Privacy Control Model for Academic Information in Private Cloud System

Ei Ei Mon, Thinn Thu Naing
University of Computer Studies, Yangon
eemucsy@gmail.com,ucsy21@most.gov.mm

Abstract

Cloud computing is one of today's most attractive technology areas due to its many advantages like Highly scalable, on-demand, web-accessed IT resources with major cost / cash and flexibility. In this paper, the privacy policies and access control strategies for academic private cloud system are proposed to reduce the risk such as stealing and misuse of the private personal data, user profile management and accessing services offered on the private cloud environment. This system has been configured with Eucalyptus open source cloud infrastructure in which the privacy policies and access control strategies are enhanced. In order to solve these security problems on the private cloud model, we can take advantages of the existing Eucalyptus cloud infrastructure and security in which we can elect to store highly sensitive data of the university in the private storage cloud and less sensitive data in public storage of the private cloud. The main idea of privacy enhancement on a private cloud computing model is applying the role-based access control model as well as deploying privacy policies and security constraints into the sensitive resources. For security purpose, this paper extends the Role Based Access Control model with role relationships and delegation for access rights.

Keywords: Privacy, Eucalyptus, Role-based Access Control, Private Cloud

1. Introduction

Cloud computing is the state of the art technology that enables the functionality of an IT infrastructure, platform and products to be exposed as a set of services in a seamlessly scalable model so that the consumers of these services can use what they really want and pay for only those services that they use (pay per use). Computing is being transformed to a model consisting of services that are commoditized and delivered in a manner similar to traditional utilities such as water, electricity, gas, and telephony [4].

Using this model user can access services as per their requirement without knowing that where the services are hosted and how they are delivered. The cloud model represents nothing less than a fundamental change to the economics of computing and the location of computing resources [3]. With the growth in Internet usage, the proliferation of mobile devices and the need for energy and processing efficiency, the stage has been set for a different computing model – the idea of computing as a utility.

1.1 Cloud computing

Cloud computing is a model for enabling ubiquitous, convenient, on-demand network access to a shared pool of configurable computing resources (e.g., networks, servers, storage, applications, and services) that can be rapidly provisioned and released with minimal management effort or service provider interaction [3].

Today, clients are capable of running their software applications in remote computing clouds where data storage and processing resources could be acquired and released, almost, instantaneously. The virtualization layer on top of the commodity hardware in computing clouds is the driving force that allows cloud providers to "elastically" and promptly respond to client resource demands and requirements [13].

Cloud computing technology enables the IT functionality to be exposed as service in a multi-tenant manner. The enabling technology includes (but not limited to) virtualization, grid technologies, SaaS (Software as a service) enabled application platform (SEAP), Service Oriented Architecture (SOA), Metering tools and technologies etc. Cloud services are provided by the cloud vendor and that can be used by the cloud consumer on a pay per use basis. These service exposed as industry standard interfaces like web services (using Service Oriented Architecture SOA [13]) or Representational state transfer (REST) [15] services or any proprietary services.

Cloud computing is a computing paradigm, where a large pool of systems are connected in private or public networks, to provide dynamically scalable infrastructure for application, data and file storage. With the advent of this technology, the cost of computation, application hosting, content storage and delivery is reduced significantly.

The rest of this paper is organized as follows: Section 2 describes the related work. Section 3 mentions privacy of cloud computing. Section 4 mentions technique preliminaries and the proposed private cloud model for academic information system is described in section 5. Section 6 mentions the

functions involved in the cloud privacy model. In section 7, Delegation Access Rights to Virtual Machine based on Relationships of Role-based Access Control in Private Cloud. Section 8 describes the evaluation result of the proposed algorithm is mentioned. Finally this paper concludes in section 9.

