【专业文献13】

Android Malware Detection & Protection: A Survey （2016）

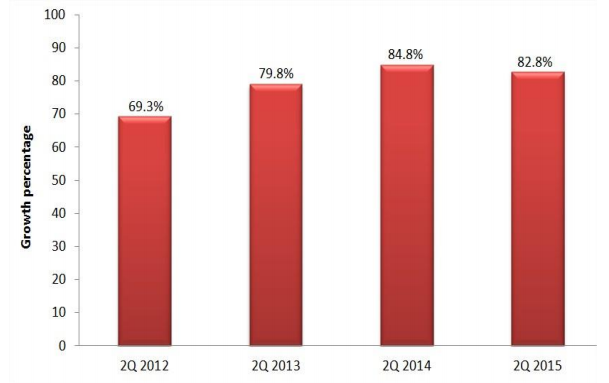
Android 恶意软件检测和保护：一项调查

摘要——Android已经成为最流行的智能手机操作系统。与前几年相比，Android应用的迅速增加导致了恶意软件数量的显著增加。在移动系统中存在着大量的反恶意软件程序，旨在有效地保护用户的敏感数据免受此类攻击。在本文中，我们的贡献是双重的。首先，我们分析了安卓恶意软件及其用于攻击系统的渗透技术，以及用于保护安卓系统的反病毒程序。我们根据检测方法对许多最新的反恶意软件技术进行分类。我们的目的是为恶意软件检测和保护机制提供一个简单明了的视图，并推断它们的优点和局限性。其次，我们预测了截至2018年的安卓市场趋势，并提供了独特的混合安全解决方案，同时考虑了安卓应用程序的静态和动态分析。

关键词——安卓； 权限； 签名

1. 引言（引言部分详细介绍了Android在市场份额的变化历程，其优点和漏洞，现实中存在的恶意软件的攻击场景，提出了静态方法和动态方法来应对恶意应用威胁，简要提出了对不同的恶意软件的分析，和一种Android安全的混合解决方案，并对论文其余部分组织进行了简单介绍。）

自2008年以来，智能手机的使用率大幅上升。智能手机提供不同的连接选项，如Wi-Fi、GSM、GPS、CDMA和蓝牙等。这使得它们成为无处不在的设备。谷歌表示，每天有130万台Android设备被激活[1]。Android操作系统在2013年占据了超过78%的市场份额，其竞争对手远远落后[2]。Gartner 2013年智能手机销售报告显示，与2012年相比，智能手机的销售增长了42.3%。根据国际数据公司IDC的数据，在2015年第2季度，Android操作系统以82.8%的总市场份额占据主导地位[3]。图1显示了Android操作系统每年的市场份额。可以看出，安卓系统多年来已经成为使用最广泛的操作系统。



Android平台以极低的成本提供了复杂的功能，并已成为手持设备最流行的操作系统。随着Android的流行，它已经成为攻击者和恶意软件开发者的主要目标。Android官方市场每天都有数百万的应用程序被用户下载[4]。Android提供了一个开放市场模型，没有任何应用程序经过任何安全专家的验证，这使得Android很容易成为开发者将恶意内容嵌入其应用程序的目标。用户的敏感数据很容易被泄露，并且可以传输到其他服务器。此外，第三方应用商店的存在助长了Android恶意软件的传播，因为Google Play也托管了第三方开发者的应用程序。Android官方市场使用Bouncer保护市场免受恶意软件攻击[5]。但是，Bouncer不会分析上传应用的漏洞。恶意软件开发者通过重新打包Google Play的流行应用并将其分发到其他第三方应用商店，利用应用之间的漏洞。这会降低应用商店和开发者的声誉。恶意软件包括计算机病毒、特洛伊木马、广告软件、后门软件、间谍软件和其他旨在破坏操作系统以及窃取个人、财务或商业信息的恶意程序。恶意软件开发人员使用代码混淆方法、动态执行、隐形技术、加密和重新打包来绕过Android平台提供的现有反恶意软件技术。

据美国媒体报道，周四谷歌推出一项新服务Bouncer，在保护Android用户免受恶意软件侵害的同时，为开发者提供比苹果应用程序审查政策更自由的政策。谷歌在博客上表示，将在当地时间周四首次推出这个服务。目前不清楚需要多久才会启动并运行。其工作原理如下：开发者如往常一样将应用程序上传到Android Market，但谷歌将在后台根据已知的恶意软件、间谍软件或特洛伊木马特征进行扫描。该服务还将在谷歌内部服务器上，模拟所上传的应用程序的活动，检测应用程序是否在用户不知情的情况下收集和发布了个人信息。

为了防止此类恶意软件，重要的是对其有准确深入的了解，以便采取相应的安全措施保护用户数据。存在大量攻击场景，攻击者可以利用Android操作系统的漏洞危害用户数据。例如，特洛伊木马应用程序在用户许可的情况下下载一些高清墙纸，但该许可可能允许该应用程序访问用户的联系人或其他个人信息，并将用户的机密数据从设备秘密泄漏到其他服务器。在这种情况下，墙纸应用程序将具有互联网下载权限。用户可能不太注意其他请求的访问权限，可能会意外授予READ\_CONTATCS权限。因此，应用程序可能会修改设备设置，损坏用户数据，并将私人数据传输到一些未知的远程服务器。这会导致用户的业务数据和其他个人信息丢失。攻击者可将被盗数据用于绑架、勒索或商业损失目的。在另一个攻击场景中，攻击者将恶意应用程序作为一些流行应用程序的重新打包版本分发，这些应用程序可能提供基于位置的服务，因此在这种情况下，恶意应用程序会通过过度使用GPS和无线电等耗尽其电池来杀死受害设备。一些恶意程序会获取用户设备的IMEI号码，并将其发送到远程服务器服务器 这些IMEI号码在黑市中具有重要价值，在黑市中，被盗设备的IMEI号码可以用用户的IMEI修改[6]。

