International Journal of Grid and Distributed Computing
Vol.7, No.2 (2014), pp.79-88
http://dx.doi.org/10.14257/ ijgdc.2014.7.2.08

Toward Fine-grained Data-level Access Control Model for Multi-tenant Applications

Kun Ma[1, 3], Weijuan Zhang[2] and Zijie Tang[3]

1 Shandong Provincial Key Laboratory of Network Based Intelligent Computing, University of Jinan, Jinan 250022, Shandong, China
2 CVIC SOFTWARE ENGINEERING CO.,LTD., Jinan 250014, Shandong, China
3 School of Information Science and Engineering, University of Jinan, Jinan 250022, Shandong, China
ise_mak@ujn.edu.cn, zhang_wjuan@cvicse.com, zijie@mail.ujn.edu.cn

Abstract

Cloud computing presents new security and privacy challenges to control access to multi-tenant applications in the cloud. However, this solution has more challenges once the number of access control list (ACL) increases in the cloud, such as efficiency of policy resolution, multi-tenancy and data isolation. To address these limitations, this paper describes a fine-grained data-level access control model (FDACM) suitable for multi-tenant applications where role-based and data-based access control are both supported. Lightweight expressions are proposed to present complicated policy rules in our solution. Moreover, we discuss the most important part of FDACM in detail: query privilege model and decision privilege model. Furthermore, we also propose the architecture and authorization procedure which implements these two models. Some technical implementation details together with the performance results from the prototype are provided. Finally, a case study of FDACM is illustrated to evaluate the effect of the application in practice.

Keywords: *Multi-tenancy; Access Control; Role Based Access Control; Cloud Computing*

1. Introduction

Cloud computing [1] is an emerging computing technology which makes software even more attractive as a service using on-demand infrastructures. These on-demand infrastructures may enable end users to rent business services or software without installing them at any computer. To achieve this end, multi-tenancy [2] support is defined as the capability of a single software instance to provide its service to several parties simultaneously.

Organizations use the cloud in a variety of different service and deployment models. There are a number of security concerns associated with cloud computing, but these fall into the issues by the providers and their tenants [3]. Access control plays an important part in multi-tenant applications, which is used to solve the controllability problem of the service and data. An access control list (ACL) specifies which users are granted access to objects, as well as what operations are allowed on given objects. When a subject requests an operation on an object in an ACL-based security model, the authorization system first checks the ACL for an applicable entry to decide whether the requested operation is authorized or not.

There are a number of ACL approaches but these issues fall into two broad categories: role-level and data-level access control. Best practice indicates that traditional Role-based Access Control (RBAC) and its extended models solve the role-level issues neatly. However,

ISSN: 2005-4262 IJGDC

we are facing the challenges on how to handle the control of the data-level access. First, we are determining how to describe the data-level access control rules that are complex in practice. Second, we expect to restrict the read/write permissions of users in the cloud. Finally, we expect to define the fine-grained data-level model by two dimensions: row and column.

To address this limitation, this paper describes a fine-grained data-level access control model (FDACM) suitable for multi-tenant applications. Moreover, we discuss two important parts of this model in detail: query privilege model and decision privilege model. The contributions of this paper are divided into several folds. First of all, this model provides fine-grained data-level access control to ensure security and privacy of multi-tenant applications. Second, the business is loosely decoupled from this model. We can update the access control rules on the fly without the reboot or modification of the multi-tenant applications in the cloud. Finally, our access control model is accurate to each row and column of fine-grained business data.

The remainder of the paper is organized as follows. Section 2 describes related work on the access control for multi-tenant applications in the cloud. Our fine-grained data-level access control model (FDACM) is explained in section 3. Next, section 4 describes the architecture of multi-tenant applications using our proposed model. Some performance results are shown in section 5. Finally, section 6 provides some conclusions and discusses future work.

2. Related Work

This section describes related work on some authorization models and systems related to cloud computing.

