# 基于机器学习的网页恶意代码检测方法*

李 洋[1]，刘 飚[2]，封化民[1,2]

（1.西安电子科技大学 通信工程学院，西安 710071；
2.北京电子科技学院，北京 100070）

**摘 要**：网络中大量的恶意网页已经成为网络用户的主要安全威胁。本文提出了一种基于机器学习分类器的网页恶意JavaScript代码分析方法。通过对训练样本训练学习，建立分类模型，最后对测试样本检测。实验表明，本方法能够有效的检测出大部分恶意网页JavaScript代码，检测准确率达到88.5%

**关键词**：恶意网页代码；JavaScript；特征提取
**中图分类号**：TP393.081 **文献标识码**：A **文章编号**：1672-464X(2012)04-36-06

## Malicious Web Pages Detection Based on Machine Learning

LI Yang[1], LIU Biao[2], FENG Hua-min[1,2]

(1. School of Telecommunication Engineering, Xidian University, Xi'an Shanxi 710071, China;
2. Beijing Electronic Science and Technology Institution, Beijing 100070, China)

**Abstract**: A large number of malicious website in the network has become a major security threat of network users. This paper puts forward a kind of malicious JavaScript code analysis method based on machine learning classifier. Through the study of the training sample training, establish the classification model, finally, the detectin is tested on soonples. The experimental results show that this method can more effectively detect the most malicious JavaScript code, accuracy up to 88.5%.

**Keywords**: malicious web page; javascript; feature extraction

## 引 言

近年来，随着互联网的日益普及，越来越多的应用提供基于WEB的服务，已经出现了浏览器操作系统的趋势，浏览器及浏览器插件渐渐地成为了主要攻击的对象，浏览器及其插件的漏洞取代了利用操作系统及应用程序漏洞，网页逐渐成为恶意代码传播或攻击的主要渠道。恶意网页指的是包含恶意内容以使得病毒、木马等可借其进行传播或攻击的网页，包含的恶意内容也被称为网页木马，其并非传统意义上的木马，而是以网页为介质进行传播或攻击的恶意代码，一般以JavaScript，VBScript

* 本文得到国家自然科学基金项目“基于多模态特征的多媒体语义分析关键理论与技术研究(NO. 60972139)”和北京市自然科学基金项目“基于网络多媒体信息语义的网络舆情分析研究(NO. 4092041)”的资助。

等脚本语言编写，包含在网页之中，通过各种方式进行代码混淆以逃避检测，在网页中插入恶意内容的行为也被称为“网页挂马”。

网页恶意代码通过利用用户的浏览器或插件中的漏洞，在用户毫不知情的情况下，下载和运行恶意软件，如广告软件、木马和病毒等。正常网页也可能被植入恶意代码，所以即使用户访问一些看似正常的网站，也有可能受到这类恶意代码的攻击。由于网页恶意代码大量使用了代码混淆技术，传统的反病毒软件的漏报率很高，这也导致越来越多的攻击者使用网页恶意代码来传播恶意软件。恶意网页检测方法通常可以分为静态检测（基于网页内容或网址）和动态检测（基于浏览网页引发的行为），以及两者混合的方法[1]。传统静态检测方法简单快速，但只能检测已知的特征，难以处理页面代码混淆，因此会出现大量的漏报和误报，因此，现有系统多使用动态检测的方法，通过在虚拟机中开启一个浏览器来打开网页，监控系统运行状态来找寻恶意行为。动态监测方法准确性较高，但资源消耗比较大，无法用来检测互联网上存在的大规模的网页。

本文提出了一种基于分类的网页恶意代码检测方法，通过分析页面代码，提取特征，进行机器学习得到分类模型。对一个小规模的数据样本进行实验，实验结果表明该系统能够较为准确有效的完成恶意网页检测，准确率达到 88.5%。

## 1　相关研究

JavaScript 具有使网页增加互动性，有规律地重复的 HTML 文段简化，减少下载时间，能及时响应用户的操作，对提交表单做即时的检查等诸多优点，几乎所有的用户默认允许其执行。为了保护用户，目前的浏览器使用沙箱限制 JavaScript 对资源的访问。随着技术的发展，JavaScript 攻击时，会出现恶意代码绕过沙箱或利用合法的方式，来欺骗用户采取不安全的行动指令。现在的 JavaScript 混淆工具使静态代码分析技术收到很大的制约[2]。

黄建军等人提出了基于植入特征的网页恶意代码检测方法，通过检测恶意代码对网页框架代码特征来判别网页中是否植入恶意代码。然而没有真正对恶意代码进行分析，只要恶意代码对其所采用的代码特征改变，就很难发现网页中是否含有恶意 JavaScript 代码[3]。

