Multi-Tenant Access Control for Collaborative Cloud Services

CS6393 Spring 2014

PhD Seminar

Bo Tang

April 11, 2014

What is Multi-Tenancy?

Cloud & Multi-Tenancy

- Shared infrastructure
  - [$$$] -----> [$|$|$]
- Multi-Tenancy
  - Virtually dedicated resources
    - E.g.: rent-a-car
- Problems:
  - Who owns the data?
  - How to collaborate across tenants?
    - Even if across my own tenants?

amazon web services™

Source: http://blog.box.com/2011/06/box-and-google-docs-accelerating-the-cloud-workforce/

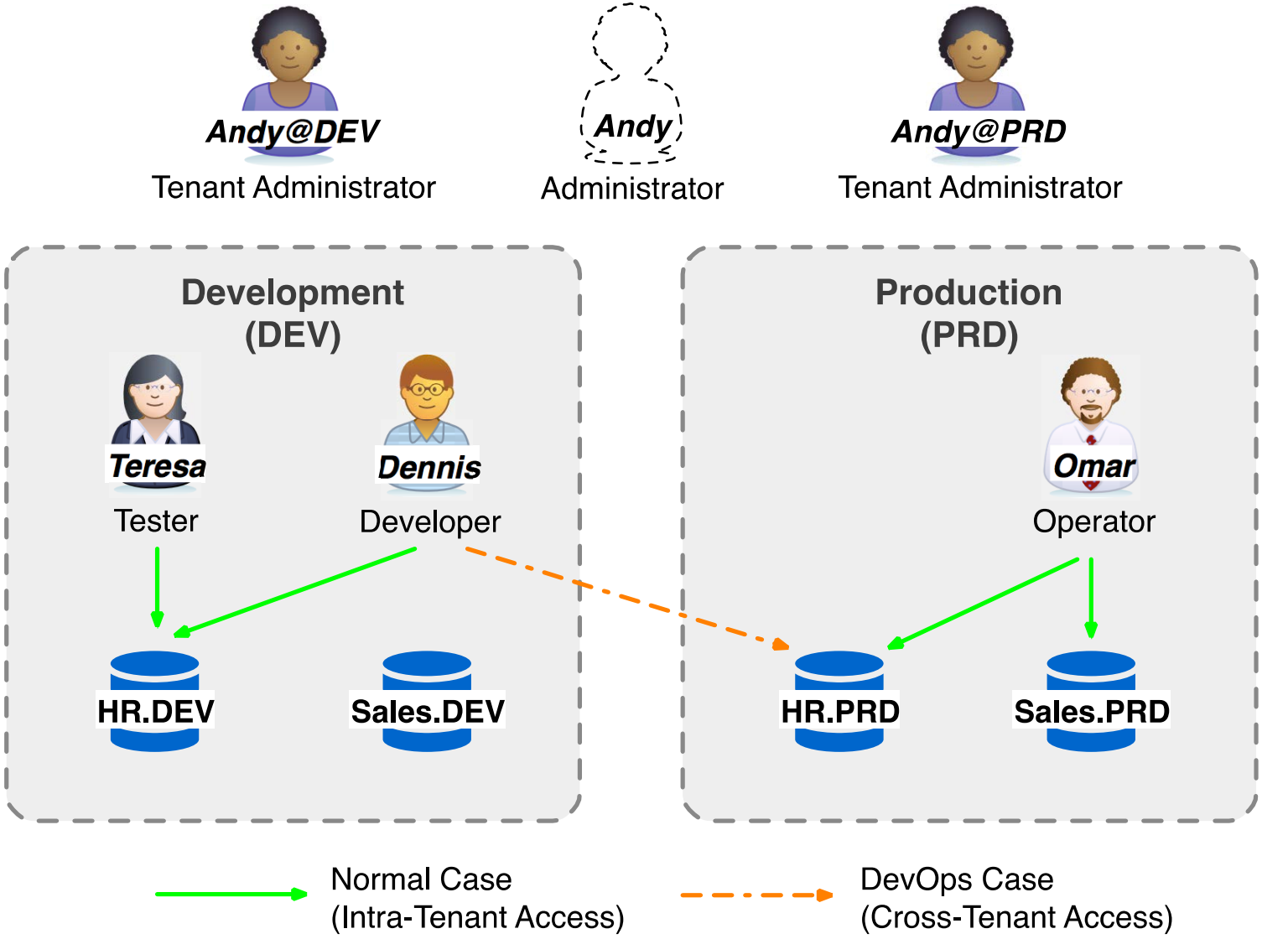
Characteristics of Cloud

- Distributed Authority
  - Each tenant manages its own authorization
- Centralized Facility
  - Resource pooling
- Agility
  - Tenants, users and resources are temporary
- Homogeneity
  - Identical or similar architecture and system settings
- Out-Sourcing Trust
  - Built-in collaboration spirit

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
- From consumer perspective
  - An individual, an organization or a department in an organization, etc.
  - virtually dedicated space with on-demand self-service

DevOps in the Cloud



Multi-org/dom A/C Literature

- RBAC
  - CBAC, GB-RBAC, ROBAC
  - No cross-organization interaction
  - Require central authority managing collaborations
- Delegation Models
  - dRBAC and PBDM (e.g.: allowing subleasing)
  - Lacks agility (which the cloud requires)
- Grids
  - CAS, VOMS, PERMIS
  - Heavy authorization overhead due to the absence of homogeneous infrastructure (which the cloud has)

Literature (Contd.)

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

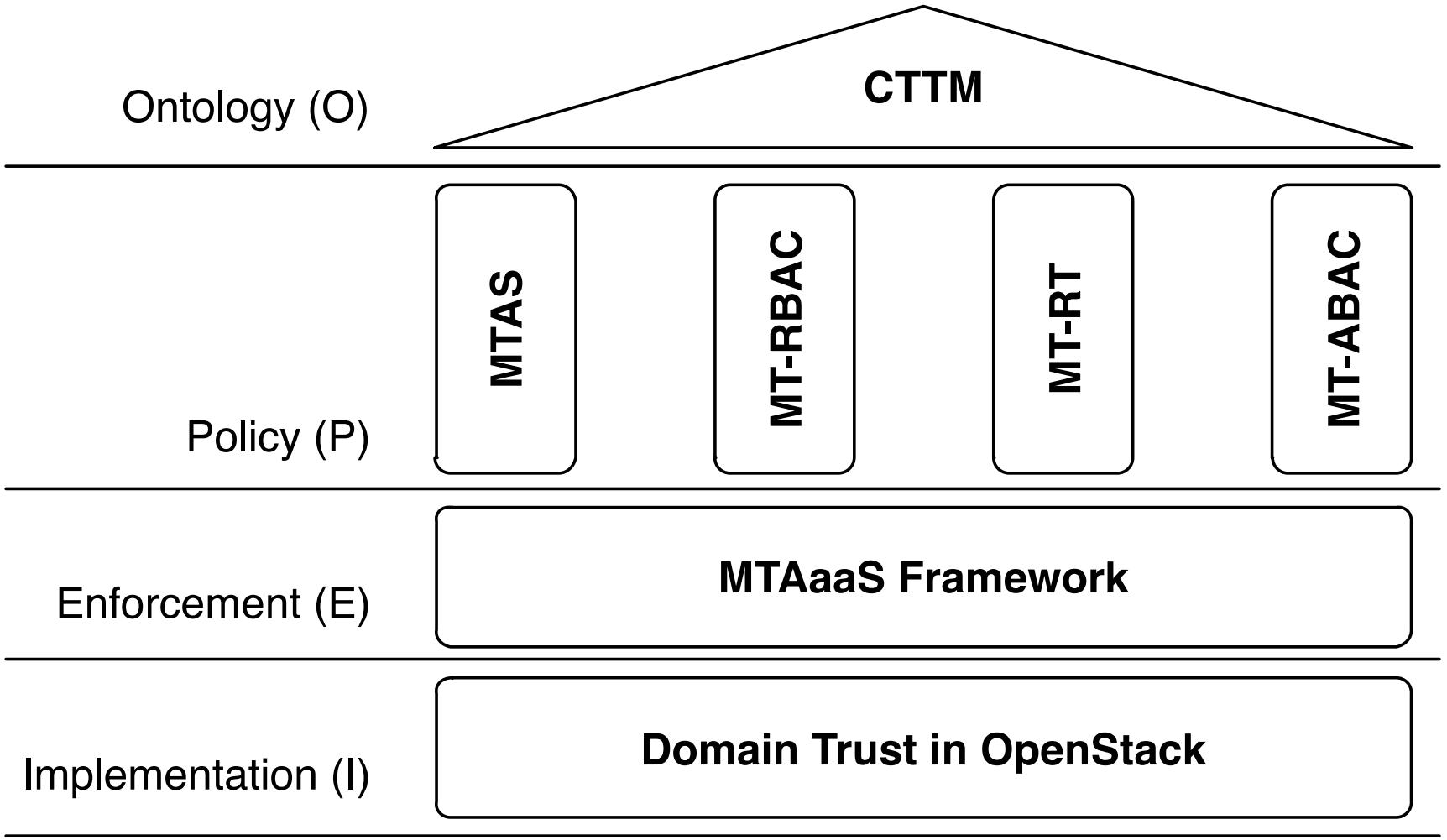
Challenge:

