

## Java 卡对象共享安全策略分析与实现

胥怡心,张其善

(北京航空航天大学 电子信息工程学院, 北京 100083)

(xyx.buaa@gmail.com; xyx1@vip.sina.com)

**摘 要:**使用形式化方法对 Java 智能卡的应用隔离与对象共享机制进行分析,用信任模型表述卡内多应用间对象共享关系,根据信任可传递的条件推断出仅由应用级安全策略控制共享对象访问的不足,提出以增加卡内全局访问控制机制来防止未经授权的信息流动这一安全策略设计思路,并给出一个简单实现。

**关键词:**Java 智能卡;对象共享;信任传递

**中图分类号:**TP309 **文献标志码:**A

## Analysis and implementation of security object sharing policy in Java card

XU Yi-xin, ZHANG Qi-shan

(School of Electronics and Information Engineering, Beihang University, Beijing 100083, China)

**Abstract:** A formal method was used to analyze the application isolation and objects sharing mechanism in Java smart card. A trust model was used to clarify the relationship of the inter-applet shareable objects. According to the constraint of trust transfer, it was inferred that, in addition to the applet level access controls, a global security policy of card was necessary to prevent the unauthorized information flow. Finally a simple implementation of the global security mechanism was designed.

**Key words:** Java smart card; object sharing; trust transfer

### 0 引言

Java 智能卡作为一种应用广泛的多应用智能卡平台,减少了用户随身携带的卡片数量,节省了发行方的市场部署时间,延长了卡片的使用周期。Java 智能卡的应用隔离与对象共享机制,一方面保证了卡内不同应用提供方开发的应用相互隔离互不干扰,另一方面也为多个合作方的应用在卡内进行数据交换提供了技术支持。

Java 智能卡技术规范<sup>[1-3]</sup>规定的应用防火墙机制本质上是将卡内对象划分为以应用上下文(context)为标签的多个集合。同一集合内的对象间访问是合法的,而不同集合间的对象访问是有条件的。这个跨集合的对象访问规则就是共享接口对象访问机制<sup>[2]</sup>。通过对对象共享机制中所涉及的 Java 智能卡 API<sup>[3]</sup>进行分析可知,获取共享接口对象引用的安全检查在服务器 Applet 中进行,但当客户 Applet 获得了共享对象的引用后服务器 Applet 无法控制该对象引用的使用,这就是 Java 卡规范中规定的共享接口对象访问机制的安全漏洞。

文献[4]提出了一种基于偏序网格的分级对象共享机制。为了控制信息流向,可以为卡内所有应用产生的对象指定一个等级,信息只可以从等级低的对象流向等级高的对象。这一方法有两个问题:一是如何为 Java 智能卡中的对象指定等级,尤其是如果新下载的应用要访问卡内原有应用的共享接口时,虚拟机如何为新创建的共享对象指定等级;二是如果每次对象访问都要比较访问者与被访问者的等级关系,那会大大降低智能卡的运行效率。

本文使用形式化分析方法对 Java 智能卡对象共享机制的安全性进行分析,得出仅依靠应用提供方的应用级安全策略不能保证对象安全共享的结论,提出以增加卡内全局访问

控制机制来防止未经授权的信息流动这一安全策略设计思路,并给出一个由 Java 卡运行环境和卡内应用共同约束对象共享的示例。

### 1 Java 智能卡对象共享机制安全策略设计

#### 1.1 Java 智能卡的对象共享流程

Applet 应用程序通过创建实现抽象接口 Shareable 的类的实例对象,可向其他 Applet 提供在该类中设计的服务,流程图如图 1 所示。

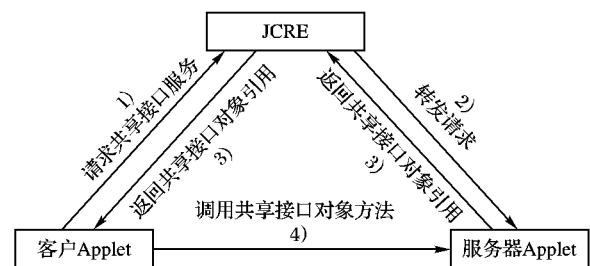


图 1 Java 智能卡内对象共享机制流程

1) 客户 Applet 调用 JCSysm.getshareableAppletInterface 方法,向 JCRC 提出共享接口服务请求,并指明服务器 Applet 的 AID。