Egele 等人通过检测在 JavaScript 字符串中是否存在 Shellcode（Shellcode 一般用在 Heap Spray 攻击）来防止 Drive-by download 攻击[4]。Hallaraker 等人设计了一个基于浏览器的审计机制，可以检测和禁用 JavaScript 执行可疑的操作，如打开太多窗口，或访问一个域 Cookie[5]。

BrowserShield[6] 利用已知的漏洞检测到恶意脚本，然后动态重写它们以网页内容转换成一个安全等效的代码。

Seifert 等人收集结合的 HTTP 请求和网页内容的特征集，（包括内置页框和使用转义字符的有无和大小），并使用它来产生一个决策树[7]。

本文提出的基于机器学习分类器的网页恶意代码检测方法，通过分析 JavaScript 脚本本身特征，来检测网页恶意 JavaScript 脚本。克服了静态监测方法对混淆代码的误报和漏报，由于不需要模拟运行脚本，大大提高了分析效率，也能避免脚本本身给计算机带来的危害。

## 2 机器学习和恶意 JavaScript 脚本

使用机器学习分类，能有效区分恶意的和良性的 JavaScript。正如在[8]中提到的，恶意 JavaScript 脚本代码中所用到的混淆技术越来越复杂，本文通过分类器检测出恶意 JavaScript，如图 1 所示，在检测之前需要两个步骤：数据收集和特征提取。

### 2.1 数据收集

分类模型的生成需要有训练集。对于正常数据，使用爬虫从安全网站爬取。Alexa 排名靠前的网站[9]，每天访问量巨大，管理严格，安全性高，可以认为是安全的，作为正常网页代码集来源。恶意代码集的收集比较困难，主要有两点原因。1）虽然有很多相关研究，但最终都只发布 URL 黑名单列表，而不是分类需要的代码样本。2）含有恶意代码网页生存期很短，如果不及时搜集，页面很快就会失效，无法收集。由于以上原因，本文采取了如下方法搜集恶意代码。

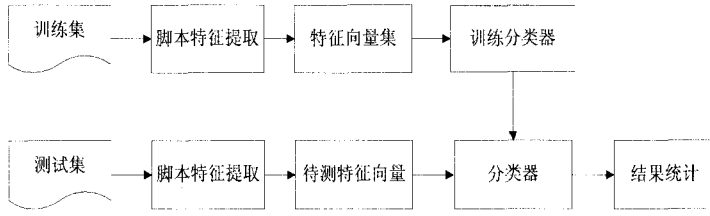


图 1 机器学习框架

首先，从知名的恶意网站发布站点去搜集含有恶意代码站点的 URL，这些网站每天会发布最新的恶意网站 URL，例如 http://www.antiy.com/和 StopBadware.org。然后，用爬虫爬取所搜集的 URL 站点，然后提取出其链接并使用 Google SafeBrowsing Api[10]进行检测，针对报出来的恶意网页进行人为分析，筛选出其中的恶意代码。搜集到的恶意页面都将在虚拟机内进行过验证，只有观察到了恶意行为才会加入最终的数据集。实验中，爬虫搜集了 300 个正常页面作为正常页面，又人为手动搜集了 300 个含有恶意 JavaScript 脚本的页面，这两类进行标注，记为正常和恶意的数据集，用来做训练数据集和测试数据集。

### 2.2 特征的选择与提取

机器学习中有效特征的选取，往往对整个系统的结果有着很大的影响。当今特征的选取主要有两个方法：1、依据专家知识选择影响力最大的特征；2、采用数学统计方法选择最好特征。本文的特征选取结合了 Peter Likarish[11]等人所提出有效特征和王松[12]等人用数学方法所统计特征。通过分类实验把一些不明显特征去掉，最后筛选出本文最终确定了如下表 1 所示 13 个特征：

