# 机器学习- 恶意PDF文件特征提取与识别

关键字：

机器学习 ，Python ，PDF ，特征提取

## 摘要ABSTRACT

在这个安全问题发展频繁的时代，恶意PDF文件在实践中仍然是对大多数计算机用户的真正威胁。尽管Adobe和其他供应商发布了一系列安全补丁，但许多用户仍然在其计算机上安装了易受攻击的客户端软件。此外，PDF格式的表现力使得攻击者能够很少地努力逃避检测。除了传统的防病毒产品之外，这些防病毒产品始终是攻击者的一大步，很少有可以部署用于保护最终用户系统的方法。

在本次实验中，我们利用 80K 样本集，依据恶意和良性PDF文件的结构属性的本质区别，参考< Detection of Malicious PDF Files Based on Hierarchical Document Structure》此论文中的一些特性，对现有的样本进行分析，特征提取，利用机器学习来对文件进行训练和分类，经过多次测试和训练，分类效果均维持在 98%，

## 1．简介INTRODUCTION

## 2．相关工作

### 2.1 收集数据集

类型：

PDF文件被分类为良性或恶意，恶意进一步分为两类：normalPDF和MalPDF。

资源

1. 数据集：

Normal sample number : 897

Malware sample number : 83442

在做训练和推测的时候，随机抽取其中的部分做训练和测试做dataset

1. 操作：这些恶意样本文件是从virusshare 和蓝盾集群收集下载下来的，正常样本是通过爬虫抓取的 。

2. 了解PDF文件结构与恶意文件架构

PDF文件格式包含以下4个部分：

**文件头——指明了该文件所遵从的PDF规范的版本号，它出现在PDF文件的第一行。**

**文件体——又称对象集合，PDF文件的主要部分，由一系列对象组成。**

**交叉引用表——对对象进行随机存取而设立的一个间接对象的地址索引表。（实际以偏移+索引的方式储存对象地址，下文会提及）**

**文件尾——声明了交叉引用表的地址，即指明了文件体的根对象（Catalog），从而能够找到PDF文件中各个对象体的位置，达到随机访问。另外还保存了PDF文件的加密等安全信息。**

**PDF文件格式图示：**



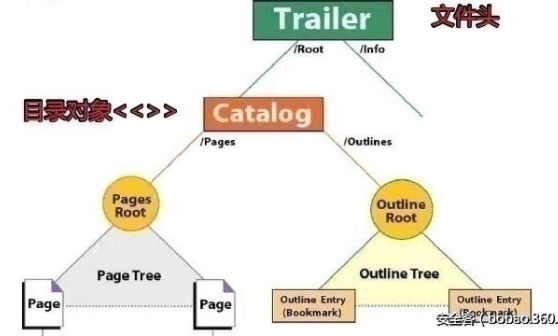
PDF文件的逻辑结构

本段主要介绍PDF文件体的读取方式。

作为一种结构化的文件格式，一个PDF文档是由一些称为“对象”的模块组成的。每个对象都有数字标号，这样的话可以这些对象就可以被其他的对象所引用。这些对象不需要按照顺序出现在PDF文档里面，出现的顺序可以是任意的，比如一个PDF文件有3页，第3页可以出现在第1页以前，对象按照顺序出现唯一的好处就是能够增加文件的可读性，对象的信息以偏移+索引的形式保存在交叉引用表内。

文件尾说明了根对象的对象号，并且说明交叉引用表的位置，通过对交叉引用表的查询可以找到目录对象(Catalog)。这个目录对象是该PDF文档的根对象，包含PDF文档的大纲(outline)和页面组对象（pages）引用。大纲对象是指PDF文件的书签树；页面组对象（pages）包含该文件的页面数，各个页面对象(page)的对象号。

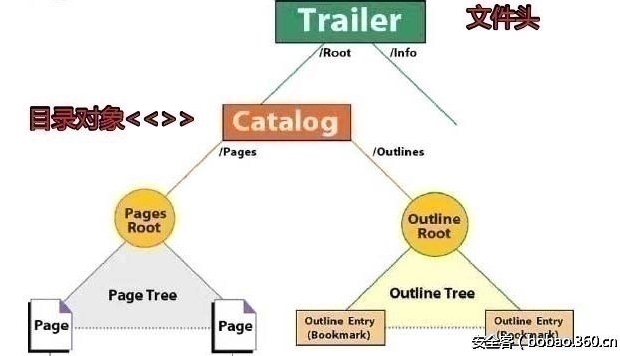
PDF的层级结构图示：



页面（page）对象为PDF中最重要的对象，包含如何显示该页面的信息，例如使用的字体，包含的内容（文字，图片等），页面的大小。里面的信息可以直接给出，当然里面的子项更多的是对其他对象的引用，真正的信息存放在其他对象里面。页面中包含的信息是包含在一个称为流（stream）的对象里，这个流的长度（字节数）必须直接给出或指向另外一个对象（包含一个整数值，表明这个流的长度）。

