计算机科学 2000Vol. 27№. 11

移动计算安全性*)

Mobile Computing Security

胡 健 刘锦德

(电子科技大学计算机科学与工程学院 成都 610054)

Abstract In the first, security issues in open system that supports mobile computing are discussed in detail, and then the fundamental principle for building a security system in the environment of mobile computing is given. According to the principle, security issues and policies related to mobile code programming language and mobile agent system are further discussed.

Keywords Security, Mobile computing, Open system, Java, Mobile agent

1 引言

安全性问题始终是开放系统中的一个核心问题，为此，国际标准化组织 ISO 曾对 OSI 环境(开放系统环境)的安全性作过深入的研究，并为其提出了安全体系的概念集[1]。然而，随着移动计算技术的出现和广泛的应用，许多新的安全性问题出现了，给现有的操作系统、分布式系统管理、程序设计语言和中间件技术都带来了一系列新的课题，传统的开放系统安全体系结构，面临许多新的挑战。移动计算环境下的开放系统安全性，已成为目前国外学术界的一个研究热点，外军军方也给予了高度重视[2]。为此，我们开展了对该课题的研究工作。

本文首先对传统的安全体系结构进行了分析，指出了在移动计算条件下所面临的一系列新的问题，同时，对移动计算的重要环境因特网现有的安全技术进行了研究，针对移动计算的特殊安全要求，通过对国外最新的安全体系的跟踪研究和比较分析，结合我们对开放系统全局安全性的研究，提出了在移动计算条件下建立开放系统安全体系的基本要求，并以此为基础，从不同层次和角度对移动代码程序设计语言安全性和移动 agent 安全性这两个问题进行了讨论。

2 移动计算环境下的开放系统安全性

2.1 传统的分布式系统安全性

为了设计和实现一个安全系统，首先应探明系统将可能受到的威胁。只有知道了系统将受到的威胁以后，才能对其进行有效的防范。国际标准化组织 ISO 曾对 OSI 环境(开放系统互连环境)的安全性作过深入的研究，并为其提出了安全体系[1]。尽管开放系统环境与 OSI 环境并不能划等号，但我们认为 ISO 对破坏安全系统安全性的各种威胁种类所作的概括是非常准确的，具有相当的普遍性，可以为我们具体分析移动计算环境下的开放系统安全体系结构将面临的新的威胁、提供有益的指导。ISO 认为破坏系统安全的威胁主要有四种类型：中断(Interruption)、窃取(Interception)、更改(Modification)和伪造(Fabrication)。这四者的威胁对象通常为信息(包括数据)或服务(包括程序)。四者的作用可以抽象地表示为如图 1 所示。这些威胁往往是由威胁源(入侵者或其入侵程序)利用系统中的脆弱环节进行入侵而产生的。

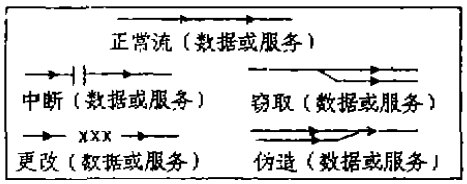

图 1 破坏安全的四种类型

ISO 对 OSI 环境所定义的威胁，其侧重面是通信系统，ISO 安全体系的目标是使互联系统中的进程之间能进行安全通信，为此它建议使用五种安全服务(对象认证、访问控制、数据保密性、数据完整性和防抵赖)和若干安全机制来保证应用进程之间所交换信息的安

*)本研究课题得到国防预研基金和国家教委博士点基金的资助。胡 健 博士生，主要研究方向：分布式对象技术，分布人工智能和多媒体技术。刘锦德 博士生导师，主要研究方向：开放式系统技术，分布式多媒体技术等。

全。

显而易见，以上的安全体系结构是侧重于OSI的下层协议的安全性，为此，ISO又陆续为OSI上层协议的安全性和系统管理中的安全性问题制定了系列标准[3~6]。

但是以上新标准都是基于传统的客户/服务器计算模式所制定的，而对于移动计算模式，由于代码是进入到目的系统之后运行的，因而，对开放系统有着新的安全性要求。

2.2 Internet安全性

由于Internet的广泛的用户基础和相对于专用网络的极低的使用费用，使得它获得了广泛的应用，但是，我们也应清楚地认识到，相对于传统的专用网络，Internet缺乏一个提供全局安全性的系统管理基础结构，这使得人们在分享Internet所提供的便利的同时，却不得不为所要求的安全性付出很高的代价。

对于Internet、提供全局安全性的系统管理的责任，完全移到了用户自身、使用Internet进行互联的各企业，为了自身的安全，其Internet服务器必须使用提供安全性的软硬件、防火墙技术和加密方法。

