Android ION Hazard: the Curse of Customizable Memory

Management System

ION is a unified memory management interface for Androidthat is widely used on virtually all ARM based Android de-vices. ION attempts to achieve several ambitious goals thathave not been simultaneously achieved before (not even onLinux). Different from managing regular memory in thesystem, ION is designed to share and manage memory withspecial constraints, e.g., physically contiguous memory. De-spite the great flexibility and performance benefits offered,such a critical subsystem, as we discover, unfortunately hasflawed security assumptions and designs.In this paper, we systematically analyze the ION relat-ed vulnerabilities from the conceptual root causes to thedetailed implementation decisions. Since ION is often cus-tomized heavily for different Android devices, the specificvulnerabilities often manifest themselves differently. By con-ducting a range of runtime testing as well as static analysis,we are able to uncover a large number of serious vulnera-bilities on the latest Android devices (e.g., Nexus 6P run-ning Android 6.0 and 7.0 preview) such as denial-of-serviceand dumping memory from the system and arbitrary appli-cations (e.g., email content, passwords). Finally, we offersuggestions on how to redesign the ION subsystem to elim-inate these flaws. We believe that the lessons learned canhelp guide the future design of similar memory managementsubsystems.

Android operating system has gained tremendous pop-ularity in the past few years thanks to the huge vendorsupport behind it. Unlike iOS that runs on only Apple-assembled hardware, Android is open source and encouragesother vendors to build smartphones using it. This modelworks well as vendors do not need to build a new OS fromscratch, and they can still heavily customize the phones todifferentiate themselves on the market. The customizationhappens at all layers including hardware, OS, and applica-tions. Major vendors such as Samsung, HTC, and HuaweiPermission to make digital or hard copies of all or part of this work for personal orclassroom use is granted without fee provided that copies are not made or distributedfor profit or commercial advantage and that copies bear this notice and the full cita-tion on the first page. Copyrights for components of this work owned by others thanACM must be honored. Abstracting with credit is permitted. To copy otherwise, or re-publish, to post on servers or to redistribute to lists, requires prior specific permissionand/or a fee. Request permissions from permissions@acm.org.CCS’16, October 24-28, 2016, Vienna, Austriac ? 2016 ACM. ISBN 978-1-4503-4139-4/16/10...$15.00DOI: http://dx.doi.org/10.1145/2976749.2978320all perform customizations to attract customers with fea-tures like better screens, audio, and even security [29]. Whilesuch customization itself is encouraged, it has been shownthat the process of customization at the software layer oftenintroduces security vulnerabilities [28, 29, 27].In this study, we investigate an important OS subsystem,called ION, that is commonly customized to satisfy differentrequirements from the underlying hardware devices. ION [8]is a unified memory management interface widely used onARM based Android platforms. First introduced by Googlein Android 4.0, it was initially designed to replace previ-ous fragmented interfaces originated from System-on-Chip(SoC) vendors [8]. Its main goal is to support the specialrequirements set by hardware devices such as the GPU andcamera. For instance, some devices require physically con-tiguous memory to operate and some require certain cachecoherency protocol for DMA to function correctly. To sat-isfy such requirements, on a given Android phone, ION iscustomized with a set of pre-configured memory heaps forthe underlying hardware devices. Even though AOSP pro-vides a set of pre-defined heap types and implementationsof heap allocation and management, customization is com-monplace for performance tuning and other purposes (as wewill show in the paper). In addition, for hardware devicesnot covered by AOSP, vendors often need to define new heaptypes as well as provide their own implementations of heapallocation and management.Unfortunately, the framework for supporting such cus-tomization is not well thought out regarding its securityimplications. For instance, we discover that the lack of fine-grained access control to individual memory heaps can eas-ily cause denial-of-service of specific system services or theentire OS. Moreover, its buffer sharing capability exposesdifferent types of kernel memory to user space without be-ing screened carefully for security consequences. To demon-strate the seriousness of the identified vulnerabilities, attackdemos and analysis can be found on our project website [1].In this paper, we make three main contributions:1) We systematically analyze the security properties fromthe design and implementation of ION, and reveal two ma-jor security flaws that lead to many vulnerabilitis and cor-responding exploits, which are already reported to and con-firmed by Google.2) To detect the specific instances of vulnerabilities, wedevelop both a runtime testing procedure and a novel stat-ic taint analysis tool that help uncover vulnerabilities onnewest flagship models like Nexus 6P and Samsung S7 run-ning Android 6.0 and 7.0 preview (latest at the time of writ-3) By analyzing the root causes of the problem, we proposean alternative design that preserves the ION functionalitywhile improving its security stance significantly. We believethe lessons learned can shed light on future designs of cus-tomizable and extensible memory management system.The remaining part of the paper will be organized as fol-lowing: §2 will briefly introduce some ION-related back-ground knowledge, §3 will give a thorough analysis of IONrelated vulnerabilities, §4 will detail our methodology tosystematically identify the vulnerabilities on a wide rangeof Android devices, §5 will summarize the vulnerabilitieswe found so far on various devices and evaluate the effec-tiveness of our methodology, §6 will demonstrate our actu-al exploitations against ION related vulnerabilities on somerepresentative devices. In §7, we discuss possible defense a-gainst the vulnerabilities we found. §8 d

