移动 Agent 系统中的安全问题和技术研究综述*)

张 阳 曹迎春 黄 皓 谢 立

(南京大学计算机软件新技术国家重点实验室 南京大学计算机科学与技术系 南京210093)

摘 要 移动 Agent 是分布式计算领域中一种新的计算模式。由于其在异步性、自治性以及移动性等方面的优势,移动 Agent 技术应用范围日益广泛,而随之而来的系统安全性问题也日益突出。本文对移动 Agent 系统中的主要安全威胁做了细致的分析;在此基础上,从主机和 Agent 两技术方面以及社会角度总结了目前主要存在的安全保障措施;描述并比较了几种具代表性的移动 Agent 系统及各自的安全实现机制。

关键词 移动 Agent 系统,分布式计算模式,安全机制

A Study of Key Security Threats and Countermeasures in Mobile Agent Systems

ZHANG Yang CAO Ying-Chun HUNG Hao XIE Li

(Department of Computer Science and Technology, National Laboratory of Novel Software Technology, Nanjing University, Nanjing 210093)

Abstract Mobile agent is a new kind of network computing paradigm in the field of distributed computing. Because of its distinguished advantages, such as asynchrony, autonomy, mobility, mobile agent is likely to become the dominant approach in network applications. However, with the popularity of mobile agent technology, more and more challenges come from those possible security threats in mobile agent systems. Starting from a detailed analysis of these security threats on mobile agent systems, we describe and compare the primary countermeasures for mobile agent security. Some representative mobile agent systems are introduced and their security mechanisms are evaluated.

Keywords Mobile agent system, Distributed computing paradigm, Security mechanism

1 引言

随着网络通讯技术的飞速发展,网络互联日益普及,面对网络中海量的数据信息,如何进行高效的处理和利用,研究人员提出了三种分布式网络计算模式[1]:(1)Client/Server 模式(如 CORBA, COM/DCOM 等);(2)Code-on-Demand 模式(如 Applet);(3)移动 Agent 模式。

基于 RPC(Remote Procedure Call)的 Client/Server 计算模式[2]因其结构简单,构建方便已成为当今网络计算平台上使用最为广泛的分布式计算模式,但由于其对象的"静态配置"及同步通信方式限制了 Client/Server 模式下系统的灵活性和可靠性。在 Code-on-Demand 模式系统[3]中,实现了 Java 代码的移动,具有一定网络适应性和结构灵活性,但该模式的语言局限性(目前仅限于 Java 语言)限制了此模式的进一步推广和普及。

移动 Agent 是一段程序代码,它能够在异构网络内的不同移动 Agent 平台间进行自主迁移,同时可以与其它 Agent 实体进行通讯及交互,以完成所需任务[4]。与传统的 Client/Server 和 Code-on-Demand 两种计算模式相比,移动 Agent 能减少网络负载,降低网络延迟,更好地解决网络异构问题。另外,移动 Agent 异步交互和自主迁移的特性保证了系统用户离线情况下,Agent 仍能继续完成其"使命"。正是由于其无可比拟的优势,移动 Agent 计算模式已经被用来创建包括电子商务、网络管理、分布信息抽取、移动计算以及安全代理等在内的多种动态灵活的分布式应用,成为"软件领域一次新的革命"[5]。

然而,移动 Agent 的自主性也同样给移动 Agent 系统带来了众多不安全因素。由于 Agent 可以在多个移动 Agent 平台间迁移,将可能受到恶意 Agent 平台的攻击;而未经授权的 Agent 也将可能对所登录的未受保护的移动 Agent 平台造成破坏;同时,在一个多 Agent 系统中,Agent 与 Agent 之间也存在安全威胁。以上种种安全问题都将可能对移动 Agent 系统造成致命的破坏,从而限制移动 Agent 系统在一些重要领域中的应用。