但是，研究发现，尽管防火墙模型在商业上相当成功，但它犯了一个根本性的错误[7]：它假设在网络内部运行的所有程序只能根据内部用户请求而行动，并且通过防火墙的所有数据是不会产生其它后果的。假如通过Internet传递的信息只是简单的ASCII码文本，显然不会有任何问题，即便是使用FTP下载一个二进制文件、要运行的话，也必须由内部用户显式地安装和执行，其危险性也相对较低。但是，现在的Internet用户所下载的文档，常常包含有主动的内容[8]，在这样的文档中包含有诸如"scripts"，"applets"，"custom controls"，"plugins"等内容、从根本上讲，它们都属于移动代码系统的范畴，并且都违反了防火墙模型所作的假设。现在，程序可以作为电子邮件或浏览器页面的附件到来，并且可以自行安装，通过鼠标点击就能启动运行。以前，一个防火墙可以假设攻击只能来自外部，现在，对于移动代码来说，也可以从内部进行攻击，对此，防火墙不能提供任何保护。

同样，密码系统对于移动代码的安全性也只能提供很有限的帮助。通过使用数字签名技术，可以确定移动代码的码源和保证接收到的移动代码在传输中未被篡改，但对移动代码在执行过程中所产生的结果却不能提供任何保证。

2.3 移动计算安全性

2.3.1 *研究领域* 通过以上的分析，我们发现，现有的安全体系和技术，对于移动计算安全性很难提供全面有效的支持，为建立移动计算环境下的开放系统安全体系，必须从操作系统、分布式系统管理、程序设计语言等多个角度，对移动计算进行深入研究，本文对移动代码程序设计语言安全性和移动agent系统安全性两个方面进行了讨论。

2.3.2 *安全系统的基本原则* 建立移动计算环境下开放系统安全体系，必须首先确立建立一个移动计算环境下的安全系统的基本原则。1985年12月，美国国家计算机安全中心发布了DTCSEC(DoD Trusted Computer System Evaluation Criteria)，1994年8月，美国国防部(DoD)发布了新的信息系统安全框架TAFIM(Technical Framework for Information Management)[10]，1996年4月30日，发布了到目前为止最新的版本，称为DGSA(DoD Goal Security Architecture)[11]，以上三份文件，对开放系统安全体系产生了广泛的影响、围绕DGSA的各种讨论和研究，依然是目前计算机安全领域一个热门的话题[2,9]。通过跟踪研究和分析比较、结合我们对开放系统全局安全性所作的研究，可以确定在移动计算环境下建立一个安全系统的基本原则和要求为：

- 系统必须具备定义明确的安全策略。系统必须使用一套规则，用于管理对于敏感信息的访问权限。
- 访问控制的标记必须与有关信息对象结合在一起。必须对系统中的每一个信息对象附加一个访问控制标记。
- 必须对每一个访问者进行认证。用于认证的所有信息必须安全地加以保存。
- 系统必须具备审计能力。审计信息必须选择性地加以安全地保存，以便对于影响到系统安全性的行为、能够通过审计信息追查到责任方。
- 系统必须具备一套独立的机制用以评估系统是否能够满足以上的安全性要求。
- 用以实现以上安全要求的系统软硬件资源必须有效地加以保护。

以上原则，是保证开放系统安全性的基本要求，在移动计算环境下依然可以作为指导原则。

3 移动代码程序设计语言安全性

移动代码(mobile code)是指可以在跨越不同的安全域的异质网络上移动，并且在到达目的地后可以自动执行的任何软件，它并不局限于Java程序。但是随浏览器技术和Java语言的迅速发展和紧密结合，二者极大地推动了移动代码技术的发展，使之成为了Internet上不可缺少的计算模式，Java也成为了最为重要的移动代码程序设计语言，与此同时，在这种新的计算模式下，系统所面临的一系列新的安全性问题也日益凸现。关于Java语言的安全性问题，我们将重点分

析其可下载执行的代码(即 Java Applets)的安全性问题,并将对 Java 安全模型进行讨论。

3.1 移动代码的安全性问题

对于传统的应用程序,在系统中运行时,将获得系统中某些资源的访问权限。类似地,可下载执行的代码也可以获得这些资源的访问权限。在获得资源的访问权限后,对于传统的应用,便可以合理地使用它们,这是因为我们认为我们启动执行的应用程序是可以信赖的(在 UNIX 系统中可以使用 TCB 机制加以保证);对于移动代码来说,情况就不同了,因为我们无法确定其执行的结果会对系统造成什么影响,嵌入到 Web 页面的敌意的 Applets 会攻击 Web 用户的系统,用户可能因此而蒙受损失。在本文 2.1 节中提到:为了设计和实现一个安全系统,首先应探明系统将可能受到的威胁,为此,我们首先对敌意的 Applets 可能实施的攻击进行了分类[12]:

· 攻击系统的完整性。非法删改系统文件、修改当前内存、杀死运行中的进程或线程是这类攻击的后果。

· 窃取用户的机密,通过网络将用户个人或公司的数据送往其它系统,是这类攻击的惯用方法。

· 限制系统资源的可获得性。通过创建数量巨大的窗口或高优先级的进程以占用大量的 CPU 时间、分配大量的内存、使用完可用的全部文件指针从而大大降低系统的可获得性甚至使系统瘫痪,是这类攻击的常用方法。

· 干扰用户的正常工作。通过施放 Trojan 木马,破坏正常的安全性、或通过显示不健康的图片以及播放讨厌的声音以干扰用户是该类攻击的惯用手段。

由以上分析可知,为了保证系统安全性,必须引入相应的约束机制,以制约 Applets 的行为,同时应保证所引入的约束机制不会限制正常功能的实现,下面,我们将对现有的 Java 安全机制进行分析,并将根据 2.3.2 节中所提出的安全系统的基本原则进行对比分析。

3.2 Java 安全模型分析

Java 语言最重要的一个设计目标便是它的可移植性,这使得 Java 语言具有结构中立性和平台独立性的特点。更准确地说,通过介于本地的操作系统和 Java 应用程序之间的 Java 虚拟机(JVM),可以使得同样的 Java 二进制程序(Java Bytecode)可以运行在许多不同的系统之上。正是由于其独特的可移植性使得 Java 语言成为最为理想的移动代码程序设计语言。

Java 安全模型[13]的中心是保护用户不受下载的有敌意的 Applets 的攻击。为实现这一目标,通常的做法是将 Applets 的活动限制在 Web 浏览器的一个专门的区域内,通常被称为沙箱(sandbox),Java 安全模型也常被称为沙箱模型。

在其 sandbox 内,Applet 可以做任何它想做的事,但是不允许访问用户的文件系统、网络连接和其它系统资源。例如,通过其 sandbox,对于一个不可信的 Java Applet,以下操作将被禁止:读写本地硬盘;除该 Applet 的宿主机外,和任何其它主机建立网络连接;创建新进程;加载新的动态连接库或直接调用本地方法。

为实现这样一个 sandbox 模型,既要用到 Java 语言和 JVM 所提供的基本的安全性支持,也要用到一个被称为 Applet 安全管理器的抽象类。Java 安全模型也可以通过从支持 Java 的 Web 浏览器下载 Applet 到允许它执行的整个过程来加以说明:

· ClassLoader 从网上下载 Applet 字节码数据流并负责把它转换为表示 Applet 类的内部数据结构。

· 在由 JVM 的运行时系统(runtime system)运行新下载的 Applet 之前,ClassLoader 调用字节码检验器(Bytecode Verifier)对新下载的 Applet 类进行合法性检查。

· 无论何时,当 Applet 试图执行一个有可能破坏本地主机或访问控制信息的操作时,JVM 都将首先询问安全管理器该操作是否能安全地执行。如果安全管理器许可,JVM 就将执行该操作,否则,JVM 就会产生一个安全异常,并会自动向 Java 控制台写一个错误信息。安全管理器是在浏览器启动时启动的,并且不允许下载的 Applet 对其进行修改。

3.3 Java 安全模型分析

尽管利用 Java 安全模型可以提高系统的安全性,特别是提高系统抵御敌意的 Applets 攻击的能力,但是我们认为单纯利用该模型并不能使系统达到高安全性的要求,这是因为 Java 系统中存在以下几个重要的缺陷:

· 现有的支持 Java 的浏览器,如 Netscape Navigator、Microsoft Internet Explorer 都存在着各种各样的安全性问题[14],并且都没有正式定义任何的安全策略,这与安全系统的基本原则第一条,即"系统必须具备定义明确的安全策略"相抵触。

· Java 系统没有定义任何审计能力,这与安全系统的基本要求的第四条,即"系统必须具备审计能力"相抵触。

· 目前的 Java 运行时系统有很大部分的代码是用 C 语言写成的,这意味着难以发挥 Java 语言所固有的内存保护的能力,使得 Java 运行时系统易受到缓存溢出(buffer overflow)攻击。

