关于网格计算授权机制的研究*

吴毓毅, 贺也平
(中国科学院 软件研究所, 北京 100089)

摘　要: 分析了目前网格计算中最流行的安全机制 GSI(Grid Security Infrastructure, 网格安全基础设施) 和基于 GSI 的 CAS(Community Authorization Service, 组织授权服务) , 提出了一种基于本地角色授权的、能够解决大规模 VO(Virtual Organization, 虚拟组织) 的授权问题的方案。同 GSI 和 CAS 不同的是, 本方案中的用户只需要进行本地认证就能够根据其在本地组织的角色来访问 VO。

关键词: 虚拟组织; 网格安全基础设施; 组织授权服务; 代理凭证
中图法分类号: TP303　　**文献标识码:** A　　**文章编号:** 1001-3695(2005) 08-0081-03

Research about Grid Computing Authorization

WU Yu-yi, HE Ye-ping
(Institute of Software, Chinese Academy of Sciences, Beijing 100089, *China)*

Abstract: Analyzes the most popular secure mechanism GSI in grid computing and the authorization system CAS based on GSI. Proposes own authorization scheme, which is based on local authentication and CAS. Different from GSI and CAS, users in the scheme only need to authenticate to local system before accessing to VO.

Key words: VO(Virtual Organization) ; GSI(Grid Security Infrastructure) ; CAS(Community Authorization Service) ; Proxy Credential

1　引言

网格计算的思想就是整合分布的、异构的资源, 提供一定 QoS 的服务和能力。因此, 对于大规模的协同合作的网格计算, 一个非常重要的问题就是要解决网格用户跨组织、跨异构平台的认证和授权。目前关于这方面的文章主要集中在解决认证问题, 而授权方面的文章相对比较少。

关于授权的一种解决方案是, 将属于不同独立组织的资源和/或人员组织起来创建一个 VO(Virtual Organization, 虚拟组织) [1] 。但在 VO 中经常需要复杂和动态的策略来监控到底谁可以使用哪些资源, 以及出于什么目的使用这些资源。为了减少策略变动给 VO 管理者和资源提供者带来的负担, 大规模 VO 中的策略实施需要解决好以下几个问题:

(1) 可扩展性(Scalability) , 简单地说, 就是实体要能够方便地加入或者退出一个 VO。

(2) 灵活性(Flexibility) 和描述性(Expressibility) 是指策略实施要能够方便灵活地描述 VO 策略经常变动的特性。

(3) 策略层次性(Policy Hierarchy) 是指 VO 中可能包含子 VO, 策略实施要能够解决 VO 策略和子 VO 策略的一致性等问题。

目前网格计算中应用非常广泛的安全机制是 GSI(Grid Security Infrastructure, 网格安全基础设施) [5, 6, 8] 。但是, GSI 主要解决了认证和消息保护问题, 并没有解决扩展性、灵活性、描述性以及策略层次性的问题。CAS(Community Authorization Service, 组织授权服务) [2, 3] 基于 GSI, 并在每个 VO 中引入一个 CAS 服务器集中管理 VO 中的授权问题, 很好地解决了上述问题。

虽然 CAS 采用集中管理解决了上述问题, 但是当 VO 的规模很大时, VO 管理者(CAS 服务器管理员) 的负担还是非常重的。考虑到 VO 实际上是由一个或多个独立组织中的人员和资源组成的, 我们提出了一个基于本地组织中的角色使用 VO 资源的方案。在这个方案中, 独立组织中的人员并不是以独立的身份加入 VO, 而是以其在本地组织中的角色加入 VO。这样一来, 就可以大幅度削减 VO 的用户规模, 并能够让用户受到本地策略的约束(比如规定用户可以担任哪些本地角色) 。

2　网格安全基础设施 GSI

GSI 是基于公钥加密、X. 509 证书和 SSL 通信协议的一种安全机制, 主要解决了 VO 中的认证和消息保护问题。在早期的一些工作[6] 中, GSI 使用私钥和 X. 509 [7] 证书作为用户向网格资源认证的基础。现在, GSI 使用一种称为代理凭证(Proxy Credential) 的临时凭证。通过代理凭证, GSI 用户在访问分布在多个站点上的资源时就不必重复认证, 亦即可支持单点登录。另外, 通过代理凭证 GSI 用户还能够将用户权限委托(Delegation) 给远程进程。

2.1　单点登录

在一个网格计算环境中, 用户可能需要同时访问多个站点上的资源, 这样就有可能要求用户进行多次认证, 从而要求用户多次键入其私钥加密密码。这无论从方便性还是安全性的

