DESIGN A FINE GRAIN ROLE BASED ACCESS CONTROL FRAMEWORK FOR
CLOUD COMPUTING

by

Shefali Modi

B.S. Punjab Technical University, 2005

A thesis submitted to the
University of Colorado at Denver
in partial fulfillment
of the requirements for the degree of
Master of Science
Computer Science
2012
The thesis for the Master of Science degree by

Shefali Modi

has been approved for the

Master of Science in Computer Science

by

Dr. Ilkyeun Ra, Chair

Dr. Tom Altman

Dr. Bogdan Chlebus

November 13, 2012
Shefali Modi (M.S. Computer Science)

Design a fine grain role based access control framework for cloud computing

Thesis directed by Assistant Professor Ilkyeun RA

ABSTRACT

Cloud Computing becomes very attractive and widely accepted as new paradigm of computing environment by IT industry in today. Its main strengths like service ubiquity, virtually unlimited computing resources, low cost of TCO (Total cost of ownership) and high ROI (Return of Investment). However, despite of this growing popularity of cloud computing, securing cloud computing resources still remains as a major challenge and concern for many organizations who either already adapt and use the cloud computing or are planning to migrate their business applications to cloud.

The main reason for this concern is originated from sharing cloud resources with others’ applications (multi-tenancy), and the physical location of stored data that could be subject to different legal regulations by the physical location. Furthermore, customers have no knowledge whom they are sharing their data with, and how securely their data is maintained and protected. Especially, their concerns even escalate when their data contain their customers’ private information and their company’s business secret data.

There are many research efforts to resolve the cloud computing security issues in various aspects such as data encryption, securing data communication, firewalls, strong user authentication, and access control management. We believe that providing secure and reliable cloud computing begins with securing cloud resources from malicious or unauthorized access.

In this thesis, we proposes a fine grained role based access control framework with various features including security of sensitive data, fine grained authorization policy and secure data from hackers. Our proposed role based access control algorithm provides tailored and fine level of user access control services without adding complexity, and supports access privileges updates dynamically when a user’s role is added or updated.

The form and content of this abstract are approved. I recommend its publication.

Approved: Ilkyeun Ra
ACKNOWLEDGEMENTS

My sincere thanks to my advisor, Ilkyeun Ra, for all his insight and support throughout my graduate studies. I would like to thank all my committee members for their insight and participation.
# TABLE OF CONTENTS

Chapter

1. Introduction ................................................................. 1
   1.1 Motivation ............................................................... 1
   1.2 Problem Statement ..................................................... 2
   1.3 Outline ................................................................. 3

2. Related Work .............................................................. 4
   2.1 Cloud Computing ....................................................... 4
   2.2 Deployment Models .................................................... 6
      2.2.1 Public Cloud ...................................................... 6
      2.2.2 Private Cloud ................................................... 7
      2.2.3 Hybrid cloud .................................................... 8
      2.2.4 Community Cloud ............................................... 9
   2.3 Cloud Architecture ................................................... 10
      2.3.1 Software as a Services (SaaS) ................................. 10
      2.3.2 Platform as a Service (PaaS) ................................ 10
      2.3.3 Infrastructure as a Service (IaaS) ............................ 10
   2.4 Benefits of Cloud Computing ....................................... 11
   2.5 Cloud Security and Privacy ......................................... 12
   2.6 Role Based Access Control (RBAC) ................................ 13
      2.6.1 RBAC stages .................................................... 14
   2.7 RBAC Framework .................................................... 15
      2.7.1 Eucalyptus ...................................................... 15
      2.7.2 How RBAC works in Eucalyptus ............................... 18
LIST OF FIGURES

Figure

2.1 Cloud Computing [12] ........................................................................................................5
2.2.1 Public Cloud [10]........................................................................................................7
2.2.2 Private Cloud [10].........................................................................................................8
2.2.3 Hybrid Cloud [10].........................................................................................................9
2.2.4 Community Cloud [10]..................................................................................................9
2.3 Cloud Computing stack [14]...........................................................................................10
2.7.1 Eucalyptus Architecture [9] ........................................................................................16
2.7.1.1 Eucalyptus Framework [9]......................................................................................17
3.2 Proposed Framework .........................................................................................................21
4.1 Oracle Tables......................................................................................................................29
4.2 Traditional Framework Results ........................................................................................30
4.3 Proposed Framework Results ..........................................................................................31
4.3.1 Proposed framework case table ..................................................................................32
4.3.2 Proposed framework case implementation .................................................................33
LIST OF TABLES

Table

4.5 Comparison Table.................................................................34
1. Introduction

The era of cloud computing started in the year of 2011. From that year onwards every organization wants to shift their business to this paradigm. But everybody is confused about what cloud computing is? Is this a new paradigm? No, this is not. This story starts with when the IT companies stores or centralizes their data on servers present on their premises. Slowly, organizations start sharing these servers with other business organizations in shared service centers (SSC). With passing years; they started outsourcing to third parties. With the evolution of IT Cloud Computing becomes the new paradigm for the organizations [1].

With the advent of cloud computing servers stores all the applications and data with the help of virtualization technology. With this virtualization technology, applications present in the cloud computing can be run independently without any particular configuration [2]. The features like on demand services, pay-per-use, elasticity, broad network access, resource pooling make cloud computing more appealing. All these services are provided in three service models which give information about the component delivered by cloud service provider; Infrastructure as a service (IaaS), Platform as a service (PaaS), Software as a service (SaaS). There are four deployment models in cloud computing on which these services are provided and which tells with whom resources are shared; public cloud, private cloud, hybrid cloud and community cloud. These concepts are discussed in chapter 2.

With all cloud computing boon of delivering services and infrastructures it is still in a question of using it or not due to its security concerns. The major security issues in cloud computing are distributed processing technology, massive network traffic issue, virtualization technology issue, application security, access control, authentication and password. One should get the full benefits of cloud computing if all these issues are put into consideration and appropriate solutions are provided.

1.1 Motivation

As the popularity of cloud computing increases, more and more organizations want to migrate their data and applications to cloud computing. As a result the main
concern for all cloud service providers is to provide security to their information and to their data. For that the identity of all the users must be known to the cloud provider administrator. To solve the security problem of cloud computing, one should first solve the user access. By implementing role based access control (RBAC) cost and complexity of security can be reduced [3]. With RBAC, the administrators grant permissions to the roles that he created according to job functions performed in an organization, and then assign users to the roles on the basis of their specific job responsibilities. To access the cloud computing resources user first have to register themselves into one or more classes and get credentials to identify themselves [4]. In a cloud numbers of systems are implementing RBAC. Each system has its own user accounts or system accounts with credentials. As the environment grows, number of accounts will also increase which leads to the increase of credentials. And all this is managed by system administrator.

1.2 Problem Statement

A RBAC concept is used by many clouds computing software’s. One of the examples is Eucalyptus. In Eucalyptus RBAC an administrator creates users, groups, and assigns policies to all the users. As discussed in previous section if the number of users grows their credentials, data and information will also increase. The main aim of the cloud is to share and manage the identity and credentials of cloud users in a seamless and secure manner. As all the information is increasing day by day managed in a distributed manner leads to potential point failure or human factor open. When the human factor comes in a large oversight potential can occur due to which the sensitive data can be stored in insecure places or can be assign to unauthorized user. Due to wrong storage this data can be accessed by hackers or unauthorized users which led to security breach.

To overcome the above challenges, we proposes a fine grained role based access control framework with features like securing sensitive information, giving fine grained authorization policy and securing data from unauthorized users.
1.3 Outline

This thesis is organized into five chapters. The first chapter is the introduction where the main motivation behind working on this thesis is discussed. This chapter also discusses the problem related to RBAC. Second chapter concludes the literature review of cloud computing, different types of service and deployment models, benefits of cloud computing. This chapter also discuss about RBAC model, how it works and its framework in Eucalyptus. Chapter third introduces the proposed framework after discussing limitations of Eucalyptus framework. In this chapter various algorithms and flowcharts related to proposed framework are also discussed. Chapter four includes the evaluations that are done in Java to compare the running time of proposed framework and traditional framework. Finally, chapter fifth will present conclusion and any future work.
2. Related Work

For providing secure and reliable cloud computing one should first secure the cloud resources from unauthorized access. Now a day’s many cloud computing platforms implementing role based access control. Still lots of researches are going on to secure RBAC in cloud. Georgia institute of Technology introduced a middleware security platform CASA which provides security with user bio information or location information [5]. For context-information modeling SOCAM proposes OWL, which consists of several components [6]. Komlenovic proposes distributed access for role based access control. Their approach uses directed graph, access matrix. If there is limit on number of users and permission than access matrix is an optimal choice and if it is variable then directed graph [7]. Ching-Ching Lee proposes distributed authorization caching technique which helps to improve performance, scalability of an authorization system [8]. Ei Ei Mon combines RBAC and Attribute based access control system and proposes a new framework ARBAC which supports both mandatory and discretionary needs [9].

