2013 International Symposium on Biometrics and Security Technologies

Design Role-Based Multi-Tenancy Access Control Scheme for Cloud Services

Shin-Jer Yang, Pei-Ci Lai
Dept. of Computer Science & Information Management
Soochow University
Taipei, Taiwan
sjyang@csim.scu.edu.tw; 99356004@scu.edu.tw

Jyhjong Lin
Dept. of Information Management
Ming Chuan University
Taipei, Taiwan
jlin@mail.mcu.edu.tw

Abstract—**Cloud Computing is the next generation Internet service and data center, and it is also used for public utilities and on-demand computing. Cloud computing is not a totally new technology, but rather a derived concept of application and service innovation in which, multi-tenancy is one of the important issues among the core technologies of cloud computing applications. Many tenants can access the different applications and computing resources in the same cloud server, whereas concurrent use by many users on a database or application will lead to large data volume, time consuming and security issues. Under these circumstances, it is particularly important to separate application and data for conflicts avoidance to enhance the system and data security. This paper emphasizes the cloud service model under a Multi-Tenant Architecture (MTA), using identity management and Role-Based Access Control, to propose and design a Role-Based Multi-Tenancy Access Control (RB-MTAC). The RB-MTAC applies identity management to determine the user's identity and applicable roles, since different users possess different functional roles with respective privileges for processing. Such role-based assignments can easily and efficiently manage a user's access rights to achieve application independence and data isolation for improving the processing performance of cloud multi-tenant services and hardening the security and privacy of cloud applications.**

Keywords-Cloud Computing; Multi-Tenancy; RB-MTAC; MTA; RBAC.

I. INTRODUCTION

People used to draw a cloud to represent the Internet, and that is the cloud we know today. There are an increasing number of people setting up their data or websites in the cloud, even using software based on cloud services (i.e. Salesforce.com) to help with corporate operations. There are also many organizations and corporations that have started to develop cloud computing in Taiwan. The Multi-Tenancy Control is a cloud platform technology, which covers SaaS, PaaS and IaaS. The multi-tenancy access control to the cloud platform is considerably important. When a massive number of tenants share the same hardware/software resources, each tenant undergoes customized configuration according to the resources needed without affecting the usage of other users. Research shows that the top concerns among corporations with regards to SaaS service are in security and privacy issues and service quality with 80% and 70% of the respondents respectively. Such data clearly indicate that users are concerned about the cloud security and privacy, therefore application independence and data isolation becomes an issue that can not be neglected under the Multi-Tenant Architecture (MTA) today. Also,, the multi-tenancy system is considerably more important in the SaaS cloud service under MTA..

This paper emphasizes on the use of identity management and Role-Based Access Control (RBAC) under the multi-tenant cloud environment, so that each user is assigned one to many roles while each role is assigned to different privileges. Such privilege assignment can easily and effectively manage the access rights of users, thereby maintaining application independence and data isolation or protection while improving process performance and cloud security. This paper takes into consideration the different perspectives of roles, since different users can be assigned to different role sets, which corresponds to different privileges. Moreover, the different privileges allow access to different Web sites and different data in order to improve security and efficiency. Using the characteristics of different user authority could produce environmental isolation and data isolation. As distinguished from the Discretionary Access Control and Mandatory Access Control, the RBAC offers more flexibility in the granting of privileges, making it easier to manage. Users are assigned to a role and granted privileges that correspond to that role directly. Using the role-based access control simplifies the management of privileges, since the addition, deletion, query, and modification of privileges are applied to the "role" instead of the individual user.

Based on the above reasons, this paper proposes Role-Based Multi-Tenancy Access Control called RB-MTAC that integrates a set of identity management and RBAC with consideration of multi-tenant and multi-user cloud environment. The RB-MTAC method can easily assign the functions or resource with access privilege to users, in order to enhance processing performance, quality of service (QoS), and security as well as privacy on the cloud. Hence, the purposes of RB-MTAC scheme in cloud computing are follows.

(1) RB-MTAC combines the identity management and role-based access control of a multi-tenancy environment in cloud computing, which is effective and simple to manage privileges that protect the security of application systems and data privacy.

978-0-7695-5010-7/13 $26.00 © 2013 IEEE
DOI 10.1109/ISBAST.2013.48

(2) The method provides good identity management and access control to privileges to block a non-tenant user accessing through identity management, while access control prevent tenant users without specific privilege from viewing and accessing specific applications and database.

