

Web 安全漏洞检测研究综述

郭锡泉^{1,2} 冯继舜²

¹(清远职业技术学院网络中心 广东 清远 511510)

²(清远市网络空间安全工程技术研究开发中心 广东 清远 511510)

摘要 Web 服务是最主要的互联网应用之一,然而 Web 安全漏洞的种类和数量还在不断增加,对用户的正常业务和信息资产构成极大的威胁,因此对 Web 安全漏洞检测的研究具有非常重要的意义。分析 Web 应用的通信模型和软件开发架构,介绍 Web 安全漏洞的种类,研究 Web 安全漏洞检测工具软件的应用,总结 Web 安全漏洞检测的方法。分析当前研究中存在的不足之处,并对未来 Web 安全漏洞的研究方向进行探讨和展望。

关键词 Web 应用 Web 安全 漏洞 检测

中图分类号 TP393.08

文献标志码 A

DOI:10.3969/j.issn.1000-386x.2026.04.001

A SURVEY OF WEB SECURITY VULNERABILITY DETECTION

Guo Xiquan^{1,2} Feng Jishun²

¹(Network Center, Qingyuan Polytechnic, Qingyuan 511510, Guangdong, China)

²(Qingyuan Research and Development Center of Cybersecurity, Qingyuan 511510, Guangdong, China)

Abstract Web service is one of the most important Internet applications. However, the types and number of Web security vulnerabilities are increasing, which poses a great threat to users' normal business and information assets. Therefore, the research on Web security vulnerability detection is of great significance. This paper analyzed the communication model and software development architecture of Web applications, introduced the types of Web security vulnerabilities, studied the application of Web security vulnerability detection tool software, and summarized the methods of Web security vulnerability detection. This paper pointed out the shortcomings of the current research, and discussed and prospected the future research direction of Web security vulnerabilities.

Keywords Web application Web security Vulnerability Detection

0 引言

国家信息安全漏洞共享平台(CNVD)当期收录通用型安全漏洞 13 083 个^[1],其中,Web 应用漏洞高达 29.6%。攻击者对应用层漏洞的关注程度远高于操作系统和其他对象,Web 应用已成为攻击者的“重头戏”。网络安全厂商纷纷推出检测软件,包括一系列商业软件和开源软件。伴随学术界和工业界的共同推动,Web 安全漏洞检测目前已形成一套相对成熟的方法和体系。本文从 Web 应用的通信模型和软件开发架构、Web 安全漏洞的种类进行分析,列举总结 Web

安全漏洞检测工具软件的应用、Web 安全漏洞检测的方法,并在此基础上提出当前研究的不足和今后发展的方向。

1 Web 应用开发与 Web 安全漏洞

1.1 Web 应用的软件开发

浏览器是用户访问网站的客户端软件,浏览器软件是通用的,一个浏览器可以访问任何不同的网站。对于 Web 应用,网站内容和功能的开发任务全部落在服务器一侧(图 1)。对于 Web 应用开发,最为核心的则是动态网页语言和数据库的选择。在 Web 应用开

发变得越来越复杂化、多样化的同时,Web 安全漏洞数量与日俱增,Web 应用的安全风险也越来越高。



图1 Web 应用软件开发的结构

1.2 Web 安全漏洞

关于 Web 安全漏洞的分类,目前还没有统一和全面的标准。在应用实践中,最为流行和通用的 Web 漏洞分类方式莫过于开放 Web 应用程序安全项目(Open Web Application Security Project)发布的 OWASP Top 10。该列表依次发布了 2013 年、2017 年和 2021 年三个版本^[2-3](表 1 所示为 2017 和 2021 版)。首先,漏洞的命名一般包含产生的具体领域及其表现形式,如失效的访问控制、加密机制失效、不安全设计、安全配置错误等。其次,漏洞的命名也可以是攻击手法,如注入、跨站脚本、跨站请求伪造、服务器端请求伪造等。

表 1 OWASP Top 10 列表

序号	2017 版	2021 版
A1	注入	失效的访问控制
A2	失效的身份认证	加密机制失效
A3	敏感信息泄露	注入
A4	XML 外部实体 (XXE)	不安全设计
A5	失效的访问控制	安全配置错误
A6	安全配置错误	自带缺陷和过时的组件
A7	跨站脚本 (XSS)	身份识别和身份验证错误
A8	不安全的反序列化	软件和数据完整性故障
A9	使用含有已知漏洞的组件	安全日志和监控故障
A10	不足的日志记录和监控	服务器端请求伪造

