Community-Based Secure Information and Resource Sharing in AWS Public Cloud

Yun Zhang, Farhan Patwa and Ravi Sandhu
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

***Abstract*—A public cloud provides enterprises and organizations with a secure and efficient environment to deploy their systems. While organizations and companies benefit from moving to cloud platform, it is likely that similar cyber attacks will happen to organizations which share the same cloud platform. One way to mitigate this risk is to share cyber security information among these organizations. Unfortunately, popular public cloud platform AWS is lacking an accepted access control model for cyber security information sharing. We propose an access control model for customers who use AWS platform as their infrastructure platform to securely share cyber attack information. Our model enables secure cyber information sharing and collaborations in public cloud environment on a community basis.**

***Keywords*-Cloud; Incident Response; Security Information Sharing; AWS;**

I. Introduction

Securely and effectively sharing cyber information across multiple organizations and cooperating on cyber security incidents has been a significant research topic in recent years. Cyber security information sharing allows organizations to share threat analysis and incident response information with collaborative communities formed to handle both existing and potential cyber threats.

With growing sophisticated cyber attacks every year, defending a single organization on its own becomes increasingly difficult. All organizations, regardless of size, could be the target of a cyber attack putting its critical digital assets at risk. A cyber breach can result in substantial economic loss. Establishing a general cyber incident response mechanism for organizations enhances cross-organization coordination, speeds up incident analysis and decision making process, helps identify and understand the attack and take quick response actions. A good response can minimize the damage caused by cyber incidents on valuable digital assets.

Cloud technology puts multiple organizations in a single cloud system infrastructure, giving additional opportunities for cyber adversaries to attack organizations of similar systems in a single cloud. On the other hand, by virtue of sharing same cloud infrastructures it is more likely that organizations will have similar concerns regarding security and privacy. Having suitable cyber security risk management mechanisms in public-cloud communities could be significantly valuable to every organization in that cloud.

While cloud technology significantly improves efficiency and flexibility of business systems, it also facilitates cyber collaborations. To our knowledge, current dominant cloud platforms are lacking broadly accepted cyber incident response mechanisms. The traditional approach for cyber security collaboration is mainly through subscription services where organizations can get threat analysis, cyber attack reports and alerts, and so on. Individual organizations submit their security data to a centralized security service center for this purpose. These traditional approaches focus on simply security information exchange and sharing. In our work we seek to develop approaches allowing participating organizations to actively collaborate and interact through the life cycle of a cyber incident which impacts multiple organizations in a well-defined community.

In this paper, we investigate models for secure information and resources sharing in Amazon Web Services (AWS) [1] public cloud. A public cloud is one of the main deployment models of contemporary cloud platforms, providing services for the general public [5]. AWS is a public cloud service offered by Amazon. It provides businesses rapid access to flexible and low cost IT resources via services such as remote computing (Amazon EC2) and storage (Amazon S3).

We propose a public cloud model to facilitate the formation of communities comprising of organizations willing to collaborate with other community members in context of cyber incidents. The goal is that organizations in such a well-defined community can rapidly share cyber security information and resources. The community runs a standing Cyber Security Committee, which enables its member organizations to continuously communicate and coordinate on security and privacy issues. It also runs a Cyber Security Forum, which provides a general place for users from the allied organizations to share security information. The security committee enables organization security leaders to be aware of the overall security and privacy situation, and is limited to select individuals form each member organization. The security forum provides an open forum for security education and awareness for the community, which is limited to individuals from the member organizations who can voluntarily join and leave.

When a cyber incident occurs, affected organizations within the community can quickly form an instant cyber incident response team with internal and external security

978-1-5090-0089-0/15 $31.00 © 2015 IEEE
DOI 10.1109/CIC.2015.42

specialists. Security information and resources for the incident are shared in the incident response team. A cyber security service is provided in the public cloud, which enables organizations having cross-organization collaborations to communicate and coordinate with other organizations during life cycle of a cyber incident. Organizations share their security data with other members in the community.

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

II. Related Work

We have presented several access control models for secure information and resources sharing in a collaborative community of organizations. We developed the OpenStack Access Control model with SID extension (OSAC-SID) [9], which is a basic model for organizations sharing information in a OpenStack cloud platform [2]. We also designed the advanced Hierarchical OpenStack Access Control model with SID extension (OSAC-HMT-SID) [10], which provides organizations additional cyber security control with routine cyber information collection and processing, a community security committee and a public security forum in the community. This paper is a continuation of this line of work, but in AWS public cloud.

