·安全技术· 文章编号：1000-3428(2017)10-0134-07 文献标志码：A 中图分类号：TP309

面向 Android 隐私保护机制的多域隔离模型设计

王帅丽[1,2]，孙 磊[1,2]，韩静丹[1,2]，徐 宁[3]，王泽武[1,2]

（1. 解放军信息工程大学 密码工程学院，郑州 450001；2. 河南省信息安全重点实验室，郑州 450001；
3. 中标软件有限公司，上海 200030）

摘 要： 针对 Android 系统粗粒度的权限机制及隐私保护机制安全性较低的问题，提出粒度可控的多域隔离隐私保护模型 MDSDroid，并在 Android 系统上设计模型实现框架。通过定义模型变量以及访问控制策略，实现应用程序及其数据间的隔离和强安全访问控制机制。采用 Z 语言对模型进行形式定义，并运用形式验证工具 Z/EVES 进行形式分析，保证模型策略的正确执行，在增强系统安全的同时保护隐私数据的安全性。实验结果表明，该模型系统与 Android 原生系统相比具有较低的性能损耗。

关键词： 安卓系统；隐私数据；域隔离；安全策略；形式化

中文引用格式： 王帅丽，孙 磊，韩静丹，等. 面向 Android 隐私保护机制的多域隔离模型设计[J]. 计算机工程，2017，43(10)：134-140.

英文引用格式： WANG Shuaili，SUN Lei，HAN Jingdan，et al. Design of Multi-domain Isolation Model for Android Privacy Protection Mechanism[J]. Computer Engineering，2017，43(10)：134-140.

Design of Multi-domain Isolation Model for Android Privacy Protection Mechanism

WANG Shuaili[1,2]，SUN Lei[1,2]，HAN Jingdan[1,2]，XU Ning[3]，WANG Zewu[1,2]

（1. Institute of Cryptography Engineering，PLA Information Engineering University，Zhengzhou 450001，China；
2. Henan Province Key Laboratory of Information Security，Zhengzhou 450001，China；
3. China Standard Software Co.，Ltd.，Shanghai 200030，China）

【Abstract】 Aiming at the problem of coarse granularity access mechanism and weak privacy protection mechanism in Android system. Multi-domain isolation privacy protection model（MDSDroid）with adjustable granularity is proposed and the model implementation framework is designed in the Android system. Through defining the model variables and access control policy，the framework can effectively isolate the applications and application data，and realize strong security access control mechanism. Z language is used to define the model which is then verified with the help of Z/EVES tool to ensure the correct performance of the model policy. The system security is enhanced and the security of privacy data is protected. Experimental results show that the model system is less than Android primary system performance.

【Key words】 Android system；privacy data；domain isolation；security policy；formalization

DOI： 10.3969/j.issn.1000-3428.2017.10.023

0 概述

近年来移动互联网的快速发展促使智能手机的使用越来越普及，在众多移动平台中，由 Google 基于 Linux 内核设计的 Android 操作系统以其出众的用户体验和其开放性受到广大用户和手机制造商的青睐。根据 Kantar2016 年 2 月统计数据分析[1]，Android 手机操作系统市场占有率高达 80.7%，在智能手机市场占据第 1 位。随着 Android 智能手机的功能逐渐完善以及计算能力和存储能力的逐渐提升，越来越多的用户习惯用手机进行异地通信、休闲娱乐、网络社交、移动支付和远程办公等。

Android 手机中存储着大量与用户相关的隐私数据信息，然而，由于手机用户可以从管理混乱的第三方平台安装应用程序，Android 系统中的隐私数据信息成为越来越多恶意软件和黑客攻击的目标[2-4]。这使得手机上的用户隐私信息、网银业务等面临很大的安全风险。腾讯安全实验室发布的报告称[5]，2016 年上半年新增 Android 病毒包 918.25 万个，同比增长 53.90%，是 2014 年全年新增病毒包（100.33 万个）的

基金项目： 国家重点研发计划项目“协同精密定位技术”（2016YFB0501900）；国家部委基金。

作者简介： 王帅丽（1992—），女，硕士研究生，主研方向为信息安全；孙 磊，研究员、博士；韩静丹，硕士研究生；徐 宁，博士；王泽武，硕士研究生。

收稿日期： 2016-07-25 **修回日期：** 2016-11-11 **E-mail：** 972515859@qq.com

9.15 倍,其中,新增支付病毒包 32.33 万个,同比增长 986.14%,支付病毒形式严峻。面对恶意软件攻击[6-8]时,用户最为担心的是信用卡和银行账户信息、个人身份资料以及手机联系人信息等与用户隐私相关的数据信息被窃取。Android 系统的安全性以及如何保护用户隐私数据信息的安全成为人们关注的焦点。本文从保护用户隐私数据的角度出发,分析近年来 Android 系统在保护隐私数据方面的研究,设计一个多域安全隔离模型 MDSDroid。