有些漏洞一直位于三个版本的列表之中,如注入和安全配置错误。尽管有些漏洞名称有所变化,但都指向相同的问题,如身份认证、访问控制和组件。而注入、身份认证、访问控制和组件这些漏洞都共同指向 Web 应用开发中动态网页和数据库这一核心。大多数网络攻击是从黑客直接向服务器发起的,如注入攻击、跨站请求伪造、上传漏洞攻击等。也有一些攻击是黑客操纵服务器向正常用户发起的,如跨站脚本、服务器

端请求伪造等。

2 Web 安全漏洞检测软件及其研究

2.1 相关术语及其计算方法

在对检测结果进行评估时,常常涉及一些术语及其计算方法。关于系统源代码某处“是-否”存在漏洞(正样本与负样本),会产生“对”和“错”两种判断,一共形成 TP、FP、TN、FN 等 4 种结果。

TP(True Positive):判断为正样本实际也为正样本,即正确识别出正样本的数量。

FP(False Positive):判断为正样本实际为负样本,即误报为正样本的数量。

TN(True Negative):判断为负样本实际也为负样本,即正确识别出负样本的数量。

FN(False Negative):判断为负样本实际为正样本,即误报为负样本的数量或漏报的正样本数量。

进一步,可用准确率(Accuracy)来反映检测总的正确程度,即正样本和负样本都找对了的比例。

$$A_{accuracy} = \frac{(T_p + T_n)}{(T_p + T_n + F_p + F_n)}$$

对于正样本的查找,还有查全率(Recall)和查准率(Precision)两个概念。

$$R_{recall} = \frac{T_p}{(T_p + F_n)}$$

查全率反映到底找出了多大比例的正样本(阳性敏感度),查全率高,则漏报率低;反之,查全率低,则漏报率高。

$$P_{precision} = \frac{T_p}{(T_p + F_p)}$$

查准率反映正样本检测的正确程度(预测准确率),查准率高,则误报率低;反之,查准率低,则误报率高。

2.2 检测软件运用效果的评估研究

文献[4]以采用动态测试技术的开源软件为对象,具体选取 Arachni(0.4.3 版)、Arachni(1.0.2 版)、Wapiti(2.3.0 版)和 Skipfish(2.1 版)等 4 款 Web 漏洞检测软件,从多个方面综合评估它们的检测能力。基于 Web Application Vulnerability Scanner Evaluation Project(WAVSEP)的测试案例集,对四款软件在 SQL 漏洞和 XSS 漏洞两方面进行识别准确率的测量,得到数据如表 2 所示。结果显示,Arachni(1.0.2 版)对两种类型漏洞的检测能力俱佳;Arachni(0.4.3 版)、Wapiti(2.3.0 版)检测 SQL 漏洞的能力较强,但检测

XSS 漏洞的能力较弱;Skipfish(2.1 版)检测 SQL 漏洞的能力较弱,但检测 XSS 漏洞的能力较强。由于实验的局限性,该文仍认为四款软件在漏洞检测能力方面并没有十分显著的性能差别,这些软件产生的检测报告不一致也并非检测能力的差异所决定。该文建议 Web 应用开发者学会使用有关安全基准和测试集来评估所使用的漏洞检测软件的性能表现。

表 2 四款开源软件检测能力对比(%)

软件	查全率		查准率		准确率	
	S	X	S	X	S	X
A0	99.26	64.38	98.53	100	97.93	67.50
A1	100	87.67	100	100	100	88.75
W	97.04	61.64	95.50	93.75	95.86	61.25
S	77.04	86.30	100	100	78.62	87.50

注:四款软件依次用 A0、A1、W、S 进行表示,S 代表 SQL 漏洞,X 代表 XSS 漏洞。

文献[5]在开源软件的基础上同时加入商业软件为对象,具体选取了如下 8 款软件参与对比:(1) Acunetix;(2) HP WebInspect;(3) IBM AppScan;(4) OWASP ZAP;(5) Skipfish;(6) Arachni;(7) Vega;(8) Iron WASP。该文以 WebGoat 和 Damn Vulnerable Web Application(DVWA)为测试平台,研究上述软件的漏洞检测能力。该文的主要实验结果如下:(1) 全部 8 款软件的查全率均达到 100%。(2) 查准率方面,Skipfish 对 DVWA 和 WebGoat 均为 75%;Acunetix 对 DVWA 为 68%,对 WebGoat 为 64%;ZAP、Arachni、Vega 对 DVWA 均为 56%。该文认为商业软件的漏洞检测能力是有效的,部分开源软件(如 Skipfish 和 ZAP)也有效。该文建议用户通过使用商业软件或合适的开源软件提高代码检查的覆盖面和检测率,并减少误判。