- Trust relation
- Finer-grained models
- Administration

Scope and Assumptions

- Standardized APIs
  - Cross-tenant accesses are functionally available
- Proper authentication of users
- Removable assumptions:
  - One Cloud Service
    - Of a kind: IaaS, PaaS or SaaS etc.
  - Two-Tenant Trust (rather than community trust)
  - Unidirectional Trust Relations (like follow in Twitter)
  - Unilateral Trust Relations (trustor or trustee)

Multi-Tenant Access Control



Tenant Trust

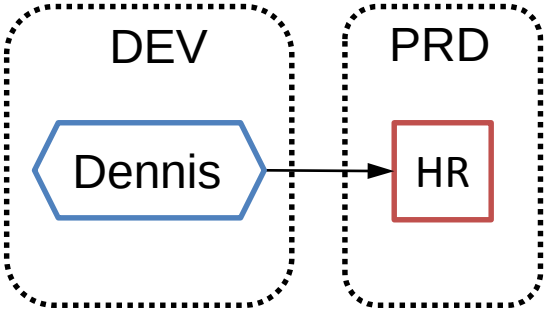
- Tenant Trust (*TT*) relation is not partial order
- It is
  - Reflexive: $A \trianglelefteq A$
  - But not transitive: $A \trianglelefteq B \wedge B \trianglelefteq C \nRightarrow A \trianglelefteq C$
  - Neither symmetric: $A \trianglelefteq B \nRightarrow B \trianglelefteq A$
  - Nor anti-symmetric: $A \trianglelefteq B \wedge B \trianglelefteq A \nRightarrow A \equiv B$

CTTM Trust Types

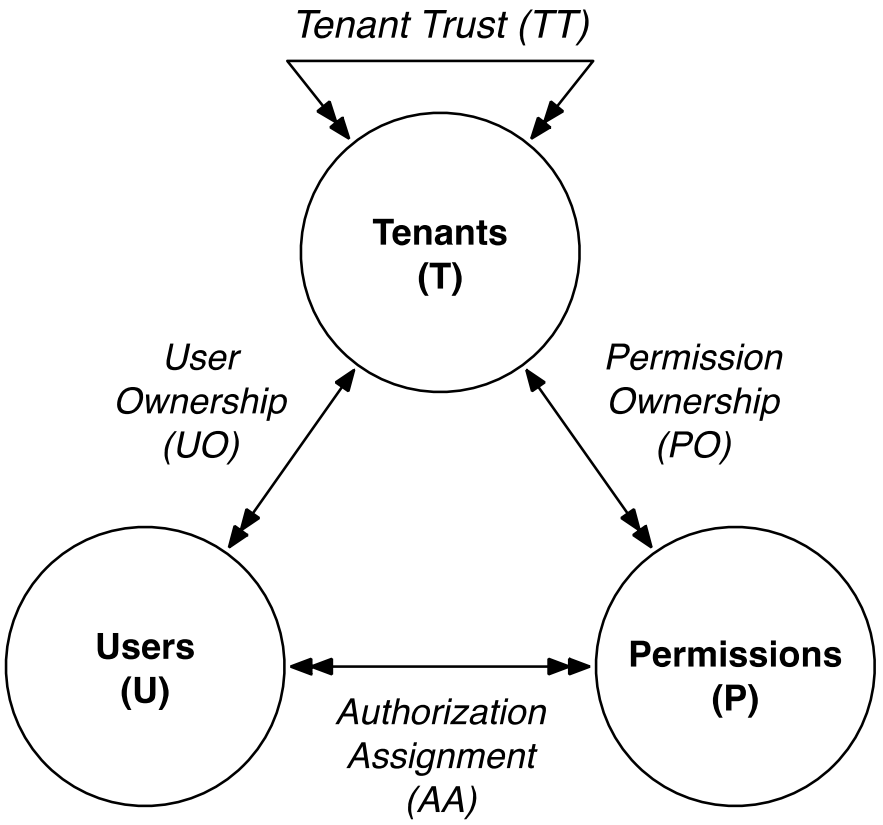
- Four potential trust types:
  - Type-α: trustor can give access to trustee.
  - Type-β: trustee can give access to trustor.
  - Type-γ: trustee can take access from trustor.
  - ~~Type-δ: trustor can take access from trustee.~~
    - No meaningful use case, since the trustor holds all the control of the cross-tenant assignments of the trustee's permissions.

Cross-Tenant Trust in DevOps

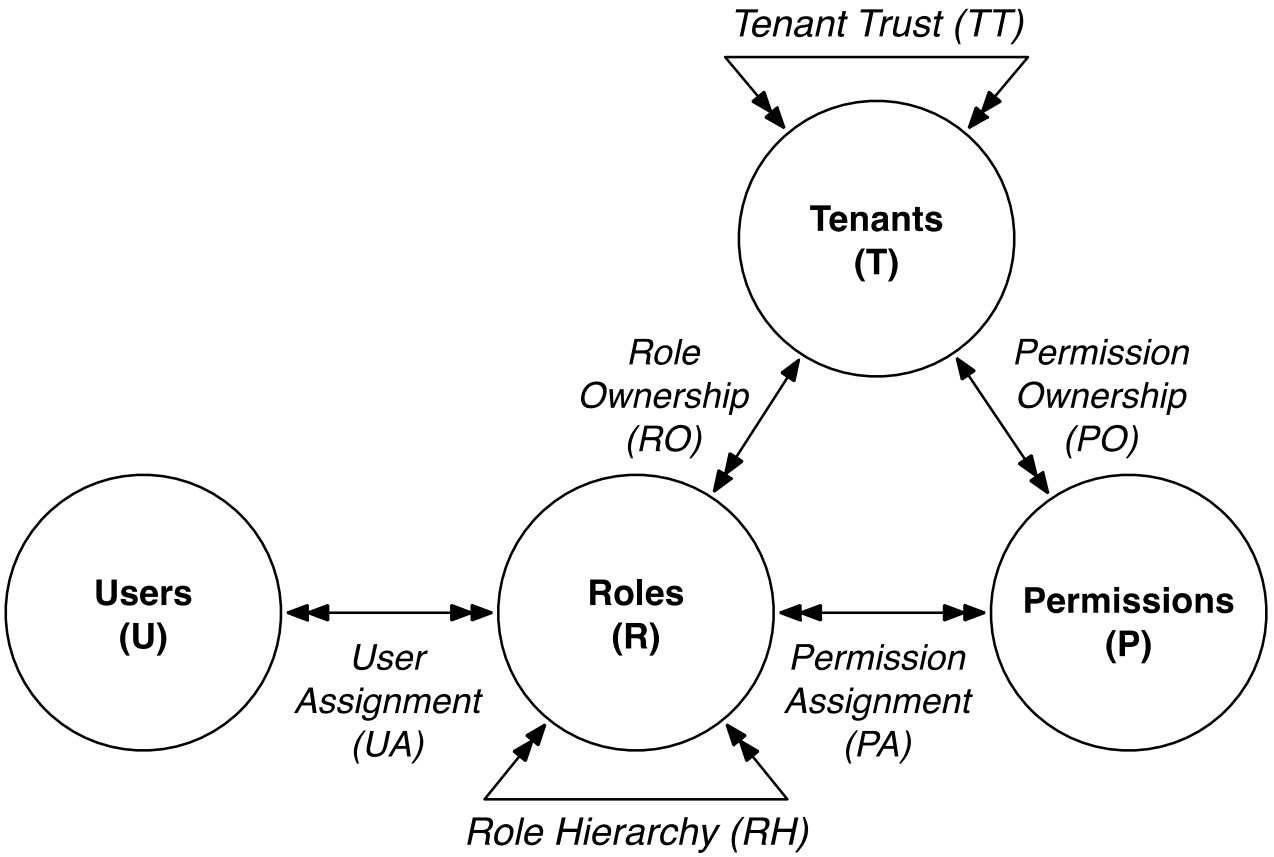
- Example: Temporary DevOps access
  - [$]: grant Dennis@DEV access to HR.PRD
  - Trust-α:
    - PRD trusts DEV so that PRD can say [$].
  - Trust-β:
    - DEV trusts PRD so that PRD can say [$].
  - Trust-γ:
    - PRD trusts DEV so that DEV can say [$].



