Sensors & Transducers, Vol. 176, Issue 8, August 2014, pp. 181-188

Sensors & Transducers

Study on Information Security of Distributed Sensor Networks Based on RBAC's Resource Conflict

[1] Su Yu, [2] Zhou Wei, [3] Xiao Xiao Dong, [4] Yin Wang, [5] Ghassan M. Azar
[1, 2] College of Electronic and Electrical Engineering, Shanghai University of Engineering Science, Shanghai, 201620, China
[3] Power Supply Branch, Shanghai Metro Maintenance Security CO., Ltd. Shanghai, 200233, China
[4, 5] Department of Mathematics and Computer Science, Lawrence Technological University, Southfield, MI, 48075, USA
[1] Tel.: (021) 67791138, fax: (021) 67791138
E-mail: suyu_sh@hotmail.com, zhouwei@sues.edu.cn, xxx820522@163.com, ywang12@ltu.edu, gazar@ltu.edu

Received: 21 April 2014 /Accepted: 31 July 2014 /Published: 31 August 2014

Abstract: Wireless sensor networks will be widely deployed in the near future. While much research has focused on making these networks feasible and useful, security has received more attention. There exists a need for multilevel access control in these types of networks, in order to give authorization based on a node's role – this is also called role-based access control (RBAC). SOD of RBAC is a good way to solve the problem of resource allocation and right management for improving the security of the system. According to the needs of the MVC-based information management system security, this paper first researches separation of duties (SOD) in Role-Based Access Control, as well as its application in practice, and then gives a program to implement RBAC model as the framework of SSH. Finally, this paper describes how to realize the RBAC in a specific information management system. *Copyright © 2014 IFSA Publishing, S. L.*

Keywords: Distributed sensor network, RBAC, SOD, SSH, Resource conflict, Security.

1. Introduction

While much research has focused on making these wireless sensor networks feasible and useful, security has received more attention. A wireless sensor network (WSN) typically comprises of a large number of sensor nodes deployed in high density in an area. These nodes have limited processing capability, limited memory and are powered by very limited energy batteries – they are, thus, heavily resource constrained, and has limited life. Sensor nodes work together and coordinate among themselves in order to do a common task. Typically, they are left unattended in an area for capturing physical phenomena occurring in their ranges.

WSNs are also used in highly sensitive areas where secret information has to be transmitted to the base station using multi-hop communication. As they use the open air as a communication medium, an adversary can easily eavesdrop the information can attack the already deployed nodes, making them malicious in order to have access to the information, or can introduce new malicious nodes in the existing network [5]. Intruders can also harm the network's security, and their detection [6–8] and eviction are important considerations. Introducing security mechanisms generally involves introducing cryptographic methods, which require high memory usage and power consumption in processing and communication. Different types of users in the

system have the same privileges to some information, and they can modify and delete arbitrarily, which is very dangerous for some important information. As scientific management of information, the highest level the user has the highest authority of the information, and at the bottom or sub underlying user has the least permissions information. In addition, most of the Web systems are the lack of a good access control mechanisms. In the mechanism users, roles and privileges are set, and then the roles are assigned for different users, the privileges are assigned for different roles. Finally privileges are associated with the users, for formation of an effective system of safety management. In conclusion, when the enterprises build their own web-based management system, not only to consider the functional integrity and simplicity of operation, but also to consider the security of the system. Under the promise of safety, the functional integrity and simplicity of operation makes sense.

Compared with traditional lattice-based access control policies, such as Discretionary Access Control (DAC) and Mandatory Access Control (MAC) developed primarily for military systems [2], RBAC can more effectively meet the needs of commercial systems [3-4]. Role-based access control mechanisms rely on convenient resource management. In RBAC there are two important factors. They are role inheritance and separation of duties. Role inherited makes resource allocation management easier; Role mutex achieves separation of duties and improves efficiency and stability of system. Separation of duties has some theoretical results, and many researchers have given a variety of theoretical models. But in the previous literature, how to implement the separation of duties in the information management system was rarely described. The implementation of the separation of duties is far less than the theoretical research. So, the implementation of the separation of duties still need further study [1].

