Hierarchical Secure Information and Resource Sharing in OpenStack Community Cloud

Yun Zhang, Farhan Patwa, Ravi Sandhu and Bo Tang
Institute for Cyber Security and Department of Computer Science
University of Texas at San Antonio, San Antonio, TX, USA
Email: amy.u.zhang@gmail.com, farhan.patwa@utsa.edu, ravi.sandhu@utsa.edu, townbull@gmail.com

***Abstract*—Community clouds provide efficient and secure environments for organizations with similar organization structures or business models to host their systems. Since threat analysis and incident response infrastructure and resources can be rapidly shared on a community cloud, the participating organizations save time and cost in handling cyber incidents. Unfortunately, contemporary cloud platforms are lacking a widely accepted access control model for secure information and resource sharing. Following the recent innovation of Hierarchical Multitenancy in OpenStack community, we propose a hierarchical secure information and resource sharing model in the context of an OpenStack community cloud. Our model enables secure and effective management of information sharing in a community cloud for both routine and cyber incident response needs. We believe this model is applicable in community clouds beyond OpenStack as well.**

***Keywords*-Cloud Computing; Hierarchical Multitenancy; Incident Response; Security Information Sharing; OpenStack;**

I. Introduction

Threat analysis and incident response information needs to be shared with collaborative groups formed to handle both potential and existing cyber incidents. The emergence of cloud as a shared infrastructure, significantly improves the efficiency and flexibility of business systems, as well as incident response processes.

The deployment models of clouds can be categorized into public, private, community and hybrid clouds [5]. A public cloud provides services for open use by the general public. A private cloud provides services for exclusive use by a single organization. A community cloud provides services for exclusive use by a specific community, which contains organizations with shared concern, such as mission, security requirements, business models, etc. In some cases, a big corporate group with multiple subsidiaries may own one community cloud for business needs. A hybrid cloud is a composition of multiple distinct clouds, which may be public, private or community clouds. In this paper, we investigate models information sharing in a community cloud constructed using OpenStack cloud platform.

Cyber attacks are becoming increasingly sophisticated and difficult to defend by a single organization on its own. Cyber attacks have resulted in significant economic losses. Determined adversaries and organized cyber criminals are aiming at organizations of all sizes putting their valuable digital information at risk. Establishing cyber incident response mechanisms in an organization improves the decision making process and internal and external coordination, which potentially minimize the damage of cyber incidents. By explicitly designating users and roles who are in charge of security issues associated with organization systems, quick decisions can be made if a cyber attack happens. By explicitly establishing a standard cyber security process, organizations can easily identify the problems, schedule the defense process and prevent themselves from further loss caused by improper handling of cyber incidents.

Currently, the way organizations collaborate on cyber security is more like a subscription service they get from a collaboration center. Take FS-ISAC [2] for example. The member organizations submit their security information and get security services like reports and alerts from the collaboration center. This type of cyber collaboration has several limitations. Organizations manually submit security information. Organizations are not actively participating in analyzing and processing the cyber information they submit. Sharing information is mainly by subscription, rather than interactive sharing in a group.

With cloud technology development, we believe with organizations transferring to cloud environment, the way they share cyber information will change as well. A community cloud shares the infrastructure across multiple organizations from a specific community with common concerns in terms of security, privacy and compliance. We propose a community cloud model to allow organizations to rapidly and meaningfully share cyber security information and resources. The community runs a standing Cyber Security Committee, which enables executives and technology leaders to provide oversight of privacy and security while enabling effective information sharing. This cross-organizational committee is in constant communication to coordinate such sharing while meeting privacy and security needs.

Organizations will collect and analyze their security data as usual, while sharing cyber security information with other members through community cyber security committee, in order to make informed decisions about the community security governance. In most cases, organizations maintain

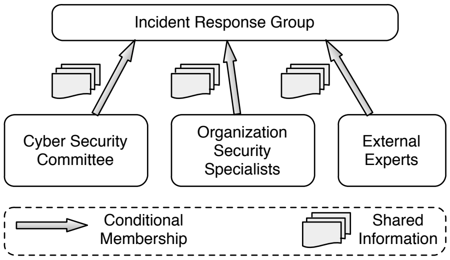


Figure 1. Community Cyber Incident Response Governance

their group of security specialists, who manage security policies, conduct security audits and investigate security related events. A community also maintains a group of external security experts, who help organizations with security issues. During the occurrence of cyber security incident, the Cyber Security Committee members start an incident response group with cross-organization security team including organizations internal security specialists and external security experts, as illustrated in Figure 1. Security information about this incident is shared within the incident response group.

In this paper, we present an access control model for cyber security information sharing within a community cloud for cyber incident response. This paper proceeds as follows. We present some related work and background knowledge in Section 2. We introduce OpenStack Access Control model with Hierarchical Multitenancy (OSAC-HMT) in Section 3. In Section 4, we define the OSAC-HMT with Secure Isolated Domain extension (OSAC-HMT-SID), which is our model for cyber incident response. We give some enforcement suggestions in Section 5. Finally we conclude our work in Section 6.