Formalized CTTM Model

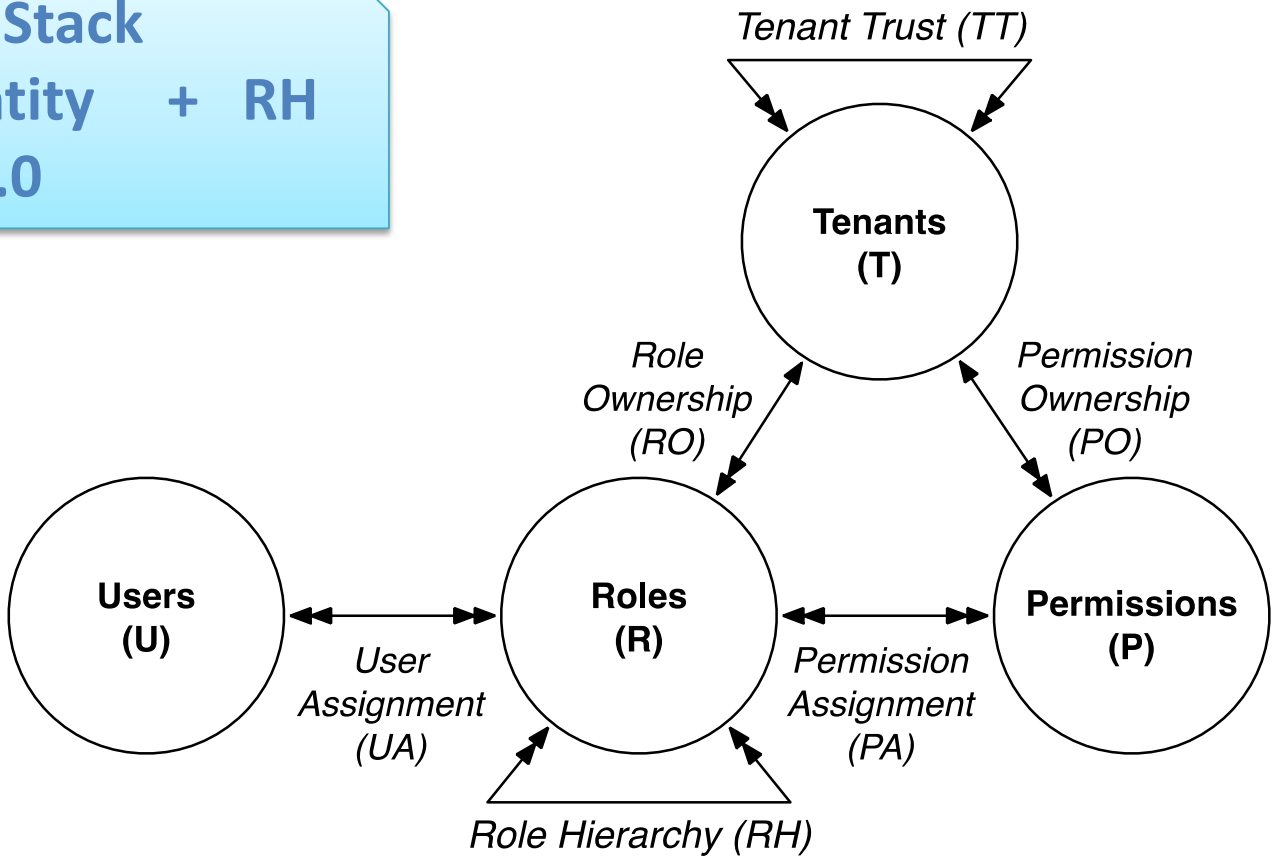


Role-Based CTTM

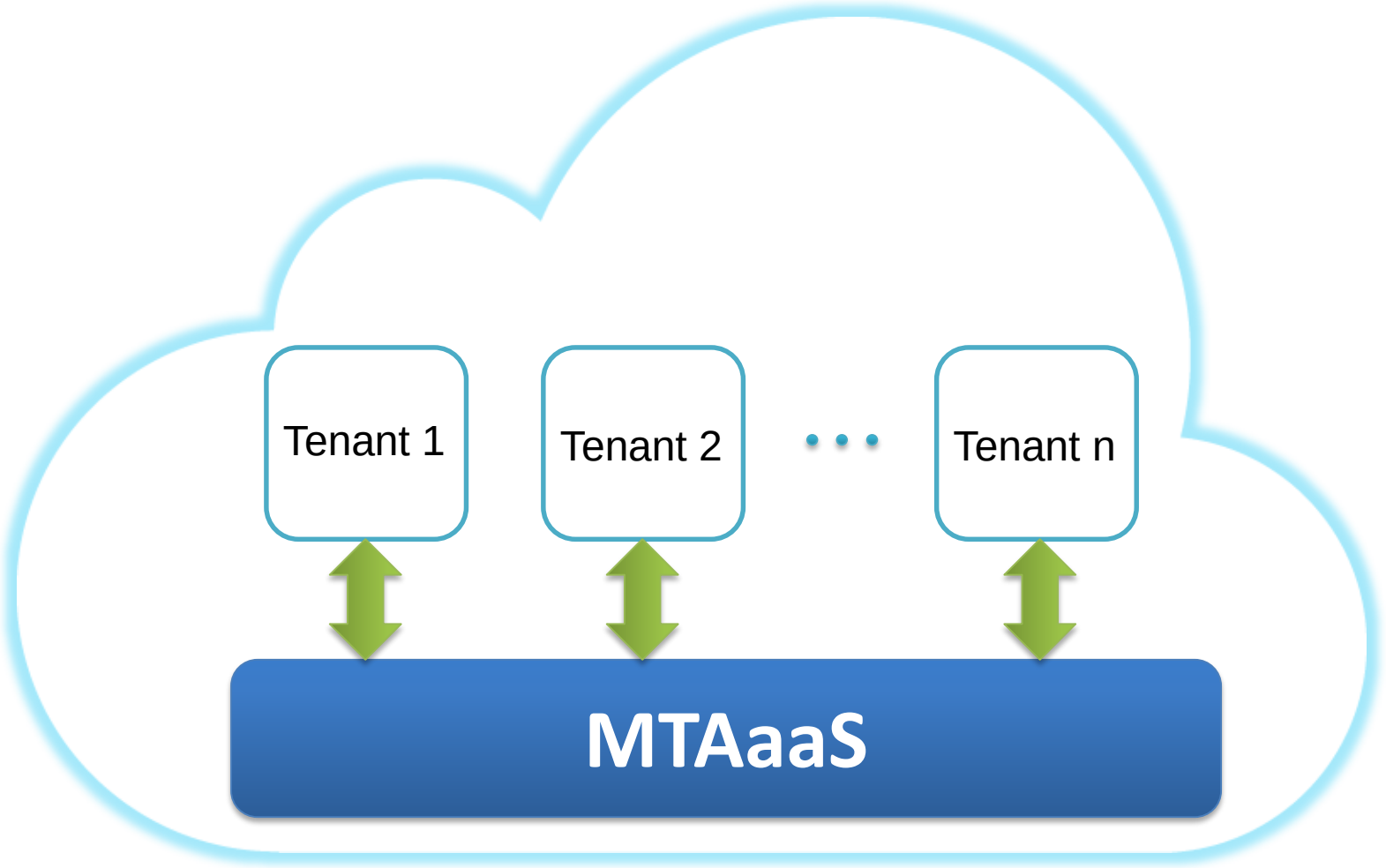


Role-Based CTTM

OpenStack Identity + RH v2.0

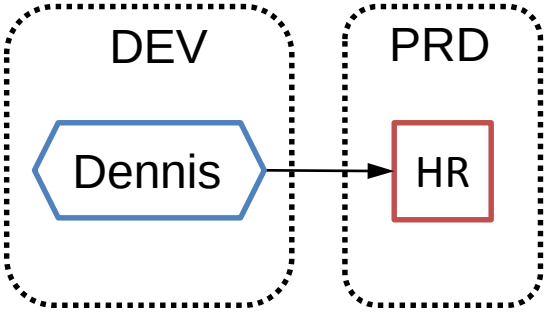


Multi-Tenant Authz. as a Service (MTAaaS)