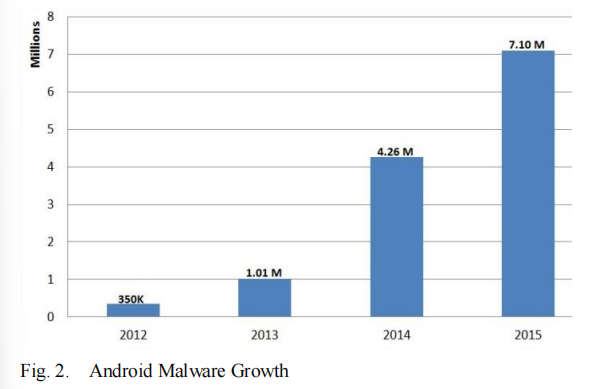
有数百种恶意软件技术以多种方式攻击Android平台，如在受害者不知情的情况下发送消息并自行删除，将用户的私人信息发送到其他服务器等。因此，非常需要保护用户数据免受这些恶意软件的侵害。

这种不断增加的恶意软件威胁迫使安卓反恶意软件行业开发解决方案，以缓解安卓智能手机和其他安卓设备上的恶意应用威胁。为此，主要采用两种方法：静态方法和动态方法。防病毒程序使用这些方法中的任何一种来保护移动系统免受恶意软件攻击。他们检测恶意应用程序并通知用户此类应用程序，并采取措施删除这些恶意软件。随着威胁级别数量的增加，防病毒检测率也随之提高。由于威胁和恶意软件以及安卓反恶意软件程序提供的保护机制，安卓用户的总体风险状况难以评估[7]。

在本文中，我们分析了不同的恶意软件、它们的行为和不同类型的恶意软件用来攻击Android设备的技术。此外，本文还详细介绍了各种反恶意软件技术及其优点和局限性。在此基础上，提出了一种Android安全的混合解决方案。论文的其余部分组织如下。第二节根据现有恶意软件的行为对其进行分类。第三节包括攻击者采用的恶意软件渗透技术。第四节对安卓设备的检测和删除方法进行了详细分析。第五节包括反恶意软件机制的性能评估。第六节介绍了Android市场份额和恶意软件增长的未来趋势以及现有反恶意软件方法的局限性。本节还提出了旨在提供更好安全机制的解决方案。本文件的结论见第七节。

1. 安卓恶意软件分析（给出了恶意软件的数量变化图，简要介绍了几种典型的恶意软件及其实现方法，以及检测到的不同恶意软件类型占比表和图）

广泛的恶意软件被检测到，并且恶意软件的数量每年都在增加。根据TrendMicro的数据，2015年上半年（上半年）恶意软件已增至710万[8][9]。图2显示了近年来Android恶意软件数量的增加。不同恶意软件系列的行为将在后续章节中提供。



1. 特洛伊木马

特洛伊木马在用户看来是一种良性应用[5]。事实上，他们实际上在用户不知情的情况下窃取了用户的机密信息。此类应用程序可以轻松访问受害者设备的浏览历史记录、消息、联系人和设备IMEI号码等，并在未经用户同意的情况下窃取这些信息。FakeNetflix[10]就是此类恶意软件的一个例子，它提供与原始Netflix应用程序相同的用户界面，并收集用户的登录凭据。SMS特洛伊木马程序利用高级服务给受害者造成经济损失。Fakeplayer是一种著名的SMS特洛伊木马，它在用户不知情的情况下将消息发送到高费率号码[11]。Zsone[12]和Android.foney也是此类SMS特洛伊木马程序的例子。恶意软件还捕获用户的银行信息，如账号和密码。Zitmo和Spitmo特洛伊木马旨在窃取用户的MTAN（移动交易认证号码），然后这些MTAN以静默方式完成交易[13]。

1. 后门

后门利用root漏洞向恶意软件授予root权限，并帮助它们隐藏起来以防被使用。Exploid、Rageagainstthecage（RATC）和Zimperlich是获得设备完全控制权的前三个根攻击[14]。DroidKungFu[15]以加密的形式使用根漏洞，即Exploid和RageAgainstCache。当DroidKungFu执行时，它首先解密并启动根攻击。如果根攻击成功获得对设备和根权限的控制，恶意软件将能够在设备上执行任何操作，甚至安装应用程序，从而使用户不知道此行为[16]。

1. 蠕虫

这类恶意软件创建其副本并通过网络分发。例如，蓝牙蠕虫通过蓝牙网络向配对设备发送恶意软件的副本来传播恶意软件。Android.Obad.OS就是蓝牙蠕虫的例子[17]。

1. 间谍软件

Nickspy[11]和GPSSpy[18]是间谍软件应用程序的例子，它们看起来是良性应用程序，但实际上它监控用户的机密信息，如消息、联系人、银行账号、位置等，以产生一些不良后果。个人间谍软件可以在受害者不知情的情况下安装恶意负载。它将用户信息（如短信、联系人等）发送给在受害者设备上安装该软件的攻击者[6]。

1. 僵尸网络

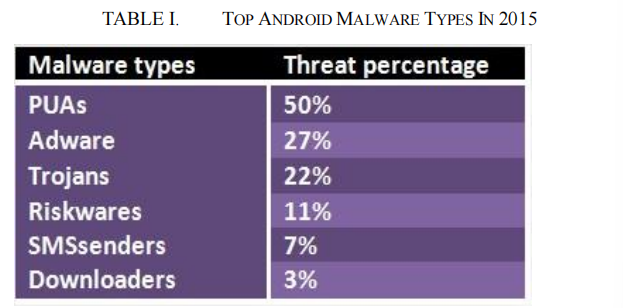
僵尸网络是一个安卓设备受损的网络。远程服务器Botmaster通过C&C网络控制僵尸网络。Geinimi[11]是Android僵尸网络之一。

