[19] 中华人民共和国国家知识产权局

[51] Int. Cl.
H04L 9/00 (2006.01)

[12] 发明专利说明书

专利号 ZL 02814820.7

[45] 授权公告日 2008 年 1 月 9 日

[11] 授权公告号 CN 100361435C

[22] 申请日 2002.5.29 [21] 申请号 02814820.7

[30] 优先权

[32] 2001. 6.11 [33] US [31] 09/878,536

[86] 国际申请 PCT/US2002/016644 2002.5.29

[87] 国际公布 WO2002/101973 英 2002.12.19

[85] 进入国家阶段日期 2004.1.29

[73] 专利权人 BEA 系统公司

地址 美国加利福尼亚州

[72] 发明人 保罗·帕特里克

[56] 参考文献

US5802276 1998.9.1

US1253438A 2000.5.17

US5925126 1999.7.20

审查员 刘剑波

[74] 专利代理机构 北京市柳沈律师事务所

代理人 马 莹 邵亚丽

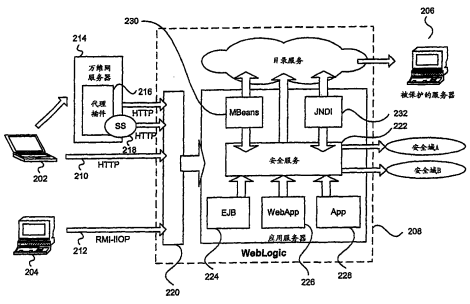
权利要求书 5 页 说明书 23 页 附图 10 页

[54] 发明名称

用于服务器安全和权限处理的系统和方法

[57] 摘要

一种可插入的体系结构允许安全和商业逻辑软插件被插入到由服务器(206)作为主机接纳的安全服务中，并且控制对于在那个服务器上、在安全域(208)内的另一个服务器上或在安全域之间的一个或多个所保护资源的访问。 安全服务(222)可以作为用于安全实施、访问权利确定的焦点。 并且在一个登录处理中使用或确定的信息可以透明地和自动地流到其他的登录处理。 权限表示在特定的上下文中特定的用户(204)可以或不可以对于特定的资源(226)做什么。 权限不仅反映安全环境的技术方面(许可或拒绝概念)，而且可以用于表示由服务提供者要求的商业逻辑或功能。 以这种方式，权限填补了简单安全平台和复杂商业策略平台之间的空档(gap)。



1. 一种安全系统，用于使得客户可以通过应用容器访问所保护的资源，所述安全系统包括:

应用容器，为所保护的资源提供服务，其中当应用容器从客户接收对于所保护资源的访问请求时，所述应用容器通过将访问请求和回调处理器传递到安全模块，向安全模块委派授权决定;

所述安全模块，用于作出许可或拒绝所述访问请求的决定，其中所述安全模块包括多个可以被插入到所述安全模块的安全提供者，并且其中所述多个安全提供者利用回调处理器来从所述应用容器为所述访问请求而请求上下文信息，并且其中根据来自每个安全提供者的输出，所述安全模块确定客户使用所保护的资源的权限，其中上下文信息包括一个或多个描述访问请求的参数值，并且可以通过安全模块利用回调处理器从应用容器检索该上下文信息;

所述安全模块位于第一计算机，并且所述所保护的资源位于相同的第一计算机或者第二计算机; 以及

资源接口，用于向所述所保护的资源通告被许可的访问请求。

2. 按照权利要求 1 的安全系统，其中应用容器读出应用配置描述，并且将所述应用配置描述登记到所述安全模块内。

3. 按照权利要求 2 的安全系统，其中所述应用容器是由 ENTERPRISE JAVABEANS 规范定义的 ENTERPRISE JAVABEANS 容器。

4. 按照权利要求 2 的安全系统，其中所述应用容器是万维网应用容器。

5. 按照权利要求 1 的安全系统，其中所述安全模块包括多个访问决定模块，用于定义访问策略，和多个访问决定模块的每个可以确定其自身的有贡献的决定以允许、拒绝或放弃所述访问请求。

6. 按照权利要求 5 的安全系统，其中所述安全模块还包括访问控制器，用于向所述多个访问决定模块传送所述访问请求，并且用于通过所述安全模块来将所述有贡献的决定组合为一个整体决定，以允许或拒绝所述访问请求。

7. 按照权利要求 5 的安全系统，其中所述访问决定模块的一个或多个表示与商业功能相关联的访问策略。

8. 按照权利要求 5 的安全系统，其中访问决定可以被加到安全模块以反

映在所述访问策略中的变化。

9. 按照权利要求 5 的安全系统，其中所述多个访问决定模块用于定义所述客户访问所述所保护资源的权限。

10. 按照权利要求 5 的安全系统，其中由所述多个访问决定模块的任何一个的拒绝或放弃使得安全模块拒绝所述访问请求。

11. 按照权利要求 5 的安全系统，其中由所述多个访问决定模块的任何一个的放弃不使得安全模块拒绝所述访问请求。

12. 按照权利要求 5 的安全系统，其中所述安全模块还包括审查模块，用于审查所述多个访问请求的确定。

13. 按照权利要求 1 的安全系统，其中所述资源接口包括用于向所保护资源或从所保护资源传送访问请求的接口模块。

14. 按照权利要求 13 的安全系统，其中所述接口模块包括由 J2EE 规范定义的 J2EE 安全接口。

15. 按照权利要求 13 的安全系统，其中所述接口模块包括安全提供者接口。

16. 按照权利要求 13 的安全系统，其中所述接口模块被作为插件包括在所述资源接口中。

17. 按照权利要求 1 的安全系统，其中所述安全模块还确定是否允许或拒绝从所述所保护资源向所述客户的、对于所述访问请求的响应。

18. 按照权利要求 1 的安全系统，其中，权限包括商业逻辑权限和功能权限的至少一个。

19. 按照权利要求 1 的安全系统，其中，上下文信息包括下述的至少一个：所保护资源的身份、访问请求参数的一个或多个值和客户的网络或因特网协议地址。

20. 一种允许客户通过应用容器访问所保护资源的方法，所述方法包括：

在向其包含的资源提供服务的应用容器从所述客户接收对于访问所保护资源的访问请求；

将所述访问请求从所述应用容器与所述访问请求和回调处理器一起通告给安全模块，其中当应用容器从客户接收对于所保护资源的访问请求时，所述应用容器通过将访问请求和回调处理器传送到安全模块，向安全模块委派授权决定；

在所述安全模块作出决定，以允许或拒绝所述访问请求，其中所述安全模块包括可以被插入到所述安全模块的多个安全提供者，

在每个安全提供者使用所述回调处理器来对于所述访问请求从所述应用容器请求上下文信息，其中上下文信息包括一个或多个描述访问请求的参数值，并且可以通过安全模块利用回调处理器从应用容器检索该上下文信息；

根据来自每个安全提供者的输出利用上下文信息来确定客户使用所述所保护资源的权限；并且

向所述所保护资源通过资源接口通告被许可的访问请求。

21. 按照权利要求 20 的方法，其中所述应用容器读出应用配置描述，并且将所述配置描述登记到所述安全模块内。

22. 按照权利要求 21 的方法，其中所述应用容器是由 ENTERPRISE JAVABEANS 规范定义的 ENTERPRISE JAVABEANS 容器。

23. 按照权利要求 21 的方法，其中所述应用容器是万维网应用容器。

24. 按照权利要求 20 的方法，还包括：

经由在所述安全模块的多个访问决定模块来定义访问策略；

在每个访问决定模块确定有贡献的决定，以允许、拒绝或放弃所述访问请求。

25. 按照权利要求 24 的方法，还包括：

经由一个访问控制器来向所述多个访问决定模块传送所述访问请求，并且通过安全模块来将所述有贡献的决定组合为一个整体决定以允许或拒绝所述访问请求。

26. 按照权利要求 24 的方法，其中所述多个访问决定模块的一个或多个表示与商业功能相关联的访问策略。

27. 按照权利要求 24 的方法，其中访问决定可以被加到安全模块以反映在所述访问策略中的变化。

28. 按照权利要求 24 的方法，还包括：

使用所述多个访问决定模块来定义所述客户访问所述所保护资源的权限。

29. 按照权利要求 24 的方法，其中由所述多个访问决定模块的任何一个的拒绝或放弃使得安全模块拒绝所述访问请求。

30. 按照权利要求 24 的方法，其中由所述多个访问决定模块的任何一个

的放弃不使得安全模块拒绝所述访问请求。

31. 按照权利要求 24 的方法，还包括:

经由一个审查模块来审查多个访问决定模块的确定。

32. 按照权利要求 24 的方法，其中所述通告所述访问请求的步骤包括:经由一个接口模块向所保护资源或从所保护资源传送访问请求。

33. 按照权利要求 32 的方法，其中所述接口模块包括由 J2EE 规范定义的 J2EE 安全接口。

34. 按照权利要求 32 的方法，其中所述接口模块包括安全提供者接口。

35. 按照权利要求 32 的方法，其中所述接口模块被作为插件包括在所述资源接口中。

36. 按照权利要求 20 的方法，还包括:

确定是否允许或拒绝从所述所保护资源向所述客户的对于所述访问请求的响应。

37. 一种用于确定在安全环境中访问所保护资源的用户权限的方法，包括:

通过使用访问请求和回调来调用安全模块，从用户应用接收对于访问一个所保护资源的访问请求;

利用上下文信息来确定访问所述所保护资源的用户权限，其中，所述确定包括轮询可以被插入所述安全模块的多个安全提供者，并且其中，所述多个安全提供者使用回调处理器来对于所述访问请求从应用容器请求上下文信息;

根据用户权限来在安全模块进行确定以许可或拒绝所述访问请求；并且

进行下述任何一个步骤:

（a）向所述所保护资源通告一个被允许的访问请求，或

（b）拒绝对于所述所保护资源的一个被拒绝的访问请求。

38. 按照权利要 37 的方法，其中，如果许可所述访问请求，则用户权限也确定所述所保护资源的用户可以获得的访问的类型。

39. 按照权利要求 38 的方法，其中所述访问类型包括查看、修改、删除或复制所述所保护资源的任何部分或全部的其中一个。

40. 按照权利要求 37 的方法，其中可以从第一安全域向第二安全域通告关于所述用户权限的信息。

41. 按照权利要求 40 的方法，其中，在从所述第一安全域向所述第二安全域通告关于所述用户权限的所述信息之前，可以使用来自第一安全域的附加信息来修改用户权限。

42. 一种允许客户应用对被保护资源进行访问的方法，所述应用与容器相关联，所述方法包括:

当所述应用寻求访问被保护资源时，通过所述应用来调用所述容器;

通过所述容器来调用安全服务;

通过在所述安全服务内的输出访问控制器而相对于所述应用的访问请求来确定要发生哪些访问决定;

轮询所选择的访问决定插件，以便确定每个访问决定插件的贡献，每个访问决定的输出是拒绝、许可或放弃;

通过审查机制来跟踪每个访问决定的结果;

由所述应用容器根据访问决定的输出来访问所保护的资源;

从所述应用向所述所保护的资源或从所述所保护的资源向所述应用传送数据。

43. 按照权利要求 42 的方法，其中，所述容器是企业 Java Bean 容器。

44. 按照权利要求 42 的方法，其中，所述容器是万维网应用容器。

用于服务器安全和权限处理的系统和方法

本申请要求 2001 年 6 月 11 日提交的、题目为“用于服务器安全和权限处理的系统和方法”的美国专利申请第 09/878,536 号的优先权，其申请以引用方式被包含在此。

技术领域

本发明一般地涉及服务器安全机制，具体涉及用于服务器安全的体系结构。

背景技术

这些年来，用于服务器技术、特别是电子商务服务器的术语“安全”已经扩展为覆盖安全环境的几个方面。这个术语的原始定义总是涉及一种机制，用于验证特定用户（或客户）的身份，并且防止用户检索、查看或否则访问不允许访问的、在服务器上的信息。