本文主要从安全性角度对移动 Agent 系统中存在的问题及现有的处理技术进行了分析和探讨。文章第2部分详细介绍了移动 Agent 系统中的主要安全问题;针对存在的安全威胁,采取何种措施和技术进行处理以最大程度保证移动 Agent 系统的安全和稳定将是第3部分讨论的重点;在第4部分,列举了几个较有代表性的移动 Agent 系统,对它们的安全保障技术进行了比较;最后做出结论,并介绍了进一步的研究方向。

2 移动 Agent 系统中的安全性问题

一个典型的移动 Agent 系统包括 Agent 本身、主机(Host)以及联结各主机间的网络。

图1展示了一个简单的移动 Agent 系统模型框架,这个简单的模型已经足以用于进行移动 Agent 系统的安全性分

*)本课题研究得到国家863高科技发展计划(编号:2003AA142010)和国家973重点基础研究发展规划(编号:2002CB312002)资助。张 阳 硕士研究生,主要研究方向:分布式系统与并行计算,移动 agent;曹迎春 硕士研究生,主要研究方向:分布式与并行计算;黄 皓 教授,博士生导师,主要研究方向:网络信息安全;谢 立 教授,博士生导师,主要研究方向:分布式系统与并行计算,网络安全。

析[6]。Agent是一组代码和状态的活动实体，主机为Agent运行和执行操作的平台，各主机通过网络互联。Agent和主机间的关系，决定了移动Agent系统中的安全问题可以从以下四个方面进行分析：Agent对Agent的威胁、Agent对主机的威胁、主机对Agent的威胁以及其它外部实体对系统的威胁。

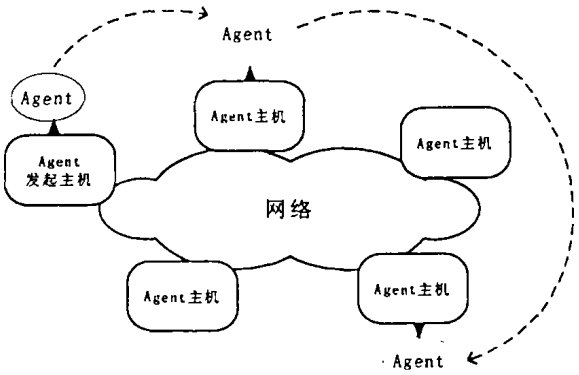

图1 移动Agent系统模型

2.1 恶意Agent对其它Agent的威胁

恶意Agent对其它Agent的威胁指某些Agent在与其它Agent进行通信或操作时，对同一主机甚至其它主机上运行的Agent进行攻击，导致这些Agent被破坏或不能正常工作，此类攻击主要包括：

伪装欺骗(Spoofing) Agent对Agent的伪装欺骗发生在Agent间直接相互通信过程中，某些恶意Agent掩盖自己的身份，伪装成其它合法的Agent，骗取其它与之通信的Agent的信任，从而获得非法利益或信息。最常见的情况发生在电子交易过程中，恶意Agent可以伪装成其它合法的服务提供商，博取客户Agent的信任，最终在交易过程中骗得客户的信用卡信息、银行账户信息或者客户其它私人信息。伪装欺骗不仅使得客户信息泄漏，资金流失，更严重的后果是造成真正合法服务提供商信誉值的降低以及客户对整个电子交易社区的不信任。

拒绝服务(DoS) 当Agent与Agent建立直接通信期间，Agent收到其它Agent的服务或通信请求时，必须做出相应的接受或拒绝处理。恶意Agent利用这个特点，当它想对某Agent发起攻击时，可以向此Agent不停地发送服务请求或者无用信息，即使此Agent发现所有请求及信息均为恶意的，做出拒绝处理，整个决定处理过程仍将占用Agent的处理时间和资源，从而导致此Agent接收其它正常请求或信息的能力下降，运行性能大大降低。

非法访问(Unauthorized Access) 所谓Agent间的非法访问，即是指在Agent没有得到其它Agent授权认证的情况下，获得其它Agent所提供的服务或调用其它Agent的公开方法，甚至对其它Agent的数据或代码进行非法修改，从而对其它Agent造成致命的破坏。由于目前特别针对Agent的访问控制机制都不完善，非法访问攻击成为Agent间攻击的常用手段。在电子交易中，恶意Agent通过非法访问修改合法客户Agent中的身份信息，使合法客户变成非法客户，或者修改客户Agent中的数据信息，造成客户电子现金的损失。