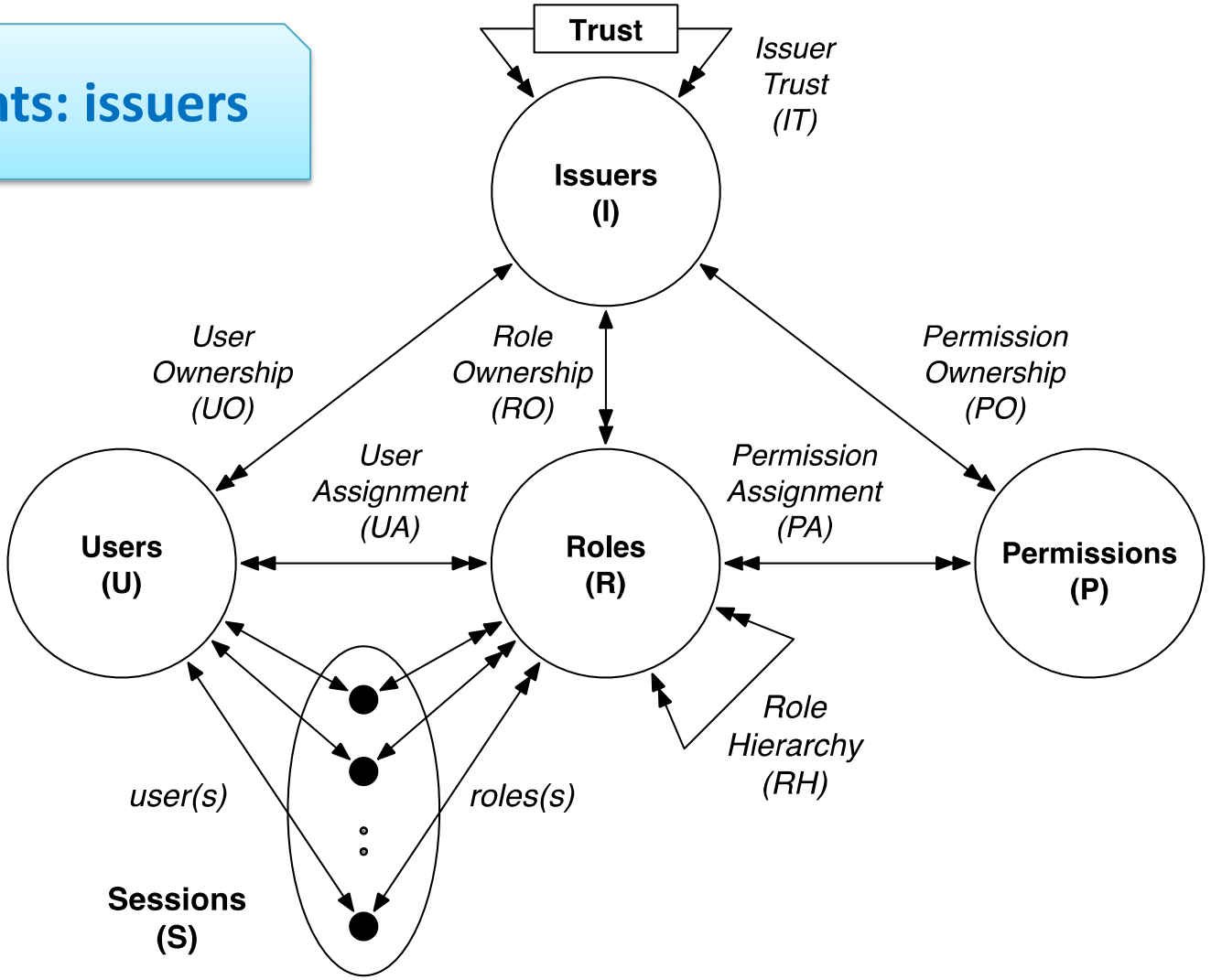
Cross-Tenant Trust in DevOps

- Example: Temporary DevOps access
  - [$]: grant Dennis@DEV access to HR.PRD
  - Trust-α (RT):
    - PRD trusts DEV so that PRD can say [$].
  - Trust-β (MTAS):
    - DEV trusts PRD so that PRD can say [$].
  - Trust-γ (MT-RBAC):
    - PRD trusts DEV so that DEV can say [$].



MTAS

Tenants: issuers



AMTAS

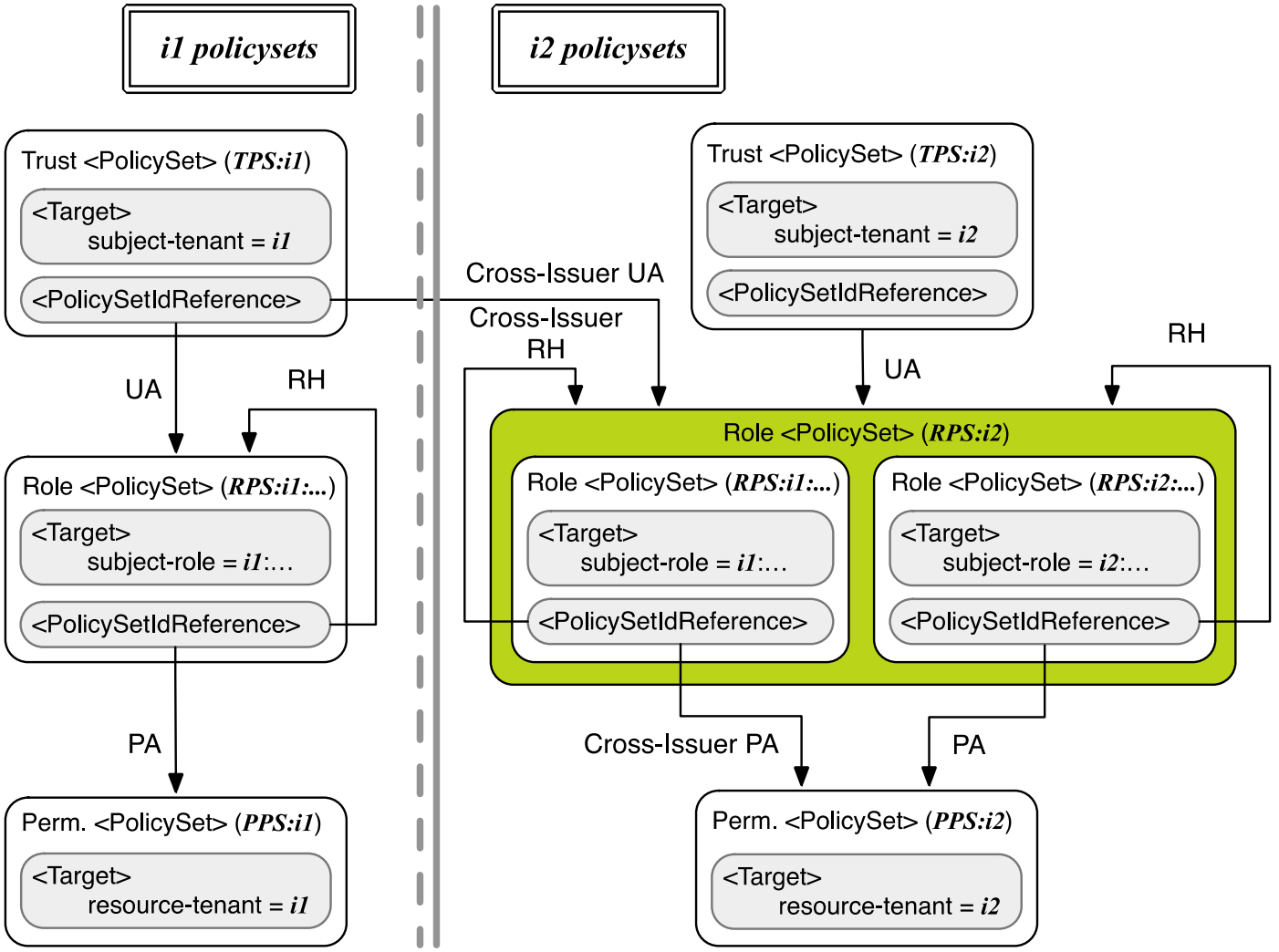
- Issuers are administered by the CSP
- Each issuer administer:
  - trust relations with other issuers
  - entity components: users, roles and permissions
  - UA, PA and RH assignments
    - Cross-tenant assignments are issued by the trustee
      - UA: trustor users to trustee roles
      - PA: trustee permissions to trustor roles
      - RH: trustee roles junior to trustor roles

Finer-grained Trust Models

- Problem of MTAS trust model
  - Over exposure of trustor's authorization information
- Trustor-Centric Public Role (TCPR)
  - Expose only the trustor's public roles
- Relation-Centric Public Role (RCPR)
  - Expose public roles specific for each trust relation

