第 34 卷 第 7 期 计 算 机 工 程 2008 年 4 月
Vol.34 No.7 Computer Engineering April 2008
·安全技术· 文章编号：1000—3428(2008)07—0137—02 文献标识码：A 中图分类号：N945

一种基于访问控制的安全 Web 服务发现机制

韩 隽[1,2]，淮晓永[1]，赵 琛[1]
(1. 中国科学院软件研究所互联网技术实验室，北京 100080；2. 中国科学院研究生院，北京 100039)

摘 要：当前的 Web 服务发现机制大多依赖集中式的统一描述、发现和集成注册中心，但组织机构出于安全和地域的考虑，倾向于构建私有的分布式注册中心，只有注册且可信的请求者才能浏览到他们有权限访问的服务信息。该文给出 Web 服务发现阶段基于角色的访问控制模型 RBAC4WSD，发现代理依照服务提供者指定的安全策略对请求者实施访问控制，并以跨国公司内部的文档服务为例介绍原型系统的实现。

关键词：Web 服务；发现代理；统一描述、发现和集成；基于角色的访问控制；隐私保护

Secure Web Services Discovery Method Based on Access Control

HAN Jun[1,2], HUAI Xiao-yong[1], ZHAO Chen[1]
(1. Lab for Internet Technology, Institute of Software, Chinese Academy of Sciences, Beijing 100080;
2. Graduate University of Chinese Academy of Sciences, Beijing 100039)

【Abstract】 Most current Web services discovery methods rely on centralized UDDI registries. Due to the security and area, organizations usually build distributed private registries and enforce access control mechanisms. The registered and trusted people can browse the information of services they have permissions to access. A RBAC model for Web services discovery phase named RBAC4WSD is proposed. Discovery agencies are designed to perform access control on service requestors upon security policies specified by service providers. In terms of a scenario of distributed document service within a multinational company, a prototype system is described.

【Key words】 Web services; discovery agency; lUDDI; RBAC; privacy protection

1 概述

Web 服务是当前分布式计算的一个重要组成部分，因其松耦合的特性而被广泛应用于异构环境下的企业服务集成。在服务发现阶段，服务提供者将业务 Web 服务信息发布到发现代理(discovery agency)上，服务请求者查询发现代理以获取服务信息。按照服务信息在发现代理上的存储方式，Web 服务发现可以分为集中式、分布式和混合式。目前，基于统一描述、发现和集成(UDDI)的集中式注册中心已经比较成熟，研究者逐渐把研究兴趣转向分布式环境。在分布式环境中，大部分解决方案属于混合式，部署了多个 UDDI 注册中心，但是较少考虑全分布式，即 Web 服务信息完全不依赖于集中式注册的情形。

分布式环境下的安全、隐私问题得到越来越多的关注，例如，服务提供者不希望将自己的个人信息和服务的调用信息暴露给未经授权的组织或个人。目前，关于Web服务隐私方面的讨论忽略了发现代理，与它相关的安全、隐私问题存在于服务的发布、发现和绑定阶段[1]。本文讨论发现代理如何根据服务提供者在发布Web服务时设定的安全策略，在服务发现阶段对请求者实施访问控制。并针对以上问题，提出一种基于访问控制的安全Web服务发现机制SeWSD (Secure method for Web Services Discovery)，其特点包括：(1)在发现阶段执行访问控制；(2)分布式的服务发布和发现框架；(3)防止恶意的请求者伪造身份。为了验证SeWSD框架的可行性，考虑如下应用场景：某跨国公司需要在公司内部部署文档管理服务(Document Management Service, DMS)，DMS依托P2P、Web服务等技术搭建一个基于P2P网络的文档管理平台。

2 相关工作

文献[2]提到集中式UDDI注册中心的不足，使得更多的研究者倾向于分布式的解决方案。文献[3]提出增强UDDI(exUDDI)并使之组成非结构化对等网络。WS-security规范[4]增强SOAP消息以保证其完整性和机密性。文献[5]提出SRBAC模型，讨论如何在调用阶段进行基于角色的访问控制。