可见stream流对象我们恶代分析需要获取的重点。

页面信息图示：



理解了上面的内容之后，我们可以得出针对恶代分析的PDF文件的大致解析思路：

文件去除混淆（下详）

↓

查找关键字

↓

根据关键字获得可能存在恶意代码的流或者buffer

↓

解码流获取恶意代码

当然，也可以采取针对PDF层级结构的文档解析方式，见仁见智，因人而异。

3．通过机器学习的方法，训练并得出预测模型

## 3．特征描述DATA & FEATURE DESCRIPTION

每一个PDF文件都包含有前7个字段，也有可能不包含strea和endstream。据说有一些ODF文件没有xref或则trailer，但是这种情况比较少见。如果一个PDF文件没有xref或者trailer关键字段，那么可以确定它不是恶意的PDF文件。  
  
/xref 交叉引用表，描述每个间接对象的编号、版本和绝对的文件位置。PDF文档中的第一个索引必须从版本65535的0号对象开始，标识符/xref后面的第一个数字是第一个间接对象（即0号对象）的编号，第二个数字是/xref（交叉引用表）的大小。  
  
/Page指明PDF文件的页数，大多数恶意PDF文件仅仅只有一页  
  
/Encrypt指明PDF文件有数字水印或者是被加密过的。  
  
/ObjStm是object streams的数量。object streams是一个可以包含其他Object对象的数据流对象。  
  
/JS与/JavaScript指明PDF文件中含嵌有JavaScript代码。通常恶意的PDF文件都嵌套有JavaScript代码，这里一般都是利用JavaScript的解析漏洞或者使用JavaScript来实现堆喷射（heap spray），也有很多正常的PDF文件里会含有JavaScript代码  
  
/AA、/OpenAction和/AcroForm指明当查看PDF文件或者PDF的某页时会有动作随其执行，几乎所有嵌有JavaScript代码的恶意PDF文件都有自动执行JavaScript代码的动作(action)。如果一个PDF文件包含有/AA或/OpenAction自动执行动作的关键字段，而且含有JavaScript代码，那么这个PDF文件就极有可能是恶意的PDF文件  
  
/URI 如果你要在PDF文件中执行打开网页的动作就需要这个关键字段  
  
/Filter 一般为FlateDecode则是使用了Zlib压缩解压缩算法。  
  
/JBIG2Decode指明PDF文件使用了JBIG2压缩。虽然JBIG2压缩本身可能会有漏洞（CVE-2010-1297）。但/JBIG2Decode关键字段并不能说明PDF文件是否可疑  
  