The main goal of this paper studies RBAC model structure, specially focus on RBAC's resource conflict based on SOD and restrictions. After intensive research RBAC theory, this paper designs the RBAC model structure based on SOD and it can be used in company's information management systems. The RBAC model structure developed by this project can be used in the web-based information management system. The system uses SSH framework based on MVC, development software is Java, database is MYSQL. Access control software based on RBAC can be applied to the Waigaoqiao company's management system.

2. Security Access Control Mechanisms

Access level is important to computer system security. To compromise a system, attackers try to gain any possible level of access and then try to escalate that level until they are able to obtain restricted data or make unapproved system modifications. Because each user has some level of system access, every user account on your system increases the potential for abuse. System security has historically relied on trusting users not to abuse their access, but this trust has proven to be problematic. The goal of access control is to prevent unauthorized users from using the information system resources. Mainstream access control mechanism in three ways: Discretionary Access Control (DAC), Mandatory Access Control (MAC) and role-based access control (RBAC).

2.1. Discretionary Access Control (DAC) and Mandatory Access Control (MAC)

In DAC, generally the resource owner (a user) controls who has access to a resource. A user is granted permissions to a resource by being placed on an access control list (ACL) associated with resource. An entry on a resource's ACL is known as an Access Control Entry (ACE). When a user (or group) is the owner of an object in the DAC model, the user can grant permission to other users and groups. The DAC model is based on resource ownership. For convenience, some users commonly set dangerous DAC file permissions that allow every user on the system to read, write, and execute many files that they own. In addition, a process started by a user can modify or delete any file to which the user has access. Processes that elevate their privileges high enough could therefore modify or delete system files. These instances are some of the disadvantages of DAC. Another example: A can access B, B can access C, then A will be able to access the C. As the access rights of the user are "independent", the user can arbitrarily decided that the permission given to a person, this system can't control, so there may be a leak of the data. These instances are some of the disadvantages of DAC.

MAC in the Mandatory Access Control (MAC) model, users are given permissions to resources by an administrator. Only an administrator can grant permissions or right to objects and resources. Access to resources is based on an object's security level, while users are granted security clearance. Only administrators can modify an object's security label or a user's security clearance. The MAC's disadvantage is the lack of flexibility. The MAC mechanism is used primarily for the security level of the multi-level applications, such as military defense and government departments.

2.2. Role-Based Access Control (RBAC)

In the Role-Based Access Control (RBAC) model, access to resources is based on the role assigned to a user. In this model, an administrator assigns a user to a role that has certain predetermined right and privileges. Because of the user's association

with the role, the user can access certain resources and perform specific tasks. RBAC is also known as Non-Discretionary Access Control. The roles assigned to users are centrally administered. Access control is used to restrict access to a resource. The basic concept of RBAC is shown in Fig. 1.

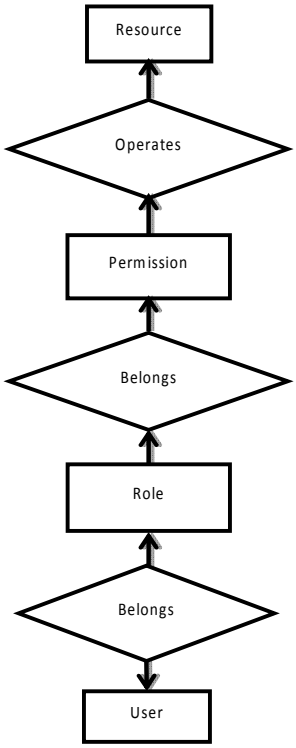

Fig. 1. The basic concept of RBAC.

The basic concept of RBAC is that users are assigned to roles, permissions are assigned to roles, and users acquire permissions by being members of roles. $RBAC_0$ includes requirements that user-role and permission-role assignment can be many-to-many. Thus the same user can be assigned to many roles and a single role can have many users. For permissions, a single permission can be assigned to many roles and a single role can be assigned to many permissions.