图 1 示出了今天通常使用的安全机制。客户 102、104 经由多个协议来访问诸如应用服务器 106 的安全资源，所述协议包括例如 http（超文本传送协议）110 和 IIOP（因特网 inter-ORB（交互对象请求代理）协议）112。通过安全层 108 来过滤访问请求，安全层 108 通常根据一组硬连线的规则来确定是否允许或不允许所请求的访问。安全层可以是所保护资源的一个组成部分（例如应用服务器本身），或者可以作为独立的实体（例如作为防火墙系统或器件的部件）。

当前的系统很少或不分析访问请求的特征、客户的类型或被保护的资源的类型。因而，安全经常被看作简单的“允许或拒绝”机制，它被设计使得明白由服务器提供的所保护资源，并且向那些资源附加预定的一组管理用户或客户的访问权的规则。很少或不明白进行请求的方式或其中特定用户可以请求访问资源的环境设置，并且安全机制不有助于容易地修改所述规则以反映在关于安全的商业策略中的新变化。

已经做出努力来发展安全、尤其是服务器安全的概念，以包括反映用户的特定环境的信息和以短语表示访问特定的所保护资源的请求的方式。类似地，已经做出努力来提供可以容易地进行修改以反映商业策略、安全方针和访问权中的变化的安全机制。这些方法通常仍然需要应用编程者负责将一组商业策略查询组合在一起并且问问题："这样好吗？—— 这个访问按照安全规则是可以接受的吗？"。所需要的是这样一种机制：它使得这个过程包括全部，以便编程者不必明白商业语义和商业策略规则而能够开发咨询安全相关的商业策略问题的程序和应用。

近来，一组新的涉及所保护资源的要求已经通过与应用服务器客户和系统集成者的讨论而出现。这些新的要求从应用而不是基础结构的观点以安全的形式被表达。在这些新要求内的关键区别是存在商业策略实施。通常以用户根据他们充当的角色和商业请求的上下文被"授权"执行的行为的形式来描述商业策略的实施。客户和系统集成者对于需要他们将实施商业策略的代码嵌入到应用程序中的事实感到沮丧。嵌入这种逻辑产生了配置问题：每次改变商业策略时必须修改、测试和重新配置应用程序。根据商业策略改变的频率，当前的用于修改和重新配置的要求是不可接受的。

客户和系统集成者理想上希望将这些新的授权能力普遍地应用到不同类型的执行和资源容器，例如包括用于企业 JavaBeans 组件模型（EJB）、万维网应用程序（小服务程序，JSP）以及其他类型的商业和资源容器的那些新的授权能力。基于角色的访问控制（RBAC）正在变为由客户和系统集成者请求的授权的主要形式之一。在角色使用中的趋势使得当进行授权决定时组织身份被用作身份的抽象形式。角色的使用提供了一种降低在应用安全的管理中的成本和误差的机制，因为它简化了管理量。

Java 2 企业版（J2EE）定义了基于角色的访问控制的说明形式。但是，因为它基于说明性方案，因此角色与主方的联系是静态的。另外，当前的 Java 2 企业版规范未提供任何手段，通过该手段，当确定与给定的主方相关联的角色时可以考虑诸如请求的参数或目标的身份的商业请求上下文。结果，需要应用程序开发者在应用程序中实现商业策略规则以计算动态角色关联，以便支持诸如"拥有者"的概念。

客户和系统集成者现今要求比由 Java 2 安全所定义的基于许可的安全和在 Java 2 企业版中定义的基于角色的访问控制的那些授权能力更丰富的一组

授权能力。可以在诸如访问控制列表的传统授权机制和商业策略实施的交集中找到用于这个更丰富级的授权的要求的基础；所述基础包括当进行授权决定时考虑到商业请求的上下文的能力。所述请求上下文可以包括目标对象的身份、请求的参数的值、诸如网络或启动客户的 IP 地址的可能的环境信息。

对于通过其来集成这些新的授权能力而与执行或资源容器类型无关的单独机制的缺少是客户和系统集成者的沮丧点。服务提供者接口（SPI）是由几个应用服务器使用的机制，以便允许与外部授权提供者的集成，其中所述几个应用服务器包括加利福尼亚 San Jose 的 BEA 系统公司的 WebLogic 服务器产品。这种 SPI 领域具有多个限制，这些限制限制了它被用作集成第三方的授权机制的成功手段或由客户和系统集成者要求的新授权能力的能力。

对于当前“领域”SPI 的最大的限制之一是实施的范围。当前领域机制的实施范围不包括诸如企业 JavaBeans 组件模型和万维网应用程序的资源。具体上，所述领域着重于 RMI（远程方法调用）型资源，诸如 JMS（Java 消息传递服务）目的地、在 JNDI（Java 命名的目录接口）中的输入项和在粗略等级(course-grained level)上的小服务程序。虽然所述领域可以想象地被更新以保存支持这种资源的保护所需要的授权策略的定义，但是其他的限制是最后使得所述领域成为处理所有的授权要求的不切实际的机制。

当前的领域 SPI 的第二个限制是实施点。当前领域机制不允许在所述领域本身中存在进行决定以允许访问所保护资源的点。相反，实施点在应用服务器本身中。因为这种处理，当前的领域被定义为仅仅支持基于许可的授权机制，其中所述领域仅仅作为访问控制列表的数据库。多数厂商也不能接受这样的复杂性：提供也可以支持 Java 2 砂盒（运行程序安全区）规则的一个 Java 2 策略对象。

另一个限制是当前领域 SPI 的实施机制支持。如同实施点的情况一样，由当前领域 SPI 允许的实施机制限于以基于许可的授权机制为基础的那些。这与主导的第三方授权提供者的能力相反，其集成被客户和系统集成者以日为基础来请求。

这些限制一起约束了可以与企业应用服务器集成而不最小化提供者的价值问题的授权提供者的类型。不存在获得请求上下文以便提供丰富的被请求的授权的当前能力。

JAVA、JAVA TWO ENTERPRISE EDITION (J2EE)、和 ENTERPRISE

JAVABEANS (EJB)是 Sun Microsystems 公司的商标。WEBLOGIC 是 BEA 系统公司的商标。

发明内容

本发明一般地涉及一种服务器安全机制，具体涉及提供服务器安全和权限处理的体系结构。一种可插入的体系结构允许安全和商业逻辑软插件被插入到由服务器作为主机接纳的安全服务中，并且控制对于在那个服务器上、在安全域内的另一个服务器上或在安全域之间的一个或多个所保护资源的访问。安全服务可以作为用于安全实施、访问权利确定的焦点，并且在一个登录处理中使用或确定的信息可以透明地和自动地流到其他的登录处理。

本发明也引入了用于访问上下文内的权限的概念。在本申请的上下文中使用的“用户”或“客户”可以指同一事物——自然人、在自然人控制下的硬件装置或软件应用程序、或者在自动控制下运行而不需要用户干预的硬件器件或软件应用程序。用户（客户）是试图访问在服务器上的所保护资源的实体。这个所保护资源可以例如是运行在服务器上的软件应用程序、网站的特定的网页或一部分、或者数据库等。

当用户试图访问资源时，安全服务可以确定访问请求的类型、目的地（所保护资源）和其中进行请求的设置——以下称为访问上下文或简称为上下文。从这个信息，安全服务可以确定用于用户的一个“权限”或一组权限。权限清楚地表示在特定的上下文中特定的用户可以或不可以对于特定的资源做什么。权限不仅反映安全环境的技术方面（许可或拒绝概念），而且可以用于表示由服务器提供者要求的商业逻辑或功能。以这种方式，权限填补了简单安全平台和复杂商业策略平台之间的空档。

为了说明所述能力，考虑下面的商业示例：

对于问题“史密斯医生能够更新病人的医疗图表吗”的回答依赖于问出问题的上下文。在基于许可的授权系统中，这个上下文是不存在的，因为资源是‘医疗图表’对象的一些实例，所述请求是要‘更新’，所述主方是‘史密斯先生’。结果，如果所提供的回答是‘是’，则史密斯医生可以更新任何病人的医疗图表。在基于能力的授权系统中，有可能加入谁是所述病人的必要上下文。因此，现在问题重新被改述为“史密斯医生能够更新 Jon Joe 的医疗图表吗？”。在确定对这个问题的回答时，我们现在需要知道是否史密斯医

生是 Jon Joe 的个人医生或在医疗中心的主治医生。使用简单的循环，我们可以将所述概念表示如下：

在一个实施例中，本发明包括一种安全系统，用于使得客户可以通过应用容器访问所保护的资源，所述安全系统包括：应用容器，为所保护的资源提供服务，其中当应用容器从客户接收对于所保护资源的访问请求时，所述应用容器通过将访问请求和回调处理器传递到安全模块，向安全模块委派授权决定；所述安全模块，用于作出许可或拒绝所述访问请求的决定，其中所述安全模块包括多个可以被插入到所述安全模块的安全提供者，并且其中所述多个安全提供者利用回调处理器来从所述应用容器为所述访问请求而请求上下文信息，并且其中根据来自每个安全提供者的输出，所述安全模块确定客户使用所保护的资源的权限，其中上下文信息包括一个或多个描述访问请求的参数值，并且可以通过安全模块利用回调处理器从应用容器检索该上下文信息；所述安全模块位于第一计算机，并且所述所保护的资源位于相同的第一计算机或者第二计算机；以及资源接口，用于向所述所保护的资源通告被许可的访问请求。

在另一个实施例中，本发明包括一种允许客户通过应用容器访问所保护资源的方法，所述方法包括：在向其包含的资源提供服务的应用容器从所述客户接收对于访问所保护资源的访问请求；将所述访问请求从所述应用容器与所述访问请求和回调处理器一起通告给安全模块，其中当应用容器从客户接收对于所保护资源的访问请求时，所述应用容器通过将访问请求和回调处理器传送到安全模块，向安全模块委派授权决定；在所述安全模块作出决定，以允许或拒绝所述访问请求，其中所述安全模块包括可以被插入到所述安全模块的多个安全提供者，在每个安全提供者使用所述回调处理器来对于所述访问请求从所述应用容器请求上下文信息，其中上下文信息包括一个或多个描述访问请求的参数值，并且可以通过安全模块利用回调处理器从应用容器检索该上下文信息；根据来自每个安全提供者的输出利用上下文信息来确定客户使用所述所保护资源的权限；并且向所述所保护资源通过资源接口通告被许可的访问请求。

本发明还包括一种用于确定在安全环境中访问所保护资源的用户权限的方法，包括：通过使用访问请求和回调来调用安全模块，从用户应用接收对于访问一个所保护资源的访问请求；利用上下文信息来确定访问所述所保护

资源的用户权限，其中，所述确定包括轮询可以被插入所述安全模块的多个安全提供者，并且其中，所述多个安全提供者使用回调处理器来对于所述访问请求从应用容器请求上下文信息；根据用户权限来在安全模块进行确定以许可或拒绝所述访问请求；并且进行下述任何一个步骤：向所述所保护资源通告一个被允许的访问请求，或拒绝对于所述所保护资源的一个被拒绝的访问请求。

本发明还包括一种允许客户应用对被保护资源进行访问的方法，所述应用与容器相关联，所述方法包括：当所述应用寻求访问被保护资源时，通过所述应用来调用所述容器；通过所述容器来调用安全服务；通过在所述安全服务内的输出访问控制器而相对于所述应用的访问请求来确定要发生哪些访问决定；轮询所选择的访问决定插件，以便确定每个访问决定插件的贡献，每个访问决定的输出是拒绝、许可或放弃；通过审查机制来跟踪每个访问决定的结果；由所述应用容器根据访问决定的输出来访问所保护的资源；从所述应用向所述所保护的资源或从所述所保护的资源向所述应用传送数据。