收稿日期: 2004-08-05; 修返日期: 2004-09-28
基金项目: 国家重点基础研究发展规划项目(G1999035802)

角度都是不可取的。

GSI 使用代理凭证解决了这个问题。用户生成一对新的公私钥对,并用其用户证书签发生成一个新证书。新证书包含了用户的身份和新生成的公钥,并被略作修改以指示这是一个代理证书。代理证书是由用户签发的一个短暂证书。代理凭证包含了新生成的私钥、代理证书和用户证书。用户在使用代理凭证认证时,必须向验证方同时提供用户证书和代理证书。验证方首先检查用户证书的有效性,然后检查代理证书是否为用户所签发,最后判断用户是否拥有代理证书的私钥。若条件都满足,用户认证成功。

代理凭证通过运行 GSI 的客户端工具 grid-proxy-init 生成。代理凭证的私钥可以以明文的形式保存在本地存储系统中,并禁止其他用户对私钥的读取。这是因为代理证书是一个短期的证书,所以代理证书的私钥不必像用户私钥那样加密保存。代理凭证一旦创建,在凭证过期前,VO 用户可以用它同资源提供者进行相互认证,而无须再键入密码。

代理证书除了包含用户身份之外,还可以在其扩展项中承载一些策略信息来限制其使用,包含有这样的策略信息的代理证书叫做受限代理(Restricted Proxy) 。在后面部分我们会看到受限代理在 CAS 中的使用。

2.2 委托

在分布式应用中,用户提交一个作业到某个远程站点 A 上去执行,这个作业可能需要访问用户保存在另一个远程站点 B 上的文件。这样,用户的作业需要能够代表用户访问保存在站点 B 上的文件。GSI 允许用户委托一个代理凭证给在远程站点上的作业。在上面的例子中,用户在站点 A 上的远程作业就能够使用用户委托的代理凭证,代表用户向站点 B 的存储系统认证。

2.3 授权

VO 中的资源共享是通过服务实现的,控制用户对资源服务的访问权限是通过一个映射文件来实现的。映射文件由一系列用户 DN(区别名) 到本地账号(如一个本地的 UNIX 用户账号) 的映射项组成。用户认证成功后,资源提供者从用户的代理证书中提取用户 DN,然后资源提供者根据用户请求的服务从相应的映射文件中查看是否有该 DN 的映射项。如果存在,则说明用户有权限访问其请求的服务,资源提供者将以用户 DN 对应的本地账号运行被请求的服务。这样一来,用户请求的这个资源服务的行为将受到本地账号的约束。换句话说,GSI 是通过映射机制将对用户的访问控制转变为资源提供者对本地账号的访问控制,如文件访问控制、CPU 限制等。

3 组织授权服务 CAS

GSI 采用资源提供者和资源消费者之间直接的信任关系(映射文件中是否有用户 DN 的映射项) 来描述授权策略,所以对大规模的 VO,这样会有扩展性、灵活性、描述性以及缺少策略层次性的问题。例如,当一个用户加入 VO 时,可能需要同所有资源提供者交互以建立信任关系;当一个资源提供者加入 VO 时,也可能需要同所有 VO 用户交互以建立信任关系。

CAS 服务器是 VO 为解决这些问题引入的一个可信第三方,负责管理监控对 VO 中资源的访问的策略。用户要访问 VO 中的资源时,首先要同 CAS 服务器打交道,CAS 服务器根据用户的请求和其在 VO 中的角色委托一定的权限给用户。这些权限以能力(Capabilities, CAS 委托给用户的扩展的 X. 509 证书,也就是前面提到的受限代理) 的形式表示。用户可以将这些 Capabilities 出示给资源提供者获取代表 VO 的访问权限。用户最终得到的有效的权限是资源提供者授予 CAS 服务器的权限集合与 CAS 服务器授予用户的 Capabilities 中的权限集合的交集。

资源提供者和资源消费者通过 CAS 服务器建立信任关系,使得加入 VO 的实体只需要同 CAS 服务器打交道,因此扩展 VO 变得相当容易。资源提供者提供了哪些共享资源,这些资源用于什么目的等都可直接在 CAS 服务器中描述(CAS 提供了策略描述语言) ,这又使得我们可以非常方便灵活地反应 VO 策略的变化。最后,我们还可以通过建立另一个 CAS 服务器,负责管理子 VO 的策略来解决策略层次性问题。

图 1 所示为用户和 CAS 服务器、用户和资源服务器的交互过程。在图 1 中,用户为取得对 VO 中资源的一组操作权限,向 CAS 服务器请求一个 Capability。如果用户的请求同 VO 中的策略相一致,那么 CAS 就返回一个适当的 Capability 给用户,然后用户就可以使用这个 Capability 向资源服务器认证,并执行 Capability 里描述的权限。当然,前提是资源提供者已经授予这些权限给 CAS 服务器。