(3) The method also offers a mechanism that manages user privileges using role-based viewpoints, so administrators only need to modify role privileges to easily change user privileges, so that would reduce potential errors from constant modifications.

(4) Based on the RB-MTAC method, building a prototype system and simulation experiment, which supports this is better than user-based system.

(5) The application independence and data isolation of different tenants. Under the cloud environment, the systems and data of different tenants could be stored in the same place and that can prevent other tenants using the data and system of that tenant either intentionally or unintentionally.

The rest of this paper is organized as follows. Section II surveys and discusses the relevant literature required for this paper. Section III comprises the RB-MTAC and designs its pseudo-code. Section IV lists the simulation environments and examines the analysis of results. Section V makes the conclusion.

II. Related Works

In this Section, we introduce related research on cloud service models, RBAC and multi-tenant control in addition to survey related works.

A. Cloud Services Model

The National Institute of Standards and Technology (NIST) defines cloud computing as the composition of five essential characteristics: three service models and four deployment models [5]. The three service models are briefly introduced below:

(1) Software as a Service (SaaS): The capability provided to the consumer is to use the provider's applications running on a cloud infrastructure. The applications are accessible from various client devices through either a thin client interface or a program interface. The consumer does not manage or control the underlying cloud infrastructure such as the Internet and operating system.

(2) Platform as a Service (PaaS): The capability provided to the consumer is to deploy onto a cloud infrastructure the consumer-created or acquired applications. The consumer does not need to manage or control the cloud infrastructure, but has to control the deployed applications and configuration settings for the application-hosting environment.

(3) Infrastructure as a Service (IaaS): The capability provided to the consumer is to provision processing, storage, networks, and other fundamental computing resources where the consumer is able to deploy and run arbitrary software, which can include the operating system and application. The consumer does not manage or control the underlying cloud infrastructure but has to control the operating system, storage, and to deploy applications, and possibly limited control of selected network components.

Cloud services reduce enterprises' IT costs since they only need to pay for services on demand [4]. From society's perspective, cloud computing can optimize the utility of IT resources and provide better and more comprehensive services. The SaaS is closed to end users and cloud computing provides software application services to hundreds or thousands of customers through the Internet. From the customer's perspective, this offering saves having to invest on service hardware and software. For providers, this approach only requires the maintenance of one application project on the server, which substantially reduces the operating costs. SaaS usually adopts the layered development mechanism. For example, the Java platform development could have adopted the MVC pattern; however, the model is usually the same while the controller is also identical for the different users. The main difference for customers lies in the view, because different users may need different viewing content and format, such as different forms, reports, interactive interface and process, terminology, syntax, and help. In most circumstances, the different demand of users only needs an adjustment of the view settings or programs without adjusting other logic.

B. Role-Based Access Control

RBAC has become an accepted standard for access control; one of the main advantages of RBAC is that it can be implemented in a natural way with higher level organizational rules [1, 8]. Sandhu proposed the concept of RBAC in 1996 where system administrators of an RBAC system formulate roles in accordance with existing job functions of the company or organization [4, 7]. Those roles are given users and privileges are assigned with the different roles according to their specific work functions and qualifications. The majority of access control systems frequently provide users with Access Control List (ACL). The main difference between Group and Role is that a group often represents a set of users rather than a set of privileges. The role is treated as the median between the users and authorization sets. The RBAC basic chart is shown in Figure 1 [7] with the following definitions:

- U, R, P, S (users, roles, permissions, and sessions, respectively);
- $PA \subseteq P \times R$, a many-to-many permission-to-role assignment relation;
- $UA \subseteq U \times R$, a many-to-many user-to-role assignment relation;
- $user: S \rightarrow U$, a function mapping each session s_i to the single user $user(s_i)$ (constant for session's lifetime); and
- $roles : S \rightarrow 2^R$, a function mapping each session s_i to a set of roles $role(s_i) \subseteq \{r \mid (user(s_i), r) \in UA\}$ and session s_i has the permissions $\cup_{r \in roles(si)} \{p \mid (p, r) \in PA\}$