2.1 Cloud Computing

Different researchers have different definition for cloud computing. Few of them are:

- NIST [10] definition of cloud computing: “Cloud computing is a model for enabling 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”.

- Buyya [11] defined Cloud Computing as follows: “A Cloud is a type of parallel and distributed system consisting of a collection of inter-connected and virtualized computers that are dynamically provisioned and presented as one or more unified computing resource(s) based on service-level agreements established through negotiation between the service provider and consumers.”
In the software ecosystem, cloud computing is being defined quite a lot of times, which has identical base – Internet. Earlier, big businesses were catered by hacks of servers, huge data centers; exuberantly investment over software’s and this cost was immense for small business to survive in this competitive market. With the advent of Cloud computing, the businesses are able to cut huge costs to a minimal, switched to a flexible operation and more secure.

Cloud Computing is a technology that creates a virtual ecosystem by centralizing the remote server to sustain data and applications. Cloud computing lets every consumer & business to employ & utilize the applications. It is an efficient way to access to data storage, processing & bandwidth. For instance, a consumer doesn’t need to install any software or having remote server just to share the data. All you need is to just log in to Yahoo email, Gmail, or Hotmail etc. to send emails as these businesses are based on cloud computing.

The driver of cloud computing is Virtualization (Hypervisor) and virtual appliance. The Hypervisor is software which is installed inside the computer to assist in downloading files. A virtual appliance is a function that works together with all components to run an operating system. End users are not exposed to the Virtualization of computers and operating systems as it is inbuilt application [7].

![Figure 2.1 Cloud Computing](image.png)
2.2 Deployment Models

As long as the consumer is connected to the Internet, he can access to various applications through any mode that can be a computer, a Smartphone or a personal digital assistant (PDA). The centralized server gives access to the pool of resources rather than depending on just a single dedicated server.

There are four types of cloud computing: Public Cloud, Private Cloud, Hybrid Cloud and Community Cloud [13].

2.2.1 Public Cloud

The provider provides the resources over a public network i.e. Internet services where consumer has no access over the operations. The server is stored outside the reach of the consumer by a third party they rely on. The major issue is of data privacy as it is a public network being stored on remote server. The trust level is a big concern and that is why is named as external cloud [13].

It is a basically based on the standard cloud computing model, where a service provider offers resources, like applications and storage. The services are either free or offered on a paid model.

The main benefits of using a public cloud service are:

- Trouble-free and economical set-up as a consumer need not to bear hardware, application and bandwidth costs
- High magnitude of usage covers investment
- Pay-as-per-usage model helps consumer to use seamless

For instance, businesses using Public cloud model Amazon Elastic Compute Cloud (EC2), IBM’s Blue Cloud, Sun Cloud, Google AppEngine and Windows Azure Services Platform [13].
2.2.2 Private Cloud

A private cloud is a service provided to an organization for commercial and business purposes. It’s a private networking with greater benefits including self-service, privacy, scalability & flexibility. Security & privacy are the chief reason to have private cloud application. Private cloud offers additional control and customization to the organization.

Organizations can install security modus operandi and screen the approach to the information. The hardware part is crucial as If it fails, the server automatically boots on the remaining node [13].

**FIGURE 2.2.1** Public Cloud [10].
2.2.3 Hybrid cloud

The composition of Public cloud & Private cloud forms Hybrid Cloud which interoperates between private cloud and public cloud. This service is typically offered in two ways:

A vendor with private cloud grows a partnership with a public cloud provider

For instance, an organization using a public cloud service, such as Amazon Simple Storage Service (Amazon S3) for archived data but persist to maintain in-house storage for effective customer data. Preferably, the hybrid approach gives opportunity for a business to be benefited of high-scalability and cost-effectiveness that a public cloud computing environment proposes a high-secured application which doesn’t expose critical applications, information and data to third-party vulnerabilities. This refers to hybrid IT [13].


**Figure 2.2.3** Hybrid Cloud [10]

### 2.2.4 Community Cloud

Community Cloud is the multi-tenant infrastructure being shared by a number of organizations & backs community with common concerns. The infrastructure may be managed & handles by third party or the organization. The aim of a community cloud is to bring the benefits of a private cloud in practice for participating organizations featuring multi-tenancy and a pay-as-you-go billing structure [13].

**Figure 2.2.4** Community Cloud [10].
2.3 Cloud Architecture

Cloud Computing has three architectures: Software (SaaS), Platform (PaaS) & Infrastructure (IaaS) [14]. These are detailed as below:

2.3.1 Software as a Service (SaaS)

Developers set down on software and it is authorized in order to set it up on their hard disk for further use. However, users need not to purchase the software, infect they can opt for pay-per-use model. It is multi-tenant based system as the server is used by many users [14].

2.3.2 Platform as a Service (PaaS)

PaaS provides computing platform and solution stack as a service. PaaS facilitates the operation of applications that includes no cost & there is no complexity of buying and managing the hardware & software hosting. The developers will make use of vendor’s block of code to make their applications [14].

2.3.3 Infrastructure as a Service (IaaS)

Here, vendors offer the infrastructure as a service in the form of technology, IT services & datacenters. To use the applications, cloud users install operating system metaphors on the machines as well as their application software. Under IaaS model, cloud user is responsible for the patching & maintains the operating systems and application software [14].

Figure 2.3 Cloud computing stack [14]
2.4 Benefits of Cloud Computing

- **Cloud Computing curtails the cost and has given immense space to online businesses.** IBM survey reveals that 31% like the concept of cloud’s “pay-as-you-go” cost structure. The cost of installing software, building hardware, license fees is nil in cloud. Online analytics is inexpensive as cloud offers approach to tools and computing control that could be possible with large set-ups only [15].

- **Cloud facilitates every kind of businesses to utilize computing means whenever are required.** IBM survey finds that 33% respondents find this as the greatest advantage. For Instance, Netflix uses Cloud Computing to come across up & down of subscriptions online for movies and TV shows. Referring to IBM* survey report, the report explains. “As Netflix began to outgrow its data center capabilities, the company made a decision to migrate its Website and streaming service from a traditional data center implementation to a cloud environment. This move allowed the company to grow and expand its customer base without having to build and support a data center footprint to meet its growth requirements” [15].

- **Cloud provides online entertainment to reach through any devise.** Cloud assists diverse group with various devices to access entertainment data with the familiarity of ActiveVideo, maker of CloudTV. ActiveVideo is a cloud-based proposal that blends all modes of content – Web, mobile, television, video-on-demand, social be it set-top boxes, PCs, or mobile devices. “CloudTV leverages content stored and processed in the network cloud to significantly expand the reach and availability of Web-based user experiences, as well as to allow operators to quickly deploy a consistent user interface across diverse set-top boxes and connected devices,” according to the IBM survey report [15].

- **It eases the access to services even if it is based on complex technology**

  20% of respondents in IBM survey states that technology complexity is not a
hindrance as it is not veiled to the end user. Navigation of services is easier through cloud computing. For Instance, Xerox, based on Cloud Print solution, let “workers can get their desired content in printed form wherever they might be by using Xerox’s cloud to access printers outside their own organization,” the report says. “While printing from the cloud requires quite a bit of data management – with numerous files to be stored, converted to print-ready format and distributed to printers –the complexity is hidden from users”[15].

2.5 Cloud Security and Privacy

The storage isn’t done over a system; it is done on a server. That causes a worry in the user's mind on the issue of security & privacy. The end user is concerned and would like to understand the confidentiality being kept by the service provider. Security threats can happen during the operations. Cloud environment is responsible to preserve the data integrity and privacy as well as improving the interoperability across several cloud service providers.

The data security & privacy is distributed on three levels [16].

- **Network Level:** The Cloud Service Provider (CSP) will scrutinize, examine, preserve and collect information regarding the firewalls, Intrusion detection or/and prevention systems and data flow in the network.
- **Host Level:** It is a crucial activity to gather information regarding system log files - where and when applications are being logged.
- **Application Level:** Reviewing application logs, which later is useful for incident response or digital forensics.