II. Related Work and Background

A. Related Work

Sharing information and resources for collaboration in distributed systems has been studied in the literature for some time [3], [6], [8]. More recently a concept of sharing information and resources in a group of users, called Group-Centric Secure Information Sharing (g-SIS) [4] has been developed. The g-SIS model changes the emphasis of the access control unit from individual users and objects to a group of users and objects, which is suitable for collaboration scenarios.

In this paper we explore the application of g-SIS in the OpenStack cloud platform, particularly in the scenario of collaboration cyber incident response in a community cloud. We have previously developed a basic model (OSAC-SID) for this purpose [10] for the OpenStack Icehouse release. The model we present in this paper improves OSAC-SID in several ways. We put additional cyber security control and sharing on the entire community by adding a security committee and a public forum in the community. We incorporate routine cyber information collection and processing for an individual organization's regular security control. We also provide a more flexible cyber collaboration mechanism.

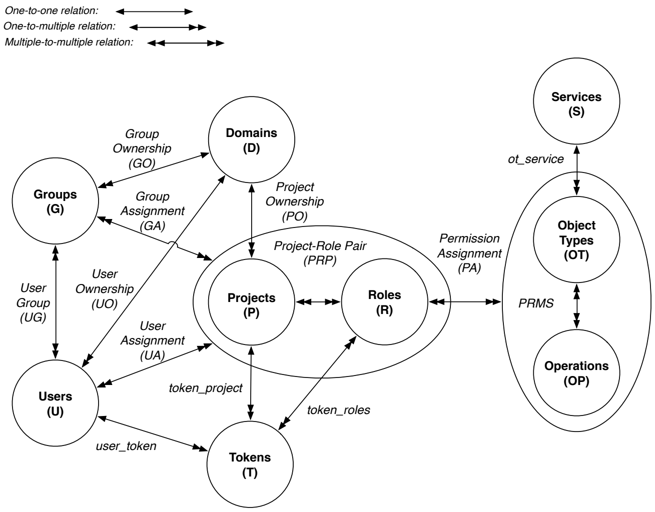


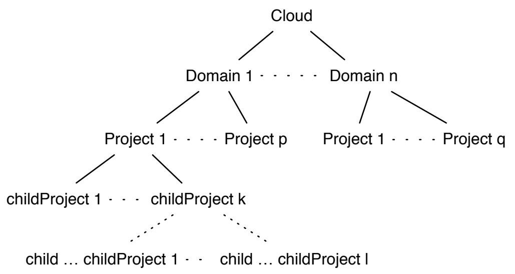
Figure 2. OpenStack Access Control (OSAC) Model [9]

B. OpenStack Access Control (OSAC) model

Tang and Sandhu [9] present a core OpenStack Access Control (OSAC) model based on the OpenStack Identity API v3 and Havana release, as shown in Figure 2. This model comprises nine entities: users, groups, projects, domains, roles, services, object types, operations, and tokens.

Users represent people who are authenticated to access OpenStack cloud resources while **groups** are sets of users. **Projects** are resource containers through which users get access to cloud **services** such as virtual machines, storage, networks, identity, and so on. Each project defines a boundary of cloud resources. **Domains** are administrative boundaries of collections of projects, users and groups. Each domain contains multiple projects, users and and groups. Conversely, each project, user and group is "owned" by a single domain. For our purpose in this paper, a domain is also called a **tenant**. From the cloud provider's perspective each tenant is an independent customer of the cloud. From an organization's perspective, in general a single organization may have a single or multiple tenants in a single cloud. For simplicity, we assume here that each organization from the community has exactly one tenant, and thereby exactly one domain, in the community cloud.

Roles are global in that each role is applicable to every project. Roles are used to specify access levels of users to services in specific projects in a given domain. Roles, and their associated permissions, are defined by the cloud service provider. Note that users are assigned to projects with a specific set of roles. By assigning a role to a user in a project, one can specify different access rights for the user.



Cloud
Domain 1 · · · · · Domain n
Project 1 · · · · Project p
Project 1 · · · · Project q
childProject 1 · · · childProject k
child ... childProject 1 · · child ... childProject l

Figure 3. OpenStack Hierarchical Multitenancy

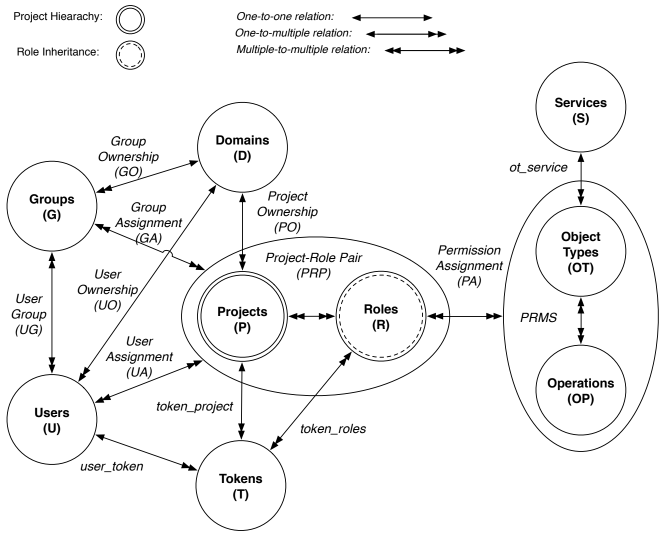


Figure 4. OpenStack Access Control (OSAC) model with HMT

For instance, by assigning the *member* role to a user, the user receives all operational permissions over the resources in a project. By assigning the *admin* role to a user, the user receives admin permissions over a project. In this paper, we recognize two required roles: *admin* and *member*, which are used in our formalization along the above lines.