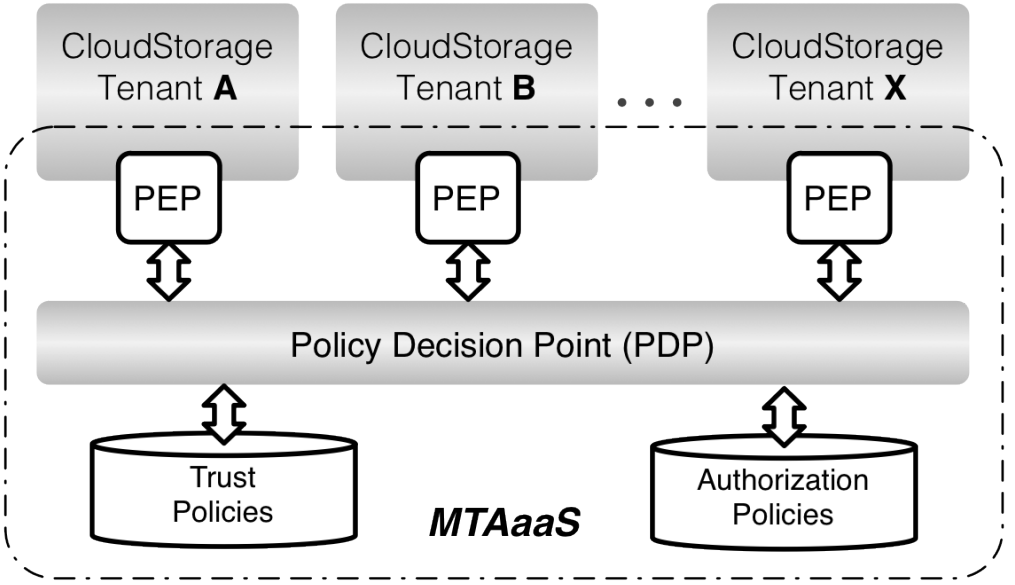
Example MTAS policy structure

i1 trust-β i2



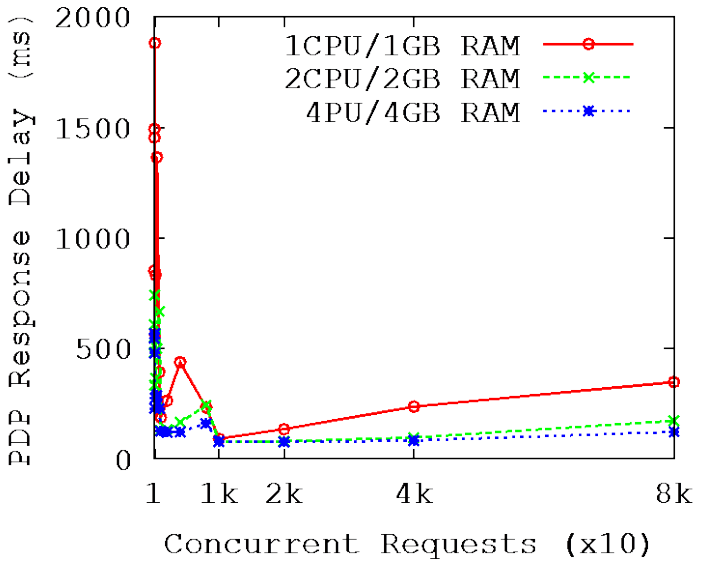
MTAaaS Platform Prototype

- Experiment Settings
  - CloudStorage: an open source web based cloud storage and sharing system.
  - Joyent, FlexCloud
- Authorization Service
  - Centralized PDP
  - Distributed PEP

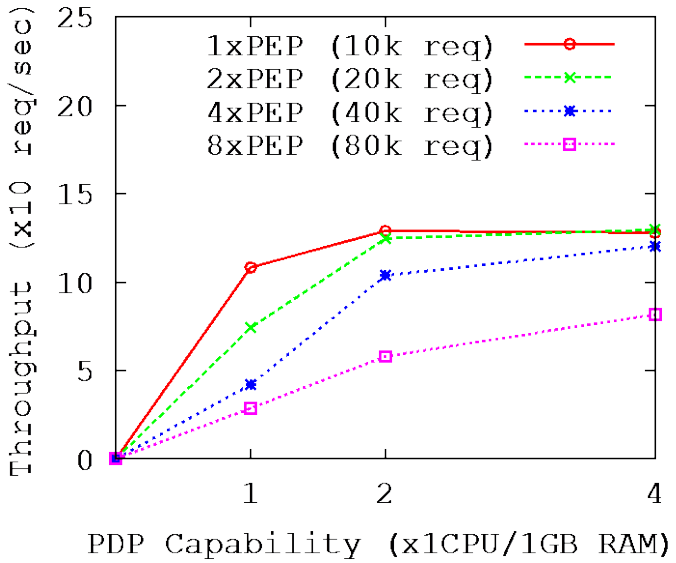


Evaluation

- MTAS introduces ≈12 ms overhead in average.
- Scalable
  - Capability proportional to throughput



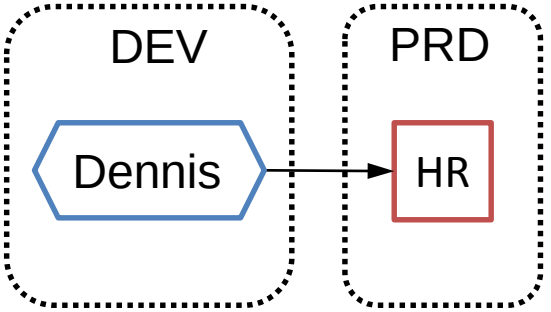
Performance



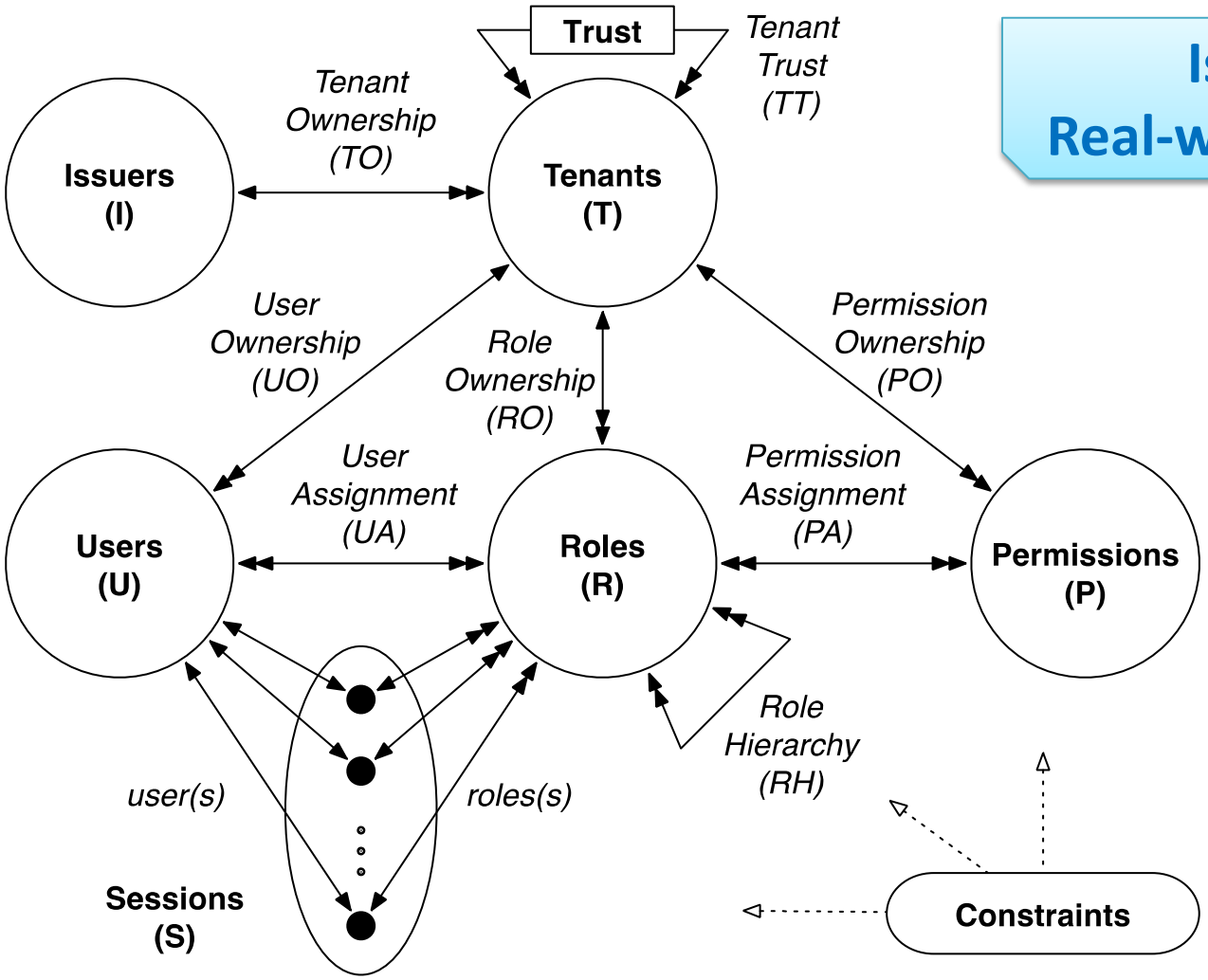
Scalability

Cross-Tenant Trust in DevOps

- Example: Temporary DevOps access
  - [$]: grant Dennis@DEV access to HR.PRD
  - Trust-α (RT):
    - PRD trusts DEV so that PRD can say [$].
  - Trust-β (MTAS):
    - DEV trusts PRD so that PRD can say [$].
  - Trust-γ (MT-RBAC):
    - PRD trusts DEV so that DEV can say [$].



