Extending OpenStack Access Control with Domain Trust

Bo Tang and Ravi Sandhu

NSS 2014

The Cloud

Moving to the Cloud

- Driving force:
  - Anytime, Anywhere (Centralized infrastructure)
  - [$$$] -----> [$|$|$] (Shared resources)
  - Pay-on-the-go (On-demand services)
  - Scalable and flexible
- Resistance:
  - Security & Privacy
    - Data governance
    - Access control
  - Other problems
    - Data Locked-in
    - Lack Standard APIs

What is OpenStack?

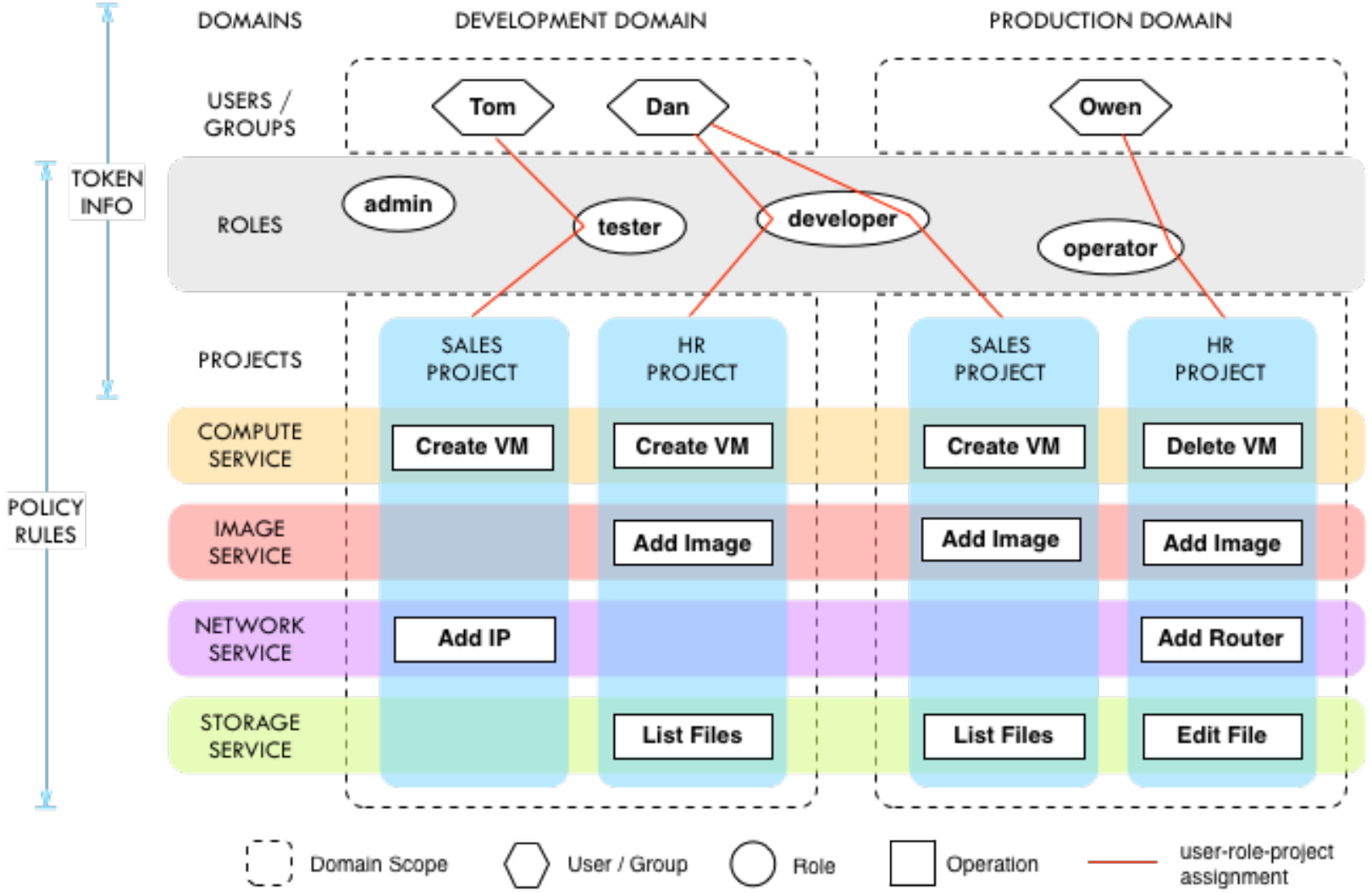
- Open source Cloud platform
  - 12,000 individual members
  - 260 supporting organizations
  - 130 countries
- Havana Release
  - Nov. 2013 - Hong Kong Summit
- Keystone (IAM)
  - Identity API v3
  - Introduction of Domain concept

