研究与开发

一种基于多视角特征融合的 Webshell 检测方法

林锋[1]，徐柳婧[2]，陈晓华[3]，戚伟强[2]，陈可[2]，朱添田[4]

（1. 浙江经贸职业技术学院信息技术系，浙江 杭州 310018;
2. 国网浙江省电力有限公司信息通信分公司，浙江 杭州 310007;
3. 湖州师范学院信息工程学院，浙江 湖州 313002;
4. 浙江工业大学计算机科学与技术学院，浙江 杭州 310023）

摘　要：Webshell 是一种 Web 端的恶意脚本文件。它通常由攻击者上传至目标服务器来达成其非法的访问控制的目的。现有 Webshell 检测方法存在诸多不足，如单一的网络流量行为、简易被绕过的签名比对、单一的正则匹配等。针对上述不足之处，基于 PHP 语言的 Webshell，提出了一种基于多视角特征融合的 Webshell 检测方法，首先，提取包括词法特征、句法特征、抽象特征在内的多种特征；其次，利用费舍尔评分对特征进行重要程度的排序与筛选；最后，通过 SVM 建立能有效区分 Webshell 和正常脚本的模型。在大规模的实验中，模型对 Webshell 和正常样本的最终分类精度达到了 92.1%。

关键词：Webshell 检测；多视角特征融合；特征选择与提取；机器学习

中图分类号：TP309.5

文献标识码：A

doi: 10.11959/j.issn.1000–0801.2020158

Method of Webshell detection based on multi-view feature fusion

LIN Feng[1], XU Liujing[2], CHEN Xiaohua[3], QI Weiqiang[2], CHEN Ke[2], ZHU Tiantian[4]

1. Department of Science and Technology, Zhejiang Institute of Economics and Trade, Hangzhou 310018, China
2. Information and Communications Branch, State Grid Zhejiang Electric Power Company, Hangzhou 310007, China
3. School of Information and Engineering, Huzhou Teachers College, Huzhou 313002, China
4. College of Computer Science and Technology, Zhejiang University of Technology, Hangzhou 310023, China

Abstract: Webshell is a malicious script file on the Web. It is usually uploaded by the attacker to the target server to achieve the purpose of illegal access control. In order to overcome the shortcoming of the existing Webshell detection methods, such as single network traffic behavior, simple by passed signature comparison, and easily bypassed signature comparison, a method of Webshell detection based on multi-view feature fusion for PHP Webshell detecting was proposed. Firstly, multiple features including lexical features, syntactic features, and abstract features were extracted. Secondly, fisher score was used to sort and filter all features according to the degree of importance. Finally, a model that can effectively distinguish Webshell from normal scripts was established through SVM. The large-scale experiment in real-world scenario shows that the final accuracy of our model can reach 92.1%.

Key words: Webshell detection, multi-view feature fusion, feature selection and filtering, machine learning

收稿日期：2020–03–19；修回日期：2020–05–15

基金项目：国家自然科学基金资助项目（No.61772026，No.U1936215）

Foundation Items: The National Natural Science Foundation of China (No. 61772026, No. U1936215)

1 引言

Webshell 是攻击者上传到受感染的 Web 服务器的恶意脚本。攻击者往往通过 SQL 注入，利用文件包含漏洞、文件上传漏洞、跨站脚本攻击等方法上传精心制作的 Webshell 脚本，其目的是对受感染计算机进行持久访问并进行一系列的恶意利用，如执行系统命令、窃取并篡改用户数据、修改网站主页等。同时，Webshell 也是高级持续性威胁（APT）攻击中的重要环节[1]，其能够对政府、军事造成不可小觑的危害。因此，设计合理有效的 Webshell 检测方法至关重要。

