Computer Science and Application 计算机科学与应用, 2016, 6(2), 65-71
Published Online February 2016 in Hans. http://www.hanspub.org/journal/csa
http://dx.doi.org/10.12677/csa.2016.62008

Hans汉斯

Android Reinforcement Scheme Based on the Container

Haoliang Cui, Tianchang Yang, Shaozhang Niu

Beijing Key Lab of Intelligent Telecommunication Software and Multimedia, Beijing University of Posts and Telecommunications, Beijing
Email: cui.haoliang@163.com

Received: Feb. 4th, 2016; accepted: Feb. 22nd, 2016; published: Feb. 25th, 2016

Abstract

Android which takes up most of the market share of smart mobile open-source platform is facing the increasingly serious security threat. Although the Android system itself provides a set of security mechanism to protect the safety of the system and application, there is still a security risk. In order to protect the security of the Android smartphone, on the basis of in-depth analysis of Android security mechanism, using the safety testing model to classify, collect and isolate the applications, the mechanism of the safe container is formed. It can be effective to protect the Android system, the application and users' personal data.

Keywords

Container, Security Context, Feature

基于容器的Android加固方案

崔浩亮，杨天长，牛少彰

北京邮电大学智能通信软件与多媒体北京市重点实验室，北京
Email: cui.haoliang@163.com

收稿日期：2016年2月4日；录用日期：2016年2月22日；发布日期：2016年2月25日

文章引用: 崔浩亮, 杨天长, 牛少彰. 基于容器的 Android 加固方案[J]. 计算机科学与应用, 2016, 6(2): 65-71.
http://dx.doi.org/10.12677/csa.2016.62008

摘　要

Android作为占用大部分市场份额的智能移动开源平台，正面临着日益严重的安全威胁。虽然Android系统本身提供了一套安全保护机制来保护系统和应用的安全，但是依然存在着安全风险。为了保护Android智能手机安全，在深入分析Android系统安全机制的基础上，利用安全检测模型对应用进行分类汇总隔离形成安全容器的机制，可以有效的对Android系统及应用进行保护，同时有效地保护用户的个人数据。

关键词

容器，安全上下文，特征

1. 引言

近年来，在移动设备在全球范围内得到了飞速的发展。智能手机作为移动设备的代表得到了前所未有的普及和发展。市场研究公司 Gartner 发布的数据显示，2015 年第一季度智能手机销量增长了 19%，智能手机的市场份额在不断的增大，同时智能手机的更新速度也在不断的增加。

智能手机做了其手机的通讯功能越来越得到弱化，但计算功能和互联功能在人们的生活中起到了不可或缺的地位，人们开始使用智能手机来处理日常事务。由于智能手机存储着用户的账号、密码等敏感数据和用户的个人照片和私密文件等隐私数据，因此极易受恶意软件的攻击。腾讯移动安全实验室 2015 年上半年手机安全报告中指出上半年新增 Android 病毒包数达到 596.7 万，同比增长 1741%。Android 由于它的开放性导致恶意应用的开发成本和难度大大降低了[1]。虽然 Android 系统提供了沙盒、权限和签名等安全机制，但是 Android 平台还存在一些问题和漏洞，降低了平台的安全性。

由于 Android 在智能手机的份额在不断的增加，而且它还有很大的发展潜力，学者们也对它的安全机制进行了诸多研究，如何完善访问控制、防止应用的权限滥用、防止用户隐私数据泄露等问题进行研究。本文将对分析 Android 自身的安全机制以及面临的安全威胁，再综合分析现有对 Android 系统和应用的检测技术，提出一种基于安全容器的系统加固方案。安全容器的加固方案利用了 Android 的权限检测技术，将应用本身真是需要的权限进行分类汇总，根据不同的权限的组合可以将应用放置到特定的安全容器下，再利用认证机制来认证不同容器之间的进程通信，保证不同类型容器之间的通信是安全的并且内容是授权的，同时本方案还会在系统启动的过程中利用信任链机制，上下层级之间相互认证，保证启动的环境不被恶意篡改或删除。

2. Android 体系的安全分析

Android 系统采用分层的体系结构，分别为：应用层、应用程序框架层、系统运行时库和 linux 内核层(见图 1)。

2.1. Android 安全机制

Android 系统采用分层的系统架构进行设计，每层都有其严格的安全规范和强健的安全架构。Android 的主要安全机制在内核层和系统框架层。

在内核层中，Android 是基于 Linux 内核的安全特性，为上层提供了灵活的自主访问控制，但是由于过于灵活，Android4.4 后又引入了强制访问控制进行限制，通过安全策略来弥补自主访问控制中过于灵