文献[6]以采用自动静态分析技术的漏洞检测软件为对象,具体选取了如下 5 款软件参与对比:(1) phpSAFE;(2) RIPS v0.55;(3) WAP v2.0.1;(4) Pixy v3.03;(5) WeVerca v20150804。该文的目的是研究复合运用上述软件对提高漏洞检测能力的影响。在不考虑相关软件费用成本的情况下,以 WordPress 这个内容管理系统(CMS)作为测试数据集,对比多种软件组合方式在 SQL 漏洞和 XSS 漏洞两方面的检测能力,得到数据如表 3 所示。结果显示,复合使用多种软件的检测能力并非总是比使用单一软件强。检测软件最佳的组合方式依赖于特定的条件,该文建议使用者对自己的安全测试基准做好合适的定位。

表 3 不同软件组合方式下检测能力对比

软件组合	查全率		查准率	
	S	X	S	X
ac	0.867	0.982	0.929	0.793
ace	0.867	0.982	0.929	0.793
abce	0.867	0.982	0.929	0.786
acde	0.867	0.982	0.929	0.786
abc	0.867	0.982	0.929	0.786
c	0.653	0.673	0.925	0.796
a	0.387	0.607	0.853	0.850

注:S 代表 SQL 漏洞,X 代表 XSS 漏洞。

除了静态分析(SAST)类软件和动态测试(DAST)类软件,后来还出现一类交互式测试(IAST))类软件。文献[7]以上述三类软件的组合为对象,每个类别选取两款软件参与对比:(1) Fortify SCA(静态);(2) FindSecurityBugs(静态);(3) OWASP ZAP(动态);(4) Arachni(动态);(5) Contrast Community Edition(交互式);(6) CxIAST(交互式)。该文的目的是研究复合运用上述类型的软件对提高漏洞检测能力的影响。在 OWASP 基准项目中随机选取 320 个测试用例,对比多种类型的组合方式的漏洞检测能力,表 4 为两种不同类型软件复合使用下检测能力的表现。该文主要结论如下:(1) 只使用一种类型软件而言,IAST 检测能力最强。(2) 两种类型软件复合使用而言,IAST + DAST 检测能力最强;其中,若联合 SAST 使用,虽然查全率也较高,但查准率较差。(3) 三种类型软件复合使用而言,Fortify + Arachni + CCE 或 Fortify + ZAP + CCE 最佳。

表 4 两种不同类型软件组合方式下检测能力对比

软件组合	查全率	查准率
Arachni + CCE	1	1
ZAP + CCE	1	0.95
Fortify + CCE	1	0.63
FSB + Fortify	1	0.56
Fortify + CxIAST	0.99	0.63
FSB + ZAP	0.98	0.59
Fortify + ZAP	0.93	0.62
Arachni + CxIAST	0.92	1
ZAP + CxIAST	0.91	0.95

上述关于 Web 漏洞检测软件运用方法的研究,对 Web 应用的开发人员和运维人员都有重要的指导意义。在 Web 应用开发的过程中,按照上述方法可及时

发现代码中的漏洞,尽早加以纠正,从而从根本上消除漏洞。在 Web 应用的运维过程中,参考上述方法加强 Web 安全漏洞的检测,可主动发现问题,尽早向开发人员反馈,从而为 Web 安全防护赢得宝贵的时间窗口。

3 传统基于软件测试和分析的方法

Web 安全漏洞检测软件在实践中发挥了积极的作用,但往往存在较高的误报率和漏报率。学术界积极探索,不断提出新的方法来提高 Web 安全漏洞检测的准确率和效率。Web 安全漏洞检测沿袭了软件漏洞检测的技术路线,传统的检测方法主要有动态测试、静态分析和混合方法等。