/RichMedia Flash文件  
  
/Launch执行动作(action)数量

## 4. 分类评估方法CLASSIFICATION METHODOLOGY

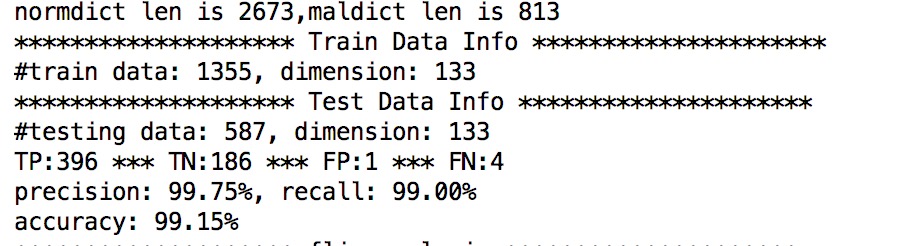
### 4.1 Feature Extraction特征提取

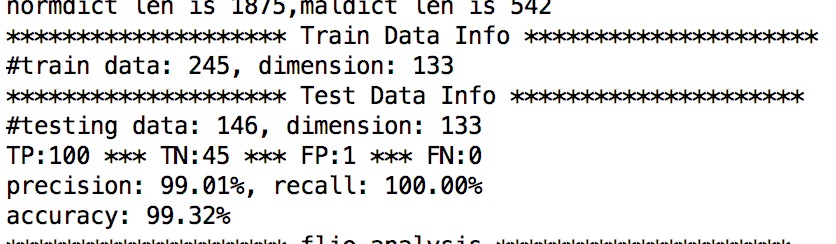
|  |  |
| --- | --- |
| xref | ('XrefSection', 2) |
|  | ('Xref\_stream', 0) |
|  | ('Xref\_size', 0) |
|  | ('Xref\_offset', 0) |
|  | ('Xref\_errors', 0) |
|  | ('Xref\_bytesPerFisId', 0) |
|  | ('Versions\_num', 3) |
|  | ('version0\_Xref Streams', 1) |
|  | ('version0\_Vulns', 0) |
|  | ('version0\_Streams', 1) |
|  | ('version0\_stream\_min\_1', 313) |
|  | ('version0\_stream\_min\_0', 0) |
|  | ('version0\_Objects\_JS\_num', 0) |
|  | ('version0\_Object Streams', 0) |
|  | ('version0\_obj\_min', 72) |
|  | ('version0\_obj\_average', 192) |
|  | ('version0\_obj\_10\_9', 0) |
|  | ('version0\_obj\_10\_8', 0) |
|  | ('version0\_obj\_10\_7', 0) |
|  | ('version0\_obj\_10\_6', 0) |
|  | ('version0\_obj\_10\_5', 0) |
|  | ('version0\_obj\_10\_4', 0) |
|  | ('version0\_obj\_10\_3', 0) |
|  | ('version0\_obj\_10\_2', 0) |
|  | ('version0\_obj\_10\_1', 72) |
|  | ('version0\_obj\_10\_0', 313) |
|  | ('version0\_Info', 9) |
|  | ('version0\_Events\_num', 0) |
|  | ('version0\_Events\_Names', 0) |
|  | ('version0\_Events\_AA', 0) |
|  | ('version0\_Encoded\_num', 1) |
|  | ('version0\_Elements\_EmbeddedFiles', 0) |
|  | ('version0\_elements', 0) |
|  | ('version0\_Decoding Errors', 0) |
|  | ('version0\_Compressd\_obj', 0) |
|  | ('version0\_Catalog', '11') |
|  | ('version0\_Actions\_num', 0) |
|  | ('version0\_Actions\_JS', 0) |
|  | ('version0\_Actions\_javascript', 0) |
|  | ('version', '1.5') |
|  | ('update', 2) |
|  | ('tree\_len', 41) |
|  | ('trailer\_num', 2) |
|  | ('subsections\_size', 0) |
|  | ('subsections\_offset', 0) |
|  | ('subsections\_numObjects', 1) |
|  | ('subsections\_firstObject', 2) |
|  | ('subsections\_errors', 0) |
|  | ('subsections\_entries', 1) |
|  | ('stream\_num', 28) |
|  | ('object\_num', 71) |
|  | ('numEncodedStreams', 26) |
|  | ('Metadata\_len', 4) |
|  | ('meta\_producer\_len', 33) |
|  | ('meta\_creator\_len', 26) |
|  | ('meta\_cration\_len', 23) |
|  | ('meta\_author\_len', 13) |
|  | ('Linearized', 1) |
|  | ('len\_URLs', 0) |
|  | ('JS\_MODULE', 1) |
|  | ('JS\_count', 0) |
|  | ('Javascript\_count', 0) |
|  | ('header\_offset', 0) |
|  | ('font\_count', 0) |
|  | ('file\_size', 136958) |
|  | ('error', 0) |
|  | ('Encrypted', 1) |
|  | ('comments', 0) |
|  | ('Binary', 1) |

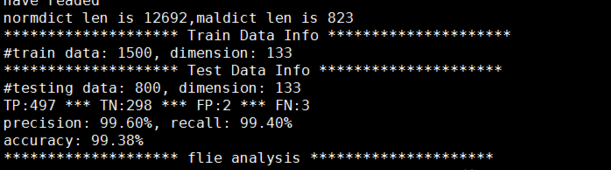
### 4.2 算法选择

## 5. PERFORMANCE EVALUATION性能评估

### 5.1 Classification & Detection Performance







## 6.总结

## 7.前景展望

## 8.参考文献

1．PDF格式详解  
C:\Users\Yonah\AppData\Local\Temp\%W@GJ$ACOF(TYDYECOKVDYB.pnghttp://blog.csdn.net/bobob/article/details/751381   
  
2.PDF Tools  
C:\Users\Yonah\AppData\Local\Temp\%W@GJ$ACOF(TYDYECOKVDYB.pnghttps://blog.didierstevens.com/programs/pdf-tools/   
  
3.恶意PDF文件解析思路  
C:\Users\Yonah\AppData\Local\Temp\%W@GJ$ACOF(TYDYECOKVDYB.pnghttps://wenku.baidu.com/view/0c05dd996529647d27285224.html

4.Malicious PDF Detection using Metadata and Structural Features

https://cs.gmu.edu/~astavrou/research/Malicious\_PDF\_Detection\_ACSAC\_12.pdf