2.1. Access Control Model

Conventional access models, such as mandatory access control (MAC), discretionary access control (DAC), and role-based access control (RBAC), are not designed to enforce data-level policies. MAC [4] refers to a type of access control by which the system constrains the ability of a subject or an initiator to access or generally perform a sort of operations on an object or a target. DAC [5] is a type of access control, as a means of restricting access to objects based on the identity of subjects and/or groups to which they belong. The controls are discretionary in the sense that a subject with a certain access permission is capable of passing that permission on to any other subject. RBAC [6] model, relies on the usage of roles as a set of privileges that could be assigned to a user. It implements the MAC or DAC. However, these access control models provide inadequate support for dynamic column and row of data level, which might be considered when determining user permissions.

Another authorization model is OAuth [7], which is an open standard for authorization. OAuth provides a method for clients to access server resources on behalf of a resource owner. With the new version OAuth 2.0 [7], it focuses on client developer simplicity while providing specific authorization flows for different applications. Although OAuth 2.0 is popular, the specification does not give the unified methods to describe the data-level authorization and data-level resources.

For the data-level authorization, this paper describes a fine-grained data-level access control model (FDACM) suitable for cloud computing in which both role-based and data-based access controls are supported. The solution supports and encourages the separation of the authorization decision from the point of use. It might be more flexible than traditional static ACLs because it does not require time-consuming maintenance on

ACLs, and rules can be implemented quickly to accommodate the changing requirements. Therefore, RBAC and FDACM are the better solution to function, and data level respectively.

2.2. Data-level Access Control Approaches

Currently, there are some data-level access control approaches. The first approach is converting the data-level access control into the function-level one. We also call this method "pseudo data-level" for short. In this way, different functions are designed in case that the operational data are different. That is to say that similar function should be redesigned rather than reused. Obviously, this model decreases the reusability of the functions. In this way, some access control lists (ACL) are described for the resources. The disadvantage of this approach is that maintaining static ACLs is very time-consuming. For multi-tenant applications, the order of magnitudes of ACL is tenant number * user number per tenant * resource number per user.

The second approach is coupling the authorization checking to the business codes. We also call this method "couple to business" for short. In this way, it judges the roles of the current user to determine which business is to be executed. When authorization decisions are weaved into the business, it is very difficult to update the decision criteria once the governing policy changes. For multi-tenant applications, it will become a bottleneck.

The third approach is attribute-based access control (ABAC), which suggests that attributes and rules could either replace RBAC or make it more simple and flexible. Although it has no clear consensus model to date, the approach's central idea asserts that the access can be determined based on various attributes presented by a subject [8]. However, this approach merely supports the column-based control rather than row-based one.

The fourth solution is rule-based data-level access control approach. In this solution, data-level access control is described as the business rules. As business rules loosely couple of fixed codes, this solution is superior to the second solution. There are some open-source products to assist this approach. Spring Security [9] provides a ACL plugin to add the support of domain object security to a multi-tenant application. The core plugin and other extended plugins support restricting access to URLs via the rules that include checking a user's authentication status or its roles. It extends this by adding the support for restricting access to instances of individual domain class. The access can be very fine-grained and can define which actions can be taken on an object - these typically include Create, Read, Update, and Delete (CRUD) but you are free to define whatever actions you like. Another software is JBoss Drools [10], which is a business rule management system (BRMS) with a forward chaining inference based rules engine. The JBoss Drools supports the Java Specification Request (JSR-94) standard [11] for its business rule engine and enterprise framework for the construction, maintenance, and enforcement of business policies in an organization, application, or service. The limitation of these approaches is that the control granularity is not fine-grained.