尽管学术界和工业界针对 Webshell 已经提出了一系列的检测方法和防御方案。例如，在程序发布之前进行源代码的审计；在程序运行时进行实时的监控等。但是由于一些服务器端操作系统补丁更新不及时、第三方恶意软件随意下载安装等人为原因，攻击者更加容易地对这些目标服务器进行利用并上传 Webshell 达到其攻击目的。假设分析人员在事发后能及时地发现 Webshell 脚本并进一步采取删除隔离等手段，亦能有效地防止攻击者进一步的危害举动。然而，现有的 Webshell 脚本检测方法面临着一系列的挑战。首先，攻击者使用 Webshell 进行 C&C（command& control）连接交互时的行为可以被伪装，攻击者能够利用非频繁的交互轻松绕过基于网络流量的检测[2-5]。（例如，将攻击的周期延长，隔一段时间才执行一次需要的命令，而单独去看某一次命令却是一次正常的行为。若采用基于上下文内容综合检测的动态方法，海量日志数据的存储问题以及检测判断时的内存开销问题将是一场灾难，故本文在此暂不考虑动态检测方法）；其次，一个 Web 服务器上往往含有大量的脚本，例如功能脚本、插件脚本等。不同的脚本随着时间也可能会发生频繁的变化（如一个包含当前时间的脚本文件的文件签名特征会不断动态变化），这种情况下，基于文件签名的 Webshell 检测方法[6]就会变得不适用；正则匹配能够较好地提取文件所特有的模式，经常被用于恶意文件的匹配与识别。但是由于 Webshell 通常由 PHP、ASP、JSP、Python 等高级语言所写，虽然采用正则匹配的方法[7]可以识别部分 Webshell，但是高级语言的词法以及句法特征无法全部用正则表达式来体现。

针对上述挑战，本文提出了一种基于多视角特征融合的 Webshell 检测方法。首先，提取 PHP 脚本中的词法特征、句法特征以及抽象特征；其次，为了判断出对分类有理的特征，本文采用一种费舍尔评分机制对特征进行重要程度的排序与筛选；最后，本文利用 SVM 进行恶意 Webshell 的有效分类。

2 相关工作

为了有效地检测与对抗 Webshell，现有的 Web 服务器的安全应对策略可以分为以下 3 种：事前检测策略、事中检测策略和事后检测策略。

事前检测策略指的是在攻击没有发生时对 Web 服务器进行漏洞扫描与脆弱点挖掘。Huang 等[8]提出了一个框架来生成测试案例以揭示 Web 服务器的漏洞。Sooel 等[9]提出了一种基于语义的方法来识别存在漏洞（例如，存在无限循环特征、缺少授权检查函数等）的 PHP Web 应用程序。此外，参考文献[10-12]利用静态程序分析检测容易受到 SQL 注入和 XSS 攻击的 Web 应用程序。除了上述研究，还有一系列的 Web 漏洞扫描工具，如 Nikto2[13]、WebInspect[14]等被工业界广泛地使用。事前检测仅仅能够帮助分析者了解 Web 服务器系统或者应用软件中存在的漏洞，但由于攻击者利用手段的复杂性与多样性，仅仅扫描并修复漏洞并不能完全防止可疑代码的上传，也无法揭示将要发生的危害。

相对应地，事中检测的方法被许多研究人员提出：即实时检测攻击者正在上传 Webshell 或正

在利用漏洞或软件脆弱点对 Web 服务器造成危害的行为。例如，Almgren 等[15]开发了一款轻量级的在线攻击检测工具对 Web 服务器的行为进行实时监控。Kruegel 等[16]建立了统计模型来描述正常的网络请求，并将明显的偏差识别为异常。Robertson 等[17]利用多特征进行 Web 异常行为的检测。异常检测技术的难点在于如何构建正常活动的标签以及如何设计统计算法，从而避免把正常的操作标记为可疑或忽略真正的入侵行为。事实上，异常检测技术在降低漏报的同时会引入大量的误报，使得其很难被应用于真实场景中。对此，混合检测技术被提出，Ko 等[18]开发了一种语言框架，通过程序执行序列与并发程序语法事件的分析，最终定制了事件跟踪策略来进行实时的 Web 入侵检测。Prem 等[19]的实验结果证明了基于策略的方法优于单一的特征检测或者异常检测方法。Hossain 等[20]通过从审计数据中提取起源信息，提出了一种独立于应用程序的策略检测 Webshell 为代表的高级持续性威胁攻击提高检测水平，利用自定义的 IP 地址白名单进行入侵筛选，但是针对 Linux/Windows 这种复杂的 Web 服务器环境却显得力不从心。事件中检测大多基于先验知识，且存在误报高、规则挖掘定制难等问题。

