安卓基于权限的安全模型的演变与挑战

Rajendra Kumar Solanki、Vijay Laxmi 和 Manoj Singh Gaur *IEEE*高级会员 *E*

arXiv:2601.00252v1 [cs.CR] 1 Jan 2026

摘要——**自2008年推出以来，安卓权限模型和应用分析一直是安卓生态系统中研究团体及利益相关者关注的重点。尽管安卓智能手机操作系统（OS）的权限模型已从“全有或全无访问”显著演变为“由用户选择的危险资源访问”，但在智能手机操作系统推出15年后，仍有一些具体挑战和问题尚未解决。本研究通过全面的文献综述和比较分析，探讨这些问题并梳理该领域的研究工作。**

本文综述的重点是安卓权限模型以及2010—2022年间的相关研究。我们系统梳理了以下方面的知识：（i）安卓 API 调用与权限的映射；（ii）安卓权限的演变；（iii）权限的检查方式。此外，本文还归纳了过去十年中已探讨的权限相关问题及相应研究，并引用了这些领域的开创性研究。最后，我们总结了已发现的研究空白，并为初入该领域和经验丰富的研究人员提出未来方向。

索引词—**Android权限、安卓 API、数据隐私、信息安全、权限检查。**

I. 引言

A. 动机

智能手机应用的下载和使用统计数据 [1] 揭示了一个有趣的趋势：用户大部分的智能手机使用时间都花在第三方开发者应用上。智能手机用户平均每天使用十款应用，每月使用约三十款。我们使用应用来进行社交互动、阅读新闻、网上银行、电子商务、提高生产力、玩游戏等。智能手机应用经济的发展超出所有人的预期，并在过去十年中迅速扩张。超过两百五十万款安卓应用运行在数十亿台安卓设备上 [2], [3]。

随着全球智能手机用户不断增长，安卓凭借出货设备数量获得了显著的操作系统市场份额 [4], ，并成为部署最广泛的操作系统之一。这一趋势仍在持续发展。到2023年，智能手机经济预计将创造超过9350亿美元的收入[1]。如此庞大的创收平台也吸引了恶意软件编写者，他们窃取个人数据以谋取经济利益，并根据用户在安卓设备上的活动对其进行追踪。

当用户没有意识到移动应用中伪装的恶意和不受欢迎的软件时，其个人身份信息（PII）的隐私就会受到威胁 [5], [6]。终端用户会在应用上提交大量敏感数据。应用可能会请求远超实际所需的权限。这种行为使此类欺骗性应用引发隐私与安全问题[7], [8], [9]。

在安卓生态系统中，确保数据安全与隐私的责任由安卓系统、应用开发者和终端用户共同承担 [10]。鉴于用户群体可能十分多样，且过去十年间曾有恶意软件被报告，用户层似乎是执行安全与隐私相关决策时的薄弱环节。应用及其相关的负面影响会影响我们的日常生活。社交网站的选择可能影响民主 [11], [12]。视频会议应用即使在静音时，也可能监控麦克风和活动[13]。不能把这些决策完全交给用户，指望他们自行弄清一切。

因此，安卓系统必须配备合理的措施，以制定强有力的安全策略，而应用开发者应确保其得到落实。只要掌握与相应安卓 API 调用对应的功能，第三方应用开发者便能够落实此类策略决策。

为保护系统资源并实施访问控制，安卓使用权限。每项权限都通过 API 调用来管理对资源的访问。应让开发者了解特定权限如何映射到相关 API 调用。准确映射 Android API 调用与相应 Android权限具有重要意义。此类映射可用于恶意软件检测框架的安全策略 [14],、欺诈应用检测框架以及对权限过度应用二进制文件的研究 [8], [15], [16], [17]。

尽管 API 映射数据对研究具有重要影响，但除了少数有限或如今已过时的研究 [16],[17], [18], [19]之外，近期鲜有工作探讨安卓 API 在实践中如何映射到权限。这些论文通过合计引用次数影响了数千项研究；截至2022 年初的数据表明，权限规范相关研究具有重要影响和意义。本文全面分析了安卓研究出版物中 Android权限的使用情况。

我们探索的另一个方面，是从权限的角度研究安卓生态系统的演变。在这一方向上，重要的里程碑包括引入运行时权限、在安装后撤销权限、自动授予的权限、一次性权限、仅在使用期间有效的权限，以及限制默认处理程序以外的应用使用特定权限等。此外，运行时权限实际如何管理也值得关注——检查授权状态、请求并解释权限，最后存储用户对权限提示的响应结果。不请求权限以及不检查已授予的权限，可能会导致安卓运行时权限（ARP）问题 [20]。

我们首先研究最为全面的工作之一：该工作从Android 源代码以及过去十年间采用的技术入手，对Android权限规范进行了剖析。进一步地，我们展示权限模型的演变及权限检查方式，并介绍围绕这些主题开展的研究工作。

贡献：本研究的主要贡献围绕以下研究问题展开：

a) **RQ1：为安卓 API 调用与所需权限建立映射方面，已取得哪些成果？** 围绕这一问题，我们调查了为建立 API 与权限之间的映射而开展的各项研究工作，分析其相关性，以及它们在当今现代应用中的适用性。我们还考察了所提出的工具和数据集是否可用，以便复现第三节中的研究结果。

b) **RQ2：安卓生态系统在基于权限的模型方面发生了怎样的演变？** 过去十年间，随着多个安卓 API 级别的推出和发布，我们在第四节中调查了基于权限的模型如何随着安卓用户和开发者的需求而演变，以及相关研究工作取得了怎样的进展。

c)**RQ3：如何检查权限？** 关于安卓的文献浩如烟海，有时人们往往不清楚如何落实基于权限的保护；我们将在第五节从新的视角探讨这一问题。

d)**RQ4. 应解决哪些研究空白？** 最后，我们综述了过去十年研究工作尚未解决的挑战。

B. 出版物与数据集的提取

为了找出所纳入出版物中提出的机制、工具和数据集，我们利用了知名的电子文献库和出版物索引，即 IEEE Xplore、ACM 数字图书馆、ScienceDirect、SpringerLink 和 DBLP。本综述考察了 2011—2020年间一些知名会议和期刊上的出版物，例如 IEEE信息取证与安全汇刊（TIFS）、Computers & Security、IEEE Communications Survey and Tutorials、USENIX Security、ACM SIGSAC计算机与通信安全会议（CCS）、ACM Computing Surveys（CSUR）等。本综述聚焦于安全、隐私和软件工程领域的出版物，同时关注权限和隐私问题。我们遵循了一些安全与隐私研究综述的最佳实践[21]。

表 I 展示了本研究与近期发表的综述在研究差异方面的比较。尚无其他研究探讨权限映射、权限检查以及权限的演变，也未涵盖 API 级别 23 之后、引入运行时权限的情况。已有一些研究讨论了恶意软件检测框架[24], [25], [26], [27]，也有少数研究定义了安全模型[22], [25]。

组织结构：本文其余部分安排如下：第二节介绍.....所需的基本背景信息

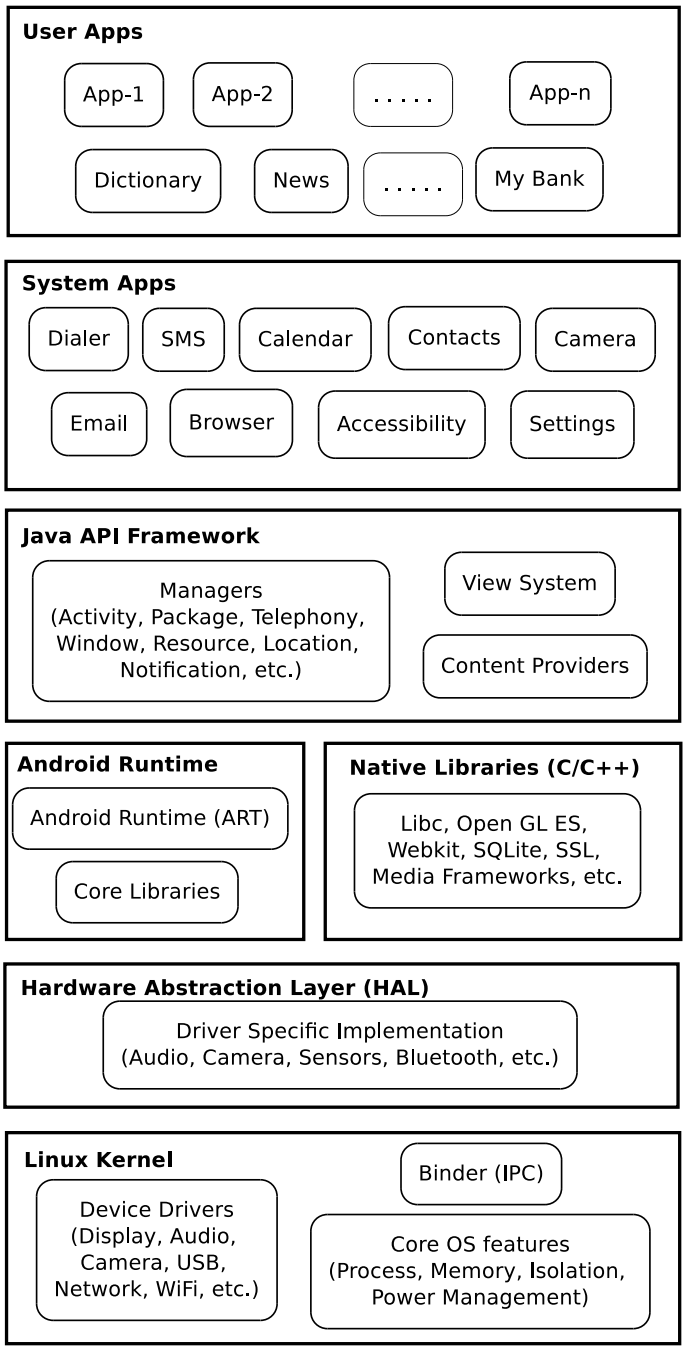

图 1：安卓架构

本研究；具体而言，我们探讨 Android 操作系统、安全模型及其面临的挑战。第三节介绍了如何精确映射 Android API 调用与权限的相关研究。第四节阐述了安卓基于权限的安全模型的主要演进与变通方案，以及这一方向上的相关研究工作。第五节调查了框架的 Java 和本机代码中如何检查安卓权限。我们在第五节 D 部分和第六节总结了从文献中获得的经验教训与未来方向，作为主要结论。最后，第七节总结本研究。