2. Related Work

Research on cloud data privacy is still challenging since its early stages. The reports on the topic are presented in [12] which discusses the risks imposed by the adoption of cloud computing on data privacy and legal compliance, and [14] which emphasizes on the need to develop a sound digital identity infrastructure to support tackling privacy and security concerns in computing clouds. [6] and [7] present a comprehensive set of guidelines on designing privacy-aware cloud services. [6] summarizes the privacy patterns in 6 recommended practices: "(1) minimizing customer personal information sent to and stored in the computing cloud; (2) protecting sensitive customer information in the cloud; (3) Maximizing user control; (4) Allowing user choice; (5) Specifying and limiting the purpose of data usage; (6) providing the customer with privacy feedback". 2 to 6 are addressed in this paper. Data as a service offers data in various formats and from various sources could be accessed via services by users on the network, in a transparent, logical or semantic way. Users could, for example, manipulate remote data just like operate on a local disk or access data in a semantic way in the Internet [15]. Encryption keys should securely manage to prevent misuse or disclosure. Key management methodology can be used to secure key distribution. Keys should be recycled from time to time and old keys should be destroyed [16]. Encryption is the most important part of the cloud security. Encryption on the data can be used at various places [11], For example, data in transit, data at rest and data in memory or process. Encrypting and managing encryption keys of data in transit to the cloud or at rest in the service provider's data center are critical to protect data privacy and comply with compliance mandates [10].

3. **Privacy of Cloud Computing**

Privacy in cloud computing is the ability to control for a user or a business information that they are revealed about them-selves over the cloud or to a cloud service provider, and the ability to control who can access that information. Numerous existing privacy laws impose the standards for the collection, maintenance, use, and disclosure of personal identifiable information that must be satisfied by cloud providers. The nature of cloud computing has significant implications for the privacy of personal, business and governmental information. Cloud Service Providers (SPs) can store information at multiple locations or outsource it, then it is very difficult to determine, how secure it is and who has access to it [5]. A cloud SP is a third party that maintains information about, or on behalf of, another entity. Whenever an individual, a business, a government agency, or other entity shares information in the cloud, privacy or confidentiality questions may arise [5]. Trusting a third party requires taking the risk of assuming that the trusted third party will act as it is expected. The main problems associated with such a model are:

- Loss of control: Data, applications, and resources are located with SP. The cloud handles IDM as well as user access control rules, security policies and enforcement. The user has to rely on the provider to ensure data security and privacy, resource availability, monitoring of services and resources.
- Lack of trust: Trusting a third party requires taking risks. Basically trust and risk are opposite sides of the same coin. Some monitoring or auditing capabilities would be required to increase the level of trust.
- Multi-tenancy: Tenants share resources and may have opposing goals which could be conflicting. There is a need to provide a degree of separation between tenants.

4. Technique preliminaries

4.1. Role Based Access Control (RBAC)

Access control model is a framework that dictates access control using various access control technologies. RBAC is a more recent model than DAC and MAC. The central administration assigns roles to users and these roles determine what each user in a role can or cannot do. Since permissions need not be repeatedly assigned to individual users, RBAC scales better than DAC and reduces administration efforts.

RBAC simplifies management of permissions as users can be grouped into roles which are created for various job functions and responsibilities in an organization. Rights are given implicitly as each member of a role automatically receives the same rights as anyone else in the role. A role defines the terms of the operations and tasks the members of a role can perform. Thus, each role has a set of privileges and permissions assigned to it. Users can easily be reassigned to another role and permissions could be granted to or revoked from a role as needed.
It is important to distinguish roles as a policy component and separate policy from mechanism as roles can also be implemented using groups or

compartments. Conversely, MAC and DAC can also be simulated and enforced using RBAC. The principal concern is rather the protection of integrity of information. This point is however disputed as particular configurations of RBAC can also have a strong discretionary flavor.

The model can be seen as a combination of users, roles, permissions and constraints (for example, in the academic private cloud system) U= {Faculty, Staff, Teacher, Students, Researchers, Research Students, Cloud Service Provider, System Administrator, Privacy Manager, etc.,}, a set of roles R and a set of permissions P (where permission is an object-action pair). According to each user sub-sets, it needs to define the corresponding access role set R and a set of permission P (where permission is an object-action pair) [10].