1　相关研究

近年来,在 Android 安全方面的研究大幅增加,相关学术论文也覆盖很广,如 Android 安全综述[9-12]、Android 权限机制方面[13-15]等。其中,针对 Android 系统中隐私数据保护方面,相关研究人员在静态分析、动态分析方面做了一些研究工作。

在静态分析方面,文献[16]静态分析反编译的应用程序源码,将提取出的 API 集合与自己建立的映射库进行对比,判断应用程序是否有窃取隐私数据的行为。文献[17]将隐私数据源进行标记,当信息流达到发送点时,利用静态信息流分析工具检查隐私是否被发送出去。这些静态检测方法虽然能够查出一些恶意行为,但存在较高的错误率,且这些方式只起到预防作用,不能阻止隐私数据泄露。

在动态分析方面,文献[18]使用污点跟踪技术对敏感的隐私数据进行标记,将应用数据与企业数据进行隔离,但这种机制只是单纯地保护企业隐私数据被私人应用访问的问题,并没有提供细粒度的权限策略来控制隐私数据的流通。文献[19]利用动态污点技术对隐私数据进行标签标记,当隐私数据被发送出去时,系统内部会打出日志,日志经过解析后给用户发出警告。但这种方法不能阻止隐私数据的泄露,仅仅实现了对隐私数据流通的监控。

在 Android 系统中,应用程序产生的非隐私数据和隐私数据存储在同一个数据库中,任何应用程序都可以调用数据库中的数据。为防止隐私数据被非法调用造成用户隐私数据泄露,最近几年也有相关研究者对 Android 系统的隔离方法进行研究,从系统安全方面保护隐私数据信息的安全。

文献[20]将 Android 系统封装在虚拟机中,将应用安装在不同的虚拟机划分的域中并实时监控 Android 系统各个应用的行为,从而实现对 Android 系统的安全防护。但是将虚拟机应用于移动设备上,运行效率并不乐观。文献[21]采用轻量级虚拟化技术在系统中创建不同的安全池进行划域隔离,每个应用程序安装在不同的安全池中,并利用 TaintDroid[19]的标签标记功能实现对隐私数据的跟踪和保护。但是使用虚拟化技术设计的安全池在切换时有一定的负载,影响系统的运行性能。

手机中存储着与用户相关的大量数据信息,将不同种类的数据信息进行分类管理是保护用户隐私数据信息的可行方法。针对上述隐私数据保护研究方法中存在的不足,本文提出一个粒度可调的多域隔离隐私保护模型 MDSDroid,并在 Android 系统上设计了实现框架,该模型能有效地防止串谋攻击,实现强访问控制机制,在增强系统安全性的同时能有效保护用户的隐私数据的安全性。另外,文中采用 Z 语言对模型及相关策略进行形式化描述,并借助 Z/EVES 工具进行形式化分析以保证模型策略的安全执行。

2　MDSDroid 模型设计

MDSDroid 是将 DTE 的域隔离技术、BLP 的机密性策略以及根据 Android 实际使用时的特点而添加的授权访问策略结合起来而设计的域隔离模型。

2.1　MDSDroid 模型

MDSDroid 模型如图 1 所示。模型以设置的安全等级进行划域,其中,安全等级为整数且每个域的安全等级互不相同,设 0 级别安全域为通用域且不可删除。用户可根据自身安全需求对模型的安全域进行增加或删除。用户添加新的域时需要设置域的安全等级,系统根据用户设置的域安全等级生成新的安全域。

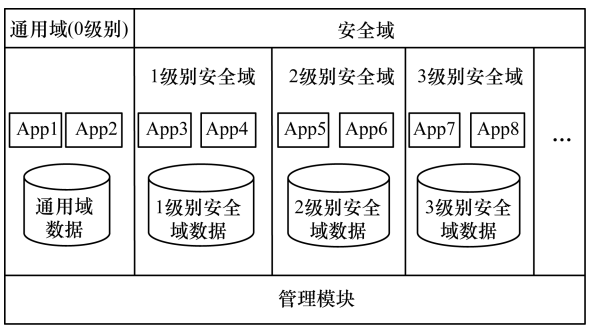

图 1　MDSDroid 模型

每个域代表一个逻辑隔离单元,域间通过域标签进行隔离。域中包括应用程序及应用程序产生的相关数据。另外,用户可根据自身情况将应用程序产生的相关数据重新定义安全级别,将其移至其他安全级别的域中。MDSDroid 模型中设计了管理模块。管理模块负责应用程序安全等级的配置、标签管理、安全域的扩充与管理等。