表I：关于安卓和权限的近期综述比较

相关研究	Year	权限 Map	演变 权限	权限 检查	恶意软件 检测	讨论帖 API 级别 23
本研究	2022	yes	yes	yes		yes
Mayrhofer 等人 [22]	2021		yes			
Iman 等人 [23]	2020		yes			yes
Bakour 等人 [24]	2019				yes	
Tam 等人 [25]	2017				yes	
Faruki 等人 [26]	2016				yes	
Sufatrio 等人 [27]	2015		yes		yes	

II. 安卓背景信息

在本背景章节中，我们首先在第二节 B 部分研究 Android 应用。本节回顾并解释关键术语，以帮助读者理解本研究。

A. Android 架构

安卓是一种基于 Linux 内核的开源操作系统（OS），为现代设备而创建，提供类似应用的体验和丰富的用户界面。安卓操作系统用于多种类型的设备，包括智能手机、平板电脑、智能显示屏、电视、语音数字助理等。安卓框架由六个主要层组成，如图 1 所示。这些层在堆栈中由底部到顶部依次为：（i）Linux 内核；（ii）硬件抽象层（HAL）；（iii）原生库（C/C++）和安卓运行时（ART）；（iv）Java API 框架；（v）系统应用层；以及（vi）用户应用层。

经过修改的 Linux 内核包含设备驱动程序、用于进程间通信的 Binder，以及核心操作系统功能，包括基于权限的保护。HAL 包含面向 OEM 的特定于驱动程序的实现 [28]。Java API 框架由用于在用户应用层创建 Android 应用的各种 API 组成。这些 API 包括各种服务的管理器、视图系统和内容提供程序。API 简化了核心组件的重复使用。系统应用层包含用于电话、短信、电子邮件、日历、联系人、网页浏览、相机以及更多资源的核心应用。用户应用层包含设备上预装的 Android 应用，或用户可以从应用商店下载的应用。

1) Android 安全：从系统安全角度来看，安卓操作系统的主要目标是保护系统资源、保护用户数据并为应用提供隔离。为实现这些目标，安卓操作系统提供了以下安全机制。

- **Linux 内核**：提供核心安全功能，以及使用 Linux UID/GID 实现的面向用户和组的访问控制（文件权限）、进程隔离、安全进程间通信、SE Linux 策略强制执行等。
- **强制应用沙箱**：每个进程各自运行。两个不同的应用将在两个不同的用户标识符下运行。
- **应用签名**：在 Android 设备上运行的每个应用都必须由应用开发者使用自签名证书或第三方 OEM 进行签名。应用签名让应用开发者能够识别应用的作者或发布者。签名后的应用证书定义了与应用关联的用户ID，并有助于实现应用沙箱化 [29]。
- **应用级权限**：每个应用都会声明访问系统硬件或特定功能所需的权限，随后由用户同意或拒绝。更多详细信息将在下文各节中介绍。

B. Android 应用

安卓应用或应用以压缩的 zip 对齐文件——应用程序包（APK）二进制文件的形式存储在设备上。APK 文件包含 AndroidManifest.xml、classes.dex、resources.arsc 等文件，以及 assets、lib、META-INF、res 等文件夹，如图 2 所示。AndroidManifest.xml 文件包含应用部署布局、权限、Intent 和内容提供者的详细信息。classes.dex 文件包含 Java 字节码，包括类和接口实现，并打包在单个 .dex 文件中。resources.arsc 文件是已编译和未编译资源的存档。已编译资源由字符串资源 XML 文件和布局资源 XML 文件组成，通常存储在 res/xml 子目录中。未编译资源由图像、音频和视频文件等原始文件组成，通常存储在 res/raw 子目录中。

Android 应用运行时环境：安卓应用需要虚拟机才能运行。在 API 级别 5.0 之前，虚拟机称为 Dalvik 虚拟机（DVM）；从 API 级别 5.0 开始，则称为 Android 运行时（ART）。ART 和 Dalvik 运行时都可以执行 Dalvik 可执行文件（Dex）字节码格式。ART 原生支持 APK 二进制文件中的多个 Dex 文件。ART 使用预先编译（AOT）技术，在应用安装时扫描一个或多个 Dex 文件，并将其编译成单个 .oat 文件，供安卓设备执行，从而提升性能 [30]。

应用的可用性：安卓设备上预装了一些应用，用户也可以从 Google Play

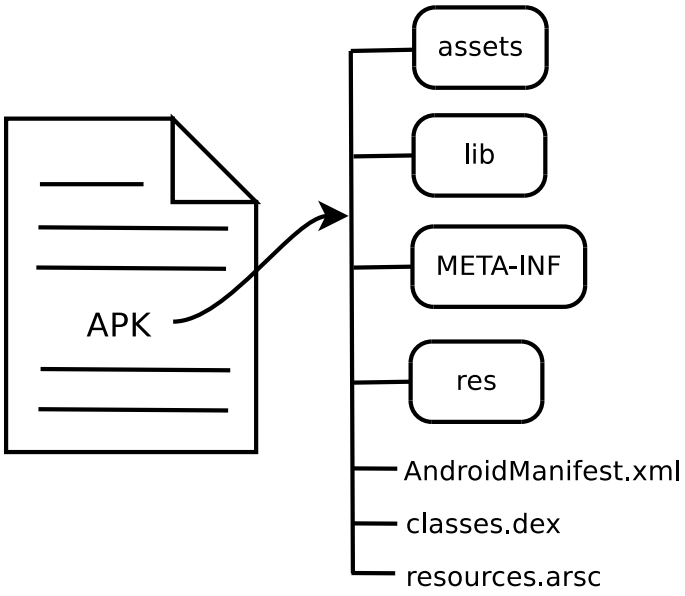

图2：安卓应用程序

商店安装应用。安卓的官方应用库称为 Google Play。它在 2008年 推出时，名称为 Android Market。安卓允许安装带有有效开发者签名证书的应用。还有其他应用市场，例如 Samsung Galaxy Apps、F-Droid、Amazon App Store、Huawei App Store 等。图 3 展示了 2015 年第一季度至 2021年 Google Play 商店中安卓应用的数量。可用应用数量首次出现下降是在 2018年6月，当时 Google Play 商店移除了近百万个不需要的应用 [31], [32], [33]。

一个安卓应用包含以下组件：

a) 活动：这些组件是终端用户与之交互的屏幕或用户界面（UI）组件。一个活动可以使用 Intent 与其他活动通信。b) 服务：这些应用组件在后台运行。它们类似于守护进程，为其他正在运行的进程提供服务。c) 广播意图和接收器：这些组件发送消息，供所有其他应用程序或特定的单个应用程序接收。d) 内容提供程序：这些组件是应用程序的数据存储，可在应用程序之间共享数据。

安卓新增了 Play Protect [33]，用于检测设备上已安装或即将安装的潜在有害应用（PHAs）。Google Play Protect 是一款实时恶意软件扫描器，可扫描设备上的所有应用程序，无论它们来自何处。2018年发布的 Android安全报告显示，仅使用 Google Play 下载应用的设备中，有 0.08% 受到 PHAs 的影响 [34]。考虑到设备安装基数，这仍可能意味着相当数量的设备。

C. 攻击面

本节将回顾 Android 应用程序的攻击面，以及恶意软件编写者如何利用这一点来规避基于权限的安全机制。在 Android 应用的背景下，减少攻击面通常意味着遵循最小权限原则。

安卓恶意软件或欺骗性应用程序利用的主要入口点如下：

•敏感资源：用户的个人信息和活动容易通过电话、短信、联系人、日历、位置和传感器泄露。

• 设备跟踪标识符：安卓应用开发者可以使用 IMEI、SIM用户ID、SIM卡序列号、硬件序列号等唯一设备标识符，近乎实时地跟踪用户或设备。利用这些标识符，设备上存储的用户信息可以与这些标识符关联起来，从而实现对每位用户的跟踪 [35]。很多时候，这是由于应用配置不当或应用权限过高所致；此类应用已获得读取手机状态权限——这是一项属于电话权限组的危险权限。

• 用户配置文件标识符：广告商和其他潜在的跟踪网站会使用 Android设备ID 和 Google 服务框架（GSF）ID，按应用和用户逐一窃取信息。如果未对这些 ID 进行适当的访问保护，就可能危及用户隐私。

•用户或设备位置：欺骗性应用可以通过网络或 WiFi 访问相关普通权限，或位置相关危险权限，获取用户或设备的位置数据。应用可以利用设备所连接的 WiFi 网络或蜂窝基站，跟踪设备和用户的精确位置坐标。因此，位置追踪功能应仅限于核心功能是提供位置详情的应用使用，例如 GPS 追踪器，并且须征得用户明确同意。

• Android 设备硬件：应用可以在用户不知情的情况下录制音频或视频，或拍摄图片。应用可以记录并传输与健康和用户活动相关的传感器数据。应用可以监控后台运行的其他进程，并可根据正在使用的同一设备协助追踪用户。

• 多个活跃的 Android API 级别：自 2014年 起，同时活跃的 Android API 级别一直有七个或更多 [3]。这扩大了 Android 应用的攻击面，因为尽管有新的操作系统版本发布以及改进的安全与隐私控制可用，应用却无法从中受益。设备制造商没有更快升级操作系统主要版本的动力，因此应用开发者别无选择，只能支持长期的向后兼容性。

