# Automatically Evading Classifiers

A Case Study on PDF Malware Classifiers

## Abstract

Machine learning is widely used to develop classiﬁers for security tasks. However, the robustness of these methods against motivated adversaries is uncertain. In this work, we propose a generic method to evaluate the robustness of classiﬁers under attack. The key idea is to stochastically manipulate a malicious sample to ﬁnd a variant that preserves the malicious behavior but is classiﬁed as benign by the classiﬁer. We present a general approach to search for evasive variants and report on results from experiments using our techniques against two PDF malware classiﬁers, PDFrate and Hidost. Our method is able to automatically ﬁnd evasive variants for both classiﬁers for all of the 500 malicious seeds in our study. Our results suggest a general method for evaluating classiﬁers used in security applications, and raise serious doubts about the effectiveness of classiﬁers based on superﬁcial features in the presence of adversaries.

摘要 - 机器学习被广泛用于开发安全任务的分类器。但是，这些方法对于有动机的对手的稳健性是不确定的。在这项工作中，我们提出了一种通用的方法来评估受到攻击的分类器的鲁棒性。关键的想法是随机操纵一个恶意样本来找到一个保留恶意行为的变体，但被分类器分类为良性。我们提出了一个通用的方法来搜索回避变体，并使用我们针对两个PDF恶意软件分类器PDFrate和Hidost的技术进行实验。我们的方法能够为我们研究中的所有500个恶意种子的两个分类器自动找到规避的变体。我们的研究结果提出了评估安全应用中使用的分类器的一般方法，并且在存在对手的情况下，基于超表面特征的分类器的有效性产生了严重的怀疑。

我们提出了一个通用的方法来搜索回避变体，并使用我们针对两个PDF恶意软件分类器PDFrate和Hidost的技术进行实验。我们的方法能够为我们研究中的所有500个恶意种子的两个分类器自动找到规避的变体。

我们在实验中使用了一台典型的台式机（Intel Core i7-2600 CPU @ 3.40GHz和32GB物理机，内存运行64位Ubuntu 14.04服务器）。Cuckoo沙箱由16个运行Windows XP SP3 32位和Adobe Acrobat Reader 8.1.1的虚拟机实例组成。使用我们的方法找到规避样本所需的资源是现成的

## I. NTRODUCTION

贡献

1.我们提出了一种通用的方法来自动找出目标分类器的回避变体。该方法不依赖于任何特定的分类算法或假设详细的特征提取知识，但只需要生成的变种的分类评分反馈和分类器使用的可能特征的粗略知识（第二节）

2.我们实现了一个原型系统，该系统能够自动发现可以逃避基于结构特征的PDF恶意软件分类的变体。这涉及设计在PDF文件上执行随机操作的操作符，确定生成的变体是否保留恶意的预言器，在进化过程中促进有希望的变体的选择机制以及每个目标分类器的适合度函数（第四节）。

3.我们评估了我们的系统的有效性，以回避两个最近的PDF恶意软件分类器：PDFrate [25]和Hidost [28]，这是一个以明确拒绝回避企图为目标的分类器。我们的系统在500个恶意软件样本种子的实验中发现了两个分类器的回避变体，实现了100％的成功率。对每个分类器的特征空间中发现的规避变体的分析表明，分类中使用的许多非鲁棒特征有助于逃避攻击（第V和第VI部分）。我们在第二部分提供机器学习分类器的背景知识，第三部分提供有关PDF恶意软件的背景知识。第八节讨论了有关逃逸攻击的工作。

## II. 概览OVERVIEW

### Machine Learning Classifiers

训练机器学习系统的努力始于特征提取。由于大多数机器学习算法不能在高度结构化的数据上操作，所以数据样本通常在特别设计的特征空间中表示。例如，恶意软件分类器可以将文件大小和函数调用跟踪提取为特征。每个特征都是特征空间中的一个维度;因此，每个样本都被表示为一个向量。当分类算法的特征数量太大时，可以执行特征选择的额外步骤以减少特征的数量。