附图说明

图 1 示出了按照现有技术的客户端/服务器结构的图解。

图 2 示出了按照本发明的客户端/服务器结构的图解。

图 3 示出了按照本发明的安全服务的图解。

图 4 示出了按照本发明的由安全服务使用的方法的流程图。

图 5 示出了按照本发明的服务器安全配置工具的图解。

图 6 示出了按照本发明的由服务器安全配置工具使用的方法的流程图。

图 7 示出了按照本发明的安全服务和所保护资源的图解；

图 8 示出了按照本发明的由安全服务使用来允许访问所保护资源的方法的流程图。

图 9 示出了按照本发明一个实施例的安全层的图解。

图 10 示出了按照本发明的一个实施例的授权机制的图解。

图 11 示出了按照本发明的在几个安全域模型上的登录的图解。

具体实施方式

本发明的一个实施例包括一个安全结构，它提供服务器安全和授权处理，允许向由服务器作为主机接纳的安全服务中插入安全和商业逻辑软插件，并且可以用于控制对于在那个服务器上、在安全域内的另一个服务器上或在安全域之间的一个或多个所保护资源的访问。安全服务作为用于安全实施、访问权利确定的焦点。并且在一个登录处理中使用的信息可以自动地流到其他的登录处理，允许单一签字或安全实施。

除了下述的新术语，在本文件中使用的术语与在关于 Java、企业 JavaBeans 组件模型、WebLogic 服务器的标准文本中定义的术语和其他通常接受的安全概念一致。

访问控制 – 对资源的访问的限制以防止其未经授权的使用。

访问控制信息（ACI） – 关于资源访问请求的发起者的信息，用于进行访问控制实施确定。

访问控制列表（ACL） – 被授权访问资源的实体以及它们的访问权的列表。

授权 – 授予权限，包括根据访问权来许可访问。

证书 – 描述用户或其他主方的安全属性（身份和/或特权）。证书通过验证或授权（delegation）而被要求，并且由访问控制使用。

授权（delegation） – 一个用户或主方授权另一个使用他的（或她的或它的）身份或特权的行为，这种使用可能带有限制。

权限 – 授予一个主方的权利或许可。

身份 – 具有唯一特性的安全属性；没有两个主方的身份可以是相同的。主方可以具有几种不同的身份，每个是唯一的。

主方 – 用户或编程实体，具有使用系统的资源的能力。

角色 – 组织身份，它限定对于被授权用户的一组可以允许的行为。

基于角色的访问控制（RBAC） – 一类安全机制，它通过被称为角色的组织身份来协调对于资源的访问。

安全属性 – 一个主体（用户或主方）的特性，其形成管理那个主体的系统策略的基础。

安全策略 – 限定系统的安全访问必须提供什么保护的数据。存在许多的安全策略，包括访问控制策略、审查策略等。

访问提供者接口（SPI） – 支持服务的子集的具体实现的一个程序包或

一组程序包。

目标 – 在授权“调用链”中的最后接收方。唯一的不是调用的始发者的、在调用链中的参与者。

目标对象 – 商业请求消息的接收方。

信任模型 – 关于下列的说明：如果系统保持安全，则必须信任系统的那些部件和在系统外的那些实体以及因为什么必须信任它们。

可信计算基础(TCB) – 必须正确地运行以便系统保持安全的部分。TCB最好是防止篡改的，并且其策略的实施应当不能被绕过。

未经授权的主方 – 还没有验证任何身份或特权的用户或其他主方。

用户 – 一个人，它使用系统来向对象发出请求以便使得它们在系统中代表自己执行功能。

如在上面的段落中所述的，“本文件中所使用的术语是与有关 JAVA、ENTERPRISE JAVABEANS、WEBLOGIC Server 的标准文本中定义的术语以及其他通用可接受的安全概念一致的”。诸如应用容器（Application Container）、上下文信息(Context Information)、配置标识符(Deployment Descriptors)、J2EE 安全接口(J2EE Security Interface)等的术语定义于 J2EE 规范中（http://java.sun.com/j2ee/j2ee-1_4-fr-spec.pdf）。诸如 EJB 容器的术语定义于 EJB 规范中(http://jcp.org/aboutJava/communityprocess/final/jsr220/index.html)。诸如调回、接口等的术语定义于 JAVA Tutorial 中(http://java.sun.com/docs/books/tutorial/)。

在此所述的本发明的实施例可以确定在访问上下文中使用的权限。通常，用户试图访问在服务器上的所保护资源。这个所保护资源可以例如是在服务器上运行的软件应用程序、网站的特定的网页或一部分、或者数据库系统。也可以使用其他类型的所保护资源，这仍然保持在本发明的精神和范围内。当用户试图访问资源时，所述安全服务确定访问请求的类型、目的地（所保护资源）和作出请求的上下文。从这个信息，安全服务可以确定用于用户的一个权限或一组权限。权限清楚地表示在特定的上下文中特定的用户可以对特定的资源做什么。可以使用权限来表示由服务器提供者要求的商业逻辑或功能，因此填补了简单安全平台和复杂商业策略平台之间的空档。

本发明的实施例另外可以允许集成第三方厂商安全产品以提供对于下述应用类型的保护：

- 企业 JavaBeans 组件模型
- 万维网应用程序（小服务程序，Java 服务器网页（JSP））
- 资源（远程方法调用（RMI），Java 消息传递服务（JMS））

本发明的实施例也可以允许集成外部公共密钥基础结构以支持先进的特征，诸如：

- 证书和密钥检索
- 证书验证
- 撤回机制（证书革新列表（CRL），在线证书状态协议（OCSP））

在开发本发明中的一个重要目标是包括对于 Java 2 企业版（J2EE）规范和与其的互用性的支持。这些 J2EE 规范的特征包括公共安全互用性（CSI）协议、用户身份权标、无状态验证服务（SAS）协议、对于安全证书通过机器、簇和/或域边界的传播的支持、对于基于策略的身份的传播的控制、对于虚拟主机/站点的增强的支持、产生范围到达域的用户身份的能力、主机/站点具体安全策略。

本发明的另一个重要目标是包括对于下述特征的支持：诸如更新的 SSL 能力、硬件加速器、传输层安全（TLS）协议、话路恢复、证书当局、网络连接的保护、万维网服务器代理、服务器到服务器的通信、话路标识符的加密和与目录服务器的增强的集成。

安全域

在本发明的上下文中使用的安全域可以跨越单个服务器、多个服务器、一簇服务器或管理域。本发明允许管理控制在安全域内、通过安全域和在安全域之间的身份传播。客户端应用程序或用户的身份可以由安全域传播和划定范围。例如：

Principal-Identifity@security-domain-name（主方身份@安全域名称）

当禁止传播时，启动身份可以被表示为匿名的。例如：

<anonymous>@initiating-security-domain-name（<匿名>@启动安全域名称）

框架

由本发明提供的安全结构或框架意欲处理所提出的要求和许多其他事情。通过完全聚焦于授权要求，所述安全结构试图除了典型的基于许可的授权之外还提供下列能力：

动态角色关联

动态角色关联是允许角色的关联与主方的后结合，它能够在确定主方被授权为的角色期间考虑到请求的上下文。动态角色关联可以被当作在运行时间主方与角色的后结合。这个后结合仅仅在对于所保护资源的授权决定之前发生，而与主方到角色的关联是否被静态地限定或动态地计算无关。因为其在调用序列中的布局，可以将任何主方到角色的关联的结果当作身份来作为任何授权决定的一部分，所述任何授权决定作为这个请求的一部分被做出。

不像静态限定的关联一样，可以在运行时间动态地计算主方到角色的关联。动态角色的计算能够访问构成请求的上下文的多段信息，包括目标的身份（如果可以获得的话）、请求的参数的值、与启动主方相关联的简要属性的值、以及可能的其他信息。

上下文信息通常被用作在要由规则或表达式评价引擎评价的一个表达式中的参数值。依赖于由所述评价引擎提供的功能，也可能表达调出到执行外部计算的外部类的能力，所述外部计算诸如由主方经过的英里数量，所述外

部计算随后被用作所述表达式的参数。

除了计算应当动态地与主体中的主方相关联的任何角色之外，这个同一功能也负责关联通过消耗 J2EE 限定的配置标识符或通过管理员来静态限定的任何角色。

这个能力的结果是在一个实施例中更新的 JAAS 主体，在给定上下文和目标的情况下，它包括在主体中包括的主方被授权接收的任何角色。这些角色随后可以由对于所保护资源的授权决定以及商业容器和应用程序代码获得。例如，个性化服务器、应用服务器或万维网服务器可以利用标准方法来确定是否由主体担当一个特定的角色来作为个性化网页的手段。这个同一机制可以被企业 JavaBeans 组件模型或其他应用程序使用，以确定是否从在数据库中的记录检索某些字段而不用知道确定是否允许访问的商业策略，这导致字段级的授权。

参数授权

参数授权是这样一种机制，它允许可能根据请求的上下文来确定所保护资源的授权决定。在由本发明提供的结构中，实施的范围已经被扩展来应用到所有的执行和资源容器。其完成是通过使得所有的执行和资源容器通过由安全服务提供的 AccessController 对象来获得授权和角色映射服务。这个 AccessController 利用通过服务提供者接口提供的授权服务，以便提供所请求的能力。

实际限定用于提供动态角色关联和参数授权的机制的服务提供者接口是基于一个被授权的授权设计，这个设计使得实施点被移到功能的提供者，而不是在应用服务器本身内。

另外，在包括授权框架的服务提供者接口上定义的方法利用一种实施机制，所述实施机制被设计来支持基于能力的授权机制，这个基于能力的授权机制更自然地与由第三方授权厂商提供的服务一致。基于能力的机制的使用使得被授权的授权方法支持基于能力的以及基于许可的实现方式。

传统的安全机制趋向于是无上下文的，因为它们完全基于对于给定的资源授予一个主方的许可。因此，可以进行的授权决定的仅有类型是：是否主体具有访问资源的必要许可。这些类型的授权决定更复杂，因为它们不趋向于表示商业决定。结果，经常在应用程序代码中需要另外的安全检查以补偿这个限制。例如，使用传统的方法，人们可以评价是否一个主方被允许向特

定的帐户划款。但是，不可能考虑转账的数量、源账户和目的账户的货币和星期日期。

在按照本发明的基于参数授权的机制中，使用请求的上下文来进行授权决定，因此授权决定更接近地表示实际的商业决定。基本上不需要补偿在应用程序本身中的安全检查。例如，在实际的银行业务示例中使用本发明，对于授权决定有可能考虑转账的数量、源账户和目的账户的货币和星期日期来作为处理商业策略的一部分，所述商业策略控制是否允许、由谁和在什么情况下转账。

对于上下文信息的访问

虽然一些其他系统已经尝试提供与上述的参数授权的说明类似的能力，但是这些较早的系统和方法全部要求调用方先前具有以下知识：被评价的商业策略的参数。对于这个知识的要求实际上提出了与在应用程序代码中进行授权决定同样的问题——商业策略的一些方面的知识。在商业策略改变来要求另一上下文信息的任何时间，必须修改和重新配置应用程序。

通过使用回调到来自授权提供者的容器而完成在不预先知道商业策略的情况下处理提供上下文信息的问题。在改变请求的参数类型或数量基本上改变应用程序需要重新配置应用程序的情况下，对于动态角色关联和参数授权提供者的实现方式有可能获得对上下文信息的访问而不需要应用程序先前知道商业策略的方面。

被定义来用于本发明并且支持授权框架的访问提供者接口可以利用与在Java验证和授权服务（JAAS）中定义的那个类似的标准回调机制。通过利用从容器向服务提供者接口的实现形式传送的回调处理器，提供者能够请求返回具体的上下文信息。当调用容器的回调处理器时，容器通过向回调输入适当的数值而响应。如果容器不明白在所述处理器中指定的回调，则它不以一个值来更新所述回调。应用程序或容器都不知道对于商业策略的改变，因为它们被查询由具体商业策略表达的评价驱动的上下文信息。

