文章编号:1001 - 9081(2006)06 - 1331 - 03

J2EE 模式下基于角色访问控制的应用

刘孝保[1],杜平安[1,2]
(1. 电子科技大学 机械电子工程学院,四川 成都 610054;
2. 机械制造系统工程国家重点实验室,陕西 西安 710049)
(blackhorse8000@ sohu. com)

摘 要:利用 J2EE 基于容器的安全机制,将基于角色的访问控制(Role-Based Access Control, RBAC)融入 J2EE 安全之中,用 RBAC 来管理 J2EE 组件安全而不需修改各组件的代码。RBAC 的实现只需通过容器部署描述符配置既可,系统运行时,容器从部署描述符中读取出相应的 RBAC 安全策略,然后根据安全策略执行安全验证,实现 RBAC 对系统资源的保护。这样,组件开发和系统安全相分离,开发人员可不关心系统安全,集中精力投入组件的业务逻辑开发,安全问题由组件组装人员、部署者和系统管理员来完成。因此,降低了对编程人员的要求和系统开发成本开发成本,也提高了系统组件的可移植性。

关键词:基于角色的访问控制;J2EE;申明性安全策略;部署描述符;配置
中图分类号: TP309.2 **文献标识码**:A

Application study of role-based access control under J2EE mode

LIU Xiao-bao[1], DU Ping-an[1,2]
(1. *Department of Mechatronics Engineering, University of Electronic Science & Technology of China, Chengdu Sichuan* 610054, *China;*
2. *State Key Laboratory for Manufacturing Systems Engineering, Xi'an Shaanxi* 710049, *China*)

Abstract: By using the container-based security mechanism of J2EE, the system security of J2EE can be managed by RBAC (Role-based Access Control) without code modification of J2EE components. RBAC was realized simply by the deployment descriptor configure of J2EE container. RBAC security strategy, executed to protect J2EE system resource based on the security truss of container, was picked up from deploy descriptor. Because the development of J2EE component separates from security, the developers of components can be absorbed in the businesses logic of components. Therefore, the cost of system development and maintenance will fall down, and the components become transplantable.

Key words: role-based access control; J2EE; declarative security strategy; deployment descriptor; configure

J2EE 由于其稳定的可用性、可伸缩性、支持异构环境、可高效地开发等优势和良好的运行机制,能够较为完备地搭建具有高伸缩性、灵活性、易维护的分布式企业应用系统,因此,已成为目前企业级软件系统的主要技术框架之一。对企业而言,安全是系统的关键,在构架 J2EE 系统时,应充分考虑其安全的实现问题。

在信息安全访问控制领域,学术界进行了许多研究,普遍认为基于角色的访问控制(Role-Based Access Control,RBAC)在数据安全控制方面是代替传统自主访问控制(DAC)和强制访问控制(MAC)的理想候选。RBAC 为管理大量的资源访问权限提供了一种灵活的、动态的方法,尤其适合大型企业级系统[1]。但如何将 RBAC 合理运用于应用系统是一个比较复杂的问题。本文主要从应用角度研究将 RBAC 技术合理应用于 J2EE 系统中,把 RBAC 这一先进有效的安全模式引入到 J2EE 中来保护系统资源。采用 J2EE 基于容器的申明性安全机制来灵活的实现 RBAC,组件开发者将不需考虑安全问题,组件安全由组件装配人员和管理员通过容器部署配置来完成,使系统的开发效率和系统组件的可移植性得以提高。

1 J2EE 基于容器的安全机制

在 J2EE 环境中,采用基于容器的安全机制。J2EE 容器主要包括 Web 容器和 EJB 容器。系统组件的安全由容纳它们的容器各自来负责,组件开发人员几乎可以不用或者很少在组件中添加有关安全代码。系统组件开发与系统安全设计相分离,开发人员就可以全心全意地进行组件业务逻辑的开发而无需考虑安全,安全问题由组件组装人员和部署者以及系统管理员来完成[2]。因此,可以提高编程效率和降低编程人员的要求。同时,由于安全策略可以由配置来完成,使系统组件具有良好的可移植性。

J2EE 规范要求 J2EE 相关产品必须为应用程序开发者提供两种形式的基于容器的安全策略:申明性安全策略和程序性安全策略。由于编程性安全策略需要将安全代码强加于系统组件,既破坏了原组件的完整性,又使组件的可移植性降低。因此,申明性安全应尽可能地代替程序性安全使用[2]。

收稿日期:2005 - 12 - 02;**修订日期**:2006 - 03 - 01
基金项目:国家 863 计划项目(2003AA411210)
作者简介:刘孝保(1978 -),男,重庆奉节人,博士研究生,主要研究方向:ERP/CAE/CAPP/CAD; 杜平安(1962 -),男,重庆开县人,教授,博士生导师,博士,主要研究方向:数字化设计、仿真与制造等.