· 现有的支持 Java 的浏览器,没有提供防御单纯对系统的可获得性进行攻击的能力。

基于以上分析，我们认为，对于一个具有较高安全性要求的系统，不能将其安全性建立在完全依赖 Java 沙箱模型的基础上，必须在操作系统和浏览器之间建立一道安全屏障，以便对整个 JVM 进行安全封装。许多国外的系统亦采用了类似的方法[15]。

下面，我们将对开放系统另一种新的计算模式，即移动 agent 的安全性问题进行讨论。

4 移动 agent 系统安全性

4.1 agent 系统可能面临的安全威胁

agent 是代表个人或组织的可自主行动的计算机程序。移动 agent 是指可以离开开始其执行的系统，在网络上不同系统间移动的 agent。agent 系统是能够创建、执行、传送和终止 agent 的一个系统。移动 agent 之间的交互作用，是指一个 agent 进入到它欲与之交互的另一个 agent 所在的 agent 系统之中，彼此调用对方所提供的操作的过程。

因为移动 agent 是能够在不同的 agent 系统间移动的计算机程序，如果不能对其进行有效的安全管理，它甚至可能象网络病毒一样对系统展开攻击，由此可见安全性对移动 agent 系统是多么重要。

现有的移动 agent 系统，由于考虑到 agent 移动性的要求，通常都采用支持移动代码的语言（例如 Java 和 Tcl）作为它们的编程语言，所以有关移动代码程序设计语言安全性的研究可以为我们进行移动 agent 系统安全性研究提供基础。事实上，现有的基于 Java 的移动 agent 系统基本上都采用了 Java 的沙箱安全模型作为其安全机制实现基础。但是，通过以上的分析，我们知道 Java 的安全模型本身就存在不完善的地方，而且移动 agent 系统对安全性有着特殊的要求，系统地进行移动 agent 系统安全性研究，对于建立一个具有较高安全性要求的支持移动 agent 计算模式的开放系统，有着重要的意义。

对于一个 agent 系统，可能面临的安全威胁可以归类为：

- 拒绝服务（Denial of service）、阻止合法的用户访问 agent 或 agent 系统。
- 非法访问和使用 agent 或 agent 系统所提供的操作。
- 非法修改和破坏 agent 或 agent 系统的数据信息。

攻击者可以运用的手段包括伪装身份（Masquerade）、施放特洛伊木马、重放（Replay）、窃听（Eavesdropping）以及异常频繁的访问（Spamming）等。

4.2 对付安全威胁的策略

为保证 agent 系统的安全性，必须为 agent 制定一组安全规则，以约束 agent 的活动。agent 系统的安全策略包括：

- 系统可以对 agent 特定的能力进行限制或授权 进行限制或授权的 agent 能力包括创建新 agent、进行移动等。
- 设置 agent 资源使用限制 agent 系统的系统管理部件应能够对 CPU 使用率、内存和磁盘用量、新创建的 agent 数量以及允许的网络连接数量进行限制。
- 设置 agent 的访问控制权限 agent 系统的系统管理部件应能够对决定是否允许其它系统中的 agent 进入到系统内部以及该 agent 所能调用的操作和能访问的数据资源。

以上的安全策略应利用 agent 编程语言和运行时系统具体实现。agent 系统应提供以下安全服务：认证服务；完整性服务；保密服务；重放检测。

4.3 具有安全性的移动 agent 系统结构

通用的基于 Java 的移动 agent 系统结构[16]由六个主要的部件组成：agent 管理器、agent 间通信管理器、安全管理器、可靠性管理器、目录管理器和应用网关，以支持可靠、安全的 agent 应用（参见图 2）。

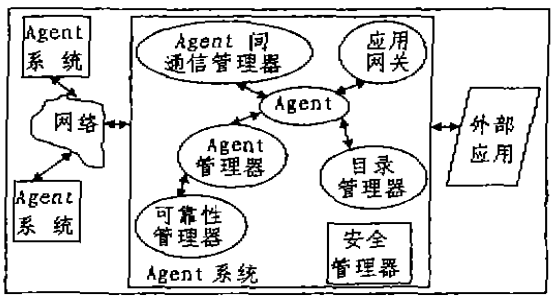

图 2 通用移动 agent 系统结构示意图

agent 管理器负责向其它系统发送本地的 agent，或从其它系统接收将在本地执行的 agent。在发送 agent 之前，agent 管理器要对 agent 的代码和状态进行序列化。对于一个可靠性的系统，agent 管理器实际上是将 agent 送往可靠性管理器，后者将确保 agent 被目的系统可靠接收。