2.3. Access Control for Multi-Tenant Applications

The remainder of this section describes the alternative access control model proposals for cloud applications. Sangroya *et al.*, [12] primarily aimed to highlight the major security issues and proposed a risk analysis approach in current cloud computing environments. This approach can be used by a prospective cloud service for analyzing

data security risks before putting the confidential data into a cloud computing environment. Moreover, Chow *et al.*, [13] characterized the problems and their impact on the adoption of cloud computing as a way of computation without outsourcing control. They claimed that traditional enterprise authentication and authorization frameworks were not naturally extended into the cloud to deal with multiple cloud resources and integrate cloud security data into their security metrics and policies. The above research illustrated the requirement for new authorization services and models suitable for the cloud computing to provide access control in a distributed and multi-tenant environment. Yu *et al.*, [14] presented a novel solution for the enforcement of access control and the management of its evolution in cloud environments. They enforced access policies based on data attributes. Almutairi *et al.*, [15] presented a distributed architecture that incorporates principles from security management and software engineering, and proposed a design model for the architecture. They mainly focus on the files in the Cloud rather than the database-level data. Calero *et al.*, [16] informally presented a multi-tenancy authorization system (MTAS) which extends the well-known role-based access control (RBAC) model by building trust relations among collaborating tenants. Subsequent studies formalize this MTAS model and propose extensions for finer-grained cross-tenant trust [17]. However, these access control models do not provide effective database-level access control for rapidly changing multi-tenant applications. Yang *et al.*, [18] designed a role-based multi-tenant access control model, applying the identity management to determine the user's identity and applicable roles. In our previous study [19], we proposed a transparent data middleware in support of multi-tenancy. Although the data in the cloud are separated logically, the thin granular access control of the row-based and column-based data is not discussed.

3. Fine-grained Data-level Access Control Model

We propose a solution to data-level access control on multi-tenant applications named fine-grained data-level access control model (FDACM). The core ideas of FDACM is identifying authentication using dynamic policies rather than static ACLs. The authentication procedure of multi-tenant applications is firstly identifying functions by RBAC, and then identifying fine-grained data by FDACM. FDACM is composed of the query privilege model and the decision privilege model. Query privilege model is used to describe the row-based and column-based query results that are visible to the users, and decision privilege model is used to describe the operational row-based and column-based data. FDACM meets the requirements of dynamic description of tenants, business data and data query in the form of the proposed expressions.

3.1. Lightweight Expression

In order to describe rules of access control, we propose a kind of lightweight expression, which is a powerful expression language that supports querying, filtering and manipulating objects at runtime. The features are driven by the requirements of describing the rules, including constants, session variables, request variables, user defined functions, and method invocation of specific codes. Generally, the operators of the rules consist of relational, logical, mathematical operators. The relational operators (=, <>, <, <=,>, >=, LIKE and IN) are supported using standard operator notation; the logical operators that are supported are and, or, and not; the addition mathematical operator (+, -, *, /, %) can be used on numbers, strings and dates. The complicated operation is composed of the compound of the operations.

3.2. Interpolation

Table 1. Expression Categories

Expression	Definition
$C{constantName}	constant
$S{parameterName}	session variable
$I{Class.method}	method invocation
$R{parameterName}	request variable

The format of interpolation is ${expression}, where *expression* can be all kind of expressions. The interpolation is used to insert the value of the expression converted to text (string). The result of the expression must be a string, number or date value. This is because only numbers and dates will be converted to string by the interpolation automatically. Table 1 shows the common interpolations.

3.3. Fine-grained Data-level Access Control Model

An access control list (ACL) is a list of permissions attached to an object. In this section, we give the key components of the fine-grained data-level access control model (FDACM). Complex applications often has the demand on defining access permissions not simply at a web request or method invocation level. Instead, security decisions are forced to comprise both who, where and what. In other words, authorization decisions also need consider the actual domain object and subject when invoking a method.

A fine-grained data-level access control model is described as "*what kind of user will be able to perform what kind of data query at what point at what kind of time"*. Therefore, it is defined as 5-tuple: {S, O, W, D, T}, where S and O are the subject and object of this policy respectively, W is the place using this policy, D means the row-based and column-based data of data query, and T indicates the time bucket. This model is time-correlated. It means that the query or decision is authorized to the user in a specific period.