The concept we used to build these models comes from Group-Centric Secure Information Sharing (g-SIS) [4], which introduces group-based information and resources sharing. G-SIS model presents a method to control access among a group of users and objects, which is well suited to the collaborative community scenario. In particular, g-SIS enables sharing using copies of original information, versus traditional information and resource sharing approaches which give access to original information and resources [3], [6], [8] to enable sharing. Sharing by copy gives additional security protection over the original information and resources, since access to the copies can be provided in a tightly controlled environment.

In this paper, we further explore information and resources sharing in AWS public cloud. Compared to OpenStack, AWS public cloud provides very different mechanisms. AWS uses local roles and policies while OpenStack uses global roles and service-owned police. AWS doesn't have hierarchical multi-tenancy while OpenStack does. AWS provides very powerful federation capability for cross-account access. Knowing the difference between AWS and OpenStack, it is interesting to explore a suitable information and resources sharing model for AWS.

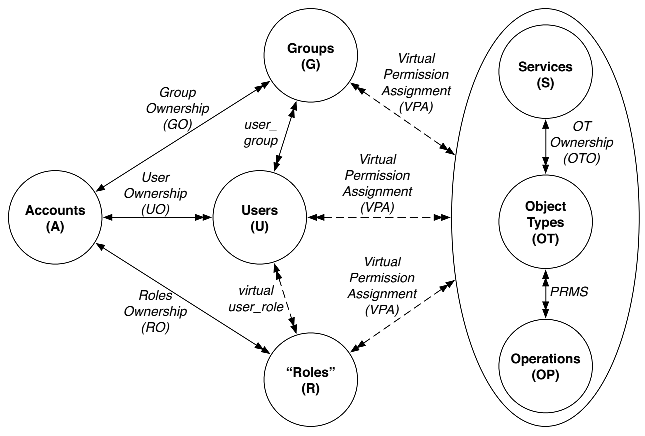


Figure 1. AWS Access Control within a Single Account

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

III. AWS Access Control Model (AWSAC)

We present the Amazon Web Service Access Control (AWSAC) model in this section. As a public cloud service provider, AWS provides web services to its customers through AWS accounts. Customers who own an account have access to cloud resources. They can create users and grant them access to cloud resources in the account. A user belongs to a unique account. Users can also access resources in other accounts with federated permissions. We discuss AWS Access Control in two perspectives: within a single account and across accounts.

AWS offers a form of policy-based access control, wherein permissions are defined over cloud resources in a policy file and policies are attached to entities such as users, groups, roles and resources. Figure 1 depicts this model within a single account. In this and other figures, the arrows denote binary relations with the single arrowhead indicating the one side and double arrowheads the many side. The dotted lines denote virtual relations between entities while the solid lines denote explicit relations. Cross-account access will be discussed later in context of Figure 2.

AWSAC has seven components: Accounts (A), Users (U), Groups (G), Roles (R), Services (S), Object Types (OT), and Operations (OP). We also introduce other entities such as policies and credentials, which are implicitly included in the model.

Accounts: In AWS, Accounts are basic resource containers, which allows customers to own specific amount of

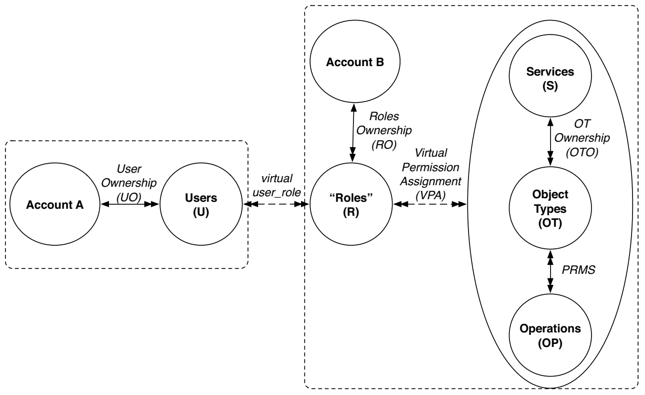


Figure 2. AWS Access Control Across Accounts [Users in account A access services and resources in account B]

(virtual) cloud resources. Accounts are the units of usages of cloud resources and billing. Customers get public cloud services through an AWS account.

Users and Groups: Users represent individuals who can be authenticated by AWS and authorized to access cloud resources through an account. A group is simply a set of users. Users and Groups belong to an account. The existence of groups is for the convenience of managing multiple users as a single unit. Each policy attached to a group will apply to all group members. For simplicity, we use the term users to represents both users and groups in the rest of the paper.