当接收到一个远地 agent 时，agent 管理器负责对 agent 和它所引用的对象进行重构，并创建该 agent 执行所需的环境上下文。在允许该 agent 执行前，安全管理器负责对它进行认证。这之后，对于任何使用系统资源的操作（如读写文件），移动 agent 运行时系统都将自动调用安全管理器进行授权。

agent 间通信管理器提供多个相互协作的 agent 之间的通信支持。移动 agent 系统通常采用消息机制

或分布式事件机制作为基本通信机制。

安全管理器负责对主机和 agent 系统进行保护。agent 系统中的其它所有部件都必须与它交互，以便对移动 agent 进行认证和授权。在传送 agent 之前，安全管理器也可以对它进行加密。在具有高安全性要求的情况下，安全管理器还可以对 agent 进行数字签名，各 agent 系统可以通过交换密钥彼此进行认证。

除安全管理器外，应用网关也担当着 agent 系统的一个安全入口点的功能，agent 必须通过它才能与各应用服务器（例如数据库）进行交互。移动 agent 可以利用目录管理器确定应用服务器的地址，从而确定它所要移向的目的 agent 系统所在的主机地址。

以上的通用移动 agent 系统结构可以较好地满足移动 agent 的应用要求，也基本上能满足系统的安全性要求。但是，agent 系统的 agent 编程语言和运行时系统必须克服 Java 等移动代码程序设计语言所固有的安全缺陷，才能达到较高的安全性要求。

基于更为成熟的分布式对象技术而实现的 agent 系统，如基于 CORBA 的移动 agent 系统实现[17]，则对于解决不同的移动 agent 系统间的互操作问题更为有利。同时，可以利用 CORBA 的对象安全服务实现系统的全局安全性要求，对此本文不再详细讨论。

结束语 移动计算技术是分布式计算领域一个方兴未艾的研究领域，其安全性问题对于该技术的推广和应用具有极为重要的意义。本文从开放系统的角度对该问题进行了剖析，希望引起国内学术界的高度重视。

致谢 特别感谢澳大利亚 Deakin 大学计算机学院 Associate Professor、电子科技大学客座教授周万雷博士与作者就本文研究内容进行的富有启发性的学术交流。

参 考 文 献

1 International Standard Organization (ISO). Information Processing Systems-OSI RM, Part2. Security Architecture. ISO/TC97 7489-2, 1988

2 Feustel E A, et al. The DGSA-unmet information security challenges for operating system designers Operating Systems Review, 1998, 32(1): 3～21

3 International Standard Organization (ISO). Information Technology-Open Systems Interconnection-Systems Management: Security Audit Trail Function ISO/IEC 10164-8: 1993

4 International Standard Organization (ISO). Information Technology-Open Systems Interconnection-Upper Layers Security Model ISO/IEC 10745, 1995

5 International Standard Organization (ISO). Information Techology-Open Systems Interconnection-Generic Upper Layers Security. Overview, Models and Notation. ISO/IEC 11586-1·1996

6 International Standard Organization (ISO). Information Technology-Guidelines for the Management of IT Security-Part I Concepts and Models for IT Security ISO/IEC TR 13335-1·1996

7 Segev A, et al. Internet security and the case of bank of America. Communications of the ACM, 1998, 41(10)·81～87

8 Sibert W O. Malicious data and computer security. In: 19th National Information Systems Security Conference. Baltimore, Maryland, Oct. 1996

9 Winfried E. A Classification of interdomain actions. Operating Systems Review, 1998, 32(4): 47～61

10 Defense Information Systems Agency, Center for Standards. Department of Defense Technical Architecture Framework for Information Management (TAFIM), Version 3.0, April 1996

11 Defense Information Systems Agency, Center for Standards Department of Defense Goal Security Architecture (DGSA), Version 3.0, April 1996

12 Stefanos G, George A. Security issues surrounding programming languages for mobile code. Operating System Review, 1998, 32(2): 16～32

13 Venner B. Java's security architecture. Available at: http://www.javaworld.com/javalworld/jw08-1997/jw08-hood.html

14 Dean D, et al. Java Security: web browsers and beyond. ACM press, New York, Oct. 1997. 241～169

15 Dahlia M, et al. Secure execution of Java applets using a remote playground In: Proc of the 1998 IEEE Symposium on Security and Privacy. Orkland, California, May 1998. 40～51

16 Wong D, et al. Java-based Mobile Agents. Communications of the ACM, 1999, 42(3). 92～102

17 Crystaliz Inc., General Magic Inc., IBM Corporation, et al. Joint Submission. Mobile Agent Facility Specification, June 1997

18 秦志光. 开放系统中全局安全性的研究: [电子科技大学博士论文]. 1996