S={tenant, user}$ is the subject that has data-level permission, where tenant is the unique identification of the tenant, and user indicates the affiliation of the subject.

O is a controlled domain object. The query and operational data are a set of object, whose attributed is usually restricted.

W is the place the access is controlled. Generally it is the business methods to be controlled.

D is the data-level rules. There are two categories of the data rule. The first category is the row-based data rule. A row-based data rule is recursively defined as rowRule:=(item[, rowRule], op), where item is denoted as (column, op, value), where column and value indicates the attribute name and value of an object, and op is the operator. rowRule is the optional item, and op is the operator. column indicates the attribute of an object. The instances of operators are the relational and mathematical introduced in Section 3.1. Next, we give an example of row-based data rule. If the current user is a administrator, all the students are displayed in the user interface. If the current user is a tutor, only his/her students are displayed in the user interface. We use the JSON-style expression to describe the semantics, denoted as the following.

```
{"rules":[{"field":"${user.role}","op":"=","value":"administrator"}],
   "groups":[
```

{"rules":[{"field":"${user.role}","op":"=","value":"tutor"},
{"field":"tutor","op":"=","value":"${user.id}"}],"op":"and"}
],
"op":"or"
}

The second category is the column-based data rule. It is recursively defined as columnRule:=(column [, columnRule]), where column indicates the attribute of an object. Next, we give an example of column-based data rule. If the current user is an administrator (a kind of role), all the attributes of the students are displayed, denoted as "*", where * is the wildcard character, which is used to substitute for any other character or characters in a string. If the current user is a tutor (a kind of role), only the student number and name are displayed, denoted as "perNum, perName".

T={start, end} is the valid time bucket, where start and end imply the lifecycle of an access control rule.

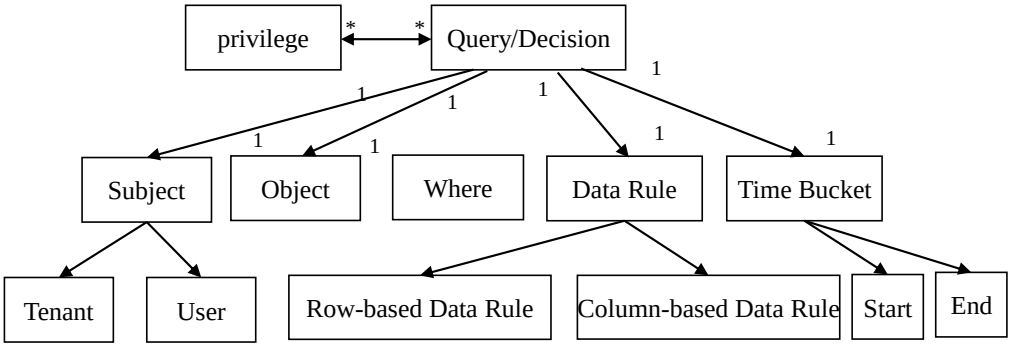

Figure 1. Fine-grained data-level Access Control Model

There are two categories of fine-grained data-level access control models: query privilege model and decision privilege model.

Query privilege model is an instance of the fine-grained data-level access control model shown in Figure 1. For the input to users, privilege engine will execute each policy in order. If the current user for tenants satisfies the user rule of specific query policy, it will return the result of the data query. If the current user does not satisfy all the user segmentation rules, it will show no results.

The following notes describe some conventions. The column-based data privilege reflects in select query, and the row-based data privilege reflects in where query. The sequence of query policy reflects in the priority. The policy with the higher priority is put in the front. Once one user of the tenant satisfies a policy, it will have the proper authorization on its query.

Decision privilege model is another instance of the fine-grained data-level access control model shown in Figure 1. For the input of users, privilege engine will execute each policy in sequence. If the current user satisfies the user segmentation rule of specific decision policy, it will return the result of the decision. If the current user does not satisfy all the user segmentation rules, it will do nothing without any permission.