At every level, it is necessary to assure security requirements to maintain data security in the cloud as confidentiality, integrity and availability as follows:
• **Confidentiality**
  Confidentially can be maintained when user data can be protected from unauthorized user and this can be achieved by proper Encrypting techniques i.e. symmetric or asymmetric encryption algorithms. For example, MozyEnterprise executes encryption techniques [16]

• **Integrity**
  Integrity is also as important as confidentiality is for cloud users. Two approaches which provide integrity, using Message Authentication Code (MAC) and Digital Signature (DS).

• **Availability**
  Availability of the data is also another issue when it requested via authorized users. “The most powerful technique is prevention through avoiding threats affecting the availability of the service or data. It is very difficult to detect threats targeting the availability. Threats targeting availability can be either Network based attacks such as Distributed Denial of Service (DDoS) attacks or CSP availability”[16].

For the security of the application the credentials of the application users must be known in advance who that user is and what permission is given to that user. Many applications use Active Directory to maintain the user information. But in today’s cloud computing world certain API’s come into play when cloud users uses cloud services. These API’s must be designed in such a way that they can be protected from malicious and accidental attack. There is a great security risk to these API’s because these API’s contain the private credentials of cloud users. So proper access control should be provided to cloud users.

The next topic in this chapter is about Role Based Access Control in Cloud Computing.

### 2.6 Role Based Access Control (RBAC)

Role Based Access Control (RBAC) is a method that offers a satisfactory level of safety & security for organizational resources & data because of rules & policies put into effect for the user in the form of login & password. However, the description is not limited to
the organization resources but gives security and protection for users’ personal information and actions.

There are two main user attributes i.e. presence & location [16]. Presence is linked with the real-time communication systems such as: Instant Message and (IM) and Voice over IP (VoIP), where it gives the required explanation about users category all through the communication and even after that also, tells the status as idle or active, online or offline and for specific tasks it is done in the form of writing documents or email.

The current application Role Based Access Control RBAC offers Authentication, Authorization and Auditing for users using the cloud computing as follows:

- **Authentication:** Cloud computing authentication includes validating the identity of users or systems. For example, facility to service authentication engages in certifying the access demand to the information which served by another service.
- **Authorization:** After the authentication process, the system will put security rules to bring legitimate users.
- **Auditing:** Auditing is a process that involves reviewing & examining the records of authorization & authentication to check over organizations compliance with set security standards & policies in order to evade system breaches.

### 2.6.1 RBAC stages

According to Mather, Kumarasuamy and Latif [17], the rbac will go through five stages as follow:

- **Provisioning and deprovisioning:**
  User will be authorized to access to the information based on the organization & role. This process is long as every user is to be provided with an identity. Nevertheless, cloud management uses techniques such as identity Management as a Service (IDaaS).

- **Authentication and Authorization:**
  A significant authentication and authorization infrastructure will be requisite to make a custom authentication and authorization representation that fulfills the business goals.
• **Self-Service:**
  Facilitating self-service in the identity management will improvise the identity management systems. Users can reset their information like password and uphold their data from any location.

• **Password Management:**
  Single Sign on (SSO) support system is to access cloud-base services. Password management comprises of how the password will be stored in the cloud database.

• **Compliance and Audit:**
  Here, the access will be scrutinized & tracked to monitor the security breaches in the system. This process also assists to audit the fulfillment to diverse access control policies, periodic auditing and reporting.

### 2.7 RBAC Framework

To explain RBAC framework in this thesis we are going to take Eucalyptus paradigm as an example.

### 2.7.1 Eucalyptus

Eucalyptus—Elastic Utility Computer Architecture Linking your program to Useful System. As name suggests Eucalyptus is an open source software infrastructure to implement cloud on existing application. Eucalyptus is compatible with many hypervisor’s, virtualization technologies and platforms like Linux, Ubuntu, RHEL, Centos, OpenSUSE, Debian and Fedora [12].

Five levels of Eucalyptus Components are:

• **Cloud Controller (CLC):** Cloud controller is an entry point where administrators, developers, managers or end users can make their request. The main responsibility of CLC is to take information about resources from node, to make scheduling decision about resources and pass it to cluster.
- **Cluster Controller (CC):** Cluster usually runs on the machine that has connectivity between node controller (NC) or Cloud Controller (CLC). It helps to schedule a VM execution on Node after getting its information.

- **Node Controller (NC):** Node controller is executed on every node and VM instances are hosted on them. It helps to execute, terminate and inspect the VM on every host machine.

- **Storage Controller (SC):** Storage Controller helps to implement the block accessed network storage e.g. EBS and also interface with other storage systems like NFS, iSCSI etc.

- **Walrus:** Walrus helps the cloud users to store persistent data in the form of buckets and objects. It is compatible with Amazon S3 and supports Amazon Machine Image (AMI) [12].

**Figure 2.7.1** Eucalyptus Architecture [9].

End User first make a request to cloud controller and cloud controller will see what kind of request it is. If it is storage request then it will be forwarded to walrus which is compatible to Amazon S3 and then to storage controller. If it not a storage request the it will forwarded to cluster and then to individual node controller.
Figure 2.7.1.1 Eucalyptus Framework [9].

- **Data Owners**: In a cloud various services like data services, applications services and VM services can be created by cloud users and can be stored in cloud storage.
- **Data Users**: According to Data owner's permission cloud users can access their services and data.
- **Cloud Service Providers**: In cloud, cloud users can operate the cloud, its components and services according to the rule defined by cloud service providers.
- **Admin**: Admin has all the rights to authorize user and give him access rights according to the policy and keeps his information confidential from other unauthorized users.
2.7.2. How RBAC works in Eucalyptus

- Create the Admin: Once admin is created, make the admin group and create the policy controlling permissions for that group.

- Create User identities: Add Users (unique identities that can be used to interact with Cloud resources). A User can be an individual, system, or application requiring access to cloud resources.

- Assign and manage security credentials: Assign security credentials (such as access keys) to each User, and rotate and/or revoke these credentials as desired.

- Organize Users in groups: Create groups to more easily manage permissions for multiple Users.

- Centralized control of User access: Control which operations each User can perform, such as accessing specific resources.

- Conditional User access: Add conditions to control how a user can use resources, such as their originating IP address, time of day, or whether they are using SSL.
3. Design and Implementation

Our proposed approach is to provide fine grained role based access control in cloud while preserving the privacy.

3.1 Issues with existing approaches to RBAC

With the increasing demand of Cloud Computing, numbers of cloud users have increased abruptly. With this reason the security of cloud is main concern and the role based access control is in priority due to number of reasons:

- With RBAC large number of users can be handled securely.
- Help to reduce the complexity of work by managing the large number of groups of users.
- Help to provide authorization and authentication to a user in more secure manner.
- Database security can be managed easily with RBAC.

In the Eucalyptus paradigm of Role and Access management, as discussed in the previous section, the Admin manages all the groups. Policies are added to users account and users are added to the groups. When a user or group is added into an account, he/she will be provided with the credentials for the identity. The addition and deletion of users, groups and policies are all under admin’s supervision.

What if there is a big organization where there are large number of users and the access to data is controlled by security groups. With the increase in users the credentials, applications and then the human factor comes in. When human factor comes into consideration then the chances of putting sensitive data on wrong place, access given to wrong user or wrong security group will increase, and then the whole system will be in mess.

Each and every on cloud data needs to be protected, not all data is created equal. Some files contain confidential information; other files contain private information like
social security number, credit card number etc. Above all there is some kind of sensitive data that needs special authorization for processing.

In the Eucalyptus paradigm nothing is discussed about security of sensitive data.

3.2 The Proposed Framework

Our framework attempts to solve the abovementioned problems. The proposed architecture is shown in figure 3.2

- **Data Owners**: In a cloud various services like data services, applications services and VM services can be created by cloud users and can be stored in cloud storage.
- **Data Users**: According to Data owner’s permission cloud users can access their services and data.
- **Cloud Service Providers**: In cloud, cloud users can operate the cloud, its components and services according to the rule defined by cloud service providers.
- **Admin**: Admin has all the rights to authorize user and give him/her access rights according to the policy and keeps his information confidential from other unauthorized users. All the group owners are under Admin. With admin’s approval Group owners can be added and deleted. Also the users can be added and deleted with admin’s approval.
- **Group Owner**: Every group has its own group owner who will give all the access control, privacy privileges to the users’. If any user has to access the sensitive data, first he/she has to take permission from the group owner. Group owner will first check user’s credentials that if that user has rights to access that resource. If so then group owner will send a key to user’s email id. With that key only user can access the sensitive resource. One user can be place in number of groups.
There can be two possible cases with this framework.