In an organization, the roles are to be created in order to complete a variety of tasks. In accordance with their responsibilities and qualifications users are to be assigned to the appropriate roles. Based on the needs of the system, the roles can be given the appropriate permissions.

2.3. Separation of Duty (SOD)

Separation of duties (SOD) is the concept of having more than one person required to complete a task. For example, in business one person can't be an accountant while he is a cashier. RBAC requires separation of duty to prevent a role from having too many permissions. By this, RBAC can secure system. Through role conflict, we can achieve SOD. Role assignment is an important part of RBAC. But when apply RBAC into a real system, security problems occurs because of no constraint on role assignment. So SOD is necessary to RBAC model. System security partly relies on SOD. What SOD does in RBAC is to separate roles which conflicts with each other. Strong constraint does not allow any roles being added into role set. Weak constraint does not allow roles in a special event being added into role set. Combining with other limits, such as frequency constraint and cardinality constraint, user will get through an SOD firewall before role assignment, separating conflicting roles, making role assignment and resources fetching safely.

SOD is very important in business system, also is one of the most desired features in RBAC system. Administration constraints may need to be enforced to prevent information misuse and prevent fraudulent activities. A typical authorization constraint, broadly relevant and well recognized, is separation of duties (SOD). Reducing the risk of fraud by not allowing any individual to have sufficient authority within the system to single-handedly perpetrate fraud is the intent of SOD. Such constraints can be easily expressed using an RBAC model through SOD constraints on roles, user-role assignments, and role-permission assignments. Furthermore, using constraints on the activation of user assigned roles, users can sign on with the least privilege set required for any access. In case of inadvertent errors, such least privilege assignments can contain damage. Simon [9] divided SOD into five categories: static SOD (SSOD), dynamic SOD (DSOD), object-based SOD (ObSOD), operation-based SOD (OpSOD), history-based SOD (HSOD). SSOD is that conflict two roles can't be given the same user at the same time. DSOD is that Conflict two roles can't be activated in the same session. ObSOD is that conflicting roles can have common user, the user also can activate simultaneously conflicting roles, but the user can't operate on the same object more than 2 times. OpSOD is that operating group allocated to the restricted role does not contain all operations to complete a task, the restricted role can be delegated to a user, this prevents a user from all operating authority to complete a task. HSOD is that conflict could be from user who was assigned a role that is mutually exclusive with the role he is assigned [10-11].

3. A Program for Implementing RBAC's Resource Conflict Based on SSH

3.1. SSH Work Mode and Configuration

Several problems can arise when applications contain a mixture of data access code, business logic code, and presentation code. Such applications are difficult to maintain, because interdependencies between all of the components cause strong ripple

effects whenever a change is made anywhere. High coupling makes classes difficult or impossible to reuse because they depend on so many other classes. Adding new data views often requires reimplementing or cutting and pasting business logic code, which then requires maintenance in multiple places. Data access code suffers from the same problem, being cut and pasted among business logic methods. The Model-View-Controller design pattern solves these problems by decoupling data access, business logic, and data presentation and user interaction.

Model-View-Controller (MVC) is a software architecture pattern which separates the representation of information from the user's interaction with it. In addition to dividing the application into three kinds of components, the MVC design defines the interactions between them.

SSH (struts, spring, hibernate, three frameworks for Java platform) is a classic MVC pattern. Actually Struts is a full MVC pattern. ActionServlet which is a part of Struts plays the controller role in MVC model. ActionForm and JavaBean play the model role. And JSP plays the view role. Spring framework is an open source application framework and inversion of control container for Java platform. It simplifies enterprise application development. Hibernate is an object-relational mapping library for Java platform, providing a framework for mapping an object-oriented domain model to a traditional relational database. Hibernate solves object-relational impedance mismatch problems by replacing direct persistence-related database accesses with high-level object handing functions.