An **object type** and **operation** pair defines actions which can be performed by end users on cloud services and resources. The concept of object types allow specifying different operations for different services. In the Nova compute service, e.g., an object type is VM and operations on VM include start, stop, etc. **Tokens** defines the scope of resources which users are authenticated to access. Users authenticate themselves to the Keystone service and obtain a token which they then use to access different services. The token contains various information including the domain the user belongs to, and the roles of the user in specific projects.

Scope: In the model we develop in this paper, we confine our attention to information and resource sharing among tenants within a single community cloud. These issues in the context of multiple/hybrid clouds is an interesting research problem left for future work.

III. OSAC-HMT MODEL

Hierarchical Multitenancy (HMT) [1] is a new feature added to OpenStack since Juno release. It changes OpenStack from the flat domain-projects structure to a hierarchical domain-parent project-child project tree structure. Prior to Juno release, OpenStack allows tenants to have domains with flat projects in them. Hierarchical Multitenancy allows tenants to have hierarchical project trees in a domain, as shown in Figure 3.

In this paper, we enhance OSAC model with the new feature of Hierarchical Multitenancy (HMT), resulting in the OSAC-HMT model shown in Figure 4. In this section, we mainly discuss the new feature of OpenStack relative to the former OSAC model [9].

The difference HMT brings to OSAC is that it changes projects and roles entities, along with the administration relation on projects. The flat projects in OSAC model become hierarchical trees in OSAC-HMT model. In addition to explicit assignment of project-role pairs, users also inherit project-role pairs along the project tree.

Projects and Project Hierarchy: Project hierarchy enables the resources to be divided into smaller management units, giving tenants more power to control their cloud resources. A domain can have multiple projects in it, each of which is a root project for a hierarchical project tree. A child project has only one parent project. Basically, child projects are a further division of resources of a parent project.

Roles and Role Inheritance: Without project hierarchy, a user is explicitly assigned with a role to a project. With project hierarchy, a user needs to be able to be assigned to a child project, which is enabled by inherited roles assignment. By assigning an inherited role to a user in a parent project, the user will automatically have the role in child projects. Currently, inherited roles assignments only work from domains to projects. In future releases of OpenStack, it is expected that the inheritance of roles will work down the entire subtree of a hierarchical project.

Token: Token allows user to have access to cloud resources in projects. Token must be scoped to the target project on which the action is performed. Inherited role allows tokens to be granted for child projects giving access to child projects.

Users/Groups: HMT does not change user/group management, which is handled at the domain level.

We formalize the OSAC-HMT model below, part of which is the same as OSAC model [9].

A. Components in OSAC-HMT

Definition 1. OSAC-HMT model has the following components.

- U, G, P, D, R, S, OT and OP are finite sets of existing users, groups, projects, domains, roles, services, object types and operations respectively in an OpenStack cloud system. We require two roles, so $\{admin, member\} \subseteq$ R.

- User Ownership (UO) : is a function UO : U $\rightarrow$ D, mapping a user to its owning domain. Equivalently viewed as a many-to-one relation UO $\subseteq$ U $\times$ D.

- Group Ownership (GO) : is a function GO : U $\rightarrow$ D, mapping a group to its owning domain. Equivalently viewed as a many-to-one relation GO $\subseteq$ G $\times$ D.

- Object Type Owner (OTO) : is a function OTO : OT $\rightarrow$ S, mapping an object type to its owning service. Equivalently viewed as a many-to-one relation OTO $\subseteq$ OT $\times$ S.

- UG $\subseteq$ U $\times$ G, is a many-to-many relation assigning users to groups where the user and group must be owned by the same domain.

- PRP = P $\times$ R, the set of project-role pairs.

- PERMS = OT $\times$ O, the set of permissions.

- PA $\subseteq$ PERMS $\times$ R, a many-to-many permission to role assignment relation.

- UA $\subseteq$ U $\times$ PRP, a many-to-many user to project-role assignment relation.

- GA $\subseteq$ G $\times$ PRP, a many-to-many group to project-role assignment relation.

- Project Hierarchy (PH) : is a function PH : P $\rightarrow$ P, mapping a project to its parent project. Equivalently viewed as a many-to-one relation PH $\subseteq$ P $\times$ P. This is required to be a forest of rooted trees.

- Role Inheritance (RI) : allows users' roles to be inherited from domain to project and from parent project to child project as discussed above.