**Case 1:** If a user is present in number of groups and the access rights that have given to him/her are different. In this case we will take optimistic approach. High priority will be given to less restriction. For example, if a user is present in group 1 as well same user is present in group 4. In group 1 the access rights of FULL ACCESS are given to him while in group 4 access rights of READ ONLY are given to him. So access rights of FULL ACCESS will be given to him. And if the access rights are of sensitive data then that user have to consult his group owner first and then he can access that resource.
Algorithm

1) [Initialize]. Set X: = 1, Y: = Number of User’s Group, access: = ‘No Access’, list [Length (y)]:= Groups of Users, Z.
2) If Y = 0, then access: = ‘User’s Access’ and Exit.
3) If Y = 1, then access: = ‘User’s Access in Group’ and Exit.
4) Repeat Steps 5 to 10 while X ≤ Y:
   5) Z: = list[X].getUserAccessInGroup.
   6) If Z = ‘Full Access’, then access: = ‘Full Access’ and Exit.
   7) If Z = ‘Read/Write’, then access: = ‘Read/Write’.
   8) If Z = ‘Read’ and access ≠ ‘Read/Write’, then access: = ‘Read’.
   9) If Z = ‘NO Access’ and access ≠ ‘Read/Write’ and access ≠ ‘Read’, then access:= ‘No Access’.
10) Set X: = X+1.
11) [End of Step 4 Loop]
12) Exit.

It first checks, in how many groups a user is present and that count is stored in variable y. For loop this count is stored in an array named list [length(y)]. Z is a variable that is used to store the access right of a user. If the user is present in one group only then the designated access right will be given. If the user is present in number of group then access rights will be decided according to less restriction.

Case 2: How shareable resources will be handled in this framework? Like if one user has access to read that resource and other user have access to write on that resource at same time. For this situation we will choose synchronization. If one user is accessing some resource then other user has to wait to access same resource.

Algorithm

[1] do{
[3] turn=j;
[4] while(flag[j]&turn==j);
resource access;
    remainder section
} while (TRUE);
In the entry section, user $i$ first raises a flag indicating a desire to access the resource. Then turn is set to $j$ to allow the other user to access the resource if user $j$ so desires. The while loop is a busy loop (notice the semicolon at the end), which makes user $i$ wait as long as user $j$ has the turn and accessing the resource. User $i$ lowers the flag[$i$] in the exit section, allowing user $j$ to continue if it has been waiting.

### 3.3 Flow Diagram

![Flow Diagram](image-url)
Is it sensitive resource?

Yes

Send email to Group admin to authorize user’s security

No

Group Admin Approve

Yes

User can access resource.

No

Resource will not be accessible.

User’s request will be in wait.

End
1

2

No

count := 0
x := No. of Groups

x := No. of Groups

count < x

Yes

Fetches User’s Access in Group

Full Access?

Yes

No

Assign read/write access to user.

count := count +1

Yes

If read/write access?

No

Assign read access to user.

No

If group access := read and read/write access has not been assigned to user?

Yes

Assign no access to user.

No

If group access is no access and (read/write or read ) access has not been assigned to user?
3.4 Features of Framework

With following features the proposed framework helps to secure the system more efficiently.

- **Security of Sensitive information:** The proposed framework helps to secure the private or sensitive information of user. When a user wants to access the sensitive information first an email will be sent to group owner, group owner will then checks the user’s credential to see if that user has access rights to use that data or not. If yes an email with security key will be sent to user. With that security key user can access that sensitive information.

- **Security from hackers:** Sometimes a user leaves his account open and anybody can access that account. If hacker wants to access any private information then group owner comes to know that somebody is misusing private data. So the group owner will block the access of that data.

- **Addition of user or group dynamically:** This framework helps to add or update user and groups dynamically. For example if a user is no longer working in an organization, a user’s access rights have been changed then this framework provides helps to make these updates dynamically.
4. Evaluation

In this part we will evaluate the running time of RBAC on traditional framework and the running time of RBAC on proposed framework. It will be first analyze the running time of traditional framework followed by proposed one. Finally, it will compare the running cost of both the framework and to see which framework is better than the other.

4.1 Platform

The platform to calculate the running time of RBAC on both frameworks is Java and Oracle is used to create tables of Access type, Eucalyptus Users, Eucalyptus Group.

**Access Type Table:** In Access Type table two fields are created Access Type ID and Access Type Name.

create table ACCESS_TYPE
(
    ACCESS_TYPE_ID      NUMERIC (1)    NOT NULL,
    ACCESS_TYPE_NM      VARCHAR (80),
    constraint XPK_ACCESS_TYPE primary key (ACCESS_TYPE_ID)
);

This table is populated with four types of Access
- **NOACCESS**
- **READ/WRITE ACCESS**
- **READ ACCESS**
- **FULL ACCESS**

**Eucalyptus Users:** In this table various Eucalyptus users and what type of Access is given to them is stored and fields in this are Eucalyptus User ID, User Login Name, User Password, User First Name, User Middle Name, User Last Name, User Email, Access Type ID

create table EUCALYPTUS_USERS
(
    EUCALYPTUS_USER_ID     NUMERIC (31)    NOT NULL,
    USER_LGN_NM            VARCHAR (80),
    USER_PSWD              VARCHAR (80),
    USER_NM_FST            VARCHAR (40),
    USER_NM_MID            CHAR (1),
...
Eucalyptus Group Table: This table contains information regarding groups that are created in Eucalyptus. Groups contain more than one user. Various fields in this table are Eucalyptus Group ID, Group Name, Group Description, Group Admin ID. Group Admin ID is created because every Group has its own Group Admin instead of one single Admin that controls all functions.

create table EUCALYPTUS_GROUP
(
    EUCALYPTUS_GROUP_ID NUMERIC (31) NOT NULL,
    GROUP_NM VARCHAR (80),
    GROUP_DESC VARCHAR (250),
    GROUP_ADMIN_ID NUMERIC (31),
    GROUP_VOID_IND CHAR (1),
    constraint XPK_EUCALYPTUS_GROUP primary key (EUCALYPTUS_GROUP_ID)
);

Group Users Cross Reference Table: This cross reference table helps to link records of Access table, User table and Group table. This cross reference table is used when there is many to many relationships between different tables. In our implementation there is many to many relationship between group table and user table. This table is used to bring normalization in our database. Various fields in this table are ID, Eucalyptus User ID, Eucalyptus Group ID, Access Type ID.

create table GROUP_USERS_XREF
(
    GUX_ID NUMERIC (31) NOT NULL,
    EUCALYPTUS_GROUP_ID NUMERIC (31),
    EUCALYPTUS_USER_ID NUMERIC (31),
    ACCESS_TYPE_ID NUMERIC (1),
    GUX_VOID_IND CHAR (1),
    constraint XPK_GUX primary key (GUX_ID)
);
It has been observed that if there are \( n \) groups and each group is having \( m \) users.
If admin will handle all groups and users policies and access rights and suppose admin is taking 1 min to handle 1 group’s policies, then for \( n \) group, total time = \( n \) minutes
Now if admin is taking 1 min to handle 1 user’s policy and access rights then for \( m \) users in a group,
Admin will take \( m \) minutes. So total time will be taken by admin to update \( m \) users access of \( n \) groups = \( m^n \times n \) minutes.

This we have implemented in Java using the above mentioned tables. So the results are:
Figure 4.2 Traditional Framework Results

Total time to update 10 users of 10 different groups by single admin is 468 milliseconds. If number of users increase with increase in number of groups then this time will increase with polynomial.

4.3 Implementation of Proposed Framework

It has been observed that if there are n groups and each group is having m users.

If each group will have its own group admin then they can work in parallel and it will take only m minutes for each group admin to change access rights of its m users of his group.

The same we implement in Java and the expected results are:
Figure 4.3 Proposed Framework Results

Total time to update 10 users of 10 different groups by different group admin is 296 milliseconds. If number of users will increase and so is groups having their own group admin’s who will work parallel, total time increase will be linear.