Virtual Permission Assignment: In AWS, users' permissions over services and resources are defined in policy files. Policy files can be attached to a user, a group, a role or a specific cloud resource. By attaching a policy to a user, a group or a role, users gain permissions to corresponding cloud resources. The policy defines the actions the user will perform and cloud resources on which the actions will function. Multiple permissions can be defined in one policy file. Multiple policy files can be attached to one entity. AWS achieves permission assignment in a virtual manner via the policies attached to various relevant entities.

Roles: Unlike roles in RBAC, roles in AWS are mainly used for cross-account federation purpose. However, roles can also be used for internal users in an account. Policy files can be attached to a role. Role also defines the trust relation between principals which can be either another AWS account and its users or the owner account and internal users. Users use roles though the *AssumeRole* action to get permissions to corresponding cloud resources. In this paper, when we mention roles, we mean roles entity in AWS. To emphasize the difference between the usual concept of roles in RBAC and the term roles in AWS, we use quotation marks around Roles in our figures.

Services: Services refer to cloud services AWS provides to its customers. Cloud Service Provider (CSP) leases cloud resources to its customers in terms of services. AWS provides customers with services such as compute, storage, networking, administration, and database.

Object Types and Operations: An Object Type represents a specific type of object. From the CSP's viewpoint, objects are more like services. We define object types as particular service types the cloud provides. For instance, with the compute service EC2, the object type is a virtual machine; with the storage service S3, the object type is a bucket; etc.

Credentials: AWS credentials are used for both authentication and authorization. Account owners can create IAM users with their own security credentials to allow these users to access AWS services and resources. Account owners can also grant external federated users from other accounts with temporary security credentials to allow them to access the account's AWS services and resources.

Cross-account access: Users in one AWS account can access services and resources in another AWS account through the action *AssumeRole* with temporary security credentials, as shown in Figure 2. In this and other figures, the thick arrow represents an action taken by a user to assume a role. Users from account A access services and resources in account B through roles created in account B, by being attached with policies of the action *AssumeRole* and a target resource defined.

With the concepts described above, we formalize AWSAC model as follows.

Definition 1. AWSAC model has the following components.

- A, U, G, R, S, OT and OP are finite sets of existing accounts, users, groups, roles, services, object types and operations respectively in an AWS public cloud system.
- User Ownership (UO) : is a function UO : $U \rightarrow A$, mapping a user to its owning account. Equivalently viewed as a many-to-one relation $UO \subseteq U \times A$.
- Group Ownership (GO) : is a function GO : $G \rightarrow A$, mapping a group to its owning account. Equivalently viewed as a many-to-one relation $GO \subseteq G \times A$.
- Roles Ownership (RO) : is a function RO : $R \rightarrow A$, mapping a role to its owning account. Equivalently viewed as a many-to-one relation $GO \subseteq R \times A$.
- 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 $OTO \subseteq OT \times S$.
- PERMS = $OT \times OP$, is the set of permissions.
- Virtual Permission Assignment (VPA) : is a many-to-many virtual relation $VPA \subseteq (U \cup G \cup R) \times PERMS$, resulting from policy attached to users, groups, roles and resources.
- user_group $\subseteq U \times G$, is a many-to-many relation assigning users to groups where the user and group must be owned by the same account.
- virtual user_role (VUR) is a virtual relation $VUR \subseteq U \times R$, resulting from policies attached to various entities.