用户安装应用程序时可根据该应用程序与隐私数据的关联程度设置相应的安全级别。系统根据设置应用程序的安全级别将应用程序安装至相应安全级别的域中。若用户设置的安全级别不在已创建的安全级别域内,系统则会在安全域中增加相应级别的安全域。应用程序设置的安全级别越高,产生的数据相对越安全,同时,应用程序的访问约束越多。

以图 1 为例,在实际应用中,用户可以将完全不涉及隐私数据的常用应用程序安装至通用域中。如有道词典、视频软件等;将使用过程中可能使用或产生涉及隐私数据的应用程序安装至 1 级别安全域中,如娱乐游戏(有时需使用 qq 账号、微信账号、手机号等进行用户绑定)、导航地图等;将涉及用户隐私数据的应用程序安装至 2 级别安全域中,如 qq、微信等;将与用户隐私关联非常密切,涉及财产安全的应用程序安装至 3 级别安全域中,如支付宝、网上银行手机客户端等。

通过将应用程序及相关数据划分至合适的域中实施域隔离保护,一方面可以将数据进行隔离,制约应用程序可作用的范围,即使遭受恶意应用程序的攻击,也能限定其对系统的破坏范围;另一方面通过限制域的权限可以防止权限过大造成的串权攻击等。

2.2 MDSDroid 模型变量定义

模型变量定义如下:

1)主体 *Subjects*。

2)客体 *Objects*。

3)域(*Domains*)。*Domains* = {*Common_d_0*, *Security_d*},*Security_d* = {*Security_d_1*, *Security_d_2*, *Security_d_3*, …}。

4)类型(*Types*)。*Types* = {*Comman_t_0*, *Security_t*},其中 *Security_t* = {*Security_t_1*, *Security_2_t*, *Security_t_3*, …}。

5)*object_type*:客体与型间的分配关系。

6)*subject_domain*:主体与域间的分配关系。*subject_auth_domain* 为授权主体。只有授权的主体才能进入到域中。

7)域间转换模式(*DCM*)。*DCM* = {*auto*, *exec*, *signal*, *null*}。

8)域对型的访问模式集 *DTAM* = {*read*(*r*), *write*(*w*), *execute*(*x*), *delete*(*d*), *add*(*a*)}。

9)域之间关系表 $DDT \subseteq Domains \times Domains \times DCM$。

10)类型关系表 $DTT \subseteq Types \times Domains \times DTAM$。

MDSDroid 模型的状态变量包括主体 *Subjects*、客体 *Objects*、域 *Domain* 和型 *Type* 的集合。主体是指激活的进程,进程的当前执行域是唯一的,即每个活动的进程有且仅有一个域标签[22]。MDSDroid 模型根据设置的安全等级进行划域,0 级别域为通用域 *Common_d_0*,用户可根据自身需要在安全域中添加不小于 1 的正整数安全等级的域。安全域新增域的域标签命名规则是 *Security_d_x*,其中 *x* 代表新增安全域时设置的安全等级,如 2 级别安全域 *Security_d_2*、3 级别安全域 *Security_d_2* 等。通用域的型标签为 *Common_t_0*,安全域新增域中的型标签命名规则为 *Security_t_y*,其中 *y* 代表文件所在域的安全等级,如 *Security_t_2*, *Security_t_3* 等。当用户不再使用某安全域时,可对其进行删除,但通用域不能进行删除操作。*object_type* 为客体与型间的分配关系,每个客体有且仅有一个型。*subject_domain* 为主体与域间的分配关系,只有授权的主体才能进入到域中。域间转换模式 DCM 即域的转换关系,包括自动转换(auto)、强制转换(exec)、单一转换(signal)和空转换(null)。域对型的访问模式 DTAM 包括文件读(r)、文件写(w)、文件执行(x)、文件删除(d)、追加(a),访问模式可以根据实际客体类型进一步扩充。域之间的关系表 DDT 包括不同的域以及域之间的交互模式。类型关系表 DTT 包括不同型/域以及它们之间的操作。

2.3 MDSDroid 模型策略

域之间以及域和型之间通过域间访问控制规则策略进行约束。其中,安全域中的新增加域均继承以下访问控制规则策略。

主要策略如下:

1)域间访问控制策略。

域之间的交互模式要根据 DDT 关系表进行换,(*Domain*, *DCM*, *Domain*) ∈ *DCM*。低安全级别域中主体可通知高安全级别域中主体对信息进行读取;高安全级别域中的主体可自动切换至低安全级别域中;低安全级别域中主体非授权不能进入高安全级别域中。以模型中存在通用域和 1 级别安全域为例,描述如下:

$\exists \alpha_1, \alpha_2, \alpha_1, \alpha_2 \in Subject$, $Common\_d\_0(\alpha_1) \times signal \times Security\_d\_1(\alpha_2)$; $\exists \alpha_2, \alpha_2 \in Subject$, $Security\_d\_1(\alpha_2) \times auto \times Common\_d\_0(\alpha_2)$; $\exists \alpha_1, \alpha_1 \in Subject$, $Common\_d\_0(\alpha_1) \times null \times Security\_d\_1(\alpha_1)$。

2)域中的主体对型文件的访问控制策略。

每个域中的主体均能操作本域中的型文件;高安全级别域中主体对低安全级别域中型数据文件只能进行读取,不能写入该型数据文件;低安全级别域中主体能向高安全级别域中型数据文件写入,但未经授权不能读取高安全级别域中的型数据文件。以模型中存在通用域和 1 级别安全域为例,描述如下:

$(Common\_d\_0, Common\_t\_0, DTAM) \in DTT$, $(Security\_d\_1, Security\_t\_1, DTAM) \in DTT$, $\exists \alpha_1, \alpha_1 \in Subject$, $Common\_d\_0(\alpha_1) \times DTAM \times Common\_t\_0$, $\exists \alpha_2, \alpha_2 \in Subject$, $Security\_d\_1(\alpha_2) \times DTAM \times Security\_t\_1$; $\exists \alpha_2, \alpha_2 \in Subject$, $Security\_d\_1(\alpha_2) \times read \times Common\_t\_0$; $\exists \alpha_1, \alpha_1 \in Subject$, $Common\_d\_0(\alpha_1) \times write \times Security\_t\_1$。

3)授权主体的访问控制策略。

在 Android 系统使用过程中,存在低安全级别域中的主体需临时访问高安全级别域中型数据文件或

进入高安全级别域中进行一些操作的情况。如支付宝、手机银行客户端和娱乐游戏安装在不同的域中,且前者域安全级别高于后者。游戏在使用过程中存在购买装备等需求,需要进行网上支付,此时,游戏应用程序主体需进入较高安全等级域中进行支付等操作。对于系统中存在的此类情况,本文通过创建临时可信管道进行实现,利用可信管道机制来保障域之间信息传输的安全性。

低安全级别域中主体经授权后,经系统创建的临时可信通信管道进入较高安全级别域中进行操作或进行信息传递。操作结束后,对临时创建的通信管道进行撤销,不仅可以防止恶意应用程序进程利用这些管道进行恶意操作,而且可以释放创建管道所占用的系统资源。

下面以授权通用域主体进入 1 级别安全域进行操作为例进行介绍。

管道 tp 相关变量定义如下:创建管道 tp_create,管道状态 $tp\_state = \{Ready, Using, Delete\}$,管道入口 tp_enter,管道出口 tp_out,管道两端属于不同的域 $tp\_enter \wedge tp\_out \subset Domains$,管道具备操作分为连接和断开 2 种状态 $tp\_operation = \{tp\_connect, tp\_disconnect\}$。

操作过程如下:

1)创建管道。

主体 α_1 和 α_2 经授权后存在于通用域和 1 级别安全域中。管道入口 tp_enter 和管道出口 tp_out 经过认证后分别加入到通用域和 1 级别安全域中。描述如下:

$\exists \alpha_1, \alpha_1 \in Subject, Common\_d\_0(\alpha_1) = subject\_auth\_domain(\alpha_1)$, $\exists \alpha_2, \alpha_2 \in Subject, Security\_d\_1(\alpha_2) = subject\_auth\_domain(\alpha_2)$, $Common\_d\_0(tp\_enter) = subject\_auth\_domain(tp\_enter)$, $Security\_d\_1(tp\_out) = subject\_auth\_domain(tp\_out)$。

主体 α_1 和 α_2 经授权后使用管道,通过管道出入口建立管道连接 $tp\_operation = tp\_connect(tp\_enter, tp\_out)$。临时通信管道创建成功后,管道状态为 $tp\_state = Ready$。

2)管道信息传递。

管道创建成功后,管道入口 tp_enter 处读取通用域中的授权主体 α_1 要传输的信息指令,对信息指令验证后传递至管道出口 tp_out 处。管道出口处将接收的信息进行完整性验证,判断信息传递过程中是否被篡改。信息验证成功后,管道出口将信息传递给主体 α_2。由 α_2 在 1 级安全级别域中进行操作。在信息传递整个过程中,管道状态为 $tp\_state = Using$,且一个管道仅可供一个主体进程使用。