Source: http://www.openstack.org/software/havana/press-release

Multi-Tenancy

- Multi-tenancy
  - From Cloud Service Provider (CSP) perspective
    - A billing customer, isolated with each other
    - 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.
- Domain in OpenStack
  - Each domain manages its own users and projects

Motivation


DOMAINS
DEVELOPMENT DOMAIN
PRODUCTION DOMAIN
USERS / GROUPS
Tom
Dan
Owen
TOKEN INFO
ROLES
admin
tester
developer
operator
PROJECTS
SALES PROJECT
HR PROJECT
SALES PROJECT
HR PROJECT
POLICY RULES
COMPUTE SERVICE
Create VM
Create VM
Create VM
Delete VM
IMAGE SERVICE
Add Image
Add Image
Add Image
NETWORK SERVICE
Add IP
Add Router
STORAGE SERVICE
List Files
List Files
Edit File
Domain Scope
User / Group
Role
Operation
user-role-project assignment

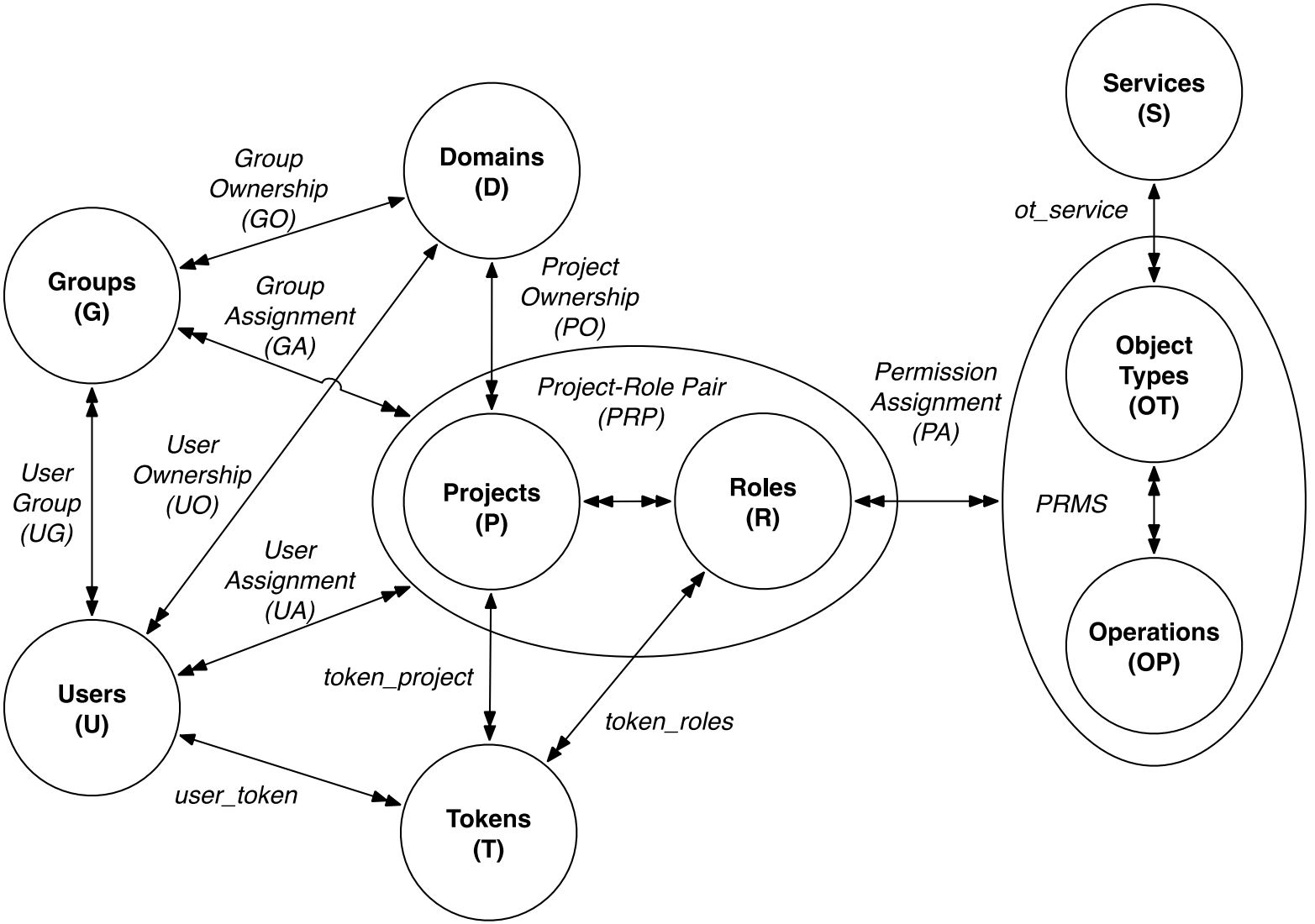
Existing Approaches

- Trust
  - Active Directory Federation Service (AD FS)
    - Multiple types of federation trust among domains
  - Cross-account trust in AWS
    - Unilateral trust with another account or external credentials