Then in core RBAC an access control policy is specified by a user-role assignment relation UA ⊆ U The following table describes the notation specification for cloud privacy model:

U	A subset of user groups U= {{u1},{u2},{u3},…..{un}}, For example: U= {{Faculty}, {Staff}, {Teacher}, {Students}, {Researchers}, {Research Students}}
R	A set of roles R={r1,r2,r3,…..rn}, For example: R={cloud administrator, cloud user, cloud data manager, ….}
P	A set of permissions P={p1,p2,p3,…pn}, For example: P={(obj1,action1), (obj2,action2),….(objn,actionn) }
URA	user-role assignment relation URA ⊆ U × R
PRA	a permission-role assignment relation PRA ⊆ P ×R

Table 1. Notation for cloud privacy model

More formally, this system assumes the existence of set of users U in which the other sub-set user groups are included such as (for example, in the academic private cloud system) U= {Faculty, Staff, Teacher, Students, Researchers, Research Students}, a set of roles R and a set of permissions P (where permission is an object-action pair). According to each user sub-sets, it needs to define the corresponding access role set R and a set of permission P (where permission is an object-action pair) [1].

Then in core RBAC an access control policy is specified by a user-role assignment relation UA ⊆ U × R and a permission-role assignment relation PA ⊆ P ×R. A user u is authorized for permission p if there exists a role r ∈ R such that (u, r) ∈ UA and (p, r) ∈ PA. In this case, a user u is authorized for permission p if there exist roles r and r′ such that (u, r) ∈ UA, r > r′ and (p, r′) ∈ PA.

The use of roles to control access can be an effective means for developing and enforcing enterprise-specific security policies, and for streamlining the security management process.

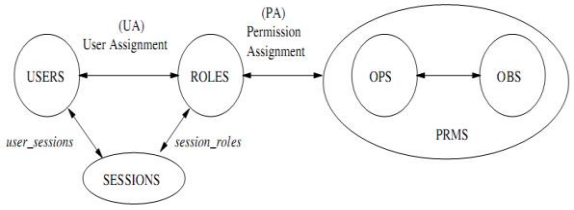

Figure 1: Core RBAC Model

4.1.1 Elements of an RBAC system

Typically, in RBAC, a user can have multiple roles to which different permissions can be granted. The elements of a typical RBAC system are:

- Subjects (Agents, Users)
- Objects (Resources) to which access should be controlled
- Access Modes (e.g. "read", "write", "execute", etc.) - sometimes also called "operations" or "actions"
- Permissions that define a certain mode of access to a resource
- Roles

Subjects and permissions are assigned to roles. Roles are thereby an efficient way to define which subject has access to which resource and through which access mode. Refinements of this basic access control model are possible:

- Roles can be organized in a hierarchy whereby a subordinate role "inherits" permission assignments of its parent role.

Subjects, objects, access modes and permissions can be organized in groups so that they can be assigned or otherwise handled in "bulk". Groups of permissions are called "policies". × R and a permission-role assignment relation PA ⊆ P ×R. A user u is authorized for permission p if there exists a role r ∈ R such that (u, r) ∈ UA and (p, r) ∈ PA. In this case, a user u is authorized for permission p if there exist roles r and r′ such that (u, r) ∈ UA, r > r′ and (p, r′) ∈ PA.

4.1.2 Elements of an RBAC system

Subjects and permissions are assigned to roles. Roles are thereby an efficient way to define which subject has access to which resource and through which access mode.

Refinements of this basic access control model are possible:

- Roles can be organized in a hierarchy whereby a subordinate role "inherits" permission assignments of its parent role.
- Subjects, objects, access modes and permissions can be organized in groups so that they can be assigned or otherwise handled in "bulk". Groups of permissions are called "policies".