3.1 动态测试方法

动态测试方法是在运行程序的情况下,通过模拟输入并观察输出,进而判断是否存在漏洞的方法。按照是否清晰了解源代码可以分成三种情形:(1) 黑盒,完全不知道或不理会源代码;(2) 白盒,掌握源代码并十分清楚程序内部结构;(3) 灰盒,介于黑盒与白盒之间,掌握程序和源代码的部分信息。模糊测试(Fuzzing)是最为常见的动态测试方法。由于进行了输入、输出的验证,动态测试的漏洞查准率很高。但动态测试不可能模拟所有的输入(特别是非正常、非预期的),该方法难以保证漏洞查全率,测试需要的开销也比较大。

文献[8]提出一种聚类算法来检测 Web 应用漏洞。对 Web 服务器返回的响应进行自动分类,并使用基于页面距离的族算法判定服务器是否存在漏洞。系统原型对 phpBB、SecurePage、HardwareStor 等五款 Web 应用进行漏洞检测,共发现 12 个漏洞。而 Skipfish、W3af、Wapiti 等三款检测软件发现的漏洞均不超过 5 个。

文献[9]设计了一种基于动态测试的跨站脚本漏洞检测方法。动态检测过程包括五个步骤:爬虫、特征构造、攻击模拟、结果检测以及报告生成。爬行器采用滑动窗口的自适应爬行策略,特征构造算法考虑到三个控件(文本框控件、文本字段控件和密码控件)、四个位置(注释、输入元素、脚本标签、页面正文)和各种标签属性等多种因素。该文以某 Web 论坛应用为测试对象,跨站脚本漏洞检测的覆盖面高于 90%,检测性能优于专业检测软件 XSS-ME。

针对反射型 XSS 漏洞的检测,文献[10]提出另外一种基于动态分析的检测方法。该方法对攻击载荷进行分解,分解后的载荷单元要进行单独测试和组合测试。为提高准确率,还需找出可以规避过滤的载荷单

元以及避开规则的组合。通过与 AWVS、AppScan 等检测软件对比,该方法在保持较高查准率的同时,也有效提升了查全率。而且,该方法使用较少的测试请求即可完成测试任务,系统整体效率高。

文献[11]针对电力 Web 应用系统 SQL 注入漏洞的检测,提出一种基于特征的自动化渗透测试方法。提取电力 Web 应用的安全特征,通过构建特征矩阵(Feature Matrix)来生成大量测试用例,以提高渗透测试的覆盖度。对于同一注入点,采用启发式的特征筛选(Heuristic Feature Selection)算法去除不符合本次测试条件或不必要的测试用例,从而提高系统的整体检测效率。实现一个原型系统,与 SQL 注入漏洞检测软件 SQLMap、WVS Acunetix 相比,有更高的查准率(98% ~ 100%)和更低的误报率。

文献[12]针对 Java 类 Web 应用的漏洞提出一种黑盒式模糊检测方法。只需要给出 Web 应用的网址,系统根据 HTML、JavaScript 等语言的约束提取(Constraints Extraction),自动产生输入构建网站的导航图(Navigation Graph)。攻击构造模块根据导航图对 Web 应用进行模糊测试,攻击结果分析模块根据服务器的响应进行漏洞检测。实验结果显示,该系统的查全率不低于 Black Widow、ZAP、SQLMap 等检测软件。在保证较高查全率和查准率的同时,该系统的软件开销更低,整体检测效率更高。

以上研究的重点,一方面是根据漏洞类型和应用场景设计针对性的动态测试方法,提高漏洞的查全率;另一方面则是精简测试用例或测试过程,降低测试开销,提高整体测试效率。

3.2 静态分析方法

静态分析是在掌握程序源代码(白盒)的前提下,对程序语法、结构和功能进行分析,以找出漏洞的方法,可以把静态分析看成动态测试的一种特定情形(完全知晓程序源代码)。静态分析一般不需要运行程序,但也可以运行程序以获得更多信息进行研判。静态分析采用的分析方法有污点分析、符号执行等。(1) 污点分析通过跟踪不确定输入在程序执行中的传播路径,并判断是否产生安全问题。污点分析包括静态污点分析和动态污点分析两种方法。静态污点分析不运行程序,在对程序源代码进行语法、词法分析的基础上,通过污染源-污点跟踪-污点聚合的模型找出漏洞。由于缺乏程序运行中的具体信息,静态污点分析容易产生误报。动态污点分析需要运行程序,通过获取程序运行中的数据流和控制流,更容易找出漏洞。但受代码覆盖率的限制,动态污点分析容易产生漏报。