- user_tokens: is a function user_tokens : U $\rightarrow 2^T$, mapping a user to a set of tokens; correspondingly, token_user is a function token_user : T $\rightarrow$ U, mapping of a token to its owning user.

- token_project: is a function token_project : T $\rightarrow$ P , mapping a token to its target project.

- token_roles: is a function token_roles : T $\rightarrow 2^R$, mapping a token to its set of roles. Formally, token_roles(t) = $\{$r $\in$ R$|$(token_user(t),(token_project(t),r)) $\in$ UA$\}$ $\cup$ ($\bigcup_{g \in user\_groups(token\_user(t))}$ $\{$r $\in$ R$|$(g, (token_project(t), r)) $\in$ GA$\}$).

- avail_token_perms: is a function avail_token_perms : T $\rightarrow 2^{PERMS}$, mapping the permissions available to a user through a token. Formally, avail_token_perms(t) = $\bigcup_{r \in token_roles(t)}\{$perm $\in$ PERMS$|$(perms,r) $\in$ PA$\}$.

IV. OSAC-HMT-SID MODEL

In our discussion, we assume that a user belongs to one organization in the community, which is consistent with the user home-domain concept in OpenStack. The concept of home-domain requires that a user can only belong to one domain in OpenStack. OpenStack allows a user to be assigned to projects across domains and access those projects separately using appropriate tokens. Given two storage options in OpenStack, here we constrain the storage to object storage only, which is provided by the Swift service. For simplicity we ignore the group mechanism in OpenStack, since it is essentially a convenience to group together a set of users in a domain and can be easily incorporated in a more complete description.

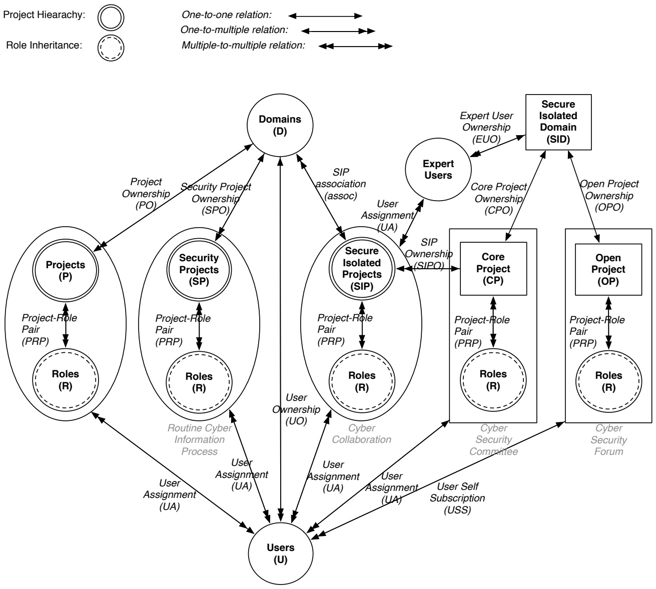


Figure 5. Hierarchical Mutitenancy OpenStack Access Control model with SID extension (OSAC-HMT-SID) (ignore group, token and services components)

A. Components in OSAC-HMT-SID

Hierarchical Mutitenancy OpenStack Access Control model with SID extension (OSAC-HMT-SID): OSAC-HMT-SID model extends OSAC-HMT model to include Secure Isolated Domain (SID) [10] functionality. We build OSAC-HMT-SID model on top of OSAC-HMT model. We will present the OSAC-HMT-SID model in a way which covers only the additional components compared to OSAC-HMT model. Figure 5 shows OSAC-HMT-SID model. We use circle to represents entities which can be created multiple times in OpenStack, while rectangle represents entities which can only be created once. The additional entity components included in OSAC-HMT-SID model are: Secure Isolated Domain (SID), Expert Users (EU), Core Project (CP), Secure Isolated Project (SIP), and Open Project (OP).

Secure Isolated Domain (SID): Secure Isolated Domain [10] is a special domain which holds the security information for cross-organization security collaboration in the community cloud. It provides an administrative boundary for cyber security information and resource collecting, passing, analyzing and exporting results, as well as providing a secure isolated environment for cyber security collaborations among organizations.

Security Project (SP): Security Projects are hierarchical projects particularly used to collect, store and analyze cyber security information for one organization. A SP provides the same capability of utilizing cloud resources as a normal project could do. Organizations keep their security information and resources in the Security Projects, with their security staff/users assigned to the corresponding level of project in the Security Project hierarchy. This separates an organization's regular projects from its security project.

Core Project (CP): Core Project is a shared project which holds the community cyber security committee [7]. Each organization in the community has at least one user in the security committee, with one as admin user of the Core Project and the rest as regular member users. Core Project holds all Secure Isolated Projects which are designed for cyber incident response and cyber security collaboration.

Open Project (OP): Open Project is a project where users share public cyber security information and resources [7]. Information published in Open Project is public to every user who is subscribed to the project.