3)管道撤销。

当操作结束后,管道状态由 $tp\_state = Using$ 转变为 $tp\_state = Ready$,管道出入口 tp_enter 和 tp_out 从域中删除,管道建立的连接被断开 $tp\_operation = p\_disconnect$。此时,管道的状态为 $tp_state = Delete$且不能被使用,同时,管道建立时所占用的资源也将被释放。

3　模型策略安全性证明

本文基于 DTE 域隔离技术、BLP 的机密性策略和授权访问策略设计了域隔离模型 MDSDroid,其安全目标是通过域隔离阻止安全域内部和外部主体非授权地访问被保护的资源,确保域之间不存在损害安全策略的信息流动。MDSDroid 模型以及相关策略提出后,对其进行形式化定义和形式分析来验证模型的安全状态及策略有效执行情况是有必要的。本文采用 Z 语言对模型及相关策略进行形式化描述,并借助 Z/EVES 工具进行形式化分析。

本文以授权通用域主体访问 1 级别安全域时的访问控制策略为例进行形式化分析。首先,模型的初始状态定义为如下模式。

$MDSDroidState$

$Subjects : P\ Subject$

$Objects : P\ Object$

$Domains : P\ Domain$

$Types : P\ Type$

$sbujectdomain : Subject \mid\!\rightarrow Domain$

$objecttype : Object \mid\!\rightarrow Type$

$DDT : P\ Domain \times Domain \times DCM$

$DTT : P\ Domain \times Type \times DTAM$

$dom\ sbuject\_domain : Subjects$

$dom\ object\_type : Objects$

其中,$Domain ::= Common\_d\_0 \mid Security\_d\_1$; $Type ::= Common\_t\_0 \mid Security\_t\_1$; $DCM ::= auto \mid exec \mid signal \mid null$; $DTAM ::= r \mid w \mid x \mid d \mid a$。

授权主体访问较高级别安全域进行操作时,需通过建立的通信管道才能进行信息的传输。所以,模型状态模式 MDSDroidState 扩充为模式 T-MDSDroidState。

$T\text{-}MDSDroidState$

$MDSDroidstate$

$TPs : P(seq(Domain \times DTAM \times Type \times DTAM \times Domain))$

$\forall tp : seq(Domain \times DTAM \times Type \times DTAM \times Domain) \mid tp \in TPs$

$\cdot (\forall i : 2..\#tp \cdot (tp\ i).1 = (tp(i-1)).5)$

$\wedge (\forall j : 1..\#tp \cdot (tp\ j).1, (tp\ j).3, (tp\ j).2) \in DTT$

$\wedge ((tp\ j).5, (tp\ j).3, (tp\ j).4) \in DTT)$

管道 tp 的定义由模式谓词部分可以看出,管道由 $Domain \times DTAM \times Type \times DTAM \times Domain$ 五元组构成的序列[22]。管道的输入输出为域,保证域与域

之间信息流动的安全性。设序列 *tp* 第一项五元组的第 1 个元素为 *de*,即(*tp*1).1 = *de*,设最后一项五元组的第 5 个元素为 *do*,即(*tp*#*tp*).5 = *do*,则表示 *de* 为管道的输入,*do* 表示为管道的输出,即 *de*(*tp*)*do*。管道的输入输出是唯一的,且管道仅可被授权主体使用,完成相关操作后管道将被撤销,防止被恶意进程利用。

首先证明初始状态的安全状态,假设初始状态为如下模式。

InitMDSDroidState

MDSDroidState

Subjects = Ø

Objects = Ø

Domains = Ø

Types = Ø

sbuject_domain = Ø

object_type = Ø

DDT = Ø

DTT = Ø

TPs = Ø

证明定理:

theorem InitMDSDroidState

∃ *MDSDroidState* · *InitMDSDroidState*

即初始状态模式⇒状态安全定义模式。

对于每一个安全策略规则存在相应的状态转换定理来证明策略规则不影响模型的安全性,即策略规则添加前后模型均处于安全状态。

对于每一个策略实施 Implement,需证明策略能有效实施且策略实施前后不影响整个模型的安全性。证明定理:

theorem InitImplementSecure

InitMDSDroidState ∧ *Implement*⇒*MDSDroidState*'

theoremImplementSecure

MDSDroidState ∧ *Implement*⇒*MDSDroidState*'

以 T-MDSDroidState 为例,它表示授权主体对较高级别安全域进行操作时通过创建通信管道进行信息流动。需要证明的定理为:

theoremInitT-MDSDroidStateSecure

InitMDSDroidState ∧ *T-MDSDroidState*⇒*MDSDroidState*'

theoremT-MDSDroidStateSecure

MDSDroidState ∧ *T-MDSDroidState*⇒*MDSDroidState*'