1. 勒索

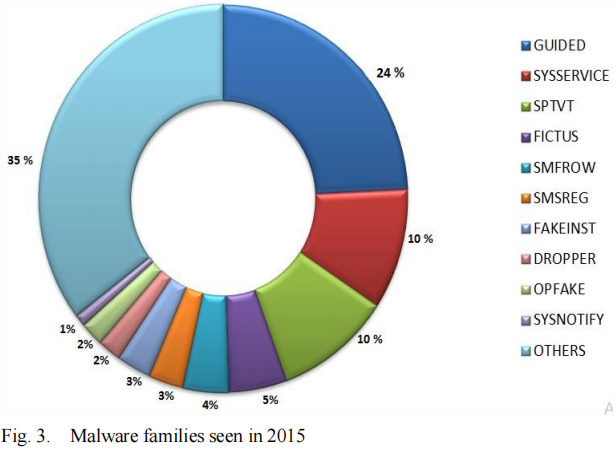
勒索软件通过锁定设备防止用户访问设备上的数据，直到支付了勒索金额。FakeDefender.B[19]是一种伪装成avast！杀毒软件的恶意软件。它锁定受害者的设备，并强制用户支付赎金金额以解锁设备。

1. 风险软件

风险软件是恶意作者利用合法软件降低设备性能或损害数据，如删除、复制或修改等[20]。下表1显示了TrendMicro在2015年检测到的顶级恶意软件类型[21]。



从[21]中获得的统计数据已计算并绘制在图3中，图3显示了TrendMicro在2015年第二季度（第2季度）记录的顶级Android恶意软件系列。根据该报告，24%的恶意软件是引导式变体，没有任何GUI，在用户不知情的情况下在后台默默运行。



1. 恶意软件渗透技术（介绍了四种恶意软件技术及其实现方法）
2. 重新包装

恶意软件作者重新打包Android官方市场、Google Play的流行应用程序，并将其分发到其他监控较少的第三方应用商店。重新打包包括分解流行的良性应用程序，包括免费和付费；附加恶意内容并重新组装应用程序。此重新打包过程由反向工程工具完成。在重新打包期间，恶意作者会更改重新打包的应用程序的签名，因此该应用程序对于反恶意软件来说似乎是新的。TrendMicro报告显示，Google Play提供的前50个免费应用中，77%都经过了重新打包[22]。

1. 下载驱动

它指的是在后台无意下载恶意软件。当用户访问包含恶意内容的网站，并在用户不知情的情况下将恶意软件注入受害者的设备时，就会发生驱动下载攻击。恶意软件开发者使用Android/NotCompatible[23]，这是一款驱动下载应用程序。

1. 动态有效载荷

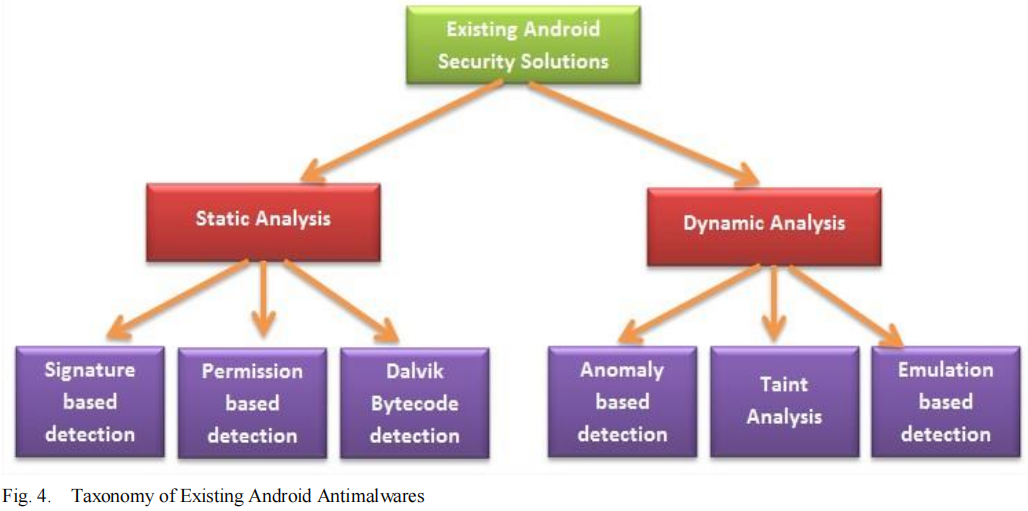
恶意软件还通过动态负载技术渗透到Android设备中。他们加密恶意内容并将其嵌入APK资源中。安装后，应用程序解密加密的恶意负载并执行恶意代码。有些恶意软件不是将有效负载作为资源嵌入，而是从远程服务器动态下载恶意内容，静态分析方法无法检测到[24]。

1. 隐形恶意软件技术

在Android设备上，由于电池等有限资源的可用性，恶意软件扫描程序无法执行深入分析。恶意软件开发人员利用这些硬件漏洞并混淆恶意代码以轻松绕过反恶意软件。不同的秘密技术，如密钥置换、动态加载、本机代码执行、代码加密和java反射，被用来攻击受害者的设备。

1. ANDROID恶意软件检测（介绍了分析恶意软件的两种方法，静态方法和动态方法，及其各自的具体类型，以及各种类型的实现方法、优缺点及局限性）

分析Android恶意软件的方法主要有两种：静态方法和动态方法。我们使用静态和动态方法对反恶意软件进行了进一步分类。图4显示了基于我们研究的现有反恶意软件技术的分类。



1. 静态方法

静态方法是通过反汇编和分析应用程序的源代码来检查应用程序的功能和恶意性的一种方法，而无需执行应用程序。它有助于发现在特定情况发生之前可能无法运行的恶意行为。

1. 基于签名的方法

商业反恶意软件产品通常使用基于特征码的恶意软件检测方法。该方法提取语义模式并创建唯一的签名[25]。如果程序的签名与现有恶意软件系列的签名相匹配，则该程序被归类为恶意软件。基于签名的检测的主要缺点是，它可以很容易地通过代码混淆来规避，因为它只能识别现有的恶意软件，并且无法抵御看不见的恶意软件变体。当检测到恶意软件变体时，它需要立即更新。