SSH as a classic MVC pattern, the three frameworks play different roles. Struts is responsible for the web layer. Spring is for service layer (or called manager layer). And Hibernate is for persistence layer. Struts implements the MVC hierarchy, it makes JSP page, Action scheduling and specific business logic processing separated; Hibernate makes JDBC a very lightweight object package, which can manipulate the database arbitrarily using the object-oriented programming; Spring realizes interface-oriented programming using JavaBean, and provides many enterprise application functionality. Spring is an open source framework, created by Rod Johnson. It is created in order to solve the complexity of enterprise application development. Spring uses the basic JavaBean to complete things that may be completed by the EJB. However, Spring's purpose is not limited to the development of server-side. When the program is running, if you need help to call another object without creating callee, you have to depend on the external injection. Spring's dependency injection of the caller and the callee almost has no requirements to fully support the management of POJO dependency. Spring's dependency injection relationship as shown below Fig. 2.

Spring offers Hibernate call method. In the project Spring configuration file is applicationContext.xml, mainly to configure a data source, dependency injection structure.

Data source configuration as shown below.

```
<bean id="dataSource"
class="org.apache.commons.dbcp.BasicDataSource">
    <property name="driverClassName"
value="com.mysql.jdbc.Driver"/>
    <property name="url"
value="j`dbc:mysql://localhost:3306/person"/>
    <property name="username" value="root"/>
    <property name="password" value=""/>
</bean>
```

<bean> tag is the basic element. In the Spring bean able to rely on other beans, it can also be injected into other beans, the dataSource is the most basic bean.

Hibernate.hbm.xml configuration as shown below.

```
<bean id="sessionFactory"
    class="org.springframework.orm.hibernate3.Loc
alSessionFactoryBean">
    <property name="dataSource" ref="dataSource"/>
    <property name="hibernateProperties">
        <props>
            <prop key="hibernate.dialect">
                org.hibernate.dialect.MySQLDialect
            </prop>
            <prop
key="hibernate.show_sql">true</prop>
            <prop
key="hibernate.format_sql">true</prop>
        </props>
    </property>
    <property name="mappingResources">
        <list>
<value>jsp_struts_hibernate/hibernate/UA.hbm.xml<
/value>
<value>jsp_struts_hibernate/hibernate/Customer.hbm
.xml</value>
<value>jsp_struts_hibernate/hibernate/SOD.hbm.xml
</value>
<value>jsp_struts_hibernate/hibernate/Department.h
bm.xml</value>
<value>jsp_struts_hibernate/hibernate/Roles.hbm.xm
l</value>
<value>jsp_struts_hibernate/hibernate/Permission.hb
m.xml</value>
<value>jsp_struts_hibernate/hibernate/Resource.hbm
.xml</value>
<value>jsp_struts_hibernate/hibernate/PR.hbm.xml</
value>
<value>jsp_struts_hibernate/hibernate/Users.hbm.xm
l</value>
<value>jsp_struts_hibernate/hibernate/PA.hbm.xml</
value>
<value>qiantai/hibernate/Password.hbm.xml</value>
<value>qiantai/hibernate/Event.hbm.xml</value>
<value>qiantai/hibernate/EUR.hbm.xml</value>
        </list>
    </property>
</bean>
```

