PLATPAL: Detecting Malicious Documents with Platform Diversity检测带有平台多样性的恶意文档

Meng Xu and Taesoo Kim

Georgia Institute of Technology

**Abstract:**

Due to the **continued exploitation** of Adobe Reader, **malicious document (maldoc)** detection has become a pressing problem. Although many solutions have been proposed, recent works have **highlighted some common drawbacks**, such as **parser-confusion and classifier-evasion attacks**.

In response to this, we propose a new perspective for **maldoc detection**: **platform diversity**. In particular, we identify eight factors in OS design and implementation that could **cause behavioral divergences under attack**, ranging from syscall semantics (more obvious) to heap object metadata structure (more subtle) and further show how they can **thwart attackers from finding bugs**, exploiting bugs, or performing malicious activities.

We further prototype PLATPAL to systematically harvest platform diversity. **PLATPAL hooks into Adobe Reader to trace internal PDF processing** and also uses sandboxed execution to capture a maldoc’s impact on the host system. Execution traces on **different platforms are compared**, and maldoc detection is based on the observation that a benign document behaves the same across platforms, while a **maldoc behaves differently during exploitation**. Evaluations show that PLATPAL raises **no false alarms in benign samples, detects a variety of behavioral discrepancies in malicious samples, and is a scalable and practical solution.**

**Q**uestions:

      What is the problem? 检测带有平台多样性的恶意文档

Why is the problem important?

网络攻击者正在转向基于文件的恶意软件，因为用户聪明起来恶意电子邮件附件和网络链接，许多防病毒（AV）供应商[如建议39 ，50 ，54 ，57 ]。用户通常会被浏览器，电子邮件代理或AV产品更多地警告可执行文件的危险，而像PDF这样的文档由于受到静态文件的影响而受到的关注和审查要少得多，并且几乎没有什么危害

随着时间的推移，内容复杂性的增长为攻击者提供了更多的武器来发动强大的攻击，并且更灵活地隐藏恶意有效载荷（例如，加密，隐藏为图像，字体或Flash内容）并逃避检测。

      Why is the problem difficult?

鉴于文档阅读器越来越复杂以及库/系统组件的依赖性越来越大，攻击者的攻击面越来越大。新的漏洞继续被发现，Adobe Acrobat Reader（AAR）。AAR的普及及其大型攻F击面使其成为攻击者的首要攻击目标[ 25 ]，其次是浏览器和操作系统内核。

      What is the old technique?

近年来已经提出了许多解决方案来检测具有恶意有效载荷的文档。这些技术可以分为两大类：静态和动态分析。

静态分析，或基于签名的检测[ 14 ，27 ，31 ，33 ，34 ，36 ，46 ，52 ，59 ]，解析的恶意内容的指示，如shellcode的或相似性与已知的恶意软件样本文档和搜索。在另一方面，动态分析，或基于执行的检测[ 45 ，48 ，58 ]，运行的部分或整个文档和迹线的恶意行为，如脆弱API调用或返回导向编程（ROP）。

然而，最近的作品突出了这些解决方案的一些共同的缺点。Carmony *等*。[ 11 ]表明，在这些解决方案中使用的PDF解析器可能已经过分简化了关于PDF规范的假设，导致恶意负载的不完整提取和分析失败。也已经证明，基于机器学习的检测有可能以原理和自动的方式被回避[ 35 ，53 ，65 ]，许多解决方案只关注JavaScript部分，而忽视了与其他PDF组件在发起攻击方面的协同作用。因此，尽管现代AV产品支持PDF漏洞检测，但即使后者仅对现有漏洞进行微小修改，也不能迅速适应新颖的模糊技术[ 55 ]。由于缺乏攻击程序和运行时间跟踪，AV产品还会出现提供零日攻击保护的问题

      What is the new technique?