The following notes describe some conventions. The sequence of decision policy reflects in the priority. The policy with the higher priority is put in the front. Once one user for tenants satisfies a policy, it will have the proper authorization on its manipulation.

4. Architecture and Implementation

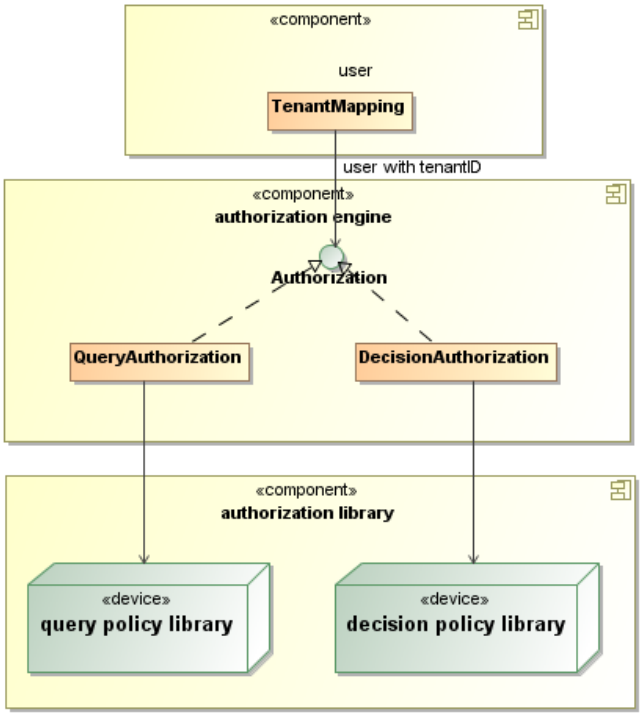

Figure 2. Architecture of Authorization

This section describes the implementation of the authorization model - FDACM described in Section 3. Figure 2 depicts the architecture. The system is composed of two layers: authorization engine and library. The first layer is the authorization engine which provides authorization for multi-tenant applications in the second layer. This API is used by different multi-tenants who want to protect their resources using the sharing applications. The second layer is the authorization library, including all the query and decision policies. To understand the authorization process, consider an authorization request in which a user is trying to access or manipulate a resource. APIs in the authorization engine is designed as follows. The first API is quer, which return the row-based and column-based filtered result of the users. The second API is decide, which return the final decision (true or false) of the operation of the row-based and column-based specific data.

List<QueryResult> query(Subject u, Object o, Where w, List<QueryRule> ps, TimeBucket t);

Boolean decide(Subject u, Object o, Where w, List<DecisionRule> ps, TimeBucket t);

5. Performance Results

This section provides some performance results gathered from the prototype described in Section 4. The authorization server is an Intel Core i5-2300 CPU 2.8 GHz 8Gb RAM with Win7 Ultimate running the authorization API and the MySQL database server. This server is interconnected by an optical fiber with a 1 Gbit Ethernet link.

The emulator starts multi-threading. Each thread represents a user using intensively the authorization system. All the threads are started in parallel in order to stress the

authorization system with concurrent invocations. We do the experiment to compare our dynamic authorization model FDACM with static ACL. The time measured includes: submission of the packet through the network, and reception and processing of the required action in the server. Figure 3 shows the average query and decision time of different three solutions. This scenario consists of ten users invoking the authorization system in parallel. The results show that the execution time of authorization request is nearly constant even when there is an exponential growth of the policies or ACLs. Although the query and decision time of "couple to business" solution is close to the one of FDACM, the FDACM solution is more flexible without the change of the business codes.