D. 与研究相关的概念

我们介绍本研究中引用的重要概念，以及与安卓研究特别相关的概念。

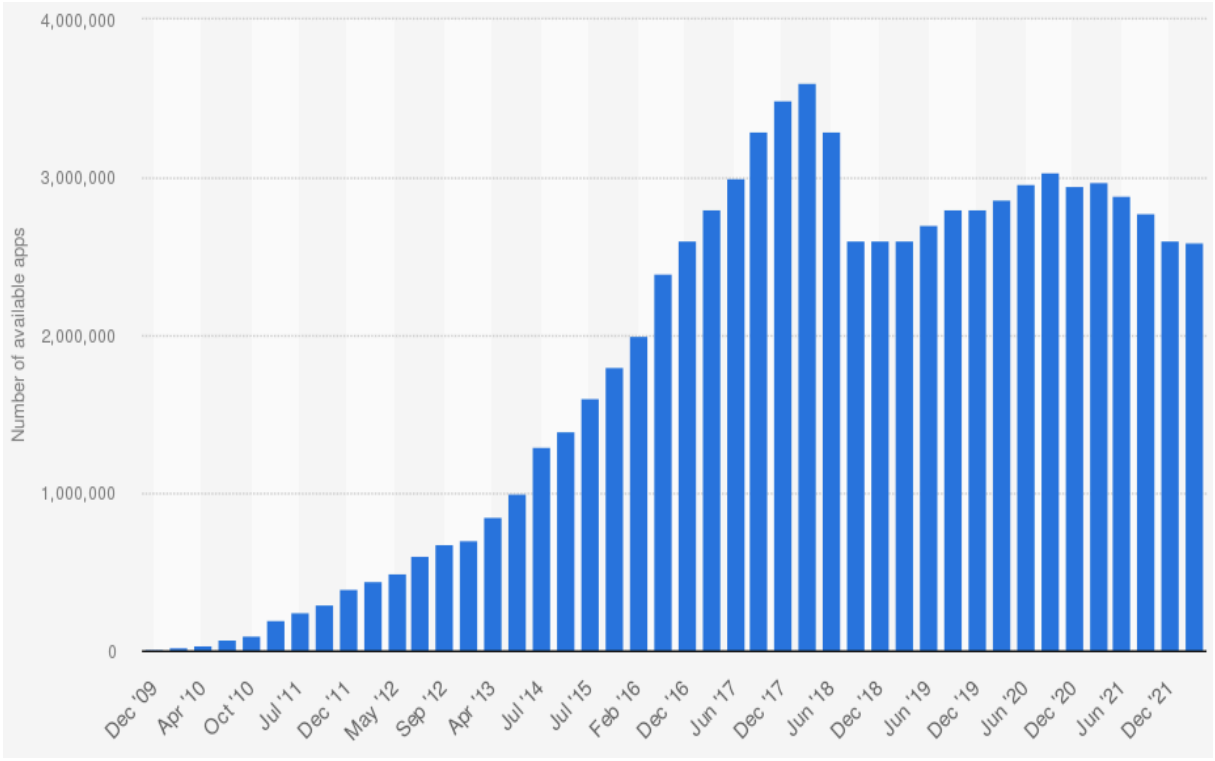

图3：2009—2022年Google Play 商店中的可用应用数量

```
1  <?xml version="1.0" encoding="utf-8" ?>
2  <manifest android:compileSdkVersion="29" android:compileSdkVersionCodename="10"
3      package="com.whatsapp">
4      <uses-feature android:name="android.hardware.camera" android:required="false"/>
5      <uses-feature android:name="android.hardware.telephony" android:required="false"/>
6  ...
7      <uses-permission android:name="android.permission.READ_PHONE_STATE"/>
8      <uses-permission android:name="android.permission.CAMERA"/>
9      <uses-permission android:name="android.permission.INTERNET"/>
10 ...
11     <application android:allowBackup="false" android:name="com.whatsapp.AppShell">
12         <activity android:name="com.whatsapp.Main">
13             <intent-filter android:label="@string/launcher_app_name">
14                 <action android:name="android.intent.action.MAIN"/>
15                 <category android:name="android.intent.category.LAUNCHER"/>
16             </intent-filter>
17         </activity>
18 ...
19         <service android:name="com.whatsapp.location.LocationSharingService"/>
20 ...
21         <receiver android:enabled="true" android:name="com.whatsapp.BootReceiver">
22             <intent-filter>
23                 <action android:name="android.intent.action.BOOT_COMPLETED"/>
24             </intent-filter>
25         </receiver>
26 ...
27         <provider android:authorities="com.whatsapp.provider.contact"
28                   android:name="com.whatsapp.contact.ContactProvider"/>
29     </application>
30 </manifest>
```

Permissions

Activity

Service

Broadcast Receiver

Content Provider

图4：显示活动、服务、广播接收器和内容提供者的应用清单 er

1）最小权限原则（*PoLP*）：每个安卓应用及其引用的库程序都应仅使用完成应用功能所必需的最少权限。该原则凸显了因权限提升（EoP）事故或错误可能造成的损害。最小权限原则强调，将特权二进制程序之间可能发生的交互数量减至满足预期功能所需的最低限度，从而降低在任何情况下意外、非预期或不当使用权限的可能性。因此，如果需要查明权限滥用，必须检查的程序及其交互数量就会降至最低 [36]。本研究中使用的“权限过度”一词，指的是违反最小权限原则。

2）情境完整性：Nissenbaum 的“作为情境完整性的隐私”理论强调，信息的收集与传播（或流动）应遵循信息流的情境规范 [37]。此外，违反适当性情境规范或信息流规范，都会造成隐私侵犯。通过绕过基于权限的模型（例如自动授予权限、跟踪持久标识符等），应用可以窃取个人数据并将其传播到设备之外，这违背了用户的预期。如果应用未明确请求并获得访问用户个人信息的权限，那么将用户数据与该应用或任何第三方关联起来，就会违反用户数据的情境完整性，进而侵犯隐私 [38]。

3）告知与同意框架（*NCF*）：“告知与同意”框架规范隐私实践：组织可以告知用户其隐私政策和实践，用户则可以明确同意这些实践。在网站上接受 Cookie 以及在安卓中授予权限，都是这一框架的简单示例。在实践中，当欺骗性应用未向毫无戒心的用户透明说明其完整行为规范时，其行为可能导致侵犯用户的数据隐私。

4）安全、隐私与保护：“安全”“隐私”和“保护”这些术语常用于安卓和信息系统领域。值得注意的是，并非所有作者对这些术语的用法都相同。本研究采用计算机科学和 Android 研究文献中常见的定义。“安全”一词指用于控制谁可以使用或修改计算机或设备，或其中所含信息的技术。

“隐私”一词指社会所界定的个人（或组织）决定是否、何时以及向谁披露个人（或组织）信息的能力。

“保护”一词在此特指用于控制正在执行的程序访问设备上存储信息的机制和技术 [36]。

5) 潜在有害应用（*PHA*）：潜在有害应用是可能危及用户数据或设备的应用。常见的潜在有害应用包括木马、间谍软件、勒索软件、广告软件、钓鱼应用、点击欺诈应用等。 [39]由于安卓生态系统碎片化，潜在有害应用也可能作为预装应用出现在设备上。此外，它们还可能通过捆绑在某些正版系统更新中的无线更新传播，或通过某些正版系统更新中的无线更新传播。有时，用户可能会安装潜在有害应用，这些应用会更改默认设置、禁用安全功能、获取设备的 root 权限，或削弱现行安全策略。一项研究表明，潜在有害应用在 Google Play 上平均可存留 77 天，在第三方应用商店中平均可存留 34 天 [40]。安卓系统应清晰提示相关风险，并提醒用户，以便他们做出明智的决定。

6) 不需要的软件（*UwS*）： 不需要的软件应用程序是尚未被归类为恶意软件、但会危害应用生态系统的应用。Google 将移动不需要的软件（MUwS）定义为冒充其他应用的应用，或未经用户同意收集以下至少一项信息的应用：设备电话号码、设备上的主电子邮件地址、查询其他已安装的应用，以及与第三方账户关联的信息 [39]。这些欺骗性应用权限过度，会对用户的数据隐私构成风险。Swajcer 等人 [41] 针对移动应用情境下的潜在不需要的应用程序（PUA）提出了一套分类体系。PUA 包括但不限于间谍软件、广告软件、黑客工具、声誉较差的应用等。

7) 恶意软件：：恶意软件是一种带有恶意意图的软件，会在用户不知情的情况下执行有害操作 [42]。此类软件可以窃取数据，甚至取得其运行设备的特权控制权。恶意软件编写者通常会使用隐蔽技术，逃避现有保护方法的检测 [26]。

8) 安卓 *API*、源代码与汇 *API*：每次安卓操作系统发布重大版本时，都会分配一个 API 级别，用于表示新版本中可用的一组 API 调用的更新。安卓 API 调用可以是源 API，也可以是汇 API。

安卓源 API 定义为调用资源方法，并将变量值返回到应用中，例如 getDeviceId()。它返回国际移动设备识别码（IMEI）号码。IMEI 号码在每部移动电话上都是唯一的。安卓汇 API 定义为调用资源方法，这些方法至少接受应用提供的一个变量值作为参数之一；如果在资源上写入新值或覆盖现有值，也属于此类调用。例如，对于用户消息传递，调用 sendTextMessage()。它接收消息文本正文和电话号码作为参数，两者都是变量值。

E. 分析机制

研究人员主要使用两种机制来理解和分析应用行为——静态分析和动态分析。静态分析通过读取和检查应用程序来进行研究——无论是人类可读的源代码，还是某种机器可读的二进制代码。动态分析则通过执行应用程序来进行研究——无论是执行测试用例，还是在受控环境中运行应用程序。

1) 静态分析： 静态分析会遍历代码中的所有执行路径，以分析应用的所有可能行为。借助这种分析，可以查看代码，寻找可能表明假设行为的证据，而这些行为可能导致恶意行为、访问控制违规或隐私泄露。静态分析无法生成轨迹、运行时

上下文，也无法提供此类问题实际发生的实例。不过，如果现有代码被执行，它可以提示此类问题可能会出现。静态检测工具使用反编译或逆向工程技术来分析代码及相关的数据流或控制流 [26], [43]。

静态分析可以借助自动化，对数千甚至数十万应用程序进行相当大规模的分析。应用开发者可以通过代码转换、代码混淆、动态代码加载等技术，规避静态分析对行为的检测。在这些情况下，静态分析可能不足以应对。

2）动态分析：动态分析通过实际执行应用程序代码，进而分析和审查其运行时行为。动态分析面临测试覆盖率问题——可以执行列举出的测试用例，但要执行所有可能的路径并不现实。使用动态分析技术大规模分析大量应用程序，既不简单，也绝非易事。

3）混合分析：可以结合静态和动态分析技术来分析应用程序行为。这种混合分析有助于克服静态和动态方法的两大主要局限——应用程序代码覆盖率以及分析的整体可扩展性。可以使用静态分析技术检测所有相关的可能执行路径，再利用动态分析技术筛除这些组合，并找出更有可能出现问题的场景以及问题仍然存在的实际情况。或者，也可以先通过动态分析找出潜在用例，再利用静态分析将其推广，以便进行大规模分析。

III. API 调用与权限之间的映射

本节介绍 2011—2020 年间生成或提取权限规范的开创性研究。表 II 展示了有效的对比，突显本研究在填补基于Android权限的分析领域文献空白方面所作的贡献。