Secure Isolated Project (SIP): Secure Isolated Project [10] is a special project with constraints over its user membership, information and resources utilization. A SIP provides a controlled environment for organizations to collaborate on security incidents.

Expert Users (EU): To get outside-community professionals involved, expert users [7] are introduced to SID. Expert Users originally don't belong to the community. They bring expertise from different cyber security categories. For instance, they may come from a IT consultant company which focusses on specific cyber attacks. They may be cyber security law enforcement officers specializing in cyber crime. The involvement of Expert Users is to help organizations handle cyber collaborations more effectively.

In the following, we give formalization of concepts introduced above, as well as the relation among them.

Definition 2 OSAC-HMT-SID model has the following components in addition to OSAC-HMT.

- SID is an implicitly existing Secure Isolated Domain, which is transparent to users. SID owns Expert Users (EU), Core Project (CP), Open Project (OP), and Secure Isolated Projects(SIP), correspondingly represented by Expert User Ownership (EOU), Core Project Ownership (CPO), Open Project Ownership (OPO)and Secure Isolated Project Ownership (SIPO).
- SP, SIP, EU and SO are finite sets of Security Projects, Secure Isolated Projects, Expert Users and Swift Objects.
- Security Project Ownership (SPO) : is a function SPO : SP $\rightarrow$ D, mapping a Security Project to its owning domain. Equivalently viewed as a one-to-one relation SPO $\subseteq$ D.
- Swift Object Ownership (SOO) : is a function SOO : SO $\rightarrow$ P, mapping a swift object to its owning project. Equivalently viewed as a many-to-one relation SOO $\subseteq$ SO $\times$ P.
- User Self Subscription (USS) : USS $\subseteq$ U $\times$ $\{<$ OP, *member* $>\}$, a many-to-one user to project-role assignment relation for the *member* role in the single open project OP.
- SIP association (assoc): is a function assoc : SIP $\rightarrow 2^D$, mapping a SIP to all its member domains/organizations.

B. Administrative OSAC-HMT-SID Model

The administrative aspects of OSAC-HMT-SID are discussed informally below. A formal specification is given in Table I.

Creation of SID, Core Project, Open Project and Security Project: SID with Core Project and Open Project is part of community cloud functionality which the CSP provides to its customers on behalf of organizations responding collaboratively to cyber incidents. SID, Core Project and Open Project are created when the community cloud is set up. Each domain has one corresponding Security Project with it. The creation of a Security Project is automatically done with the creation of a domain.

Initial user assignment for SID, Core Project, Open Project and Security Project: SID has no admin users assigned on domain level. The admin users of Core Project come from organizations' domain. When a domain is created, cloud admin assigns domain admin user as an admin of Core Project. We assume there is only one admin user for each domain. Domain admins assign admin users for their Security Projects. Open Project doesn't have admin user assigned to it. Each user in the cloud can self subscribe or unsubscribe as a member in Open Project.

Create a SIP: Let *uSet* denote a set of domain admin users. A group of organizations come together to create a SIP. Each organization in the group has equal administrative power over the SIP. The creation of SIP succeeds based on agreement among the group of organizations. The organization membership in the SIP is established with the creation of the SIP. The size of the group range from one organization to the total number of organizations held in the community cloud. The group of organizations set up a SIP by sending the SIP creation request to the cloud admin. The users who are allowed to issue SIP creation are admin users in Core Projects, who are domain admins as well. When a SIP is created, users who issue SIP creation command will automatically become the admin users of the SIP.

Delete a SIP: After the collaboration is finished, a SIP needs to be securely deleted. The delete command is issued by the same set of admin users (*uSet*) who issue the SIP creation. All information and resources are securely deleted. All users assigned to the SIP are removed from it. Removing information and resources guarantees no information and resources will leak after the SIP being deleted. Removing users guarantees no users will have access to information and resource that belonged to a SIP.

Create/delete an Expert User: New Expert Users are created in case when additional cyber expertise is needed,

Table I
OSAC-HMT-SID Administrative model

Operation	Authorization Requirement	Update

SipCreate(uSet, sip)
/ A subset of Core Project/domain admin users together create a sip */*
	$\forall$ u $\in$ uSet.(u $\in$ U $\wedge$ (u, <CP, admin>) $\in$ UA) $\wedge$ sip $\notin$ SIP	assoc(sip) = $\bigcup_{u \in uSet}$ UO(u)
		SIP$'$ = SIP $\cup$ {sip}
		UA$'$ = UA $\cup$ uSet $\times$ {<sip, admin>}

SipDelete(uSet, sip)
/ The same subset of Core Project/domain admin users together delete a sip*/*
	$\forall$ u $\in$ uSet.(u $\in$ U $\wedge$ (u, <sip, admin>) $\in$ UA $\wedge$ (u, <CP, admin>) $\in$ UA) $\wedge$ assoc(sip) = $\bigcup_{u \in uSet}$ UO(u) $\wedge$ sip $\in$ SIP	assoc(sip) = NULL
		SIP$'$ = SIP - {sip}
		UA$'$ = UA - uSet $\times$ {<sip, admin>}

