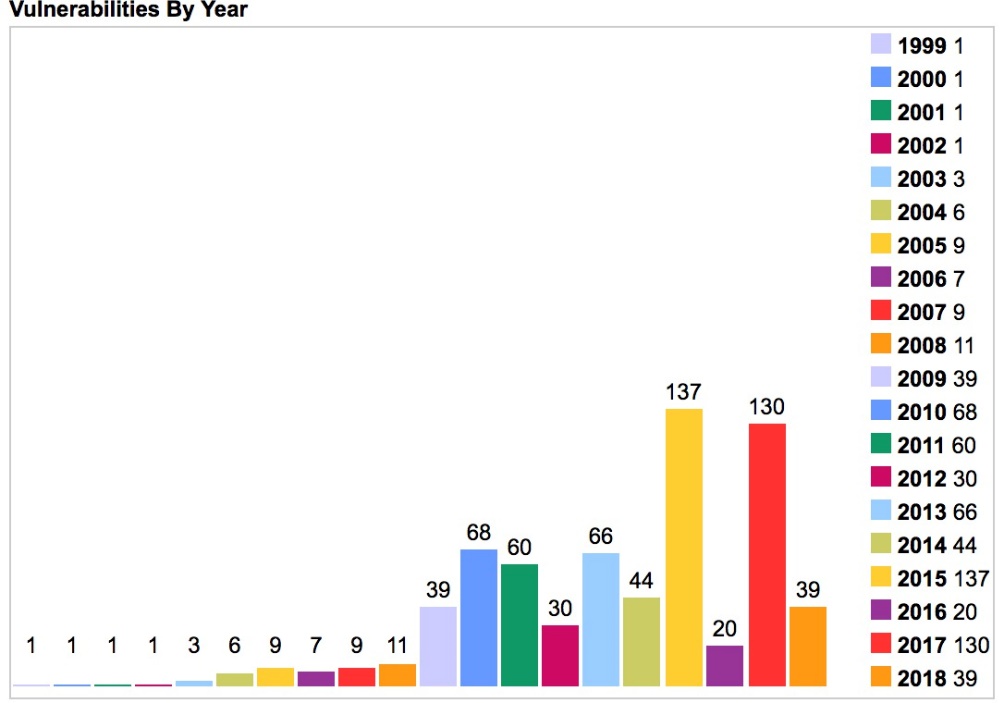
# 不容忽视的PDF恶意文件发展

随着时间的推移，PDF规格和样式都在改变，变得丰富。增加的脚本功能可以使文档与可执行文件几乎相同的方式工作，包括连接到Internet的能力，运行进程以及与其他文件/程序进行交互。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。由于企业和个人用户系统未采用最新的浏览器插件（例如，Adobe的 Flash Player 和 Acrobat Reader）修补程序进行更新，这些攻击得以取得成功。

在2012年，开创性的漏洞攻击工具Blackhole专门针对Java，PDF和Flash文件。2013年，通过网络传输的不可执行恶意软件主要是针对Adobe Reader和Microsoft Office应用程序的PDF和Flash文件。在2014年发现的24个0day中，有16个是针对Adobe Reader和Flash Player的）如图1 是近几年来Adobe Reader在CVE上公布的一些漏洞统计，由图可以看出，从2015年开始关于Adobe Reader发现的漏洞就在线性增长，就表明了针对PDF文档的攻击开始增多。

图1 Adobe Reader近几年CVE的变化图



在2017年的Symantec 的年度互联网安全威胁报告（Internet Security Threat Report，简称ISTR）中显示中国依然是垃圾邮件的重灾区，中国在Malicious activity by source 总排名中名列前茅，其中在Spam zombies的统计中更是远超其他国家，排在了最前面的位置。