功能说明

可以被使用来满足参数授权的要求的权限处理引擎（权限引擎）的实现方式必须支持下列要求：

一种实现方式应该能够指定，来自商业请求的参数的值、主密钥、来自表示被验证的用户的 JAAS 主体中包括的主方的属性、任何被当作上下文的

一部分的可能的其他信息被表示为规则表达的一部分。

一种实现方式应当能够根据要求从调用方请求参数信息，而不需要由在规则评价的开始传送由规则的评价可能使用的任何参数信息。用于请求信息的机制应当是以与对于 JAAS 定义的那个类似的回调的形式。所述实现方式可以独立地回调每段信息，请求在单个回调中的多段信息，或上述的两者的结合。

一种实现方式应该能够指定将作为规则的评价的一部分被调用的一个 Java 类的名称。从所述 Java 类返回的值将用于评估规则表达。这可以被当作宏设施，其中用户所提供的代码被调用来计算一个值，这个值随后当评估所述规则时被考虑。例如在旅行订票的示例中，一组规则表达可以指定对用户提供的类的调用，这个用户提供的类计算或查找由主体表示的旅客至今已经经过的英里数。

一组规则的评价将产生一个布尔值，它指示是否所述规则被已经被成功地满足。

本发明向企业应用开发者提供一种框架结构，在其中可以配置或“插入”应用实现方式，具体是登录、授权和审查实现方式，以便使得可以开发能够容易适配的和定制的商业功能和方案。这些系统可以用于解释在特定方案下的上下文中的主方身份，并且定制商业功能以符合被解释的身份。

由本发明提供的所述框架结构具体上允许 Java 开发者可以使用现有的 Java 安全特征，但是以下述的方式来大大地扩展这些安全特征的使用：对于开发者（或用户）透明，并且可以允许所提供的安全被分布在企业而不需要安全处理的另外分层或翻译。

本发明的实施例在企业和服务器安全和商业业务流程的领域中特别有用。因为所述框架使用可插入的结构，因此它允许第三方厂商提供安全的实现插件而不需要修改核心结构。这些插件可以包括用于数字证书和密钥检索、证书验证和撤回机制（诸如 CRL 和 OCSP）的工具。所述可插入的框架结构因此允许安全访问控制企业资源，诸如企业 JavaBeans 组件模型（EJB）、应用程序（包括小服务程序和 JSP）、其他的联网资源（诸如 RMI、JNDI 和 JMS 目的地）。

另外，本发明的框架结构特别适用于提供一种安全授权和在企业的安全域中的安全资源访问的方法。每个具有一个域的服务器可以提供一个不同组

的特征、服务和应用程序。本发明使得开发者可以配置客户端应用，所述客户端应用可以通过单独的登录机制来使用每个服务器的特定服务，所述登录机制确定客户的在所述域中或在协作域之间的访问特权。

图 2 示出了按照本发明的一个实施例的安全结构的示例。如在此所示，客户 202、204（它们可以是物理硬件客户或软件应用程序）可以试图经由一个交易或应用服务器 208 访问所保护服务或资源 206，诸如永久目录服务器。虽然本发明可以用于任何其他的服务器产品或等同的系统，这样的交易服务器的一个示例是加利福尼亚 San Jose 的 BEA 系统公司的 WebLogic 服务器产品。因特网 CORBA 客户将通常试图通过因特网 inter-ORB 协议（IIOP）请求 212 来进行这样的访问。万维网客户通常试图直接或经由万维网服务器 214 或经由代理插件 216（在这种情况下，所述代理也可以提供附加的功能，诸如加密套接字协议层（SSL）加密 218）来通过一系列超文本传送协议（http）请求 210 而进行访问。在任何情况下，连接尝试经常经由初始连接滤波器 220 被交易服务器接收，并且被传送到安全访问 222。按照本发明，安全访问 222 是用于安全确定的焦点，包括客户端和用户级资源访问、授权、认证、特权评估和权限确定。企业 JavaBeans 组件模型（EJB）224、万维网应用程序（WebApp）226 和其他形式的应用程序全可以通过使用容器来使用安全服务。所述安全服务处理从这些容器向所保护资源的调用，这是图 2 的情况。所述调用可以由例如多个被管理的组件模型（MBeans）230 或 Java 命名的目录接口（JNDI）232 处理。

图 3 更详细地图解了安全服务结构 300 的实施例。所述安全服务增加由标准 Java 2 企业版安全集提供的基本安全服务和特征。如在这个示例中所示，基本 Java 安全集 302 包括安全提供者接口 304，用于尤其是密钥存储、验证、证书验证和安全套接字。客户应用程序 306 可以被写入以直接利用 Java 安全层和这些 SPI。如图 3 所示的本发明大大地增强了在利用这些和其他安全特征中的开发者的选项。按照本发明，在诸如 WebApp 容器 310 的 EJB 容器 308 的容器内配置客户应用。所述容器直接与安全服务 314 通信（在此与安全服务 222 相同），它继而与 Java 安全层 302 和它的安全 SPI 304 通信。这使得可以将对于安全授权决定的责任从应用转到安全服务层中。

另外，本发明理想上适用于将预先存在的 Java 安全 SPI 304 与常规的 SPI 316、318 集成。可以在安全服务器中包括或与其集成另外的 SPI，诸如连接

滤波器、访问决定接口、审查信道和证书认证器。所述安全服务协调对于被保护的企业资源的整个范围的访问。

图 4 图解了一个处理，通过它，一个应用程序可以使用本发明来访问安全资源。在步骤 402，开发者建立一个客户应用程序。这个应用程序可以是由自然用户或客户使用的应用程序，或可以是允许另一个应用程序使用安全服务器的应用接口程序。在步骤 404，所述应用程序被配置在一个容器内，下面更详细地公开所述配置处理。所述应用程序然后在步骤 406 使用这个容器来访问所述安全服务。选用地，在步骤 408，安全服务可以直接与一组符合 J2EE 的 Java 安全特征连接。或者，在步骤 410，所述容器被用于允许对第三方或常规安全提供者接口的访问。在步骤 412，如果安全服务已经指示应当允许所请求的访问，则所述安全提供者接口允许应用访问安全资源。

图 5 图解了用于在容器内配置一个应用的配置机制的一个实施例。如图 5 所示，数据被注册在安全服务 502（在此与安全服务 314 相同）中以供应用容器 504 使用。传统上，所述容器从配置说明 506 读出信息，并且实际的容器然后进行实施。如果存在不同的容器，一个用于 WebApp，一个用于 EJB，则需要有不同的实现引擎。另外，在 J2EE 环境中存在一系列资源，它们没有容器，并且没有配置描述符。在这种方法中的主要问题是在单个环境中以不同的方式实施安全的不同方面。本发明通过下列方式处理这个问题：首先，使得读出配置描述符的容器向安全服务通知安全限制和其中限定的策略，这导致在安全服务内提供安全策略。第二，当容器以后做出访问所保护资源的请求时，容器向安全服务授权所有授权决定，于是将实施点从容器移到安全服务。通过参考经由配置描述而提供的安全限制和策略以及经由管理工具由管理员限定的任何安全限制和策略可以做出授权决定。也可以使用配置工具来读出配置描述符和向安全服务记录其中包括的安全信息。所述配置工具可以随后直接达到安全服务而不用读取配置描述符。

图 6 图解了一个处理，通过它，在应用配置期间向安全服务输入数据。在步骤 602，开发者建立配置描述。在步骤 604，所述描述被传送到容器，其中所述容器分析信息以评估应当向安全服务传送什么信息。选用地，在步骤 606，所述配置描述符可以被提供给配置工具，所述配置工具将所述信息传送给安全服务。

图 7 图解了使用按照本发明的安全服务器以访问在网络或服务器上的所

保护资源的一个客户端的示例。所述处理以进行请求的客户端 702 开始，这在一个示例中可以是验证其本身的客户端的简单情况。所述客户端建立到容器的连接，由箭头 704 所示。权标 706 与所述请求一起被传送。作为进行连接的一个部分，系统（包括安全服务器 710）确定资源 708 被保护，因此客户端需要验证。在这个示例中，权标表示客户端的被验证的用户，它们依次在启用请求中被传送到所述容器，所述容器随后在调用安全服务的后面步骤中使用它们。在一些实施例中，所述权标表示客户端本身的被验证身份。客户端向安全服务 712（相同的安全服务或安全服务 314）提出问题 714，“由这个权标识别的用户能够访问和执行在目标资源上的被请求能力吗？”。

为了回答这个问题，所述安全服务使用图 7 中被指示为 AC 716 的访问控制器。AC 716 做两件事：它作为扩展在一系列访问决定（AD）插件 718、720、722 的输出端，并且它也作为裁判员。当所述 AC 调用所述访问决定时，它将所述权标依次传递给每个访问决定，以第一个访问决定 718 开始，即问“这可以吗？”所述访问决定返回三个响应之一：允许、否定或放弃。如果有下一个，则所述访问控制器然后移动到下一个，直到已经轮询了所有的访问决定。实际上，所述安全机制就像一系列窗玻璃。如果 X 718 说“允许”，则处理继续；如果 X 718 说“否定”，则处理停止。所述 AC 随后移动到访问决定 Y 720 并且问提供给完全相同的实体的完全相同的问题，Y 720 类似地以允许、否定或放弃来响应。

但是，这个处理继续用于所述服务所具有的这些插件中的许多个。一个裁判员策略确定是否任何人说否定，然后处理结束，或简单返回一个完全否定。所述结构允许做出这个确定的裁判员策略足够灵活，以便如果 X 718 是允许，并且 Y 720 是否定，C 722 是放弃，则那可以被当作允许。其他的实施方式可以更为突出：如果存在任何否定，则它是完全否定。每当所述 AC 问所述访问决定一个所述访问决定不唯一明白的问题，并且因此不能做出明确的决定，则返回一个放弃。每当包括一个需要用于一些但不是全部的操作的传统授权系统时，或者当 X 718 是公司策略并且 Y 720 是行业时，这是最普通的。在这种情况下，可能有被问的问题，它们被公司策略而不是被行业实施，因此行业决定做出者说“我不明白此，我放弃”。

当轮询所有的访问决定做出者并且直到请求最后通过的点也没有人说否定时，则所述访问请求可以达到没人否定、一些人允许和一些人放弃的点，

例如其中 Y 720 放弃，X 718 和 C 722 都允许。所述裁判策略可以随后用于控制如何确定结果。在一个实施例中，仅仅存在两个结果到容器：允许或否定。因此访问控制器查看裁判策略以确定如何做出决定。例如在一些超出范围的实施方式中，系统可能使用这样的裁判策略，它说“我需要无异议的允许吗？”。在这种情况下，X、Y 和 C 全部必须允许。如果它们中的任何一个不允许，则结果是否定。其他的手段是不需要无异议的同意，在这种情况下，只要没有人否定，则允许访问发生，这个决定是根据裁判策略被做出的。

无论在访问决定处理中发生什么，安全服务本身将请求传送到审查子系统 724 以请求它审查刚发生了什么和结果。仅仅在这个点，安全服务才告诉容器最后的结果是允许或否定。如果是允许，则容器随后使得可以向所保护资源分派所请求的操作。如果是否定，则请求不达到那个点，并且立即被拒绝。