Faruki等人[26]提出了AndroSimular，这是一种稳健的统计签名方法，用于检测现有恶意软件的未知变体，这些未知变体通常通过重新打包和代码混淆技术生成。它为被测应用程序生成可变长度签名，并将其与类似恶意软件数据库中的签名进行比较，并根据相似性百分比将该应用程序识别为恶意软件和良性应用程序。作者对1260个应用程序进行了测试，其中6779个应用程序是Google Play应用程序，545个应用程序来自第三方应用商店。他们还使用代码混淆技术，如方法重命名、字符串加密、控制流混淆和垃圾方法插入技术来更改代码的签名，并针对426个样本测试了AndroSimular的有效性。该解决方案正确检测到超过 60% 的样本。AndroSimular比较应用程序的签名，以便区分恶意软件和良性应用程序，但与其他防病毒解决方案相比，它的签名数据库有限。因此，任何看不见的恶意软件都不会被发现。此外，相似性百分比会产生误报，因为它可能会根据百分比将干净的应用程序归类为恶意应用程序。

DroidAnalytics [27] 是一个基于签名的分析系统，它在操作码级别提取和分析应用程序。 它不仅会生成签名，还会在识别出恶意内容后将恶意软件与现有恶意软件关联起来。 它生成 3 个级别的签名。 首先它通过 API 调用跟踪在方法级别生成签名，然后将类中的所有方法签名组合起来生成类级别签名，在第三级它通过组合应用程序中的类的签名生成应用程序签名。 作者使用 DroidAnalytics 检测了来自 102 个恶意软件系列的 2,494 个恶意软件样本和来自其他六个恶意软件系列的 342 个重新打包的恶意软件。 这种方法的局限性包括，它根据恶意软件家族主要使用的类别将应用程序分类为恶意软件，但在实验中他们发现了一些合法应用程序和恶意软件都使用的签名。 此外，用于检测重新打包的恶意软件的相似度分数不能提供 100% 的解决方案，或者它也可能提供误报，将合法应用程序归类为恶意软件。

·基于签名的检测的局限性：虽然基于签名的检测对于已知恶意软件非常有效，但它无法检测未知的恶意软件类型。此外，由于签名数据库有限，大多数恶意软件仍然未被发现。

1. 基于权限的分析：

在Android系统中，应用程序请求的权限在管理访问权限方面起着至关重要的作用。默认情况下，应用程序无权访问用户数据并影响系统安全。安装期间，用户必须允许应用访问应用请求的所有资源。开发人员必须在AndroidManifest.xml文件中提及为资源请求的权限。但是，所有声明的权限不一定是该特定应用程序所需的权限。

参考文献[28]表明，大多数情况下，开发人员都声明了应用程序实际不需要的权限，这使得很难检测应用程序的恶意行为。反恶意软件分析Android Manifest.xml文件，其中提到应用程序所需资源的所有权限。Stoway[28]暴露了Android中的权限优先问题，即应用程序请求的权限超过了实际使用的权限。Stoway执行静态分析以确定应用程序调用的API调用，然后映射API调用所需的权限。他们发现，在940个安卓应用程序样本中，有三分之一的应用程序特权过高。它无法解析应用程序使用java反射调用的API调用。

在[29]中，作者提出了一种轻型恶意软件检测机制，该机制仅分析清单文件并提取权限、意图过滤器（操作、类别和优先级）、进程名称和重新定义的权限数等信息，以检测应用程序的恶意行为。在提取这些信息之后，他们将其与所提出的方法中提供的关键字列表进行比较，然后计算恶性度评分。他们使用Weka[30]，这是一种用于计算阈值的数据挖掘工具。最后，他们将恶性评分与阈值进行比较，如果恶性评分超过阈值，则将应用程序归类为恶意软件。他们使用了365个样本来测试提议的解决方案的效率，该解决方案提供了90%的准确检测。这是一种节省成本的机制，因为它只包括对清单文件的分析，并且可以在其他检测体系结构中实现，以便有效地检测恶意软件。它甚至可以检测那些基于特征码的检测方法无法检测到的恶意软件。此建议的解决方案仅限于清单文件信息。此外，它无法检测广告软件样本。

C.Y.Haung等人[31]提出了一种更好地检测基于权限的恶意软件检测的方法，该方法包括分析请求的权限和要求的权限，因为大多数时候恶意软件作者在清单文件中声明的权限比他们实际需要的应用程序多。此外，它还分析了易于检索的功能，然后将应用程序标记为良性或恶意软件。三种不同的标签类型用于此目的，包括基于站点的标签；基于扫描仪的标签和混合标签。在基于站点的标记中，如果应用程序是从谷歌官方应用程序市场下载的，则会将其标记为良性；如果应用程序是从某个恶意源下载的，则会将其标记为恶意。在第二种标记方案中，如果防病毒扫描程序将应用程序声明为良性，则应用程序将被标记为良性，并且与恶意软件案例相同。在混合标签中，应用程序的标签基于基于站点和基于扫描仪的标签。标记后，将所有样本划分为三个数据集，并通过机器学习算法（如朴素贝叶斯、AdaBoost、支持向量机和决策树）分析这些数据集的请求权限[32]。基于这些分类器生成的结果，我们可以评估基于权限的检测方法的性能。在[31]中，作者对124769个良性应用程序和480个恶意应用程序的数据集进行了实验。他们分析了基于权限的恶意软件检测的性能，结果表明，基于权限的检测方法可以检测到81%以上的恶意应用程序样本。该方法为恶意软件检测提供了快速过滤器，但分类器生成的性能值并不完美，我们不能完全依赖这些结果。