  - Trust in OpenStack
    - User to user delegation via roles

Scope and Assumptions

- Standardized APIs
  - Cross-tenant accesses are functionally available
- Properly authenticated users
- One Cloud Service
  - Of a kind: IaaS, PaaS or SaaS.
  - Multi-tenancy collaboration on a single cloud

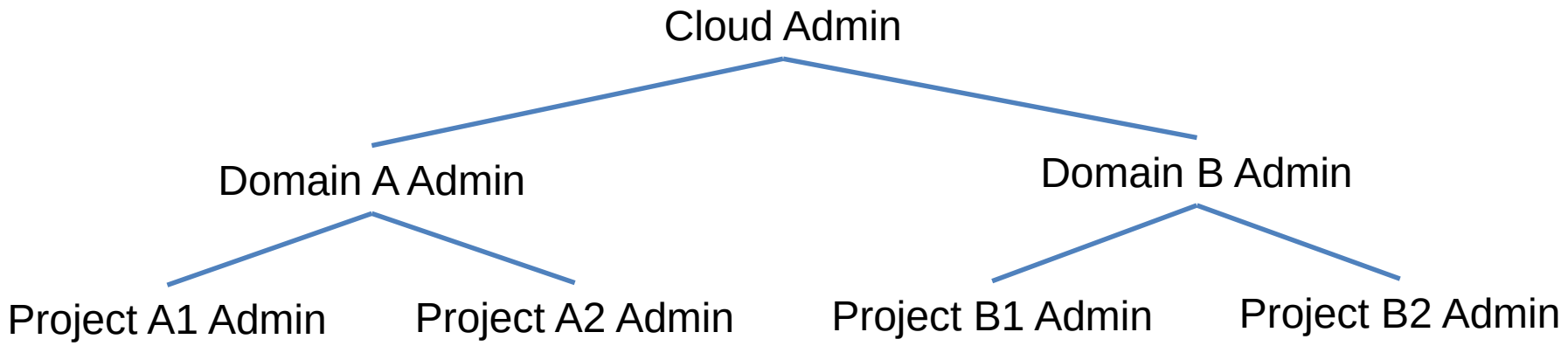
Core OSAC


Services
(S)
Domains
(D)
Group
Ownership
(GO)
ot_service
Groups
(G)
Group
Assignment
(GA)
Project
Ownership
(PO)
Object
Types
(OT)
Project-Role Pair
(PRP)
Permission
Assignment
(PA)
User
Group
(UG)
User
Ownership
(UO)
Projects
(P)
Roles
(R)
PRMS
User
Assignment
(UA)
Operations
(OP)
token_project
token_roles
Users
(U)
user_token
Tokens
(T)

Definitions of OSAC

- Roles
  - Globally available
  - Not owned by domains or projects
- Tokens
  - Credentials issued to authenticated users
  - Will expire, similar to session concept in RBAC
- Services
  - Examples: Nova, Glance, Neutron
  - Different services have different policies based on the role-permission assignments