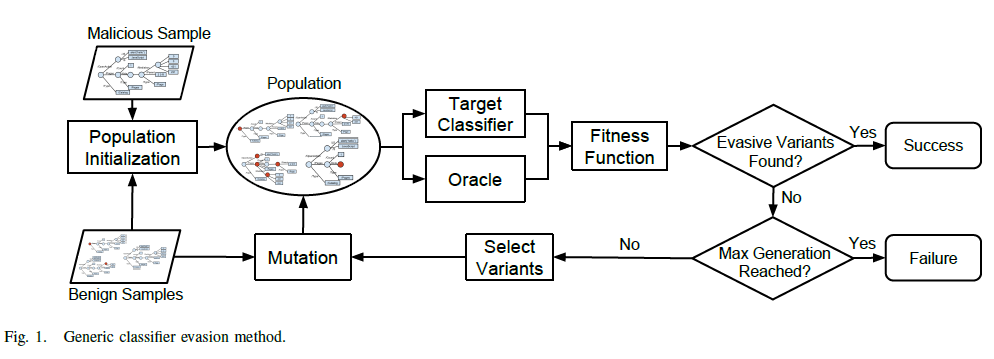
恶意软件分类器通常使用静态属性来预测恶意，如文件结构（the ﬁle structure），文件大小，元数据（metadata），令牌克数（grams of tokens）或系统调用（system calls）。尽管这种方法在验证测试中经常达到很高的准确性，但是分类器可能学习的是属性是训练数据的超痕迹，而不是与恶意软件本身相关的属性。这是因为训练数据中的恶意软件样本可能与良性样本在许多方面有所不同，而这些样本对其恶意行为并不重要。

### Threat Model威胁模型

假设攻击者可以对目标分类器进行黑箱访问，并且可以向该分类器提交许多变体。对于每个提交的变体，攻击者学习其分类分数。分类评分是一个数字（通常是介于0和1之间的实数），表示分类器对恶意的预测，其中高于某个阈值（比如0.5）的值被认为是恶意的，而分类评分低的样本被认为是良性的。我们不假设攻击者有任何有关分类器的内部信息，只是它可以用它作为黑盒来输出输入样本的分类得分。我们假设分类运算符不会将分类器适配到提交的变体（如果攻击者能够访问分类器，则必须是这种情况）。

### C. Finding Evasive Samples找到逃逸样本

我们的方法使用**遗传编程技术**来执行对可能样本空间的定向搜索，以找到逃避分类器的样本，同时保留所需的恶意行为。



2图1.通用分类器逃避方法。包括修复遗留软件错误[17]，软件逆向工程[13]和软件重新设计[23]。方法。我们的过程如图1所示。它从一个展现恶意行为的种子样本开始，被目标分类器分类为恶意行为。我们的方法旨在发现一个避开恶意行为但被目标分类器错误分类为良性的回避样本。

首先，我们通过对恶意种子执行随机操作来初始化变体种群。

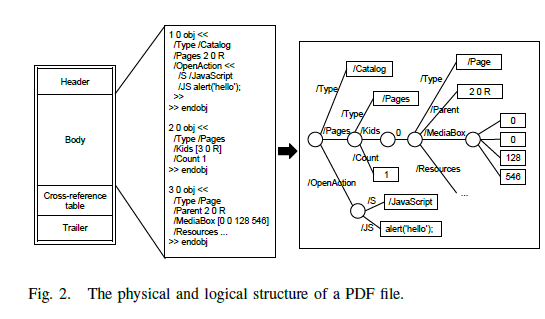
然后，每个变体都由一个目标分类器和一个预言器进行评估。

接下来，所选择的变体被变异算子随机操纵以产生下一代种群。这个过程一直持续到找到一个回避的样本或达到了一个世代的阈值。

## III. PDF MALWARE AND CLASSIFIERS

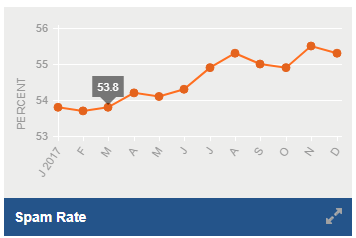
### PDF Malware

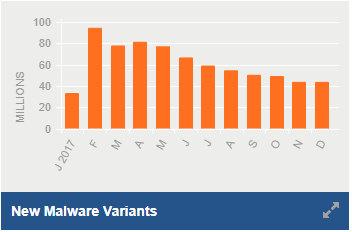
一个PDF文件由四部分组成：标题，正文，交叉引用表（CRT）和预告片。标题包含PDF幻数和格式版本指示符。正文是一组包含文件内容的PDF对象，而CRT则是索引正文中的对象。预告片指定如何找到CRT和其他特殊对象，如根对象。因此，PDF阅读器通常从文件末尾开始阅读PDF以提高效率。