2) JCRC 收到客户 Applet 的申请后,查找指定的服务器 Applet,然后调用服务器 Applet 的 getshareableAppletInterface 方法将客户 Applet 的请求转发给服务器 Applet,并指明客户 Applet 的 AID。

3) 服务器 Applet 判断是否向客户 Applet 提供共享接口中包含的服务,如果可以则将共享接口对象的引用通过 JCRC 返回给客户 Applet。

收稿日期:2008-12-12;修回日期:2009-02-24。

**作者简介:**胥怡心(1978-),男,四川剑阁人,博士研究生,主要研究方向:智能卡技术、信息安全;张其善(1936-),男,浙江浦江人,教授,博士生导师,主要研究方向:通信与信息系统、信息安全。

4) 客户 Applet 使用共享接口对象就可以调用该共享接口提供的服务。

在第 3) 步中服务器 Applet 是通过检查客户 Applet 的 AID 是否符合自身的安全策略来决定是否提供共享接口服务的。也就是说,对象共享机制是受应用级的安全策略保护,这对两个 Applet 之间的对象共享是有效的(前提是传递的客户 Applet 的 AID 是真实的),但对两个以上 Applet 之间的安全对象共享就存在问题<sup>[5]</sup>。

## 1.2 多应用间对象共享机制形式化分析

Java 智能卡内的对象共享机制可以使用信任关系模型表示,在这个模型中主要考虑信任关系的建立以及信任关系的传递。类似于文献[6]的模型,本文对 Java 智能卡内的信任模型进行如下形式化分析。

定义三个基本集合:

1) APPLT 表示 Java 智能卡内已安装的所有 Applet 的集合。

2) SIO 表示所有共享接口对象的集合。

3) POLICY 表示 Applet 的对象共享策略。共享策略定义了信任方评价受信方是否值得信任的评价标准、受信方必须具备的相关属性,以及对信任事件所处环境的要求等。如第一部分所述,对象共享策略属于应用级安全策略。

定义逻辑谓词:

$\text{Trust}(\text{server}, \text{client}, \text{sio}_s) \text{ server}$

$\text{client} \in \text{APPLET}$

$\text{sio}_s \in \text{SIO}$

表示信任关系,当且仅当 server 信任 client,并向 client 提供 sio 时该谓词为真。其中 server 是服务器 Applet,client 是客户 Applet,sio<sub>s</sub> 是服务器 Applet 创建的共享接口对象。

定义逻辑条件:

$\text{Satisfy}(\text{policy}_s, \text{client}) \text{ policy}_s \in \text{POLICY}$

$\text{client} \in \text{APPLETS}$

表示共享策略检查,当且仅当客户 Applet 满足服务器 Applet 的共享策略时该条件为真。

信任策略可以表示为:

$\text{Trust}(\text{server}, \text{client}, \text{sio}_s) \leftrightarrow \text{Satisfy}(\text{policy}_s, \text{client})$  (1)

即当且仅当客户 Applet 满足服务器 Applet 的共享策略时,服务器 Applet 对客户 Applet 的信任关系成立。

假设 AppletA 可向 AppletB 提供共享接口服务,而 AppletB 可向 AppletC 提供共享接口服务,即存在如下关系:

$\text{Trust}(\text{AppletA}, \text{AppletB}, \text{sio}_A)$

$\text{Trust}(\text{AppletB}, \text{AppletC}, \text{sio}_B)$

如果 AppletC 想通过 AppletB 获得 AppletA 的共享接口服务,则可以认为:

$\text{sio}_A \equiv \text{sio}_B \equiv \text{sio}$

那么,由式(1)可知,

$\text{Trust}(\text{AppletA}, \text{AppletB}, \text{sio}) \wedge \text{Trust}(\text{AppletB}, \text{AppletC}, \text{sio}) \rightarrow \text{Trust}(\text{AppletA}, \text{AppletC}, \text{sio})$  (2)

等价于

$\text{Satisfy}(\text{policy}_A, \text{AppletB}) \wedge \text{Satisfy}(\text{policy}_B, \text{AppletC}) \rightarrow \text{Satisfy}(\text{policy}_A, \text{AppletC})$  (3)

根据连接符“ $\rightarrow$ ”的真值表,当

$\text{Satisfy}(\text{policy}_A, \text{AppletB}) \wedge \text{Satisfy}(\text{policy}_B, \text{AppletC})$  与  $\text{Satisfy}(\text{policy}_A, \text{AppletC})$  均为真时,式(3)为真,从而可知式

(2)为真,即若 AppletC 想通过 AppletB 间接获得 AppletA 的共享接口服务,必须要进行三次应用级安全策略的验证:

1) 验证 AppletC 是否满足 AppletB 的共享策略;

2) 验证 AppletB 是否满足 AppletA 的共享策略;

3) 验证 AppletC 是否满足 AppletA 的共享策略。

这就是多应用间对象共享的安全策略,显然这个安全策略属于 Java 智能卡的平台级安全策略,必须由 Java 卡运行环境执行。

## 2 多应用间安全对象共享机制的实现

对第 1 章描述的对象共享机制进行调整,以三个 Applet (客户 Applet、中介 Applet 和服务器 Applet)间对象共享为例,在 JCSys. getsherableAppletInterface 方法中增加对客户 Applet 是否满足服务器 Applet 共享策略的验证,流程如图 2 所示。

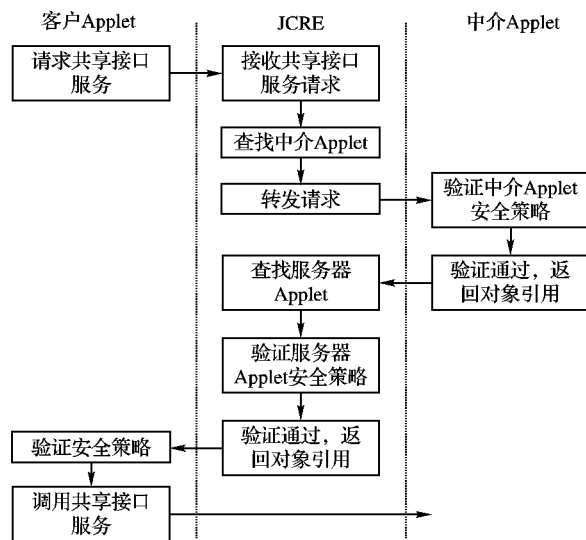


图 2 改进的 Java 卡多应用对象共享机制流程

图 2 中的流程没有包括中介 Applet 与服务器 Applet 间的交互,这是假设中介 Applet 已通过了服务器 Applet 的安全验证并获得了共享接口对象的引用,该流程与图 1 一致。JCRE 需要在收到中介 Applet 的正确返回后,根据中介 Applet 的 AID 查找向其提供共享接口服务的服务器 Applet,对于简单的情况可以从中介 Applet 的导入包中找到服务器 Applet。然后验证客户 Applet 是否满足服务器 Applet 的安全策略,通过后才返回共享对象引用给客户 Applet。

## 3 结语

本文使用形式化方法对 Java 智能卡的对象共享机制进行分析,通过建立信任模型来表示多应用间的对象共享,以信任关系的建立条件分析对象共享安全策略的实质,以信任关系可传递性的限制分析多应用间对象共享机制的信息流向,指出如果要阻止应用间非授权信息传递,仅依靠应用级的安全策略是不够的,必须在卡片范围内制定保护策略,并在最后给出了一个简单环境下该策略的实现流程。这种安全策略实现相对文献[1]中的方式,只需为共享对象设定相关安全属性,并由 JCRE 检查其属性是否满足所属应用的安全策略即可,无需为所有对象指定等级并一一比对,提高了执行效率。

