Multi Cloud IaaS with Domain Trust in OpenStack

Navid Pustchi
Institute for Cyber Security
Dept. of Computer Science
Univ of Texas at San Antonio
tam498@my.utsa.edu

Farhan Patwa
Institute for Cyber Security
Univ of Texas at San Antonio
farhan.patwa@utsa.edu

Ravi Sandhu
Institute for Cyber Security
Dept. of Computer Science
Univ of Texas at San Antonio
ravi.sandhu@utsa.edu

ABSTRACT

As cloud services have been firmly accepted by enterprises, the current challenge is how to share these resources among increasing number of cloud platforms. Currently, cloud platforms such as OpenStack, the *de facto* open-source platform for cloud Infrastructure-as-a-Service (IaaS), offer limited cross-cloud access capabilities in their federation APIs. In this paper, we present a fine-grained cross-cloud domain-trust model enabling resource sharing between domains across distinct homogeneous clouds. We further present a formalized description of core multi-cloud OpenStack access control (MC-OSAC) with proposed domain trust extension. We have implemented a proof of concept with extending OpenStack identity and federation services to support cross-cloud domain trust. Our approach does not introduce any authorization overhead within current OpenStack federation model.

CCS Concepts

•**Security and privacy** → **Access control;**

Keywords

Multi Cloud; Multi Domain; Federation; Access control; Trust Management

1. INTRODUCTION

Cloud federation is a promising mechanism to share resources across multiple public or private clouds in order to fulfill demanding IT portfolio of enterprises. Cloud service provider (CSP) lock-in as one main drawbacks of cloud adoption by industry can be avoided by federation strategy. Federation allows use of multiple cloud providers while data and applications reside in distinct clouds with different security or privacy measures. Cloud federation offers greater resource pooling, flexibility and dynamicity for organizations. In this scenario, single or multiple collaborating organizations aim to share resources dispersed across multiple CSPs. OpenStack [2] and Amazon Web Services (AWS) [1] as leading cloud platforms currently support coarse-grained federation mechanisms. However, federation administration in terms of relation establishment and federated-user assignments to resources across domains are not supported.

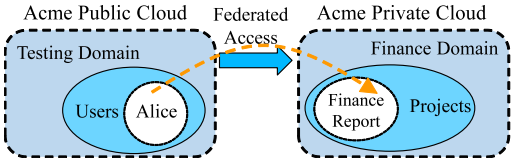

Figure 1: Cross-Cloud Domain Federation.

In this paper, we present a fine-grained mechanism to establish trust between domains across clouds and enable domain administrators to manage trust and user assignment. Our contribution is scoped within federation in homogenous cloud IaaS platforms. In contrast to other cloud federation [6] models, our scope of contribution is to extend OpenStack with domain-trust federation.

To motivate the problem, we use an enterprise such as Acme in figure 1 which stores it's financial data in Acme's private cloud and development applications in Acme's public cloud. Finance domain hosts finance projects in private cloud while Testing domain in public cloud hosts software developer users. Alice as a software developer is working on reports which finance data access is necessary. For security and privacy reasons financial data should not be transferred to other domains. Meanwhile, for administrative reasons it is impractical to provision or assign other domains' users permanently in Finance domain. The practical approach is for Testing domain administrator to establish a relation between two domains in which Finance domain administrator is authorized to assign Alice to Finance report projects. Upon task completion Finance domain administrator can remove user assignments. Testing domain administrator can remove federated relation with Finance domain at any time.

2. BACKGROUND

OpenStack is an open-source cloud IaaS platform consisting of RESTful API services such as Nova (Compute), Keystone (identity), Neutron (networking), and so on. Keystone is both the authentication and authorization service in OpenStack. OpenStack access control model consists of entities such as users, groups, projects, domains, and roles.

CODASPY'16 March 09-11, 2016, New Orleans, LA, USA

ACM ISBN 978-1-4503-3935-3/16/03.

DOI: http://dx.doi.org/10.1145/2857705.2857745

Users are individuals authenticated to Keystone while each group is a set of users. Projects define a container of cloud resources such as virtual machines, storage, and etc. Domain is an administrative boundary which owns users, groups, and projects. Each cloud consist of multiple domains representing an organization (in public clouds), a department of an organization, or an individual who uses cloud services. Roles are global within a cloud boundary. Users and groups are both assigned to roles within project or domain scope. Currently OpenStack supports cloud admin and domain admin roles whereas domain admin can only administer within it's residing domain.

As of Kilo release OpenStack federation API supports SAML assertions [5] (Keystone generates and consumes assertions). Specifically Keystone to Keystone federation is supported which it allows authenticated users (Keystone as identity provider) to swap their token for a SAML assertion. This assertion is redirected to another Keystone (Keystone as service provider) to get a token back from the second Keystone. Federated user is then mapped to a user or group from second Keystone and based on assigned roles, it can request a project scoped token. Such coarse-grained model does not allow domain administrators to manage user and resources shared within federated clouds.