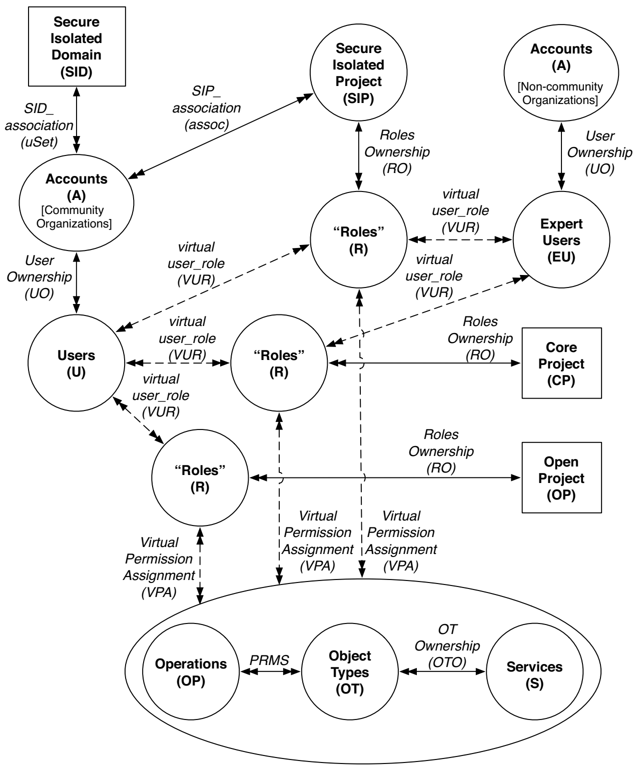


Figure 3. Amazon Web Services (AWS) Access Control Model model with SID extension (AWSAC-SID) (ignoring Groups entity)

AssumeRole is an action allowing users to activate a role authorized in VUR.

IV. AWSAC WITH SID EXTENSION (AWSAC-SID)

In this section, we present an access control model for AWS with the Secure Isolated Domain extension (AWSAC-SID). We build the AWSAC-SID model on top of the AWSAC model to include Secure Isolated Domain (SID) functionality [9]. We present the AWSAC-SID model so as to cover only the additional components added to the AWSAC model. Figure 3 shows the AWSAC-SID model, where we ignore groups for simplicity. In the rest of the paper, group is used to represent a group of organizations, rather than the groups component of AWSAC. In our discussion, we assume that a user belongs to only one organization in the public cloud. For simplicity, we also assume one organization has one AWS account.

The additional components included in AWSAC-SID model are: Secure Isolated Domain (SID), Secure Isolated Project (SIP), Expert Users (EU), Core Project (CP), and Open Project (OP). These are described below.

Secure Isolated Domain (SID): Secure Isolated Domain [9] is a special domain, holding security information and resources for cross-organizational security collaborations. SID provides an administrative boundary for cyber security information and resource collection and analysis, and a secure isolated environment for cyber security collaborations in a community of organizations. SID holds all Secure Isolated Projects (SIPs) designed for cyber incident response and security collaboration within this community of organizations. SID also holds a Core Project (CP) and an Open Project (OP), as shown in Figure 4.

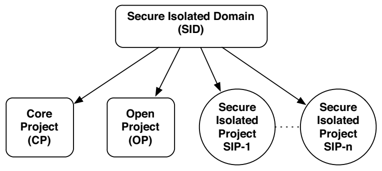

Secure Isolated Domain (SID)
Core Project (CP)
Open Project (OP)
Secure Isolated Project SIP-1
Secure Isolated Project SIP-n

Figure 4. SID Composition

Secure Isolated Project (SIP): Secure Isolated Project [9] is a special project with constraints over its user membership. It is used to collect, store and analyze cyber security information for specific security reasons. A SIP provides a controlled environment for a group of organizations within the community to collaborate and coordinate on cyber incidents and other security issues.

Core Project (CP): Core Project is a shared project holding cyber security committee [7] for the community of organizations. Each organization in the community has at least one representative security user in the committee.

Open Project (OP): Open Project is an open shared project where users from the community of organizations share common cyber security information and resources [7]. It is a common forum for all community users to share general security information. Information published in Open Project is simply public to every user who is in the project.

Expert Users (EU): To get outside professionals involved, expert users [7] are introduced to SID. Expert Users don't belong to the community of organizations. They are from other professional security organizations in the same public cloud. These experts bring different cyber security skills. 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. SID maintains an Expert Users list which is available for collaborations inside the SID.

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

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

- SIP, EU and O are finite sets of Secure Isolated Projects, Expert Users and Objects.

- SID is an unique Secure Isolated Domains serving for a community of organizations. The SID owns a Core Project

(CP). an Open Project (OP) and a number of Secure Isolated Projects (SIPs). The SID also maintain resources of Expert Users (EU).

- SIP association (assoc): is a function assoc : SIP $\rightarrow 2^A$, mapping a SIP to all its member accounts/organizations.

- Object Ownership (OO) : is a function OO : O $\rightarrow$ A, mapping an object to its owning account. Equivalently viewed as a many-to-one relation OO $\subseteq$ O $\times$ A. Object is the resource belongs to an account. We didn't include Object (O) and Object Ownership (OO) in Figure 3, since Figure 3 mainly shows the administrative perspective of the model.