本发明也提供了一种方法，用于在将结束时、即在所保护资源成功接收到一个访问请求后来调用访问决定处理。在这种情况下，可以假定系统通过对于请求的首先查看，并且允许了所有事情，因此所述方法被分派到资源。所述资源返回意欲返回客户端的数据。不是仅仅盲目地返回这个数据，而是重新进行授权查看，这一次使得安全服务、特别是访问决定查看要返回的数据以确定是否向客户端返回它的 OK。在此的一个示例是商业运行的示例，其中用户试图请求文件。当用户首先进入时，他们被分配一个安全许可，这表示他们可以查看未分类在他们的安全许可上的文件。如果他们仅仅具有保密的安全许可并且要被返回的文件是绝密的，则即使他们使得所述方法运行，系统将不要求允许结果返回用户，因为他们随后将被允许查看他们不被允许查看的一些内容。所述访问决定被重新运行，这一次查看将返回用户的输出，并且做出相同种类的决定。在这个示例中，可能有许多放弃，例如，Y 访问决定 720 说“我不关心返回什么输出，我仅仅关心进入的输入”。开发者可以混合和匹配插件以根据行业或基于公司策略而获得不同的策略。

图 8 图解了一个流程图，它示出了由本发明的一个实施例在访问一个所保护资源中使用的处理。在步骤 802，客户端在容器中开发和配置一个应用程序。当所述应用程序要求访问所述所保护资源时，一个调用被做出并且被转发到适当容器，如步骤 804 中所示。在步骤 806，所述容器用于调用安全服务。在步骤 808，在所述安全服务内，一个输出访问控制器确定在验证或

确认这个特定的访问请求中发生哪些访问决定。在步骤 810，选择并且处理或轮询每个访问决定，以便确定它对访问请求的帮助，每个访问决定的输出是否定、允许或放弃。在步骤 812，一个审查机制跟踪每个访问决定的结果。根据访问决定的结果和累积效应——如果它是允许、否定或放弃——在步骤 814，容器被允许（或拒绝）访问所保护资源。在步骤 816，数据可以随后被从客户端应用传送到所保护资源，并且反之亦然。如果客户端后来向同一资源或不同的资源（如在步骤 818 中确定的）请求，则所述请求被转发到适当的容器，并且需要另一个授权决定（步骤 820），否则会话结束（步骤 822）。

图 9 图解了按照本发明的一个实施例的安全和个性化层，包括权限层 904 以及它相对于商业应用层 908、商业策略层 906 和安全授权层 902 的定位。对本领域的技术人员显然的是，图 9 所示的图解是如何开发权限层的抽象表示，但是可以使用许多其他的变化方式，并且这样的具体层不可（实际上很可能不会）用于任何特定的设计实现方式中。图 9 示出了如何使用权限 904 以超出在传统的安全结构（表示为 902）和经常明显的商业策略结构（表示为 906）之间的边界。通过使用这样的权限而覆盖的信息用于指导被表示为 908 的公司的商业应用的操作，所述公司的商业应用的操作包括根据实体的权限来对于独立的实体个性化所述公司的商业应用。

图 10 示出了如何可以使用权限来动态地产生角色。权限处理 1002 允许在上下文中问出安全问题。本发明包括根据那个权限向诸如主方 1 1004、主方 2 1006 到主方 n 1008 的特定主体 1000 动态地分配角色的能力，所述权限进一步允许系统在安全层内提供应用上下文，并且允许按照所述应用来以词句表达授权问题。只有根据上下文提出这些问题，所述系统才可以准确地回答那些问题，就像医生和病人的普通示例那样。如果提出问题“史密斯医生能够修改病人的医疗图表吗？”，则它是与上下文无关的问题。传统的回答是“有时能够有时不能够”。正确地确定回答的唯一方式是直到问问题的上下文，即我们在说哪个病人以及也许是那个医疗图表。所述系统可以随后进行诸如所有权的角色分配或一般进行动态分配，所述动态分配允许系统进行商业决定而不是安全决定。

通过使用简单的符号，我们可以将所述概念表示如下：

对于每个主体，有效角色 AR 是主体正在使用的角色：

AR(s:主体)={主体 s 的有效角色}

每个主体可以与一个或多个授权角色 RA 相关联:

RA(s:主体)={主体 s 的授权角色}

每个角色可以被授权来执行一个或多个任务 TA:

TA(r:角色)={对于角色 r 授权的任务}

如果在当前的时间点以当前的值判定函数 exec(s,t)为真,则主体可以执行任务:

exec(s:主体,t:任务)=如果主体 s 可以执行任务 t 则为真

为了正确地评价上述的语句,需要几个规则:

1. 角色分配 – 只有主体已经被分配一个角色时,一个主体才可以执行一个交易。识别和验证处理不被作为任务的一部分,而且通过任务来进行对于系统执行的所有其他行为:

$\forall s$:主体, t:任务$(exec(s,t)AR(s) \Rightarrow \neq 0)$

2. 角色授权 – 必须对于一个主体授权所述主体的有效角色。这个规则保证用户可以仅仅承担它们被授权的角色:

$\forall s$:主体$(AR(s) \subseteq RA(s))$

3. 任务授权 – 只有对于一个主体的有效角色授权一个任务时,所述主体才执行所述任务:

$\forall s$:主体, t:任务$(exec(s,t) \Rightarrow t \in TA(AR(s))$

作为另一个示例,开发者可以使用本发明来根据一个用户被授权去做的内容来个性化网页。例如,可以使得自注册或自助特征成为信用卡网站的一部分,其中,用户仅仅是可以改变其信用卡号码的一方。厂商可以读出这些信用卡号码,但是厂商不能改变这些信用卡号码,并且其他任何人不能看到它们。系统不能确定如何进行如此,除非它知道上下文。有人可以修改信用卡字段吗?当然,有人可以修改信用卡字段。那么,现在,谁可以修改它?这是上下文的一部分。然后,系统必须问具体是哪一个,因为它不能仅仅使得任何成员修改任何其他成员的信用卡。因此,系统必须知道正在考虑哪个对象和哪个简档,所以它可以随后问“你是那个简档的拥有者吗?”过去,唯一定义的事情是静态角色。本发明通过使用权限而做的事情是允许商业分析人员和商业策略制订者定义在运行时间计算的动态角色,以便向对象或主方身份定义。这些角色不损害基础服务器结构的安全规格。例如,一个请求可以是注册参加拥有者角色的调用方。所述系统不定义拥有者角色,但是,

它可以将其动态地与用户相关联，并且然后一旦它们试图做出使得它们不再是其拥有者的一些事情，则从用户撤回拥有者角色。诸如 EJB 的方法可以利用这一点：用户被允许做一些事情是根据是否你在那个角色，并且你使用标准 EJB 来查询那个能力。

如何使用本发明的另一个示例是通过提供单一签字机制，它可以将传统的安全实现方式的元素与商业策略实现方式的元素相结合。如图 11 所示，在第一安全域 A 1102 在初始登录处理期间接收的信息可以被第一登录模块 1108 获取，并且被传送到（由箭头 1114 指示）在第二安全域 1104 中的第二个登录模块 1110，其中所述第一安全域 A 1102 例如是由第三方应用提供并且不知道用户的简档的安全域，或者是公司的商业策略的安全域。第二安全域 B 1104 可以知道商业策略，并且除了原始登录信息之外还可以确定或计算用户的权限或权限简档 1102。原始登录信息和权限信息（分别由箭头 1118 和 1120 指示）可以被传送到在第三安全域 C 1106 中的第三登录模块 1112，等等。

安全提供者接口

可以用于本发明的安全提供者接口（SPI）包括用于下列的接口：

- 验证 – Java 验证和授权服务（JAAS）LoginModule
- 授权 – AccessDecision
- 审查 – AuditChannel
- 主方映射->角色映射 – JAAS LoginModule
- 主方->证书映射 – JAAS LoginModule
- 密钥存储 – Java 2 KeyStoreSpi
- 证书检索和撤回 – CertPath
- 证书映射 – CertAuthenticator
- 硬件加速器 – Java 密码扩展
- 周边保护 – ConnectionFilter

对于本领域的技术人员显然的是，在此所述的 SPI 仅仅是可以由本发明使用的接口类型的示例，并且在此被列出以说明由本发明提供的安全服务结构的灵活性。也可以使用其他类型的接口来替换或加到所述的那些接口，这仍然在本发明的精神和范围内。

验证 SPI

本发明使用的JAAS登录模块基于标准的JAAS LoginModule，但是另外支持标准的登录模块实现方式。验证SPI被扩展来支持用户、组的管理，并且提供对于多个JAAS登录模块的支持。可以在登录模块实例上共享证书，并且可以提供以用户简档信息来更新的独立登录模块。验证SPI的责任包括根据安全域范围来验证用户，并且在JAAS主体中填充主方。

授权SPI

授权SPI供给访问决定，对于它没有足够的Java等同物。授权SPI支持多个说明性的或基于规则的授权模型。调回处理器可以用于获得对于启用上下文的访问。SPI也提供至主体信息的访问，包括主方实体，组成员和角色分配，以及用户简档信息(选用的)。多个访问决定提供者可以以堆栈来被配置。验证SPI的责任包括进行允许、否定或放弃的授权决定或者用于由应用定范围的所保护资源的访问请求。图11图解了如何可以使用本发明来在一个域内的安全域之间分布简档信息。

审查SPI

审查SPI包括审查信道，对于它未定义任何Java等同物。使用一个调回处理器来获得对于启用上下文的访问，包括对于主体信息、主方身份、组成员和角色分配，以及用户简档信息（选用的）的访问。而且，多个审查信道提供者可以以堆栈来被配置。审查SPI的责任包括确定是否应当审查信息，并且根据服务质量策略来执行数据的实际审查。

主方->角色映射SPI

主方->角色映射SPI（JAAS LoginModule）SPI是基于JAAS登录模块，并且用于向在主体中包括的主方的身份动态地映射角色。所述角色可以已经被管理员或通过配置描述符来明确定义，或者它们可以基于用于商业请求的参数、在主方的简档中的属性的值以及其他条件来动态地被计算。

主方->证书映射SPI

主方->证书映射（JAAS LoginModule）SPI是基于JAAS登录模块，并且用于当通过安全域策略或技术边界时映射主方身份。主方映射SPI的责任是基于所提供的主体，并且可以用于向主体加上具有适当信息的公共证书，诸如用于用户姓名/密码的密码证书和用于权标类型的证书的一般证书。

密钥存储SPI

Java 2 KeyStore SPI是基于Java 2 KeyStore接口，并且提供对于在多个媒

体中存储的密钥的一致访问，所述媒体包括基于文件的：（PKCS#5/#8，PKCS#12），基于 HSM 的（PKCS#11），智能卡/Java 卡。密钥存储 SPI 提供对于代码签字的必要支持，并且它的责任包括从所保护的存储器检索私钥和从所保护的存储器中检索保密内容。

证书检索 SPI

证书检索 SPI（CertPath）是基于 JSR 55，并且除了支持多个证书类型之外，还提供用于检索、验证和撤回数字证书的一致机制。CertPath SPI 的责任包括：CertStore：经由查询来从存储器检索证书；CertPath：使用 CertStore 检索证书链；CertPathChecker：验证证书；CertPathValidator：验证证书链 。

证书映射 SPI

未定义任何 Java 等同物的证书映射 SPI（CertAuthenticator）提供一种可以扩展的手段来允许在数字证书和在安全域内的主方身份之间的映射。它还提供了在调用 CertAuthenticator 之前执行的验证和撤回查看。

证书映射 SPI 的责任是基于被验证的客户端的数字证书而映射到用户姓名和主方域。

硬件加速器 SPI

硬件加速器 SPI 使用 Java 加密扩展，并且是基于 Java 2 安全结构。其责任包括在安全服务的控制下提供加密服务。

周边保护 SPI

周边保护 SPI 包括连接滤波器，用于提供内置的连接滤波器实现方式。它着重于使用方便，并且可以经由 Mbeans 从控制台被配置，允许对于用户写入的滤波器和级联滤波器的连续支持，其中每个依序被执行。这最小化了客户必须处理的条件。

