# Java2的安全机制

The Security Mechanism in Java2

李 镭 陈宏中

(同济大学计算机科学与工程系计算机标准化实验室 上海200122)

**Abstract** The security mechanism in Java2 can successfully prevent malicious Applet from doing harm to local systems and disturbing user's privacy. This paper mainly presents the Java2 runtime resource accessing control mechanism. Finally, it introduces some further research fields in Java2 security mechanism.

**Keywords** Security policy, Runtime resource accessing control mechanism

软件业正以前所未有的热情关注于电子商务应用系统的开发,Java 语言由于其平台无关性而被认为是开发这类系统的有力工具,因此 Java 的安全性特征成为业界关注的热点。

Java 适合于多种环境,从嵌入式系统、智能卡、PDA 到 NC 和主机系统等等,安全性要求也因环境不同而变化。早期的研究工作表明,采用一个公共的安全模型,并针对不同的应用环境,加以扩充的作法,与直接采用完全不同的安全模型的作法相比,更有助于降低开发应用程序以及支持库的工作量和成本。

## 一、Java 安全技术发展的回顾

Java 应用常常是在用户端的 Web 浏览器中以 Applet 的形式运行,所以 Java 安全性技术的焦点在于研究一套可实施于 Java 编译器和 JVM 的安全机制,以保护用户免受恶意 Applet 的攻击。因此,Java 的设计者对来自网上的 Applet 的功能做出严格的限制:JDK1.0规定来自网上的 Applet 要在严格限制的安全框架下运行,这样的安全框架在 Java 中被称为"沙箱"。"沙箱"禁止了来自网上的 Applet 访问本地文件系统和任意打开网络连接等功能。

JDK1.0的沙箱技术的副作用是使得 Applet 在实际 B/S 结构应用的开发中缺少灵活性。例如当用 Java 编写一个基于 B/S 结构的工资管理系统时,常常需要从运行会计程序的浏览器端计算机上直接导入一个本地的数据文件,依据 JDK1.0沙箱的机制,在浏览器端计算机上,沙箱拒绝这个来自网上的 Applet 访问本地文件的要求。由于存在这样的不便,各个浏览器开发商都提出了使来自网上的 Applet 摆脱沙箱控制的解决方案,SUN 在 JDK1.1中引入数字签名和 DSA/SHA1 加密算法的技术。开发者可以对 Applet 的类文件作数字签名和打包加密,用户可以通过检验数字签名,判断此 Applet 是否来自于可信任的站点或可信任的程序发布者,若用户判断 Applet 不能被信任,则它将被限制在沙箱中运行。如果它可信任则可以获得本地系统全部的资源访问权限,就和从本地装载的代码一样。

但是这样的安全机制在实际应用中仍有较大缺陷。例如在前面的例子里,用户只要对这个工资管理的 Applet 开放本地的一个文件,在 JDK1.1中,就要赋给这个 Applet 全部资源的访问权限,这种不安全的做法用户是无法接受的。其次,由于对来自网上的 Applet 的信任与否的判断完全要由用户人工控制,可靠性没有严格的保证。而在组织中,根据形势变化修订的信息安全策略需要及时地反映到每个用户的计算机中。举例来说,公司 C 的用户对来自站点 P 发布的 Applet 建立了信任关系,这种信任关系随着情况的变化可能会撤销,如果存在 C 的用户没有及时注意这种信任关系的改变,就有可能成为整个 C 公司网络的潜在的安全漏洞。

JDK1.2的安全模型所作的突出改进是:要求所有的代码,不论是来自远端的还是本地,都必须根据安全策略的规定来获得本地系统资源访问的权限,用户或管理员可以配置安全策略,来设定来自不同代码来源或不同的签名者的 Java 代码在访问保护资源时拥有的权限。例如,指定来自 URL:account.enterprise.com 的 Java Applet 拥有读取指定文件 c:\data\wage.dat 的权限等。如果一组类的实例被分配了相同权限,JDK1.2运行期系统就把这些类组织到同一个域(Domain)中,域作为 Java2运行期访问控制中进行权限管理的基本单位,分为系统域(SystemDomain)——包含着从本地 CLASSPATH 路径下加载的系统类,和保护域(ProtectionDomain)——包含着从其它位置加载的