For the proposed framework we have two cases that we have discussed in previous chapter. We have implemented Case 1 if user is present in different groups and have different access rights.
Figure 4.3.1 Proposed framework case table

In this Eucalyptus User with ID 300 is present in two groups with ID's 20 and 30. The same user have given different access rights one is Read Write Access and other one is Full Access. As per our proposal access rights will less restriction will get priority. So this user will get access rights of Full Access.
4.4 Comparison

In previous sections we have discussed both the frameworks. As we have seen total time to updates the same number of users with same number of groups is different in both the cases. In traditional framework the time to update the users is more than that of proposed framework. Instead of increasing numbers of group owners, dividing information between public and private information this framework is taking less time to update users as compare to traditional framework. In traditional framework time increases polynomial as we increase number of users and number of groups. But in case of proposed framework time increases linearly as number of users and groups are increased.
<table>
<thead>
<tr>
<th>Function</th>
<th>Traditional</th>
<th>Proposed</th>
</tr>
</thead>
<tbody>
<tr>
<td>User Create</td>
<td>O(n)</td>
<td>O(n)</td>
</tr>
<tr>
<td>User Update</td>
<td>O(n)</td>
<td>O(n)</td>
</tr>
<tr>
<td>User Delete</td>
<td>O(n)</td>
<td>O(n)</td>
</tr>
<tr>
<td>Group Create</td>
<td>O(n)</td>
<td>O(n)</td>
</tr>
<tr>
<td>Group Update</td>
<td>O(n)</td>
<td>O(n)</td>
</tr>
<tr>
<td>Group Delete</td>
<td>O(n)</td>
<td>O(n)</td>
</tr>
<tr>
<td>Assigning Policy to User</td>
<td>O(n^2)</td>
<td>O(n)</td>
</tr>
<tr>
<td>Change Policy</td>
<td>O(n^2)</td>
<td>O(n)</td>
</tr>
<tr>
<td>Resource Access</td>
<td>O(n)</td>
<td>O(2n)</td>
</tr>
</tbody>
</table>

**Figure 4.4** Comparison table

Both in traditional and proposed framework time to create, update and delete users, groups is same O(n) because this create, update and delete is all done by Admin. Thus, same time will be taken to perform these functions. Whereas to assign policy in traditional framework is O (n^2) because only admin is there to assign policies to all the users in groups, if there are n groups and in groups there are n users. Now if admin is taking 1 min to handle 1 user's policy and access rights then for n users in a group, Admin will take n minutes. So total time will be taken by admin to update m users access of n groups = n*n=n^2 minutes but in proposed framework group owners of individual groups will assign the policy to users by checking their credentials. Same is the case with change policy. For resource access, as per our algorithm and coding, to access number of groups of a user, code will take O(n) time. E.g. if table Group_User_Xref is having n elements, to search groups of user, it will access n rows. And suppose a user is in n groups then once again for assigning appropriate access, loop will triggers in n times. So total cost will be n + n i.e. O (2n).

To summarize instead of containing more functionality and more division of roles the proposed framework takes less time or equal time to perform the function and provides more security. If the number of users and groups increase in proposed framework then
time to perform functions in this framework will also increase but that increase will be linear. The main motive behind proposed framework is to secure sensitive data. In cloud computing cloud users data is placed on different locations and no one knows who is using it and how. Proper authentication and authorization is needed to provide security to cloud user’s data, which is main functionality of proposed framework.
5. Conclusion and Future Work

In conclusion, today’s business environment is very attracted by cloud computing paradigm because of providing services in a very effective way. On top of commodity hardware there is a virtualization layer which is drive force and helps cloud providers to respond promptly to cloud user requests.

Instead of all these advantages of cloud computing, there is still a question mark on its usage. Security and privacy are main challenges from storage and processing of sensitive data due to multi-tenancy feature of cloud computing. For the efficient use of cloud computing providing proper security is very important. Cloud computing security begins with implementing Identity and Access Management to ensure Authentication, Authorization and Auditing.

The aim of this thesis is to propose a framework that protect the sensitive information in the cloud, specify the privacy policies for the private cloud; to protect the data from hackers. This framework uses fine grain role based access control. This framework takes less or equal time as compare to traditional framework in performing functions like creating new user, creating new group, assigning policy, accessing resources or changing policy. The proposed framework improves the security of the cloud and protects the data from unauthorized user, provides confidentiality, integrity and availability. The role based access control algorithms that we have proposed in this thesis backbone access privileges when a user or group is updated dynamically.

Future work includes the research of this framework further and implementation of framework on Eucalyptus.
REFERENCES


//  File: DAOImplementation.java
//  Date: 2011-11-13
//
//  Copyright © 2012, Shefali Modi

// In computer science, a data access object (DAO) is an object that provides an abstract interface to some type of database or other persistence mechanism. DAOs provide some specific data operations without exposing details of the database. This isolation separates the concerns of what data accesses the application needs, in terms of domain-specific objects and data types (the public interface of the DAO), and how these needs can be satisfied with a specific DBMS, database schema, etc. (the implementation of the DAO).

// GroupUserDaoImpl.java

package DAOImplementation;

import java.sql.ResultSet;
import java.sql.SQLException;
import java.util.ArrayList;
import java.util.List;
import DAOInterface.GroupUserDao;
import DomainObjects.GroupUserXREF;
import JDBCInfrastructure.DMConnection;
import JDBCInfrastructure.Echo;
import JDBCInfrastructure.Query;

public class GroupUserDaoImpl implements GroupUserDao{

    private DMConnection conn = null;
    private Query getUsersGroup = null;

    private String baseQry =
    "select eucalyptus_user_id,eucalyptus_group_id,access_type_id from group_users_xref
    where eucalyptus_user_id = 300 ";

    private String gs = baseQry;

    public GroupUserDaoImpl()
    {  
        conn = new DMConnection("jdbc:oracle:thin:@localhost:1521/WYLOCALDB", 
"jdbc:oracle:thin:@localhost:1521/WYLOCALDB", 
}
"sawasthi", "sawasthi");

 getUsersGroup = new Query(conn);
 getUsersGroup.setQueryString(gs);
}

private GroupUserXREF mapGroupUser(ResultSet rs)
{
 GroupUserXREF u = new GroupUserXREF();
 try
 {
   u.setUserId(rs.getLong("eucalyptus_user_id"));
   u.setGroupId(rs.getLong("eucalyptus_group_id"));
   u.setAccessId(rs.getLong("access_type_id"));
   return u;
 }
 catch(SQLException ex)
 {
   while (ex != null)
   {
     Echo.echo("SQLException/Error while Mapping Domain Object: ");
     Echo.echo("error message = " + ex.getMessage());
     Echo.echo("SQL State = " + ex.getSQLState());
     Echo.echo("Vendor Error Code = " + ex.getErrorCode());
     ex = ex.getNextException();
   }
 }
 return null;
}

public List<GroupUserXREF> getUserGroups()
{
 ResultSet rs = getUsersGroup.execute();
 List<GroupUserXREF> result = new ArrayList<GroupUserXREF>();

 try
 {
   while(rs.next())
   {
     GroupUserXREF e = mapGroupUser(rs);
     result.add(e);
   }
   return result;
 }
 catch(SQLException e)
 {

while (e != null)
{
    Echo.echo("SQLException/Error while Processing Result Set: ");
    Echo.echo("error message = " + e.getMessage());
    Echo.echo("SQL State = " + e.getSQLState());
    Echo.echo("Vendor Error Code = " + e.getErrorCode());
    e = e.getNextException();
}
return null;
}