安全管理 SPI

安全管理 SPI 将第三方厂商控制台集成到服务器控制台，并且允许将安全从这些厂商转到小服务程序。安全管理 SPI 提供对于用户、组、角色和授权信息的全生存周期的支持。

单一签字（SSO）SPI

单一签字 SPI 使用 LoginModule 来从启动主方的身份向资源身份映射，并且提供资源证书的安全存储。

本发明的优选实施例的上述说明已经被提供来用于说明和描述。不是意

欲穷尽或将本发明限制到所公开的精确形式中。显然，对于本领域的技术人员许多修改和变化是显然的。所述实施例被选择和说明以便最好地解释本发明的原理及其实际应用，因此使得本领域的其他技术人员可以明白本发明的各种实施例和具有适合于所考虑的特定使用的不同修改的本发明。本发明的范围意欲由所附的权利要求及其等同物限定。

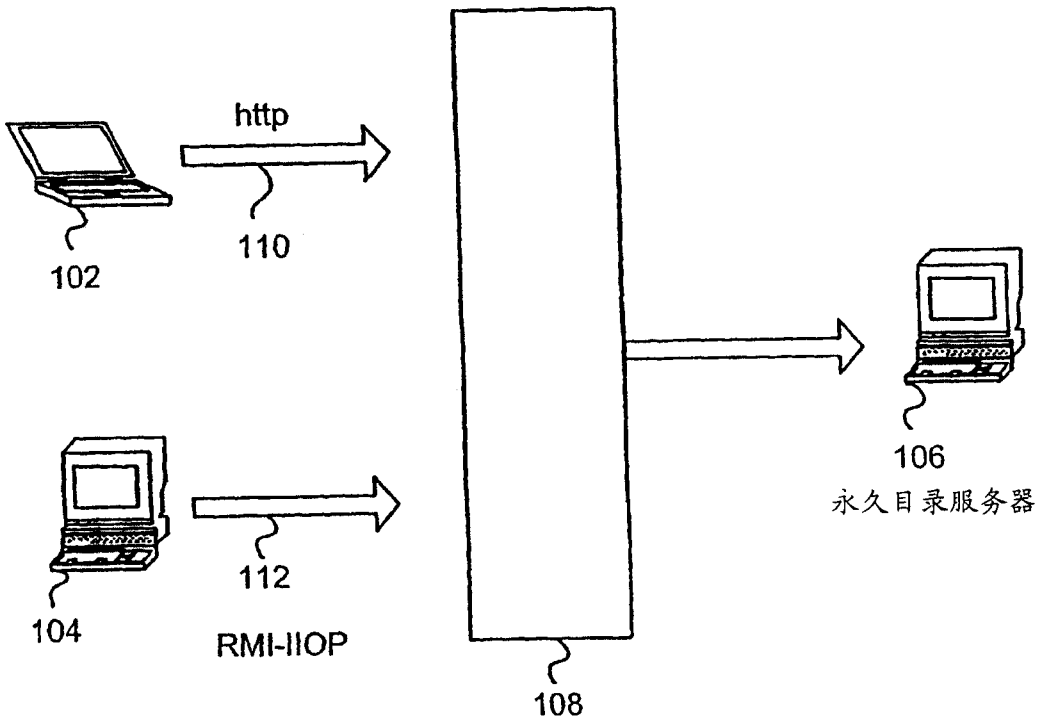


图 1

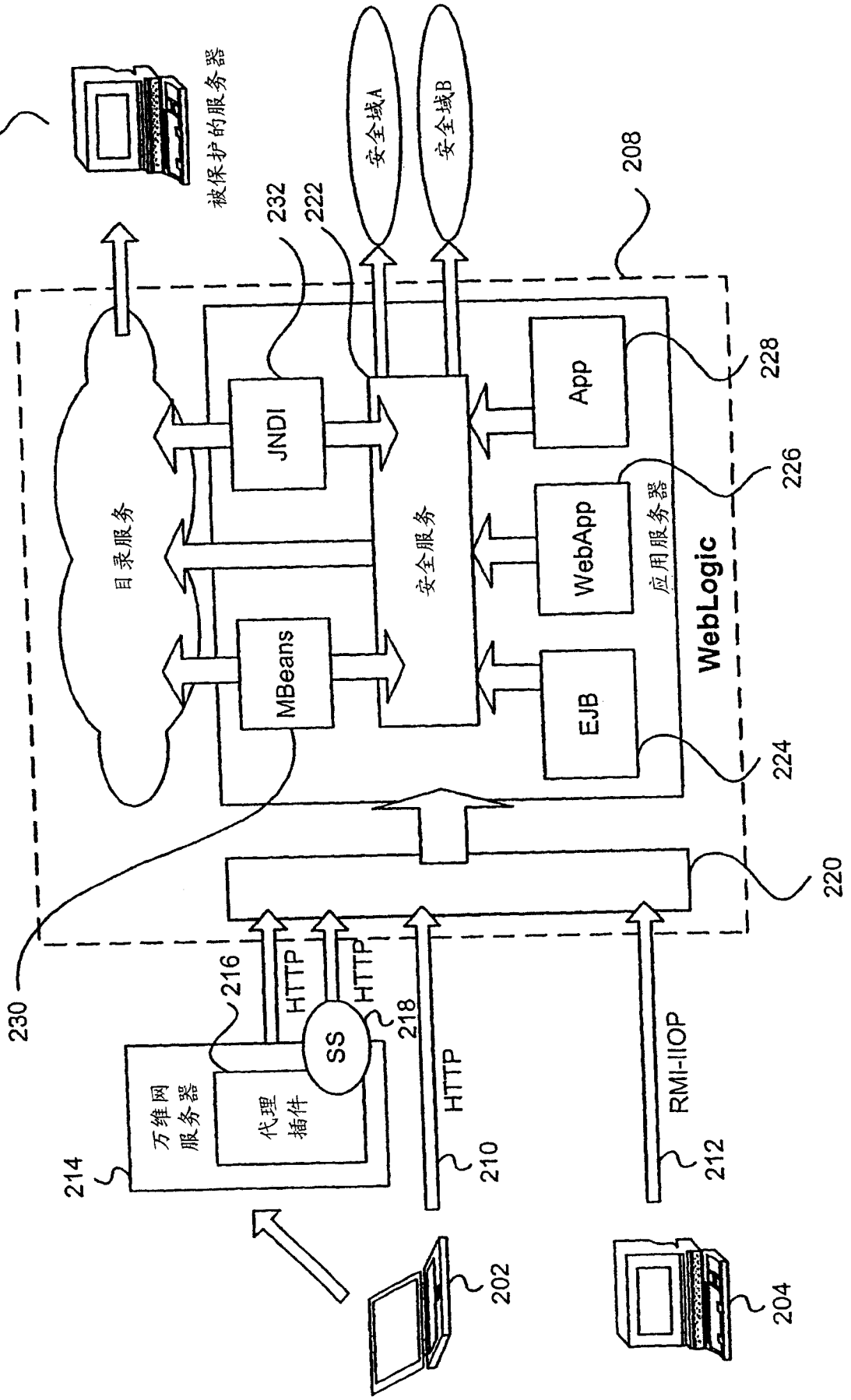

206
被保护的服务器
安全域A
安全域B
232
222
208
JNDI
App
228
目录服务
安全服务
WebApp
226
应用服务器
WebLogic
MBeans
EJB
224
220
230
216
HTTP
HTTP
SS
218
万维网
服务器
代理
插件
214
HTTP
210
RMI-IIOP
212
202
204

图 2

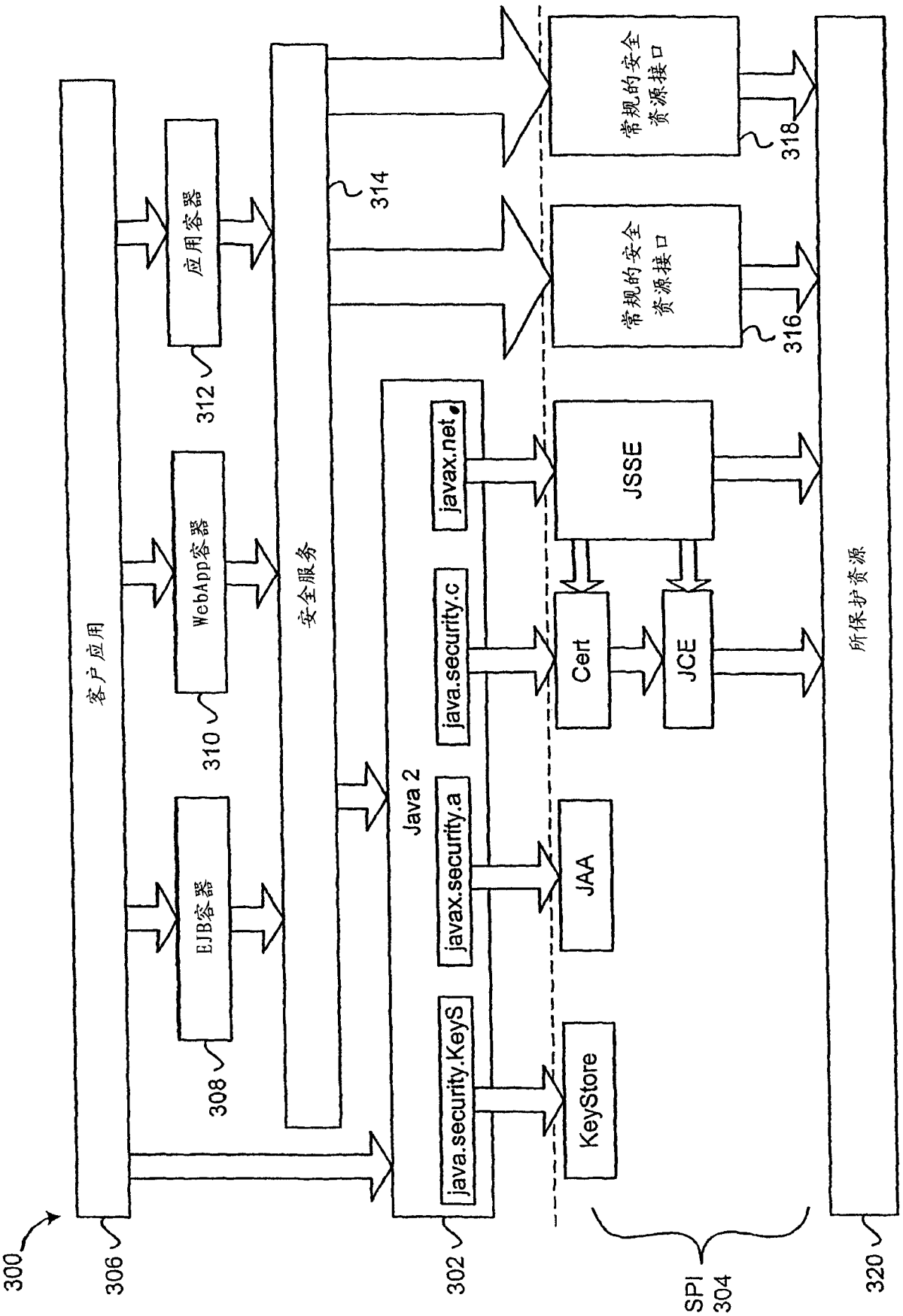

300
306
客户应用
302
Java 2
安全服务
314
应用容器
312
WebApp容器
310
EJB容器
308
javax.net
java.security.c
javax.security.a
java.security.KeyS
JSSE
Cert
JCE
JAA
KeyStore
常规的安全资源接口
318
常规的安全资源接口
316
SPI
304
所保护资源
320