模型的策略操作规则等应包含安全相关的所有操作,如域和型标签的增加和删除、域和域之间、域和型之间的访问控制关系等。由于篇幅限制,本文不再一一列举。其中,定理 *InitMDSDroidState*, *InitT-MDSDroidStateSecure*, *T-MDSDroidStateSecure* 可在工具 Z/EVES 的辅助下自动完成。对于以上定理的证明可反映出在保证模型的安全状态下实施了定义的安全策略。

4 MDSDroid 模型设计

多域隔离系统的总体架构如图 2 所示。多域隔离系统基于 Linux 安全模块框架并结合 DTE 隔离策略、BLP 机密性策略以及授权访问控制策略实现了高强度的强制访问控制机制。Linux 安全模块允许安全模块以插件形式进入内核,以便更严格地控制 Android 系统中自主访问控制的安全性,且提供安全钩子(hook)函数来管理内核对象的安全域和仲裁内核对象的访问。安全模块提供访问控制向量缓存用于提高查找安全策略效率,对象管理器用于管理客体的安全标识,安全服务器做出访问决策。策略管理工具主要负责对安全策略进行调整。域管理中心以图形化的方式进行配置和管理,包括应用程序安全等级的设置、标签管理、安全域的扩充等。

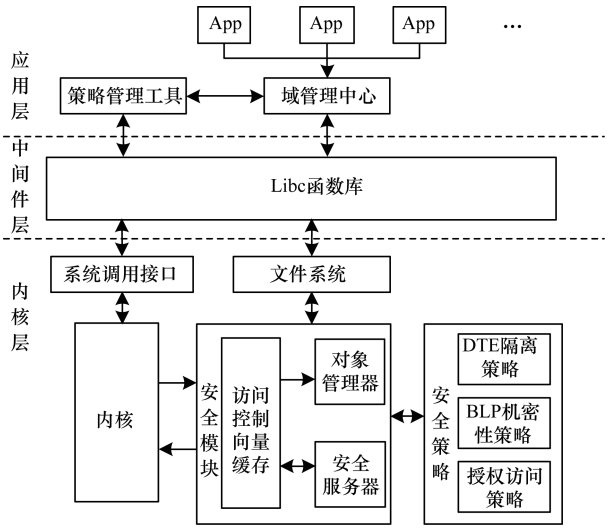

图 2 多域隔离系统总体架构

多域隔离系统在自主访问控制(DAC)基础上采用混合策略来保障 Android 系统的安全性。访问控制流程如图 3 所示。

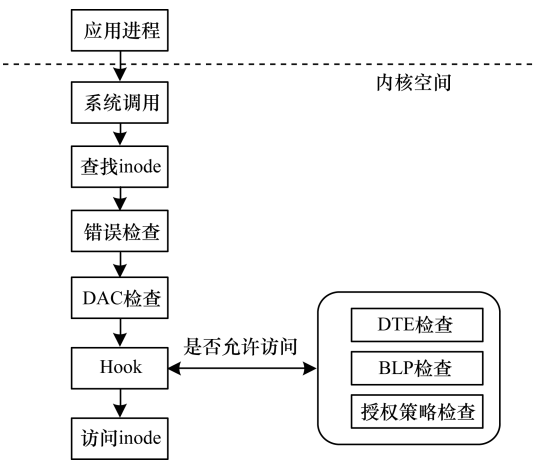

图 3 访问控制流程

当主体访问客体时,需要经过一系列的检查之后才能访问成功,应用进程通过系统调用查找到客体的 inode 节点,基本的错误检查通过后,再经自主访问控制(DAC)的权限检查,之后又通过 Linux 安全模块进程强制访问控制,检查包括 DTE 检查、BLP 检查和授权访问策略检查,只有所有的检查都通过后,才能访问客体。

5　实验结果与分析

本文实验在 Android4.1.2 模拟器上部署模型系统进行功能测试和性能测试。功能测试的目的在于检测强制访问控制机制能否有效执行。性能测试的目的在于测试系统的性能损耗。

5.1　功能测试

功能测试以 MDSDroid 模型创建 3 个域(通用域、1 级别安全域和 2 级别安全域)为例进行测试。从 Android 应用市场下载游戏应用程序安装至通用域中,淘宝、京东、qq 等应用程序安装至 1 级别安全域中,支付宝、网银等应用程序安装至 2 级别安全域中,应用程序间进行互相访问操作,并使用 AppGuard 来监控应用程序的执行情况。程序执行结果如表 1 所示。