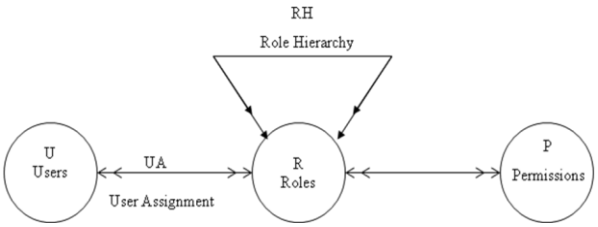

Figure 2: Simplified Version of RBAC Model In Hierarchical roles

5. Proposed System

5.1 System Architecture

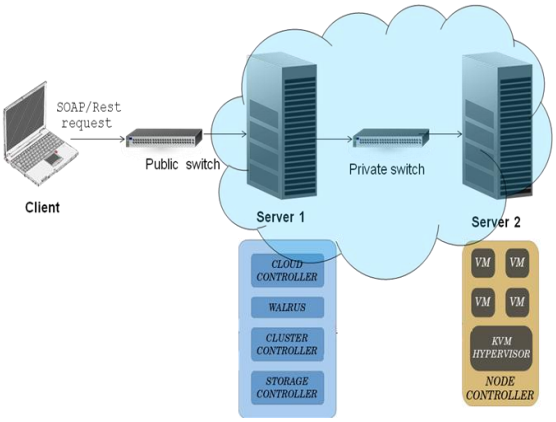

Figure 3. The architecture of the private cloud using Eucalyptus

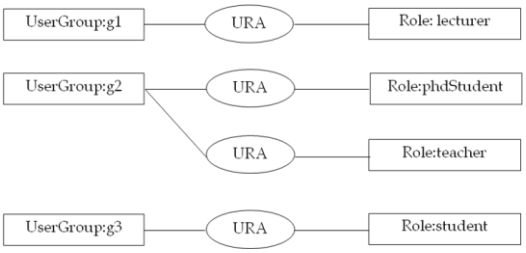

Figure 4. The sample graph for user-role assignment

This system is implemented by using RBAC with privacy constraints. In RBAC, Access Control policy is embodied in various components of RBAC such as

- Role-Permission relationships
- User-Role relationships
- Role-Role relationships

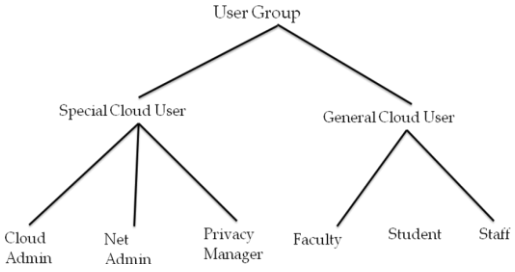

Figure 5. The available user group in the private cloud system

These components collectively determine whether a particular user will be allowed to access a particular piece of data in the system. RBAC components may be configured directly by the system owner or indirectly by appropriate roles as delegated by the system owner. The policy enforced in a particular system is the net result of the precise configuration of various RBAC components as directed by the system owner. The ability to modify policy to meet the changing needs of an organization is an important benefit of RBAC.

5.2 Resources defined

The proposed system grants access to private cloud users which represent (=identify) users accessing the private cloud services. It is of course inefficient to define individual access levels for every user. We therefore have to find a way to group users so that they can be granted access in a more efficient way.

We have to make sure that we make the groups small enough to be able to define the level of differentiated access that we need to implement our specifications. At the same time, however, we want the groups to be as large (and as few) as possible to keep the administrative burden down.

5.3 Roles

The basic unit that can be used to group users with the same access levels is called a "user role" in the private cloud system. A user role needs to unequivocally define all resources and services available to its members. Therefore, we need to create roles in a way that within one role all users have the same homogeneous permissions but that no two roles are equal.

5.4 User Groups

The private cloud system allows the cloud users to create and maintain subgroups within user roles that, from a technical viewpoint, provide the exact same level of access to the application. The private cloud

system contains a number of pre-defined user groups but organizations can freely add, rename or delete user groups per role (teacher, student, etc.). A user group always has one of the application roles assigned which unequivocally defines the roles and object access rules for that group.

5.4 Individual Rights Assignment

Based on user groups it is now possible to define highly flexible object access. A user can be granted access to an object within one of the user groups she has been assigned to. This gives the user access to all the features defined for the role attached to the user group with respect to the assigned object or resources.

5.5 Restrictions on Roles

Different user groups that are based on the same basic role cannot be granted different types of access. This includes access to either features or objects. Example: If the special cloud user group and the general cloud user group belong to the same basic role (e.g. "low level role") then our current approach makes it impossible to give access to different services within the private cloud system. The two user groups cannot be based on the same role model. The implication is that system administrators and owners of the resources need to be based on two distinct roles.