Sanz Borja等人[33]通过分析应用程序请求的权限，提出了检测恶意应用程序的PUMA。他们使用AndroidManifest.xml文件中的<uses-permission>和<uses-features>等权限标记来分析应用程序的恶意行为，并在357个良性应用程序和249个恶意应用程序的数据集上应用了不同的分类器算法。该解决方案提供了较高的检测率，但生成的结果具有较高的误报率，并且不足以有效检测恶意软件，它仍然需要与其他功能和动态分析相关的信息。

Shin等人[34]使用基于状态机的方法，正式分析了基于权限的Android安全模型。他们还验证了指定的系统满足安全属性。

Tang，Wei等人[35]提出了一个安全距离模型，用于缓解Android恶意软件。安全距离模型基于这样一个概念，即应用程序没有一个权限足以威胁Android设备的安全。例如，请求权限读取电话状态的应用程序可以访问电话号码和IMEI，但不能将数据移出设备。必须有多种权限组合才能影响设备的安全模型，例如INTERNET权限允许将设备与网络结合起来，并需要将数据移动到某个远程服务器。SD根据应用程序请求的权限测量应用程序的危险级别。作者将权限组合分为四组，并将威胁点（TP）分配给每一组，如安全SD、正常SD、危险SD和严重SD的TP-0、1、5和25。在安装新应用程序之前，它会根据应用程序请求的权限组合计算威胁点。这有助于用户在安装应用程序时了解更危险的权限。它可以很容易地检测到未知的恶意软件，具有很高的威胁点。他们发现Geinimi恶意软件有500个威胁点，这与良性应用程序的区别非常明显。此解决方案的一个限制包括，威胁点在50到100之间的应用程序不容易识别为良性和恶意软件。它们可能是具有此类权限组合或恶意软件的良性应用程序。

Enck等人[36]开发了KIRIN，一种在安装时提供轻质认证的工具。它定义了安全规则，并简单地将应用程序的请求权限与其安全规则进行比较，如果应用程序未能通过所有安全规则，则证明该应用程序为恶意软件。如果应用程序被认定为恶意软件，则该应用程序的安装将中止。作者测试了311个从官方Android market下载的应用程序，发现有5个应用程序未能通过规定的规则。建议的解决方案重量较轻，因为它只分析Menifest.xml文件。KIRIN的限制包括，它也可能将一些合法应用程序声明为恶意软件，因为为应用程序认证提供的信息不足以检测恶意软件。

DroidMat[37]是一个从清单文件中提取信息的工具，如权限、通过意图传递的消息和API调用跟踪，以分析应用程序的行为。它采用K均值聚类，提高恶意软件检测能力，并使用KNN算法将应用程序分类为良性或恶意软件[38]。它比Androgaurd[39]更有效，因为识别1738个应用程序是恶意软件还是良性软件所需的时间更少。此外，由于不需要动态模拟和手动操作，因此节省了成本。但作为一种基于静态的检测方法，它无法检测动态加载恶意内容的恶意软件，如DroidKngFu和BaseBridge。

·基于权限的检测限制：基于权限的检测是一种快速过滤器，用于扫描应用程序并识别应用程序是良性还是恶意软件，但它只分析清单文件，不分析包含恶意代码的其他文件。此外，恶意应用程序和良性应用程序使用的权限差别也很小。基于权限的方法需要第二次通过才能提供有效的恶意软件检测。

1. Dalvik字节码分析：

在Android中，Dalvik是一个基于寄存器的虚拟机。Android应用程序是用java语言开发的，用java字节码编译，然后翻译成dalvik字节码。字节码分析帮助我们分析应用程序的行为。控制和数据流分析检测恶意应用程序执行的危险功能。

Jinyung Kim等人[40]开发了一种静态分析器，用于分析应用程序的dalvik字节码，并检测应用程序中的隐私泄漏。它确定从信息源到任何远程服务器的数据流。Dalvik字节码包含分支、方法调用和跳转指令，这些指令改变了代码的执行顺序并使代码变得模糊。在执行过程中，可以通过字节码分析识别应用程序可能采用的路径。在[40]中，作者检查了来自Android官方市场的90个应用程序和来自第三方市场的8个恶意应用程序。他们在11个谷歌市场应用程序和8个第三方市场应用程序中发现了隐私泄露。由于分析应用程序会消耗更多的时间和内存，因此需要实现性能优化技术。此外，它不支持使用反射进行数据泄漏的应用程序。在这起丑闻中，作者已经手动实现了反射语义，以检测从黑市获取的恶意应用程序中的隐私泄漏。

Karlsen等人[41]提出了Dalvik字节码的首次形式化以及java反射特性。他们研究了1700个流行的安卓应用程序，以确定安卓应用程序主要使用哪些Dalvik字节码指令和功能。这种形式化有助于执行控制和数据流分析，以便检测恶意应用程序或识别执行期间调用的敏感API调用。它支持动态调度和反射功能。但它需要在并发性分析和反射处理方面进行扩展。

Zhou等人[42]实现了DroidMOSS，通过使用baksmali工具[43]提取Dalvik字节码序列和应用程序的开发人员信息，并使用模糊散列技术为每个应用程序生成指纹，以创建固定大小的80字节签名来检测重新打包的应用程序。它根据相似性评分识别重新打包的应用程序。作者应用DroidMOSS对来自六个不同第三方市场的200个样本进行了测试，发现5%到13%的应用被重新包装。如果数据库中不存在原始应用程序，则建议的解决方案无法检测重新打包的应用程序。此外，由于数据库有限，大多数恶意软件仍未被发现。谷歌游戏商店也可能包含恶意软件。此解决方案的局限性还包括，他们假设所有Google Play应用程序都是合法应用程序，然后匹配从其他应用商店获取的应用程序的签名，以检测重新打包的应用程序。