Administration (AOSAC)




```
rule:add_user_to_project -> (role:keystone_admin ||
   (role:project_admin && project_id:%(target_project_id)s) ||
   (domain_role:domain_admin && domain_id:%(target_domain_id)s))
```

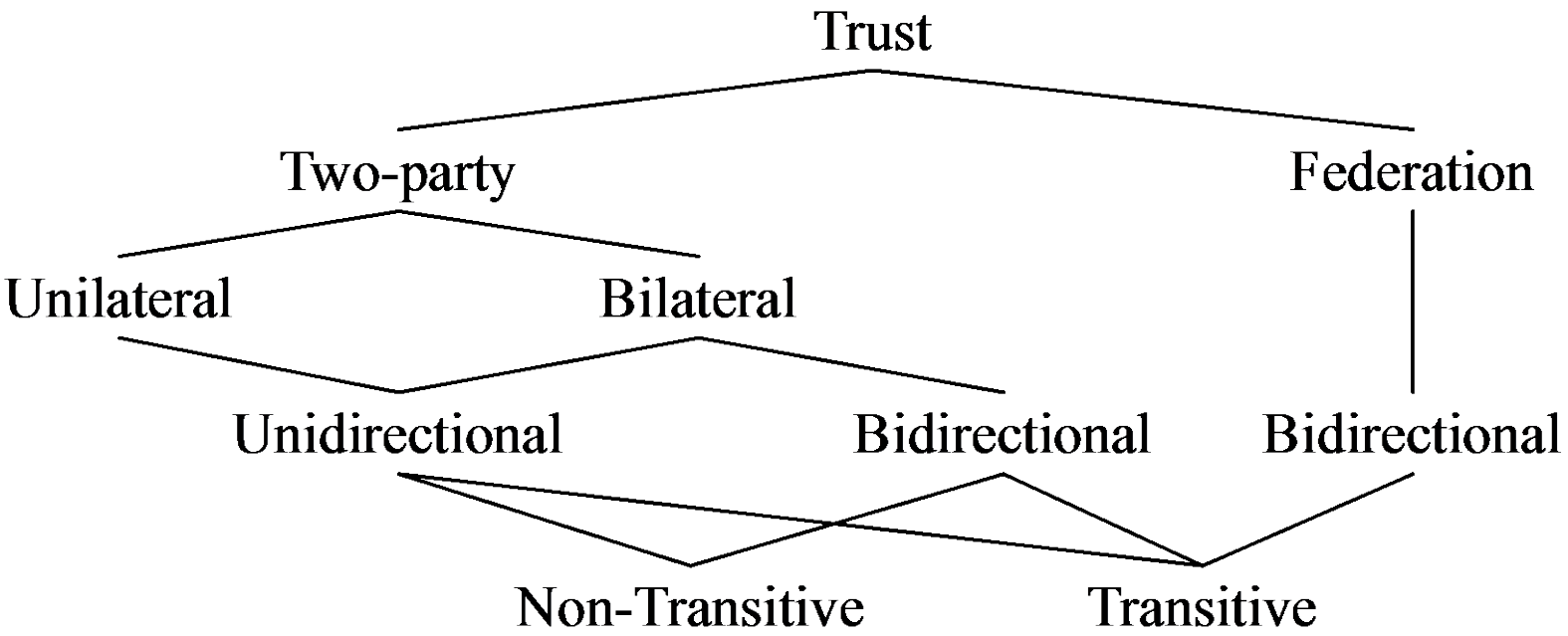
```
rule:add_project_to_domain -> (role:keystone_admin ||
   (domain_role:domain_admin && domain_id:%(target_domain_id)s))
```