A. 个别研究

Felt 等人 [16] 开发了一款名为 Stowaway 的工具，通过研究 940 多个应用来检测权限过度，并发现其中三分之一存在权限过度。Felt 等人将面向 Java 程序的面向对象测试生成器 Randoop [47] 用于 Android 2.2，通过自动化测试找出应用使用的 API 调用，并进一步列出调用每个安卓 API 所需的权限。由于采用基于测试的动态分析方法，Stowaway 存在代码覆盖率问题。由于无法处理反射调用，作者将 Java反射列为尚待解决的问题。Felt 等人还描述了权限强制执行机制，以及应用中出现非必要权限的一些原因。这项工作需要修改 Android源代码，并在应用分析期间进行一些人工干预。Felt 等人的工作没有提供源代码和权限映射，也无法扩展到较新的安卓版本。

Au 等人 [17] 提出了工具 PScout，通过对 Android 源代码进行静态分析来提取权限映射。PScout 使用 Soot 框架 [48]进行 Java 字节码分析。Au 等人基于类层次分析（CHA）生成控制流图（CFG），并对控制流图进行反向可达性遍历，以近似确定会执行权限检查的 API 调用，或不会执行权限检查的 API 调用。研究发现，由于 API 调用字符串的静态引用，PScout 生成的权限到调用的映射数量更多。

Bartel 等人 [44] 使用静态分析技术提取权限映射。他们借助 Soot [48]和 Spark [49] 框架，针对 Android 4.0.1 基于类层次分析生成控制流图。尽管他们在研究结果中提到，分析取得了令人鼓舞的结果，但他们既未提供源代码，也未指出任何可供验证的实验或实证数据。Bartel 等人生成的权限映射现已无法获取。

Backes 等人 [18] 利用 Axplorer 首次对安卓框架的受保护资源进行了高层分类。他们提出了一种新颖的映射，相比缺乏框架内部信息的 PScout 结果有了显著改进。作者提供了三种不同的映射，分别针对 API 级别 16（Android v4.1）至 25（Android v7.1）的软件开发工具包、应用框架和内容提供程序。Backes 等人运用权限局部性的概念说明，框架内部用于保护敏感操作的权限检查并未遵循职责分离原则。

Bogdanas 等人 [45] 提出了 DPSpec，该方案通过多种文档格式生成权限映射，包括 XML 文件中的注解以及 Android SDK 的 Java 文档注释。他们声称其准确率优于 Axplorer，但既未提供 DPSpec 方案的源代码，也未提供权限映射以供比较。

Lyvas 等人 [19] 提出了 Dypermin，通过运行时的安全异常、受保护的 API 方法是否可用，以及 Java 反射机制来编译 Android 权限规范。Dypermin 依赖于在调用 API 方法时因缺少所需权限而引发的运行时异常，并且不会生成误报映射条目。

Yousra 等人 [46] 提出了 ARCADE，通过考虑调用条件 API 时所需满足的用户检查，来展示条件权限规范。作者将路径敏感的控制流图纳入考量，以推导出包含用户检查（例如 UID）的精确权限映射。此映射无法直接用于其他分析，而且作者没有提供源代码。

B. 权限映射的挑战

近年来，将权限映射到 API 调用的相关研究进展停滞不前。造成这一局面的因素有很多。随着功能不断增加，安卓源代码库的规模成倍增长，权限数量也随之增加。除 Java 外，Kotlin 也逐渐应用于安卓源代码树，使框架源代码的分析更加复杂。Google 和 Android 开源项目（AOSP）既未提供应用的执行规范，也未提供可大规模分析框架源代码的工具，而相关文档往往并不完整。

表II：API 与权限之间的映射——贡献与比较

相关研究	Year	所提出的解决方案				
		Android 版本	API 级别	源代码与数据集	扩展	贡献
Stowaway, Felt 等人 [16]	2011	2.2	8	没有源代码且没有可用的数据集	无法扩展	基于方法调用的动态分析（API 模糊测试）和权限入口点监控，提取的敏感 API 列表
PScout, Au 等人 [17]	2012	2.2.3-5.1	10-22	源代码与数据集可用	No	基于控制流图的静态分析和可达性分析和类层次分析提供权限与 API 映射
Bartel 等人 [44]	2014	4.0.1	14	源代码与数据集不可用	No	基于类层次分析的静态分析（CHA）和字段敏感分析（Spark）
Axplorer Backes 等人 [18]	2016	4.1.1、4.2.2, 4.4.4, 5.1	16, 17, 19, 22	源代码不可用，数据集可用	No	使用过程间调用图和以下内容中的权限局部性权限检查
DPSpec Bogdanas 等人 [45]	2017			源代码与数据集不可用	No	基于安卓的静态分析 软件开发工具包源代码和代码，Javadoc 注解
Dypermin Lyvas 等人 [19]	2018	4.1.1，6.0	16, 23	源代码可用，数据集不可用	No	使用安卓进行动态分析 软件开发工具包源代码
ARCADE Yousra 等人 [46]	2018	7.1	25	源代码不可用，数据集可用	No	使用路径敏感静态分析提取的条件映射，包含 UID 检查和用户检查

表 III：权限规范：评估的出版物完整列表

Year	出版物类型	出版物名称	出版物标题	参考文献
2011	ACM	CCS	Android 权限揭秘	[16]
2012	ACM	CCS	PScout：分析 Android权限规范	[17]
2014	IEEE	TSE	用于提取大型框架权限检查的静态分析框架：分析安卓的挑战与解决方案	[44]
2016	USENIX	USENIX 安全会议	揭秘 Android 应用程序框架：重新审视 Android 权限规范分析	[18]
2017	arXiv	arXiv	Dperm：协助 Android 应用迁移至运行时权限	[45]
2018	ScienceDirect	计算机与安全	Dypermin：安卓平台的动态权限挖掘框架	[19]
2018	ACM	CCS	精确的安卓 API 保护映射推导与推理	[46]

IV. 权限模型的演变

安卓安全模型是一种多方同意模型，要求只有在所有相关方（设备用户、应用开发者、平台或服务提供商）都同意的情况下，某项操作才会执行 [22]。如果任何一方不同意，默认情况下，该操作应被阻止。

权限是嵌入安卓的安全与隐私机制的重要组成部分。一般而言，权限有两个用途：执行保护策略，以及描述应用的功能。

A. 权限请求与授予

随着安卓框架的演变，应用请求权限的方式以及权限的授予方式也经历了数次变化。以下概述请求和授予权限的一般方法；随后我们将介绍相关的演变变化：

- 安卓通过为每个应用分配唯一的用户ID（UID），使每个应用都以不同的系统身份运行。
- 每个应用都在进程沙箱中运行，因此与设备上的其他应用程序相互隔离。
- 因此，应用必须明确请求访问其自身沙箱之外的资源和数据。

- 应用通过声明所需的权限来请求此类访问。

- 可以在安装应用时请求权限，也可以在运行应用时（运行时）稍后请求。
- 系统可以自动授予权限，也可以提示用户批准或拒绝权限请求。

- 权限在应用清单文件中声明。在应用安装或升级期间，这些权限会以单页形式呈现给用户；也可能在使用特定功能时，以多个屏幕对话框的形式呈现，例如使用相机或麦克风。
- 某些权限无需通过界面向用户授予。这些是安卓定义的普通权限，并隐藏在折叠的显示区域中。

- 在向用户告知权限方面，有观察发现，权限详情隐藏得很深，用户需要在安装选项所在位置点击多次才能找到。即使是知情且有经验的用户，也会觉得在安装应用前查找此类信息十分繁琐。

权限分为安装时权限、运行时权限和特殊权限 [50]。安装时权限包括普通权限和签名权限。普通权限在应用安装时授予。如果应用的签名与定义这些权限的应用的签名相匹配，则会在安装时向该应用授予签名权限。运行时权限包括一些保护级别为 危险 的权限。安卓平台和设备制造商（OEM）会使用一些特殊权限，并通过*appop*保护级别实现其细节。

从 Android v6 API 级别 23 开始，在运行时请求权限时，会显示一个或多个权限对话框。图 5 展示了一个权限对话框示例，其中包含

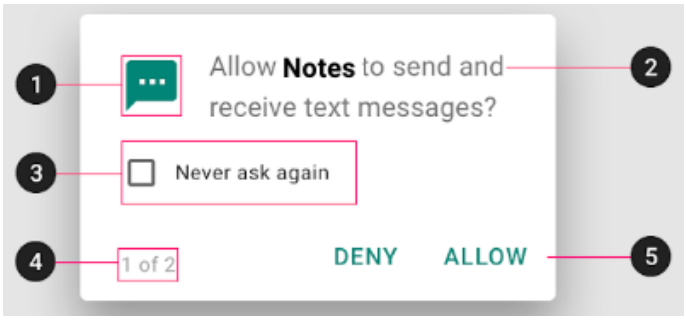

图 5：运行时请求权限对话框

在运行时请求权限时显示的标签。以下是这些标签的摘要。

1) 权限组图标，例如短信。2) 应用名称。3) 用户两次拒绝权限后，会显示“不再询问”复选框。4) 多个对话框的计数指示器。5) 面向用户的请求操作——允许或拒绝。

B. 演进变化

Enck 等人 [51] 最早研究了安卓安全模型，并在安卓发展的初期揭示了其复杂性。Enck 等人 [56] 提出了 Kirin，利用基于模板的安全规则检测应用中实现的危险功能。在另一项工作中，Enck 等人 [52] 介绍了 Ded 编译器，用于从安卓应用程序中还原源代码。他们揭示了广告与分析网络使用手机标识符的情况。Yuri 等人研究了 Android API 级别 23 及更早版本中与权限相关的变化 [54]。Jeon 等人 [57]提出使用 RefineDroid 检测所需的细粒度权限，并在修改后的应用二进制文件中强制实施这些权限。此外，我们还介绍了安卓中引入的一些重大变化。

1）在安装时授予所有权限：在安卓 API 级别 23（Android 6.0 Marshmallow）之前，用户想要安装应用时，安装程序通常会显示所有与权限相关的文本。安装程序若要继续，用户必须同意并授予应用所需的所有权限。用户无法自行选择在安装时不授予某些特定权限。此外，系统也没有提供撤销已授予权限的机制。这种非此即彼的强制做法可能会诱使用户在不了解所请求权限详情的情况下，轻率地授予所有权限。在 API 级别 22 及之前，应用清单中请求的所有权限都会被授予。

2) 运行时权限： 从 Android 6.0 API 级别 23 开始，权限根据首次请求用户同意的时间，分为安装时权限（应用正常运行所必需）和运行时权限 [58]。用户安装应用时，普通权限和签名权限（少数例外情况除外）会被永久授予。危险权限现在会在运行时授予，用户可以随时撤销这些权限。在安装时权限备受批评之后，这