// UserDaoImpl.java
package DAOImplementation;
import java.sql.ResultSet;
import java.sql.SQLException;
import DAOInterface.UserDao;
import DomainObjects.User;
import JDBCInfrastructure.DMConnection;
import JDBCInfrastructure.Echo;
import JDBCInfrastructure.Query;

public class UserDaoImp implements UserDao{
    private DMConnection conn = null;
    private Query getUserAccess = null;
    private String baseQry =
"SELECT ACCESS_TYPE_ID FROM EUCALYPTUS_USERS WHERE EUCALYPTUS_USER_ID = 3 ";
    private String gs = baseQry;
    public UserDaoImp()
    {
        conn = new DMConnection( 
"jdbc:oracle:thin:@localhost:1521/WYLOCALDB", 
"sawasthi", "sawasthi");
}
}
getUserAccess = new Query(conn);
getUserAccess.setStringQuery(gs);
}

private User mapUser(ResultSet rs)
{
    User u = new User();
    try {
        u.setAccessId(rs.getLong("access_type_id"));
        return u;
    }
    catch(SQLException ex) {
        while (ex != null) {
            Echo.echo("SQLException/Error while Mapping Domain Object: ");
            Echo.echo("error message = " + ex.getMessage());
            Echo.echo("SQL State = " + ex.getSQLState());
            Echo.echo("Vendor Error Code = " + ex.getErrorCode());
            ex = ex.getNextException();
        }
        return null;
    }
}

public Long getUsersAccess()
{
    ResultSet rs = getUserAccess.execute();
    Long result = null;
    try {
        while(rs.next()) {
            User e = mapUser(rs);
            result = e.getAccessId();
        }
    }
    catch(SQLException ex) {
        while (ex != null) {
            Echo.echo("SQLException/Error while Mapping Domain Object: ");
            Echo.echo("error message = " + ex.getMessage());
            Echo.echo("SQL State = " + ex.getSQLState());
            Echo.echo("Vendor Error Code = " + ex.getErrorCode());
            ex = ex.getNextException();
        }
        return null;
    }
}
return result;
}
catch(SQLException e)
{
    while (e != null)
    {
        Echo.echo("SQLException/Error while Processing Result Set: ");
        Echo.echo("error message = " + e.getMessage());
        Echo.echo("SQL State = " + e.getSQLState());
        Echo.echo("Vendor Error Code = " + e.getErrorCode());
        e = e.getNextException();
    }
    return null;
}
APPENDIX B
DAOInterface Class

// File: DAOInterface.java
// Date: 2011-11-13
//
// Copyright © 2012, Shefali Modi

// In computer science, a data access object (DAO) is an object that provides an
abstract interface to some type of database or other persistence mechanism. DAOs
provide some specific data operations without exposing details of the database. This
isolation separates the concerns of what data accesses the application needs, in terms of
domain-specific objects and data types (the public interface of the DAO), and how these
needs can be satisfied with a specific DBMS, database schema, etc. (the
implementation of the DAO).

// GroupUserDao.java
package DAOInterface;

import java.util.List;
import DomainObjects.GroupUserXREF;

public interface GroupUserDao {
    List<GroupUserXREF> getUserGroups();
}

// UserDao.java
package DAOInterface;

public interface UserDao {
    Long getUsersAccess();
}
APPENDIX C
Domain Object

// File: DomainObjects.java
// Date: 2011-11-13
//
// Copyright © 2012, Shefali Modi

// Domain object model contains the entities, their attributes and relationship among these entities. The domain object model gives the structural view of the domain.

// Group.java
	package DomainObjects;

	public class Group {
		private Long groupId;
		private String groupNm;
		private String groupDesc;
		private Long groupAdminId;

		public Long getGroupId() {
		return groupId;
	}
	public void setGroupId(Long groupId) {
		this.groupId = groupId;
	}
	public String getGroupNm() {
		return groupNm;
	}
	public void setGroupNm(String groupNm) {
		this.groupNm = groupNm;
	}
	public String getGroupDesc() {
		return groupDesc;
	}
	public void setGroupDesc(String groupDesc) {
		this.groupDesc = groupDesc;
	}

	public Long getGroupAdminId() {
		return groupAdminId;
	}
	public void setGroupAdminId(Long groupAdminId) {
		this.groupAdminId = groupAdminId;
	}
public class User {

    private Long userId;
    private String loginNm;
    private String loginPasswd;
    private String firstName;
    private String middleName;
    private String lastName;
    private String emailAddress;
    private Long accessId;

    public Long getUserId() {
        return userId;
    }

    public void setUserId(Long userId) {
        this.userId = userId;
    }

    public String getLoginNm() {
        return loginNm;
    }

    public void setLoginNm(String loginNm) {
        this.loginNm = loginNm;
    }

    public String getLoginPasswd() {
        return loginPasswd;
    }

    public void setLoginPasswd(String loginPasswd) {
        this.loginPasswd = loginPasswd;
    }

    public String getFirstName() {
        return firstName;
    }

    public void setFirstName(String firstName) {
    }
}
this.firstName = firstName;
}
public String getMiddleName() {
    return middleName;
}
public void setMiddleName(String middleName) {
    this.middleName = middleName;
}
public String getLastName() {
    return lastName;
}
public void setLastName(String lastName) {
    this.lastName = lastName;
}
public String getEmailAddress() {
    return emailAddress;
}
public void setEmailAddress(String emailAddress) {
    this.emailAddress = emailAddress;
}
public Long getAccessId() {
    return accessId;
}
public void setAccessId(Long accessId) {
    this.accessId = accessId;
}

//GroupUserXREF.java

package DomainObjects;

public class GroupUserXREF {
    private Long recordId;
    private Long groupId;
    private Long userId;
    private Long accessId;
    }
public Long getRecordId() {
    return recordId;
}
public void setRecordId(Long recordId) {
    this.recordId = recordId;
}

public Long getGroupId() {
    return groupId;
}
public void setGroupId(Long groupId) {
    this.groupId = groupId;
}

public Long getUserId() {
    return userId;
}
public void setUserId(Long userId) {
    this.userId = userId;
}

public Long getAccessId() {
    return accessId;
}
public void setAccessId(Long accessId) {
    this.accessId = accessId;
}
APPENDIX D
EXECUTION Class

// File: Execution.java
// Date: 2011-11-13
//
// Copyright © 2012, Shefali Modi

// In execution class the real execution takes places. All the
// DAOImplementation, DAOInterface, DomainObjects java files are imported. In this java
// files SQL queries are written.

// Test1.java

package Execution;
import java.util.ArrayList;
import java.util.Iterator;
import java.util.List;
import DAOImplementation.GroupUserDaoImpl;
import DAOImplementation.UserDaoImp;
import DAOInterface.GroupUserDao;
import DAOInterface.UserDao;
import DomainObjects.GroupUserXREF;
import JDBCInfrastructure.DMConnection;
import JDBCInfrastructure.Driver;

// This Class created to calculate total time taken by proposed algorithm

public class Test1 {

    public static void main(String arg[]) {
        long startTime = System.currentTimeMillis();
        System.out.println("Use Security Access of User Id 300 is "+ getUserSecurityAccess());
        long stopTime = System.currentTimeMillis();
        long elapsedTime = stopTime - startTime;
        System.out.println("Total time to find out user access : "+ elapsedTime + " milliseconds");
    }

    private static String getUserSecurityAccess(){
        Long userAccess = getUserAccess();
        if(userAccess.compareTo(new Long(3))==0){
            return "Full Access";
        }
        if(userAccess.compareTo(new Long(2))==0){
            return "READ/WRITE ACCESS";
        }
    }

}
if(userAccess.compareTo(new Long(1))==0) {
    return "READ ACCESS";
}
if(userAccess.compareTo(new Long(0))==0){
    return "NO ACCESS";
}
return "Undefined Access";
}

private static Long getUserAccess(){
    // Defining Constants

    Long FULL_ACCESS = new Long(3);
    Long READ_WRITE_ACCESS = new Long(2);
    Long READ_ACCESS = new Long(1);
    Long NO_ACCESS = new Long(0);
    Long finalAccess = null;

    //JDBC Database configuration

    String driver = "oracle.jdbc.driver.OracleDriver";
    String schemaName = "sawasthi";

    Driver d = new Driver(driver);
    DMConnection conn = new DMConnection("jdbc:oracle:thin:@localhost:1521/WYLOCALDB", "sawasthi", "sawasthi");
    try {
        List<GroupUserXREF> userGroups = new ArrayList<GroupUserXREF>();
        GroupUserDao gu = new GroupUserDaoImpl();
        userGroups = gu.getUserGroups();
        if(userGroups == null || userGroups.isEmpty()){
            UserDao u = new UserDaoImp();
            return u.getUsersAccess();
        } else if(userGroups.size() == 1){
            return userGroups.get(0).getAccessId();
        }
Iterator userGroupsItr = userGroups.iterator();
while (userGroupsItr.hasNext()){
    GroupUserXREF groupUser = (GroupUserXREF) userGroupsItr.next();
    Long access = groupUser.getAccessId();
    if (access.compareTo(FULL_ACCESS) == 0 ){
        return FULL_ACCESS;
    }
    if (access.compareTo(READ_WRITE_ACCESS) == 0 ){
        finalAccess = READ_WRITE_ACCESS;
    }
    if (access.compareTo(READ_ACCESS) == 0 &&
        finalAccess.compareTo(READ_WRITE_ACCESS) != 0){
        finalAccess = READ_ACCESS;
    }
    if (access.compareTo(NO_ACCESS) == 0 &&
        finalAccess.compareTo(READ_WRITE_ACCESS) != 0 &&
        finalAccess.compareTo(READ_ACCESS) != 0){
        finalAccess = NO_ACCESS;
    }
}
return finalAccess;
}