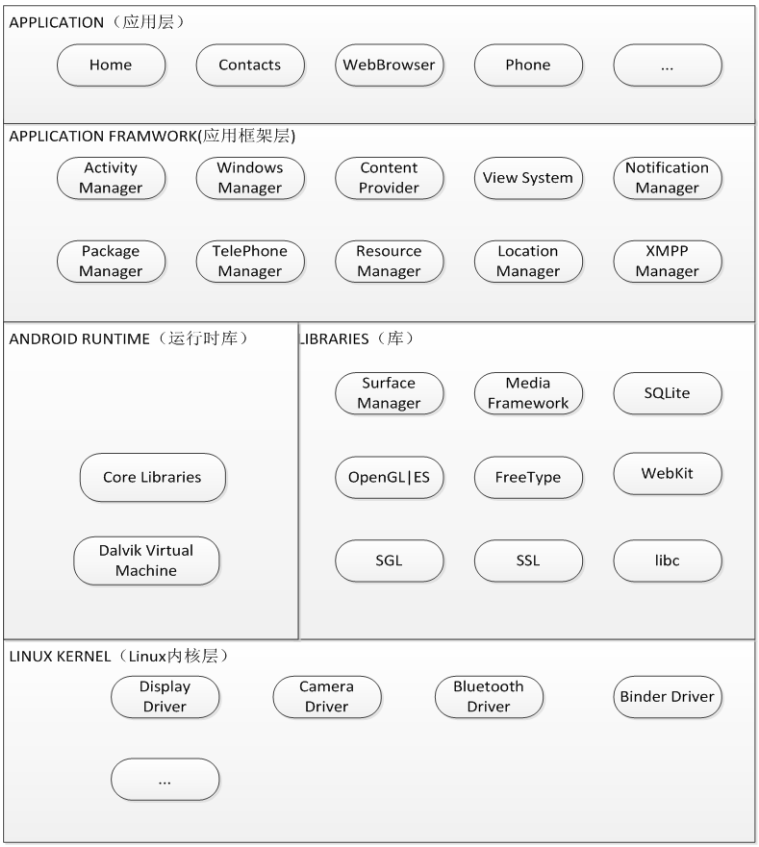

Figure 1. Android architecture
图 1. Android 体系结构

活所产生的不足。

在系统框架层，Android 则采用了权限机制、签名机制和沙箱机制，来保护系统的安全，隔离不同进程之间的资源访问，授权访问系统资源和服务，禁止非授权访问的发生。权限机制是 Android 系统框架层提供的，是对应用访问公共资源和服务进行强制访问的安全策略。应用如果要使用公共资源和服务，必须要在配置文件中声明需要使用对应资源的权限，否则将无法得到授权，致使不能使用其资源。但是权限机制采用的全部肯定和全部否定的方式，把最终的确定权交给了用户，用户在不具备相关的安全知识的情况下是很难做出合适的判断的。签名机制是应用中包含一个采用非对称加密方式存储的数字证书，由于私钥在开发人员手中，所以数字证书能够保证开发人员对其应用的合法拥有权。虽然 Android 系统提供了签名机制来保护开发人员的合法拥有权，但是并没有提供验证应用是否被恶意的二次重打包，这使得应用被注入一些恶意的代码，来获取非法的收益或达成恶意的目的。沙箱虽然提供了进程隔离的机制，保证运行时应用的进程空间不会被恶意的修改，但是沙箱并没有在运行时验证是否运行环境是否安全，是否系统的关键函数的地址发生了改变，一旦系统的在进入沙箱前就已经被恶意修改，沙箱中的应用也将受到恶意的威胁。

2.2. Android 系统中存在的安全风险

Android 系统采用分层的体系结构，层与层之间都使用了解耦合的方式，这样不但增加了模块自身的通用性还增加了模块之间移植的灵活性。但是 Android 系统并没有对各模块之间的通信进行有效的验证。Android 系统在启动的过程中也没有加入完整的认证链，来认证每个环节的完整性，例如 Xposed 的 Hook