Android ION危害：可定制内存管理系统的诅咒

ION是Android的统一内存管理接口，广泛应用于所有基于ARM的Android设备。 ION试图实现以前没有同时实现的几个雄心勃勃的目标（甚至不是在Linux上）。与管理系统中的常规内存不同，ION旨在共享和管理具有特定约束的内存，例如物理上连续的内存。尽管提供了巨大的灵活性和性能优势，但正如我们发现的那样，这样一个关键子系统不幸地影响了安全假设和设计。本文系统地分析了从概念根源到详细实现决策的ION相关漏洞。由于ION通常针对不同的Android设备大量定制，因此具体的通常情况会有所不同。通过进行一系列运行时测试以及静态分析，我们能够发现最新Android设备上的大量严重漏洞（例如Nexus 6P运行Android 6.0和7.0预览版），例如拒绝服务和从系统转储内存和任意应用程序（例如，电子邮件内容，密码）。最后，我们提供了关于如何重新设计ION子系统以消除这些缺陷的建议。我们相信这些经验教训可以帮助指导类似内存管理子系统的未来设计。

在过去几年中，Android操作系统凭借巨大的供应商支持而获得了巨大的成功。与仅在Apple组装的硬件上运行的iOS不同，Android是开源的，并鼓励其他供应商使用它构建智能手机。这种模式很适合厂商不需要从零开始构建新的操作系统，而且它们仍然可以在市场上大量定制手机来区别自己。定制会在所有层面出现，包括硬件，操作系统和应用程序。主要供应商如三星，宏达电和华为公司允许用于个人或课堂使用的全部或部分作品的数字或硬拷贝免费提供，前提是复制品不是为了营利或商业利益而制作或发布的，执行定制以吸引顾客喜欢更好的屏幕，音频等特性。甚至安全[29]。虽然这种定制本身受到鼓励，但已经表明，软件层的定制过程通常会引入安全漏洞[28,29,27]。在这项研究中，我们调查了一个重要的操作系统子系统，称为ION，它通常被定制以满足不同的需求来自底层的硬件设备。 ION [8]是基于ARM的Android平台上广泛使用的统一内存管理接口。谷歌首先在Android 4.0中引入了它，它最初的目的是取代源自片上系统（SoC）厂商的先前零散接口[8]。其主要目标是支持硬件设备（如GPU和相机）设置的特殊要求。例如，有些设备需要物理存储器才能运行，有些设备需要某些高速缓存一致性协议才能使DMA正常工作。为了满足这样的需求，在给定的Android手机上，ION被定制为一组预先配置的内存堆，用于底层硬件设备。尽管AOSP提供了一组预定义的堆类型和堆分配和管理的实现，但是为了性能调整和其他目的（我们将在论文中展示），定制是很常见的。另外，对于未被AOSP覆盖的硬件设备，供应商通常需要定义新的heaptypes，并提供自己的堆分配和管理实现。不幸的是，支持这种定制的框架在安全性方面没有很好的考虑。例如，我们发现缺乏对单个内存堆的细粒度访问控制很容易导致对特定系统服务或完整操作系统的拒绝服务。而且，它的缓冲区共享功能可以将不同类型的内核内存暴露给用户空间，而不需要仔细筛选安全后果。为了证明已识别漏洞的严重性，可以在我们的项目网站上找到攻击者的分析和分析[1]。本文主要做出三点贡献：1）从ION的设计和实现中系统分析安全属性，并揭示两个主要的安全漏洞，导致许多漏洞和相应的漏洞，这些漏洞已经被Google报告和确认。2）为了检测漏洞的具体实例，我们开发了运行时测试程序和小说stat-ic taint分析工具，帮助发现Nexus 6P和Samsung S7运行Android 6.0和7.0预览版（最新的时间为writ-3）等最新旗舰机型上的漏洞。通过分析问题的根源，我们提出了一种替代设计在保持ION功能的同时显着提高其安全性。我们相信吸取的经验教训可以为可定制和可扩展的内存管理系统的未来设计提供参考。本文的其余部分将按以下方式组织：§2将简要介绍一些与ION相关的背景知识，§3将对ION相关漏洞进行透彻分析，§4将详细描述我们的方法系统地识别各种Android设备上的漏洞，§5将总结我们目前在各种设备上发现的漏洞并评估我们方法的有效性，§6将展示我们在某些代表性设备上针对ION相关漏洞的实际利用。在第7节中，我们讨论了可能的防御，以获得我们发现的漏洞。