如今检测PDF文件的方法有很多，主要的方法可以分为两类，动态检测和静态检测，纵然这有一定程度上的重合。动态方法需要把文档放在某一个特定探针环境（instrumented environment）中打开运行；而静态方法则无需打开运行仅通过静态特征即可完成检测。表1 列出了现有方法的总结



|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| 方法 | 分析重点 | 检测技术 | 作者 | 年份 | 外置解析器 | ML |
| 静态分析 | JavaScript | Lexical 分析 | Nedim Srndic | 2011 | Y | Y |
| JavaScript | Token 聚类 | Vatamanu et al. | 2012 | Y | Y |
| JavaScript | API 调用分类 | Igino Corona, Davide Maiorca | 2014 | Y | Y |
| JavaScript | Shellcode and opcode 签名 | Xun Lu | 2013 | N | N |
| Metadata | Linearized object path | Davide Maiorca et al. | 2012 | Y | Y |
| Metadata | 分层结构检测 | Srndic et al. | 2013 | Y | Y |
| Metadata | 基于内容和Metadata | Charles Smutz et al. | 2012 | Y | Y |
| Both | 基于结构和内容解析 | Maiorca et al. | 2015 | Y | Y |
| Both | 结合上述几种技术解析分类 | Maiorca et al. | 2016 | Y | Y |
| 动态分析 | JavaScript | Shellcode and opcode签名检测 | Zacharias Tzermias et al. | 2011 | Y | N |
| JavaScript | 已知的攻击模式 | Florian Schmitt et al. | 2012 | Y | N |
| JavaScript | 内存访问模式 | Srinivas Krishnan et al. | 2011 | Y | N |
| JavaScript | 常见 maldoc 行为分析 | Liu et al. | 2014 | N | N |
| JavaScript | 独立平台的tap point标识技术 | Curtis Carmony et al. | 2016 | N | N |
| 文档类型 | 异常内存访问约束变量 | Carsten Willems et al. | 2012 | N | N |
|  | 平台多样性 | 系统平台多样性利用 | Xu et al. | 2017 | Y | N |

由表1可见，静态分析一般聚焦于JavaScript本身或使用结构进行分析。代表性的检测技术有基于Shellcode和OPCode签名的MPScan、基于结构与内容两者的分类。动态分析技术一般聚焦于提取嵌入在PDF文档中的JavaScript代码，再通过实际试运行这些代码片段，检测出恶意行为。

在以上工作中，有12/15的工作使用外置的PDF解析器，这使得外置PDF解析器的健壮性成为研究焦点。这是因为外置PDF解析器一般设计和实现均较为简单，恶意样本经少量变异即能轻易逃逸此类解析器。这种攻击在Carmony et al.[20]的工作中被称为解析器混淆性攻击（Parser Confusion Attacks）。

从表1可知，机器学习一般并不适合于动态分析，而几乎所有的静态分析工作，都在某种程度上使用了机器学习的技术。这部分的典型工作有PDFrate[24]、PDF Malware Slayer[11]等。这些工作均声称分类器能在低功耗环境下达到很高的检测精度，但对模型本身的安全性，恶意样本逃逸分类器等对抗性学习的研究内容却鲜有提及。这种攻击在Xu et al[14]的工作中被提出，作者通过构建一个能自动生成恶意样本变种的框架，使得在每一次的样本变异迭代过程中，原始输入样本集会经过某种遗传算法把良性PDF对象加入到恶意样本的文件结构中。在不断的变异过程中系统一方面需保持恶意样本的恶意本来面目不变，另一方面则需要达到迷惑分类器的目的。这种专门针对分类器的攻击及其框架被称为分类器逃逸攻击（Classifier Evasion Attack）

一些有趣的研究性问题是这样的：是否可以通过不改变原文件的恶意属性，用增加良性行为部分的方法，以成功逃逸分类器的检测？是否可以通过不改变原文件的善意行为，用增加恶意行为的方法，使这些样本通过隐藏方式，成功逃逸分类器。Srndic et al. [4]的工作从恶意样本着手，聚焦于前一种攻击，我们把其称为模拟性攻击（Mimicry Attack）；而Maiorca et al.[10]的工作从良性样本着手，我们在这里称其为反向模拟性攻击（Reverse Mimicry Attacks）。

综上所述：对于外置PDF解析器，现有攻击手段是解析器混淆攻击（Parser-Confusion Attacks）；对于机器学习模型，现有的攻击手段是自动化分类器逃逸攻击（Automatic Classifier Evasion Attacks）；对于假设性的“可检测的分辨力”(Detectable Discrepancy)，现有攻击手段为模拟和反向模拟攻击（Mimicry and Reverse Mimicry）。这些攻击手段对于模型本身的安全提出了很大挑战，在我们的工作中不仅生成了一个准确度高的模型，并且在模型本身的安全也有所建树。