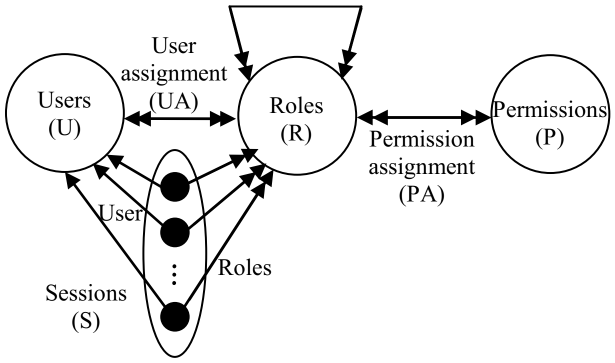

Figure 1. RBAC Basic Models.

C. *Multi-Tenancy Control*

In the architecture of multi-tenant cloud environment, tenants often refer to the customers using the system or computing resources [11]. But in a multi-tenant control or system, tenants are included in all system data that could be identified for assigned users, such as the account number and statistical data; users can build the various data in the system, and the customized applications of users all belong to the scope of the tenant. The tenants use the application system or computing resources developed or built by the cloud suppliers, where the application system is designed with the capacity of multiple tenants under the same environment. To provide multiple tenants with the ability to use the same application on the same cloud computing environment, the application and computing environments should be designed carefully. In addition to allowing multiple identical applications to concurrently execute on the cloud platform, the protection of the tenant's data security and privacy is also one of the key multi-tenancy technologies.

Basically, the key technology of a multi-tenancy system implementation lies on Application Context Independence and Data Isolation of different tenants in order to maintain the different applications between tenants without mutual interference, while maintaining adequate data confidentiality.

1) Applications: The procedures or carriers environment for supporting the concurrent execution of multiple applications or using the way of threads in the same server program are used for program isolation.

2) Data: The different mechanisms are used to isolate the different tenant data. For example, the Force.com adopts metadata technique for data isolation and Microsoft NSDN isolates the technical files via using the structured definitions.

Data isolation is an important feature in multi-tenancy technology, which refers to the mutual isolation and storage of a tenant's business data without mutual interference when multiple tenants are using the same application system. In addition, multi-tenancy technology requires secure and highly-efficient data isolation to ensure a tenant's data security and the overall efficacy of multi-tenancy platforms. There are three basic methods to multi-tenancy database management:

(1) Each tenant builds an independent database with the advantage of full isolation for user data and the disadvantage of high cost and consumption for database management.

(2) Multiple tenants' data are stored in one database with the adoption of different schema, which reduces managerial costs and consumption of the database to certain extent; however, it also relatively weakens the effect of data isolation.

(3) Storing multiple tenants' data in one database with the adoption of identical schema. In other words, the data is preserved in one cell or one cell with the same schema and differentiated through the label code of the tenant. In spite of such low managerial costs and consumption associated, the effect of data isolation also proves to be the worst, as it requires an incredibly secure indexing function to protect the data isolation between tenants.

III. Design Issues of RB-MTAC Scheme

In this Section, an ideal RB-MTAC method, which supports identity management and access control privilege is proposed with the design of RB-MTAC prototype scheme. Hence, this study offers identity management and role-based access control under the MTA environment, which combines the concept of multiple tenants and the characteristics of role-based access control, collectively known as the RB-MTAC as depicted in Figure 2. In Figure 2, every tenant or user needs to log in with an account number and password while the system authenticates the user account through identity management. If the users are valid, the role assignment will capture the role corresponding to the user from the RB-MTAC database and assign the role and access rights, which belong to the user. The user can carry out activities, access functions and computing resources in the system through RB-MTAC.

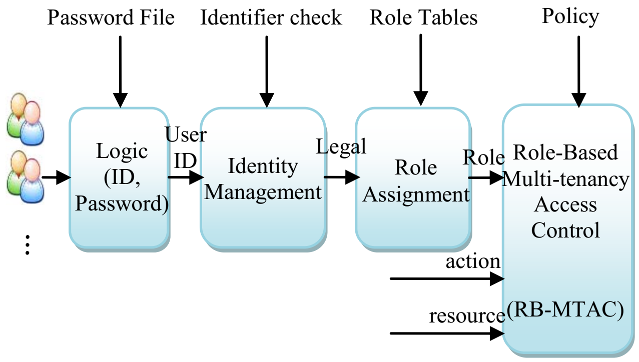

Figure 2. Operating Schematic of RB-MTAC.