(2) 符号执行把程序的变量符号化,随机生成一个输入并仿真程序运行,收集当前路径上的约束条件,对约束条件取反求解;得到的解作为新的输入,如此循环反复直到行经所有的路径(或符合条件结束程序)。还可以把漏洞分析规则引入到符号执行的过程中,根据程序运行是否符合规则来判断是否存在漏洞。

文献[13]针对 Java Web 应用,提出一种轻量级静态分析方法自动检测 Java Web 应用的逻辑漏洞。在该方法中,逻辑错误是以程序功能的错误调用为特征的。程序分片技术被用来捕获唤起关键程序功能的过程,在监视程序功能调用的同时,设计一种回溯算法提取其控制流。通过控制流的比对,逻辑漏洞检测器挖掘出异常的程序功能调用。该文开发了一个原型系统,并应用到市面 7 款 Java Web 应用软件(Mvnforum、EShop、JspCart 等)上进行检测,与逻辑漏洞检测专业软件 Waler 对比,该文的方法覆盖面更广,而成本开销更易于接受。该方法在程序语义分析阶段,使用模块分析和数据摘要技术代替指针分析和程序交互分析,导致产生一定程度的误判。

文献[14]集成了若干种方法来减少漏洞检测的误判。首先使用污点分析标记可能的漏洞,随后用基于逻辑回归(Logistic Regression)的数据挖掘方法预测误判的可能性。在该文设计的测试中,系统原型获得 92.1% 的查准率和 92.5% 的准确率,两项指标均优于专业检测软件 Pixy 和 PhpMinerII。在对污点分析发现的漏洞进一步研判时,该方法在人工识别和自动识别之间获得一种平衡。

文献[15]对一种旁路攻击(side-channel)漏洞给出形式化描述,并通过轻量化的静态分析算法对其进行自动检测。该文实现了针对 PHP 应用的系统原型,在 WordPress、Joomla、OpenClinic、Gallery 等 10 个流行的 Web 应用中发现 18 个零日漏洞,其中只有 2 个为误判。对于该文方法发现的漏洞,攻击者可通过跨站脚本攻击的方法获取用户的私密信息,如用药记录、交易记录等。

文献[16]研究了代码风格(Coding Styles)对静态分析软件检测结果的影响。该文选取 phpSAFE、RIPS、WAP 三款开源的检测软件,对六种场景下不同代码风格的源程序进行漏洞检测,发现三款检测软件的行为和结果都是一样的:TP=15, FN=3。该文认为静态分析软件常常产生误判(FP、FN),它们被设计成一心只想着检测特定的漏洞,却没有考虑到源代码的形式和风格。该文建议新一代的静态分析软件应能充分理解所要处理的程序和代码。

文献[17]同时采用污点分析和符号执行两种技

术并进行优化。污点分析时采用更细的粒度,精确记录代码分析过程中生成的对象状态、污点及其聚合(sink)过程,从而减少了过污染、欠污染两种情况。进行符号执行时,通过验证漏洞位置是否可达从而排除一部分错误的漏洞警告。采用上述两方面的措施后,误报率和漏报率均显著减低。在 OWASP Benchmark 数据集上,误报率和漏报均低于专业检测软件 Fortify,如对 SQL 注入漏洞的检测,该文的漏报率和误报率均低于 1%,而 Fortify 的漏报率和误报率高达 35% 和 45%。

文献[18]提出了一种增强的静态污点分析方法,分析源代码并跟踪程序中受污染的输入。它首先检查源代码以找到程序中每个路径的可行性,受污染的变量一直被跟踪分析直到污点聚合。该文还提出一种专门的算法来增强静态分析器,以处理 PHP 中的变量处理函数。通过 SARD 数据集和大型 PHP 程序对该方法进行了评估,结果表明检测 XSS 漏洞的查准率比 RIPS 提高了 26%,检测 SQL 注入漏洞的查准率比 RIPS 提高了 19%。

上述研究的指导意义在于:(1) 由于 Web 应用开发的复杂化和多样化,研究人员应根据开发语言和漏洞类型,建立合适的模型,如文献[13]、文献[15];(2) 对传统静态分析的有关技术进行改进和优化,进一步减少误报、漏报,从而提高漏洞的查准率和查全率,如文献[14]、文献[17]、文献[18]。