ExpertUserCreate(coreadmin, eu) */* Core Project admin users can create an expert user */*	coreadmin $\in$ U $\wedge$ (coreadmin, <CP, admin>) $\in$ UA $\wedge$ eu $\notin$ EU	EU$'$ = EU $\cup$ {eu}
ExpertUserDelete(coreadmin, eu) */* Core Project admin users can delete an expert user */*	coreadmin $\in$ U $\wedge$ (coreadmin, <CP, admin>) $\in$ UA $\wedge$ eu $\in$ EU	EU$'$ = EU - {eu}
ExpertUserList(adminuser) */* Admin users of Core Project and SIPs can list expert users */*	adminuser $\in$ U $\wedge$ ($\exists$ proj) {proj $\in$ ({CP} $\cup$ SIP) $\wedge$ (adminuser, <proj, admin>) $\in$ UA}	
ExpertUserAdd(adminuser, r, eu, proj) */* Core Project/sip admin can add an expert user to Core Project/sip*/*	adminuser $\in$ U $\wedge$ proj $\in$ ({CP} $\cup$ SIP) $\wedge$ (adminuser, <proj, admin>) $\in$ UA $\wedge$ eu $\in$ EU $\wedge$ r $\in$ R	UA$'$ = UA $\cup$ (eu, (proj, r))
ExpertUserRemove(adminuser, r, eu, proj) */* Core Project/sip admin can remove an expert user from Core Project/sip */*	adminuser $\in$ U $\wedge$ proj $\in$ ({CP} $\cup$ SIP) $\wedge$ (adminuser, <proj, admin>) $\in$ UA $\wedge$ eu $\in$ EU $\wedge$ r $\in$ R $\wedge$ (eu, (proj, r)) $\in$ UA	UA$'$ = UA - (eu, (proj, r))
UserAdd(adminuser, r, u, sp, p) */* CP/Sip admin can add a user from his home domain Security Project to CP/sip*/*	adminuser $\in$ U $\wedge$ (adminuser, <p, admin>) $\in$ UA $\wedge$ p $\in$ ({CP} $\cup$ SIP) $\wedge$ r $\in$ R $\wedge$ u $\in$ U $\wedge$ (u, <sp, r>) $\in$ UA $\wedge$ SPO(sp) = UO(adminuser)	UA$'$ = UA $\cup$ (u, (p, r))
UserRemove(adminuser, r, u, sp, p) */* CP/Sip admin can remove a user from the Core Project/sip */*	adminuser $\in$ U $\wedge$ (adminuser, <p, admin>) $\in$ UA $\wedge$ p $\in$ ({CP} $\cup$ SIP) $\wedge$ r $\in$ R $\wedge$ u $\in$ U $\wedge$ (u, <sp, r>) $\in$ UA $\wedge$ SPO(sp) = UO(adminuser) $\wedge$ (u, (p, r)) $\in$ UA	UA$'$ = UA - (u, (p, r))
OpenUserSubscribe(u, member, OP) */* Users subscribe to Open Project */*	u $\in$ U $\wedge$ (u, <OP, member>) $\notin$ USS	USS$'$ = USS $\cup$ (u, <OP, member>)
OpenUserUnsubscribe(u, member, OP) */* Users unsubcsribe from Open Project */*	u $\in$ U $\wedge$ (u, <OP, member>) $\in$ USS	USS$'$ = USS - (u, <OP, member>)
CopyObject(u, so1, sp, so2, p) */* Copy object from Security Project to Core Project/SIP */*	so1 $\in$ SO $\wedge$ sp $\in$ SP $\wedge$ so2 $\notin$ SO $\wedge$ SOO(so1)=sp $\wedge$ UO(u)=SPO(sp) $\wedge$ u $\in$ U $\wedge$ ($\exists$ r $\in$ R) {(u, <sp, r>) $\in$ UA $\wedge$ (u, <p, r>) $\in$ UA)} $\wedge$ p $\in$ ({CP} $\cup$ SIP)	SO$'$ = SO $\cup$ {so2} SOO(so2) = p
ExportObject(adminuser, so1, p, so2, sp) */* Export object from Core Project/SIP to Security Project */*	adminuser $\in$ U $\wedge$ (adminuser, <p, admin>) $\in$ UA $\wedge$ p $\in$ ({CP} $\cup$ SIP) $\wedge$ so1 $\in$ SO $\wedge$ SOO(so1)=p $\wedge$ so2 $\notin$ SO $\wedge$ sp $\in$ SP $\wedge$ (adminuser, <sp, admin>) $\in$ UA	SO$'$ = SO $\cup$ {so2} SOO(so2) = sp

such as consultant company is introduced to the community, or a new cyber security agent is involved with one of the collaboration groups. Core Project admin users request the creation command of Expert Users to cloud admin. Cloud admin returns the new Expert User and add the user to Expert User list. Core Project admin users can request to delete a Expert User. After the Expert User is deleted, the user will lose all access to any information and resource in the community cloud.