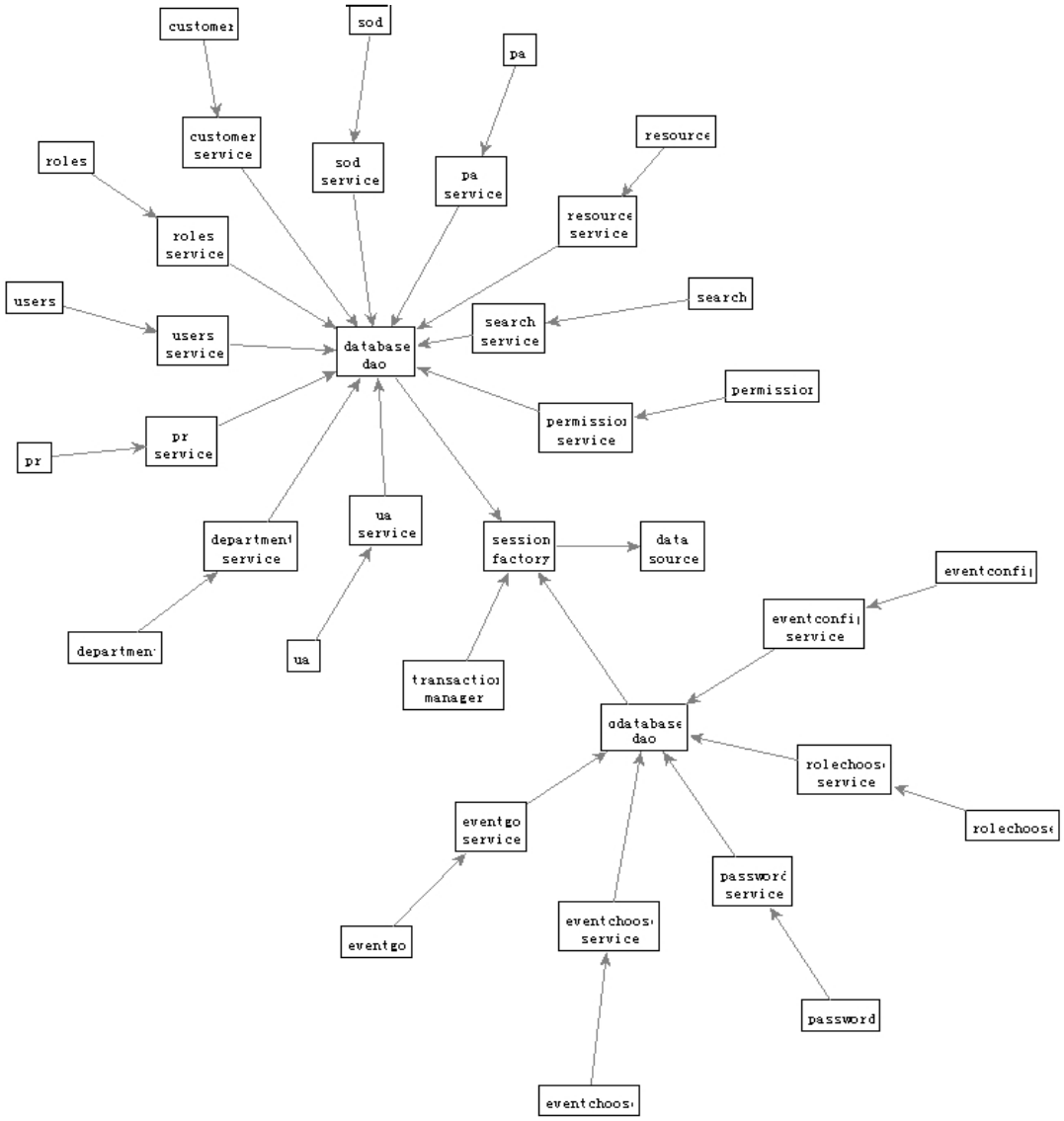

Fig. 2. Spring's dependency injection relationship.

The sessionFactory relays on dataSource. Bean is a basic database connections, and other database operations relays on the bean to be connected. DatabaseDAO and qdatabaseDAO, as the bean, respectively connect database of background and foreground, and they dependent sessionFactory. In databaseDAO class there are various types of database access operations, such as addition and deletion of the various types of tables and different query methods. The sessionFactory contains Hibernate function. Through inheritance HibernateDaoSupport class, Hibernate's api is used. getHibernateTemplate () method lets users do not need to know transaction's process. The creation and destruction of the session is done automatically.

Business logic dependency injection as shown below.

```
<bean id="uaService"
class="jsp_struts_hibernate.service.impl.UAServiceI
mpl">
<property name="databaseDAO">
  <ref local="databaseDAO"/>
</property>
</bean>
```

Service accepts various requests from Action for processing, and then returns results to the Action. In this project Service contains the foreground and background. The background: user password processing "customerService", user role assignment processing "uaService", separation of duties information processing "sodService", department information processing "departmentService", the role information processing "rolesService", permission information processing "permissionService", resource information processing "resourceService", permissions resource allocation processing "prService", user information processing "usersService", role permissions allocation processing "paService", the user roles inquiry processing "searchService"; foreground password inquiry processing "passwordService", event set

processing “eventconfigService”, event judgment processing “eventchooseService”, event workflow processing “eventgoService”, role selection processing “rolechooseService”.