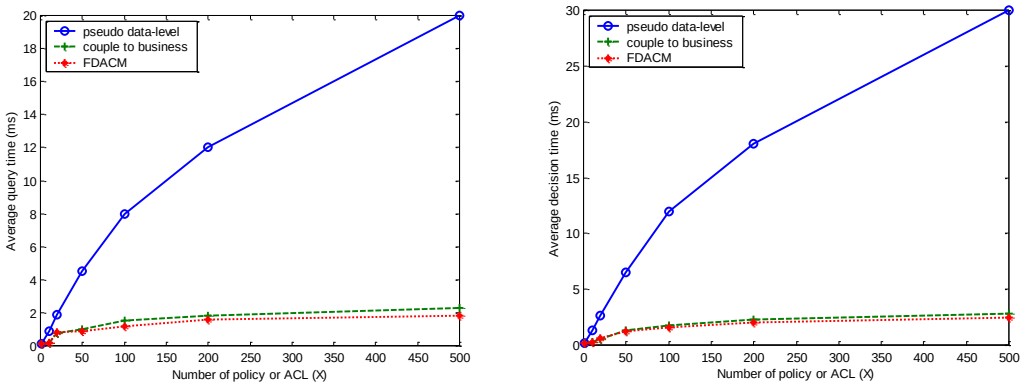

Figure 3. Comparison of Average Query and Decision Time

6. Conclusion

The fine-grained data-level access control model (FDACM) proposed in this paper supports multi-tenancy and data-level access control, which is the combination with traditional RBAC. These features of query privilege model and decision privilege model are intended to provide a convenient authorization service for multi-tenants. We have described the architecture and implementation of the model and demonstrated its scalability. When the business is loosely decoupled from the authorization decision, authorization policies can be updated on the fly and affect business immediately. The case study of FDACM is illustrated further to evaluate the effect of the application in practice.

Future work is targeted in two directions to complete and improve the current proposal. The first target is to provide a uniform identity authentication Cloud service to all the tenants. These features are intended to provide a convenient authorization service for cloud services, such as REST. The second target is to make additional improvements in scalability could be made by experimenting with different underlying databases.

Acknowledgements

This work was supported by the Doctoral Fund of University of Jinan (XBS1237), the Teaching Research Project of University of Jinan (J1344), the Technology development Program of Shandong Province (2011GGX10116), and the National Key Technology R&D Program (2012BAF12B07).