否认抵赖(Repudiation) Agent间发生了某次操作、通信或交易后，某Agent单方面否认此次操作、通信或交易的存在或者抵赖自己曾经参与本次事件过程，这称为Agent间的否认抵赖。恶意Agent可以在从其它Agent处获得服务或利益，或者对其它Agent进行恶意攻击后，完全否认整个过程，从而使其它合法Agent蒙受不必要的损失，甚至造成整个Agent系统内Agent间的不信任。

2.2 恶意Agent对主机的威胁

恶意Agent对主机的威胁指某些Agent利用Agent系统平台的安全缺陷，对所登录的主机或Agent平台进行攻击，致使整个系统主机的信息泄漏、篡改甚至导致系统瘫痪。具体攻击包括：

伪装欺骗(Spoofing) 与Agent间的伪装欺骗类似，恶意Agent可以伪装成其它合法Agent，登录合法主机，获得本该由合法授权的Agent才能获得的服务和资源。甚至在登录主机后，对主机或整个系统平台进行破坏，从而损坏了它所伪装的合法Agent的声誉和信誉。在电子社区中，恶意Agent可以伪装成高级管理级的Agent登录系统主机，使得主机上的资料信息泄漏或被篡改。

拒绝服务(DoS) 恶意Agent不停地向登录主机发送服务请求，主机必须不停地做出处理，以致主机资源耗尽，最终瘫痪或崩溃，相对于Agent间的拒绝服务攻击，Agent对主机的拒绝服务攻击更普遍。无论是在移动计算、电子商务还是其它应用中，恶意Agent往往利用主机程序漏洞，通过运行恶意攻击脚本对主机进行拒绝服务攻击，侵占主机服务资源和处理能力，导致系统性能降低。

非法访问(Unauthorized Access) 同样，如果未经授权的Agent登录了合法主机，或者某些Agent接触到了主机上级别以外不该接触的数据信息，会给Agent系统或主机留下安全隐患。Trojan木马和多种病毒程序就可以看作是恶意Agent程序对主机的非法访问。

2.3 恶意主机对Agent的威胁

恶意主机对Agent的威胁指某些主机采取某种非正常手段使得运行于其上的Agent所包含的代码或信息的正确性及保密性得不到保证，从而对整个系统的安全性产生影响。主要包括：

非法窃听(Eavesdropping) 非法窃听是网络应用中长期存在的安全问题，由于网络传输的不可靠性，通信过程中信息常被非法监听程序所窃取，造成信息的泄漏。在Agent系统中，非法窃听的问题更为突出。当Agent运行到某个Agent主机平台上后，将执行一系列的通信交互等操作，如果安全防范措施不够，一个恶意主机所能窃听的，将不仅仅是Agent所携带的数据代码信息，也包括Agent在主机上的所有活动和操作行为。恶意平台通过非法窃听，可能获得Agent的重要信息，特别在涉及电子商务的应用中，将会引起客户重要信息的泄漏。

信息篡改(Alteration) Agent总是在某个Agent主机平台上进行操作，如果恶意主机对运行于其上的Agent的代码、状态或数据信息进行篡改，将破坏Agent数据的完整性与一致性，造成Agent系统的混乱。同样在电子交易中，恶意主机可能在未经Agent授权的情况下强行修改Agent的数据库系统，或者利用Agent设计的漏洞，在Agent未察觉的情况下，对Agent携带的数据进行篡改，这都将对整个Agent系统和客户造成一定的影响。

克隆拷贝(Copy and Replay) 当一个Agent由一个主机平台移动到另一个主机平台时，主机可对Agent进行某些控制。特别地，恶意主机可能会对所登录的Agent所执行的