类。

利用 JDK1.2所支持的详细划分的安全策略，对特定代码来源的 Applet 可以开放具体的访问权限，而且这种安全策略可以用策略文件来描述。使用这种基于策略文件的安全机制，可以集中化地管理组织的安全策略，保证任何更新都能及时地落实到连接到网络上的每台用户计算机上。可见，JDK1.2的安全模型进一步解放了 Applet 的灵活性和加强了 Java 在实际应用环境中的安全性。

## 二、Java2的安全性框架

从层次的角度看，Java2安全性框架由七个层次组成：①Java 语言规范；②Java 编译器；③字节码检验器；④安全管理器；⑤运行期的动态装载与资源访问检查机制以及垃圾收集器；⑥安全性策略；⑦字节码解释器。这些层次提供的系统安全特性的有机结合建立了稳固的 Java2安全性框架，如图1所示。

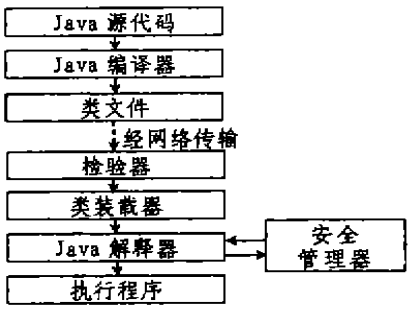


图1 Java 安全性框架

①Java 语言规范 Java 在语言中的安全性特征是整个 Java 安全性体系结构最基础的部分，例如取消指针避免直接对内存的特定单元进行访问；数组边界检查防止利用越界数组访问产生的安全问题，final 类和方法声明可以防止对已验证过的可信任代码的修改等。

②Java 编译器 Java 编译器在编译过程中，保证语言的所有安全特征得以实现，在编译阶段，隐含在 Java 语言句法和语义中的安全机制都要经过检查，包括私有和公共声明的一致性、类型安全和所有变量的初始化。与 C 和 C++不同，Java 中内存布局不是由编译器，而是由运行系统决定的，从而使得内存布局和引用模型对程序员透明。

③检验器 为了防止有人设法绕过编译器产生别有用心的字节码，Java 引入了字节码检验器，对加载的代码进行检验，以避免伪造的指针、越权访问、重载 final 类和方法以及非法的参数调用等，并保证类文件符合正常的格式。

④类装载器 经过检验器认证为合法的字节码将由类装载器负责加载。类装载器决定 Applet 装载的方式和时机，并管理类名空间，防止那些担负关键任务的系统类被替换。

⑤安全管理器和解释器 安全管理器负责在代码装载过程中初始化运行期 Applet 对文件和网络 I/O 的访问控制，建立新的类装载器，操纵线程或线程组，启动操作系统的进程，装载本地代码，并负责在运行过程中管理 Java 代码的资源访问请求。最后，解释器产生安全的可执行代码。

⑥安全性策略 Java2允许用户对特定的程序（既可以是 Applet，也可以是 Application）建立详细的安全策略。安全策略在系统中以 policy 对象来表示，该对象描述了允许程序执行的与安全相关的操作。执行诸如以下任务：检查是否可以在某端口上创建与另一个远端系统建立套接字连接；检查是否可以删除、读取或写入一个本地文件；检查是否可以执行本地系统的一个程序；检查是否可以侦听某个网络端口上的连接请求；标识可以使用 System.getProperty()方法访问的系统属性等的方法在 java.lang.SecurityManager 类实现。这就使得与安全有关的任务的执行被限制在系统根据安全策略可以控制的范围内。

## 三、Java2运行期环境的安全机制

Java2的运行期环境是 Java2安全性框架的重要组成部分，它负责检验将要加载的 Applet 类文件的合法性和对加载后 Applet 实现运行期资源访问控制。访问控制包括 Java 安全性框架的三部分：①字节码检验器（ByteCode Verifier），②安全管理器（Security Manager），③类装载器（Class Loader）。下面从 JVM 对 Java 运行期资源访问控制的角度分析 Java2运行期环境的安全机制。

当 Java 程序在执行过程中提出保护资源的访问要求时，为了验证该执行线程是否有权执行这样的操作，JVM 将调用 SecurityManager 的 CheckPermission（permissionToCheck，简称 PTC）方法，CheckPermission（PTC）随之调用 Access Controller.CheckPermission（PTC）方法。该方法检查当前的线程是否拥有访问保护资源所需的权限。它自顶向下扫描当前线程的堆栈帧，检查当前线程的堆栈中每个类的 ProtectionDomain，并将 PTC 和 ProtectionDomain 中所含的一组 permission 对象作比较，如果 ProtectionDomain 包含有和 PTC 相一致的 permission，则 CheckPermission 继续向下检查线程帧堆栈，如果检查成功地到达栈底，或是到达了 beginPrivilege()方法修改过的标志，则访问控制的检查成功，此次对保护资源访问就可以继续