Action dependency injection as shown below.

```
<bean name="ua"
class="jsp_struts_hibernate.action.ShowUAAction">
<property name="uaService">
        <ref local="uaService"/>
</property>
</bean>
```

Service mentioned in the project has a corresponding Action.

3.2. A Program for Implementing RBAC’s Resource Conflict

This project uses MySql as a background database. The main line is user - role - permission – resource, respectively corresponding to the many-to-many relationship. Three intermediate tables are used. Extended separation of duties requires a separate table to configure, User department also requires the department table associated with the user table. Login password for security purposes should establish a password table associated with user table.

Database table structure: User Table (user), Role Table (role), Permissions Table (permission), Resource Table (resources), Corresponding Table of Users - Role (ua), Corresponding Table of Role - Permissions (pa), Corresponding Table of Permission - Resource (pr), Separation of Duties Enumeration Table (sod), Administrator Password Table (customer), User Password Table (password), Departments Tables (department), Event Table (event), Event User Table (eur), Session Table (sessions). E-R diagram for system as shown Fig. 3.

The main function of foreground is to provide users with reasonable access resources programs, and to achieve the separation of duties rules. The foreground has event set, event selection, role selection and of resource operation. In which event set to judge whether the presence of the Minister of user roles permission, if not the authority of the Secretary can't set the event. Selection of the event, the first to determine whether the user Minister added to this event, if it has not added to the event, you can't enter this event. And then determine whether the user first log on to the event, if it is the first time you log in, you need to select the role in the event, in the role is selected, it is necessary to determine whether the role that has expired, and whether more than the frequency of use, if you choose a multi-a role, the need to determine whether the role of the role of the group have been included in the table dynamic separation of duties conflict.

The Admin main task is to modify users, roles, permissions, resources, separation of duties, as well as correspondence between most of the backend interfaces are used to configure this information. Information configuration page include: user password modification, department information changes, user information modify the role information modify permissions information changes, modifications of the resource information corresponding information to modify user roles, role permissions corresponding information to modify permissions to modify the resources corresponding information, static separation of duties allocated to modify, the historical separation of duties assigned to modify, dynamic separation of duties assigned to modify.

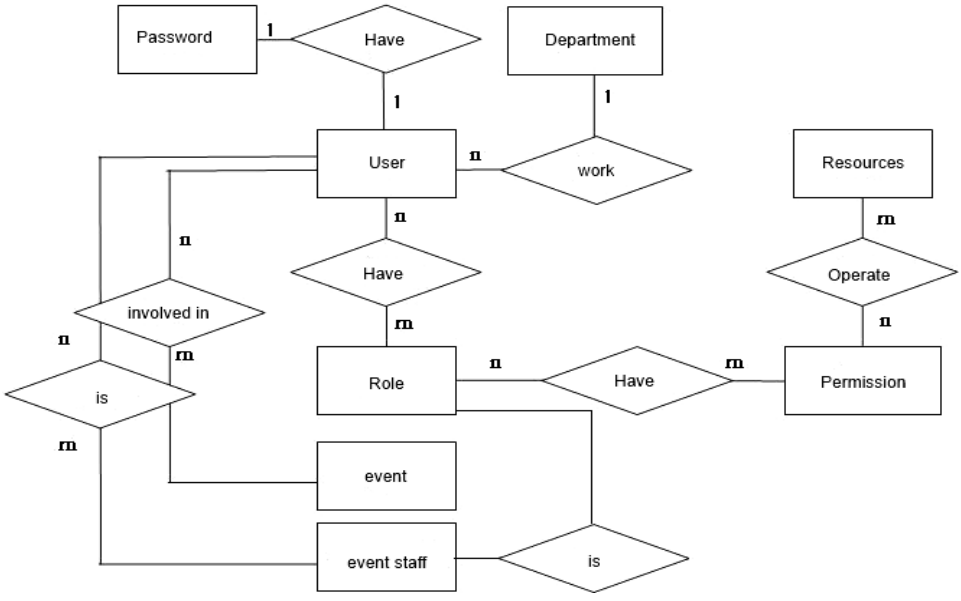

Fig. 3. E-R diagram.