以上解决方案不是基于 UDDI 就是没有考虑发现阶段的安全因素。本文给出了一个全分布式、不依赖集中注册中心的 Web 服务发布和发现框架，并能在服务发现阶段执行细粒度的访问控制。

3 SeWSD 模型

3.1 SeWSD 框架

如图 1 所示，用户登录系统门户(Portal)时须进行身份验证(a)，因此，所有登录成功的用户都是注册过且具有特定组织内角色的、可信的用户。加入访问控制机制后，一个完整的 Web 服务发现过程为：(1)服务提供者发布 Web 服务(b.1, b.2)；(2)服务请求者查询安全 Web 服务发现代理(Secure Web Services Discovery Agency, SWSDA)(c.1~c.3)；(3)SWSDA 执行访问控制，决定是否响应以及响应的内容(c.4)；(4)会话(Session)把查询结果呈现给请求者(c.5)。

基金项目：国家“十五”科技攻关计划基金资助项目(2005BA113A02)
作者简介：韩 隽(1981－)，男，硕士研究生，主研方向：Web 服务，P2P 网络；淮晓永，副研究员、博士；赵 琛，教授、博士
收稿日期：2007-05-09 **E-mail：**jun.han.iscas@gmail.com

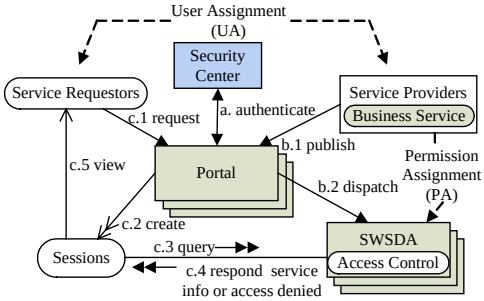

图 1　SeWSD 框架

SeWSD 框架包含：(1)服务请求者(Service Requestors)，登录 Portal 查询 Web 服务；(2)安全中心(Security Center)，执行角色指派(UA)操作，为组织内部所有在其上注册的用户分配角色；(3)服务提供者(Service Providers)，登录 Portal，发布 Web 服务，定义安全访问策略，进行权限指派(PA)；(4)Portal，验证用户身份，提供服务发布和查询的入口；(5)Sessions，由 Portal 动态创建，包含请求者的角色信息，查询活动的 SWSDA，把结果通过网页形式显示给请求者；(6)SWSDA，接收并存储 Portal 发送的服务信息和服务提供者设定的访问控制策略；对 Session 实施基于角色的访问控制。

在组织内部，安全中心对用户和角色信息采取集中的统一管理是为了避免部署多套用户系统而造成用户信息的不一致，减少系统部署复杂度和用户信息的维护成本。

3.2　基于角色的访问控制(RBAC)模型

本文参考经典 RBAC 模型，给出服务发现阶段基于角色的访问控制模型——RBAC4WSD(RBAC for Web Services Discovery)。设定操作集包含浏览 Web 服务信息的操作，因为本文不讨论 Web 服务绑定和调用，所以未加入服务绑定调用操作。为了实施细粒度的访问控制，通过属性集来区分服务的基本信息和接口信息。RBAC4WSD 的定义如下：

$RBAC4WSD=\{U, R, S, O, SA, P, WS, PA, UA, user, roles, C\}$

其中，U,R,WS分别代表用户集Users、角色集Roles、候选Web服务集Web Services；O表示Operations操作集，$O=\{browse\}$，$browse$表示浏览Web服务信息；SA表示Service Attributes服务属性集，$SA=\{svcInfo\}\cup I$，$svcInfo$为基本信息，I为服务的接口集，且$I=\{i_1, i_2, \cdots, i_n\}$，$i_j(1\le j\le n)$为不同接口；$P$为Permissions权限集，$P\subseteq O\times SA$，建立操作与服务属性的多对多关系；$C$为Constraints约束集，存在约束：

$$o_k(i_j)\Rightarrow o(svcInfo), o_k\in O, i_j\in I$$

即如果具有操作接口i_j的权限，那么也具有操作$svcInfo$的权限；S表示Sessions会话，把用户映射到角色，在用户激活角色时生成；$PA: PA\subseteq R\times P$表示角色权限分配，建立角色与权限的多对多关系；$UA: UA\subseteq U\times R$表示用户角色分配，建立用户与角色的多对多关系；$user: S\rightarrow U$表示函数$user(s_i)$，把每个会话$s_i$映射到唯一的用户；$roles: S\rightarrow 2^R$表示函数$roles(s_i)$，把每个会话$s_i$映射到一个角色集。