本文提出了一个新的视角maldoc检测：平台的多样性

具体来说，我们确定了操作系统设计和实现中可能导致行为分歧的八个因素，从系统调用语义（更明显）到堆对象元数据结构（更加微妙），并进一步展示如何阻止攻击者发现错误，利用错误或执行恶意活动。

我们进一步对P LAT P AL 进行原型系统收集平台多样性。P LAT P AL 挂接到Adobe Reader以跟踪内部PDF处理，并使用沙箱执行来捕获maldoc对主机系统的影响。比较不同平台上的执行跟踪，并且maldoc检测基于良性文档在不同平台上表现相同的观察结果，而maldoc在开发过程中表现不同。评估表明，P LAT P AL 在良性样本中不会产生虚假警报，在恶意样本中检测到各种行为差异，是一个可扩展和实用的解决方案

本文做出如下贡献：

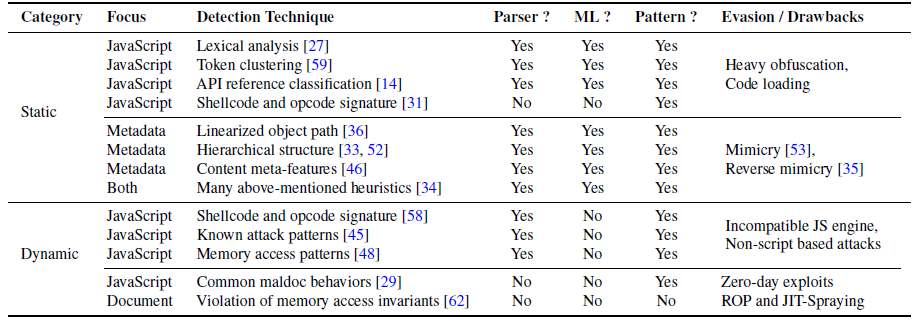
•    我们建议跨不同平台执行文档，并使用行为差异作为maldoc检测的指标。

•    我们对平台多样性进行深入分析和分类，并展示它们如何用于检测Maldoc攻击。

•    我们基于这些见解对P LAT P AL进行了原型设计。评估证明P LAT P AL 是可扩展的，不会引起虚假警报，并检测恶意样本中的各种行为差异。

我们计划开源P LAT P AL，以提示使用平台多样性进行Maldoc检测，并启动一个PDF maldoc扫描服务供公众使用。

现有的maldoc检测方法可以大致分为两类：1）动态分析，在特定的仪器环境中执行和检测恶意代码; 和2）静态分析，其中执行检测而不执行代码。表1 列出了现有方法的总结。



恶意PDF文档检测技术的分类。 该分类法部分基于系统调查报告[40]，2013年之后添加了作品以及摘要分析器，机器学习和模式依赖关系以及回避技术

在本文中，我们提出了一个Maldoc检测方案P LAT P AL ，用于分析不同平台（例如Windows或Macintosh（Mac））上恶意文档文件的行为差异。与依赖现有恶意软件样本构建启发式的静态和动态检测方案不同，P LAT P AL 基于一套完全不同的见解：1）良性文档在不同平台上表现相同（在一定水平上），而2 ）恶意文档在不同平台上启动漏洞时会导致不同的行为。

第一个假设可以通过打开许多使用跨平台的各种PDF功能的良性样本进行实证验证。为了支持第二个假设，我们调查了OS受到攻击时可能导致行为分歧的因素，并从系统调用语义（更明显）到堆对象元数据结构（更精细）等8个因素中找到了这些因素。我们进一步展示了如何利用它们阻止攻击者发现错误，利用错误或执行恶意活动。

P LAT P AL 基于这些见解。要检测文档是否具有恶意负载，P LAT P AL 将使用相同版本的AAR实例打开它，但在不同的操作系统上运行。P LAT P AL 在处理文档时记录AAR的运行时跟踪，并随后在各个平台上进行比较。执行痕迹和产出的共识表明文件的健康，而分歧则表示攻击。

虽然这个过程听起来简单直观，但是需要解决两个实际的问题，以使得P LAT P AL 工作：1）在不同的平台上，“行为”可能有什么不同？2）他们如何被普遍追踪？P LAT P AL 追踪并比较了两种类型的行为。内部行为包括AAR在PDF处理周期中执行的关键功能，如加载，解析，呈现和脚本执行。外部行为包括文件系统操作，网络活动和程序启动。这与典型的恶意软件分析工具，如杜鹃沙箱[ 44 ]。

值得强调的是，P LAT P AL 不应被视为当前的恶意软件分析工具（如Cuckoo [ 44 ]）的竞争对手，因为它们依赖于不同的假设。目前的工具在很大程度上依赖于OS范围的活动的黑名单（或白名单）的可用性，因此可以对样本的行为进行审查。这种方法适用于已知的恶意软件，但可能会在0天的PDF漏洞攻击中丧失优势。另一方面，P LAT P AL 不要求这样的列表起作用，而仅仅依赖于这样一个事实，即攻击者很难在Windows和Mac平台上以完全相同的方式制作一个利用AAR的恶意PDF 。