主体是最重要的。它包含八种基本类型的对象，即布尔值namely Booleans，数字numbers，字符串strings，名称names，数组arrays，字典dictionaries，流streams和空对象the null objects。如图2右侧所示。这个树状结构非常适合**基因编程技术**，因为它很容易改变和移动子树来生成新的变体。

根据最近的互联网安全威胁报告[30]，2014年，PDF在鱼叉钓鱼电子邮件中排名前7位。我们预计PDF恶意软件攻击将继续存在，因为Acrobat Reader中的128个新漏洞已经在CVE到2015年（到12月8日）为止，几乎是2014年的三倍[8]。（https://www.symantec.com/security\_response/publications/monthlythreatreport.jsp）





PDF恶意软件通常包含利用特定PDF阅读器（通常为Adobe Acrobat）的漏洞的JavaScript对象或其他对象的漏洞。PDF文件的物理和逻辑结构PDF恶意软件也可能在流对象中携带其他编码的有效载荷，这些对象将在利用后触发[25]

### Target Classifiers目标分类器

由于不是所有的PDF恶意软件都涉及嵌入式JavaScript，PDF恶意软件作者已经发现隐藏JavaScript代码的许多技巧[24]，最近的PDF恶意软件分类器已经集中在PDF文件的结构特征上。在这项工作中，我们针对最先进的基于结构特征的分类器。

PDFrate

PDFrate是一个随机森林分类器，它使用由大量决策树组成的集合学习模型，这些决策树旨在减少预测方差。利用训练数据的随机子集和特征的随机子集，对每个决策树进行训练以使其训练子集上的预测误差最小化。在训练之后，PDFrate的输出分数是输出“恶意”的树的分数，范围从0到1.阈值典型地为0.5，尽管PDFrate作者声称将阈值从0.2调整到0.8对准确度的影响不大因为大多数样本的得分非常接近0或1。

除了对象关键字之外，PDFrate还使用PDF元数据和对象的几个属性作为分类特征，PDF元数据包括作者，标题和创建日期。对象属性包括位置，计数和长度。

我们在这项工作中使用的是由Nedim Srndic和Pavel Laskov实施的模仿PDFrate进行恶意软件逃避实验的PDFrate命名为Mimicus [27]的开源重新实现[29]。 Mimicus接受了135个文件记录的PDFrate特征和与PDFrate.1相同的培训.1 Mimicus已被证明具有与PDFrate [29]几乎相同的分类性能。

Hidost

Hidost是一个支持向量机（SVM）分类模型。支持向量机（SVM）是一种最佳的边缘分类器，它试图找到一些支持向量（数据点），用高维空间的超平面来分隔两类的所有数据点。使用内核技巧，可以将其扩展为非线性分类器来处理更复杂的分类问题。 Hidost使用径向基函数（RBF）核将数据点映射到无限维空间。在测试时间，数据点到超平面的（正或负）距离被输出作为预测结果。积极的距离被认为是恶意的，消极的是良性的。

Hidost使用对象的结构路径作为分类特征。

## IV. EVADING PDF MALWARE CLASSIFIERS

所提出的方法可以应用于任何安全分类器，尽管它的有效性取决于能够找到好的遗传编程算子来有效地搜索特征空间，并且有一个合适的适合度函数来指导搜索。在本节中，我们将展示如何实例化我们的设计，以找到避开PDF的恶意软件。

### PDF Parser and Repacker 分析和重新包装

第一步是将PDF文件解析为树形表示。我们还需要从树形表示中重新生成一个PDF文件，在它被操作后产生一个新的变体。为此，我们使用pdfrw [21]，这是一个基于python的开源库，用于将PDF文件解析为树状结构，并将该结构序列化为输出PDF文件。

为了避免由于PDF解析问题而丢失太多样本，我们修改了pdfrw来放宽其语法检查。这显着提高了重新包装PDF恶意软件样本的成功率。pdfrw的修改版本可在https://github.com/mzweilin/pdfrw上找到。