我们将在未来的工作中继续深入研究形式化方法在制定

(下转第 1621 页)

图4(e)为交叉定位的结果。仿真结果表明,在多区域篡改的情况下,本文算法相对于文献[7]的交叉定位方法具有很高的定位精度和很低的检测定位虚警。

本文算法的定位精度由图像特征量的提取方法及定位方法决定,算法以 $8 \times 8$ 的图像块为单位提取特征量,采用行列交叉和区域复合的定位方法,由于取的是三者的交集,所以其定位精度和文献[2,6-7]相同,均为 $8 \times 8$ 像素。

#### 4.3 算法的安全性分析

由于该算法采用了 $K_1$ 、 $K_2$ 方式的密钥, $K_1$ 共有 $127! \times 128$ 种选择, $K_2$ 共有 $255! \times 256$ 种选择,所以,攻击者很难采用穷举法获得密钥,从而无法得到正确的摘要。同时由于本算法属于签名认证,可有效抵抗块置换、拼贴及伪造攻击。

#### 4.4 仿真实验结果

本文采用大量图像进行了多区域篡改认证实验,实验结果和性能分析所得的结论一致。由第2章的理论分析可知本文提取的特征量在 JPEG 压缩的情况下保持不变,所以对非压缩的篡改图像的认证结果和压缩后的篡改图像的认证结果是相同的,所以下面只给出压缩后的篡改图像的部分认证实验结果。

图3中篡改是在图像中随机插入大小为 $16 \times 16$ 像素的两个CS图片,一个大小为 $16 \times 16$ 像素的随机剪切;图4中篡改是改变图中一小汽车的位置;压缩图像指经质量因子 $QF = 25$ 的 JPEG 压缩后的图像。依据图像传输的一般原理,这里的压缩与篡改图像是指先压缩后篡改图像。图3(b)、图4(b)为预处理后图像,图像没有明显失真,且峰值信噪比较高(注意:这里的PSNR是指仿射变换和预处理后图像的峰值信噪比);图3(d)、图4(d)为对图3(c)、图4(c)单纯的区域检测定位结果,结果表明,其定位精度较低,但检测概率高;图3(e)、图4(e)为对图3(c)、图4(c)单纯的交叉检测定位结果,结果表明,其定位精度较高,但虚警严重,会判定图像大面积被篡改;图3(f)、图4(f)为本文提出的行、列及区域复合交叉检测定位方法对图3(c)、图4(c)的检测定位结果,结果表明,检测定位准确,精度高,虚警很小;同时也表明,质量因子 $QF \geq 25$ 的 JPEG 压缩对篡改检测和定位未造成任何影响。

## 5 结语

本文在研究摘要提取和图像预处理新方法的基础上,提出了一种基于数字签名的近似图像认证算法。该算法在抗 JPEG 压缩、篡改检测概率、虚警概率及篡改定位等方面都取得了良好的结果。在多区域篡改情况下,该算法具有一定的优势。该算法仅从图像块均值中提取摘要,对剪切、替换及位置改变等篡改认证效果良好,但是对细节性篡改认证有一定