表IV：权限演变：评估的出版物完整列表

Year	出版物类型	出版物名称	出版物标题	参考文献
2009	IEEE	S&P	理解Android安全	[51]
2011	USENIX	USENIX 安全会议	Android应用安全研究	[52]
2012	ACM	ACSAC	Android 生态系统中的权限演变	[53]
2016	Springer	RAID	小变化，大变化：Android 权限系统的新视角	[54]
			Android 权限系统	
2020	ACM	MSR	安卓应用中的自动授予权限：一项关于其普遍性及其对隐私的潜在威胁的实证	[55]
			关于其普遍程度及其对隐私的潜在威胁的研究	

是 Android 6.0 带来的一项重大变化，它允许用户在组级别选择性地撤销权限。通常情况下，特定权限会默认授予应用；对于某些权限，系统会提示用户允许或拒绝该权限。不过，用户可以通过应用配置设置撤销一项或多项权限。API 级别 23 的一项重大变化是，应用清单中请求的权限并非都会在安装时授予。这一变化源于人们意识到，并非所有用户都具备在安装时做出此类决定的足够能力 [59]。

3) SELinux：安卓使用安全增强型 Linux（SELinux）对所有进程实施强制访问控制（MAC）。这包括具有 root 或超级用户权限的进程。SELinux 遵循默认拒绝原则，即访问控制策略未明确允许的任何内容都将被视为拒绝。

4）存储权限：在 API 级别 18 及之前，所有安卓应用在应用安装时都可以读取存储在外部 SD 卡存储中的内容。从 API 级别 19 开始，系统强制实施两项存储权限——读取外部存储和写入外部存储。此外，读取或写入外部存储上的应用专属目录，无需这些权限。进一步来说，任何声明了写入外部存储权限的应用，都会被隐式授予读取外部存储权限 [60]。

5）*MAC* 地址随机化：安卓在 v8 中加入了 MAC 地址随机化0 [61]，以防止监听者和流量嗅探器创建基于设备 MAC 地址的画像。这项变更旨在限制应用的跟踪功能并提升用户隐私 [62]。然而，用户跟踪网络和定向广告机构会使用其他标识符（国际移动用户识别码、广告 ID 或 ANDROID ID、Google 服务框架（GSF）ID 等），在很大程度上或以抽象的组级别唯一识别用户。不过，在安卓 10 API 级别 29 中，针对企业设备使用，设备所有者应用可以通过 getWifiMacAddress() API 获取 MAC 地址 [63]，，从而允许远程进行企业设备管理。

6) 仅在使用应用时授予的情境依赖权限：2019年，安卓 10 API 级别 29 新增了一种非二元的情境依赖权限设置，用于访问设备的位置数据。除了始终允许 和拒绝之外，还新增了第三种设置，可将位置访问权限设为仅在使用应用时允许。此举旨在提升位置透明度，并在不需要时赋予用户更多限制位置数据使用的控制权 [64]。有趣的是，第三种设置仅在应用于前台运行时授予位置权限，从而在应用于后台运行时悄然限制其访问位置数据。从安卓 11 API级别30 开始，相机和麦克风权限也适用这一第三种设置，即仅在使用应用时允许。此举可阻止应用在用户未主动知情的情况下于后台访问相机或麦克风 [22]。Nauman 等人 [65]此前提出过一种扩展方案，允许有条件地授予权限，而非始终授予。

7）一次性权限：从安卓 10（API 级别 29）开始，当应用请求位置、麦克风或相机权限时，用户可以选择“仅限这一次”。此选项会授予应用临时一次性权限，从而限制应用每次启动时访问敏感数据的能力。

8）自动重置权限：在安卓 11 上，如果用户几个月没有使用某个应用，系统会自动重置敏感权限。不过，这项变更生效前的时长尚不明确。据观察，在应用设置中查看这些权限时，部分已重置的权限会重新授予应用。

9）分区存储：安卓 API 级别 29 引入了分区存储的概念，并从 API 级别 30 起强制实施。此前，应用可以访问整个外部存储空间，而不只是特定文件和文件夹。启用分区存储后，应用只能读取或写入特定文件和文件夹，而非整个外部存储空间。分区存储显著改善了设备上用户内容的隐私保护。

C. 更改权限组

权限组也随着权限一同演变。危险权限被划分为不同的组，以便访问敏感资源。在安卓 8 API 级别 27 之前，共有九个权限组：日历、相机、联系人、位置、麦克风、电话、传感器、短信和存储。在安卓 9 API 级别 28 中，新增了通话记录，以便受限应用读取和编辑通话记录。此外，在安卓 10 API 级别 29 中，新增了活动识别。将针对特定资源的写入或编辑权限与读取权限捆绑在一起

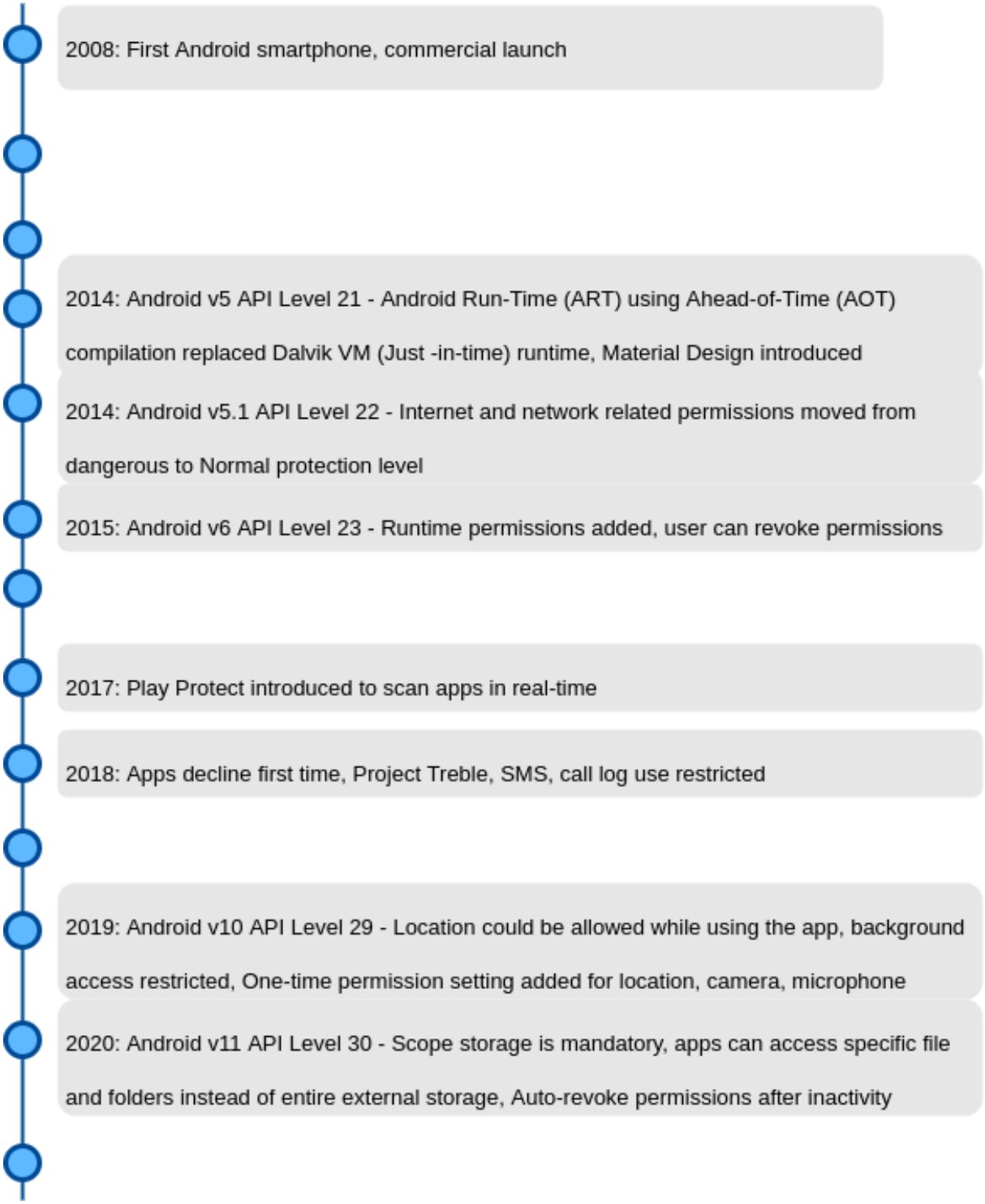

图6：过去十年安卓的发展时间线。

会损害授予权限的粒度。

V. 权限检查

本节将综述权限的检查方式，并对权限进行分类。

A. 权限检查

权限检查的目的是确定应用是否拥有执行 API 调用或访问特权资源的权限。权限检查可以在两个位置进行：(a) 应用使用安卓 API 时，在应用二进制文件中进行；(b) 在实现实际 API 的系统进程中进行。我们观察到，权限检查分布在 Java

表V：权限检查：评估的出版物完整列表

Year	出版物类型	出版物名称	出版物标题	参考文献
2011	ACM	CCS	Android 权限揭秘	[16]
2014	IEEE	TSE	用于提取大型框架权限检查的静态分析框架：分析安卓所面临的挑战与解决方案	[44]
2022	IEEE	TSE	安卓应用中的运行时权限问题：分类体系、实践，以及未来方向	[20]
2022	ACM	ICSE	APER：演化感知的运行时权限滥用检测	[66]

API 层和 Native C/C++ 库中，还有少数权限在 Linux 内核层强制执行。

安卓应用应在应用清单文件（AndroidManifest.xml）中使用 ¡uses-permission¿ XML 标记声明权限。从 Android API 级别 23 起，普通权限会自动授予，无需向用户显示任何提示。为避免运行时权限（ARP）问题，必须明确请求危险权限 [20]。