在我们修改的pdfrw中，我们忽略了几个可能被破坏的，畸形的或误导性的辅助元素。PDF原始字节中的*EOF*标记被忽略; 相反，解析器读取文件的所有字节。该*交叉引用表*被忽略; 相反，它直接解析*身体中的*对象而没有任何索引。流长度指标被忽略; 相反，解析器使用*endstream*检测流的长度令牌。解析字典时，未配对的键或值也会被忽略。忽略这些辅助元素会显着降低解析效率，因此仅适用于重新包装种子恶意软件样本。所有的种子都用正确的辅助元素重新打包，以便以后进行高效解析。另外，我们增加了对解析恶意软件样本中存在的空对象的支持。字典数据结构被修改以使得能够从种子复制变体的深拷贝。

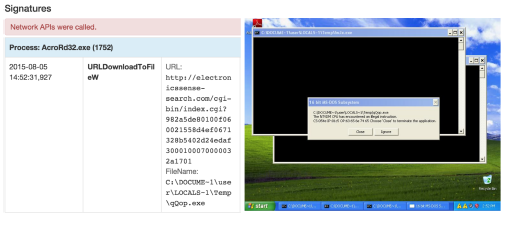
### Genetic Operators遗传算法

我们需要通过在该级别上操作PDF文件来生成变体。（如果我们的目标是基于JavaScript的分类器，我们将需要通过操纵嵌入的JavaScript代码来生成变体）。由于可能的静态特征的数量有限，我们认为假设攻击者知道操纵水平。

我们使用生物进化中的*突变*计算类似物来生成回避性的PDF恶意软件变体。变异算子以低概率改变PDF文件的树状结构中的任何对象。一个对象被变异概率给定的*突变率*，通常是一个小于0.5的数字。突变可以是*删除*（*删除*对象），*插入*（*插入*另一个对象后）或*替换*（此对象替换为其他对象）。

我们在这些选项中选择均匀的随机概率。在插入或替换的情况下，还从由良性PDF分割的大量对象中随机地均匀地选择第二对象。外部基因组有助于产生更多样化的人群。

另一个着名的算子，*交叉*，常用于遗传算法，没有在这项工作中使用。我们找到了



### Oracle

我们需要一个oracle来确定一个变种是否保留了种子的恶意行为。

### D. Fitness Function

### E. Selection选择

根据Wepawet的结果，这500个恶意软件样本利用Acrobat Readers中的两个不同漏洞：其中333个使用CVE-2007-5659中报告的多个缓冲区溢出漏洞，另外167个使用CVE-2009-0927报告的基于堆栈的缓冲区溢出漏洞。这两个漏洞都可以被利用来执行任意代码。总之，500个样本中的有效载荷访问255个不同的主机，从互联网上下载额外的恶意软件。

着眼于恶意软件样本的网络行为，我们可以提取由Cuckoo报告的各种网络行为作为签名，如DNS查询，HTTP URL请求和网络目的地

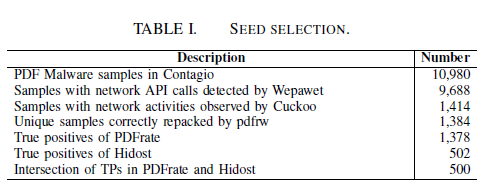
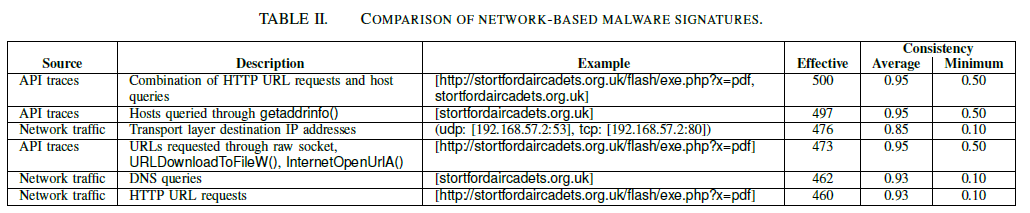
### F．Trace Collection and Replay跟踪采集与回收

## V. EXPERIMENT实验

### Dataset and Experiment Setup

Malicious PDF Dataset

Table I summarizes the sample selection procedure.



## VI. RESULTS

### A．PDFrate

### B. Hidost