申明性安全策略不需组件开发人员编写任何安全相关的代码,一切都通过配置部署描述符配置完成。容器的申明性安全策略通过 J2EE 容器部署描述符来代表应用程序的安全需求,一般包括安全角色、访问控制和验证要求等[3]。

部署描述符是组件开发者和应用程序部署者或应用程序组装者之间交流的平台。应用程序开发者用它来表示应用中的安全需求,应用程序部署者或应用程序组装者将安全角色与部署环境中的用户和组映射起来。在程序运行时,容器从部署描述符中读取出相应的安全策略,然后容器根据安全策略执行安全验证[4]。部署描述配置主要包括:指定用户登录认证方式的登录策略配置、指定访问控制权限的认证策略配置、实现角色主体委托的委托策略配置和安全域的配置等。

2 基于角色的访问控制

基于角色的访问控制由 Ferraiolo 等人于 20 世纪 90 年代初提出来的。美国国家标准化和技术委员会(NIST)成立了 RBAC 研究机构,专门对基于角色的访问控制进行研究。Sandhu 等人在对 RBAC 进行深入研究的基础上,在 1996 年提出了一个基于角色的访问控制参考模型,即 RBAC96 模型[2]。此后,又相继提出了 RBAC97、RBAC99 以及各种扩展模型。图 1 为 RBAC 的核心思想:通过引入角色这一“中介”,将用户和权限相分离,使用户和具体权限不直接相关,用户通过角色来拥有访问权限。有了“角色”的介入使系统的权限管理的更加方便和灵活[3~5]。目前,RBAC 大有取代 DAC 和 MAC 的趋势。

图 1 RBAC 核心思想

但是,目前 RBAC 研究主要侧重于本身角色关系和约束的研究[1],而涉及的具体运用实施研究相对较少。一般地,RBAC 的实现比较复杂,需要在系统的相关组件中编写相应的安全代码而破坏原系统的完整性。同时,系统通常会分布运行在第三方软件运行环境中(如 Tomcat WAS,Tomcat 网络服务器),使系统运行安全难以控制。

3 RBAC 在 J2EE 模式下的实现

用 RBAC 来管理系统安全而不破坏系统各组件的完整性,一般说来比较难以实现,而 J2EE 基于容器的申明性安全策略却使其变得可能。利用 J2EE 容器的安全实现框架和申明性策略的相关配置,将 RBAC 融入系统的安全管理中,构建一个安全灵活的 J2EE。由于 RBAC 的实现只是需在容器的部署描述进行配置,因此系统将具有良好的可移植性、较低成本等特点,同时也兼顾了 RBAC 安全管理的灵活性和良好的可管理性。

RBAC 访问权限验证流程如图 2 所示。用户访问系统时,首先进行身份验证,然后对验证通过的用户进行访问权限的验证,对于验证成功的正确显示结果。结合 J2EE 基于容器的申明性安全策略,RBAC 在 J2EE 中的安全验证流程如图 3 所示。其中虚线表示容器相应的申明性策略配置。

RBAC 的实现主要包括 RBAC 信息的保存与取用、身份验证、访问权限控制策略等。结合 J2EE 基于容器的申明性安全策略,RBAC 实现方式为:RBAC 基础信息(用户信息、角色信息以及用户/角色的相应授权信息)通常保存在数据库、LDAP(Lightweight Directory Access Protocol)服务器、或者数据文件中,而 RBAC 基础信息和 RBAC 相应安全策略则通过如下部署配置方式来实现有机结合:

1)利用容器部署描述 XML 文件中的登录策略配置来指定 RBAC 用户登录认证方式。认证方式分为三种:HTTP 认证方式、表单认证方式和基于证书的认证方式。认证方式的指定由 login-config 描述符的 auth-method 元素指定。对于 HTTP 认证方式还需指定 realm-name 中的安全区域名称。对于表单认证方式,还需要配置 form-login-page 和 form-login-error 来指定登录验证后的正确显示页面和错误显示页面。对于企业级的 Web 应用通常采用基于表单的认证方式[5]。