表 1 有效特征表

| | 特 征 | 描 述 |
|---|---|---|
| 1 | eval 字符串个数 | eval 字符串在样本中出现的次数 |

| | 特　征 | 描　述 |
|---|---|---|
| 2 | unesape 字符串个数 | unescape 字符串在样本中出现的次数 |
| 3 | .exe 或者.EXE 所出现的次数 | .exe 或者.EXE 字符串在样本中出现的次数 |
| 4 | 特殊字符个数百分比 | 特殊字符(～,`,!,@,",#,&,MYM,%,^,&,*,(,),_,+,?,>,<,[,],{,},\)在样本中出现的频率(特殊字符数/样本字符数) |
| 5 | unicode 字符串个数 | Unicode 字符串在样本中出现的次数 |
| 6 | 空格字符个数百分比 | 空格字符在样本中出现的概率(空格字符数/样本字符数) |
| 7 | 大写字母个数百分比 | 大写字母(A — Z)在样本中出现的概率(大写字母数/样本字符数) |
| 8 | 数字字符个数百分比 | 数字字符(0 — 9)在样本中出现的概率(数字字符数/样本字符数) |
| 9 | 长字符串个数 | 字符串长度达到 150 及以上的字符串个数 |
| 10 | 平均字符串长度 | 样本总字符串长度/样本字符串总数 |
| 11 | 个数百分比 | 在样本中出现的次数/样本字符串数 |
| 12 | %u 个数百分比 | %u 在样本中出现的次数/样本字符串数 |
| 13 | 可读字符串个数百分比 | 字符串中字母出现次数到达%70 就以上的字符串定义为可读字符串(可读字符串个数/样本字符串数) |

## 3　分类判决

本文选择了以下分类器进行分类实验:SVM、决策树和朴素贝叶斯。以上三种分类器将在用 Java 编写的开源工具 WEKA 中建立模型。下面将简要介绍以下这三种分类器:

①SVM:支持向量机(SVM)被广泛认为是最先进的多维度数据二元分类方法,在理论和实际方面中全都取得了成功[13]。SVM 把向量映射到更高维度的空间里,在这个空间中建立有一个最大间隔的超平面。在分开数据的超平面两边建立两个互相平行的超平面。分隔的超平面使两个相互平行超平面的距离最大。如果两个平行超平面间距离或差距越大,则分类器的误差越小。

②决策树分类器:决策树是一个树形的结构,每个内部的节点都是一个属性的测试,一个分支表示一个测试输出,一个叶节点表示类和类分布。决策树所产生的分类规则很容易理解,准确率高。决策树归纳基本的算法是贪心算法,它用自顶向下递归的各个击破的方式构造判定树。如果样本都在同一个类中,那么该节点成为叶子节点,否则就选择一个能够最好的将样本分类的属性作为判定属性创建分支,选择过程中使用信息增益来度量。同时采用剪枝方法来消除噪声与孤立点。本文所采用的决策树算法是 C4.5 算法,该算法能够处理连续数值型的属性,采用后剪枝方法避免树的高度无节制的增长,添加了对缺失值的处理。通过训练样本集训练出决策树分类器模型,对给定样本,从根节点处开始顺着查找就可以得到分类结果。

③朴素贝叶斯分类器:朴素贝叶斯是一种很简单的分类思想,对于给出的待分类项,求解在此项出现的条件下各个类别出现的概率,哪个概率最大就认为该待分类项属于哪个类别。

本文使用 WEKA 中三种分类器模型建立的算法,只需从每个脚本代码中提取所述特征,并将其结果按 WEKA 要求转化成其所能读取的格式 ARFF 的文件中,训练上述分类器,并用训练后的分类

器，对测试集进行测试。

## 4 实验结果及其分析

本文把正确检测出的恶意网页脚本代码数量记为 TP，把正确检测出的正常网页数量记为 TN，总的测试样本数量为 T，恶意测试样本数量和正常测试样本数量都为 T/2，并采用常用的评价分类模型的 3 个重要指标：

准确率：被正确检测出的样本数/测试集所有样本数×100%，即 (TP+TN)/T×100%；

正确率，被正确分类的恶意代码样本数在测试集的所有恶意代码中所占比例，即 TP×2/T×100%；

误检率，将正常代码样本错误识别成恶意代码样本在测试集合的所有正常代码样本中所占比例，即(T/2-TN)/T×100%。

本文采用的验证方法为十折交叉验证(10-Fold cross-validation )，将数据集分成十份，轮流将其中 9 份作为训练数据，1 份作为测试数据，进行试验。每次试验都会得出相应的结果数据。10 次的结果数据的平均值作为试验的结果，本文又进行 5 次 10 折交叉验证，再求其均值，作为对最终试验的结果。实验结果如表 2 所示：

**表 2 实验结果**

| 分类器 | 准确率(%) | 正确率(%) | 误检率(%) |
|---|---|---|---|
| 决策树分类器 | 85.4 | 83.2 | 2.3 |
| SVM 分类器 | 88.5 | 86.4 | 2.8 |
| 朴素贝叶斯分类器 | 80.3 | 78.6 | 3.6 |

从表 2 可以看出，三种分类器中 SVM 分类器和决策树分类器这两种分类器的结果相差不是太大，而朴素贝叶斯分类器的结果较前两种分类器的结果在三个方面都有些差距。造成以上结果的原因是由于所采用的特征之间的相关性比较大，而朴素贝叶斯分类器的性能需要在特征之间相关性较小的时候才能表现的较好。