图 3

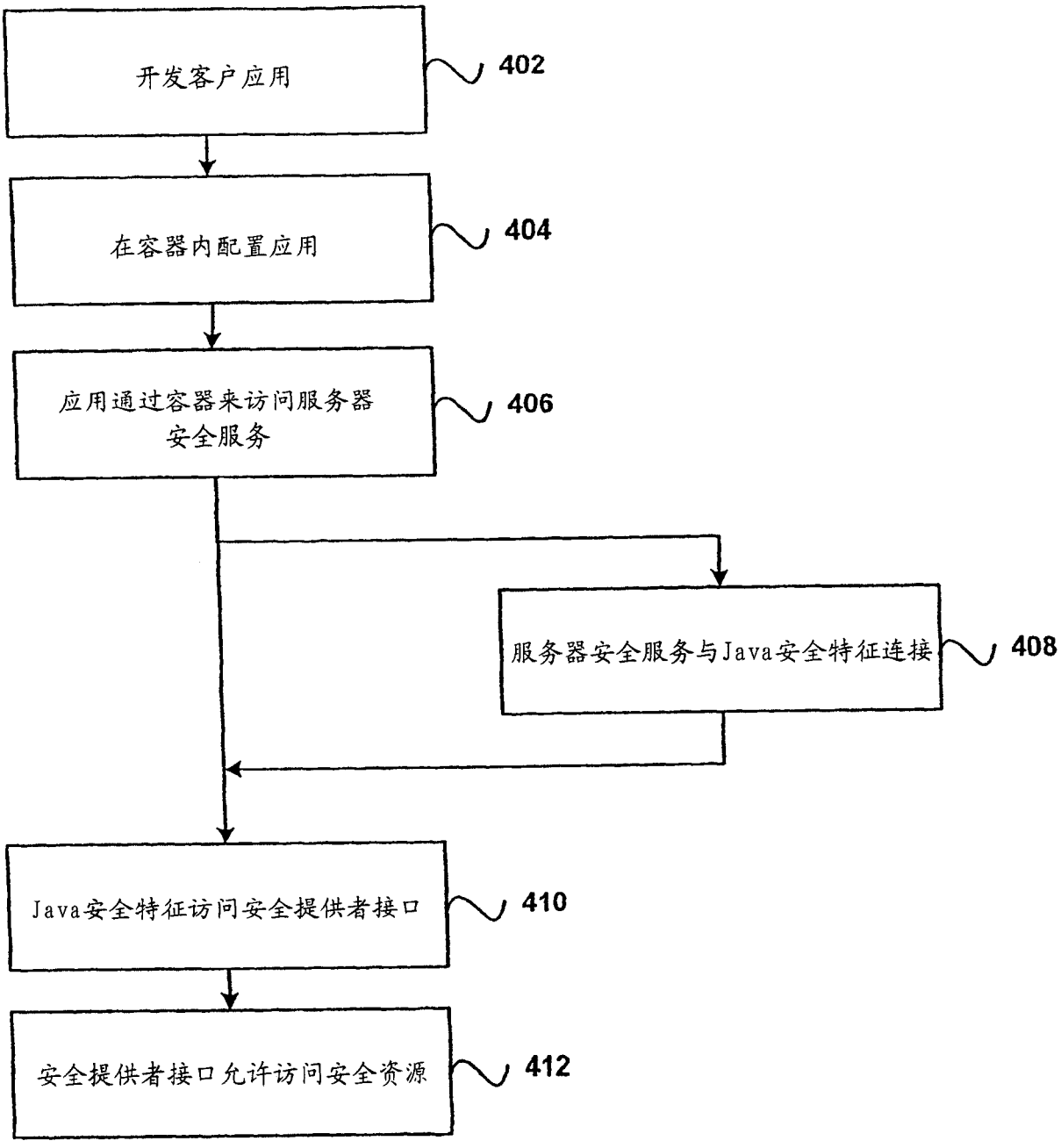

开发客户应用
402
在容器内配置应用
404
应用通过容器来访问服务器安全服务
406
服务器安全服务与Java安全特征连接
408
Java安全特征访问安全提供者接口
410
安全提供者接口允许访问安全资源
412