A. Administrative AWSAC-SID Model

The general concept of a SID is a secure isolated container for a community of organizations to share their security data. In context of public cloud, there are two approaches to build SID functionality. One is to integrate SID function as part of the cloud functionality. The other is to build it as a service provided by a third party in the cloud to its customers. The first approach is applicable only if the cloud provider can do so. The second approach requires to build additional services on the cloud platform. For proprietary products such as AWS, SID functionality can be provided as a service by a third party who is a customer of AWS.

As a security service provided for customers in AWS public cloud, SID service might belong to a cyber security company who has the trust of the organizations in AWS. In general there may be multiple SID providers. Each SID has a Core Project and an Open Project as a security service provided to all organizations in the SID community. SID, Core Project and Open Project are created when the SID request is sent by a community of organizations. Each organization can join several SIDs with different communities of organizations. Each of these SIDs are isolated from each other. Due to the length of the paper, we only discuss the model in the case that one SID serving for one community of organizations in AWS public cloud. It can be extended to multiple SIDs serving for multiple communities of organizations in the future.

We constrain the roles in two types: administrative role and member role, which separately denotes the permission of being able to manage users and permissions only for resources. We respectively use roles *CPadmin* and *SIPadmin* to represent limited administrative power in the Core Project or a SIP, which gives the CP or SIP admin users permission to add and remove other users from their home account to the Core Project or a SIP. We user roles *CPmember*, *OPmember* and *SIPmember* to represent operative permissions which can be given to normal users to access to the Core Project, Open Project or a SIP. Since roles in AWS are local, *SIPadmin* and *SIPmember* are two sets of roles, separately representing the set of admin roles and the set of member roles in all SIPs; while *CPadmin*, *CPmember* and *OPmember* are single roles in an account.

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

Initially setup the SID: In the case of one SID serving for one community of organizations, we can initially setup the SID with one Core Project and one Open Project. The member organizations of the SID is fixed. Let *uSet* denotes a fixed group of security admin users from all organizations of the community, with one admin user for one organization. Each organization in the community has equal limited administrative power in the SID, which is carried through *uSet*. SID maintains *uSet* as a core group of admin users in SID. Only users from *uSet* later can dynamically create SIPs in the SID.

With the setting-up of SID, users in *uSet* automatically get limited administrative permission of Core Project, which represented by role *CPadmin*. With this role, CP admin users can add and remove other users from their home account to the Core Project. The Open Project is open for all the users from the community of organizations. There is no admin users needed for Open Project. All the users can add themselves to Open Project with role *OPmember* as a normal member user.

Create a SIP: A SIP is created whenever there is a need for cyber collaborations among a subset of the community organizations. It might be because of a cyber incident, a collaborative security program, or a secure information sharing program. A subset of the community of organizations representative security admin users *subuSet* together create a SIP. The creation of a SIP succeeds based on agreement among the subset of the community of organizations. Each organization in the SIP has equal limited administrative power, which represented by a role in *SIPadmin*. The role gives the SIP admin users the permission to add and remove other users from their home account to the SIP. Organizations set up a SIP by sending the SIP creation request to the SID manager account.

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 the security admin users (*subuSet*) who issue the SIP creation. All information and resources are securely deleted in the SIP. All users assigned to the SIP are removed from it.

Add/remove a user to/from a Core Project: Core Project admin users are the set of security administrative users (*uSet*) from the community of organizations. These limited administrative users can add/remove users of their organizations to/from Core Project. All the users added to the Core Project are existing users from an organization's account. The limited administrative users don't have the permission to create new users. They can only add existing users to the Core Project. When users are removed from the Core Project, they will lose the access to corresponding information and resources in the Core Project, regardless of