作为最佳实践，在调用受权限保护的 API 之前，应用应调用 checkSelfPermission() API 检查是否已获授相应权限 [50]。应用应谨慎地在运行时调用 requestPermissions() API 请求危险权限。系统会通过弹出对话框提示用户授予或拒绝权限，甚至可以阻止后续请求（“不再询问”选项）。应用可以通过 onRequestPermissionsResult() API 保存权限授予结果，并向用户作出响应[50]。图 7 展示了一段代码片段，用以说明这些主要的权限管理 API。

```java
if (ContextCompat.checkSelfPermission(CONTEXT,
      Manifest.permission.REQUESTED_PERMISSION) ==
      PackageManager.PERMISSION_GRANTED) {
      // Permission <REQUESTED_PERMISSION> is granted,
      // now you can use the permission-protected API
      doSomething(...);
} else if (shouldShowRequestPermissionRationale(...)) {
    // If user denied permission earlier or
    // selected "Never ask again", then
    // explain to the user why the app
    // requires this permission through dialog
    showUI(...);
} else {
    // Directly ask for (multiple) permissions
    requestPermissions(CONTEXT,
      new String[]
      { Manifest.permission.REQUESTED_PERMISSION },
      REQUEST_CODE);
}

@Override
public void onRequestPermissionsResult(int requestCode,
    String[] permissions, int[] grantResults) {
    // After user responds to permission request and
    // to store result of granted permissions
}

// REQUESTED_PERMISSION can be GET_ACCOUNTS, CAMERA,
// ACCESS_COARSE_LOCATION, ACCESS_FINE_LOCATION, or
// any valid permission string from Manifest.permission
```

图 7：用于检查、请求和说明权限的示例代码

安卓操作系统最初被设想为封闭式智能手机操作系统平台的开放替代方案。然而，仔细审视后我们发现，安卓平台也存在封闭的部分。Google 的安卓代码库中有些部分是封闭的（例如设备驱动程序）。用户无法访问安卓的一些核心部分，例如精简内核和系统守护进程；软件和策略机制限制了用户的访问。控制权掌握在 Google、移动运营商和智能手机制造商（原始设备制造商）手中。由于安卓生态系统中存在这种限制，与权限检查相关的具体实现细节并未广泛公开。

B. 权限特征分析

互联网及其他网络相关权限在 API 级别 22 [54]被调整为普通保护级别。虽然将互联网权限调整为普通保护级别是否有益尚有争议，但显而易见，应用不会明确请求用户注意其访问互联网的行为。应用所使用的任何第三方库也是如此。

许多用户和开发者认为读取短信权限具有侵入性，因为应用一旦获得此权限，就能读取设备上的所有短信。Play 商店在 Project Strobe[67] 中开始要求提供长达六页的文档，说明为何需要此权限后，该权限便受到限制。因此，能够访问短信和通话记录权限在一年内出乎意料地减少了 98% [68]。

位置和后台位置：应用可以请求两项与位置相关的权限——ACCESSCOURSE LOCATION 和 ACCESS FINE LOCATION。

Android 6.0（Marshmallow）新增了一项特殊权限 SYSTEM ALERT WINDOW，用于创建覆盖在所有其他应用屏幕之上的悬浮窗口。少数系统级应用应使用此权限与用户互动。API 级别 23 或更高的应用必须明确请求

此权限，并通过发送包含操作 ACTION MANAGE OVERLAY PERMISSION 的显式 intent，征求用户同意。应用应调用 android.provider.Settings.canDrawOverlays() [60] 检查权限是否已获授权。Android 6.0 的早期版本向所有应用授予了这项危险权限。Fratantonio等人展示了多种用户界面攻击，仅使用 SYSTEM ALERT WINDOW 和 BIND ACCESSIBILITY SERVICE 这两项权限，以 Cloak and Dagger 的方式实施攻击。

另一项特殊权限 WRITESETTINGS 允许应用读取或写入系统设置。API 级别 23 或更高版本的应用必须通过发送包含操作 ACTION MANAGE WRITE SETTINGS 的 intent，明确请求此权限以征得用户同意。应用应通过调用 android.provider.Settings.System.canWrite() [60] 来获取权限授权。

C. 第三方库的使用

安卓生态系统中的免费应用依靠广告网络、跟踪器和数据经纪商提供支持并实现变现。这些第三方组织提供软件库，以集成广告投放并在未经用户同意的情况下跟踪用户 [35]。值得注意的是，广告和分析库与应用代码以相同的访问权限运行，因此能够访问与应用本身相同的权限。因此，这些第三方库可以与数据经纪商共享设备标识符和用户特定的敏感信息，以投放广告。这对终端用户存储和提交的数据构成安全与隐私风险。图 8 展示了一个包含 Firebase 库引用的应用清单示例 [69]。

AppCensus 项目帮助人们了解免费应用的隐私代价。 [70]。一项针对一百万个网站的研究 [71] 旨在进行网络隐私测量，研究发现，每个第三方服务提供商都可能是跟踪器。随着移动优先趋势的发展，应用也能以类似方式跟踪用户活动，甚至触及更广泛的范围。

D. 权限差距的成因

从本文提及的研究可以明显看出，应用声明的权限多于其所需权限。本节总结了对权限差距的因果分析；这一差距源于权限过度的安卓应用二进制文件：

a) 同时或成对请求网络权限：常见的情况是，应用会同时请求互联网、无线局域网和网络访问相关权限，即使它们根本没有使用这些权限。这是因为人们误以为每个应用始终都需要互联网和网络相关权限。应用可能只需要其中一项权限，但应用开发者却把所有权限都加上了。在那些不必要地请求无线局域网或网络权限的应用程序中，实际上确实需要网络权限或无线局域网权限的应用所占比例分别要低得多 [16]。

b) 文档质量不佳或不完整：应用开发者通常会参考安卓开发者参考手册 [1] 获取与 API 相关的文档。众所周知，API文档并不完整，也不是最新的，其中还存在隐藏 API [75]。这也是导致经验不足的开发者可能找不到相关信息，最终使用了本可以在一开始获得更好帮助后避免使用的权限。

c) 使用已弃用的权限：在安卓应用清单文件中，开发者会指定其目标 API 级别的最低值和最高值。后续应用更新可以更改清单头部中的这些信息。在更新应用时，开发者可能没有注意到移除已弃用权限，因为安卓 API 级别会引入新的行为变更。由于应用开发工具或集成开发环境不会强制移除这些权限，已弃用权限也会出现在已发布应用中。用户甚至可能不会获知休眠权限的使用情况，而这些权限会在任何一次升级时被调用 [76]。

d) 活跃安卓 API 级别的跨度：截至2021年初，安卓 API 级别30（版本11）是最新的稳定版本。与此同时，仍有相当一部分活跃的 Android 设备使用 API 级别 19-23（版本4.4-6.0），这些版本早在 2013 [3] 就已发布。值得注意的是，2013年-2015年间，活跃安卓设备中并存着五到六个 API 级别，这一数量在2018年-2020年间翻了一番。活跃 API 级别跨度如此之大，迫使应用开发者为同一应用二进制文件纳入范围广泛的最低和最高 API 级别，以支持向后兼容性，导致较新版本的应用质量欠佳或配置错误。

e) 纠正权限过度声明的成本：开发者需要在越来越短的时间内发布功能越来越丰富的应用。他们可能会由于上述原因而过度声明权限。纠正过度声明的成本很高，即开发者需要调查各项权限的具体用途并更新清单。如果安卓 SDK 框架或开发工具能够就这种过度声明向开发者发出警告，他们就能以较低的成本加以修正 [77]。

VI. 潜在的未来方向

我们回顾了过去 10 至 12 年（2010—2022年）间以权限为重点的 Android 研究中的一些开创性工作。基于这项综述，我们总结出若干见解，列举如下。

A. 近期安卓版本的权限映射

本研究建议改进近 5 年发布的安卓版本中 API 调用与权限之间映射的可用性，具体而言，是安卓 8 版本或 API 级别 26。由于安卓框架源代码复杂，且缺少有关安卓内部框架细节的研究，这方面的研究工作一直停滞不前。这些权限映射将有助于开展有关权限过多或配置错误的应用、欺骗性应用以及恶意软件检测的研究。

[1]https://developer.android.com/docs

```
<manifest android:compileSdkVersion="29" android:compileSdkVersionCodename="10"
    package="com.whatsapp">
...
    <application ...
...
        <receiver android:name="com.whatsapp.phoneid.PhoneIdRequestReceiver">
            <intent-filter>
                <action android:name="com.facebook.GET_PHONE_ID"/>
            </intent-filter>
        </receiver>

        <service android:exported="true"
                 android:name="com.google.firebase.iid.FirebaseInstanceIdService">
            <intent-filter>
                <action android:name="com.google.firebase.INSTANCE_ID_EVENT"/>
            </intent-filter>
        </service>

    </application>
</manifest>
```

Read Business Phone numbers

Analytics, Personalised Advertising, Authentication

图8：应用清单展示了用于分析和广告的第三方库

表 VI：权限过度授权：评估的出版物完整列表

Year	出版物类型	出版物名称	出版物标题	参考文献
2011	IEEE	W2SP	遏制安卓权限蔓延	[72]
2011	ACM	CCS	Android 权限揭秘	[16]
2013	USENIX	USENIX 安全	WHYPER：迈向自动化风险评估移动应用	[73]
2014	ACM	CCS	AutoCog：衡量描述与权限的一致性在 Android 应用中	[74]
2021	ScienceDirect	计算机与安全	MapperDroid：验证应用描述中的应用能力权限和 API 调用	[38]

B. 评估数据集的可用性

在所分析的研究中，使用较旧数据集的情况占主导地位，无论是恶意软件应用，还是与应用元数据相关的数据集 [78]。这迫使新研究的作者使用这些较旧的数据集进行比较，因此也引发了人们对其在 2018年-2022年这一时期相关性的质疑。我们观察到，表 II 中提到的研究均无法扩展并在较新的 Android 版本上复现。建议新研究提供充分的材料以复现研究成果，并提供数据集来评估各种方法和进行公平比较。

C. 第三方库相关权限的分离

如果应用与第三方库的配置文件之间存在冲突，可能会引发安卓运行时权限问题。这些第三方库可能会使用受权限保护的 API，却没有正确检查并仅请求所需的权限 [20]。目前，应用开发者和用户并不会获知第三方库具体需要哪些权限。使用 Unity 框架 v5.1 构建的应用会为每个应用添加互联网访问和读取手机状态的权限，尽管开发者并无此意。糖尿病应用Glucosio被发现因使用第三方库 InstaBug 而具有录音权限 [79]。如果向应用开发者和终端用户分别提供此类权限使用情况的信息，就能及早发现配置错误或具有欺骗性的应用。这适用于应用的全新安装以及后续的任何更新。

D. 第三方应用的自定义权限

Tuncay 等人 [80]对自定义权限提出了担忧。作者指出，用于规范对应用组件访问的机制可能会遭到滥用，从而获得未经授权的访问。

E. 活跃的安卓 *API* 级别的最大数量

正如我们在 V-D 节中所指出的，同一时间存在大量活跃的安卓 API 级别。如果将活跃 API 级别的最大数量限制为四个或更少，就会促使设备制造商（OEM）比实际情况更快地升级安卓主要版本。这反过来又会促使应用开发者紧跟后续操作系统版本发布中的各种改进，并加以利用。这项举措有望缩小安卓攻击面，因为设备软件的升级速度会比截至 2021 年初的实际情况更快。