6. The functions involved in the cloud privacy model

The following table describes the command to add user and role as well as assigning user and role together:

AddUser(user): pre-cond: user notin USERS; USERS = USERS + {user}	AddRole(role): pre-cond: role notin ROLES; ROLES = ROLES + {role}
AssignUser(user,role): pre-cond: user in USERS, role in ROLES, [user,role] notin UR; UR = UR + {[user, role]}	

The follow command can be used to assign the permission grant for cloud services, object and role:

```
GrantPermission(operation, object, role):
pre-cond: operation in OPS, object in OBJS, role in ROLES, [[operation,object],role] notin PR;
PR = PR + {[[operation,object],role]}
```

The following commands can be used to delete cloud users who are not going to involve in future.

```
DeleteUser(user):
pre-cond: user in USERS;
UR = UR - {[user,r]: r in ROLES} //maintain UR
for s in SESSIONS | //maintain SESSIONS
[s,user] in SU:
```

In order to delete the assigned sessions that have been assigned before.

```
DeleteSession(user,s)//DeleteSession defined below
USERS = USERS - {user}
ROLES = ROLES - {role}
UR = UR - {[user,role]}
```

Similarly, the assigned role can be removed by using following commands:

```
DeleteRole(role):
pre-cond: role in ROLES;
PR = PR - {[[op,o],role]: op in OPS, o in OBJS} //maintain PR
UR = UR - {[u,role]: u in USERS} //maintain UR
for s in SESSIONS, u in USERS | [s,u] in SU, [s,role] in SR: //maintain SESSIONS
```

On the other hand, if the user wants to assign again, the following command can be used:

```
DeassignUser(user, role):
pre-cond: user in USERS,role in ROLES,
[user,role] in UR;
for s in SESSIONS | [s,user] in SU,[s,role] in SR://maintain SESSIONS
```

The following command can be used to revoke permission offered by the private cloud.

```
RevokePermission(operation, object, role):
pre-cond: operation in OPS, object in OBJS,
role in ROLES, [[operation,object],role] in PR;
PR = PR - {[[operation,object],role]}
```

In order to create session for user request, the following command can be used.

```
CreateSession(user, session, ars):
pre-cond: user in USERS, session notin SESSIONS,
ars subset AssignedRoles(user);
// AssignedRoles is defined below
SU = SU + {[session,user]}
SR = SR + {[session,r]: r in ars}
SESSIONS = SESSIONS + {session}
```

The following commands can be applied to add active and delete active role for the users in the private cloud system.

```
AddActiveRole(user, session, role):
pre-cond: user in USERS, session in SESSIONS,
role in ROLES, [session,user] in SU, [session,role] notin SR,
role in AssignedRoles(user);
SR = SR + {[session,role]}
```

```
DropActiveRole(user, session, role):
pre-cond: user in USERS, session in SESSIONS, role in ROLES, [session,user] in SU, [session,role] in SR;
SR = SR - {[session,role]}
```

The permission role assignment, user to role permission, session role mapping session permission, role to operation mapping on objects and user to operation mapping on object commands are as follows:

```
RolePermissions(role):
pre-cond: role in ROLES;
return {[op,o]: op in OPS, o in OBJS
| [[op,o],role] in PR}
```

UserPermissions(user):
pre-cond: user in USERS;
return {[op,o]: r in ROLES, op in OPS, o in OBJS | [user,r] in UR, [[op,o],r] in PR}

SessionRoles(session):
pre-cond: session in SESSIONS;
return {r: r in ROLES | [session,r] in SR}

SessionPermissions(session):
pre-cond: session in SESSIONS;
return {[op,o]: r in ROLES, op in OPS, o in OBJS | [session,r] in SR, [[op,o],r] in PR}

RoleOperationsOnObject(role, object):
pre-cond: role in ROLES, object in OBJS;
return {op: op in OPS | [[op,object],role] in PR}

UserOperationsOnObject(user, object):
pre-cond: user in USERS, object in OBJS;
return {op: r in ROLES, op in OPS
| [user,r] in UR, [[op,object],r] in PR}

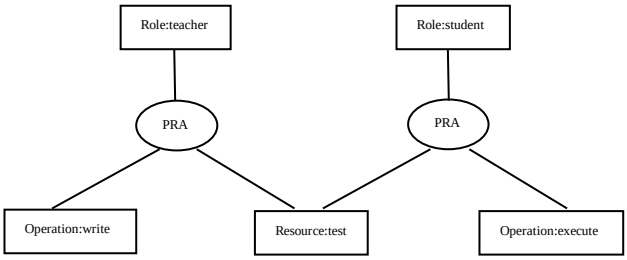

Figure 6: Role-Permission Assignment Sample Graph

Figure 6 shows the role-permission assignment graph for role to operation mapping in the proposed model. The data security level is classified (described in Table 2) based on significance and sensitivity in the organization.