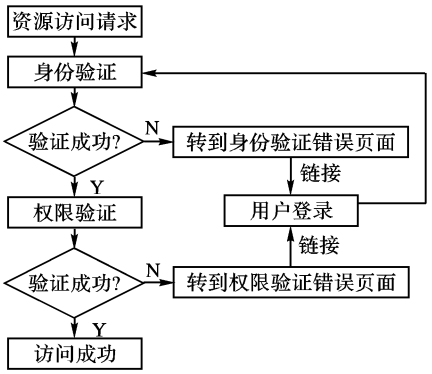

图 2 RBAC 访问权限验证流程

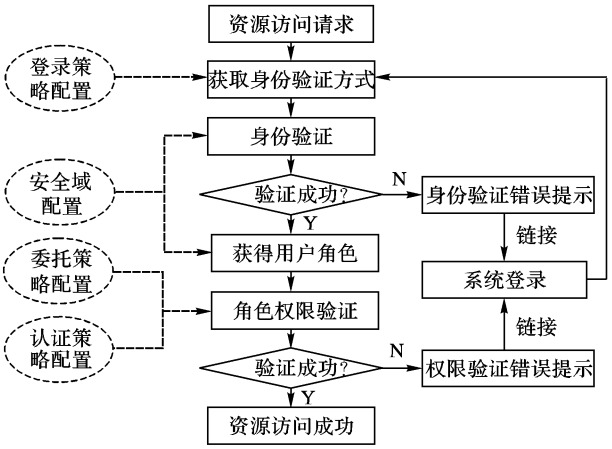

图 3 RBAC 在 J2EE 中的安全验证流程

2)利用容器部署描述 XML 文件中的认证策略配置来实现 RBAC 角色访问权限的指定。通过在 security-constrain 中配置角色名、资源名称、资源操作来实现角色对某个资源的资源操作权限的限制。如下为一个 Web 容器的认证策略配置:

```
<security-constraint>                          //访问权限控制部分
  <web-resource-collection>                    //受控制的资源
    <web-resource-name>firstPage</web-resource-name>
    //资源名称
    <url-pattern>/welcome.jsp</url-pattern>
    //具体资源
    <http-method>GET</http-method>
                                   //受控制的资源操作方法
  </web-resource-collection>
  <auth-constraint>
    <role-name>role1</role-name>
                                   //可以访问的安全角色
  </auth-constraint>
</security-constraint>
```

如上配置表示角色 role1 具有对资源/welcome.jsp 的 GET 方式的 HTTP 访问操作权限。而对 EJB 的访问权限配置

与 JSP 和 Servlet 的配置方式有所不同,需指定 EJB 的名称(ejb-name)和方法(ejb-method)替代 JSP 的 web-resource-name 和 http-method。

3) 利用容器部署描述 XML 文件中的委托策略配置来实现 RBAC 的角色继承。通过指定 security-identity 元素来确定其是否继承,其默认为不继承(user-caller-identity),设置为 run - as 表示继承且需指定继承角色名(role-name)。

4) 利用容器安全域(realm)配置来实现 RBAC 的用户身份验证策略指定,以及 RBAC 角色与 RBAC 用户关系的映射。安全域的配置主要包括指定用户登录验证模块 LoginModule 以及其需要指定的相应实现参数(如:数据库 URL、数据库驱动等)。一般地,容器部署描述 XML 文件中并没有包含安全角色对实际用户的映射,J2EE 部署描述文件的 DTD 中并没有定义用于安全角色和实际用户的映射元素,这是因为实际环境中有多种不同的用户系统(如关系数据库、系统文件形式和 LDAP 系统等)。安全角色和实际用户的映射方式是由各 J2EE 容器厂商具体指定,因此不同的容器提供商提供的安全域配置方式也有所不同。各容器厂商通常都会提供几种通用的 LoginModule 来实现其用户与角色关系的映射,以及用户身份的验证策略指定。另外,J2EE 容器一般都提供给开发者自己开发 LoginModule 的功能,在 RBAC 模型比较复杂时,可根据 LoginModule 规范来开发自己的 LoginModule。因此,可以针对不同的 RBAC 模型自己开发合适的 LoginModule。