F. 授予权限的粒度

从 API 级别 23 引入运行时权限以来，直至 API 级别 30，危险权限都是在权限组级别授予的。因此，应用会不必要地获得同一权限组中其他权限的访问权。例如，拥有读取手机状态权限的应用也可以访问拨打电话权限以及电话组中的其他权限。Paolo 等人在最近的一项研究中指出，[55]尽管应用描述未说明用途，仍有 56% 的应用使用了自动授予的权限。我们强烈建议将权限授予粒度细化至当前权限组级别以下，以避免滥用自动授予的权限。

G. 不再局限于一次分析一个应用

在已发表的研究中，我们经常发现静态或动态分析技术一次只分析一个应用，有时还会忽略设备上由同一开发者或其他开发者开发的其他应用。单个应用可能不会独自泄露敏感信息，但可能会借助设备上一个或多个其他应用获得的权限来泄露信息。

H. 请求权限的标准用户界面

需要为请求权限的用户界面制定一套基准标准。该标准界面应说明请求某项权限或一组权限的原因。应用应明确说明将收集哪些信息，以及如何使用所收集的信息。应用不应提前请求未来更新中可能需要的不必要权限。相反，未来应在引入新功能时提示用户授予相应权限 [81]。

VII. 结论

十多年来，多项研究报告了安卓权限模型在实现过程中面临的挑战。如果应用开发者和终端用户未能正确理解权限模型，就可能做出不当决策，从而危及设备上用户数据的安全与隐私。本研究探讨了与安卓安全研究相关的三个重要维度，填补了这些主题在系统化知识方面的空白。本研究审视了有关权限规范的研究，并指出需要为更新的 Android 版本制定此类精确规范。权限映射的知识和可用性可能影响未来检测欺骗性应用和 Android 恶意软件的研究。

我们介绍了权限的发展历程以及研究权限模型挑战的开创性贡献。本研究进一步加深了人们对权限检查与授予方式的认识。我们还基于所分析的多项研究提出了未来方向。分析得出的结论是，在分析现代应用和可复用数据集以评估欺骗性应用分析方面，相关研究仍然十分匮乏。我们还建议以表格形式提供简短的参考文献目录，引导读者查阅现有最实用、最新且重要的出版物。

参考文献

[1] BuildFire，《移动应用下载与使用统计。2021年。》[在线]。网址：buildfire.com/app-statistics[2] Statista，《Google Play 商店可用应用数量报告。2022年。》[在线]。网址：https://www.statista.com/statistics/266210/number-of-available-applications-in-the-google-play-store/ [3] ——，《设备上的安卓平台份额。2021年。》 [在线]。网址：https://bit.ly/3httTYL [4] IDC，《按操作系统划分的智能手机市场份额。》 [在线]。网址：www.idc.com/promo/smartphone-market-share/os [5] B. Krishnamurthy 和 C. E. Wills，《通过在线社交网络泄露个人身份信息》，SIGCOMM Comput. Commun. Rev.，第 40 卷，第 1 期，第 112–117 页，2010 年 1 月。 [在线]。网址：https://doi.org/10.1145/1672308.1672328[6]《避免发送个人身份信息（PII）的最佳实践》，https://support.google.com/analytics/answer/6366371，2022年，最后访问于2022年5月。[7] R. K. Solanki、V. Laxmi 和 M. S. Gaur，《MAPPER：将应用描述映射到权限》，《互联网与系统的风险和安全》。CRiSIS 2019。《计算机科学讲义》，第 12026 卷，第 84–98 页，2020年。[8] R. K. Solanki、V. Laxmi、B. Bezawada 和 M. S. Gaur，《MapperDroid：验证从描述到权限和 API 调用的应用能力》，《计算机与安全》，第 111 卷，第 102493 页，2021年。[在线]。网址：https://www.sciencedirect.com/science/article/pii/S0167404821003175 [9] 2021，Google Play 商店。《隐私、欺骗与设备滥用》，https://bit.ly/3FotAIW，2021年，最后访问于2022年5月。[10] A. K. Jha、S. Lee 和 W. J. Lee，《安卓应用中的基于权限的安全：从策略专家到终端用户》，载于《2015年自适应与融合系统研究会议论文集》，系列：RACS。美国纽约州纽约市：计算机协会，2015年，第 319–320 页。 [在线]。网址：https://doi.org/10 1145 2811411 2811493 。/。《剑桥分析丑闻》，《Vox》，[11] 2018年。 [在线]。网址：https://bit.ly/3tw8IYE [12]《剑桥分析》，《卫报》，2018年。 [在线]。网址：https://bit.ly/2SJphUf [13] Y. Yang、J. West、G. K. Thiruvathukal、N. Klingensmith 和 K. Fawaz，《你真的静音了吗？：视频会议应用静音按钮的隐私分析》，2022年。 [在线]。网址：https://arxiv.org/abs/ 2204 06128 。D. Arp、M. Spreitzenbarth、M. Hubner、H. Gascon、K. Rieck 和 C. Siemens，《Drebin：有效且可解释的口袋安卓恶意软件检测》，载于 Ndss，第 14 卷，2014年，第 23–26 页。[15] D. Geneiatakis、I. N. Fovino、I. Kounelis 和 P. Stirparo，《一种安卓移动应用权限验证方法》，《计算机与安全》，第 49 卷，C 期，第 192–205 页，2015年3月。 [在线]。网址：https://doi.org/10 1016 2014 10 005 。/j.cose. . . A. P. Felt、[14] E. Chin、S. Hanna、D. Song 和 D. Wagner，《Android 权限揭秘》，载于《第18届ACM计算机与通信安全会议论文集》，系列：CCS '11。美国纽约州纽约市：计算机协会，2011年，第 627–638 页。[在线]。网址：https://doi.org/10.1145/2046707.2046779 [16]