4. Conclusions

Access control plays an important role in providing security in a wireless sensor network, keeping in mind the node's constraints in the network. The main topic of the role-based access control mechanism is to solve the problem of the management of the information management system. This paper details the entire project from theoretical research to planning and design, and then to complete the final step. The foreground and background of the design is based on a three-tier architecture, mainly to solve the user how to obtain resources, and managers how to allocate resources. According to the needs of the MVC-based information management system security, this paper first researches separation of duties (SOD) in Role-Based Access Control, as well as its application in practice, and then gives a program to implement RBAC model as the framework of SSH. Finally, this paper describes how to realize the RBAC in a specific information management system. The implementation of the project is based on the Wai Gao Qiao Shipyard ERP, but because of the amount of data is too large, the interception of the part of the data used to test and validate the feasibility of the program. The paper has a certain theoretical and practical value.

Acknowledgements

Here special thanks to the two foundations for funding the project. National High Technology Research and Development Program (863 Program) supported by Ministry of Science and Technology of the People's Republic of China, project number is 2013AA040302. Shanghai Local Colleges and Universities "十二五" Connotation to Build Project support by Shanghai City Hall, project number is nhjx-2012-07.

References

[1]. David Farraiolo, Richard Kuhn, Role-Based Access Control, in *Proceedings of the 15th NIST-NCSC National Computer Security Conference*, 1992, pp. 554-563.

[2]. Ravi Sandhu, Lattice Based Access Control Models, *IEEE Computer*, 26, 11, November 1993, pp. 9-19.

[3]. David D. Clark, David R. Wilson, A Comparison of Commercial and Military Computer Security Policies, in *Proceedings of the IEEE Symposium on Research in Security and Privacy (SP'87)*, May 1987, pp. pp. 184-194.

[4]. Michael J. Nash, Keith R. Poland, Some Conundrums Concerning Separation of Duty, in *Proceedings of the IEEE Symposium on Research in Security and Privacy*, May 1990, pp. 184-194.

[5]. L. Maccari, L. Mainardi, M. A. Marchitti, N. R. Prasad, R. Fantacci, Lightweight, distributed access control for wireless sensor networks supporting mobility, in *Proceedings of the IEEE International Conference on Communication (ICC'08)*, Vol. 7, May 2008, pp. 1141–1145.

[6]. C. Karlof, N. Sastry, D. Wagner, TinySec: a link layer security architecture for wireless sensor networks, in *Proceedings of the 2nd ACM Conference on Embedded Networked Sensor Systems (SensSys'04)*, Baltimore, Maryland, November 2004, pp. 162-175.

[7]. Y. Zhou, Y. Zhang, Y. Fang, Access control in wireless sensor networks, *Ad Hoc Networks*, 5, 1, 2007, pp. 3–13.

[8]. L. Eschenauer, V. Gligor, A key management scheme for distributed sensor networks, in *Proceedings of the 9th ACM Conference on Computer and Communications Security (CCS'02)*, Washington, DC, 2002, pp.41-47.

[9]. Jerome H. Saltzer, Michael D. Schroeder, The Protection of Information in Computer Systems, *Proceedings of the IEEE*, Vol. 63, Issue: 9, 1975, pp. 1278 - 1308.

[10]. Ravi Sandhu, *et al*., Role Based Access-Control Models, *IEEE Computer*, Vol. 29, No. 2, February 1996, pages 38-47.

[11]. Ravi Sandhu, Rationale for the RBAC96 Family of Access Control Models, in *Proceedings of the 1st ACM Workshop on Role-based Access Control*, Article No.9, February 1996.

[12]. David Ferraiolo, *et al*., Proposed NIST Standard for Role-Based Access Control, *ACM Transactions on Information and System Security*, Vol. 4 Issue 3, August 2001, pp. 224-274.