} catch (Exception e) {
    e.printStackTrace();
}
return null;
}
import JDBCInfrastructure.Driver;
import JDBCInfrastructure.Query;

// This Class created to calculate total time if a admin will change access of
users of all group

public class Test2 {

private static Query getQuery = null;
private static String query1 =
"update GROUP_USERS_XREF set ACCESS_TYPE_ID = 2 where
EUCALYPTUS_GROUP_ID = 10 and EUCALYPTUS_USER_ID = 100 ";
private static String query2 =
"update GROUP_USERS_XREF set ACCESS_TYPE_ID = 1 where
EUCALYPTUS_GROUP_ID = 20 and EUCALYPTUS_USER_ID = 300 ";
private static String query3 =
"update GROUP_USERS_XREF set ACCESS_TYPE_ID = 3 where
EUCALYPTUS_GROUP_ID = 30 and EUCALYPTUS_USER_ID = 200 ";
private static String query4 =
"update GROUP_USERS_XREF set ACCESS_TYPE_ID = 3 where
EUCALYPTUS_GROUP_ID = 40 and EUCALYPTUS_USER_ID = 201 ";
private static String query5 =
"update GROUP_USERS_XREF set ACCESS_TYPE_ID = 1 where
EUCALYPTUS_GROUP_ID = 50 and EUCALYPTUS_USER_ID = 202 ";
private static String query6 =
"update GROUP_USERS_XREF set ACCESS_TYPE_ID = 0 where
EUCALYPTUS_GROUP_ID = 60 and EUCALYPTUS_USER_ID = 203 ";
private static String query7 =
"update GROUP_USERS_XREF set ACCESS_TYPE_ID = 0 where
EUCALYPTUS_GROUP_ID = 70 and EUCALYPTUS_USER_ID = 204 ";
private static String query8 =
"update GROUP_USERS_XREF set ACCESS_TYPE_ID = 0 where
EUCALYPTUS_GROUP_ID = 80 and EUCALYPTUS_USER_ID = 205 ";
private static String query9 =
"update GROUP_USERS_XREF set ACCESS_TYPE_ID = 1 where
EUCALYPTUS_GROUP_ID = 90 and EUCALYPTUS_USER_ID = 206 ";
private static String query10 =
"update GROUP_USERS_XREF set ACCESS_TYPE_ID = 1 where
EUCALYPTUS_GROUP_ID = 100 and EUCALYPTUS_USER_ID = 207 ";
public static void main(String[] args) {
    long startTime = System.currentTimeMillis();

    // JDBC Database configuration
    String driver = "oracle.jdbc.driver.OracleDriver";
    String schemaName = "sawasthi";

    Driver d = new Driver(driver);
    DMConnection conn = new DMConnection("jdbc:oracle:thin:@localhost:1521/WYLOCALDB", "sawasthi", "sawasthi");
    Query q = new Query(conn);
    q.setQueryString(query1);
    q.execute();

    q.setQueryString(query2);
    q.execute();

    q.setQueryString(query3);
    q.execute();

    q.setQueryString(query4);
    q.execute();

    q.setQueryString(query5);
    q.execute();

    q.setQueryString(query6);
    q.execute();

    q.setQueryString(query7);
    q.execute();

    q.setQueryString(query8);
    q.execute();

    q.setQueryString(query9);
    q.execute();
}
getQuery.setQueryString(query10);
getQuery.execute();

System.out.println("Updated access of 10 users of 10 different groups.");

long stopTime = System.currentTimeMillis();
long elapsedTime = stopTime - startTime;
System.out.println("Total time to Updated access of 10 users of 10 different groups : "+elapsedTime + " milliseconds");
}
}

// Test 3.java

package Execution;

import JDBCInfrastructure.DMConnection;
import JDBCInfrastructure.Driver;
import JDBCInfrastructure.Query;

// This Class created to calculate total time if a group admin will change access of
* its own user security

public class Test3 {

private static Query getQuery = null;
private static String query1 =
"update GROUP_USERS_XREF set ACCESS_TYPE_ID = 2 where
EUCALYPTUS_GROUP_ID = 10 and EUCALYPTUS_USER_ID = 100 ";

public static void main(String arg[]) {
long startTime = System.currentTimeMillis();
/*JDBC Database configuration*/
String driver = "oracle.jdbc.driver.OracleDriver";
String schemaName = "sawasthi";

Driver d = new Driver(driver);
DMConnection conn = new DMConnection(
"jdbc:oracle:thin:@localhost:1521/WYLOCALDB",
"sawasthi", "sawasthi");
getQuery = new Query(conn);
getQuery.setQueryString(query1);
getQuery.execute();
System.out.println("Updated access of 1 user by Group Admin.");

long stopTime = System.currentTimeMillis();
long elapsedTime = stopTime - startTime;
System.out.println("Total time to Updated access of 1 user by Group Admin : "+elapsedTime + " milliseconds");
APPENDIX E
SQL Tables

Eucalyptus User

This table is populated with information of 14 users

```sql
INSERT INTO EUCALYPTUS_USERS(EUCALYPTUS_USER_ID, USER_LGN_NM, USER_PSWD, USER_NM_FST, USER_NM_MID, USER_NM_LST, USER_EMAIL, ACCESS_TYPE_ID, USER_VOID_IND) VALUES(100, 'jsnitker', 'X03MO1qnZdYdgyfeuLPmQ==', 'Jimmy', null, 'Snitker', 'jim.snitker@test.com', 0, 'n');
INSERT INTO EUCALYPTUS_USERS(EUCALYPTUS_USER_ID, USER_LGN_NM, USER_PSWD, USER_NM_FST, USER_NM_MID, USER_NM_LST, USER_EMAIL, ACCESS_TYPE_ID, USER_VOID_IND) VALUES(200, 'pkosch', 'X03MO1qnZdYdgyfeuLPmQ==', 'Pete', null, 'Koschorke', 'peter.koschorke@test.com', 0, 'n');
INSERT INTO EUCALYPTUS_USERS(EUCALYPTUS_USER_ID, USER_LGN_NM, USER_PSWD, USER_NM_FST, USER_NM_MID, USER_NM_LST, USER_EMAIL, ACCESS_TYPE_ID, USER_VOID_IND) VALUES(300, 'olin', 'X03MO1qnZdYdgyfeuLPmQ==', 'Owen', null, 'Lin', 'owen.lin@test.com', 0, 'n');
INSERT INTO EUCALYPTUS_USERS(EUCALYPTUS_USER_ID, USER_LGN_NM, USER_PSWD, USER_NM_FST, USER_NM_MID, USER_NM_LST, USER_EMAIL, ACCESS_TYPE_ID, USER_VOID_IND) VALUES(400, 'ptyagi', 'X03MO1qnZdYdgyfeuLPmQ==', 'Praveen', null, 'Tyagi', 'praveen.tyagi@test.com', 0, 'n');
INSERT INTO EUCALYPTUS_USERS(EUCALYPTUS_USER_ID, USER_LGN_NM, USER_PSWD, USER_NM_FST, USER_NM_MID, USER_NM_LST, USER_EMAIL, ACCESS_TYPE_ID, USER_VOID_IND) VALUES(201, 'alogin', 'X03MO1qnZdYdgyfeuLPmQ==', 'A', null, 'A', 'A.A@test.com', 0, 'n');
INSERT INTO EUCALYPTUS_USERS(EUCALYPTUS_USER_ID, USER_LGN_NM, USER_PSWD, USER_NM_FST, USER_NM_MID, USER_NM_LST, USER_EMAIL, ACCESS_TYPE_ID, USER_VOID_IND) VALUES(202, 'blogin', 'X03MO1qnZdYdgyfeuLPmQ==', 'B', null, 'B', 'B.B@test.com', 0, 'n');
INSERT INTO EUCALYPTUS_USERS(EUCALYPTUS_USER_ID, USER_LGN_NM, USER_PSWD, USER_NM_FST, USER_NM_MID, USER_NM_LST, USER_EMAIL, ACCESS_TYPE_ID, USER_VOID_IND) VALUES(203, 'clogin', 'X03MO1qnZdYdgyfeuLPmQ==', 'C', null, 'C', 'C.C@test.com', 0, 'n');
INSERT INTO EUCALYPTUS_USERS(EUCALYPTUS_USER_ID, USER_LGN_NM, USER_PSWD, USER_NM_FST, USER_NM_MID, USER_NM_LST, USER_EMAIL, ACCESS_TYPE_ID, USER_VOID_IND) VALUES(204, 'dlogin', 'X03MO1qnZdYdgyfeuLPmQ==', 'D', null, 'D', 'D.D@test.com', 0, 'n');
INSERT INTO EUCALYPTUS_USERS(EUCALYPTUS_USER_ID, USER_LGN_NM, USER_PSWD, USER_NM_FST, USER_NM_MID, USER_NM_LST, USER_EMAIL, ACCESS_TYPE_ID, USER_VOID_IND) VALUES(205, 'elogin', 'X03MO1qnZdYdgyfeuLPmQ==', 'E', null, 'E', 'E.E@test.com', 0, 'n');
```
INSERT INTO EUCALYPTUS_USERS(EUCALYPTUS_USER_ID, USER_LGN_NM, USER_PSWD, USER_NM_FST, USER_NM_MID, USER_NM_LST, USER_EMAIL, ACCESS_TYPE_ID, USER_VOID_IND) VALUES(206, 'flogin', 'X03MO1qnZdYdgyfeulPmQ==', 'F', null, 'F', 'F.F@test.com', 0, 'n');
INSERT INTO EUCALYPTUS_USERS(EUCALYPTUS_USER_ID, USER_LGN_NM, USER_PSWD, USER_NM_FST, USER_NM_MID, USER_NM_LST, USER_EMAIL, ACCESS_TYPE_ID, USER_VOID_IND) VALUES(207, 'glogin', 'X03MO1qnZdYdgyfeulPmQ==', 'G', null, 'G', 'G.G@test.com', 0, 'n');
INSERT INTO EUCALYPTUS_USERS(EUCALYPTUS_USER_ID, USER_LGN_NM, USER_PSWD, USER_NM_FST, USER_NM_MID, USER_NM_LST, USER_EMAIL, ACCESS_TYPE_ID, USER_VOID_IND) VALUES(208, 'hlogin', 'X03MO1qnZdYdgyfeulPmQ==', 'H', null, 'H', 'H.H@test.com', 0, 'n');
INSERT INTO EUCALYPTUS_USERS(EUCALYPTUS_USER_ID, USER_LGN_NM, USER_PSWD, USER_NM_FST, USER_NM_MID, USER_NM_LST, USER_EMAIL, ACCESS_TYPE_ID, USER_VOID_IND) VALUES(209, 'ilogin', 'X03MO1qnZdYdgyfeulPmQ==', 'K', null, 'K', 'K.K@test.com', 0, 'n');
INSERT INTO EUCALYPTUS_USERS(EUCALYPTUS_USER_ID, USER_LGN_NM, USER_PSWD, USER_NM_FST, USER_NM_MID, USER_NM_LST, USER_EMAIL, ACCESS_TYPE_ID, USER_VOID_IND) VALUES(210, 'jlogin', 'X03MO1qnZdYdgyfeulPmQ==', 'L', null, 'L', 'L.L@test.com', 0, 'n');