尽管事前检测和事中检测都可以在一定程度上避免 Webshell 带来危害，但是仍然无法保证完全避开所有的漏洞和攻击。假设分析人员在事发后能及时地发现 Webshell 脚本并进一步采取删除隔离等手段，亦能有效地防止攻击者进一步的危害举动，因此需要设计切实有效的事后检测和缓解措施。Starov 等[21]对大量真实的 Webshell 进行了分析以揭示其功能，并通过建立“蜜罐”来研究攻击者如何使用 Webshell。然而，配置“蜜罐”并获取对应的攻击手段需要耗费大量时间，无法保证实时有效的 Webshell 检测。此外，有些 Webshell 有对“蜜罐”的检测功能，会在“蜜罐”中自动禁用某些恶意行为从而绕过检测。Tian 等[5]提出了一种网络流量的 Webshell 检测方法，利用卷积神经网络对 Webshell 运行时的流量进行监控建模。这种方法有两个缺点，首先，若攻击者将攻击的周期延长，隔一段时间才执行一次需要的命令，而该命令提出来是一次正常的行为，无疑可以绕过这种检测方案；第二，实时运行 Webshell 将可能会导致系统关键数据被破坏，存在着高风险。而本文提出的 Webshell 检测方法是基于 Webshell 不同特征的静态检测，可以有效避免上述缺陷。Tu 等[7]和 Cui 等[22]利用正则匹配和关键词特征匹配的方法来识别 Webshell，虽然可以从一定程度上有效识别可疑脚本，但是由于 Webshell 通常由高级语言所编写，其所包含的词法以及句法特征无法用正则表达式来全部体现。本文提出的方法综合考虑了这类高级语言（特别是 PHP 语言）中存在的词法、句法及抽象特征，且提出了一种基于费舍尔评分的特征综合排序选择方法，能更加准确高效地检测 Webshell。

3　基于多视角特征融合的 Webshell 检测方法

本文提出的方法包含如下 4 个步骤，Webshell 检测流程如图 1 所示：

（1）利用爬虫在 GitHub 上收集大量的 Webshell 脚本[23-25]，并对其进行去噪、去冗等预处理；

（2）分析并提取 Webshell 脚本中词法、句法及抽象特征；

（3）利用费舍尔评分对特征进行重要程度的排序与筛选；

（4）通过 SVM 建模并分类。

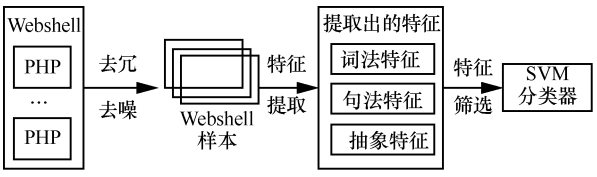

图 1　Webshell 检测流程

3.1 数据预处理

从 GitHub 上采集到相关的 Webshell 脚本之后，由于研究的针对性，本文首先提取所有的 PHP 脚本，共计得到 723 个 PHP Webshell。其次，对数据集进行预处理，以提高数据样本的质量用于之后的训练。主要包含以下三大类。

（1）去噪

首先考虑长度过大的脚本，参考文献[21]表明过大的脚本往往包含了大量语义无关内容与噪音，本文将大小超过 3 MB（通过大量分析发现，该数量级的脚本一般无法有效运行）的脚本作为噪声文本去除，得到 662 个有效脚本。此外，有些 PHP 脚本中会有大量的混淆与 BASE64 编码，使得基于静态分析的检测方法失去作用，在此，本文将包含的混淆代码和 BASE64 编码的脚本文件进行反混淆，通过利用 UnPHP 技术[26]，最终得到了 514 个可以有效运行而不超时的脚本。

（2）去冗

为了对 Webshell 脚本进行去冗，本文首先对所有的脚本进行了标准化操作，包括删除所有的代码注释、空行。其次，利用 TF-IDF 模型进行文本相似度的计算，所有相似度超过 95%的脚本仅保留一份。最终，本文得到了 407 个有效且可执行的脚本文件。