操作和信息进行拦截、拷贝和克隆，并在此基础上进行处理，对发起 Agent 的客户造成威胁。

此外，伪装欺骗和拒绝服务攻击，也同样是恶意主机对 Agent 的主要威胁。

2.4 其它外部实体对 Agent 系统的威胁

即使整个 Agent 系统内所有主机和 Agent 都正常运行，Agent 系统外部和内部的其它实体对 Agent 系统的威胁也不容忽视。比如目前支撑主机间通信的互联网络都没有特别的安全保障机制，传输协议也没有专门对安全性做考虑，从而，数据信息将很可能在主机间传输过程中被截取或窃听，这都将对现有的 Agent 系统的安全性造成威胁。

3 移动 Agent 系统中的安全保护措施

基于以上不安全因素，一个实用的移动 Agent 系统必须采取有效措施，提供一个结构良好、功能完善的安全保障构件，以加强移动 Agent 系统的安全性、可靠性与稳定性，保证系统不会轻易受到非法 Agent、主机或其它实体的攻击。本部分以下将主要从保护对象：Agent 和主机两个技术方面以及社会的非技术方面对移动 Agent 系统中系统安全保护措施进行介绍和分析。

3.1 针对主机的保护技术

主机是移动 Agent 执行代码的平台，对主机实施保护即是对运行于其上的移动 Agent 进行验证和选择，防止恶意 Agent 的攻击与破坏，对主机的保护将涉及到网络安全中传统的访问控制中的一些问题和技术，主要有：

沙箱技术 沙箱是一种基于软件的错误隔离方法[7]。沙箱通过软件为各 Agent 的不同执行模块开辟独立的执行域，由于 Agent 代码都是不可靠的，将它们限制在给定的虚拟地址空间中独立执行可以防止它们对其它 Agent 代码的干扰，当运行出现问题时，缩小了在整个主机系统中的影响范围。同时，每个执行域都分配有单一的标识符，通过此单一标识符，可对各 Agent 使用系统资源的情况进行有效管理。但由于执行地址空间的分离，沙箱将增加模块切换消耗，导致分布式应用执行时间的加长。

历史记录技术 历史记录技术[8]要求 Agent 能够记录所在生命周期内经过的所有主机的相关信息，当 Agent 要求登录某个主机平台时，必须向主机提供这样一份历史记录，主机以此为依据，决定 Agent 的登录权限。如果 Agent 之前登录的主机信息列表中包含有恶意主机，则可以认为此 Agent 是不可信赖的，此主机可以拒绝此 Agent 登录或对其权限进行限制，预先防止了此 Agent 对主机系统破坏的发生。但随着 Agent 路径主机的增多，历史记录条目将愈来愈长，导致登录主机前的路径认证代价愈大。

代码验证技术 代码验证[9]指 Agent 代码的编制人员通过正式的方式给出此 Agent 代码完全可靠或符合某种安全策略的保证，在 Agent 代码执行之前，主机必须对代码附带的安全保证进行验证，通过验证的 Agent 代码可认为至少对此主机不会造成危害，无需加以限制。通过附加的代码验证，可以防止恶意代码对主机的攻击，但安全保证的产生和验证难以实现。

状态评估技术 与代码验证技术类似，状态评估[10]要求 Agent 代码的编制人员在 Agent 中附带一个状态评估函数，此函数将表明 Agent 的状态情况并将状态和操作的权力范围相关联。当 Agent 登录主机前，主机可使用 Agent 提供的状态评估函数判断当前 Agent 的状态是否正常，代码是否被篡改，是否有服务或资源的使用限制。同样，由于 Agent 状态的多样性而没有统一的 Agent 状态标准，状态评估技术的实现有一定难度。

数字签名技术 数字签名技术[11]是保证 Agent 代码可靠性使用最广泛的技术之一。Agent 代码的编制者，Agent 的所有者或 Agent 代码的验证者，都可对 Agent 代码进行签名，表明了 Agent 代码的可靠性。主机根据签名的相关信息确定 Agent 相应的安全级别。由于数字签名的存在，签名人将不可对自己的签名进行抵赖，有效防止了恶意代码的产生。