表 1　应用程序执行结果

操作	是否需要授权	是否授权	操作结果
京东/淘宝使用支付宝进行支付	是	是	成功
京东/淘宝使用支付宝进行支付	是	否	失败
游戏程序购买装备使用网银进行支付	是	是	成功
游戏程序购买装备使用网银进行支付	是	否	失败
淘宝/京东链接分享至 QQ	否	-	成功
支付宝访问淘宝	否	-	成功

由应用程序执行结果可以看出,低安全级别域中应用程序访问高安全级别域中应用程序时需进行授权,否则无法访问。相同安全级别域中的应用程序间无需授权,可以互相访问。高安全级别域中应用程序访问低安全级别域中应用程序时无需授权,可直接进行访问。实验结果表明,域间强安全访问控制机制能有效执行。

5.2　性能测试

Android 系统中的应用程序执行时被编译为 Dalvik 字节码并由 Dalvik 虚拟机执行。性能测试采用 Java 度量基准(Java Microbenchmark)来评估系统的性能损耗并使用 CaffeineMark3.0 工具[23]完成的 Java 度量值来测试 Android 原生系统和 MDSDroid-2、MDSDroid-3 系统的性能损耗情况。其中,MDSDroid-2 表示 MDSDroid 模型创建 2 个域(通用域和 1 级别安全域);MDSDroid-3 表示 MDSDroid 模型创建 3 个域(通用域、1 级别安全域和 2 级别安全域)。

从 Android 应用市场随机下载 12 个应用程序,这些应用程序包括:游戏、音乐等娱乐应用程序,微信、qq 等聊天应用程序,支付宝、网银等应用程序。将这些应用程序安装至 Android 原生系统和 MDSDroid-2 系统、MDSDroid-3 系统的不同域中。在系统上执行 CaffeineMark,度量结果如图 4 所示。

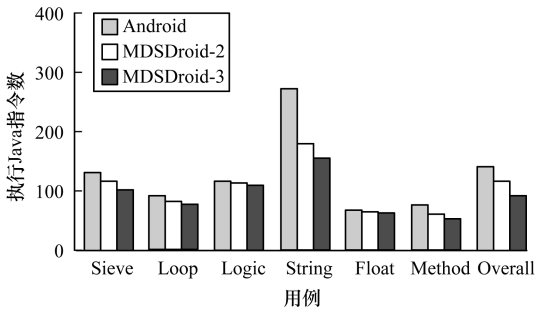

图 4　性能测试度量结果

CaffeineMark 以度量每秒执行的指令数目来判断程序的执行速度,进而反映出系统的性能损耗。由图 4 中的测试数据可知,MDSDroid 模型与原生系统相比,字符串操作产生的性能损耗相对于其他几个操作较为明显,而筛法运算、循环运算、逻辑运算、算术运算、方法运算的性能损耗相对较小。且随着 MDSDroid 创建域的数目的增多,各项度量指标有一定程度下降,系统的性能损耗则有一定程度的增加。根据各个独立基准的平均值 *overall* 可以判断出 MDSDroid 与原生系统相比性能有一定的下降,但下降幅度不大。

另外,为能直观地体现 MDSDroid 模型对 Android 系统的性能损耗情况,本文采用了性能评测工具安兔兔(AnTuTu)[24]来进行评估。根据 Android 系统性能使用特点,对几个主流的性能指标进行测试。测试主要从内存性能、CPU 整数性能、CPU 浮点性能进行评分,使用安兔兔性能测试软件测试 100 次,取平均值。性能损失比率即被测系统测试项得分与原生系统该项得分差值占原生系统该项得分的比例,如表 2 所示。

表 2　性能损失比率　　%

测试项	MDSDroid-2	MDSDroid-3
内存性能	4.36	4.72
CPU 整数性能	8.24	8.63
CPU 浮点性能	3.32	3.53

由表 2 可知,MDSDroid 与原生系统相比性能有一定的损耗,但在可接受范围内。由于本文实验在模拟器上进行测试,存在一定的误差,但对 Android 多域隔离系统的研究有一定的借鉴意义。

6 结束语

本文从保护用户隐私数据的角度出发,分析近年来 Android 系统在保护隐私数据方面的研究,设计一种多域安全隔离模型 MDSDroid,并在 Android 系统上设计了实现框架。将主体分配到不同的域中,不同的客体分配不同的型,通过安全策略来判断域与域和域与型之间的访问权限,并采用 Z 语言和 Z/EVES工具对模型进行形式分析和验证,实现了强安全访问控制机制,在增强 Android 系统安全的同时达到了保护隐私数据信息的目的。下一步将在系统中引入加密模块,从而实现较高安全级别域中信息传输过程中的安全保护。

参考文献

[1] Gartner February Report [EB/OL]. [2016-02-18]. http://www.gartner.com/newsroom/id/3215217.