（3）按照家族聚类

由于 Webshell 之间存在一系列的变体，如 c99 的 Webshell 加入提权功能之后被重新命名为 c999，同时，c99 中的一些功能模块又可以在 Fx29 的 Webshell 中找到。它们之间的功能模块存在继承关系，但是又有所区别。故本文利用传递闭包的思想对家族进行聚类，即如果脚本 m1 和 m2 存在一部分相同的功能模块，而脚本 m2 和 m3 存在一部分相同的功能模块，可以推断 m1、m2、m3 隶属于一个家族。这种传递特性使得可能得到同一个家族的所有成员信息。同时，在这个过程中，一个家族中的 Webshell 的对应功能也逐步明确。在得到包含 407 个有效 Webshell 脚本文件的数据集后，本文对其进行家族的聚类。将聚类的阈值设置为 30%，即有 30%功能模块完全相同就被聚为一类。最终的聚类结果见表 1。

表 1 有效的 Webshell 家族聚类结果

家族名字	有效个数/个
c99	152
r57	97
WSO	65
B374k	31
NST	24
NCC	19
Crystal	19

3.2 特征提取

在得到数据集后，接下来需要对其进行特征提取。本文根据语义内容考虑词法、句法以及抽象 3 类特征。

（1）词法特征

词法特征指的是脚本中某些词语使用的方式、频率等。具体如下。

本文观察到在恶意的 Webshell 脚本中，由于攻击者需要通过 Webshell 接收各类命令与受害服务器进行信息交互，因此需要大量的全局变量，并将其参数与之结合，以便执行预定义的活动。由于全局变量是向 PHP 脚本提供外部数据的主要方式，因此恶意 Webshell 倾向于使用各种高度参数化的全局变量。因此，本文使用计算代表接受信息的全局变量的数量作为特征，包括$ _GET（F1）、$ _POST（F2）、$ _COOKIE（F3）、$ _REQUEST（F4）、$ _FILES（F5）、$ _SESSION（F6）

（2）句法特征

句法特征是指攻击者在编写 Webshell 脚本时所用到的表达方式。具体如下。

在攻击者上传 Webshell 之前，他们并不能清楚地知道对应平台的详细配置信息，例如 Web 服

务器开放了哪些服务，某个邮箱对应的密码是什么。这就需要对应的 Webshell 能够自动地适应各类操作系统并自动地尝试获取相关软件的权限。对此，有以下几类句法特征将被考虑。

- 条件语句占比。代表条件语句占脚本所有语句的百分比，包括脚本中的 if（F7）、elseif（F8）、else（F9）、case（F10）。
- 循环语句占比。代表循环语句占脚本所有语句的百分比，包括脚本中的 for（F11）、while（F12）和 foreach（F13）。

（3）抽象特征

抽象特征代表除了词法特征和句法特征之外的其他相关特征。具体如下。

- 敏感函数匹配度。代表 PHP 语言中一些关键词的运用情况，如伪装执行函数（eval）、文件获取函数（wget、curl、lynx、get、fetch）、反向连接函数（perl、python、gcc、chmod、nohup、nc），信息搜集函数（uname、id、ver、sysctl、whoami、$OSTYPE、pwd）等，常被用于 Webshell 来执行一些可疑行为。本文将是否含有这几类函数作为特征，即含有伪装执行函数（F14）、含有文件获取函数（F15）、含有反向连接函数（F16）、含有信息搜集函数（F17）。

此外，还添加了脚本中词的最大长度（F18），PHP 源码中行的最大长度（F19）和信息熵（F20）这 3 个常用的特征[27]。

综上所述，本文通过特征提取共计得到 20 个与 Webshell 判断相关的特征。

3.3　特征排序与筛选

在机器学习训练模型时，特征不仅影响了模型的收敛速度，还决定了最终分类的精度。在特征提取的基础上进一步进行有效的特征选择，可以减少特征数量避免过拟合，还可以让分析者对各个特征的优劣有一个直观的了解。费舍尔评分已经在相关工作中被证明是一种进行特征选择的好方法[28]。其理论依据是一个好的特征与同一类所有对应的值的方差应该尽可能地小。