Table I
AWSAC-SID Administrative model

Operation	Authorization Requirement	Update
SipCreate(subuSet, sip) *`/*` A subset of organization security admin users together create a sip `*/`*	$\forall$ u $\in$ subuSet.(u $\in$ *uSet*) $\wedge$ sip $\notin$ SIP	assoc(sip) = $\bigcup_{u \in subuSet}$UO(u) SIP$'$ = SIP $\cup$ {sip}
SipDelete(subuSet, sip) *`/*` The same subset of security admin users together delete a sip`*/`*	$\forall$ u $\in$ subuSet.(u $\in$ *uSet*) $\wedge$ sip $\in$ SIP $\wedge$ assoc(sip) = $\bigcup_{u \in subuSet}$UO(u)	assoc(sip) = NULL SIP$'$ = SIP - {sip}
CpUserAdd(adminu, u) *`/*` CP admin add a user from his home account to CP `*/`*	adminu $\in$ *uSet* $\wedge$ u $\in$ U $\wedge$ UO(u) = UO(adminu)	VUR$'$ = VUR $\cup$ {(u, *CPmember*)}
CpUserRemove(adminu, u) *`/*` CP admin remove a user from CP `*/`*	adminu $\in$ *uSet* $\wedge$ u $\in$ U $\wedge$ UO(u) = UO(adminu) $\wedge$ (u, *CPmember*) $\in$ VUR	VUR$'$ = VUR - {(u, *CPmember*)}
SIPUserAdd(adminu, u, r, sip) *`/*` Sip admin add a user from his home account to SIP`*/`*	adminu $\in$ *uSet* $\wedge$ UO(adminu) $\in$ assoc(sip) $\wedge$ u $\in$ U $\wedge$ r $\in$ *SIPmember* $\wedge$ RO(r) = sip $\wedge$ sip $\in$ SIP $\wedge$ UO(u) = UO(adminu)	VUR$'$ = VUR $\cup$ {(u, r)}
SIPUserRemove(adminu, u, r, sip) *`/*` Sip admin remove a user from SIP `*/`*	adminu $\in$ *uSet* $\wedge$ UO(adminu) $\in$ assoc(sip) $\wedge$ u $\in$ U $\wedge$ r $\in$ *SIPmember* $\wedge$ RO(r) = sip $\wedge$ (u, r) $\in$ VUR $\wedge$ sip $\in$ SIP $\wedge$ UO(u) = UO(adminu)	VUR$'$ = VUR - {(u, r)}
OpenUserAdd(u) *`/*` Users add themselves to OP`*/`*	u $\in$ U $\wedge$ UO(u) $\in$ UO(*uSet*)	VUR$'$ = VUR $\cup$ {(u, *OPmember*)}
OpenUserRemove(u) *`/*` Users remove themselves from OP `*/`*	u $\in$ U $\wedge$ UO(u) $\in$ UO(*uSet*) $\wedge$ (u, *OPmember*) $\in$ VUR	VUR$'$ = VUR - {(u, *OPmember*)}
CpEUserAdd(adminu, eu) *`/*` CP admin add an expert user to CP `*/`*	adminu $\in$ *uSet* $\wedge$ eu $\in$ EU	VUR$'$ = VUR $\cup$ {(eu, *CPmember*)}
CpEUserRemove(adminu, eu) *`/*` CP admin remove an expert user from CP `*/`*	adminu $\in$ *uSet* $\wedge$ eu $\in$ EU $\wedge$ (eu, *CPmember*) $\in$ VUR	VUR$'$ = VUR - {(eu, *CPmember*)}
SipEUserAdd(adminu, eu, r, sip) *`/*` SIP admin add an expert user to SIP `*/`*	adminu $\in$ *uSet* $\wedge$ UO(adminu) $\in$ assoc(sip) $\wedge$ eu $\in$ EU $\wedge$ r $\in$ *SIPmember* $\wedge$ RO(r) = sip $\wedge$ sip $\in$ SIP	VUR$'$ = VUR $\cup$ {(eu, r)}
SipEUserRemove(adminu, eu, r, sip) *`/*` SIP admin remove an expert user from SIP `*/`*	adminu $\in$ *uSet* $\wedge$ UO(adminu) $\in$ assoc(sip) $\wedge$ eu $\in$ EU $\wedge$ r $\in$ *SIPmember* $\wedge$ RO(r) = sip $\wedge$ (eu, r) $\in$ VUR $\wedge$ sip $\in$ SIP	VUR$'$ = VUR - {(eu, r)}
CpCopyObject(u, o1, o2) *`/*`Users copy object from organization accounts to CP `*/`*	o1 $\in$ O $\wedge$ o2 $\notin$ O $\wedge$ UO(u)=OO(o1) $\wedge$ u $\in$ U $\wedge$ (u, *CPmember*) $\in$ VUR	O$'$ = O $\cup$ {o2} OO(o2) = CP
CpExportObject(adminu, o1, o2) *`/*` Admin users export object from CP to organizations accounts `*/`*	adminu $\in$ *uSet* $\wedge$ o1 $\in$ O $\wedge$ OO(o1)=CP $\wedge$ o2 $\notin$ O	O$'$ = O $\cup$ {o2} OO(o2) = UO(adminu)
SipCopyObject(u, r, o1, o2, sip) *`/*`Users copy object from organization accounts to a SIP `*/`*	o1 $\in$ O $\wedge$ o2 $\notin$ O $\wedge$ UO(u)=OO(o1) $\wedge$ u $\in$ U $\wedge$ r $\in$ *SIPmember* $\wedge$ RO(r) = sip $\wedge$ (u, r) $\in$ VUR $\wedge$ sip $\in$ SIP	O$'$ = O $\cup$ {o2} OO(o2) = sip
SipExportObject(adminu, o1, o2, sip) *`/*` Admin users export object from SIP to organization accounts `*/`*	adminu $\in$ *uSet* $\wedge$ UO(adminu) $\in$ assoc(sip) $\wedge$ o1 $\in$ O $\wedge$ OO(o1)=sip $\wedge$ o2 $\notin$ O	O$'$ = O $\cup$ {o2} OO(o2) = UO(adminu)