The proposed check access algorithm academic information system checks all the basic access control decision points those should be considered when using role-based access control model for academic information system in the private cloud. In Figure 4, the user role assignments (URA) are being mapped between user group and their respective roles.

Data Security Level	**Type of Privacy**	**Properties**
Full trust(LL)	No Privacy (NP)	This data is not sensitive and can be disclosed in public.
Compliance-based trust (AL)	Privacy with trusted provider (PTP)	This level of data is sensitive. They should not be disclosed because users can access according to their roles with RBAC.
No trust (HL)	Privacy with non-trusted provider (NPTP)	This level of data is very sensitive and can be accessed only by privileged users.

Table 2: The predefined data security level classification

The assignment of subjects (users) to user groups, the assignment of user groups to roles and the assignment of users to certain types of data objects (e.g. special or general cloud services) are controlled as follows:

- User-user group assignment is done on the user registration pages which shows all user groups and allows users with managerial access level to add/remove users to/from user groups.
- For the moment being user group-role assignment is only relevant to the private cloud system does not yet implement the user group abstraction but directly assign users to roles (as shown in figure 5). In the system, there will be a grid which is part of the profile settings that allows members of managerial roles to add new user groups with a given role assigned or remove user groups from a role.

Assignment of users to data objects is done at several roles in the private cloud system. In the case of delegation, we implement an "owner" concept. This means that the submitter of resources is automatically being assigned to it as its creator.

Symbol	Definition
p	permission
PRMS	Set of permission
PA	Authorized permission
r	Role
ts	Time stamp

Table 3. Notation for CheckAccess Algorithm

6.1 CheckAccess Algorithm

```
boolean CheckAccess (session, operation, object)
{
p = (operation,object)
p ∈ PRMS
access_role = GetMinAccessRole (session_roles( session), perm);
if (∃(r, ts) ∈ active_session_roles : (p, r) ∈ in PA)
{
access_role.ts = current_system_time;
return true;
}
if (∃(r, ts) ∈ expired_session_roles : (p, r) ∈ in PA) {
if (RoleFaultHandler(access_role)) {
access_role.ts = current_system_time;
return true;
}
return false;
}
return false;
}
```

In this algorithm, there is predefined a list of all active roles in the current session (active_session_roles) and a list of all expired roles in the current session (expired_session_roles). GetMinAccessRole (session_roles(session), perm) identifies the minimum access role. This role is, according to the well ordering, the smallest role in the set of session roles that contains the needed permission. There is an active role with the needed permission. RoleFaultHandler(access_role) check role fault case.There is no active role containing the needed permission, but only an expired one. If role fault handler could not be answered , the grant is access denied. This algorithm will ouptut Access denied if Permission not at all in user session.

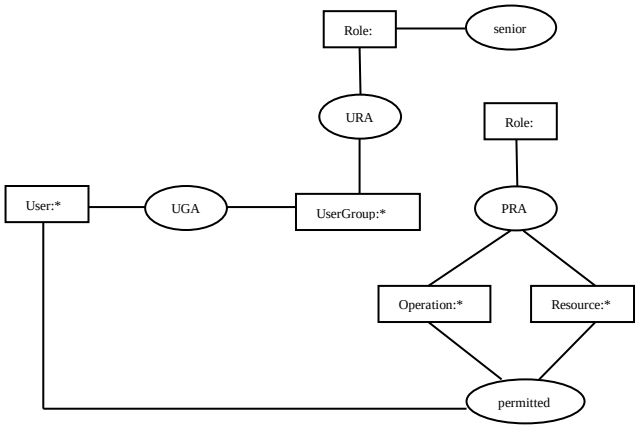

Figure 7: The authorization decision rule sample graph

Roles are identified with various job functions in an organization and users are assigned to roles based on their job responsibilities and qualifications. Permissions are associated with roles. Users acquire permissions through the roles allocated for them.