5) 由于 Web 容器采用了 SSO(single-sign on, 一次登录资源尽享)机制,因此并不需用户访问资源时重复输入用户名和密码[2]。

4　应用实例

利用 RBAC 技术和 Oracle9i/BES6.5(Borland Enterprise Server)/J2EE 成功开发了一套 B/S 模式下的 CIMS 安全管理系统。该系统采用三层 RBAC 来管理位于系统 3 个不同应用层(database servers/Web server/browsers)的资源安全。RBAC 用户、角色及其授权信息等都以数据表的形式保存在数据库中。系统采用前台(browsers)集中管理的方式进行,利用 Oracle 数据库触发器和应用程序将前台 RBAC 维护结果同步映射到 Oracle 数据库和 Web 容器中,以充分利用 Oracle 数据库和 BES 的安全机制来实现 RBAC。RBAC 在 BES 中的实现就是通过配置 BES 的部署描述符来完成,通过登录策略、认证策略、安全域、委托策略等的配置来实现容器中的 RBAC。其安全域中的登录模块采用的 Borland 企业应用服务器(BES)提供的 JDBCLoginModule 登录模块。其安全域配置如下[6]:

```
myRealm {
    com.borland.security.provider.authn.JDBCLoginModule required
    //指定登录模块
    DRIVER = oracle.jdbc.driver.OracleDriver
    //指定数据库驱动器
    URL = "jdbc:oracle:thin:@localhost:1521:mydb"
    //指定数据库 URL
    DBTYPE = ORACLE                          //指定数据库类型
    USERNAME = admin PASSWORD = admin;
    //指定数据库登录用户名、密码
};
```

该配置包括:保存用户及其角色信息的 Oracle 数据库的驱动、数据库 URL、以及登录用户名、密码等。对于其他的登录模块其相应的参数将不尽相同。在 JDBCLoginModule 类中,首先进行用户验证,然后将验证通过的用户的授权角色返回给 Web 容器。由于采用了 Oracle 触发器同步机制,系统用户同时也是 Oracle 用户,因此获得用户角色的其代码为[7]:

```
return getUserGroups("select granted_role from dba_role_privs where
    grantee = '" + s.toUpperCase() + "'");
```

其中,s 指当前登录用户。获得角色后的容器会根据 Web 容器部署描符的相应配置来读取相应的安全策略,对访问 Web 资源进行验证,实现对系统 Web 资源的保护。

5　结语

RBAC 技术已经成为当今安全管理的主流技术,如何将其快捷有效地应用在项目中是需要解决的问题之一。如今,很多第三方软件都提供了自己的安全策略,如果能够结合其本身的策略来实现 RBAC 将在很大程度上减少系统开发成本以及提高系统的安全管理效率。利用 J2EE 基于容器的安全机制来实现 RBAC 正是充分结合第三方软件(这里为 Web 容器)安全机制的体现。利用 BES 的部署描述和安全域配置来方便、灵活地实现 RBAC,用 RBAC 来管理系统资源,构建一个安全的系统。

参考文献:

[1] 张东站,薛劲松,宋瀚涛. 基于 I-RBAC 的 CIMS 集成访问控制. 计算机集成制造系统——CIMS,2004,10(1):50 - 54.

[2] PISTOIA M, NAGARATNAM N, KOVED L, *et al*. Enterprise Java Security: Building Secure J2EE Applications[M]. 北京:清华大学出版社,2005.41 - 140.

[3] J2EE安全系统[EB/OL]. http://www.blogjava.net/kapok/archive/2005/05/17/4405.html,2005.

[4] BODOFF S, *et al*. J2EE1.4 标准教材[M]. 第 2 版. 田玉敏,等译. 北京:电子工业出版社,2005.805 - 850.

[5] 陆荣杰,刘知贵,黄晓芳. J2EE 中基于容器管理的 Web 客户端安全验证[J]. 兵工自动化,2005,24(3):47 - 48.

[6] Borland Company. Borland Enterprise Server TM developer guide [M]. version 6.5,2004.

(上接第 1330 页)

[2] 叶阿勇,许力. 移动 Ad Hoc 网络安全策略研究[J]. 微计算机应用, 2004,25(4): 385 - 390.

[3] 许力,钱晓聪. 移动自组网环境下基于图论的智能优化分簇策略[A]. 2005 年中国计算机大会论文集[C]. 北京:清华大学出版社, 2005.10.

[4] BECHLER M, HOF H-J, PAHLKE DK, *et al*. A cluster-based security architecture for ad hoc networks[J]. proc. IEEE. INFOCOM, 2004,4:2393 - 2403.

[5] SEUNGHUN J, CHANIL P, DAESEON C, *et al*. Cluster-Based Trust Evaluation Scheme in an Ad Hoc Network[J]. ETRI journal, 2005,27(4):465 - 468.

[6] LI X-Y, WAMG Y, FRIEDER O. Efficient hybrid key agreement protocol for wireless ad hoc networks[J/OL]. http://www.ls.iit.edu/~xli/,2003.

[7] KIM Y, PERRING A, TSUDIK G. Tree-based Group Key Agreement [J]. ACM Transaction on Information and system security, 2004,7 (1):60 - 96.