框架就是通过替换系统app_process文件来实现在沙箱启动之前对系统进行修改，完成对自身程序的加载。

Android 应用主要是面临的威胁是通过静态分析、动态分析和动静结合的方式，来获取应用中潜在的漏洞信息或应用本身的流程图。根据这些信息黑客可以针对应用存在的漏洞进行特定的攻击，从而获取非授权的服务或者获取用户的隐私数据，进而导致用户的名誉或财产的损失。所以为应用提供一个安全的运行环境是至关重要的，将用户的数据隔离在指定的区域内，禁止外传或将其加密进行外传，来保证用户的隐私数据不会被恶意利用。

Android 系统采用沙箱机制实现进程之间的隔离，使得进程之间不能互相访问，进而保护了进程本身的安全，防止被动态篡改的发生。但是 Android 并没有对应用间的通信进行有效的验证，也没有对通信内容进行适当的过滤。因此 Android 的恶意应用将其功能分散到多个应用中，利用这多个应用所获取的权限来实现共谋攻击，获取用户的隐私数据。

3. 现有安全技术存在的问题

Android 安全技术主要从安全检测、系统增强等方面进行深入研究。安全检测是通过对系统和应用进行安全扫描，检测系统中是否隐含接口和多余的无用接口和应用中声明的权限是否符合最小权限集合。加固增强则是对 Android 系统中出现的不足进行加强和对权限的细粒度化，动态化和对访问控制的加强。

对应用的权限检测主要是检测应用是否违背了最小权限的原则，过度申请了权限。Felt 等人检测出权限过度申请的应用中有 56%的应用存在一个没有使用到的权限，94%的应用存在小于 4 个无用的权限[2]。这说明过度申请权限的现象很普遍的，所以对应用的权限检测，对 android 平台的安全是至关重要的。

由于传统的基于系统调用的分析方式是无法获取组件间的通信情况的，所以 Yuan Zhang 提出了针对 Android 平台的系统权限分析技术，记录应用在运行时产生的行为，分析应用的这些操作行为是否存在敏感行为，并可以重现敏感操作行为[3]。这样可以全面了解应用访问了哪些系统资源，是否存在敏感数据的泄露的发生。

虽然 Yuan Zhang 对权限检测模型进行了改进，但是并没有提供可视化的呈现方法给用户。Christopher S. Gates 等人提出了一种如何最好的向用户传达风险信息的呈现方式来告诉用户可能存在的安全风险，使得用户能够直观的感受存在的风险[4]。同时 Christopher S. Gates 等人也提出来一种基于机器学习技术为基础的应用检测技术，能够对应用进行恰当的风险评估分数[5]。由于移动设备的计算能力不足，Saman Zonouz 等人提出了利用云平台来实时和移动端进行数据交互，同步移动端的输入和输出，从而将移动应用的检测放于云端上，解决了移动端计算能力的限制[6]。

虽然现在对权限检测的正确率和准确度在不断提升，但是这些检测的结果并没有直接反馈给系统本身，还是需要用户来确定是否安装应用。所以我们提出了利用权限的检测模型的结果来确定应用本身的环境空间，这样使得系统和应用更佳安全。

在权限的访问控制方面，Mauro Conti 等人提出一个针对 Android 的细粒度的上下文相关的策略执行系统 CRêPE 系统[7]，但是还是需要用户来配置安全策略来确保应用本身的安全。根据存在的这个问题，我们提出利用应用的检测模型的结果来确定应用本身的安全配置策略的方案，使得用户不必要熟悉具体的安全策略，系统会默认给出安全策略和初始化对应安全策略所需的环境，从而保护应用的安全和应用数据的安全。

4. 基于容器的安全加固方案

容器的概念最早是由 G. Banga 等人提出的，其主要是为了应对如何提高应用程序对于系统的各类资

源的控制和管理[8]。容器的目的是为了加强应用程序对系统的程序对资源文件的访问控制，构建一个隔离的、独立的执行环境来保护应用自身的安全和产生数据的安全，防止程序被恶意破坏和防止应用产生的隐私数据被窃取。

容器的作用是容器可以管理加载的应用程序，自动配置对应的安全策略，检测被访问的服务是否是合法授权的，也能防止像重放攻击这样的恶意行为，减少应用自身的暴露区域。

4.1. 方案介绍

我们采用利用 Android 的应用检测模型来对应用进行分类汇总，利用静态分析、动态分析和静动结合的方法，将不同种类的应用进行划分，检测应用中所有使用到的权限和应用中是否存在恶意的代码，根据检测的特征对应用进行分类，赋予不同的安全上下文。应用启动初期会根据检测结果赋予的应用的安全上下文初始化对应的容器来加载此应用程序。每个容器都有不同的安全策略和本身特有的保护机制来保护自身不受到威胁。我们会根据检测模型中生成的每个安全上下文所对应的特征来确定不同容器间的通信是否允许访问，这样保护应用不会被恶意的访问的发生，为应用提供安全良好的运行环境。安全检测流程如图 2。