[13]. Simon R., Zurko M. E., Separation of duty in role based access control environments, in *Proceedings of the 10th IEEE Workshop on Computer Security Foundations*, Rockport, MA, 10-12 June 1997, pp. 183–194.

[14]. Virgil D. Gligor, Serban I. Gavrila, David Ferraiolo, On the formal definition of separation-of-duty policies and their composition, in *Proceedings of the IEEE Symposium on Research in Security and Privacy*, Oakland, CA, May 1998, pp. 172-183.

[15]. Gail-Joon Ahn & Ravi Sandhu, Role-based authorization constraints specification, *ACM Transactions on Information and System Security*, 3, 4, 2000, pp. 207-226.

[16]. Chunyang Yuan, *et al*., A Verifiable Formal Specification for RBAC, *Lecture Notes in Computer Science*, Vol. 4318, 2006, pp. 196-210.

[17]. Ferraiolo D., Cugini J., Kuhn D. R., Role-Based Access Control (RBAC): Features and Motivations, in *Proceedings of the Computer Security Applications Conference*, December 1995, pp. 241-248.

[18]. Ahn G.-J., Sandhu R., The RSL99 language for role-based separation of duty constraints, in *Proceedings of the 4th ACM Workshop on Role-Based Access Control (RBAC'99)*, Fairfax, VA, October 28-29, 1999, pp. 43–54.

[19]. G. J. Ahn, R. Sandhu, The RSL99 language for role-based separation of duty constraints, in *Proceedings of the ACM Workshop on Role-Based Access Control*, Fairfax, Virginia, USA, 1999, pp.43-54.

[20]. G. J. Ahn, R. Sandhu, Role-based authorization constraints specification, *ACM Trans on Information and System Security*, 3, 4, 2000, pp. 207-226.

[21]. D. R. Kuhn, Mutual exclusion of roles as a means of implementing separation of duty in role-based access control system, in *Proceedings of the 2nd ACM*

Workshop on Role-Based Access Control, Fairfax, VA, 1977,

[22]. N. H. Li, Q. H. Wang, M. V. Tripumitara, Beyond separation of duty: An algebra for specifying high-level security policies, *Purdue University, CERIAS, Tech Rep: 2005-75*, 2005.

[23]. Joon S. Park, *et al*., Role-based access control on the web, *ACM Transactions on Information and System Security*, Vol. 4, No. 1, February 2001, pp.37-71.

[24]. Jean Bacon, *et al*., A model of OASIS role-based access control and its support for active security, *ACM Transactions on Information and System Security*, Vol. 5, No. 4, November 2002, pp. 492–540.

[25]. Jason Crampton, Hemanth Khambhammettu, Delegation in role-based access control, *International Journal of Information Security*, Vol. 7, Issue 2, April 2008, pp.123-136.

[26]. Zhang Zhiyong, Collaboration Access Control Model for MAS Based on Role and Agent Cooperative Scenarios, in *Proceedings of the IEEE International Conference on Mechatronics and Automation*, 2006, pp. 825 - 830.

[27]. B. Panja, S. K. Madria, B. Bhargava, A role-based access in a hierarchical sensor network architecture to provide multilevel security, *Computer Communications*, 31, 4, 2008, pp. 793–806.

[28]. I. F. Akyildiz, W. Su, Y. Sankarasubramaniam, E. Cayirci, A survey on sensor networks, *IEEE Communications Magazine*, 40, 8, 2002, pp. 102–114.

[29]. A. Perrig, R. Szewczyk, J. D. Tygar, V. Wen, D. E. Culler, SPINS: security protocols for sensor networks, *Wireless Networks*, 8, 2002, pp. 521–534.

[30]. Y. Shen, J. Ma, Q. Pei, An access control scheme in wireless sensor network, in *Proceedings of the 4th IFIP International Conference on Network and Parallel Computing Workshops (NPC'07)*, Dalian, China, September 2007, pp. 362–367.

[31]. A. Boukerch, L. Xu, K. El-Khatib, Trust-based security for wireless ad hoc and sensor networks, *Computer Communications*, 30, 2007, pp. 2413–2427.