图 2 则表示当用户向资源服务器出示一个 Capability 时,服务器是如何决定是否允许用户对本地组织资源的访问的。资源服务器首先使用本地的访问控制机制,决定是否允许 VO 这样的请求,然后再检查 VO 委托给用户的 Capability 是否也允许这样的请求。如果两次检查都通过,资源服务器就允许执行用户的请求。

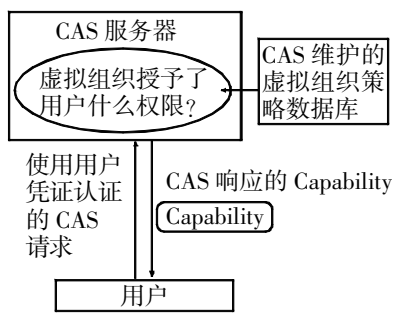

图 1 交互过程示意图

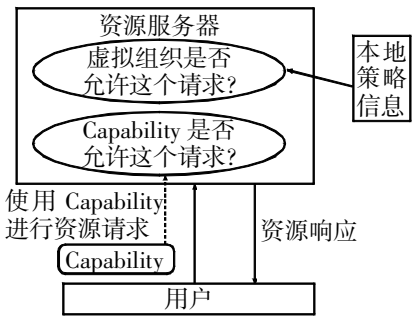

图 2 请求访问示意图

下面通过一个更具体的例子来说明 CAS 的使用过程。图 3 显示了用户通过 CAS 访问 GridFTP 服务器时典型的交互过程。

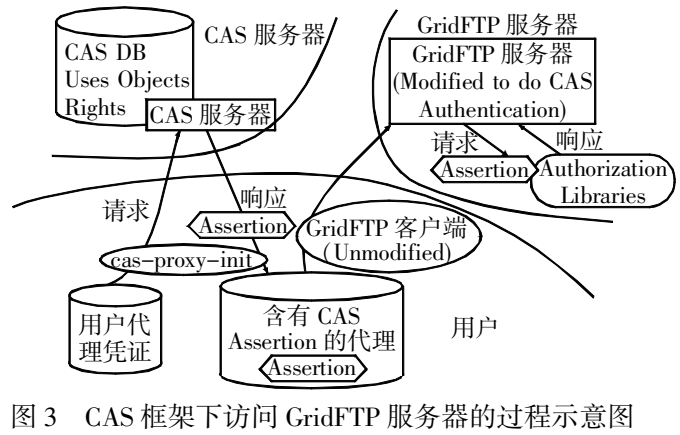

图 3 CAS 框架下访问 GridFTP 服务器的过程示意图

(1) 用户首先执行一个客户端命令 grid-proxy-init, grid-proxy-init 与用户交互,要求用户输入私钥加密密码后为用户生

成一个用户代理凭证。

(2) 用户再使用命令 cas-proxy-init 向 CAS 服务器请求一个 Capability。Cas-proxy-init 使用用户代理凭证向 CAS 服务器进行身份认证, CAS 服务器再基于在 CAS 数据库中授予用户的权限产生一个 Capability 并响应给用户。

(3) 用户在使用 GridFTP 的客户端工具访问 GridFTP 服务器的时候, 利用 CAS 服务器返回的 Capability 向资源服务器认证, 资源服务器再根据 Capability 中的策略对用户访问资源的权限进行决策。

4　方案设计

考虑这样一种情形: 当 VO 的规模非常大时, VO 的管理者可能需要管理成千上万的 VO 用户, 这对 VO 管理者的负担是非常大的。另外, 每当一个组织的用户发生变动(如组织人员增加或离开) 时, 都必须同 CAS 服务器交互, 而且每一个想访问 VO 资源的用户都必须持有用户证书。

针对这样的情形, 我们设计这样一种方案: 用户不是以独立的身份加入 VO, 而是以用户在本地组织中的角色加入 VO。这样一来, 就可以大幅度削减 VO 的用户规模, 并能够让用户受到本地策略的约束。在这个方案中, 不必为任何用户申请证书, 而只需要为想加入 VO 的本地角色向可信的第三方分别申请一个相应的证书。这样用户只需要基于本地认证就可以使用虚拟资源。