According to the operating procedures as depicted in Figure 2, we design RB-MTAC with a combination of identity management, multi-tenancy and role-based access control methods as shown in Figure 3. The RB-MTAC can execute the identity management first when users log in to the application or system, then the identity is confirmed with the account number, password and other relevant authentication credentials. The user account will then enter the Role Tables to capture the role sets of the user and integrate the functional privilege within role sets through function-based access control, generating the Access Control List (ACL) for users in addition to providing encryption and privacy security services. This paper explains the first four steps, namely the identity management, RBAC, Function-based access control, and the generation of ACLs.

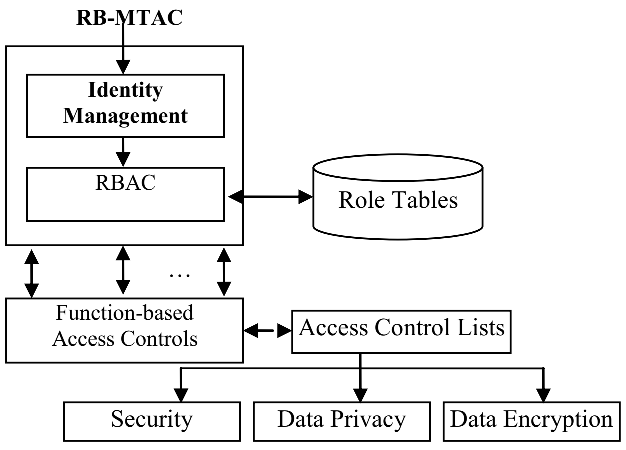

Figure 3. RB-MTAC Conceptual System Structure.

A database of a multi-tenancy system should have the following tables as shown in Figure 4, which refers to the associated module of databases with the following descriptions:

1) Tenant: Storing tenant information: Tenant name, password and any necessary administrative data.

2) User: Basic information of users: User name, password and tenant to which they belong.

3) Application: It contains the necessary data from an application, like name and description, the path to the application and its icon, for example.

4) Extension: Description of the customized tenant column with name and data pattern (integers, strings, and dates).

5) Data: Content (values) of the virtual table made up of users and extensions of a tenant.

Figure 4 refers to the model design of the multi-tenancy database. It is a chart depicting the relationship corresponding to the tenant, user and applications. Through the scalable database module, it can achieve customization and increase the flexibility of usage as well as application independence and data isolation.

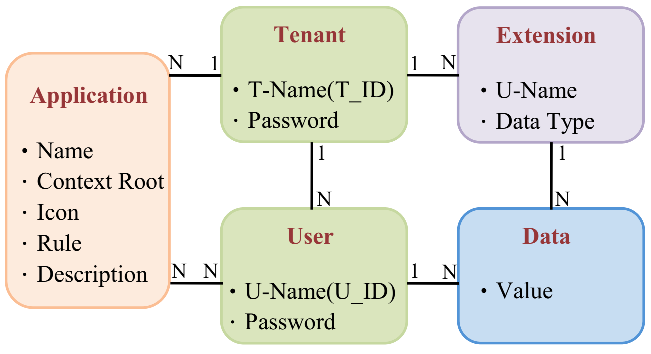

Figure 4. Database Module with Extension.

This study utilizes the role perspective because users and roles are a multiple-to-multiple relationship. For example, the departmental heads of enterprises can concurrently possess the employee and supervisor identities. According to corresponding functions and rules of roles, each user owns different privileges. The illustrated examples are shown in Figures 5 and 6. Figure 5 shows the users having different role identities with Role A or B owning the respective function and the corresponding rules: Role A only owns the privilege to query system X and query system *m* while Role B owns the privilege to query and modify Document Y and to modify Database Z. Role *m* can add or modify System X, add or delete Document Y and delete data from Database Z. According to the content in Figure 5, due to the inheritance of role privileges, we can generate the ACL for each user, as shown in Figure 6, whereby Q denotes Query, U denotes Update, I denotes Inserts, D denotes Delete. Although User Number 1 owns the identity of Role A, which can only query System X, it also has the role of *m*, which can add or modify System X. Each user has its own ACL and when the user logs in, the system will determine the access privileges according to the ACL of the user.