3. MULTI-CLOUD OSAC MODEL

In multi-cloud model, cross-domain access is granted upon domain-trust between two domains across distinct clouds.[1] In addition to OSAC model [8], federation relation is represented by trustor clouds (identity provider) and trustee clouds (service provider) which is a many-to-one relation. Identity providers are set of clouds which trust to federate their users to the current cloud. Similarly service providers are set of clouds in which current cloud trust to federate its users to their resources. For example in figure 1 for Acme Private Cloud, Public Cloud is an identity provider and in Acme Public Cloud, Private Cloud is a service provider. In MC-OSAC, administrative model consists of two levels of administrative roles: *cloud-admin* and *domain admin*. *Cloud-admin* refers to cloud-level administrators managing all cloud identity service components. *Domain-admin* is administrator role at domain-level which manages components within its associated domain.

In MC-OSAC we enable domain-trust by remote assignment which is administered by *domain-admin*. Mapping rules define a set of accepted remote users or groups to local domain users and groups. Federated users are mapped to local users and groups by remote mapping in a many-to-one relation as it is depicted in figure 2. We define domain-trust in MC-OSAC as a many-to-many relation between domains in federated clouds.

Trust properties characterizes which *domain-admin* controls trust relation or cross-domain assignments. Our trust type is specifically motivated by previously defined type-β trust [7]. We define type-β domain-trust as:
If $domain_A \trianglelefteq_\beta domain_B$ *(*$\trianglelefteq$ *is trust notion),* $domain_B -$ *admin is authorized to assign* $domain_A$ *users to its projects.* $Domain_A -$*admin controls trust relation.*[2] In trust relation, $domain_A$ is trustor domain and $domain_B$ is trustee domain.

Recall our Acme federation example in figure 1, Testing *domain-admin* establishes type-β domain-trust between Finance domain. Finance *domain-admin* can assign Alice to set of projects in its domain and upon collaboration completion Finance *domain-admin* removes assignments. Testing *domain-admin* can remove trust relation which results in removing all assignments of its users to projects in Finance domain. This approach enables collaborating domains to form collaboration relation dynamically while upon task completion, they can end collaboration with ease.

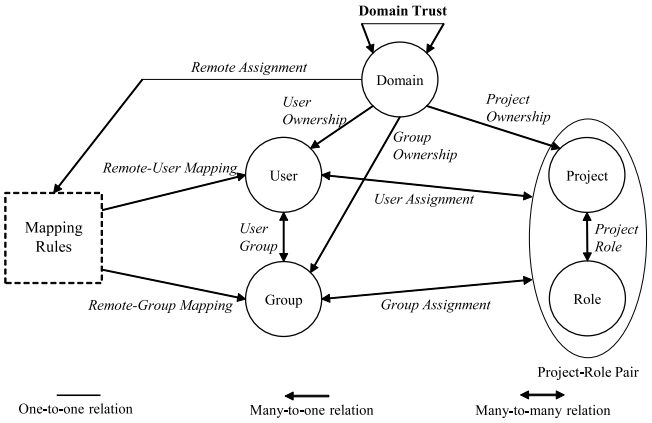

Figure 2: Multi Cloud OpenStack Access Control Model (MC-OSAC) with Domain-Trust.[4]

4. IMPLEMENTATION

In this section, implementation considerations are discussed. Keystone as authentication and authorization service is organized as a group of internal services exposed on one or many endpoints. Such internal services are identity, assignment, resource, token, policy, catalog, and federation. In order to deploy our model in OpenStack platform, we modified assignment, policy, and federation internal services in Keystone. We assumed each OpenStack Cloud belongs to a single organization while domains represent departments of each organization. In the case of collaboration, multiple domains from distinct OpenStack clouds share resources within specified domain-trust. In our model domain-trust relation is stored in the trustee cloud as *federation-domain-trust* table. Figure 3 depicts entries for such table. Trust relation is stored as remote-domain name, local-domain id, and trust-type.

We implemented the model with two OpenStack DevStack instances. To establish domain-trust and cross-domain user assignments, policy service and policy module was modified to enable *domain-admin* assignments. In assignment service, we added methods to restrict assignments to trusted domains. To enable remote assignments, in federation service we added methods to enable and restrict remote mapping (user and group) only within residing *domain-admin*'s domain.

In our model, cross-domain trust establishment steps are

[1]Since our contribution is at domain-level, we do not focus on cloud federation at cloud-level.

[2]We use type-β for simplicity, other types of trust as α, γ, and δ are also applicable to our model [6].