通过 Android 应用检测分析技术，获取应用中信息流和系统接口调用的特征，利用这些特征进行机器学习的训练，从这些特征归纳出应用的主要功能和应用所需权限的最小集合，根据机器学习的方式获取到的应用的主要功能可以为应用赋予真正的应用的安全上下文，应用所需的最小权限集合同样也能合理的检测当前应用是否存在权限过度申请的问题和是否存在恶意的申请权限的问题，这样利用应用检测的结果就不用将权限选择的责任直接推给应用使用者，保证用户在不了解权限的情况下也能保护自己的隐私数据不被泄露。

Android 采用的沙箱来隔离应用的运行环境，容器则为沙箱和应用提供了一个外部的过滤机制来保护沙箱和应用自身的环境不受到安全威胁。根据不同应用之间的共性来进行分类，将应用分类到不同的安全上下文中，然后再根据应用之间的差异性来加固安全上下文所属的容器的安全策略，这样充分利用了应用本身的共同点和差异处，使得容器的保护更加具有针对性和安全性。我们利用 Android 内核中的 Binder 通信来实现不同容器之间的隔离[9]，如图 3。

对于容器间的通信，我们则采用在 Android 基础的进程通信的基础上进行通信内容的安全审查，通过容器来隔断非授权的应用请求和防止应用的数据在授权情况下，进行非法的传播。我们将利用加密机制来保护用户的隐私数据，在用户的隐私数据在传递出容器的时候，容器将隐私数据进行加密，保护只有合法的授权容器才能正确的接收隐私数据，进而防止数据劫持的发生。对于那些拥有访问权限，但没有获得容器解密权限的应用，获取的隐私数据只能用来标记用户信息，不能用来识别出用户的信息。对于容器内的通信，由于具有相同或类似的安全上下文，应用之间申请的权限的共性很高，应用彼此的依赖性就很低，所有将不会对容器内的通信进行安全限制。

4.2. 方案安全性分析

Android 系统没有提供认证机制来保护系统在启动过程中的完整性和一致性，所以 Android 系统存在系统文件被恶意替换的情况发生，使得恶意的应用可以被加载运行。针对这个问题，我们将利用双向认证机制来保护容器在启动的过程中的安全，如图 4。

在容器启动初期就会验证下一个节点的完整性及其依赖资源文件的完整性，同时下一个节点在启动后也将验证上一个节点的完整性，检测是否存在被恶意替换的情况。这样的双向验证保证容器在启动的时候的安全。同时 Android 系统存在是否可以动态修改代码可对应依赖的关系的现象，容器中为了防止

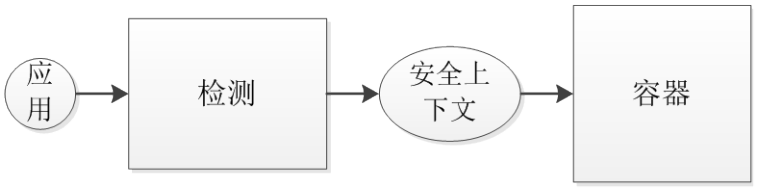

Figure 2. Safety vessel inspection process
图 2. 安全容器检测流程

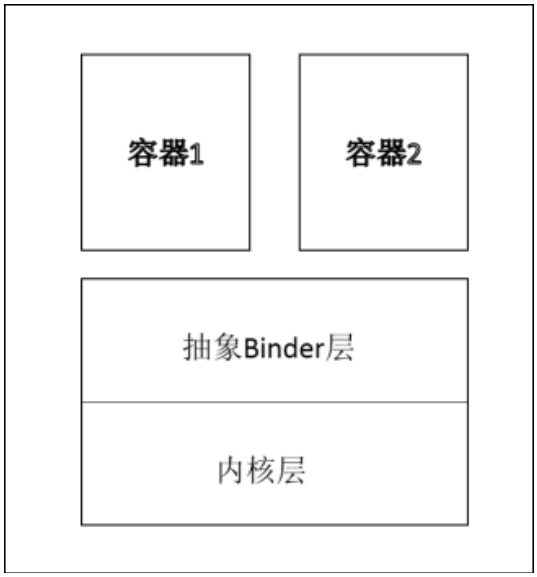

Figure 3. Implementation of container isolation
图 3. 容器隔离的实现

Figure 4. Container start bidirectional authentication process
图 4. 容器启动双向认证过程

系统函数和依赖库被修改的发生，容器在运行的过程中，会随时监控容器自身的内存空间中的函数依赖关系是否发生了变化，如果发生变化则将恢复为原来的依赖关系中，防止应用本身被恶意修改。如图 5。