在实际应用中，Session 相当于用户代理，包含用户身份和角色间的映射关系，由三元组<*user, role, time_contraint*>表示，即在时间段内 *user* 具有 *role*。*RBAC4WSD* 中的权限指派 *PA* 由三元组<*r, opt, sa*>表示，即角色 *r* 可以对服务属性 *sa* 执行 *opt*，由此可以得出针对该 Web 服务的访问控制矩阵。

3.3　安全的 Web 服务发现流程

3.3.1　Web 服务发布

发布 Web 服务的步骤为图 1 中的 b.1 和 b.2，最后由 SWSDA 接收并存储服务配置。

服务配置 *SvcConf* 包括业务 Web 服务的基本信息 *wsInfo* 和访问控制规则两部分。*wsInfo* 包括<*Name, Provider, Desc, WsdlURL*>，分别代表服务名、服务提供者、服务描述和 WSDL 文档的 URL。访问控制规则即 *RBAC4WSD* 的 *PA* 部分。

DMS 系统包含 3 类服务：文档下载，更新和删除。组织内部包含 3 个角色：成员，组长和经理。在发布文档更新服务时，设定安全规则如下：(1)允许组员浏览服务的基本信息；(2)允许组长和经理查看接口 *updateDoc* 的信息。文档更新服务的 XML 描述如下：

```
<SvcConf>
  <wsInfo>
    <Name>DocumentUpdateService</Name>
    <Provider>www.foo.com</Provider>
    <Desc>update document</Desc>
    <WsdlURL>http://www.foo.com/ws/dms/update?wsdl
    </WsdlURL>
  </wsInfo>
<constraint r="member" opt="browse" sa="svcInfo" />
<constraint r="leader" opt="browse" sa="updateDoc" />
<constraint r="manager" opt="browse" sa="updateDoc" />
</SvcConf>
```

对应的访问控制矩阵如图 2 所示。

r \ opt, sa	*svcInfo*	*updateDoc*
member	*browse*	
leader	*browse*	*browse*
manager	*browse*	*browse*

图 2　文档更新服务的访问控制矩阵

3.3.2　Web 服务查询

本文使用 Portal 来防止请求者伪造身份，因为桌面应用程序本身是不可信的，很容易被修改并滥用。在请求者登录时，Portal 验证请求者身份，动态生成 Session，并由 Session 把执行结果呈现给请求者。在浏览器端，服务请求者只能发出查询请求和查看结果。

Portal 的优点在于：(1)请求者无法截获系统中的通信和伪造身份；(2)请求者无法发起对服务提供者的攻击，例如中间人(man-in-the-middle)或者重放(replay/playback)攻击；(3)为 Session 加上时间戳，高权限用户不慎泄漏的用户名和密码只会在一定时间内有效，把危害降到最低。

3.3.3　响应服务查询请求

如图 1 的 c.4 所示，SWSDA 在接收到查询请求后，首先从中抽取请求者角色信息，然后根据访问控制矩阵决定是否响应请求者。

算法　发现代理 SWSDA 响应 Session 发出的查询请求

输入：Session 发出的查询请求

输出：响应的服务信息

(1)检查时间戳是否在允许的时段内；

(2)获取 Session 中的请求者角色；

(3)依照角色信息查找访问控制矩阵，决定是否响应，以及响应什么内容；

(4)如果是 *svcInfo*，向 Session 返回服务基本信息，包括<*Name, Provider, Desc*>；

(5)如果是$interface_i$，除了向Session返回服务基本信息外，还返回接口访问信息；

（下转第 141 页）