7. Delegation Access Rights to Virtual Machine based on Relationships of Role-based Access Control in Private Cloud

For security purpose, this paper extends the Role Based Access Control model with context awareness, exceptions and delegation. By combining this extended model with delegation of RBAC based on role relationships is for controlling shared virtual machine access rights in private cloud environment. This feature of role-based models greatly simplifies the management of permissions. Delegation is a mechanism of assigning access rights to a user. Delegation may occur in two forms: administrative delegation and user delegation. An administrative delegation allows an administrative user to assign access rights to a user and does not, necessarily, require that the administrative user possesses the ability to use the access right. A user delegation allows a user to assign a subset of his available rights to another user. However, a user delegation operation requires that the user performing the delegation must possess the ability to use the access right. User delegation is a mechanism for assigning access rights available to one user to another user. A delegation can either be a grant or transfer operation. This paper emphasizes the user delegation mechanism based on their role relationships. User delegation is defined by the following relationships (Figure 7).

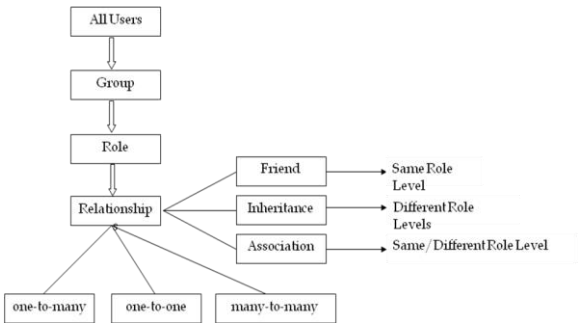

Figure 8: The user roles and relationships graph

7.1 Defined Authentication and Access to VMs for Cloud Users

This system implements a cloud control, privacy and access VMs for the cloud user who are involved in the private cloud system. The privacy manager control access management enforcement to minimize loss of control and manage the bounded limit of virtual machines (VMs) allowed for each user. It is also manage the type of VM according to security rules. This system provides authentication filtering for the instances based on the concept of security groups. A Security Group is a named set of rules that the private cloud system applies to the incoming

packets for the instances in Managed and Managed-NOVLAN modes. The cloud privacy manager can specify a security group while launching an instance. Each security group can have multiple rules associated with it. Each rule specifies the source IP/network, protocol type, destination ports etc. Any packet matching these parameters specified in a rule is allowed in. Rests of the packets are blocked. The basic model defines only the smallest set of relations to allow a role-based access control setting. This model contains four entities: Users, Roles, Permissions, and Sessions. This extension to the base model defines a relation, which is a partial order, on roles that defines permissions that are inherited from another role. In this system, the owner of VM can launch his own VM by using his private keypair and he can send his keypair to another who can have relationships with him. The owner can send his key according to the relationship types. The receiver can launch the VM of the owner with the receiving private key according to the permissions. To enhance security of key delegation, this system use encryption method to encrypt key. In private cloud system, the cloud controller generates a keypair consisting of private key/public key to be able to launch instances on Eucalyptus. These keys are injected into the instance and used to make passwordless SSH acess to the instance possible. This creates a new keypair called mykey.
Keypairs can also be generated using the following commands:

```
if [ ! -e ~/.euca/mykey.priv ]; then
    mkdir -p -m 700 ~/.euca
    touch ~/.euca/mykey.priv
    chmod 0600 ~/.euca/mykey.priv
add-keypair mykey > ~/.euca/mykey.priv
fi
```

Before running an instance of the image, it should be first created a keypair (ssh key) that can be used to log into the instance as root, once it boots. The key is stored have to do this once. Run the following command:

```
$ euca run instance emi 721D0EBA – eemkey c1.medium
```

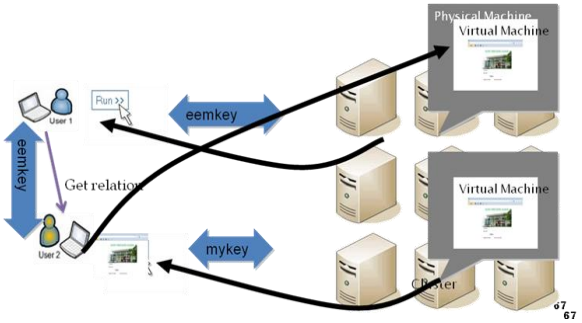

Figure 9: Key delegation of VM access rights

Figure 10: The professor can delegate access rights to her candidate

8. Evaluation Result of the System

The proposed CheckAccess Algorithm has been simulated in several users and roles within the academic-based information system in the private cloud. There are 3 administrative roles, 500 users, 15 basic roles, 50 processes, 50 resources and more than 100 objects in the system. There are 15 access control rules in RPA. The User-Role-Permission algorithm that is proportional to the number of roles and tasks assigned with each user.