the ownership of the piece of information in the past.

Add/remove a user to/from a SIP: Users from *subuSet* who are assigned with role *SIPadmin* has the limited administrative power in the SIP. They can add/remove users of their home accounts to/from the corresponding SIP due to the need of collaboration. Users will lose access to information and resources after they are removed from the SIP. Administrative users in a SIP can see all users added from the community of organizations, as well as information and resources they bring in, which means there is no hidden users, information and resources in a SIP.

Add/remove a user to an Open Project: Every user in the collaborative community of organizations is allowed to join the Open Project. Users in Open Project have equal but limited permissions. They can share cyber data, but have no control over other users. We have used role *OPmember* to represent this limited permission. Users add/remove themselves from their organizations to/from Open Project. Users will not be able to access and share any data once they leave the Open Project.

Add/remove an Expert User to/from a SIP: Expert Users are needed when external cyber expertise need to be involved. For instance, a cyber incident needs experts from security consultant companies, government cyber experts, cyber polices, etc. SID services maintain a relation with external expertise. Expert users can be added/remove to/from core projects and SIPs as a member. Users from *uSet* can request to add/remove expert users to/from the Core Project while users from *subuSet* can request to add/remove expert users to/from a SIP. There are situations that an existing Expert User in a SIP needs to be removed. For instance, the contract with a cyber consultant company ends, or an cyber security agent finished his task towards a cyber collaboration. In such a case, securely deleting an Expert User is necessary. After the Expert User is deleted, the user will lose all access to any information and resource in the SIP.

Copy data between organization accounts and a Core Project/SIP: Users can copy data from their home accounts to the Core Project or a SIP. The administrative users from sets *uSet* or *subuSet* can export data from the Core Project or a SIP to their home accounts.

V. ENFORCEMENT

We discuss the enforcement of AWSAC-SID model on the current AWS release. Accounts form the basic resource boundary in AWS. Inside one account, there is no clear further sub-division of resources. Owning an AWS account give a customer root user privilege over the account. Root user has complete access power over all the services and resources in the account. Root user can create users of all level of access permissions in the account. Root user can create administrative user for the account, who can have the full access permissions except owning the account. Administrative users can further create roles and other users. Users permissions over services and resources are defined in policy files which are attached to users (among other entities).

One way to enforce SID model in AWS public cloud is to design SID as a service, which is provided by a third trusted party in AWS public cloud. For cyber security collaborations among a community of organizations, they can request multiple SIPs for collaborations from the SID security service. The number of SIPs depends on how many collaborations are going on among these organizations.

Setting up SID service: We design SID service to consist of a SID manager account, a Core Project account, an Open Project account and a number of SID operational accounts. SID manager account is the interface which respond to organizations' requests. SID manager account manages the SID operational accounts in response to the requests. SID operational accounts will be used as SIPs for a group of organizations in the community.

The Core Project has two roles created in the account: CPadmin and CPmember. CPadmin role has a policy specifying the permission which allows the user have limited

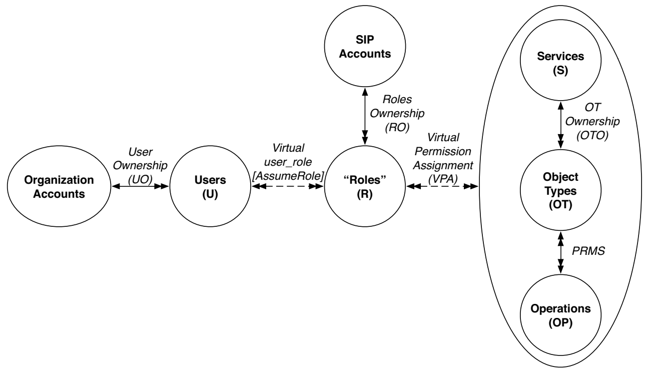


Figure 5. SID User Assignment

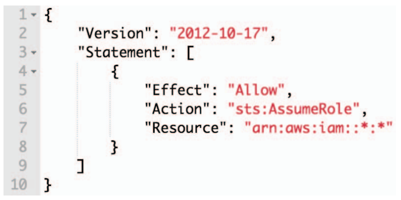
administrative power to use the role CPmember and specify policies for users from his organization. The Open Project has the role OPmember created, which has the policies allows all users from the community to access the account.