5. 结束语

Android 采用的沙箱来进行进程隔离，但是 Android 系统没有参考应用之间的共性和差异性，导致系统不能区分不同沙箱之间的通信行为是否是恶意的行为。该模型利用机器学习机制来检测不同应用之间的共性和差异性，将应用进行有效的归类汇总，将不同应用赋予一个固定分类的安全上下文，系统将根据应用自身的安全上下文来进行应用所在容器的初始化工作。容器本身保护着应用和沙箱的运行环境，同时过滤和授权通过的数据流，使得用户的隐私数据不会被恶意的使用和传播。为了保护应用在运行时的安全，即使应用存在潜在的漏洞也能在容器之间进行隔离，保证应用的运行不会被恶意的试探和其它动态监测。同时，在用户未授权的情况，访问用户的敏感数据并发送到应用之外的时候，容器可以将用

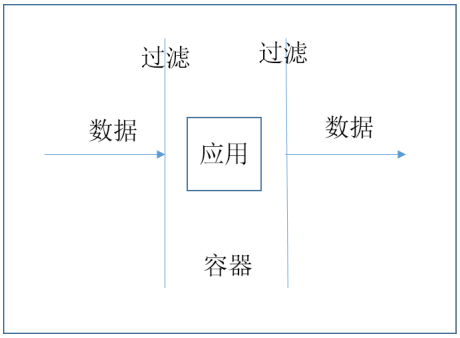

Figure 5. Isolation mechanism for application communication
图 5. 应用通信的隔离机制

户特定的敏感数据进行加密，防止在未授权的情况下被恶意传播。

基金项目

国家自然科学基金项目(61070207, 61370195)。

参考文献 (References)

[1] 腾讯移动安全实验室 2015 年上半年手机安全报告[EB/OL]. http://m.qq.com/security_lab/news_detail_321.html

[2] Felt, A.P., Chin, E., Hanna, S., Song, D. and Wagner, D. (2011) Android Permissions Demystified. *Proceedings of ACM CCS*. *ACM*, Chicago, 17-21 October 2011, 627-638.

[3] Zhang, Y., Yang, M., Yang, Z.M., Gu, G.F., Ning, P. and Zang, B.Y. (2014) Permission Use Analysis for Vetting Undesirable Behaviors in Android Apps. *IEEE Transactions on Information Forensics and Security*, **9**, 611-622.

[4] Gates, C.S., Chen, J., Li, N.H. and Proctor, R.W. (2014) Effective Risk Communication for Android Apps. *IEEE Transactions on Dependable and Secure Computing*, **11**, 252-265.

[5] Gates, C.S., Li, N.H., Peng, H., Sarma, B., Qi, Y., Potharaju, R., Nita-Rotaru, C. and Molloy, I. (2014) Generating Summary Risk Scores for Mobile Applications. *IEEE Transactions on Dependable and Secure Computing*, **11**, 238-251.

[6] Zonouz, S., Houmansadr, A., Berthier, R., Borisov, N. and Sanders, W. (2013) Secloud: A Cloud-Based Comprehensive and Lightweight Security Solution for Smartphones. *Computers & Security*, **37**, 215-227. http://dx.doi.org/10.1016/j.cose.2013.02.002

[7] Conti, M., Crispo, B., Fernandes, E. and Zhauniarovich, Y. (2012) CRePE: A System for Enforcing Fine-Grained Context-Related Policies on Android. *IEEE Transactions on Information Forensics and Security*, **7**, 1426-1438.

[8] Soltesz, S., Potzl, H., Fiuczynski, M.E., *et al.* (2007) Container-Based Operating System Virtualization: A Scalable, High-Performance Alternative to Hypervisors. *ACM SIGOPS Operating Systems Review*, **41**, 275-287. http://dx.doi.org/10.1145/1272998.1273025

[9] Chen, W., Xu, L., Li, G. and Xiang, Y. (2015) A Lightweight Virtualization Solution for Android Devices. *IEEE Transactions on Computers*, **64**, 2741-2751. http://dx.doi.org/10.1109/tc.2015.2389791