安全代码解释技术 安全代码解释旨在增强异构计算机环境下对 Agent 平台的支持。Agent 主要由解释性脚本语言（如 JavaScript）或其它编程语言（如 C）编制而成，由于平台环境的异构性，在某个平台上认为安全的代码在其它平台上可能被认为存在安全隐患，从而，需要一种标准对代码的安全性做出统一的解释。像 Safe Tcl 就是特别针对 Tcl 脚本语言的安全解释器。

3.2 针对 Agent 的保护技术

移动 Agent 包含有可执行的代码和数据，虽然 Agent 中代码的执行有一定的自主性，但相对于主机仍处于被动地位，执行受到主机行为的约束，从而保护 Agent 不受攻击比较困难。目前主要提出的保护技术包括：

限时黑箱技术 为了防止 Agent 信息泄漏，将原始 Agent 代码依据某种混淆算法打乱，形成一个与原始 Agent 功能属性完全相同的可执行 Agent 副本。相同的输入，此副本 Agent 能产生相同的处理输出，但如同一个黑箱，攻击者难以了解其内部情况，从而使得企图对 Agent 代码进行恶意攻击的对象在短期内无法对杂乱的代码进行识别[12]。经过限定的时间后，所有代码数据都将更新，黑箱中原有的过期信息全部失效，这样即使攻击者将原有代码还原，得到的也均为无效信息，有效防止了 Agent 数据信息的泄漏。由于目前存在有针对黑箱的测试攻击，通过不断地对输入和输出结果进行分析比较，仍能获得 Agent 的某些特性。

执行跟踪技术 执行跟踪技术[13]主要用于发现 Agent 在其所在的主机平台上未经授权的修改。通过读取各主机上建立的 Agent 活动日志，Agent 可以方便地对自身执行的操作和状态进行跟踪，当发现有 Agent 代码或数据信息未经自身授权而被修改时，可以及早发现并进行相应的处理。执行跟踪技术中同样存在当 Agent 数量增多后各主机上日志量急剧增大的问题，而且随着 Agent 活动增多，自身日志的查询工作也相当繁琐。

错误恢复技术 当 Agent 受到攻击不能正常工作后，如何提高系统的容错能力，并能在最短的时间内恢复 Agent 的正常工作，这就是错误恢复技术需要考虑的问题[14]。最常用的方法是定期对 Agent 进行复制，通过 Agent 备份的方法防止单个 Agent 拷贝丢失，一旦当前 Agent 出错，对其进行回收，并使用备用的 Agent 继续执行其任务。选择适当的时机对 Agent 进行复制以及原始和备份 Agent 执行任务的传递，是错误恢复技术中的难点。

数据加密技术 数据加密技术与限时黑箱技术类似，通过使用给定的数据加密算法对 Agent 代码中的数据信息进行加密，使得 Agent 即使运行在不可靠的环境下仍不用担心信息的泄漏。加密函数的选取是加密技术的关键，但选择一个容易实现而难以破解的加密函数十分困难。部分结果认证[15]

是从加密的分步来考虑加密的有效性，在 Agent 的发起主机上产生一系列密钥，每到达一个主机，将为其分配一个专用密钥以对所使用的加密数据进行解密，当要离开此主机时，此专用密钥将被销毁，从而保证了即使在整个执行过程中受到恶意主机的攻击，Agent 仍能确保受到攻击之前的结果的完整性和保密性。

除此之外，数字签名技术也同样适用于对 Agent 的保护。

由于实现技术上的难度，特别针对 Agent 的保护目前还仅处于研究阶段，在具体的移动 Agent 系统平台中所实现的针对 Agent 的保护机制还相当不完善。

3.3 其它非技术保护

除了从技术角度对移动 Agent 系统的主机和 Agent 实施保护以外，还可以从其它非技术角度（如社会立法等）考虑对系统安全性的保护，通过强制手段对恶意投放破坏性 Agent 或建立非法 Agent 主机的第三方进行惩罚，这都需要社会法律法规的支持。