假设 u_{f_i} 表示第 i 个特征在数据集上的平均值，$u_{f_i}^k$ 表示第 i 个特征在第 k 类中的平均值，n_k 表示第 k 类中的样本数量，f_{ij}^k 表示第 i 个特征在第 k 类中的第 j 个位置的值。那么，第 i 个特征的费舍尔评分可以表示为：

$$F(f_i)=\frac{\sum_{k=1}^{c} n_k(u_{f_i}^k-u_{f_i})^2}{\sum_{k=1}^{c}\sum_{y_j=k}(f_{i,j}^k-u_{f_i}^k)^2} \tag{1}$$

若 $F(f_i)$ 越大，则说明对应的特征在同一类所有对应的值的方差越小，该特征也就越好。

3.4　建模与分类

在经过数据的预处理、特征提取与选择之后，最后需要构建对应的模型进行恶意 Webshell 判断。针对分类目标，结合实际的数据特征来选择合适的分类器有助于提高模型的判别能力。综合考虑传感器数据边界的非线性特点、特征的多维特性与 Webshell 各个维度特征之间的依赖关系，本文选择支持向量机（support vector machine，SVM）作为分类器，理由如下：首先，由于 Webshell 的复杂性，很多正常的脚本文件也会做与 Webshell 类似的行为，故恶意 Webshell 在用特征进行分类时并不是线性可分的，而 SVM 存在多种核函数（如多项式核函数、高斯核函数）来处理非线性可分问题。其次，本文提取了 20 维（特征选择前）的多维特征向量，已有相关工作[28]表明 SVM 在多维特征分类上取得了令人满意的效果。

4　实验与分析

4.1　实验设置

本文的恶意 Webshell 样本来源于 GitHub，并经过了预处理与聚类，最后得到了 407 个有效 Webshell 脚本文件（第 3.1 节中已经详细介绍）。为了进行二分类，本文额外从 GitHub 上通过爬虫采集了

407 个正常的 PHP 脚本用于训练和测试。

4.2 特征选择实验

本文采用费舍尔评分进行特征选择。对于每一类特征，其费舍尔评分的计算值与其对最终分类精度造成的影响之间的关系如图 2 所示。

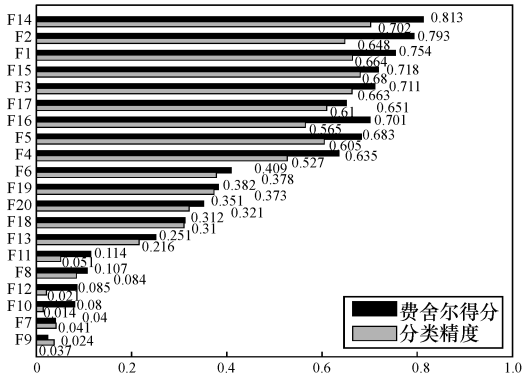

图 2 费舍尔得分与分类精度的关系

由图 2 可知，若某个特征不能很好地代表用户独特的行为，那么其费舍尔评分将会接近于 0，利用该特征单独做分类时的精度也会变得非常低。相反，若某个特征非常具有代表性，那么其费舍尔评分将会远远大于 0，利用该特征单独做分类时的精度也会变得较好。在图 2 中，通过比较可以看到较好的特征包括 4 类函数（即伪装执行函数、含有文件获取函数、含有反向连接函数和含有信息搜集函数）以及代表接受信息的全局变量的数量；而代表性较弱的特征有条件语句占比中的 if 和 else。在此，本文以费舍尔得分 0.1 为阈值，本文取排名前 16 位的特征组成最终的特征向量。