SID manager maintains a list of security administrative users (*uSet*) from organizations. Each organization in the community has one and only one security administrative user in *uSet*, which represents the organization in the SID. SID manager also maintains the associations for each SIP with its member organizations in the community.

SIP request handling When a subset of security administrative users from *uSet* send a SIP request to SID manager, SID manager creates a SIP by using non-possessed SID operational accounts, and associates the group of organizations to the SIP. The new SIP has two roles created in the account: *SIPadmin* and *SIPmember*. Role *SIPadmin* has a policy specifying the permission which allows the user have limited administrative power to use the role *SIPmember* and specify policies for users from his organization to join the SIP. SID manager returns an account number with the name of the SIPadmin role to each user from the subset of the group of security administrative users *uSet*. We call users who got *SIPadmin* role SIP admin users. Knowing the role names, SIP account numbers, security administrative users can access to the SIP and assign users from their organization account to the SIP. Figure 5 illustrates this process. Users from organization accounts get access to SIP account by assuming roles in these accounts.

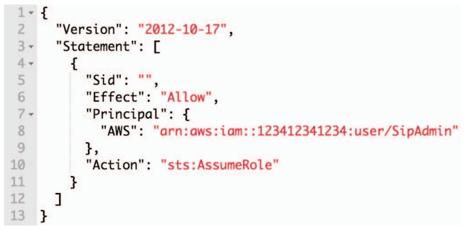
AssumeRole: Organizations can simply put a policy for their users to be able to use *AssumeRole* action, which gives users ability to access accounts in a SID. Figure 6 shows a sample of a policy attached to a user which allows the user to have access to any other account through *AssumeRole* action.

Role creation: In order to give users access to a SIP, roles like *SIPadmin* and *SIPmember* need to be created with the creation of a SIP. Role creation happens every time a SIP is requested by organizations. After a SIP is created, SIP



```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": "sts:AssumeRole",
            "Resource": "arn:aws:iam::*:*"
        }
    ]
}
```

Figure 6. AssumeRole



```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Sid": "",
      "Effect": "Allow",
      "Principal": {
        "AWS": "arn:aws:iam::123412341234:user/SipAdmin"
      },
      "Action": "sts:AssumeRole"
    }
  ]
}
```

Figure 7. Trust relationship in a Role

admin users are able to specify policies for users from their organizations to access the SIP. In a role, a trust relationship defines who from another account can access current account with the role. Figure 7 shows a sample of trust relationship, which gives user SipAdmin from account with account number *123412341234* access to current account. Similar role creation happens to Core Project and Open Project with their initial setting up. After the role was created, users from organizations can assume that role to join the SIP.

Delete a SIP: After a collaboration is completed, organizations can request to delete the SIP. All the roles which are created for the specific group of organizations will be deleted. All the information and resources that is created during the collaboration will be cleaned up. Deleting a SIP doesn't mean to delete the SIP account, it means to delete every information and created resources in the SIP account, making the SIP account clean for next SIP usage.

VI. Conclusion and Future Work

AWS is a popular public cloud platform which provides considerable convenience for enterprises and organizations to facilitate their business. Information and resources sharing in cyber security collaboration in public cloud is an important topic. The model we defined in this paper is one way to achieve such sharing in AWS. We are also interested in exploring other options in general. For the future work, we would like to explore other model options on other cloud platforms. We would also like to investigate local roles in the model, since AWS provides very fine-grained policy definition which gives scope to define fine-grained local roles in the model. Finally, it would be valuable to compare all the models in different cloud platforms especially the dominant proprietary ones.

VII. Acknowledgment

This work is partially supported by NSF CNS-1111925 and CNS-1423481.

References

[1] http://aws.amazon.com/.

[2] https://www.openstack.org/.

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

[10] Y. Zhang, F. Patwa, R. Sandhu, and B. Tang. Hierarchical secure information and resource sharing in openstack community cloud. In *IEEE Conference on Information Reuse and Integration (IRI)*, 2015.