为了简单起见, 用一个主机来代表实际的组织或机构, 如图 3 中的用户主机。这里假设已经为主机上用户的每一种角色都申请了相应的证书并保存在数据库中。

4.1　认证过程的变化

前面讲到了 CAS 的使用过程当中, 用户首先要用其证书生成一个临时的用户代理凭证, 然后利用这个用户代理凭证与 CAS 服务器交互, 获取一个包含有用户权限策略的 CAS 凭证/ Capability。也就是说, 用户在访问 VO 资源时实际上是使用用户证书在进行认证。在我们的方案中, 用户并没有自己的证书, 系统中不同的证书代表的是本地组织中的不同角色。

(1) 对于单台主机, 首先是用户登录主机, 如在 UNIX 环境中以正确的账号/密码 Login。这时用户完成了本地认证, 然后可以像在访问本地资源一样地访问 VO 资源。当然, 访问 VO 资源也需要有访问权限, 就像用户访问本地资源需要有访问权限一样, 只不过 VO 资源的访问权限策略是针对本地角色的。这里有两种情形: ①用户的角色不能使用虚拟网格资源, 也就是说这个角色不参与这个 VO, 没有相应的证书; ②角色是 VO 用户, 该角色对 VO 资源的访问由 CAS 服务控制。

(2) 用户可以调用命令 grid-proxy-init-mod(使用 grid-proxy-init-mod 区别于原先的 grid-proxy-init 命令), 原先的 grid-proxy-init 命令是读取用户证书, 并与用户交互获取私钥加密密码, 然后生成用户代理凭证。但在这里没有用户代理凭证, 只有角色代理凭证。角色代理凭证对用户是透明的, 用户只需要知道本地的账号/密码, grid-proxy-init-mod 会自动从保存角色证书的地方读取与用户角色相应的证书, 然后生成角色代理凭证。

(3) 用户可以使用这个角色代理凭证来访问 VO 资源, 其使用过程与 CAS 系统的使用类似, 在图 3 中就是将用户主机上的用户代理凭证换成角色代理凭证。

4.2　CAS 代理凭证的变化

存在这样一种情形, 有多个用户以同一个本地角色访问虚拟网格资源, 而在资源提供者所在的主机那一端, 代表资源的服务是以一个本地账号在运行。在很多组织的安全策略里是禁止这种 “共享账号 ”的, 因为这样没法对具体的每个用户做审计。在这里就是只能对角色做审计。

考虑到这种情形, 需要改变 CAS 代理证书中的区别名(DN) , 比如, 原来的 DN 为 Subject: /CN = Manager Role /CN = Proxy, 现在变成 Subject: /CN = Manager Role /CN = Joe User。我们将实际用户的账号作为代理凭证中主体 DN 的后缀。这样一来资源提供者在接收用户以角色身份访问时, 就能够通过 CAS 代理判断出具体是哪一个组织的用户在充当这个角色。为了实现这个目的我们要修改 CAS 的客户端软件 cas-proxy-init 来满足我们的需求。

5　结束语

在分析网格安全机制 GSI 和 CAS 的基础上提出了一种基于 CAS 的授权方案。这个方案能够缓解在大规模 VO 中 VO 管理员的负担, 同时能够利用参与 VO 的独立组织或机构以及现有的认证机制和安全策略, 使得用户可以更加方便地使用 VO 资源。

参考文献:

[1] oster I, Kesselman C, Tuecke S. The Anatomy of the Grid: Enabling Scalable Virtual Organizations[J]. International Journal of High Performance Computing Applications, 2001, 15(3): 200-222.

[2] Pearlman L, Welch V, Foster I, *et al*. The Community Authorization Service: Status and Future[C]. CHEP, 2003.

[3] Pearlman L, Welch V, Foster I, *et al*. A Community Authorization Service for Group Collaboration[C]. IEEE the 3rd International Workshop on Policies for Distributed Systems and Networks, 2002.

[4] Foster I, Kesselman C. Globus: A Metacomputing Infrastructure Toolkit[J]. International Journal of Supercomputer Applications, 1998, 11(2): 115-129.

[5] Butler R, Engert D, Foster I, *et al*. A National-Scale Authentication Infrastructure[J]. IEEE Computer, 2000, 33(12): 60-66.

[6] Foster I, Kesselman C, Tsudik G, *et al*. A Security Architecture for Computational Grids[C]. ACM Conference on Computers and Security, 1998. 83-91.

[7] CCITT Recommendation X.509, the Directory-Authentication Framework[S]. 1988.

[8] Globus Security Policy and Implementation[EB/OL]. http://www.globus.org/security/, 1997.

作者简介:

吴毓毅(1979-), 男, 硕士研究生, 主要研究方向为分布式授权; 贺也平(1962-), 男, 信息安全技术工程研究中心主任, 研究员, 博士, 目前研究的领域为安全操作系统形式化模型、无线网络安全模型形式化分析和安全协议的形式化分析。