Source: https://wiki.openstack.org/wiki/Domains

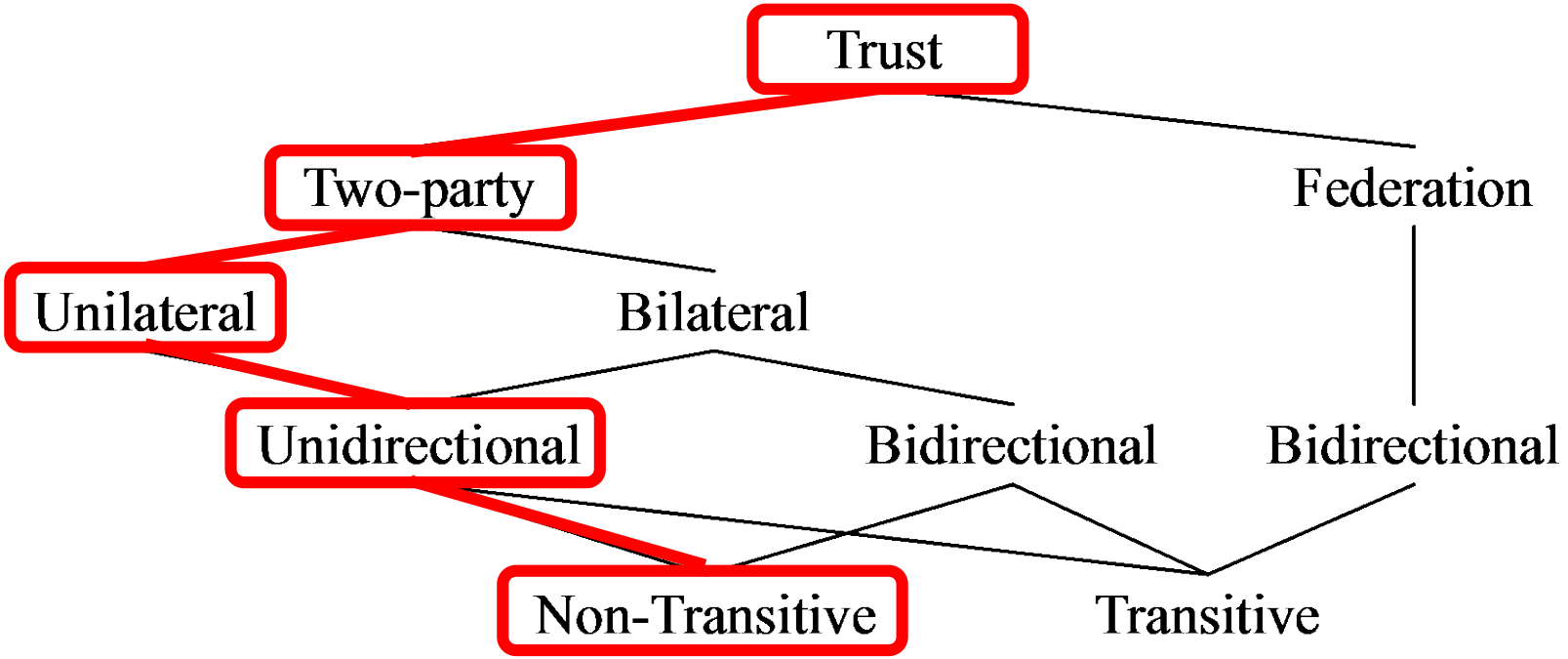
Domain-Level Collaboration

- Basic scenario
  - User: u1 from Domain: d1
  - Project: p2 from Domain: d2
- Cross-domain actions
  - Administrative
    - Assign u1 to roles in p2
  - Operational
    - Allowing u1 to access p2 with the assigned roles
  - Require proper trust relation between d1 and d2