References

[1] M. Armbrust, A. Fox, R. Griffith, A. D. Joseph, R. Katz, A. Konwinski, G. Lee, D. Patterson, A. Rabkin, I. Stoica and M. Zaharia, "A view of cloud computing", Communications of the ACM, ACM, **(2010)**, pp. 50-58.
[2] R. Mietzner, F. Leymann and T. Ungera, "Horizontal and vertical combination of multi-tenancy patterns in service-oriented applications", Enterprise Information Systems, Taylor \& Francis, **(2011)**, pp. 59-77.
[3] A. Younis, M. Merabti and K. Kifayat, "Secure Cloud Computing for Critical Infrastructure: A Survey", Proceedings of 2013 the convergence of Networking, Broadcasting and Telecommunications, IEEE, Salamanca, **(2013)**, pp. 1-6.
[4] V. C. Hu, D. Richard Kuhn, T. Xie and J. Hwang, "Model checking for verification of mandatory access control models and properties", International Journal of Software Engineering and Knowledge Engineering, World Scientific Publishing Co., **(2009)**, pp. 103-127.
[5] S. Osborn, R. Sandhu and Q. Munawer, "Configuring role-based access control to enforce mandatory and discretionary access control policies", ACM Transactions on Information and System Security, ACM, **(2009)**, pp. 85-106.
[6] R. Sandhu, D. Ferraiolo and R. Kuhn, "The NIST model for role-based access control: towards a unified standard", Proceedings of the fifth ACM workshop on Role-based access control, ACM, Berlin, **(2000)**, pp. 47-63.
[7] D. Hardt, "The OAuth 2.0 Authorization Framework", Internet Engineering Task Force (IETF) Request for Comments: 6749, IETF, **(2012)**.
[8] D. Richard Kuhn, E. J. Coyne and T. R. Weil, "Adding attributes to role-based access Control", IEEE Computer, IEEE, **(2010)**, pp. 79-81.
[9] K. Sirbi and P. Jayanth Kulkarni, "Modularization of Enterprise Application Security Through Spring AOP", International Journal of Computer Science & Communication, IJCSCN, **(2010)**, pp. 227-231.
[10] M. Proctor, "Drools: A Rule Engine for Complex Event Processing", Proceedings of the AGTIVE 2011, Springer, Berlin, **(2012)**, pp. 2-2.
[11] Q. H. Mahmoud, "Getting started with the java rule engine api (jsr 94): Toward rule-based applications", SUN Developer Network (SDN), **(2005)**.
[12] A. Sangroya, S. Kumar, J. Dhok and V. Varma, "Towards analyzing data security risks in cloud computing environments", Communications in Computer and Information Science, Springer, **(2010)**, pp. 255-265.
[13] R. Chow, P. Golle, M. Jakobsson, E. Shi, J. Staddon, R. Masuoka and J. Molina, "Controlling data in the cloud: outsourcing computation without outsourcing control", Proceedings of the 2009 ACM workshop on Cloud computing security, ACM, Chicago, **(2009)**, pp. 85-95.
[14] S. Yu, C. Wang, K. Ren and W. Lou, "Achieving secure, scalable, and fine-grained data access control in cloud computing", Proceedings of the 29th conference on Information communications, IEEE Press, San Diego, **(2010)**, pp. 534-542.
[15] A. Almutairi, M. Sarfraz, S. Basalamah, W. G. Aref and A. Ghafoor, "A Distributed Access Control Architecture for Cloud Computing", IEEE Software, IEEE, **(2012)**, pp. 36-44.
[16] J. M. A. Calero, N. Edwards, J. Kirschnick, L. Wilcock and M. Wray, "A Toward a multi-tenancy authorization system for cloud services", IEEE Security & Privacy, IEEE, **(2010)**, pp. 48-55.
[17] B. Tang, R. Sandhu and Q. Li, "Multi-tenancy authorization models for collaborative cloud services", Proceedings of 2013 International Conference on Collaboration Technologies and Systems, IEEE, San Diego, **(2013),** pp. 132-138.
[18] S.-J. Yang, P.-C. Lai and J. Lin, "Design Role-Based Multi-tenancy Access Control Scheme for Cloud Services", Proceedings of 2013 International Symposium on Biometrics and Security Technologies, IEEE, Salamanca, **(2013)**, pp. 273-279.
[19] K. Ma, Z. Chen, A. Abraham, B. Yang and R. Sun, "A Transparent Data Middleware in Support of Multi-tenancy", Proceedings of 7th International Conference on Next Generation Web Services Practices, IEEE, Salamanca, **(2011)**, pp. 11-19.

Authors

Kun Ma, received his Ph.D degree in Computer Software and Theory from Shandong University, Jinan, Shandong, China, in 2011. He is a senior lecturer in Provincial Key Laboratory for Network based Intelligent Computing and School of Information Science and Engineering, University of Jinan, China. He has authored and coauthored over 30 research publications in peer-reviewed reputed journals and conference proceedings. He has served as the program committee member of various international conferences and reviewer for various international journals. He is the Co-Editor-in- Chief of International Journal of Computer Information Systems and Industrial Management Applications (IJCISIM). He is the managing editor of Journal of Information Assurance and Security (JIAS) and Information Assurance and Security Letters (IASL). He is the editorial board member of International Journal of Intelligent Systems Design and Computing and Journal of Software. He is the guest editor of International Journal of Grid and Utility Computing. His research interests include model-driven engineering (MDE), data intensive computing, big data management, and multi-tenant techniques.

Weijuan Zhang, is an IT System Architect at CVIC SOFTWARE ENGINEERING CO., LTD in Jinan, China. Her professional interests focus on approaches and technologies targeted at developing enterprise applications.

Zijie Tang, is an undergraduate student of Dr. Kun Ma, studying at School of Information Science and Engineering, University of Jinan. His research interests include software development of innovative applications, data intensive computing, and big data management.