[2] 李静华,慕德俊,杨鸣坤,等. Android 恶意程序行为分析系统设计[J]. 北京邮电大学学报,2014,37(增刊):104-107.

[3] ELISH K O, SHU X, YAO D, et al. Profiling User-trigger Dependence for Android Malware Detection[J]. Computers & Security,2015,49(1):255-273.

[4] WEN W P, RUI M, GE N, et al. Malware Detection Technology Analysis and Applied Research of Android Platform[J]. Journal on Communications, 2014(8):78-85.

[5] 腾讯移动实验室 2016 年安全报告[EB/OL]. [2016-08-16]. http://www.199it.com/archives/507461.html.

[6] PROTSENKO M,MULLER T. Android Malware Detection Based on Software Complexity Metrics[C]//Proceedings of Trust,Privacy,and Security in Digital Business. Berlin, Germany:Springer,2014:24-35.

[7] DESHOTELS L, NOTANI V, LAKHOTIA A. DroidLegacy: Automated Familial Classification of Android Malware[C]// Proceedings of ACM SIGPLAN on Program Protection and Reverse Engineering Workshop. New York,USA:ACM Press, 2014:1-12.

[8] LI T,HANG D,YUAN C,et al. Description of Android Malware Feature Based on Dalvik Instructions[J]. Journal of Computer Research & Development, 2014, 51(7):1458-1466.

[9] JIANG S. A Summary on Android Security[J]. Computer Applications & Software,2012,29(10):205-210.

[10] 张玉清,王 凯,杨 欢,等. Android 安全综述[J]. 计算机研究与发展,2014,51(7):1385-1396.

[11] 卿斯汉. Android 安全研究进展[J]. 软件学报,2016,27(1):45-71.

[12] ZHANG Y,KAI W,YANG H,et al. Survey of Android OS Security[J]. Journal of Computer Research & Development,2014,51(7):1385-1396.

[13] FANG Z, HAN W, LI Y. Permission Based Android Security:Issues and Countermeasures[J]. Computers & Security,2014,43(6):205-218.

[14] MOONSAMY V, RONG J, LIU S. Mining Permission Patterns for Contrasting Clean and Malicious Android Applications[J]. Future Generation Computer Systems, 2014,36(3):122-132.

[15] HOLAVANALLI S,MANUEL D,NANJUNDASWAMY V, et al. Flow Permissions for Android[C]//Proceedings of IEEE International Conference on Automated Software Engineering. Washington D. C., USA: IEEE Press, 2013: 652-657.

[16] FELT A P, CHIN E, HANNA S, et al. Android Permissions Demystified[C]//Proceedings of ACM Conference on Computer and Communications Security. New York,USA:ACM Press,2011:627-638.

[17] YANG Z, YANG M. LeakMiner: Detect Information Leakage on Android with Static Taint Analysis[J]. Software Engineering,2013,18(20):101-104.

[18] BUGIEL S,DAVI L,DMITRIENKO A,et al. Practical and Lightweight Domain Isolation on Android[C]// Proceedings of ACM Workshop on Security and Privacy in Smart Phones and Mobile Devices. New York,USA: ACM Press,2011:51-62.

[19] ENCK W,GILBERT P,HAN S,et al. TaintDroid: An Information-flow Tracking System for Realtime Privacy Monitoring on Smart Phones[J]. ACM Transactions on Computer Systems,2014,32(2):393-407.

[20] LANGE M,LIEBERGELD S,LACKORZYNSKI A,et al. L4Android: A Generic Operating System Framework for Secure Smart Phones[C]//Proceedings of the 1st ACM Workshop on Security and Privacy in Smart Phones and Mobile Devices. New York, USA: ACM Press, 2011: 39-50.

[21] SUN Q,QI T,YANG T,et al. An Android Dynamic Data Protection Model Based on Light Virtualization[C]// Proceedings of IEEE International Conference on Communication Technology. Washington D.C.,USA:IEEE Press,2013:65-69.

[22] 卿斯汉,李丽萍,何建波,等. 基于 DTE 策略的安全域隔离 Z 形式模型[J]. 计算机研究与发展,2007,44(11):1881-1888.

[23] Pendragon Software Corporation. CaffeineMark3.0 [EB/OL]. [2016-08-01]. http://www.benchmarkhq.ru/cm30/.

[24] AnTuTu Benchmark Home Page[EB/OL]. [2016-07-22]. http://www.antutu.com/en/index.shtml.

编辑 索书志