3.4 安全保障措施比较

基于以上分析，可以得到在移动 Agent 系统中各安全保障措施的简单比较，如表1所示。

表1 移动 Agent 系统安全保障措施

保障措施	保护对象	保护方式	缺点
沙箱	主机	错误隔离	模块切换开销，执行时间长
历史记录	主机	根据 Agent 历史记录决定登录权限	认证开销
代码验证	主机	预先验证代码安全性	安全保证规范难以统一
状态评估	主机	通过状态评估函数决定 Agent 的执行状况和权限	无统一的 Agent 状态标准
安全代码解释	主机	通过安全代码解释器消除异构平台下安全隐患	缺乏代码安全的统一性标准
数字签名	主机（或 Agent）	第三方签名保证 Agent（或主机）的可靠性	不能防止 Agent（或主机）信息的泄漏
限时黑箱	Agent	基于时效的 Agent 黑箱	存在专门的黑箱测试攻击
执行跟踪	Agent	通过活动日志跟踪操作和状态	日志查询开销
错误恢复	Agent	Agent 副本拷贝	复制时机难以掌握，任务传递复杂
数据加密	Agent	加密 Agent 数据防止信息泄漏	加密认证开销，高效加密函数选取困难
法律法规	Agent，主机	社会强制手段	需要国家法律的支持

4 移动 Agent 实例系统安全性分析

至今为止已成型的移动 Agent 系统已有多种，本部分将主要从安全角度出发，对其中较有代表性的几种系统进行分析和比较。

4.1 Telescript[16]

Telescript 是 General Magic 公司提出的第一个商用移动 Agent 系统，它包括一组面向对象的专用 Agent 编程语言。在 Telescript 系统中，提供了指令级的强移动机制。同时，系统通过一套访问控制机制来保证系统的安全性。每个 Agent 和主机通过授权（authority）建立相互关联，并基于授权进行访问控制决定。

4.2 Tacoma[17]

Tacoma 为挪威 Tromsϕ 大学开发的旨在提供移动代码中间件支持的移动 Agent 系统。Tacoma 系统支持多种语言（包括 C，Perl，Python，Tcl 等）编制的 Agent，具有面向 Unix、Windows 以及 PalmOS 等操作系统环境的多种系统版本。在 Tacoma 系统中仅有简单的访问控制，而未实现其它安全控制机制。

4.3 Agent Tcl[18]和 D'Agent[19]

Agent Tcl 是 Dartmouth 学院研制的一种基于 Tcl 语言的移动 Agent 系统。Tcl 脚本可以在不同主机间迁移、执行和通信，并可进行状态查询等工作。Agent Tcl 系统通过安全 Tcl 环境支持资源的限制访问，从而防止恶意操作的执行。另外，通过外部程序来实现数据加密传输。D'Agent 是 Agent Tcl 的后续扩充系统，它提供了一个简单、开放的层次体系结构，并以 Java 虚拟机支持基于 Java 移动 Agent 的强迁移。

4.4 Aglets[1]

Aglets 系统是由 IBM 日本研究院开发的基于 Java 的移动 Agent 系统。Aglets 为用户提供了图形操作界面，是目前最流行，使用最广泛的移动 Agent 系统之一。在 Aglets 中采用事件机制来增强 Agent 的移动能力，通过事件的触发回调来保证系统的安全性。然而，由于缺少访问控制，用户间的事件回调容易混淆，一个用户可能会回调另一用户的 Agent 而造成一定的安全隐患。

4.5 Jumping Beans[20]

Jumping Beans 系统是美国 Ad Astra 工程技术公司设计的一个支持可移动 Java 应用程序构造和运行的系统平台。通过 Jumping Beans 系统，移动应用不需要预先安装在各客户设备上，系统主机会自动将所需代码数据资源迁移复制到目的主机上。Jumping Beans 系统提供了完整的四层安全保障体系结构，当有恶意程序攻击其中任意一层时，其它三层将依旧共同对系统实施保护。Jumping Beans 系统中每个移动 Agent 均附带有一个安全事件日志，用于记录此 Agent 生命周期内的所有活动。此外，Jumping Beans 系统还使用特有的智能认证技术来防止系统数据丢失或被窃取。