DroidAPIMiner[44]基于Androgaurd[39]，通过跟踪字节码中的敏感API调用、调用的危险参数和包级别信息来识别恶意软件。为了将应用程序分类为良性或恶意软件，它实现了KNN算法[38]，检测准确率高达99%，误报率为2.2%。

Fuchs等人[45]介绍了ScanRoid，它在安装Android应用程序时静态分析，并执行数据流分析，以检查通过应用程序的数据流是否一致。在数据流的基础上，它声明应用程序可以使用请求的权限安全运行。作者将其用作Android应用程序的安全认证工具。

许多研究人员致力于将Dalvik字节码转换为Java字节码，然后对Java代码进行静态分析，以检测应用程序的恶意行为。ded[46]和Dare[47]是用于将dalvik字节码转换为java字节码的工具。当开发人员不分发java源代码时，这些工具也很有用，在这种情况下，必须分析源代码，通过静态分析检测恶意软件。Dexpler工具[48]将Dalvik字节码转换为Jimple代码，该代码由名为 Soot [49] 的静态分析框架使用。它使得直接读取Dalvik字节码并执行静态分析成为可能，而无需将Dalvik字节码转换为java字节码。研究人员使用的著名静态分析框架是WALA，它对java字节码执行静态分析，以检测恶意应用程序中的隐私泄漏[50]。

Chin等人[51]提出了一个名为ComDroid的工具，用于检测Android应用程序之间基于通信的漏洞。他们分析了20个样本，在12个应用程序中发现了34个可利用的漏洞。它使用Dedexer工具[52]来分解应用程序中的dex文件。它对Dalvik文件执行静态分析，分析应用程序的manifest.xml文件中列出的权限，执行过程内分析，并检查应用程序的意图以检测通信漏洞

·Dalvik字节码检测的局限性：在这种方法中，分析是在指令级执行的，并且消耗更多的功率和存储空间。由于android设备资源匮乏，因此限制了这种检测方法。

1. 动态方法

动态分析在执行过程中检查应用程序。它可能会遗漏一些未执行的代码部分，但它可以轻松识别静态分析方法无法检测到的恶意行为。虽然静态分析方法对恶意软件的检测速度更快，但它们无法抵抗代码混淆和加密恶意软件。

在[53]中，Egele详细概述了用于区分恶意软件和良性应用程序的不同动态分析方法。动态分析方法对于恶意软件采用的多态和变形代码混淆技术是有效的[54]，但它需要更多的资源。

1. 基于异常的检测

Iker等人[55]提出了CrowDroid来动态检测应用程序的行为。应用程序调用的系统调用的详细信息由Strace工具[56]收集，然后安装在设备上的众包应用程序创建一个日志文件并将其发送到远程服务器。日志文件可能包括以下信息：设备信息、设备上安装的应用程序和系统调用。服务器端采用2-均值聚类算法将应用程序分为恶意软件和良性软件。结果存储在服务器数据库中。该解决方案提供了深入的分析，因此需要大量的资源。该解决方案要求在用户设备上安装客户端应用程序，如果合法应用程序调用更多系统调用，则可能会将其归类为恶意软件。

Shabtai等人[57]提出了Andromly，一个基于行为的Android恶意软件检测系统。为了将应用程序分类为良性或恶意软件，它会在运行时持续监控指示设备状态（如电池电量、CPU消耗等）的不同功能和模式，然后应用机器学习算法区分恶意和良性应用程序。该解决方案可以检测到连续攻击，并将这些攻击通知用户。

AntiMalDroid[58]是Zhao提出的一种使用SVM算法的恶意软件检测框架，可以在执行过程中识别恶意应用及其变体。它首先监视应用程序的行为及其特征，然后将这些特征归类为正常行为和恶意行为。然后将这两类特征放入学习模块，并生成学习模块生成的行为特征的签名。然后将签名存储在数据库中，并将其与现有恶意软件和良性应用程序签名进行比较。如果签名与已有的良性应用程序签名匹配，则将该应用程序归类为良性应用程序。该方案可以动态扩展签名数据库，具有较高的检测率。但在检测过程中会消耗更多的时间。

1. 污点分析

Enck等人[59]提出了TaintDroid，为Android提供系统范围的信息流跟踪。它可以同时跟踪多个敏感数据源，如摄像头、GPS和麦克风等，并识别第三方开发者应用程序中的数据泄漏。当受污染的数据离开设备时，它会标记敏感数据并跟踪这些数据和应用程序。它提供敏感信息的有效跟踪，但不执行控制流跟踪。此外，它无法跟踪离开设备并在网络回复中返回的信息。

控制流是指按一定的顺序排列程序元素来决定程序执行的顺序。

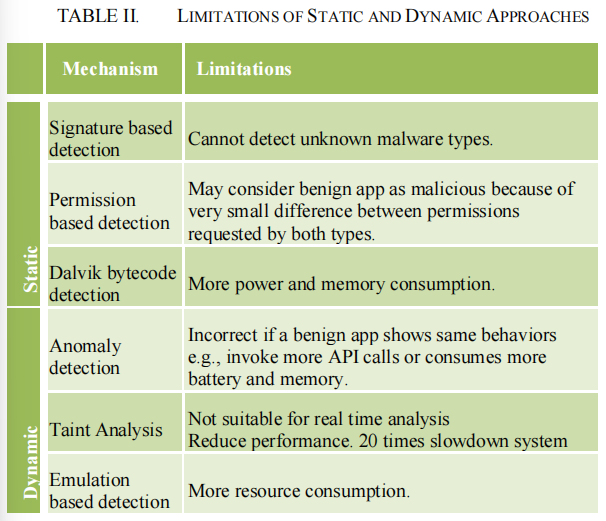
1. 基于仿真的检测

Yan等人[60]提出了基于虚拟机内省的Android动态分析平台DroidScope。由于反恶意软件检测到恶意软件的存在，因为它们都位于相同的执行环境中，因此恶意软件也可以检测到反恶意软件的存在。DroidScope通过远离执行环境来监控整个操作系统，因此拥有比恶意软件程序更多的权限。它还监视Dalvik语义，因此还可以检测到对内核的权限提升攻击。它建立在QEMU之上。机器人梦和机器人功夫[61]是用这种技术检测出来的。