MT-RBAC



Issuers:
Real-world admins

AMT-RBAC

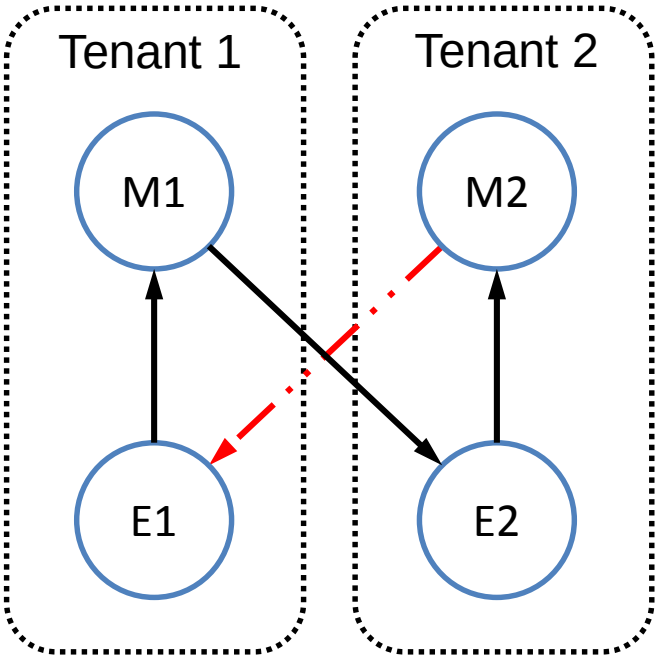
- Issuers administer tenants
- Each issuer administer:
  - Trust relations from owned tenants
  - Entity components: tenants, users, roles and permissions
  - UA, PA and RH assignments
    - Cross-tenant assignments are issued by the trustee's owning issuer
      - UA: trustee users to trustor roles
      - PA: trustor permissions to trustee roles
      - RH: trustor roles junior to trustee roles

Finer-grained Trust Models

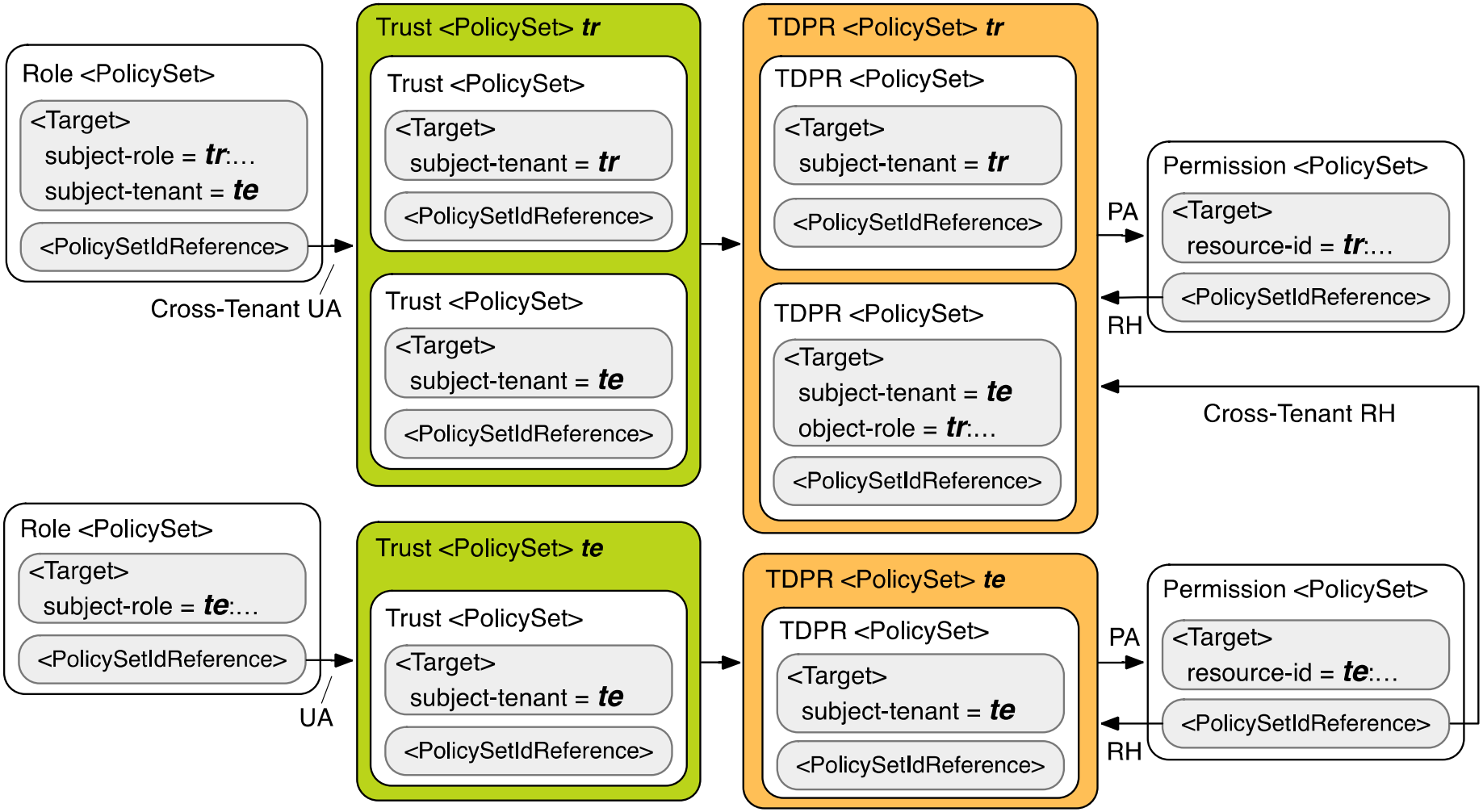
- Trustee-Independent Public Role (TIPR)
  - Expose only the trustor's public roles
- Trustee-Dependent Public Role (TDPR)
  - Expose public roles specific for each trustee

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
  - E.g.: do not share infrastructure with competitors.

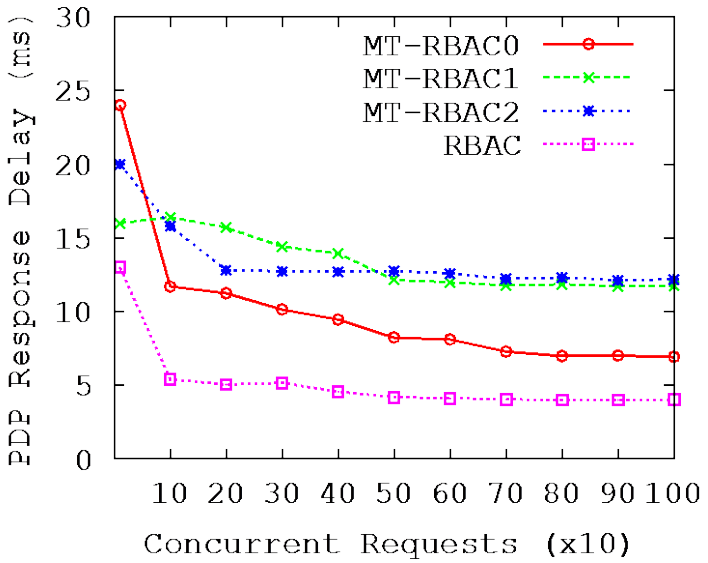


Policy Specification of MT-RBAC

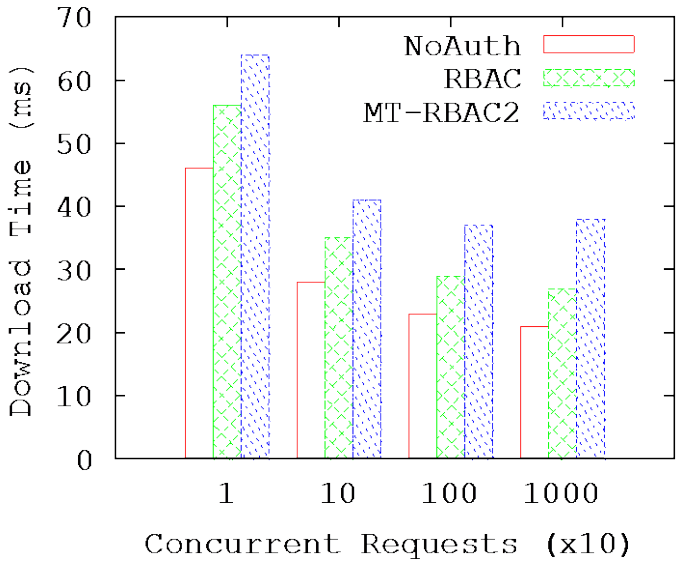


Evaluation: Performance

- MT-RBAC vs RBAC
  - More policy references incur more decision time
- MT-RBAC$_2$ introduces 16 ms overhead in average.