Trust Framework



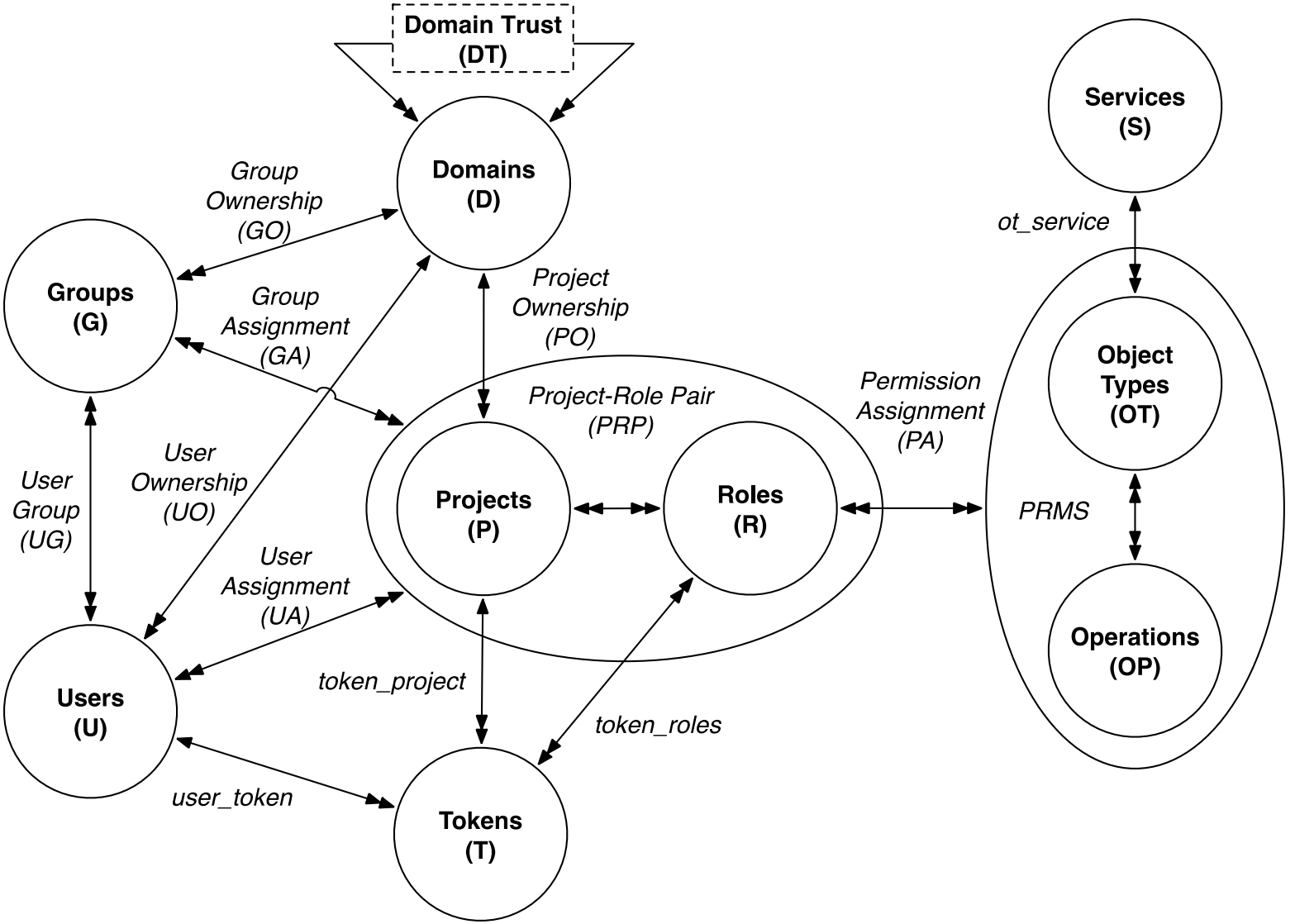
Domain Trust



Trust Types

- Two-party unilateral unidirectional non-transitive
  - Type-α, requires visibility of the trustee's user information for the trustor to assign trustee's users to roles in trustor's projects, written as "$\trianglelefteq_{\alpha}$".
  - Type-β, requires the trustor to expose its user information for the trustee to assign trustor's users to roles in trustee's projects, written as "$\trianglelefteq_{\beta}$".
  - Type-γ, requires the trustor to expose its project information for the trustee to assign trustee's users to roles in trustor's projects, written as "$\trianglelefteq_{\gamma}$".

OSAC-DT


Domain Trust
(DT)
Domains
(D)
Group
Ownership
(GO)
Groups
(G)
Group
Assignment
(GA)
Project
Ownership
(PO)
Project-Role Pair
(PRP)
Projects
(P)
Roles
(R)
Permission
Assignment
(PA)
Services
(S)
ot_service
Object
Types
(OT)
PRMS
Operations
(OP)
User
Group
(UG)
User
Ownership
(UO)
User
Assignment
(UA)
Users
(U)
token_project
token_roles
user_token
Tokens
(T)