4.6 Mogent[21]

Mogent 是由南京大学计算机软件新技术国家重点实验室开发的基于分层多模式通信框架的纯 Java 移动 Agent 平台系统。在 Mogent 系统中，不仅提供了移动迁移机制、通信机制和安全机制，还建立了一整套 Mogent 的开发、运行和监控环境，方便用户在此平台上进行应用程序的开发和实际的运用，并提供了电子商务的示范应用。Mogent 系统通过数字签名来保证用户身份的合法性，通过安全传输通道、授权和访问控制、电子货币建立了 Mogent 系统的安全支撑。但由于技术难度，在 Mogent 系统中未实现对移动 Agent 本身的安全保护支持。

4.7 安全机制比较

由于各移动 Agent 系统建立时间、技术背景以及应用领域的差别，各系统在安全机制的实现和侧重上也不尽相同，表2给出了各系统在安全体系结构实现方面的比较。

表2 移动 Agent 系统安全机制比较

系统	认证	传输通道	数字签名	对主机的保护	对 Agent 的保护
Telescript	基于 RSA	RC4加密	不支持	通过授权进行访问控制	不支持
Tacoma	不支持	无	不支持	简单的访问控制	不支持
Agent Tcl	基于 PGP	PGP 加密	支持	Agent 执行在 Safe Tcl 环境下;根据安全策略实施粗粒度的访问控制	不支持
Aglets	基于域	对称算法	不支持	通过代理对象实施访问控制	不支持
JumpingBeans	智能认证技术	SSL 协议	支持	四层安全保障体系	Agent 安全事件日志
Mogent	X.509	类似 SSL 的协议	支持	基于信任传递的授权并通过授权进行访问控制	不支持

结论 随着移动 Agent 应用领域的日益广阔,移动 Agent 系统中的安全问题也日益突出。传统的面向主机安全的保护机制主要关注于保证 Agent 平台的安全稳定,而在移动 Agent 所能执行任务复杂化的同时,面向 Agent 的保护技术也逐渐提出并迅速发展。

本文对移动 Agent 系统中存在的安全问题进行了详尽的分析和讨论;并在此基础上总结了现有的特别针对主机和 Agent 的安全保护技术;同时介绍了几种较有代表性的移动 Agent 系统实例,对它们的安全实现机制进行了比较。

不同的 Agent 应用具有不同的安全实现要求,从而需要有选择地对各种安全机制进行高效配置。如何构造一个适用于多种应用的灵活的移动 Agent 系统安全框架,对于移动 Agent 系统中的安全性问题建立一种通用的解决方案,以及如何增强系统中针对 Agent 的保护机制的研究,将是下一阶段工作的重点。