分类器对有些网页恶意代码的未能检测的原因可能有两种：其一所搜集的样本数据集太少；其二有些恶意代码和正常代码的相似性较高。

总体表明，本文所用的 13 个脚本特征对于恶意网页代码有相对较好的区分度，基于机器学习的网页恶意代码检测方法能够较为准确的检测出恶意脚本代码。

## 5 结束语

本文通过网页脚本特征提取，采用三种不同的机器学习分类方法对网页恶意脚本代码分类，实验表明，基于机器学习的网页恶意代码检测方法，所采用的 SVM 分类器可以获得平均 88.5%的准确率，86.4%正确率和 2.8%的误检率，能够较好的检测出网页恶意代码。未来工作是采集更多的网页脚本代码作为数据集，找出更有区分度的恶意网页代码特征，从而更进一步提高检测准确率。(下转 12 页)

[3] DENEUBOURG J L, GOSS S, FRANKS N, et al. The Dynamics of Collectivesorting: Robot-Like Ants and Ant-Like Robots[C]//Proc of the 1st Int'l Conf on Simulation of Adaptive Hav-iour, 1991: 356-365.
[4] LUMER E,FAIETA B. Diversity and adaptation in populations of clustering ants[C]//Proc of the 3rd International Conference on Simu-lation of Adaptive Behavior: from Animals to Animats. Cambridge: MIT Press / Bradford Books,1994: 499-508.
[5] HANDL J,MEYER B. Ant—based and swarm—based clustering[J]. Swarm Intelligence, 2007, 1(2):95-113.
[6] HOLLAND O E, MELHUISH C. Stigmergy, self—organization, and sorting in collective robotics [J]. Artificial Life, 1999, 5(5): 173? 202.
[7] 徐晓华，陈崚. 一种自适应的蚂蚁聚类算法[J]. 软件学报，2006，17(9)，1884-1889.
[8] BLAKE C L, MERZ C J. UCI Machine Learning repository of machine learning databases. 1998. http://www. ics. uci. edu/～mlearn/ MLSummary. html.
[9] 匡青，鲍梦. 改进蚁群算法的动态 K—均值聚类分析[J]. 软件导刊，2008，7(1):154-155.
[10] BANDYOPADHYAY S, MAULIK U. An evolutionary technique based on K-Means algorithm for optional clustering in $R^N$[J]. Information Sciences,2002, 146:221-237.
[11] DORIGO M, BONABEAU E, THERAULAZ G. Ant algorithms and stigmergy[J]. Future Generation Computer Systems, 2000, 16(8): 851-871.

## 作者简介：

**李　聪**(1988— )，男，陕西市宝鸡人，在读硕士研究生，主要研究方向：自然语言处理方向。

---

(上接 40 页)

## 参考文献：

[1] http://user. qzone. qq. com/95007917/blog/1274004740
[2] M. Johns. On javascript malware and related threats[C]. Computer Virology, Jan 2008
[3] 黄建军，梁彬. 基于植入特征的网页恶意代码检测[J]. 清华大学学报(自然科学版). 2009(S2)
[4] Egele. M, E. Kirda, and C. Kruegel. Defending browsers against drive-by downloads: Mitigating heap-spraying code injection attacks. Detection of Intrusions and Malware, Jan 2009.
[5] Hallaraker. O and G. Vigna. Detecting malicious javascript code in mozilla. Engineering of Complex Computer Systems, Jan 2005.
[6] Reis C,Dunagany J,Wang H J,et al. BrowserShield: Vulnerability-driven filtering of dynamic HTML[J]. ACM Transactions on the Web,2007,3(1):11.
[7] Seifert, I. Welch, and P. Komisarczuk. Identification of malicious web pages with static heuristics[C]. In Australasian Telecommunication Networks and Applications Conference, Jan 2008.
[8] Craioveanu. Server-side polymorphism: Techniques of analysis and defense. [C] In 3rd International Conference on Malicious and Unwanted Software,2008.
[9] http://www. alexa. com/topsites
[10] Google, Inc. Google safe browsing API. http://code. google. com/apis/ safebrowsing /
[11] Peter Likarish, Eunjin (EJ) Jung, and Insoon Jo. In The 4th International Malicious and Unwanted Software (Malware 2009) ,October 2009.
[12] 王松. 基于学习的恶意网页智能检测系统[D]. 南京：南京理工大学，2011:33-35
[13] B. Scholkopf and A. J. Smola. Learning with Kernels:Support Vector Machines,Regularization, Optimization, and Beyond. MIT Press, Cambridge, MA, 2002. 126～179

## 作者简介：

**李　洋**(1986— )，男，河南濮阳人，西安电子科技大学通信工程学院，硕士研究生，主要研究方向：网络安全。