此外，为了进一步说明特征可以很好地区分正常脚本和恶意脚本，在此利用费舍尔得分提取到的 16 个特征和 814 个样本的数据集（407 个正常样本和 407 个恶意 Webshell 样本）进行了实验。针对特征向量的每一个维度数据，首先，本文计算各类脚本（正常类和恶意类）的中心点，分别记为 c_n 和 c_m。然后，记录正常类和恶意类脚本的数据样本中对应维度数据到中心点 c_n 和 c_m 的欧几里得距离的平均半径，分别记为 r_n 和 r_m。最后，计算正常类和恶意类脚本中心点之间的欧几里得距离，记为 dc_{mn}。针对 16 个维度，最终的结果见表 2。由表 2 可知，r_n 和 r_m 的值比 dc_{mn} 要小得多，这说明了本文提取的特征可以很好地区分正常脚本和恶意脚本。

表 2 特征集群可分说明

参数	数值
r_n	2.423
r_m	2.951
dc_{mn}	17.394

4.3 评价指标

在精度方面，本文利用如下的指标来评判模型的优劣：

- TP（true positive），即 Webshell 被准确识别；
- FP（false positive），即正常脚本被判断为 Webshell；
- TN（true negative），即正常脚本被准确识别；
- FN（false negative），即 Webshell 被误判成正常脚本。

Webshell 的识别率：

$$\mathrm{TPR}=\frac{\mathrm{TP}}{\mathrm{TP}+\mathrm{FN}} \tag{2}$$

正常脚本的识别率：

$$\mathrm{TNR}=\frac{\mathrm{TN}}{\mathrm{FP}+\mathrm{TN}} \tag{3}$$

总体精度：

$$\mathrm{Accuracy}=\frac{\mathrm{TP}+\mathrm{TN}}{\mathrm{TP}+\mathrm{FP}+\mathrm{TN}+\mathrm{FN}} \tag{4}$$

4.4 建模训练及测试实验

首先要对数据进行训练集合测试集的划分，本文将数据集进行 6:4 分割，即 60%的 Webshell 和正常脚本用于训练模型，剩下的 40%用于测试。为了不失一般性，本文在所有家族的 Webshell 中随机抽取 60%的脚本，并与正常样本中随机抽取的 60%脚本混合在一起作为有标签的训练集，这样

做的好处是可以保证每一类的家族特点都能被模型学习到。在训练时，SVM 采用径向基核函数，因为本文中的 Webshell 分类是一个多维度的非线性问题，利用径向基核函数可以快速有效地建立模型。

为了优化模型，本文采集网格搜索法确定最佳的 SVM 训练参数。在 SVM 中，需要确定的两个主要参数分别是 C 和 γ，其中，参数 C 是惩罚系数。如果 C 变大，说明惩罚的力度变大，导致模型不够灵活、泛化能力变弱的情况。反之，如果 C 变小，那么惩罚力度就自然变小，模型就容易出现欠拟合的现象。另一个参数 γ 决定了数据映射到新的特征空间后的分布。与参数 C 的情况类似，γ 太大会导致模型并不学习向量的内容，而是仅仅记住了支持向量本身，导致泛化能力下降，容易出现过拟合的情况。而 γ 太小则会造成新特征空间的数据分布过于平滑，出现欠拟合的情况。本文首先将 SVM 收敛系数固定为 0.01，参数 γ 固定为 0.01，参数 C 与模型最终分类精度的变化如图 3 所示。由图 3 可知，当参数 C 的值为 0.8 时，分类效果最佳。同理，当 SVM 收敛系数固定为 0.01，参数 C 固定为 0.8 时，参数 γ 与模型最终分类精度的变化如图 4 所示。由图 4 可知，当参数 γ 的值为 0.03 时，分类效果最佳。综上所述，本文中采用 C=0.8 和 γ=0.03。

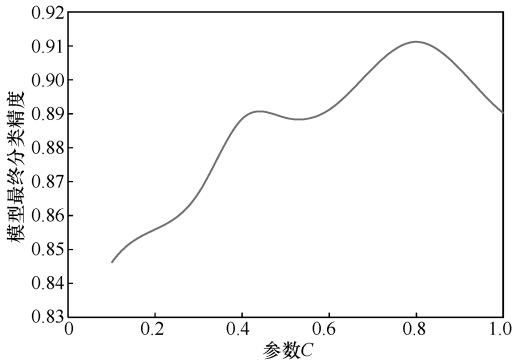

图 3　模型最终精度与参数 C 的关系

在最优参数下，本文做了 10 次交叉验证，最终得到的 TP、FP、TN、FN 的平均值见表 3。由表 3 可知 Webshell 的识别率 TPR 为 90.7%，正常脚本的识别率 TNR 为 93.6%。模型最终分类精度为 92.1%。在实际运用中，可以调整分类的阈值，例如在安全性较高的 Web 服务器上，将分类阈值变小，这样可以适当地降低正常脚本的识别率来提高对 Webshell 的检测率。