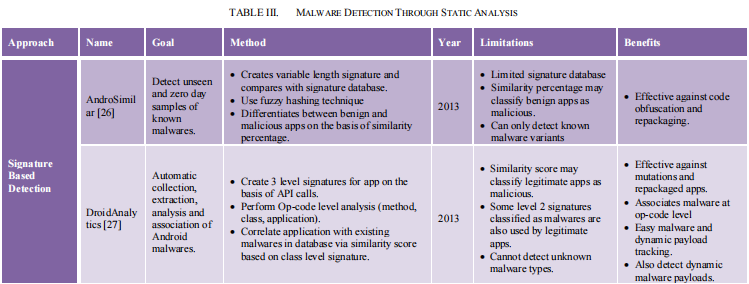
Blaising等人[62]提出了Android应用程序沙箱（Asandbox），通过对可疑应用程序执行静态和动态分析来检测这些应用程序。它首先将.dex文件提取为人类可读的形式，然后对应用程序执行静态分析。然后，通过在隔离的沙箱环境中执行应用程序，分析了与系统的底层交互。由于安全策略，应用程序的操作仅限于沙箱，不会影响设备上的数据。它使用Money工具动态分析应用程序行为，随机生成用户事件，如触摸、单击和手势等。它无法检测新的恶意软件类型。

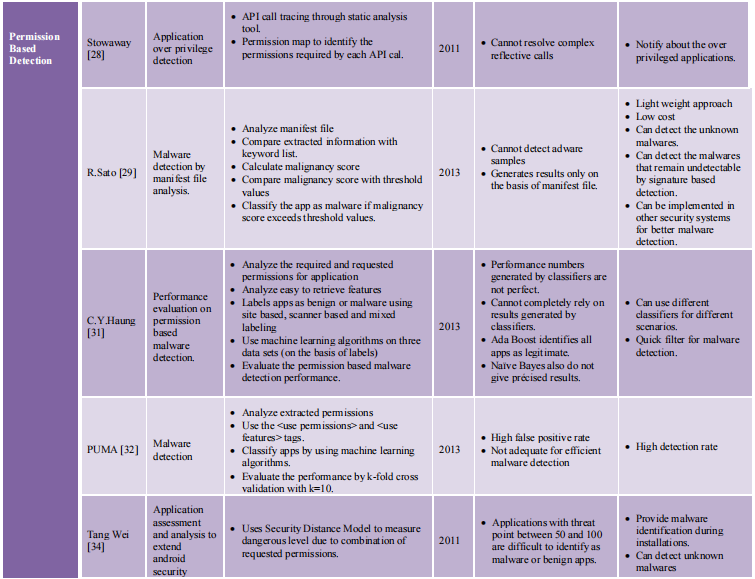
1. 性能评估与分析（给出了恶意软件检测的静态和动态方法的局限性表，和通过静态分析和动态分析进行恶意软件检测的表）

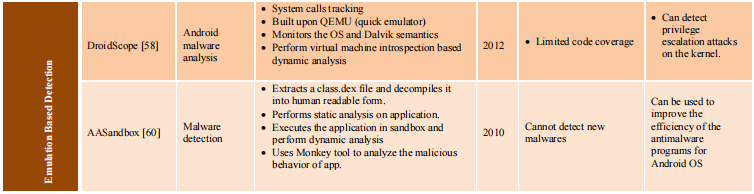
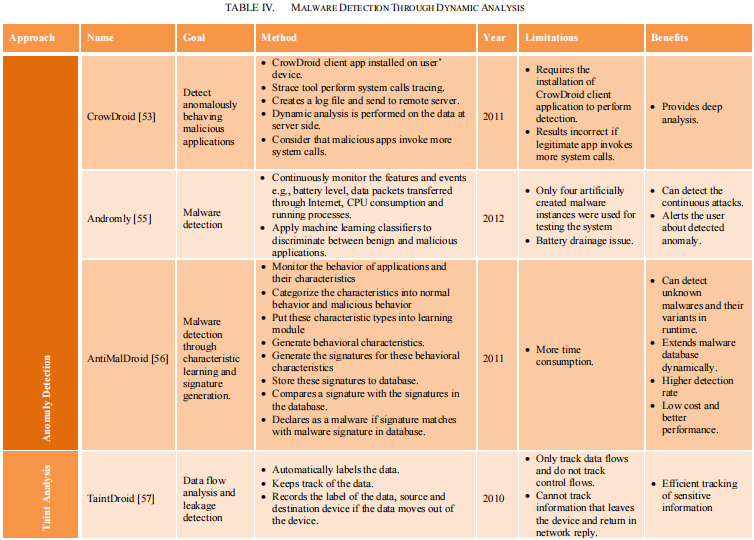
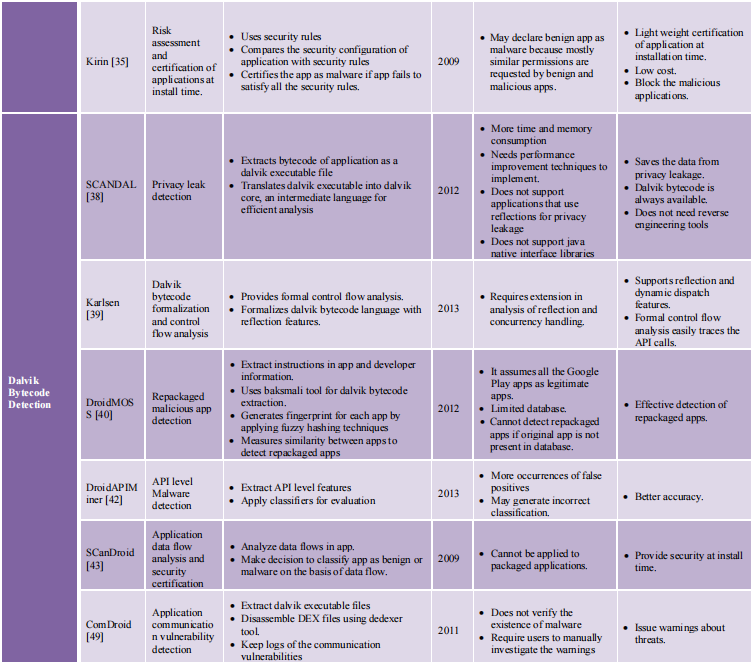
在本节中，我们将评估不同参数的性能，并对不同属性进行全面比较。表2提供了恶意软件检测的静态和动态方法的局限性。表3和表4分别提供了通过静态分析和动态分析进行的恶意软件检测。