Constraints & Administration

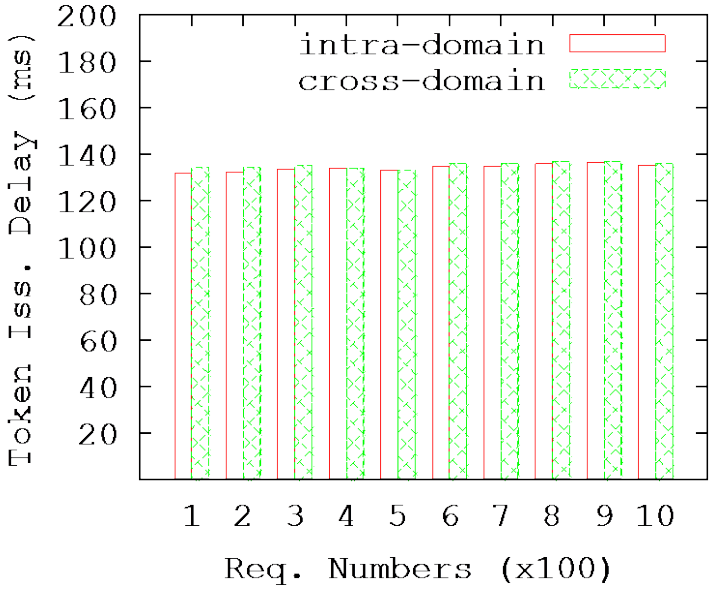
- Constraints
  - Separation of Duties (SoD)
    - Mutually exclusive domain list
  - Minimum Exposure
    - Limit exposure of project and user to other domains
  - Cardinality
    - Limit the number of domains to be trusted
- Domain Trust Administration
  - The trustor manages the trust relation and constraints

Implementation

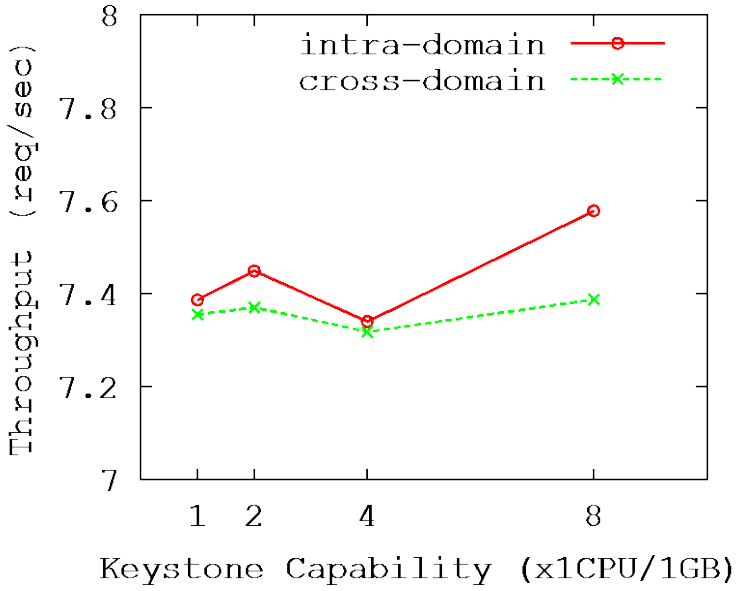
- Type-γ trust
  - Trustee manages cross-domain assignments
  - Implemented as a extension module in Keystone
- Experiment Environment
  - 1 unit = 1 CPU/1GB
  - VMs with 1, 2, 4, 8 units of capability
  - Devstack deployed in cloud environment
  - Stand-alone Keystone service
  - Test with REST API calls through curl commands

Prototype & Evaluation

- Sequential request handling (Queuing)
  - Domain trust introduces 0.7% authz. Overhead
  - Scalability changes little with domain trust



Performance



Scalability

Related Work

- RBAC extensions
  - Centralized authority is usually required
    - ROBAC, collaboration not supported
    - GB-RBAC, group does not own users
- Role-Based Delegation models
  - Delegation chain lacks support of agile entities
- Multi-Domain Interoperation
  - Role-mapping requires PA to be domain-specific
- Multi-Tenant Access Control models
  - MTAS, MT-RBAC, CTTM

Conclusion & Future Work

- Formalized OSAC model
  - Administrative model (AOSAC)
- Trust Framework & Trust Types
- Formalized OSAC-DT model
  - Administrative model (AOSAC-DT) & Constraints
- Implementation & Experiments in OpenStack
  - Acceptable performance & scalability change
- Future work
  - Hierarchical Multi-tenancy model
  - Attribute-based models
  - Implementation in future OpenStack

Acknowledgements

- Dolph Mathews
  - PTL of Keystone
- Farhan Patwa
  - Director of ICS
- Jaehone Park
  - Research Associate Profession in ICS


Q & A


Thank You!