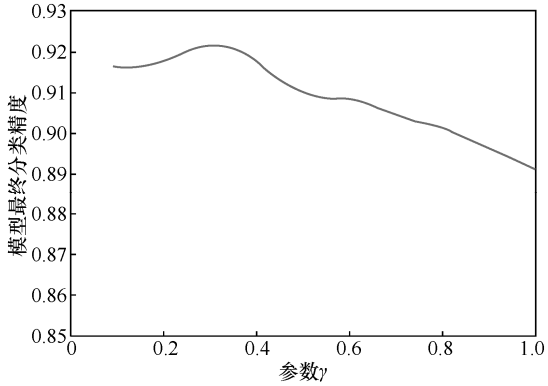

图 4　模型最终精度与参数 γ 的关系

表 3　10 次交叉验证平均分类结果

对比项	Webshell 脚本	正常脚本
Webshell 脚本	TP = 369	FN=38
正常脚本	FP=26	TN=381

5　结束语

为实现精确有效的 Webshell 检测，本文提出了一种基于多视角特征融合的 Webshell 检测方法。本方法首先提取包括词法特征、句法特征、抽象特征在内的多种特征；其次，利用费舍尔评分对特征进行重要程度的排序与筛选；最后，通过 SVM 建立代表 Webshell 的模型并分类。在大规模的实验中，模型对 Webshell 和正常样本的最终分类精度达到了 92.1%。

参考文献：

[1] Compromised web servers and web Shells[Z]. 2017.

[2] YANG W, SUN B, CUI B. A Webshell detection technology based on http traffic analysis[C]//Proceedings of International Conference on Innovative Mobile and Internet Services in Ubiquitous Computing. Berlin: Springer, 2018.

[3] 赵运弢, 徐春雨, 薄波, 等. 基于流量的 Webshell 行为分析与检测方法[J]. 网络安全技术与应用, 2018 (4): 8-9.
ZHAO Y T, XU C Y, BAO B, et al. Webshell behavior analysis