P LAT P AL 是针对使用PDF规范中的各种特征的1030个良性样本进行评估，并且报告没有差异，即**没有误报**。对于利用16个不同的CVE的320个maldoc样本的集合，P LAT P AL 可以检测其中的209个发散和另外34个样本使两个AAR实例崩溃。其余部分由于各种原因而未被发现，例如针对旧的和特定版本的AAR或未触发恶意活动。P LAT P AL 可以在不超过24秒的时间内完成文件的扫描，不需要手动驱动。

**但是，我们确实承认，鉴于PDF规范的复杂性，P LAT P AL 没有列举所有可能的平台探测技术。因此，P LAT P AL 可能会通过我们尚未发现的隐性渠道（例如，定时副通道）被回避。**

如下图为：内部PDF处理跟踪

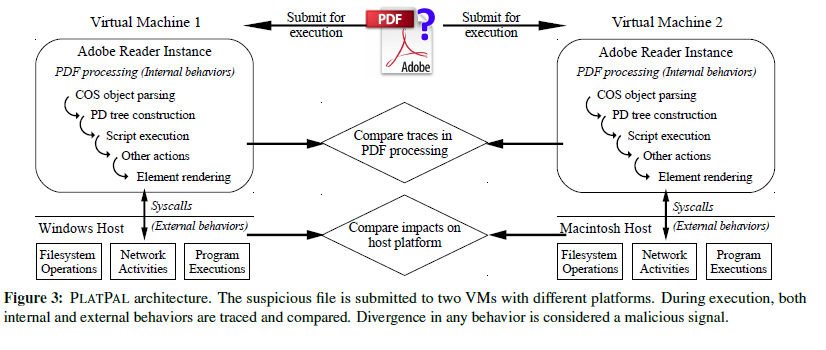


图3：P LAT P AL 架构 将可疑文件提交给两台具有不同平台的虚拟机。在执行过程中，内部和外部行为都被追踪和比较。任何行为的分歧都被视为恶意信号

以下是每个步骤中记录的跟踪摘要：

COS对象解析：P LAT P AL 输出COS对象（包括类型和内容）的解析结果。

PD树构造：P LAT P AL 输出PD树中每个具有类型和分层位置的PD组件。

脚本执行：P LAT P AL 输出每个执行的语句和相应的结果。

其他操作：P LAT P AL 输出执行此操作期间触发的每个回调，例如更改页面查看或访问的URL。

元素渲染：P LAT P AL 输出在渲染PDF元素期间触发的每个回调。

  评估

在本节中，验证了P LAT P AL 的基本假设：良性文档在不同平台上打开时表现相同，而在不同平台上进行开发时，maldoc表现出不同的表现。我们还根据完成分析所花费的总时间来评估P LAT P AL 的表现。

实验设置。实验在MacBook Pro（2016款）上进行，搭载Intel酷睿i7 2.9GHz CPU和16GB RAM，运行macOS Sierra。一个VM配置了Windows 7 Professional SP1，另一个VM配置了OSX Yosemite 10.10.1。每个虚拟机进一步配置了6个不同版本的AAR实例[[1]](file:///E:\\bdfirst\\2018\\2018Q1\\sec17-xu-meng.htm" \l "_ftn1" \o ")，如表2 所列。每个文档样本在执行一分钟后被强制关闭。

最后总结：

由于AAR的持续开发，maldoc检测已经成为一个紧迫的问题。对现有技术的调查显示，他们很容易受到最近的攻击，如解析器混淆和ML规避攻击。针对这一点，我们提出了一个新的视角：平台多样性，以及用于maldoc检测的原型P LAT P AL。P LAT P AL挂钩到AAR以跟踪内部PDF处理，还使用全面的动态分析来捕获maldoc对主机系统的外部影响。内部和外部的痕迹都进行了比较，唯一的启发式检测maldoc是基于这样的观察：良性文档在不同平台上表现相同，而maldoc在开发过程中表现不同，因为系统调用，内存管理等的多样化实现跨平台。这样的启发式算法不需要已知的maldoc样本来导出区分maldoc和良性文档的模式，这也使得P LAT P AL 能够在事先不知道攻击的情况下检测到零日攻击。评估显示P LAT P AL在良性样本中不会产生虚假警报，在恶意样本中检测到各种行为差异，是一个可扩展和实用的解决方案。