) issuer
    - UA: trustee (t2) users to trustor (t1) roles
    - RH: trustor (t1) roles junior to trustee (t2) roles
  - Cross-tenant PA assignments are intentionally banned
    - PA: trustee (t2) assign trustor (t1) permissions to trustee (t2) roles
    - Problem:
      » Trustor cannot revoke PA other than remove the trust
Finer-grained Trust Models

- **MT-RBAC0: Base Model**
  - Trustor exposes all the roles to trustees

- **MT-RBAC1: Trustee-Independent Public Role (TIPR)**
  - Expose only the trustor’s public roles
    - E.g.: E expose only the dev.E role to all the trustees

- **MT-RBAC2: Trustee-Dependent Public Role (TDPR)**
  - Expose public roles specific for each trustee
    - E.g.: E expose only the dev.E role to OS when E trusts OS
Constraints

- Cyclic Role Hierarchy: lead to implicit role upgrades in the role hierarchy
- SoD: conflict of duties
  - Tenant-level
    - E.g.: SOX compliant companies may not hire the same company for both consulting and auditing.
  - Role-level
    - Checks across tenants
- Chinese Wall: conflict of interests among tenants
  - E.g.: never share resources with competitors.
Multi-Tenant Access Control (MTAC)

CTTM

- MTAS
- MT-RBAC
- MT-ABAC

Policy (P)

Enforcement (E)

MTAaaS Framework

Implementation (I)

Domain Trust in OpenStack

Chapter 3
Chapter 4
Chapter 5

Top-Down Approach

World-Leading Research with Real-World Impact!
CTTM Trust Types

Four potential trust types:

- **Type-α**: trustor can give access to trustee. (e.g. RT)
- **Type-β**: trustee can give access to trustor. (e.g. MTAS)
- **Type-γ**: trustee can take access from trustor. (e.g. MT-RBAC)
- **Type-δ**: trustor can take access from trustee.
  - No meaningful use case, since the trustor holds all the control of the cross-tenant assignments of the trustee’s permissions.
Formalized CTTM Model

Tenants (T)

User Ownership (UO)

Permission Ownership (PO)

Authorization Assignment (AA)

Tenant Trust (TT)

Users (U)

Permissions (P)
Role-Based CTTM

Tenants (T)

Roles (R)

Permissions (P)

Users (U)

Tenant Trust (TT)

Role Ownership (RO)

Permission Ownership (PO)

User Assignment (UA)

Permission Assignment (PA)

Role Hierarchy (RH)
Multi-Tenant Access Control (MTAC)

CTTM

MTAS
MT-RBAC
MT-ABAC

Chapter 3
Chapter 4
Chapter 5

Policy (P)

Enforcement (E)

MTAaaS Framework

Implementation (I)

Domain Trust in OpenStack

World-Leading Research with Real-World Impact!
Multi-Tenant Access Example

(a) no trust required

(b) require A trust B

user  subject  object

subject ownership  permission inheritance

World-Leading Research with Real-World Impact!
Real-World Clouds

AWS

- Collaboration between accounts
  - E.g.: E trusts OS
- Unilateral trust relation (Type-α)
  - The trustor needs to map the roles

OpenStack

- User-level delegation (trust) can be established
- Cross-domain assignments bear no control
Multi-Tenant Access Control (MTAC)

Policy (P)

CTTM

MTAS

MT-RBAC

MT-ABAC

Top-Down Approach

Chapter 3

Chapter 4

Chapter 5

Enforcement (E)

MTAaaS Framework

Implementation (I)

Domain Trust in OpenStack

World-Leading Research with Real-World Impact!
Centralized (Chosen)

- Centralized PDP with distributed PEP
  - Pros: easy management
  - Cons: volume of requests may be high

Decentralized

- Distributed PDP and PEP
  - Pros: requests handling
  - Cons: keep decision consistent
Example MTAS policy structure

**OS policysets**

Trust `<PolicySet> (TPS:OS)`
- `<Target>` subject-tenant = OS
- `<PolicySetIdReference>` trust

Role `<PolicySet> (RPS:OS:...)`
- `<Target>` subject-role = OS:...
- `<PolicySetIdReference>`

Perm. `<PolicySet> (PPS:OS)`
- `<Target>` resource-tenant = OS

**E policysets**

Trust `<PolicySet> (TPS:E)`
- `<Target>` subject-tenant = E
- `<PolicySetIdReference>`

Role `<PolicySet> (RPS:E:...)`
- `<Target>` subject-role = E:...
- `<PolicySetIdReference>`

Perm. `<PolicySet> (PPS:E)`
- `<Target>` resource-tenant = E

Cross-Issuer UA
- RH
- UA

Cross-Issuer PA
- PA

RH
FlexCloud Testbed
- PEP × 8: SmartOS 1.8.1 / CPU Cap=350 / 256MB RAM
- PDP: 64-bit CentOS 6 / 1-, 2-, 4-, 8-, 16-Units
- ATC: SmartOS 1.8.4 / CPU Cap=350 / 1GB RAM
- PEPs in a same network which is different with PDP’s

1 unit = 1CPU/1GB RAM
Evaluation: Performance

- **MT-RBAC vs RBAC**
  - More policy references incur more decision time
- **MT-RBAC2 introduces 12 ms authz. overhead.**

![Graph showing PDP Performance](image1)

![Graph showing Client-End Performance](image2)

World-Leading Research with Real-World Impact!
MTAS introduces 12 ms authz. overhead.
Evaluation: Scalability

- Scalable in terms of both
  - PDP hardware capacity
  - Policy complexity

Where:

\[
\text{Throughput} = \frac{1}{\text{Average Delay} \times \text{CPU Utilization}}
\]
Multi-Tenant Access Control (MTAC)

CTTM

Policy (P)

MTAS
MT-RBAC
MT-ABAC

Enforcement (E)

MTAaaS Framework

Implementation (I)

Domain Trust in OpenStack

Top-Down Approach

Chapter 3
Chapter 4
Chapter 5

World-Leading Research with Real-World Impact!
rule:add_user_to_tenant -> (role:keystone_admin ||
(role:tenant_admin && tenant_id:%(target_tenant_id)s) ||
(domain_role:domain_admin && domain_id:%(target_domain_id)s))

rule:add_tenant_to_domain -> (role:keystone_admin ||
(domain_role:domain_admin && domain_id:%(target_domain_id)s))

Source: https://wiki.openstack.org/wiki/Domains
Prototype & Evaluation

➢ Sequential request handling (Queuing)
   ❖ Domain trust introduces 0.7% authz. Overhead
   ❖ Scalability changes little with domain trust
Chapter 6: Conclusion

➢ Policy
  ❖ MTAS: role-based Type-β trust
  ❖ MT-RBAC: role-based Type-γ trust
  ❖ CTTM: trust type taxonomy for role-based models
  ❖ MT-ABAC: attribute-based model trusts

➢ Enforcement
  ❖ MTAaaS: centralized PDP with distributed PEP

➢ Implementation
  ❖ Domain Trust in OpenStack

World-Leading Research with Real-World Impact!
Chapter 6: Future Work

- **MT-ABAC**
  - Finer-grained extensions
  - Administration, enforcement and implementation.

- **More and finer-grained trust models**
  - Trust negotiation and graded trust relations

- **More MTAC models**
  - MT-PBAC, MT-RAdAC, etc.

- **Attribute-based MTAC models in OpenStack**
Publications


Thank You!

World-Leading Research with Real-World Impact!