的局限性。在今后的工作中,可以将图像 DCT 变换的交流系数纳入认证摘要的提取范围,增强算法的认证能力。

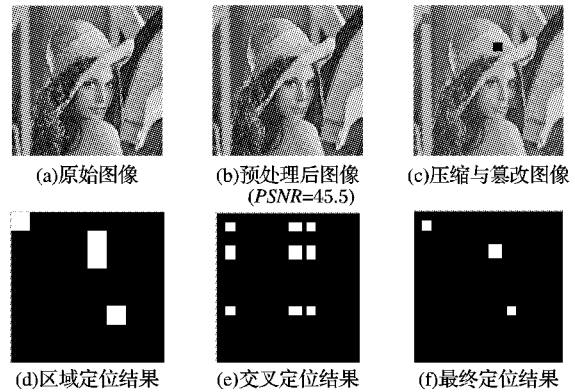


图3 Lena 图像仿真实验结果

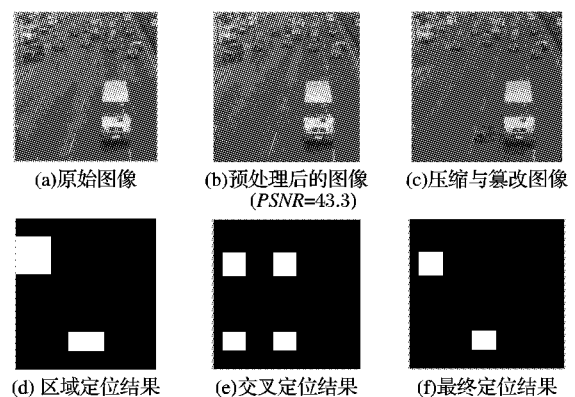


图4 交通汽车图像仿真实验结果

#### 参考文献:

- [1] 王育民, 刘建伟. 通信网的安全—理论与技术[M]. 西安: 西安电子科技大学出版社, 2002: 233-263.
- [2] XIE LIE-HUA, ARCE G R, GRAVEMAN R F. Approximate image message authentication codes[J]. IEEE Transactions on Multimedia, 2001, 3(2): 242-252.
- [3] 沃焱, 韩国强, 张波. 基于特征的静态图像内容认证方法[J]. 中国图象图形学报, 2006, 11(7): 1036-1042.
- [4] 董刚, 张良, 张春田. 一种半脆弱性数字图像水印算法[J]. 通信学报, 2003, 24(1): 33-38.
- [5] 钟桦, 焦李成. DCT 域半易损水印技术[J]. 计算机学报, 2005, 28(9): 1549-1557.
- [6] 余淼, 和红杰, 张家树. 一种高定位精度的安全 JPEG 图像认证水印算法[J]. 中国科学: E 辑, 2007, 37(2): 315-328.
- [7] 钟桦, 焦李成. 一种用于图像内容鉴别的数字签名方案[J]. 计算机学报, 2003, 26(6): 708-715.
- [8] 王新梅, 肖国镇. 纠错码——原理与方法[M]. 西安: 西安电子科技大学出版社, 2003: 242-316.

(上接第1616页)

Java 智能卡安全策略中的应用,解决复杂情况下,如多个应用间非线性共享接口调用关系,如何验证安全策略是否被满足,安全策略如何更新,以及处理新安装 Applet 对 Java 智能卡内部环境造成的改变等问题。

#### 参考文献:

- [1] Sun Microsystems. Virtual machine specification Java card platform, Version 2.2.2 [EB/OL]. [2008-10-21]. <http://java.sun.com/products/javacard/specs.html>.
- [2] Sun Microsystems. Runtime environment specification Java card platform, Version 2.2.2 [EB/OL]. [2008-10-21]. <http://java.sun.com/products/javacard/specs.html>.

- [3] Sun Microsystems. Application programming interface Java card platform, Version 2.2.2 [EB/OL]. [2008-10-21]. <http://java.sun.com/products/javacard/specs.html>.
- [4] GIRAND P. Which security policy for multiapplication smart card? [EB/OL]. [2008-10-10]. <http://www.gemplus.com/smart/rd/publications/ps/Gir99sec.ps>.
- [5] BIEBER P, CAZIN J, GIRAND P, et al. Checking secure interactions of smart card applets: extended version [J]. Journal of Computer Security, 2002, (10)4: 369-398.
- [6] DONG C Y, RUSSELLO G, DULAY N. Trust transfer in distributed systems [EB/OL]. [2008-10-10]. <http://www.doc.ic.ac.uk/~cd04/papers/itrust07.pdf>.