[17] K. W. Y. Au、Y. F. Zhou、Z. Huang 和 D. Lie，《PScout：分析 Android 权限规范》，载于《2012年ACM计算机与通信安全会议论文集》，系列：CCS '12。美国纽约州纽约市：计算机协会，2012年，第 217 228 [ ] 10 1145 2382196 2382222 – 页。在线。网址：https://doi.org/。/。M. Backes、S. Bugiel、E. Derr、P. McDaniel、D. Octeau 和 S. Weisgerber，《揭开 Android 应用程序框架的神秘面纱：重新审视 Android 权限规范分析》，载于第25届 USENIX 安全研讨会（USENIX Security 16）。德克萨斯州奥斯汀：USENIX 协会，2016年，第 1101–1118 页。 [在线]。网址：https://www.usenix.org/conference/usenixsecurity16/technical-sessions/presentation/backes [19] C. Lyvas、C. Lambrinoudakis 和 D. Geneiatakis，《Dypermin：安卓平台动态权限挖掘框架》，《计算机与安全》，第 77 卷，第 472–487 页，2018年。 [在线]。网址：https://www.sciencedirect.com/science/article/pii/S0167404818304954 [20] Y. Wang、Y. Wang、S. Wang、Y. Liu、C. Xu、S.-C. Cheung、H. Yu 和 Z. Zhu，《安卓应用中的运行时权限问题：分类体系、实践与未来方向》，《IEEE软件工程汇刊》，第 01 期，第 1–1 页，2022年2月。[21] E. M. Redmiles、Y. G. Acar、S. Fahl 和 M. L. Mazurek，《安全与隐私研究人员调查方法最佳实践综述》，载于计算机科学系技术报告，2017年。[22] R. Mayrhofer、J. V. Stoep、C. Brubaker 和 N. Kralevich，《安卓平台安全模型》，ACM Trans. Priv. Secur.，第 24 卷，第 3 期，4月。在线。网址：https://doi.org/。/ I. M. Almomani 和 A. A. Khayer，《Android 权限系统的综合分析》，IEEE Access，第 8 卷，第 216 671–216 688 页，2020年。[24] K. Bakour、H. M. ¨Unver 和 R. Ghanem，《安卓恶意软件检测系统：理想与现实》，《SN应用科学》，第 1 卷，第 1120 期，2019年。 [在线]。网址：https://doi.org/10.1007/s42452-019-1124-x[25] K. Tam、A. Feizollah、N. B. Anuar、R. Salleh 和 L. Cavallaro，《安卓恶意软件与安卓分析技术的演变》，ACM Comput. Surv.，第 49 卷，第 4 期，2017年1月。 [在线]。网址：https://doi.org/10 1145 3017427 。/ P. Faruki、A. Bharmal、V. Laxmi、V. Ganmoor、M. S. Gaur、M. Conti 和 M. Rajarajan，《Android 安全：问题、恶意软件渗透与防御综述》，IEEE通信调查与教程，第 17 卷，第 2 期，第 998–1022 页，2015年第二季度。[27] Sufatrio、D. J. J. Tan、T.-W. Chua 和 V. L. L. Thing，《安卓安全：综述、分类体系与挑战》，ACM Comput. Surv.，第 47 卷，第 4， 2015 [ ] 10 1145 2733306 期，5月。在线。网址：https://doi.org/。/《Android 架构》。[ ]。在线。网址：https://source.android.com/devices/architecture [29]《应用签名。Android 源码参考，2022年。》 [在线]。网址：https://source.android.com/security/apksigning [30]《适用于 Android 5.0 及更高版本的 Multidex 支持》，http://android-doc.github.io/tools/building/multidex.html，最后访问于2022年5月。[31]《我们如何在2018年打击不良应用和恶意开发者》。 [在线]。网址：https://bit.ly/3N09J5q [32]《Google 正在清理 Play 商店。2021年。》 [在线]。网址：https://www.androidauthority.com/google-play-store-purge-2732281/ [33]《Google Play Protect。2017年。》[在线]。网址：https://developers.google.com/android/play-protect/ [34]《Google，Android 安全报告，2018年。》 [在线]。网址：https://bit.ly/2RmsIzU [35] A. Razaghpanah、R. Nithyanand、N. Vallina-Rodriguez、S. Sundaresan、M. Allman、C. Kreibich 和 P. Gill，《应用、跟踪器、隐私与监管机构：移动跟踪生态系统的全球研究》，载于 NDSS，2018年，第 1–15 页。[36] J. Saltzer 和 M. Schroeder，《计算机系统中的信息保护》，《IEEE会刊》，第 63 卷，第 9 期，第 1278–1308 页，1975年。[37] H. Nissenbaum，《作为情境完整性的隐私》，《华盛顿法律评论》，第 79 卷，第 119–157 页，2004年。 [在线]。网址：https://digitalcommons.law.uw.edu/wlr/vol 79 1 10 /iss / R. K. Solanki、V. Laxmi、B. Bezawada 和 M. S. Gaur，《MapperDroid：验证从描述到权限和 API 调用的应用能力》，《计算机与安全》，第 111 卷，第 102493 页，2021年。[在线]。网址：https://www.sciencedirect.com/science/article/pii/S0167404821003175 [39]， [ ] 《Android 安全报告》，Google。在线。网址：https://source.android.com/security/overview/reports [40] Y. Shen 和 P.-A. Vervier，《安卓恶意应用的大规模时间测量：持久性、迁移与经验教训》，载于第31届USENIX安全研讨会（USENIX Security 22）。美国马萨诸塞州波士顿：USENIX 协会，2022年8月。 [在线]。可访问：https://www.usenix.org/conference/usenixsecurity22/presentation/shen [41] Vanja Svajcer 和 Sean McDonald，《移动环境中的潜在有害应用分类》，https://www.sophos.com/en- us/medialibrary/PDFs/technical% 20papers/SvajcerMcDonald- VB2013.pdf，2013年，最后访问于2022年5月。[42] 恶意软件。2021年。[在线]。可访问：https://support.google.com/googleplay/android- developer/answer/9888380 [43] L. Li、T. F. Bissyande、M. Papadakis、S. Rasthofer、A. Bartel、D. Octeau、J. Klein 和 L. Traon，《安卓应用的静态分析》，《信息与软件技术》，第88卷，第C期，第67–95页，2017年8月。 [在线]。可访问：https://doi.org/ 10 1016 2017 04 001 。/ j.infsof. . . A. Bartel、J. Klein、M. Monperrus 和 Y. Le Traon，《用于提取大型框架权限检查的静态分析：分析安卓的挑战与解决方案》，《IEEE软件工程汇刊》，第40卷，第6期，第617–632页，2014年。[45] D. Bogdanas，《Dperm：协助安卓应用迁移至运行时权限》，arXiv预印本arXiv:1706.05042，2017年。[46] Y. Aafer、G. Tao、J. Huang、X. Zhang 和 N. Li，《精确的Android API保护映射推导与推理》，载于《2018年ACM SIGSAC计算机与通信安全会议论文集》，系列号CCS '18。美国纽约州纽约市：计算机协会，2018年，第1151–1164页。 [在线]。可访问：https://doi.org/10 1145 3243734 3243842 。/ 。C. Pacheco 和 M. D. Ernst，《Randoop：面向Java的反馈引导随机测试》，载于《第22届ACM SIGPLAN面向对象编程系统与应用会议伴随会议论文集》，系列号OOPSLA '07。美国纽约州纽约市：计算机协会，2007年，第815–816页。 [在线]。可访问：https://doi.org/ 10 1145 1297846 1297902 。/ 。R. Vallee-Rai、P. Co、E. Gagnon、L. Hendren、P. Lam 和 V. Sundaresan，《Soot——Java字节码优化框架》，载于《1999年协作研究高级研究中心会议论文集》，系列号CASCON '99。IBM出版社，1999年，第13页。[49] O. Lhot´ak 和 L. Hendren，《使用SPARK扩展Java指向分析》，载于《第12届国际编译器构造会议论文集》，系列号CC'03。柏林、海德堡：施普林格出版社，2003年，第153–169页。[50]《请求应用权限，开发者文档，2022年》。 [在线]。可访问：https://developer.android.com/training/permissions/requesting[51] W. Enck、M. Ongtang 和 P. McDaniel，《理解Android安全》，《IEEE安全与隐私》，第7卷，第1期，第50–57页，2009年1月。[52] W. Enck、D. Octeau、P. McDaniel 和 S. Chaudhuri，《安卓应用安全研究》，载于《第20届USENIX安全会议论文集》，系列号SEC'11。美国：USENIX 协会，2011年，第21页。[53] X. Wei、L. Gomez、I. Neamtiu 和 M. Faloutsos，《Android 生态系统中的权限演变》，《第28届计算机安全应用年度会议论文集》，第31–40页，2012年。[54] Y. Zhauniarovich 和 O. Gadyatskaya，《小变化，大变化：Android 权限系统的新视角》，载于《攻击、入侵与防御研究》，F. Monrose、M. Dacier、G. Blanc 和 J. Garcia-Alfaro编。Cham：施普林格国际出版公司，2016年，第346–367页。[55] P. Calciati、K. Kuznetsov、A. Gorla 和 A. Zeller，《Android 应用中的自动授予权限：关于其普遍性及对隐私潜在威胁的实证研究》，载于《第17届国际软件仓库挖掘会议论文集》，系列号MSR '20。美国纽约州纽约市：计算机协会，2020年，第114–124页。 [在线]。可访问：https://doi.org/ 10 1145 3379597 3387469 。/ 。W. Enck、M. Ongtang 和 P. McDaniel，《轻量级手机应用认证》，载于《第16届ACM计算机与通信安全会议论文集》，系列号CCS '09。美国纽约州纽约市：计算机协会，2009年，第235–245页。 [ ] 10 1145 1653662 1653691 在线。可访问：https://doi.org/ 。/ 。J. Jeon、K. K. Micinski、J. A. Vaughan、A. Fogel、N. Reddy、J. S. Foster 和 T. Millstein，《Dr. android and mr. hide：安卓应用中的细粒度权限》，载于《第二届ACM智能手机与移动设备安全和隐私研讨会论文集》，系列号SPSM '12。美国纽约州纽约市：计算机协会，2012年，第3–14页。 [在线]。可访问：https://doi.org/10.1145/2381934.2381938

[58] 《Android 6.0变更。Android开发者手册》，https://developer.android.com/about/versions/marshmallow/android-6.0-changes，2015年，最后访问于2022年5月。[59] P. Wijesekera、A. Baokar、A. Hosseini、S. Egelman、D. Wagner 和 K. Beznosov，《重新审视Android权限：关于情境完整性的实地研究》，《第24届USENIX安全研讨会论文集》，第499–514页，2015年。[60] AOSP，《平台清单文件，AOSP镜像》。 [在线]。可访问：https://bit.ly/3gWecsu [61] R. Badhwar，《限制用户/设备跟踪的MAC地址随机化》。Cham：施普林格国际出版公司，2021年，第179–182 [ ] 10 1007 978 3 030 75354 2 19 页。在线。[62] [ ] 可访问：https://doi.org/ 。/ - - - -《Android隐私：MAC随机化》。在线。可访问：https://source.android.com/devices/tech/connect/wifi-mac-randomization [63] 《Android 10中的隐私变更。Android开发者手册》，https://developer.android.com/about/versions/10/privacy/changes，2021年，最后访问于2022年5月。[64]Jen Chai，《让用户更好地控制其位置数据。2019年》。[在线]。可访问：https://android-developers.googleblog.com/2019/03/giving-users-more-control-over-their.html[65]M. Nauman、S. Khan 和 X. Zhang，《Apex：通过用户定义的运行时约束扩展安卓权限模型与实施》，载于《第5届ACM信息、计算机与通信安全研讨会论文集》，系列号ASIACCS '10。美国纽约州纽约市：计算机协会，2010年，第328 332 [ ] 10 1145 1755688 1755732 –页。在线。[66] 可访问：https://doi.org/ 。/ 。S. Wang、Y. Wang、X. Zhan、Y. Wang、Y. Liu、X. Luo 和 S.-C. Cheung，《Aper：面向安卓应用的演化感知的运行时权限滥用检测》，2022 [ ] 2201 12542 。在线。可访问：https://arxiv.org/abs/ 。[67] 《Google的Project Strobe，》。2018 [ ] 在线。可访问：https://www.blog.google/technology/safety-security/project-strobe/ [68] 《Project Strobe更新新政策，Google，2019年》。 [在线]。可访问：https://blog.google/technology/safety-security/update-project-strobe-new-policies-chrome-and-drive/ [69] 《Firebase使用案例》。[在线]。可访问：https://firebase.google.com/use-cases [70] Serge Egelman，《AppCensus》。 [在线]。可访问：search.appcensus.io [71] S. Englehardt 和 A. Narayanan，《在线跟踪：对100万个网站的测量与分析》，载于《2016年ACM SIGSAC计算机与通信安全会议论文集》，系列号CCS '16。美国纽约州纽约市：计算机协会，2016年，第1388–1401页。 [在线]。可访问：https://doi.org/ 10 1145 2976749 2978313 。/ 。T. Vidas、N. Christin 和 L. Cranor，《遏制安卓权限蔓延》，载于W2SP，第2011卷，2011年1月。[72] [73] R. Pandita、X. Xiao、W. Yang、W. Enck 和 T. Xie，《WHYPER：迈向移动应用风险评估自动化》，《第22届USENIX安全研讨会论文集》，第527–542页，2013年。[74] Z. Qu、V. Rastogi、X. Zhang、Y. Chen、T. Zhu 和 Z. Chen，《AutoCog：衡量Android应用中的描述与权限的一致性》，《ACM计算机与通信安全会议论文集》，第1354–1365页，2014年。[75] 《Android隐藏API，Anggrayudi Hardiannicko》。 [在线]。可访问：https://github.com/anggrayudi/android-hidden-api [76] J. Sellwood 和 J. Crampton，《沉睡的Android：休眠权限的危险》，载于SPSM '13，2013年。[77] K. W. Y. Au、Y. F. Zhou、Z. Huang、P. Gill 和 D. Lie，《短文：智能手机权限模型概览》，载于《第1届ACM智能手机与移动设备安全和隐私研讨会论文集》，系列号SPSM '11。美国纽约州纽约市：计算机协会，2011年，第63–68页。 [在线]。可访问：https://doi.org/ 10 1145 2046614 2046626 。/ 。Z. Wang、Q. Liu 和 Y. Chi，《基于深度学习的Android恶意软件检测综述》，《IEEE Access》，第8卷，181 126，2020年181 102页。– 。[78] [79] 《糖尿病应用Glucosio请求录音权限，2016年》。[在线]。可访问：https://bit.ly/3fe7vQJ [80] G. S. Tuncay、S. Demetriou 和 C. A. G. Karan Ganju，《解决安卓自定义权限的困境》，载于NDSS，2018年，第1–15页。[81] 《Material Design中的Android权限，2021年》。 [在线]。可访问：https://material.io/design/platform-guidance/android-permissions.html