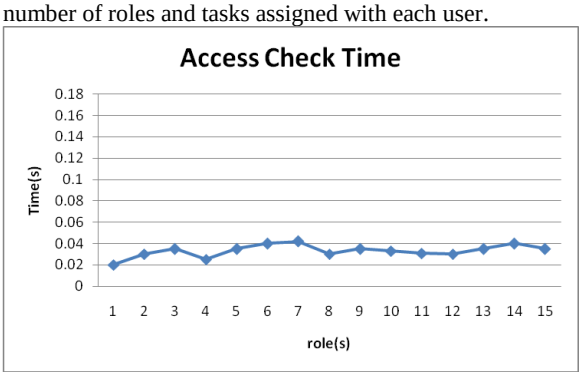

Figure 11 : Access Check Time for the proposed system

For a user with 15 roles, the user-permission assignment requires several seconds. Figure 11 shows the access check time for each role. The average access check time for each operation is about 0.03 seconds. The average access check time for each operation is about 0.03 seconds.

9. Conclusion

The proposed system is implemented by using RBAC with addition of privacy rules or constraints to yield two major benefits: reduced administrative overhead and improved security. Consequently, users can gain faster access to systems critical to their jobs. This technology is designed to minimize the potential for inside security violations by providing greater control over users' access to information and

resources. This system uses role-based access control model with privacy policies to limit access. The users of the private cloud system can access their resources against interference. Therefore, this system can enhance the security of the cloud and protect access from the unauthorized users. The enhanced security provided by RBAC model is presumed with privacy policies. Mandatory to mean that individual users do not have any choice regarding which permissions or users are assigned to a role, where discretionary signifies that individual users make these decisions. So, RBAC can have a strong mandatory flavor, while others can have a strong discretionary flavor.

References

[1] Cavoukian, "Privacy in the clouds", in Springer Identity in the Information Society, Published online: 18 December 2008.

[2] R.K.Buyya, Cloud computing and emerging IT platforms: Vision, hype, and reality for delivering computing as the 5th utility, http:// www.elsevier.com/wps/find/journaldescription.cws_home/505611/description#description.

[3] N.G. Carr, "The many ways cloud computing will disrupt IT," InfoWorld, March 25, 2009. [Online].Available: http://www.tmcnet.com/ usubmit/ 2009/03/25/4084363.htm

[4] National Institute of Standards and Technology (NIST) Definition of Cloud Computing, http://csrc.nist.gov/groups/SNS/cloud-computing/

[5] The open group service oriented architecture. http://www.opengroup.org/projects/soa

[6] REST – Representational State Transfer. http:// www.ics.uci.edu/filding/pubs/dissertation/top.html

[7] W,izhe, T.Jie, K.Marcel, R.Dharminder The Cumulus Project: Build a Scientific Cloud for a Data Center http://www.cca08.org/papers/Paper29-Lizhe-Wang.pdf

[8] IIT Hyderabad, Cloud Computing for EGovernance, http://search.iiit.ac.in/uploads/CloudComputingForEGovernance.pdf

[9] Security Guidance for Critical Areas of Focus in Cloud Computing V2.1 http:// cloudsecurityalliance. org /csaguide .pdf

[10] Cryptography as a service, http://www. infoassurance. org/ Public%20Docs/ ISSA_Cryptography as a Service.pdf

[11] Cloud computing information assurance framework, http://www.enisa.europa.eu/act/rm/files/deliverables/cloud-computing-information-assurance-framework .pdf

[12] IBM Point of View: Security and Cloud Computing, ftp://public.dhe.ibm.com/common/ssi/sa/wh/n/tiw14045usen /TIW14045USEN_HR.PDF

[13] S. Pearson, "Taking Account of Privacy when Designing Cloud Computing Services", in Proceedings of ICSE-Cloud'09, Vancouver, 2009

[14] S. Pearson and A. Charlesworth, "Accountability as a Way Forward for Privacy Protection in the Cloud", HP Labs Technical Report, HPL2009178, http://www.pl.hp.com /techreports/2009/HPL-2009-178.pdf (2009)

[15] V. Purohit, "*Authentication and Access Control*", the Cornerstone of Information Security, September 2007