List Expert Users: Core Project and SIP admin users can list expert users in SID. Expert Users are important human resources for cyber collaboration activities. By listing Expert Users in the SID, collaborative groups with SIPs can easily add experts to their SIPs.

Add/remove an Expert User: Expert Users are visible to all projects in SID except Open Project. Project admins in SID can add Expert Users to their projects due to collaboration. After the cyber collaboration is done, project admins can remove Expert Users from their projects.

Add/remove a user to/from Core Project/SIP: Admin users of Core Project/SIP add/remove users of their home security projects to/from Core Project or the corresponding SIP due to the need of collaboration. The removed user will lose access to information and resources which he/she had during collaborations in Core Project/SIP.

Subscribe/unsubscribe a user to Open Project: Ev-

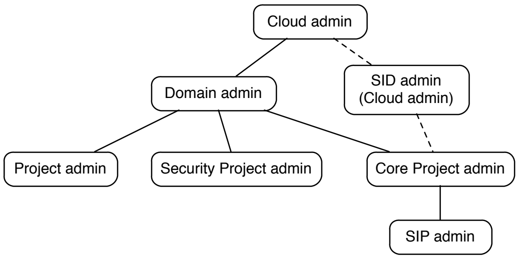


Figure 6. Administration Relation

ery user in the Open Project is a normal member user. They can share cyber data, but have no control over other users. Users subscribe/unsubscribe themselves to/from Open Project. They will not be able to access and share any data once they leave the Open Project.

Copy data between Secure Project and Core Project/SIP: Users can copy data from security projects of their home domains to CoreProject and SIP. Users may be scoped to multiple projects in their home domains, but only data from security projects are allowed to be copied to CP/SIP. Admin users can export data from Core Project and SIPs to security projects of their home domains.

C. Additional administration details

Here we give additional explanation of OSAC-HMT-SID model from administration perspective, as shown in Figure 6. Cloud admin is the super administrative user of the cloud who can create domains, users and assign users as admins for domains and projects. Some administrative operations in SID are done by cloud admin, such as creating/deleting/updating expert users and creating/deleting/updating SIPs, though the request is initiated by a subset of Core Project admin users.

Domain admin is the super administrative user for an organization. Domain admin can create/delete/update a project and user/group in the domain. Projects can also have admin users assigned to them, the difference is that project admin user cannot create/delete/update users and groups, but they can assign users/groups to the project, and create/delete/update child projects.

Domain admin users assigns users to be admin of their Security Projects. Security Project admin users can further add other users as member in Security Project.

Core Project is designed for core group [7], which is a cyber security committee for the whole community. Domain admin decides which of the organization's users will be in the cyber security committee. Domain admins are automatically assigned as admin users in Core Project when a domain is created. As Core Project admin users, they can further add users from their home domains to Core Project, create SIPs and add users to SIPs.

The administration over a SIP is similar to that in [10]. A subset of Core Project admin users create/delete/manage a SIP. Each user in this subset has equal admin power over the SIP. They can create/delete/update child projects inside the SIP. They can assign users from their organizations to the SIP. They can bring in cyber information from their Security Projects.

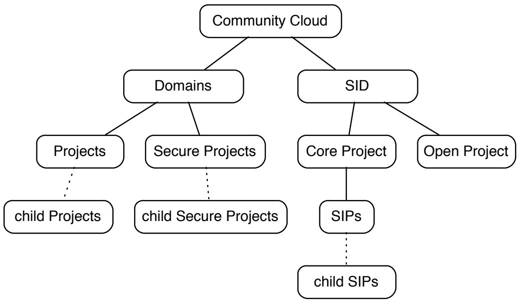


Figure 7. Resources Ownership

D. Resource ownership

From the perspective of resource ownership, we give a view of the model, as shown in Figure 7. Organizations own their resources manifested as domains in the community cloud. An organization has multiple normal projects and one Secure Project. Inside a domain, resources are divided by projects which represent different departments inside an organization. Departments can further divides the resources ownership by creating child-projects. Security Project is for each organization to contain standard cyber security data, which is used on behalf of each organization's security as well as cross-organization cyber security collaboration. A Security Project is the only place in a domain from and to where cyber information can be exchanged with the SID.

SID securely isolates cloud resources from organization domains for cyber security purpose. SID is owned by the community cloud. The Core Project belongs to SID and provides a stable and controlled place for organizations to exchange and share cyber security information. It holds all SIPs which are designed for specific cyber security purposes. SIPs can be further divided into child SIPs in the process of cyber collaborations.

V. Enforcement

We discuss the enforcement of OSAC-HMT-SID model on OpenStack Kilo release. In OpenStack, there are three levels of administrative roles: $cloud\_admin$, $domain\_admin$, and $project\_admin$, which have administrative power respectively over the whole cloud, a domain and a project.