3.3 混合方法

静态分析和动态测试是 Web 漏洞检测两种最基本的方法,也有研究将这两种方式混合使用,或者在静态(动态)方法的基础上结合其他方法一并使用^[19],以提高漏洞检测的能力。

文献[20]针对 PHP 应用提出一种静态分析和动态测试相结合的漏洞检测方法。在静态分析阶段,PHP 源代码首先被解析成摘要语法树(Abstract Syntax Tree),以跟踪和分析程序的控制流和数据流。摒弃盲目的模糊测试,根据静态分析提供的辅助信息,用模拟黑客攻击的渗透测试方法进行动态测试。与只用静态分析相比,混合方法同时提高了查全率和查准率;与只用动态测试相比,混合方法提高了查全率和代码的覆盖度(达到 92%)。

文献[21]提出一种静态污点分析和动态模式匹配(Pattern Matching)相结合的漏洞检测方法。程序源代码要进行污点分析,模式匹配是基于简单灵活且功能强大的 YARA(一款恶意软件识别工具)规则。实验表明,该方法对 Web Shell(网页木马)的查全率高达 81.24%,而其他 56 款专业检测软件(Avast、奇虎 360

等)最高的查全率也只是 71%。

文献[22]提出一种动静结合检测二阶 SQL 注入漏洞的方法。借助静态分析,生成控制流图(Control Flow Graph),通过污点分析定位漏洞点;动态测试主要起辅助作用,连接数据库获得元数据并进行漏洞验证测试,提高漏洞查准率。在对三款 PHP 应用的测试中,该方法共找出 9 个二阶 SQL 注入漏洞,明显优于phpvulhunter(找出 0 个)、RIPS(找出 5 个)两个专业检测软件。

文献[23]以动态测试方法为基础,结合高效的 Boyer-Moore 字符串搜索算法,可进行多种 Web 漏洞的检测。在该文设计的实验中,SQL 注入漏洞的误判率低于 Acunetix 和 Netsparker,缓冲区溢出漏洞的误判率低于 w3af、Firefuzzer。

文献[24]在动态测试的基础上,结合改进的渗透测试、攻击签名(Attack Signatures)与监控、实时异常检测三种手段进行 SQL 注入漏洞的检测。在实验对象(含 158 个漏洞)上,找出漏洞 119 个(误报率为 0),明显优于 WebInspect(找出 51 个,误报 61 个)、AppScan(找出 38 个,误报 60 个)。

文献[25-27]在动态测试的基础上融合爬虫技术,并分别结合特征识别(Feature Recognition)、分布式扫描(Distributed Scanning)、应用层指纹识别(Application Fingerprint Recognition)等多项技术进行漏洞检测,在漏洞查全率、查准率以及检测效率多方面得到改善。

4 基于学习的智能化方法

机器学习(ML)、深度学习(DL)是传统静态分析和动态测试之外的一种智能化检测方法。文献[28]总结了深度学习方法在软件漏洞检测中的应用框架(图 2):以源代码或二进制文件为训练样本,对输入数据做预处理后,从文本、序列、语法树等方面进行特征提取,以向量表示作为数据表征,然后通过深度学习的模型进行训练;模型训练好后即可对目标文件进行漏洞检测。

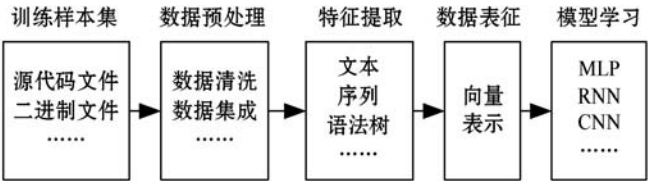


图 2 基于深度学习的漏洞检测框架

近年来 CNN(卷积神经网络)、RNN(循环神经网络)等深度学习模型也逐渐应用到网络安全和 Web 安全漏洞检测中,并取得初步成效(表 5)。

表 5 Web 安全漏洞检测的机器学习方法

论文	年份	针对漏洞	学习模型	模型输入
文献[29]	2019	不特定	LSTM	系统调用
文献[30]	2020	CSRF	ML	HTTP 请求
文献[31]	2020	SQL 注入	LSTM	PHP 源代码
文献[32]	2023	(攻击检测)	LSTM	HTTP 请求