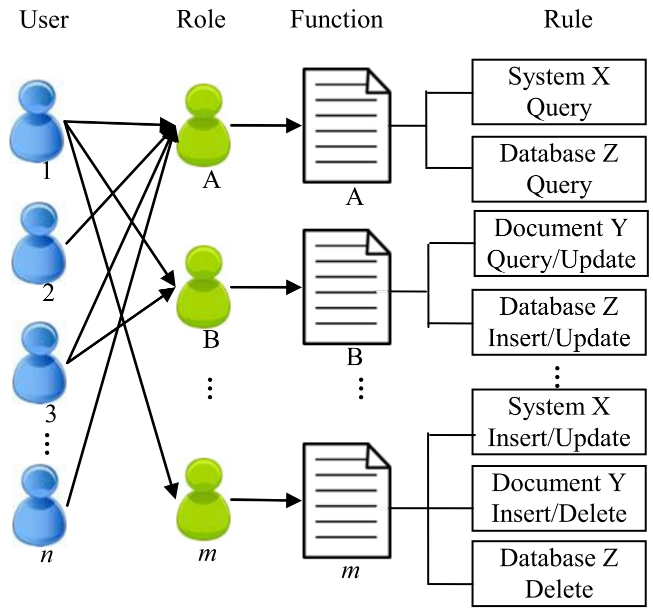

Figure 5. User and Role Rules Correspondence Instances.

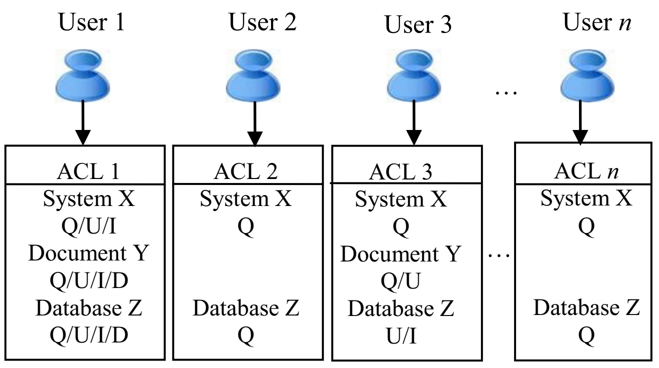

Figure 6. Instances of ACLs.

This paper proposes the RB-MTAC mainly to integrate the identity management and role-based access control under the multi-tenant cloud environment. In RB-MTAC, cloud tenants may achieve identity management and conform to the privilege attributed to the roles in order to determine whether or not the various application functions and system data can be displayed or modified. Basically, the preliminary plan for the operating procedures of RB-MTAC is described below.

(1) When logging in, the users must be authenticated as a user of the tenancy.

(2) If the user is a valid tenant member and both the account number and password are correct, the user role set will be captured.

(3) An account lockout setting of three attempts is proposed to prevent unauthorized personnel from gaining access to the system.

(4) Generating ACL through the users' role set.

(5) After entering the system's application home page, the user's ACL will be applied to determine whether the application's function and content should be displayed or hidden, and then present the results to the application.