参 考 文 献

1 Lange D B,Oshima M. Progamming and Deploying Java Mobile Agents with Aglets. Addison-Wesley,USA,1998
2 Paul M. Three Tier Client/Server Systems: Building Distributed Systems. Goteborg University Press,Sweden,2001
3 Lange D B. Mobile Objects and Mobile Agents: The Future of Distributed Computing. In:Proc. of European Conference on Object-Oriented Programming,Brussels,Belgium,1998
4 Cao J,Zhou J,Chen D,Chan T S. Mobile Agent Technology and Its Application in Internet Computing. an invited book chapter of "Computational Web Intelligence: Intelligent Technology for Web Applications",World Scientific Press
5 Guilfoyle C,Warner E. Intelligent Agents: The New Revolution in Software. [Technical Report]. Ovum Ltd,London,UK,1994
6 Wayne J,Tom K,Mobile Agent Security. NIST Special Publication 800-19,National Institute of Standards and Technology,Aug. 1999
7 McGraw G,Felten E. Understanding the keys to Java security:the Sandbox and authentication. JavaWorld,May 1997,2(5)
8 Chess D,Grosof B,et al. Itinerant Agents for Mobile Computing. In:Proc. of IEEE Personal Communications,Oct. 1995,2(5):34～49
9 Necula G C. Proof-Carrying Code. In:Proc. of Conf. Record of POPL '97: The 24th ACM SIGPLAN-SIGACT Symposium on Principles of Programming Languages,1997. 106～119
10 Farmer W,Guttman J,Swarup V. Security for Mobile Agents: Authentication and State Appraisal. In:Proc. of the European Symposium on Research in Computer Security (ESORICS),1996. 118～130
11 Pointcheval D,Stern J. Security Arguments Proofs for Digital Signatures and Blind Signatures. Journal of Cryptology,2000. 361～396
12 Hohl F. Time Limited Blackbox Security: Protecting Mobile Agents From Malicious Hosts. Mobile Agents and Security,LNCS,1998,1419:92～113
13 Vigna G. Cryptographic Traces for Mobile Agents. Mobile Agents and Security,1998. 137～153
14 Silva F A,Popescu-Zeletin R. An Approach for Providing Mobile Agent Fault Tolerance. In:Proc. of Second Int. Workshop on Mobile Agents,1998. 14～25
15 Yee B. Monotonicity and Partial Results Protection for Mobile Agents,In:Proc. of the Intl. Conf. on Distributed Computing Systems,2003. 582～591
16 Tardo J,Valente L. Mobile agent security and Telescript. In:Proc. of the 41st Intl. Conf. of the IEEE Computer Society (CompCon '96),1996. 58～63
17 Johannsen D,Renesse R V,Schneider F B. An Introduction to the TACOMA DistributedSystem:[Technical Report 95-23]. Department of Computer Science,University of Tromsϕ,Norway,1995
18 Kotz D,Gray R,Nog S,et al. AGENT TCL: Targeting the Needs of Mobile Computers. IEEE Internet Computing,Jul. 1997. 58～67
19 Gray R,Cybenko G,Kotz D,et al. D Agents: Applications and Performance of a Mobile-Agent System. Software-- Practice and Experience,2002,32(6):543～573
20 http://www.jumpingbeans.com
21 陶先平. 基于 Internet 的移动 agent 技术和应用研究:[南京大学博士学位论文]. 南京大学,2001
22 Borselius N. Mobile agent security. Electronics & Communication Engineering Journal,IEE,London,UK,Oct,2002,14(5):211～218
23 Corradi A,Montanari R,Stefanelli C. Security Issues in Mobile Agent Technology. In:Proc. of 7th IEEE Workshop on Future Trends of Distributed Computing Systems FTDCS'99,Dec. 1999. 3～8
24 Wayne J. Countermeasures for Mobile Agent Security. Computer Communications,Special Issue on Advanced Security Techniques for Network Protection,Elsevier Science BV,2000. 1667～1676

(上接第16页)

12 Pour G,Component-Based Software Development Approach: New Opportunities and Challenges. In:Proc. of the 26th Intl. Conf. on Technology of Object-Oriented Languages and Systems (TOOLS),1998
13 Larsson M,Crnkovic I. Development Experiences of a Component-based System. In:Proc. of Engineering of Computer Based Systems (ECBS 2000),IEEE,2000
14 OMG CORBA. http://www.corba.org,2001
15 Box D. Essential COM. Addison-Wesley, ISBN0-201-63446-5, 1999
16 Sun. The Java Language,Java Beans. at http://java.sun.com, Sun Microsystems
17 Kiczales G,et al. Aspect-Oriented Programming. In: Proc. of the European Conf. on Object-Oriented Programming (ECOOP). Springer-Verlag,Finland,1997
18 Czarnecki,Eisenecker U. Template- Metaprogramming. http://home.tonline.de/home/Ulrich.Eisenecker/meta.htm
19 Charles S. The death of computer languages- The birth of intentional programming:[Microsoft Research technical report MST-TR-95-52]. Sep. 1995
20 Charles S. Intentional programming-innovation in the legacy age. Presentation at IFIR WG 2.1 on Algorithmic Language and Calculi. June 1996
21 http://www.generative-programming.org
22 http://www.program-transformation.org/twiki/bin/view/Gpce