执行；而如果没有找到和 PTC 一致的 permission，则产生一个 SecurityException，对资源的访问要求就被拒绝。

例 Java2资源访问安全检查的线程堆栈检查过程

假设用户在本地计算机上设定了这样的安全策略：

```
grant signedBy "secadmin", codeBase http://www.example.com/admin/*" { permission java.util.PropertyPermission "java.home", "read";
};
```

若来自 http://www.example.com/admin 下的 ex－1调用 System.getProperty("java.home")试图访问系统属性"java.home"，于是 getProperty 方法调用 SecurityManager checkproperty()以检查当前线程是否有权读取这个系统属性。接着，安全管理器调用 AccesssController.CheckPermission(PropertyPermission("java.home","read"))以检查线程堆栈中的类是否有这样的权限。

进入 AccessController.checkPermission 方法后，线程帧堆栈的模型如图2：

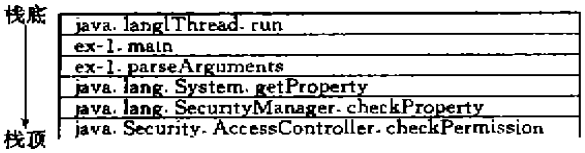


图2 检查开始时的线程堆栈帧模型

访问控制模型工作步骤如下：

1. 检查 java.Security.AccessController 类：由于 java.security.AccessController 是从本地系统的 CLASSPATH 路径下加载的系统类，因此它被划分到 SystemDomain，而 SystemDomain 含有权限 PropertyPermission("java.home","read")，当前帧通过访问权限检查；

2. 检查 java.lang.SecurityManager：java.langSecurityManager 同样是从本地系统的 CLASSPATH 路径下加载的系统类，因此它也被划分到 SystemDomain，当前帧通过访问权限检查；

3. 检查 java.lang.System：java.lang.Sysem 也是来自于本地系统的 CLASSPATH 路径下，和步骤1、2种情况一样，当前帧通过访问权限检查；

4. 检查 Ex-1类，Ex-1来自于 http://www.example.com/admin，因此安全策略声明 http://www.example.com/admin 拥有读取 java.home 的权限，所以，Ex-1所在的保护域中含有 PropertyPermission("java.home","read")权限；当前帧通过访问权限检查；

5. 检查 Ex-1类：如前所述，Ex-1所在的保护域中含有 PropertyPemission("java.home","read")权限；当前帧通过访问权限检查；

6. 检查 java.lang.thread 类：java.lang.thread 类也是来自于本地系统的 CLASSPATH 路径下的系统类，和步骤1、2、3中的情况一样，当前帧通过访问权限检查；

至此，该线程堆栈帧的权限检查完成，该线程有权访问本地系统属性"java.home"。

就 AccessController CheckPermission 判断权限的算法而言，它分成两阶段：

```
Step1.
for each class on the stack{
    Get The Class's ProtectionDomain;
    if the stack has been marked with beginPrivilege
      annotation break;/**Exit the loop*/
       If the stack frame checked was not marked
         with beginPrivileged annotation
    add the ProtectionDomain inherited by the current Thread when the current Thread was created}
Step2:
if no ProtectionDomain from step1 return;
  /**only fully trusted code is running)
for each unique ProtectionDomain P obtained in step1
{
call P's implies()method;
    search the appropriate PermissionCollection associated with the permission being checked;
    search the appropriate PermissionCollection found in this step;
    Does the permission in the previous step approve of the mission being checked?
    If no throw an exception else continue}
```

**结论** 综上所述，Java2的安全机制不但可以有效地防止程序开发中的错误造成对系统稳定的潜在破坏，而且也可防范别有用心的人编写"Java 病毒"在网络上进行攻击破坏，如：破坏程序和数据；利用特洛伊木马程序窃取敏感数据；借助运行 Applet 的计算机作为跳板，攻击网络上其他系统，同时隐藏攻击者本身，给网络安全管理员造成错觉等等。

1999年1月，Visa International 的成员银行访问 Visa 网络的信息系统——GEM（Global Endpoint Management）一期项目投入使用，基于 Java2 Standard Edition 开发的 GEM 项目成功说明 Java2完全可以满足大型跨平台网络信息系统的安全性要求。