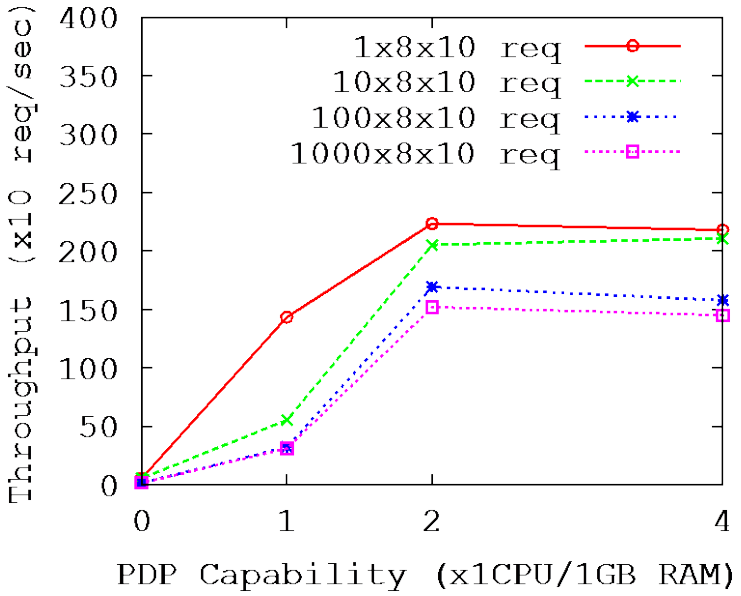
PDP Performance



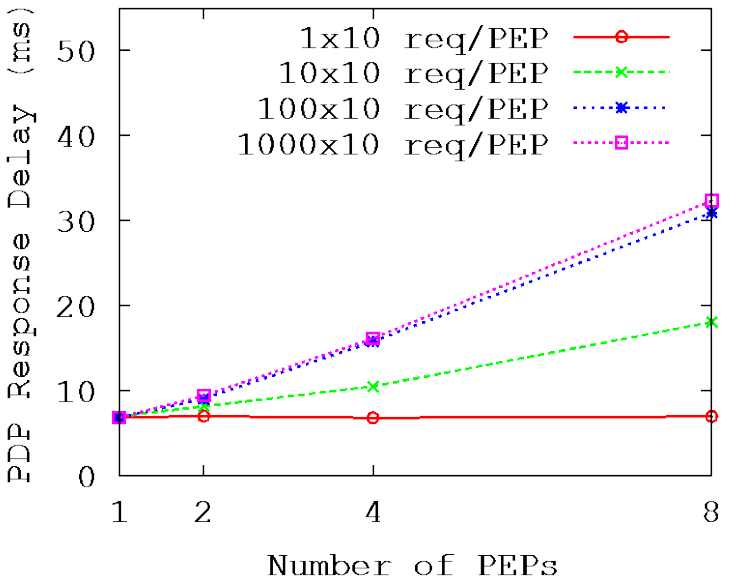
Client-End Performance

Evaluation: Scalability

- Scalable by either
  - Enhancing PDP capability; or
  - Increasing PEP amount.



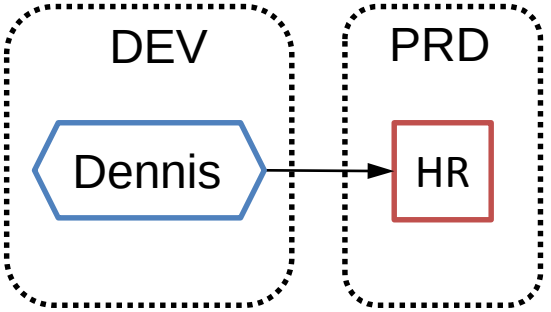
PDP Scalability



PEP Scalability

Cross-Tenant Trust in DevOps

- Example: Temporary DevOps access
  - [$]: grant Dennis@DEV access to HR.PRD
  - Trust-α (RT):
    - PRD trusts DEV so that PRD can say [$].
  - Trust-β (MTAS):
    - DEV trusts PRD so that PRD can say [$].
  - Trust-γ (MT-RBAC):
    - PRD trusts DEV so that DEV can say [$].