Based on above RB-MTAC operating procedures, the preliminary pseudo code of the RB-MTAC algorithm can be designed as follows.

```
Algorithm RB-MTAC(){
Input:
 Create a database DB
 Create a table Tenant (Tenant_ID, Tenant_Name) on data DB
 Create a table User (User_ID, User_Password)
 Create a table Role (Role_ID, Role_Name) on data DB
 Create a table Permission (Permission_ID, Permission_Name) on data DB
 Create a table Tenant_Assignment (User_ID, Tenant_ID) on data DB
 Create a table Permission_Assignment (Role_ID, Permission_ID, Attribute) on data DB
 Create a table User_Assignment (User_ID, Role_ID) on data DB
 Create another tables that user will use for extension
 Dim u_id as user input
 Dim u_psw as user input
Output: To implement RB-MTAC on the Cloud Computing using Identity Management and Role-Based Multi-tenancy Access Control
Method:
 BEGIN{
   Display 'login' Web page
   User input u_id in filed
   User input u_psw in filed

   Set 0 to N
   Set '0' to Flag
   Set Array role[]
   Set Array ACL[]
   Do{
    Do{
      Query table Tenant_Assignment field User_ID, Tenant_ID
      If 'u_id' equal to table Tenant_Assignment field User_ID
       If Tenant_ID equals this system's Tenant_ID{
        Flag == "1"
        Break;
       }
     }while(not end of table)
     Do{
      Query table User field User_ID, User_Password
      If 'u_id' equal to table User field User_ID and 'u_psw' equal to table User field User_ Password then {
        Query table User_Assignment field Role_ID into role[]
        Access(role[], User_ID)
       }
     }while(not end of table and Flag equals "1")

     If Flag equals '0'
      N = N + 1
     Else
      Break
   }while (N < 3)

   If N = 3
     Show error message
 }END

Access(role[], User_ID) {
 Do{
   Query table Permission_Assignment field Permission_ID, Attribute, value via role[]
   Set into ACL[] field Permission_ID, Attribute
 }while(not end of array role[])
 Display Portal and execute permissions
 Operation(User_ID, ACL[])
}

Operation (User_ID, ACL){
 If user want to insert database DB{
   If user's Attribute equals database DB
     If user's value equals "insert"
       Do(insert)
 }
 If user want to update database DB{
   If user's Attribute equals database DB
     If user's value equals "update"
       Do(update)
 }
 If user want to delete database DB{
   If user's Attribute equals database DB
     If user's value equals "delete"
       Do(delete)
 }
 If user want to query database DB{
   If user's Attribute equals database DB
     If user's value equals "query"
       Do(query)
```

```
  }
}

Do(event) {
  If event equals “insert”{
    Input all filed values
      Insert filed values to database DB
  }
  If event equals “update”{
    Input all filed values
      Update filed values to database DB
  }
  If event equals “delete”
    Delete filed values from database DB
  If event equals “query”{
    Input all filed values
      Query table field from database DB
  }
}
```

END RB-MTAC.

IV. Simulations Setup and Results Analysis

This Section sets up simulation environments and gives related key performance indicators to carry out the simulation experiment. This paper employs a role-based access control under the multi-tenant cloud environment to be simulated using a Java program platform development from the cloud server, JDK 7.2 and Eclipse development simulation programs to configure three different simulation environments as follows:

(1) Use the RB-MTAC method and UBAC (User-Based Access Control) system to compare the possible effect on the performance and the differences between them. The RB-MTAC’s system management is by roles while UBAC is by individual users. Determine whether the use of different access will affect the system.

(2) Compare the different number of tenants and users for a system by using the RB-MTAC method. Evaluate the effect of different quantities of tenants or users on the system and determine whether the RB-MTAC method is more suitable for many users or just a few users.

(3) Compare the possible outcomes and efficiency analysis between the different tenants sharing the same database or independent database for each tenant in a system using RB-MTAC.

In this paper, the following definitions of four Key Performance Indicators (KPIs) are proposed to analyze the simulation results.

(1) Throughput: The average rate of number of transactions or records over a multi-tenant cloud application in a period of time. The throughput is usually measured in number of transactions or records per second.

(2) Response Time: The time interval that an application or functional unit takes to react to a given input.

(3) Security *Level:* The security level is based on the security risk values of the assessment results for each cloud application. Also, the security risk value is computed from formula (1).

$$\text{Risk Value} = \text{Assets Cost} * \text{Vulnerability} * \text{Threat} \quad (1)$$

(4) Replacement Cost: The cost of a computing environment would have to pay to replace upgrade to new infrastructure.

In this paper, the expected results are analyzed as follows:

(1) The RB-MTAC scheme combines the identity management and role-based access control under MTA of cloud computing, which is effective and efficient to manage accessing privilege that protects the security of applications and data privacy.

(2) To simplify the user management model, the RB-MTAC can prevent unauthorized tenants from logging into the system. The user’s role can make it easier to manage a user’s privilege and function to each tenant in the cloud. The administrator only needs to modify the privilege of the user, so it can reduce manual errors from constant modifications.

(3) The RB-MTAC can strengthen application systems and data isolation and avoid mutual conflicts in applications and data among tenants. In a multi-tenancy system on the cloud, every tenant has their own privilege, so that any tenant can prevent other tenants from using the functions and resources in cloud computing.

(4) Emphasizing the independent privileges among tenants to prevent users from abusing privileges. According to the tenant’s privileges, the RB-MTAC can restrict users from accessing their own data to achieve data isolation and encryption.