根据它们的工作技术，我们推断出每种检测机制的主要局限性和优点。

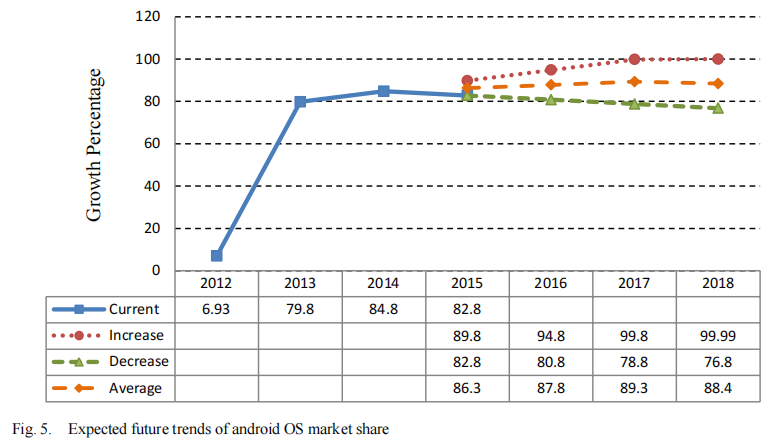


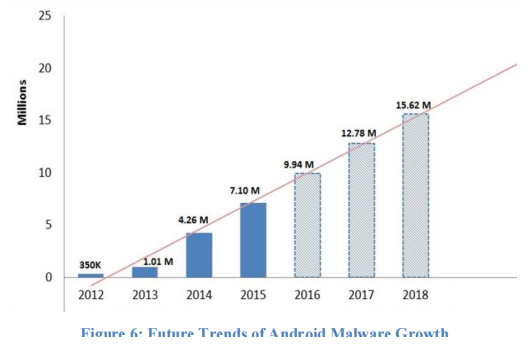




1. 讨论（给出了Android市场变化趋势预测，提出了静态和动态混合的方法将是未来工作的重点）

安卓操作系统的普及程度正在急剧上升。IDC[3]提供的年度记录显示，2015年第二季度Android操作系统的市场份额为82.8%，比2014年第二季度下降了2%。如果价值在年底前保持不变，并以同样的速度逐年下降，那么我们可以预计2018年Android市场份额将降至76.8%。根据同样的记录，2014年安卓的股票比前一年增长了5%。如果继续以同样的速度增长，到2016年底增长到89.8%，那么我们可以说2018年Android的份额将增长到99.9%。此外，预计2018年Android的市场份额将平均达到88.4%。Android市场的估计和未来预测如图5所示。应该注意的是，随着基于Android的设备使用率的增加，攻击Android的恶意软件数量正以指数级的速度增长。2015年，安卓恶意软件数量激增至710万。这一数字比上一年增加284万[8][9]。如果恶意软件的增长率保持不变，预计2018年这一数字将增加到1580万。图6预测了恶意软件的增长趋势，并提供了估计值。





与恶意软件相比，反恶意软件的设计和开发范围很广，目的是保护设备。可以推断，使用静态方法的反恶意软件在检测从远程服务器动态加载的恶意内容方面效率较低。尽管如此，动态方法还是很有效的，因为它可以持续监视应用程序，并能够在执行时检测恶意内容。但是，未执行的恶意代码部分仍然未被检测到。据信，Android中的任何单一安全解决方案都无法提供针对漏洞和恶意软件的全面保护。最好同时部署多个解决方案，例如，混合使用静态和动态两种方法。混合方法将首先静态分析应用程序，然后执行动态分析。由于可用资源（如电池、内存等）有限，这种混合解决方案可能是一种昂贵的应用方法。然而，这种混合解决方案的局限性可以从两方面解决。首先，静态分析可以在Android设备上本地执行；然后，可以通过将恶意活动或事件以日志文件的形式发送到远程服务器，以分布式方式执行动态分析。远程服务器可以快速高效地执行动态分析，因为服务器将有足够的资源来执行动态分析，并且可以针对应用程序行为生成快速响应，并且可以立即通知用户。然而，这种混合解决方案需要更多的研究，并且需要进行设计权衡。未来的工作重点将是开发这种混合反恶意软件，为android设备提供更好的安全性。

1. 结论（对全文内容重点的总结）

本文对恶意软件及其渗透技术进行了深入分析。反恶意软件根据其使用的检测方法进行分类。本文还对这些反恶意软件技术进行了详细的性能评估，并全面地推导了这些反恶意软件的优点和局限性。最后，提出了混合反恶意软件的概念，该概念将解决现有静态和动态方法的局限性。未来，它的目标是实施拟议的混合解决方案，这将是一种通用反恶意软件，通过首先静态分析本地设备上的Android应用程序，然后在远程反恶意软件服务器上执行动态分析，为Android设备提供更好的安全性。这将消耗设备上非常小的内存空间，电池消耗也将很低，因为所有动态分析都将在远程服务器上执行。