Setting up SID: SID is a functionality adding to OpenStack cloud platform. SID, Core Project, and Open Project are created when the cloud platform is set up. In implementation, we can use cloud admin to set up SID with Core Project and Open Project. Security Project is created with creation

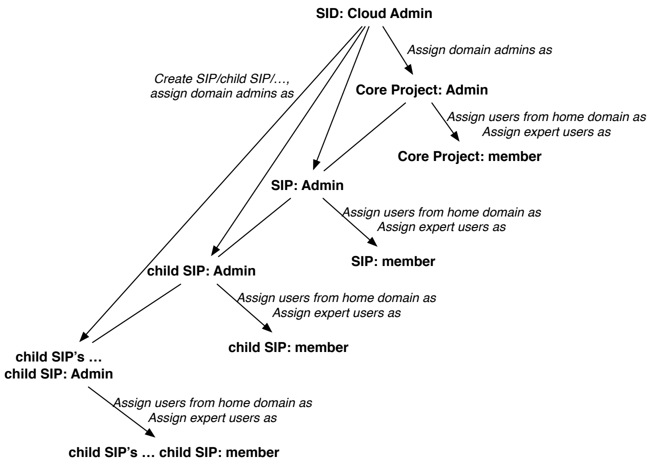


Figure 8. SIP creation and user assignment

of a domain. For simplicity, we use cloud admin to create Security Project for a domain. Cloud admin assigns domain admin as admin user for Core Project. Cloud admin assign every user in the cloud to Open Project as a member. All such cloud admin functions can be automated by providing scripts that do these activities on the cloud admin's behalf after verifying appropriate authorization,

SIP creation and user assignment: A subset of Core Project admin users create SIPs and child SIPs. These users need permission to create a project in SID. However, OpenStack doesn't allow a user to create a project if the user is not scoped to the domain. All users in SID are scoped to specific project, such as Core Project, Open Project and SIP. They don't have the scope on domain level in SID. Thus, none of these users can create a project in SID. Therefore we need cloud admin to be involved. The solution is the subset of Core Project admin users request cloud admin to create a SIP/child SIP, and cloud admin returns a project with the requesters assigned as admin user to it. As project admin, these admin users can assign users from their home domain to the project. They can also assign Expert Users to the project. Figure 8 illustrates this process.

User Verification: Only Core Project/domain admin users are allowed to send request to create/delete/update SIPs/child SIPs/Expert Users. Cloud admin need to verify that the request comes from a subset of Core Project admin users. Users in SIP/child SIP are constrained to be from a subset of domains in the cloud. The restriction is set up with the creation of a SIP. Admin users are allowed to add users only from their home domains and Expert Users from SID. This needs to be verified every time a user assignment happens.

In implementation, all operations that have to be done by cloud admin can be automated by adding code to OpenStack identity server. Other constraints can be enforced by configuring appropriate policy files.

VI. Conclusion and Future Work

OpenStack is a popular open-source cloud platform which provides a great convenience for enterprises and organizations to facilitate their business. Information and resources sharing in cyber security field has been an important topic for years with the growth of cyber attacks. Models for enforcing information and resources sharing in scenario of cyber security in OpenStack platform is an important topic. The model we give in this paper is one way to achieve it. We also explored some other options in general. However, based on the features of OpenStack, we made our model as close as possible to OpenStack architecture. For the future work, we would like to explore more on other model options. We also want to explore more on local roles in the model, which is lacking in current OpenStack. Finally, it would be valuable to research similar goals in other cloud platforms including the dominant proprietary ones.

VII. Acknowledgment

The authors thank Raildo Mascena for helping us better understand OpenStack HMT. The authors also thank the broader OpenStack community in this regard. This work is partially supported by NSF CNS-1111925.

References

[1] http://openstack.org.

[2] https://www.fsisac.com/.

[3] E. Cohen, R. K. Thomas, W. Winsborough, and D. Shands. Models for coalition-based access control (CBAC). In *Proc. 7th ACM SACMAT*, 2002.

[4] R. Krishnan, R. Sandhu, J. Niu, and W. Winsborough. Towards a framework for group-centric secure collaboration. In *5th IEEE CollaborateCom*, pages 1–10, 2009.

[5] P. Mell and T. Grance. The NIST definition of cloud computing. *NIST Sp. Pub. 800-145*, Sept. 2011.

[6] L. Pearlman, V. Welch, I. Foster, C. Kesselman, and S. Tuecke. A community authorization service for group collaboration. In *3rd IEEE International Workshop on Policies for Distributed Systems and Networks*, 2002.

[7] R. Sandhu, K. Z. Bijon, X. Jin, and R. Krishnan. RT-based administrative models for community cyber security information sharing. In *7th IEEE CollaborateCom*, 2011.

[8] D. Shands, R. Yee, J. Jacobs, and E. J. Sebes. Secure virtual enclaves: Supporting coalition use of distributed application technologies. In *IEEE DARPA Information Survivability Conference and Exposition*, volume 1, pages 335–350, 2000.

[9] B. Tang and R. Sandhu. Extending OpenStack access control with domain trust. In *8th International Conference on Network and System Security (NSS)*, October 15-17 2014.

[10] Y. Zhang, R. Krishnan, and R. Sandhu. Secure information and resource sharing in cloud infrastructure as a service. In *2014 ACM WISCS*, pages 81–90, 2014.