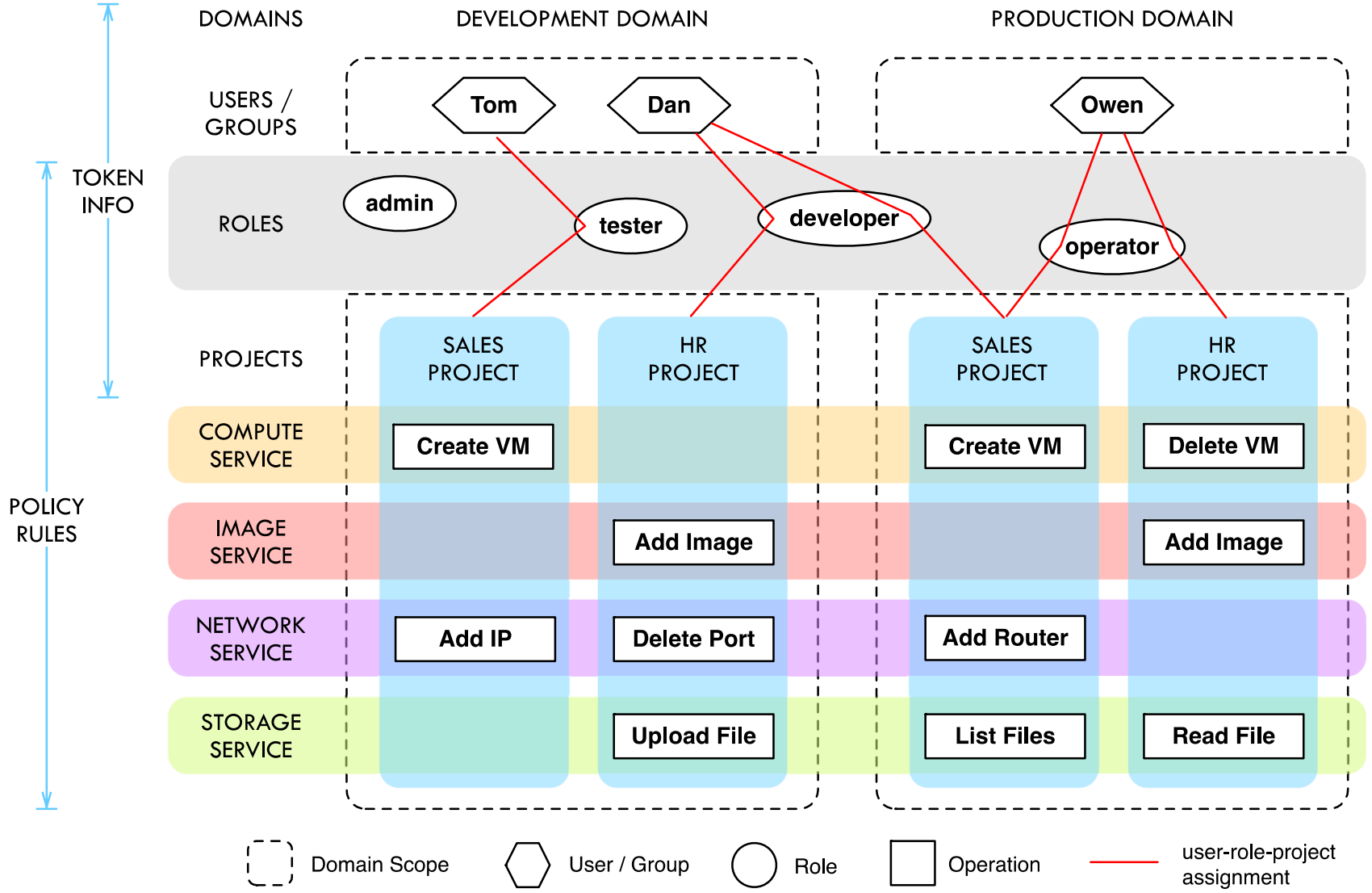


MT-RT

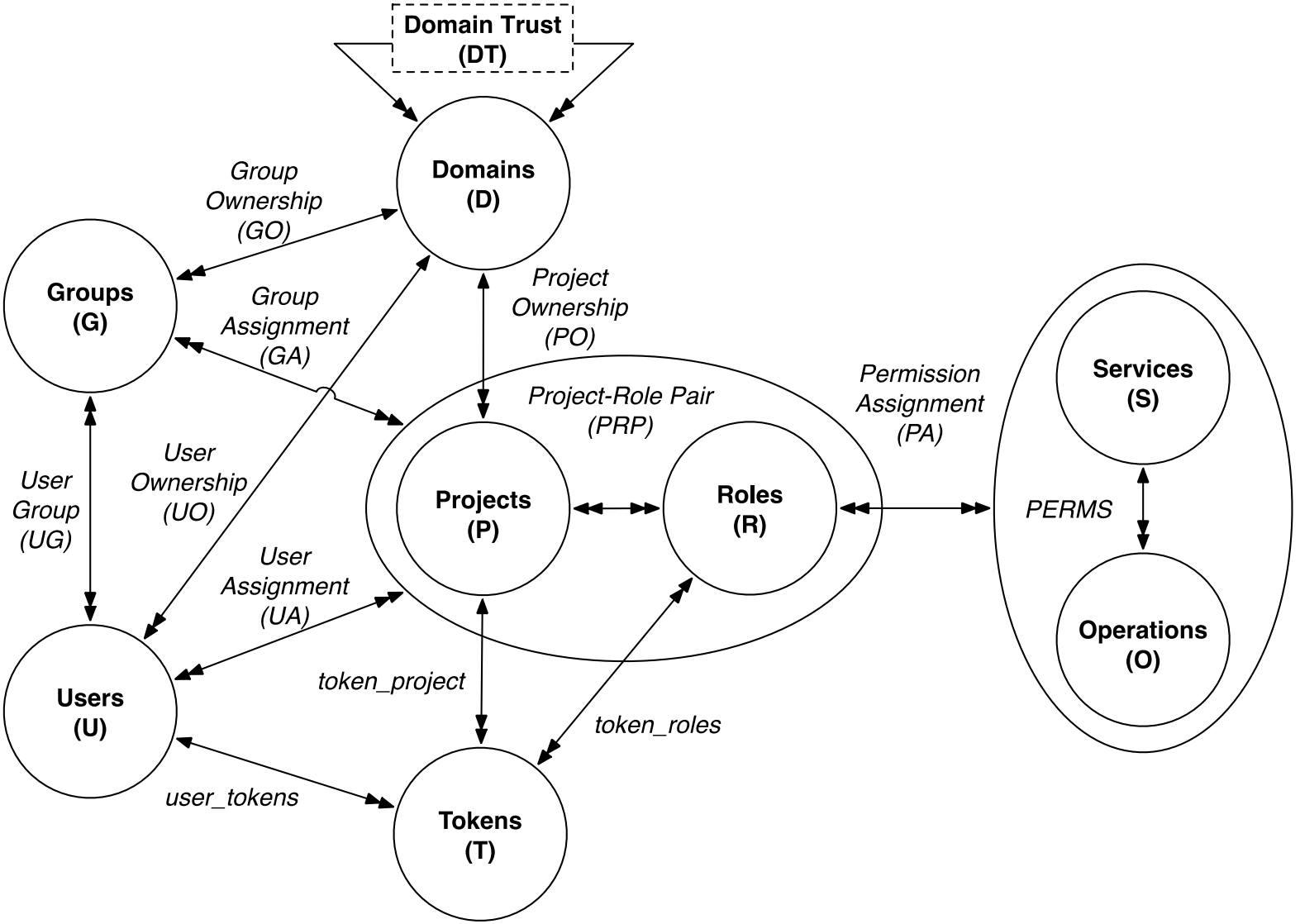
- MT-RT: “P” layer model of RT with MT features
  - No certificate is required (centralized facility)
  - Trust (delegation) in OpenStack identity?

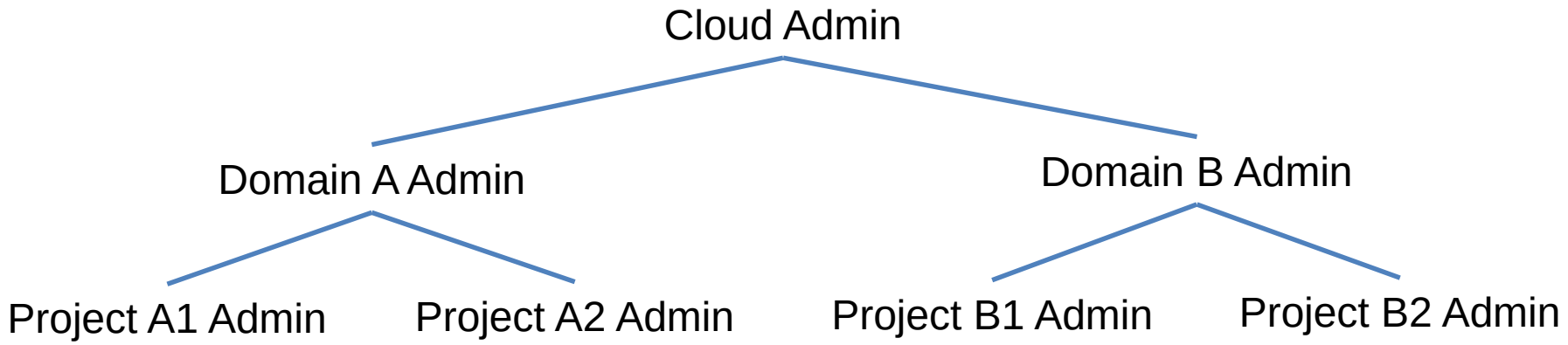
TBR

DevOps in OpenStack



OSAC


Domain Trust
(DT)
Domains
(D)
Group
Ownership
(GO)
Groups
(G)
Group
Assignment
(GA)
Project
Ownership
(PO)
Project-Role Pair
(PRP)
Permission
Assignment
(PA)
Services
(S)
User
Group
(UG)
User
Ownership
(UO)
Projects
(P)
Roles
(R)
PERMS
User
Assignment
(UA)
Operations
(O)
token_project
token_roles
Users
(U)
user_tokens
Tokens
(T)




```
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

Domain Trust in OpenStack

- Enhanced security
  - Limit visibility in the specific domain
  - Prevent malicious / dumb assignments
- Better management with Dtrust
  - Specified by domain admin available for project admin
  - Automatic revocation of cross-domain assignments
  - Finer-grained control enabled
    - Only expose users with certain roles
    - May specify collaborating users and projects

MT-ABAC

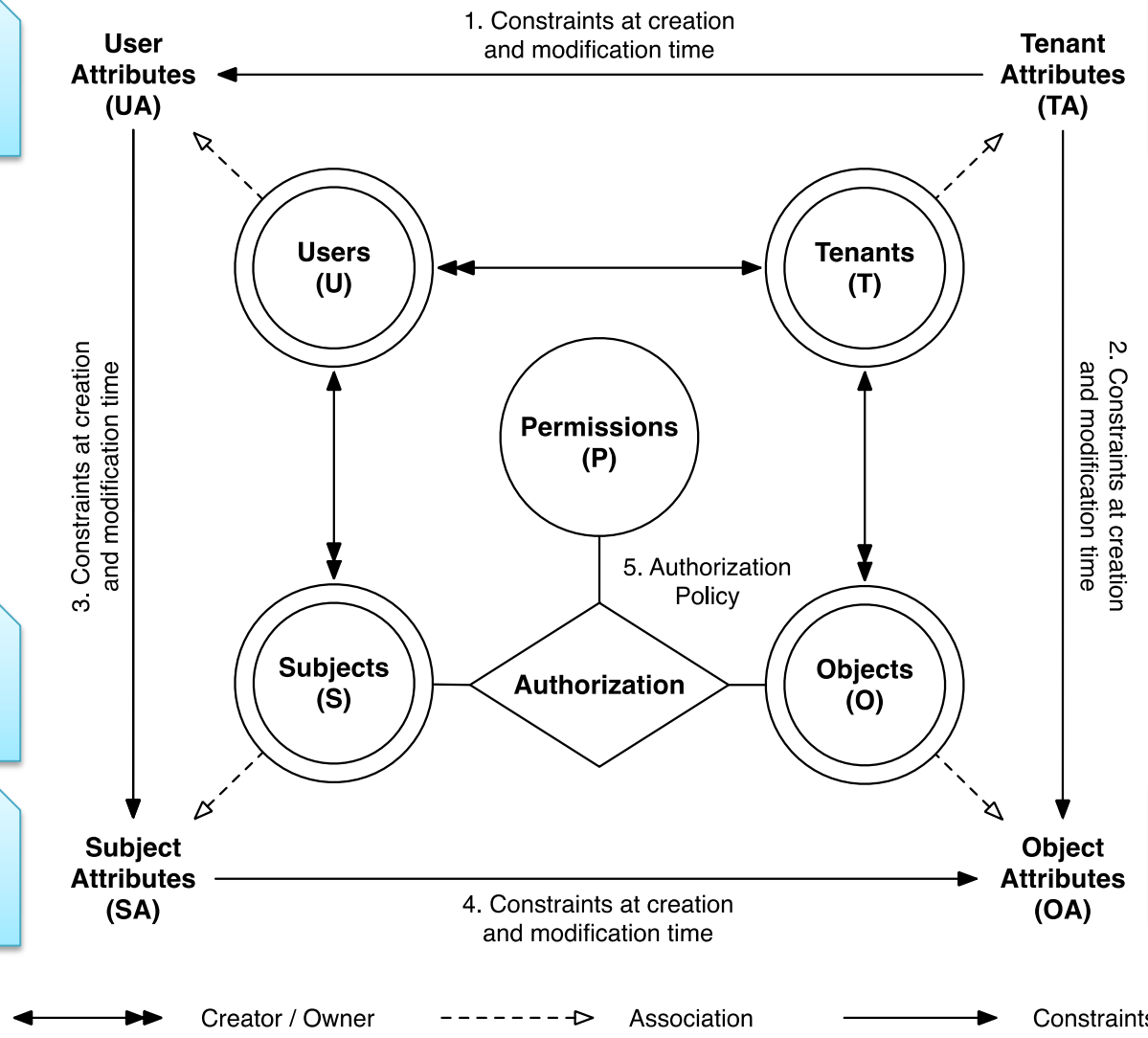
UID: u1
TID: t1

UID: u1
TID: t1

UID: u1
TID: t2

Trustee: {t1,t2}
TrustType: γ
TID: t1

OID: o1
TID: t1



Conclusion

- Completed Work
  - CTTM
  - MTAaaS
  - MTAS
  - MT-RBAC
- On-going research
  - MT-RT
  - MT-ABAC
  - Domain Trust in OpenStack


Q & A


Thank You!