1. **相关工作**

如今PDF恶意文档检测技术可大约分为两大类：静态分析和动态分析。静态分析无需使样本运行，仅通过文件头部格式，二进制层N-gram等静态模式，即可对目标样本进行预测；动态分析，则通过使目标样本运行于受控环境内，以此捕捉其恶意行为。一般而言，静态分析的优点是速度易于部署更新，动态分析速度慢，消耗资源多，但人员参与精度很高。两种方法在业界均有大量成功应用案例，更高级的解决方案如可以把静态和动态分析结合，典型的工作如Maiorca et al. [9]。表1基于之前的研究列出了对现有方法的总结。

表1 相关工作对比

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| 方法 | 分析重点 | 检测技术 | 工作 | 年份 | 外置解析器 | ML | 可检测的不同性 |
| 静态分析 | JavaScript | Lexical 分析 [5] | PJScan | 2011 | Y | Y | Y |
| JavaScript | Token 聚类 [12] | Vatamanu et al. | 2012 | Y | Y | Y |
| JavaScript | API 调用分类 [7] | Lux0r | 2014 | Y | Y | Y |
| JavaScript | Shellcode and opcode 签名 [13] | MPScan | 2013 | N | N | N |
| Metadata | Linearized object path [11] | PDF Malware Slayer | 2012 | Y | Y | Y |
| Metadata | 分层结构检测 [8, 1] | Srndic et al. | 2013 | Y | Y | Y |
| Metadata | 基于内容和Metadata [24] | PDFrate | 2012 | Y | Y | Y |
| Both | 基于结构和内容解析 [8] | Maiorca et al. | 2015 | Y | Y | Y |
| Both | 结合上述几种技术解析分类 [9] | Maiorca et al. | 2016 | Y | Y | Y |
| 动态分析 | JavaScript | Shellcode and opcode签名检测 [15] | MDScan | 2011 | Y | N | N |
| JavaScript | 已知的攻击模式 [16] | PDF Scrutinizer | 2012 | Y | N | N |
| JavaScript | 内存访问模式 [17] | ShellOS | 2011 | Y | N | Y |
| JavaScript | 常见 maldoc 行为分析 [18] | Liu et al. | 2014 | N | N | Y |
| JavaScript | 独立平台的tap point标识技术[2] | tap point | 2016 | N | N | Y |
| 文档类型 | 异常内存访问约束变量 [19] | CWXDetector | 2012 | N | N | N |
|  | 平台多样性 | 系统平台多样性利用[8] | PlatPal | 2017 | Y | N | Y |

由表1可见，静态分析一般聚焦于JavaScript本身或使用Metadata进行分析。代表性的检测技术有基于Shellcode和OPCode签名的MPScan[13]、基于结构与内容两者的分类[9]。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。这类工作的代表有基于maldoc的行为分析[8]和基于平台多样性的PlatPal[21]等。

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）

在上述工作中，有11/15的工作有如下假设：即恶意样本和良性样本间需具有明显特征分辨能力或分界线。换句话说，我们假设恶意样本和良性样本有一超平面能把其很好地在高纬度特征空间中分开。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。