and detection method based on traffic[J]. Network Security Technology and Application, 2018 (4): 8-9.
[4] 王应军. 基于流量的 Webshell 通信识别[D]. 武汉: 武汉大学, 2018.
WANG Y J. Webshell communication recognition based on traffic[D]. Wuhan: Wuhan University, 2018.
[5] TIAN Y, WANG J, ZHOU Z, et al. CNN-Webshell: malicious Webshell detection with convolutional neural network[C]// Proceedings of the 2017 VI International Conference. [S.l.:s.n.], 2017.
[6] KIM J, YOO D H, JANG H, et al. WebShark 1.0: a benchmark collection for malicious Webshell detection[J]. Journal of Information Processing Systems, 2015, 11(2): 229-238.
[7] TU D T. CHENG G, GUO X J, et al. Webshell detection techniques in Web applications[C]//Proceedings of International Conference on Computing, Communication and Networking Technologies (ICCCNT). Piscataway: IEEE Press, 2014.
[8] HUANG Y W, TSAI C H , LIN T P , et al. A testing framework for Web application security assessment[J]. Computer Networks, 2005, 48(5): 739-761.
[9] SOOEL S, VITALY S. Saferphp: finding semantic vulnerabilities in PHP applications[C]//Proceedings of ACM SIGPLAN Workshop on Programming Languages & Analysis for Security. New York: ACM Press, 2011.
[10] WASSERMANN G , SU Z . Sound and precise analysis of web applications for injection vulnerabilities[J]. ACM SIGPLAN Notices, 2007, 42(6): 32.
[11] GARY W, SU Z D. Static detection of cross-site scripting vulnerabilities[C]//Proceedings of ACM/IEEE International Conference on Software Engineering. Piscataway: IEEE Press, 2008.
[12] XIE Y, AIKEN A. Static detection of security vulnerabilities in scripting languages[J]. USENIX Security Symposium. 2006(15): 179-192.
[13] EE M, LEE Y, YOON H. An enhanced rule-based web scanner based on similarity score[J]. Advances in Electrical and Computer Engineering, 2016, 16(3):9-14.
[14] ANG X, WANG L, WEI G, et al. Hidden web crawling for SQL injection detection[C]//Proceedings of IEEE International Conference on Broadband Network and Multimedia Technology (IC-BNMT). Piscataway: IEEE Press， 2010.
[15] ALMGREN M, DEBAR H, DACIER M. A lightweight tool for detecting web server attacks[C]//Proceedings of ISOC Network and Distributed System Security Symposium. [S.l.:s.n.], 2000.
[16] KRUEGEL C, VIGNA G. Anomaly detection of web-based attacks[C]//Proceedings of the 10th ACM Conference on Computer and Communications Security. New York: ACM Press, 2003: 251-261.
[17] ROBERTSON W, VIGNA G, KRUEGEL C, et al. Using generalization and characterization techniques in the anomaly-based detection of Web attacks[C]//Proceedings of ISOC Network and Distributed System Security Symposium. [S.l.:s.n.], 2006.
[18] KO C, RUSCHITZKA M, LECITTK. Execution monitoring of security-critical programs in distributed systems: A specification-based approach[C]//Proceedings of 1997 IEEE Symposium on Security and Privacy. Piscataway: IEEE Press, 1997: 175-187.
[19] PREM U, SEKAR R. Experiences with specification-based intrusion detection[C]//Proceedings of International Workshop on Recent Advances in Intrusion Detection. Berlin: Springer, 2001: 172-189.
[20] HOSSAIN M N, MILAJERDI S M, WANG J, et al. Sleuth: real-time attack scenario reconstruction from cots audit data[C]//Proceedings of 26th Security Symposium. [S.l.:s.n.], 2017: 487-504.
[21] STAROV O, DAHSE J, AHMADS S, et al. No honor among thieves: A large-scale analysis of malicious Webshells[C]// Proceedings of the 25th International Conference on World Wide Web. [S.l.:s.n.], 2016: 1021-1032.
[22] CUI H, HUANG D, FANG Y, et al. Webshell detection based on random forest–gradient boosting decision tree algorithm[C]// Proceedings of 2018 IEEE Third International Conference on Data Science in Cyberspace (DSC). Piscataway: IEEE Press, 2018: 153-160.
[23] OHN T. JohnTroony's php-Webshells repository[EB]. 2016.
[24] IKICAT. Nikicat's web-malware-collection repository[EB]. 2016.
[25] Ennc's Webshell Repository. Tennc's Webshell repository[EB]. 2016.
[26] HANG Z, LI M, ZHU L, et al. Smart detect: a smart detection scheme for malicious Webshell codes via ensemble learning[C]//Proceedings of International Conference on Smart Computing and Communication. Berlin: Springer, 2018.
[27] HAITIN. Webshell detector[Z]. 2018.
[28] HU T, QU Z, XU H, et al. Risk Cog: unobtrusive real-time user authentication on mobile devices in the wild[J]. IEEE Transactions on Mobile Computing, 2019, 19(2): 466-483.

[作者简介]

林锋（1979– ），男，浙江经贸职业技术学院信息技术系副教授，主要研究方向为网络安全。

徐柳婧（1989– ），女，现就职于国网浙江省电力有限公司信息通信分公司，主要研究方向为信号与信息技术、信息技术管理。

陈晓华（1977– ），男，湖州师范学院信息工程学院副教授，主要研究方向为网络资源分配与安全。

戚伟强（1984– ），男，现就职于国网浙江省电力有限公司信息通信分公司，主要研究方向为网络安全和信息运维。

陈可（1988– ），男，现就职于国网浙江省电力有限公司信息通信分公司，主要研究方向为电力信息技术。

朱添田（1992– ），男，博士，浙江工业大学讲师，主要研究方向为网络安全。