但是，Java2的安全机制仍然有一些问题没有很好地解决。首先，对于不久前（2000年2月）在 Internet 上曾大规模爆发，并还有可能在今后经常出现的攻击形式：拒绝服务（Denny of Service），JDK1.2的安全模型暴露出以下不足之处：1）运行 Applet 客户机上的 CPU 资源可能被窃取；2）Applet 可分配的内存空间没有明确的限制；3）Applet 可以启动的线程数量没有限制。这三条缺陷将成为拒绝服务攻击的漏洞，SUN 公司正在研究更细致地控制 Java Applet 可使用的系统资源的可行性。

# Internet的移动访问技术研究

On Mobile Internet Access Techniques

戴方虎 周 炜 段 鲲 吴时霖

(复旦大学计算机系C&C实验室 上海200433)

**Abstract** This paper introduces the ways of Mobile Internet Access. It introduces the characters of wireless network, the model of Wireless Application Protocol. It also gives out one network solution based on WAP. The analysis and comparision are presented.

**Keywords** WAP, WAP Protocol Gateway, WML

移动通信和Internet是信息产业中发展迅速的两个领域,联接这两大领域,利用移动终端真正实现随时随地访问Internet有着诱人的前景。传统的Internet服务主要是基于有线网络的,HTML,HTTP,TCP等协议都基于高带宽、低错误率的有线网络,当移植到无线网中时这些协议就变得效率低下,性能远低于有线网络。移动终端受其移动性和便携性的制约,与桌面PC机相比,CPU功率及计算能力都较小,存储器容量、显示屏较小,输入方式各异(如键盘、语音输入等);同时,由于电源、可接收范围、移动性等因素的影响,无线网络与有线网相比,带宽窄,传输时延较大,连接的稳定性不够,可预测性低。为了克服无线网上述种种局限性,使能够通过移动终端(包括手机、BP机及个人数字助理PDA等),使用现有的Internet和Intranet所提供的服务,目前解决方案主要有两种方式:一是将移动终端当作功能简化了的PC机,这样现有的Internet协议不做大的修改就可以直接使用了。另一种方式则是重写现有的Internet协议,使其与现有协议兼容,更适合于无线应用这一特殊环境。

## 一、简化PC机方式

将移动设备终端增强功能成为简化的PC机——掌上电脑,利用PC机原有的软件和技术,来访问Internet服务。掌上电脑(包括各种PDA)不仅可以将大量重要资料装在掌上电脑中,由于它和Windows相关软件相容的特性,可以和个人电脑同步传输并加以编辑,而且可以随时传送电子邮件或传真和其他人联络。掌上电脑共有三种连接方式:

1)通过RS232电缆与台式机进行连接,实现同步化;

2)通过红外连接(两台设备的红外端口对准来实现),使掌上电脑与同类产品或具有红外端口的台式电脑及打印机连接;

3)是通过内置调制解调器(modem),利用电话线

**戴方虎** 硕士生,研究方向为计算机通信与网络技术。**周 炜** 硕士生,研究方向为计算机辅助设计。**段 鲲** 硕士生,研究方向为计算机通信与网络技术。**吴时霖** 博士生导师,研究方向为计算机通信、网络技术与Petri网理论。

其次,目前用户还无法禁止Applet将垃圾信息下载到本地,虽然用户可以拒绝这些信息的保存,但无法避免被其浪费时间和网络带宽,因此有必要考虑在标准的Java类库中集成一个可由用户定义的内容过滤器。

## 参 考 文 献

1 Li Gong. Inside Java 2 Platform Security. Addison-Wesley, Reading Mass., 1999, ISBN 0-201-31000-7

2 Ritchie S. Systems Programming in Java. IEEE Mi-cro, 1997, 17(3): 30～35

3 Dean D, Felten E W, Wallach D S. Java security: From HotJava to Netscape and beyond. In: Proc. of the 1996 IEEE Symposium on Security and Privacy. Oakland, CA, 1996. 190～200

4 Lindholm T, Yellin F. The Java Virtual Machine Specification. Addison-Wesley Publishing Co., Reading, MA 1997

5 Kemmerer R, Paoli F D, Dos Santos A L. Vulnerability of 'Secure' Web Browsers. In: 20th National Information Systems Security Conference, sponsored by NIST and the national Computer Security Center, Baltimore, MD, 1997. 488～497