**Eucalyptus Group**:

This table contains information of 10 Eucalyptus Group.

INSERT INTO EUCALYPTUS_GROUP(EUCALYPTUS_GROUP_ID, GROUP_NM, GROUP_DESC, GROUP_ADMIN_ID, GROUP_VOID_IND) VALUES(10, 'Student', 'This group is created for students', 100, 'n');
INSERT INTO EUCALYPTUS_GROUP(EUCALYPTUS_GROUP_ID, GROUP_NM, GROUP_DESC, GROUP_ADMIN_ID, GROUP_VOID_IND) VALUES(20, 'Teacher', 'This group is created for teachers', 400, 'n');
INSERT INTO EUCALYPTUS_GROUP(EUCALYPTUS_GROUP_ID, GROUP_NM, GROUP_DESC, GROUP_ADMIN_ID, GROUP_VOID_IND) VALUES(30, 'Administration', 'This group is created for admin staff', 200, 'n');
INSERT INTO EUCALYPTUS_GROUP(EUCALYPTUS_GROUP_ID, GROUP_NM, GROUP_DESC, GROUP_ADMIN_ID, GROUP_VOID_IND) VALUES(40, 'A', 'This group is created for A staff', 201, 'n');
INSERT INTO EUCALYPTUS_GROUP(EUCALYPTUS_GROUP_ID, GROUP_NM, GROUP_DESC, GROUP_ADMIN_ID, GROUP_VOID_IND) VALUES(50, 'B', 'This group is created for B staff', 202, 'n');
INSERT INTO EUCALYPTUS_GROUP(EUCALYPTUS_GROUP_ID, GROUP_NM, GROUP_DESC, GROUP_ADMIN_ID, GROUP_VOID_IND) VALUES(60, 'C', 'This group is created for C staff', 203, 'n');
INSERT INTO EUCALYPTUS_GROUP(EUCALYPTUS_GROUP_ID, GROUP_NM, GROUP_DESC, GROUP_ADMIN_ID, GROUP_VOID_IND) VALUES(70, 'D', 'This group is created for D staff', 204, 'n');
**INSERT INTO EUCALYPTUS_GROUP(EUCALYPTUS_GROUP_ID, GROUP_NM, GROUP_DESC, GROUP_ADMIN_ID, GROUP_VOID_IND) VALUES(80, 'E', 'This group is created for E staff', 205, 'n');**

**INSERT INTO EUCALYPTUS_GROUP(EUCALYPTUS_GROUP_ID, GROUP_NM, GROUP_DESC, GROUP_ADMIN_ID, GROUP_VOID_IND) VALUES(90, 'F', 'This group is created for F staff', 206, 'n');**

**INSERT INTO EUCALYPTUS_GROUP(EUCALYPTUS_GROUP_ID, GROUP_NM, GROUP_DESC, GROUP_ADMIN_ID, GROUP_VOID_IND) VALUES(100, 'G', 'This group is created for G staff', 207, 'n');**

**Group User Cross Reference Table:**

Records in this table are

**INSERT INTO GROUP_USERS_XREF(GUX_ID, EUCALYPTUS_GROUP_ID, EUCALYPTUS_USER_ID, ACCESS_TYPE_ID, GUX_VOID_IND) VALUES(1, 10, 100, 1, 'n');**

**INSERT INTO GROUP_USERS_XREF(GUX_ID, EUCALYPTUS_GROUP_ID, EUCALYPTUS_USER_ID, ACCESS_TYPE_ID, GUX_VOID_IND) VALUES(2, 20, 300, 2, 'n');**

**INSERT INTO GROUP_USERS_XREF(GUX_ID, EUCALYPTUS_GROUP_ID, EUCALYPTUS_USER_ID, ACCESS_TYPE_ID, GUX_VOID_IND) VALUES(3, 20, 400, 2, 'n');**

**INSERT INTO GROUP_USERS_XREF(GUX_ID, EUCALYPTUS_GROUP_ID, EUCALYPTUS_USER_ID, ACCESS_TYPE_ID, GUX_VOID_IND) VALUES(4, 30, 200, 3, 'n');**

**INSERT INTO GROUP_USERS_XREF(GUX_ID, EUCALYPTUS_GROUP_ID, EUCALYPTUS_USER_ID, ACCESS_TYPE_ID, GUX_VOID_IND) VALUES(5, 30, 300, 3, 'n');**

**INSERT INTO GROUP_USERS_XREF(GUX_ID, EUCALYPTUS_GROUP_ID, EUCALYPTUS_USER_ID, ACCESS_TYPE_ID, GUX_VOID_IND) VALUES(6, 40, 201, 2, 'n');**

**INSERT INTO GROUP_USERS_XREF(GUX_ID, EUCALYPTUS_GROUP_ID, EUCALYPTUS_USER_ID, ACCESS_TYPE_ID, GUX_VOID_IND) VALUES(7, 50, 202, 3, 'n');**

**INSERT INTO GROUP_USERS_XREF(GUX_ID, EUCALYPTUS_GROUP_ID, EUCALYPTUS_USER_ID, ACCESS_TYPE_ID, GUX_VOID_IND) VALUES(8, 60, 203, 1, 'n');**

**INSERT INTO GROUP_USERS_XREF(GUX_ID, EUCALYPTUS_GROUP_ID, EUCALYPTUS_USER_ID, ACCESS_TYPE_ID, GUX_VOID_IND) VALUES(9, 70, 204, 0, 'n');**