(5) The user-based systems can manage all users’ privileges in the ACL, but in the RB-MTAC, all roles’ privileges are defined by the administrator and therefore the number of roles are always smaller than the number of users. Hence, the RB-MTAC will enhance the throughput and response time, and enforce the application’s isolation and data security and privacy.

(6) Due to data isolation and application independence of RB-MTAC, the threat of cloud applications will be reduced. Also, each program of RB-MTAC will perform the vulnerability scanning to lower the vulnerability rating. Therefore, the RB-MTAC can provide better security level.

V. Conclusions

The RB-MTAC scheme proposed in this paper can separate the online tenants from cloud servers while capturing a user’s roles and privileges immediately after logging into the system. The users’ privileges are clear in the cloud application, which also helps the platform administrator to manage the user’s privileges. The main contribution of this paper is to provide a set of privileges and the identity management scheme for corporations in cloud computing environment. Such a scheme can be used to easily change employee privileges in the event of personnel changes and modify the role privileges directly when adding new functions to the system without the need to modify all employee privileges one by one. Not only does this approach facilitate customization of identity management, but it also reduces human error when a large number of employees need to have their privileges modified.

The RB-MTAC scheme will obtain better processing performance and higher security than other conventional user accessing control methods. In addition, the RB-MTAC scheme is suitable for cloud services and also for a more complicated SaaS model. With KPIs of throughput, response time and security level, we can do further simulations for analyzing the replacement costs in the future.

ACKNOWLEDGMENT

The partial work of this paper is supported by National Science Council in Taiwan under Grant NSC 101-2410-H-031-006.

REFERENCES

[1] G.-J. Ahn, "The RCL 2000 Language for Specifying Role-Based Authorization Constraints," PhD Dissertation, George Mason Univ., 1999.

[2] Michael Armbrust, Armando Fox, Rean Griffith, Anthony D.Joseph, et al, "A View of Cloud Computing," Communications of the ACM, Vol. 53, No. 4, pp. 50-58, April 2010.

[3] Chang Jie Guo; Wei Sun; Ying Huang; Zhi Hu Wang; Bo Gao, "A Framework for Native Multi-Tenancy Application Development and Management," In Proc. of the 9th IEEE International Conference on E-Commerce Technology and The 4th IEEE International Conference on Enterprise Computing, E-Commerce and E-Services (CEC-EEE), pp. 551-558, Tokyo, Japan, July 23-26, 2007.

[4] Xiao-Yong Li, Yong Shi, Yu Guo, Wei Ma, "Multi-Tenancy Based Access Control in Cloud," In Proc. of International Conference on Computational Intelligence and Software Engineering (CiSE), pp. 1-4, Wuhan, China, December 10-12, 2010.

[5] Peter Mell, Timothy Grance, "The NIST Definition of Cloud Computing," National Institute of Standards and Technology, Special Publication 800-145.

[6] George Pallis, "Cloud Computing: The New Frontier of Internet Computing," IEEE Internet Computing, Vol. 14, No. 5, pp. 70-73, September 2010.

[7] Ravi S. Sandhu, Edward J. Coyne, et al., "Role-Based Access Control Models," Computer, Vol. 29, No. 2, pp. 38-47, February 1996.

[8] Karsten Sohr, Michael Drouineaud, Gail-Joon Ahn, Senior Member IEEE, and Martin Gogolla, "Analyzing and Managing Role-Based Access Control Policies," IEEE Transactions on Knowledge and Data Engineering (TKDE), Vol. 20, No. 7, pp. 924-939, July 2008.

[9] Wei-Tek Tsai, Qihong Shao, "Role-Based Access-Control Using Reference Ontology in Clouds," In Proc. of International Symposium on Autonomous Decentralized Systems (ISADS), pp. 121-128, March 23-27, 2011, Tokyo & Hiroshima, Japan.

[10] Shin-Jer Yang, Jia-Shin Chen, Kuei-Chin Chen, "A Study of Security and Performance Issues in Designing Web-based Applications," In Proc. of IEEE International Conference on e-Business Engineering (ICEBE), pp. 81-88, October 24-26, 2007.

[11] Multitenancy, http://en.wikipedia.org/wiki/Multitenancy.