图 4

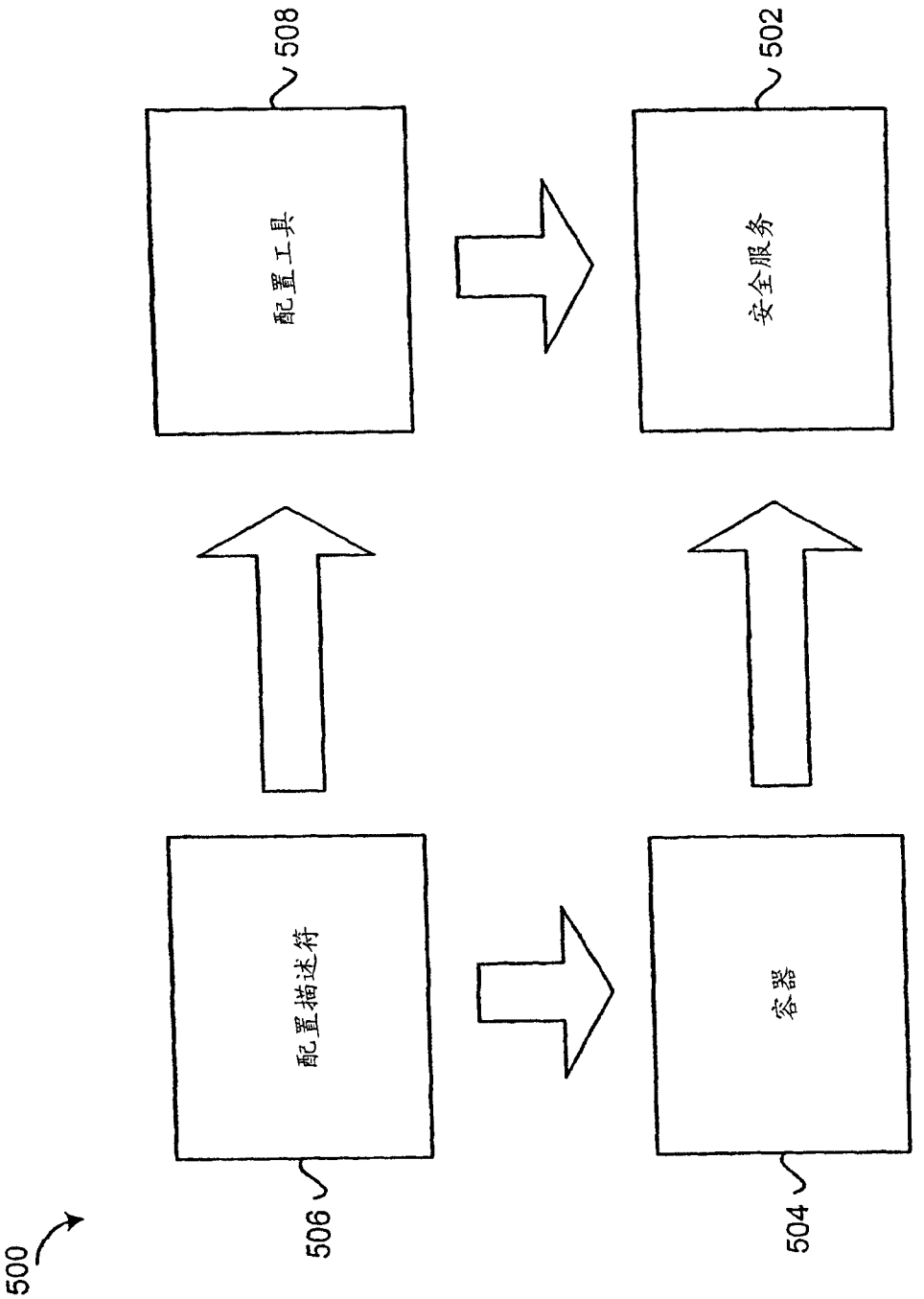


图 5

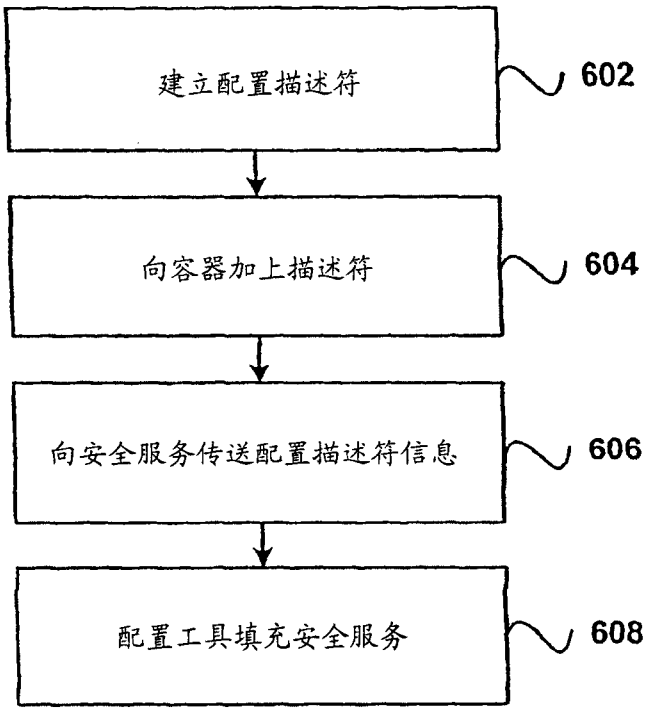

建立配置描述符
602
向容器加上描述符
604
向安全服务传送配置描述符信息
606
配置工具填充安全服务
608

图 6

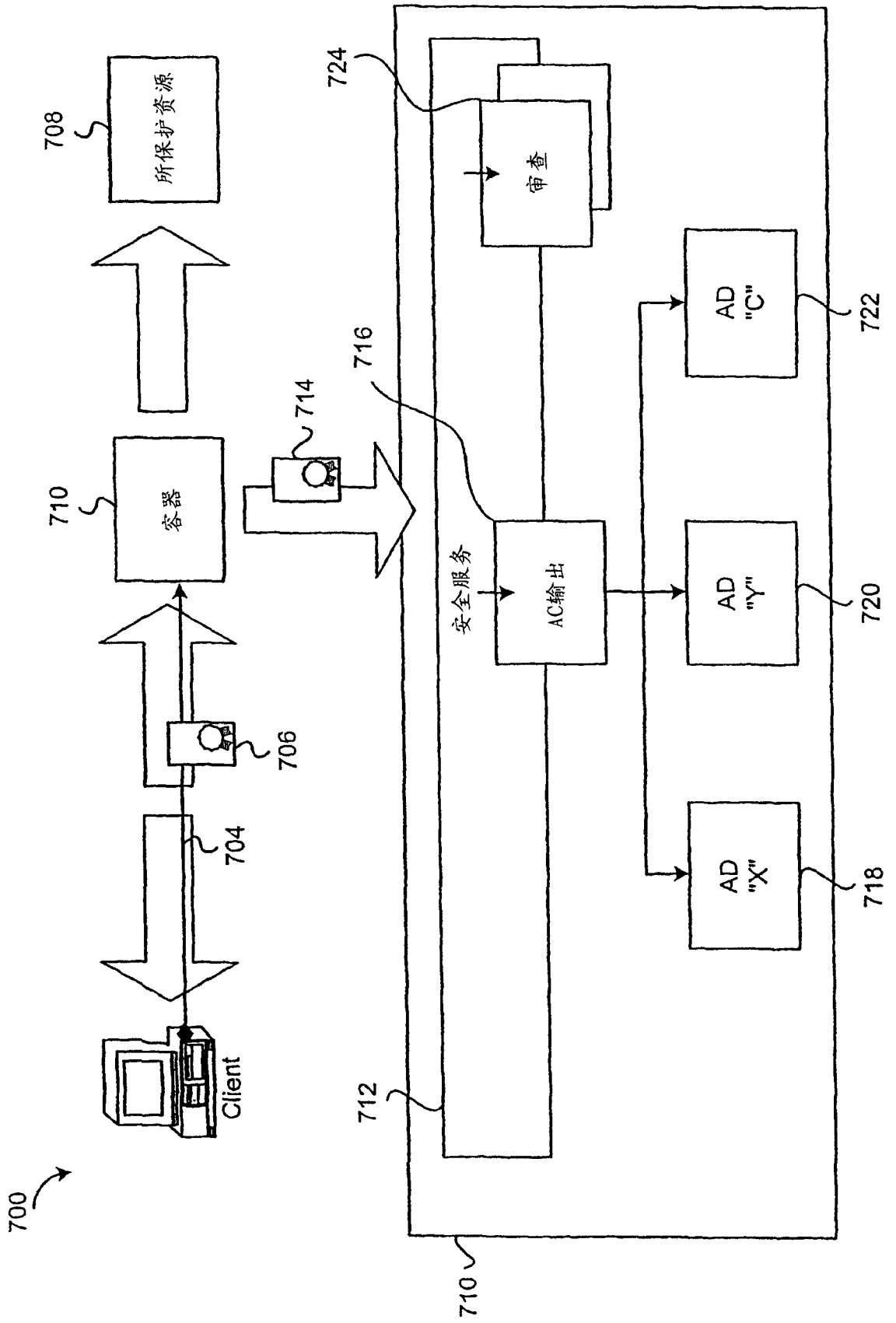

700
Client
704
706
710
容器
708
所保护资源
712
714
716
724
审查
安全服务
AC输出
AD "C"
722
AD "Y"
720
AD "X"
718
710

图 7

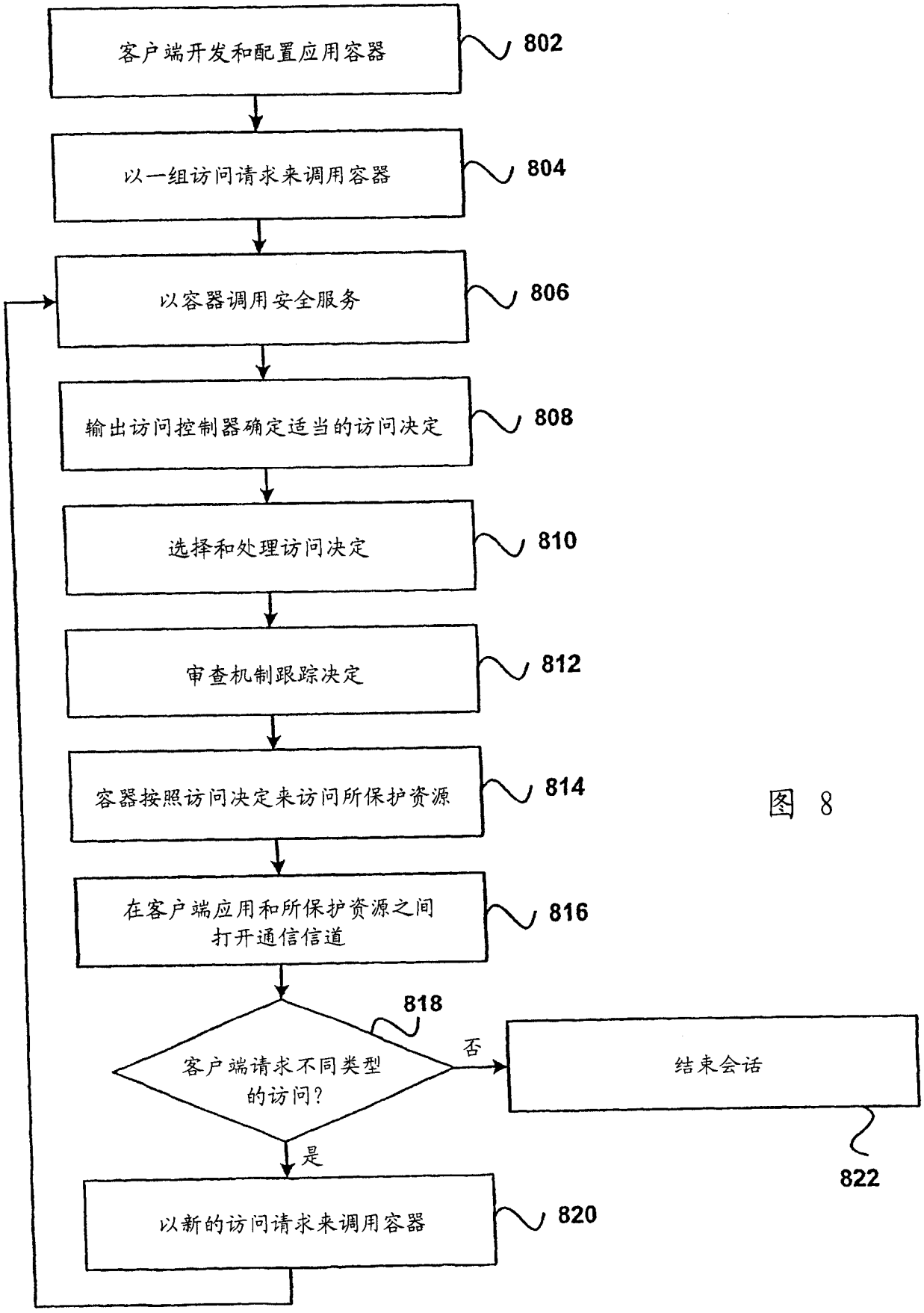

客户端开发和配置应用容器
802
以一组访问请求来调用容器
804
以容器调用安全服务
806
输出访问控制器确定适当的访问决定
808
选择和处理访问决定
810
审查机制跟踪决定
812
容器按照访问决定来访问所保护资源
814
在客户端应用和所保护资源之间
打开通信信道
816
818
客户端请求不同类型
的访问?
否
结束会话
822
是
以新的访问请求来调用容器
820

图 8

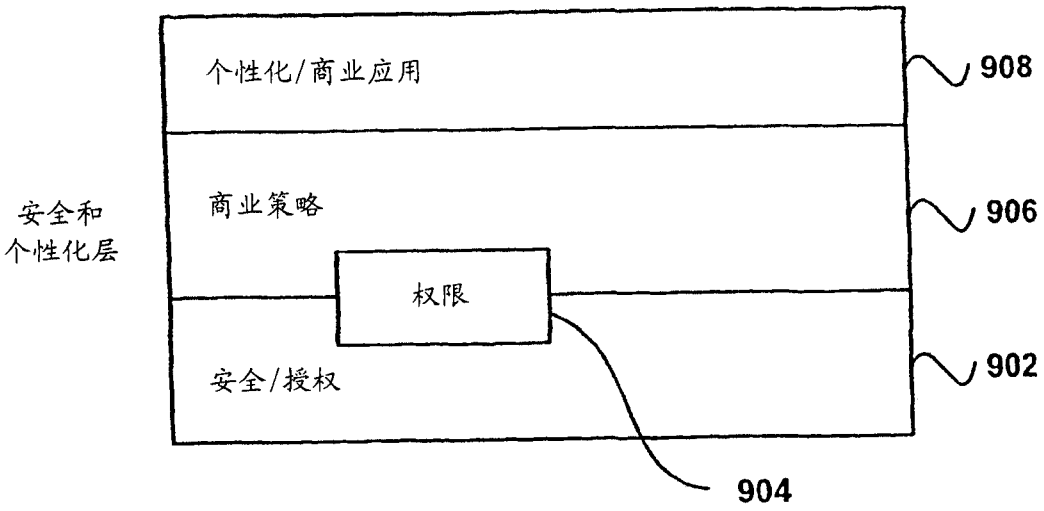

个性化/商业应用
908
商业策略
906
安全和
个性化层
权限
安全/授权
902
904

图 9

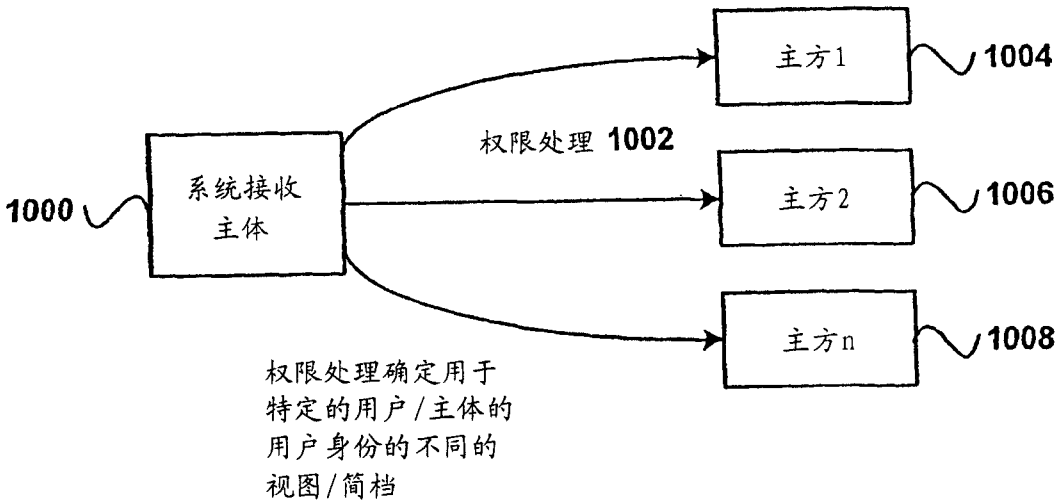

主方1
1004
权限处理 1002
1000
系统接收
主体
主方2
1006
主方n
1008
权限处理确定用于
特定的用户/主体的
用户身份的不同的
视图/简档

图 10

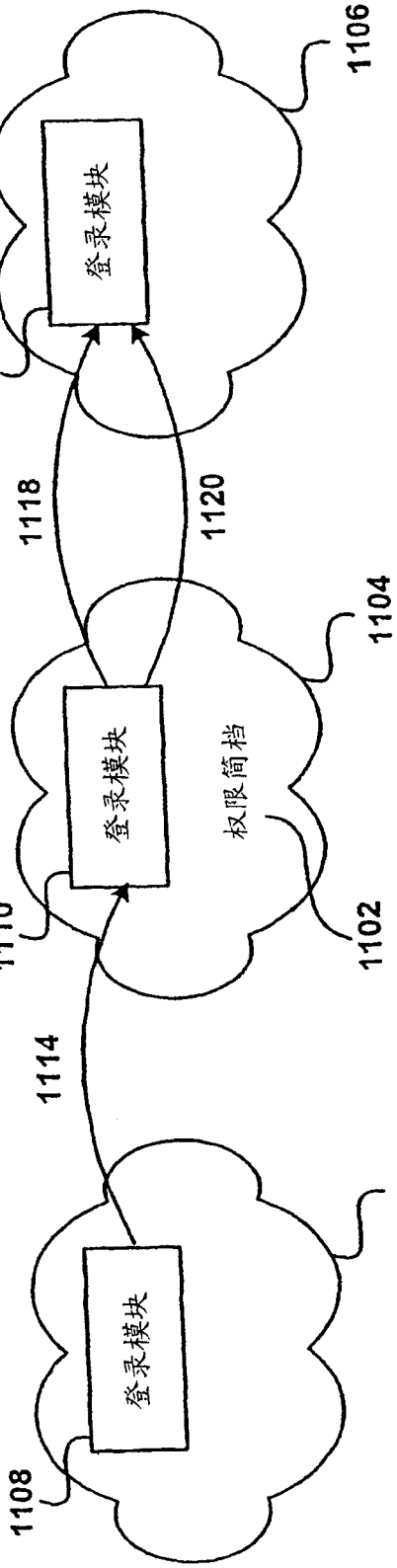

登录模块
登录模块
登录模块
权限简档
1102
1104
1106
1108
1110
1112
1114
1118
1120
安全域A
(例如不知道用户简档
的第三方系统)
安全域B
(例如商业模型
和个性化层)
安全域B
(例如商业模型
和个性化层)

图 11