[4]It is slightly different from OSAC model where Tokens, PRMS, and Services are omitted due to our scope of contribution.

remote_domain	local_domain	trust_type
Default	default	beta
domain1	default	beta

Figure 3: Federation Domain-Trust Table.

as follows.
1. A *domain-admin* in trustor cloud initiates trust relation by selecting a trusted cloud from service provider list (note that prior to scoping a token with a project, federated users can list domains with an unscoped token in OpenStack federation).
2. *Domain-admin* token is swapped with SAML assertions and redirected to service provider.
3. *Domain-admin* is mapped to *remote-domain-admin* role which is enabled to modify *federated-domain-trust* table, then *domain-admin* creates the trust relation.

In case of trust relation deletion, Keystone removes all mappings matched with remote-domain name. The cross-domain remote assignment establishment steps are as follows.
1. A *domain-admin* in trustee cloud initiates remote assignment by selecting a trusted-cloud's domain from identity provider list and *federated-domain-trust* table.
2. *Domain-admin* creates remote assignment of trustor-cloud's domain to its user or groups.
3. *Federated-domain-trust* table is checked for proper trust relation as well as user or group domain, afterwards remote assignments are created.

In case of removing remote assignments, *domain-admin* can only remove mappings it has created. Figure 4 shows the flow to establish and delete a trust relation and remote assignments in MC-OSAC model. After remote assignments is created, incoming federated users from trustor-cloud's domain are mapped to specified roles dependent on users and groups assigned based on mapping rules. Within project-role pair permission in OpenStack, federated users can request project scoped token to access authorized resources.

5. RELATED WORK

We categorize previous research into collaboration with domain-trust and federation. In [7], authors describe a domain-trust within a single cloud and extensions to OpenStack is described in [8]. In federation, focus has been on authentication federation such as [3] and [4] which identity federation in OpenStack is presented. In authorization federation, recent work discussed use of domain trust within multi-cloud environment to enable collaboration, as authors stated in [6].

Presented model in this work is proposed with focus on cloud IaaS platforms and compatibility with current federation approaches. Our model is more similar to [6], from where we adapt domain-trust concept and realize it to extend OpenStack cloud IaaS platform.

6. CONCLUSIONS

We presented and implemented a multi-cloud model with domain-trust for resource sharing, where collaboration is enabled through cross-domain user-role assignments. Our approach considers applicability to other types of trust beyond presented trust. For future, we plan to investigate and implement other trust types. Finally as future research, we envision to develop multi-cloud models to incorporate attributes within current cloud platforms.

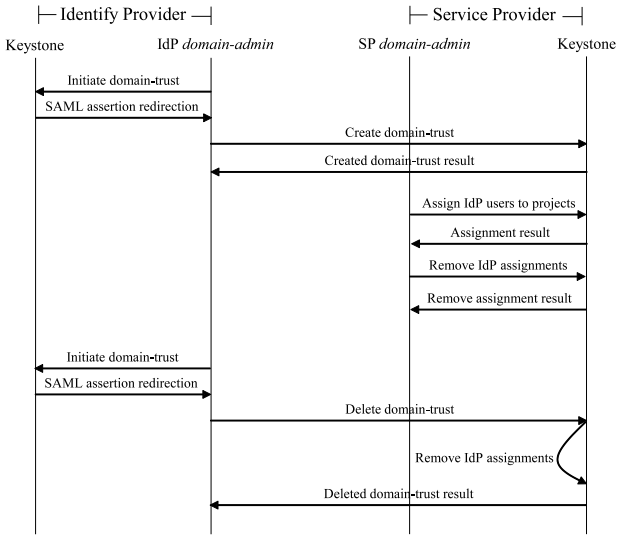

Figure 4: Cross-Domain Trust Establishment and Assignment Process.

Acknowledgement

This research is supported by NSF Grant CNS-1111925 and CNS-1423481.

7. REFERENCES

[1] Amazon Web Services. http://aws.amazon.com/es/ec2.
[2] OpenStack. http://www.openstack.org/.
[3] D. W. Chadwick. Federated identity management. In *Foundations of security analysis and design V*, pages 96–120. Springer, 2009.
[4] D. W. Chadwick, K. Siu, C. Lee, Y. Fouillat, and D. Germonville. Adding federated identity management to OpenStack. *Journal of Grid Computing*, 12(1):3–27, 2014.
[5] J. Hughes and E. Maler. Security assertion markup language (saml) v2.0 technical overview. *OASIS SSTC Working Draft sstc-saml-tech-overview-2.0-draft-08*, pages 29–38, 2005.
[6] N. Pustchi, R. Krishnan, and R. Sandhu. Authorization federation in IaaS multi cloud. In *Proc. of Security in Cloud Computing*, pages 63–71. ACM, 2015.
[7] B. Tang and R. Sandhu. Cross-tenant trust models in cloud computing. In *Proc. of Int. Conf. IRI*, pages 129–136. IEEE, 2013.
[8] B. Tang and R. Sandhu. Extending OpenStack access control with domain trust. In *Network and System Security*, pages 54–69. Springer, 2014.