文献[29]针对在 Docker 容器中的 Web 应用,提出一种基于深度学习的漏洞检测方法。通过 Sysdig 管理软件获取 Web 应用服务器上系统调用的信息,采用 LSTM(长短时记忆网络)模型,预测 Web 应用存在的漏洞。实验表明,该方法优于 PCA(Principal Component Analysis)、OCSVM(One-Class Support Vector Machines)等传统机器学习算法。而且,该方法采用无监督学习方式,不需要攻击数据对模型进行训练,可以检测多种已知漏洞,并具有发现未知漏洞的能力。

文献[30]针对跨站请求伪造漏洞提出一种基于机器学习的检测方法。该方法设计了专门的监督学习算法,基于两位安全专家近 6 000 个 HTTP 请求的数据对模型进行训练。在对 20 个真实网站的测试中,检测 CSRF 漏洞的结果为:47 个 TP、12 个 FP、7 个 FN。测试结果显示该方法对 CSRF 漏洞已具有一定的检测能力。

文献[31]针对 PHP SQL 注入漏洞提出一种基于 LSTM 的检测方法。该方法通过一种中间语言(opcodes)对 PHP 代码重新进行分片和文本表示,发挥 LSTM 模型擅长自然语言处理(Natural Language Processing)的优势,测试的查全率、查准率均达到 96% 以上。

文献[32]不直接检测 Web 应用的漏洞,转而检测有害的 HTTP 请求(图 3)。该文基于 Word2vec 的单词嵌入方法和 LSTM 构建深度学习模型进行恶意 HTTP 请求的检测,测试采用 HTTP CSIC 2010 数据集(含 223 585 个测试用例),通过参数的调整和优化,最高获得 81.54% 的查准率和 78.41% 的查全率。该方法虽不检测 Web 漏洞,但由于过滤了恶意的 HTTP 请求,同样起到了安全防护的作用,在漏洞检测之外提出了另外一种思路和方法。

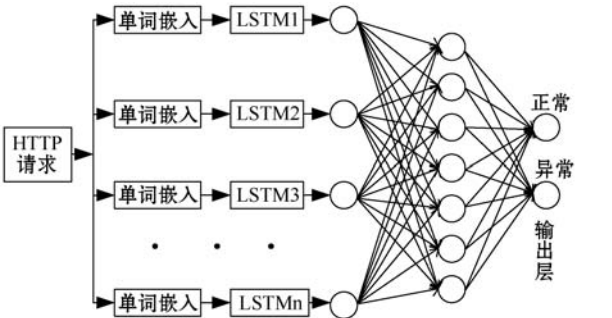


图 3 基于 LSTM 的 HTTP 请求检测

从上述研究看到,机器学习方法也可以对源代码进行分析,寻找可能存在的漏洞;也可以对 Web 应用的输入输出进行分析,并判断是否存在漏洞。但也不能简单认为机器学习能够完全替代传统的静态分析、模糊测试等方法,毕竟上述研究的测试数据都是小范围或针对特定场景的,目前工业界和学术界还缺少通用的、高认可度的 Web 安全漏洞数据供机器学习或深度学习来进行训练和测试。同时可看到,机器学习、深度学习是近年来 Web 安全漏洞检测的热点,也是该领域极其重要的发展方向。

5 各种方法在具体 Web 漏洞中的应用

下面结合跨站脚本、SQL 注入两种具体的漏洞类型对有关检测方法进行分析。

5.1 跨站脚本

XSS 漏洞分为存储型、反射型和 DOM 型三种,表 6 汇总了一些关于跨站脚本漏洞的研究论文。

表 6 XSS 漏洞检测的有关论文

论文	年份	XSS 类型	检测方法	实验数据集
文献[33]	2016	不限	State Machine	XSSed
文献[34]	2019	反射型	HMM	XSSed
文献[35]	2021	不限	OCSVM	权威网站
文献[36]	2021	不限	DLSTM	XSSed
文献[37]	2022	反射型	AST + Fuzzing	DVWA
文献[38]	2022	不限	DCNN-GRU	foxschedler

有的研究^[34,37]只针对反射型 XSS 漏洞进行检测,更多的研究并不指向特定的 XSS 类型。从检测方法看,既有基于状态机(State Machine)的静态分析方法^[33],也有基于 AST + Fuzzing 的动静相结合方法^[37]。近年来更多的文献采用机器学习方法来检测 XSS 漏洞,选用的机器学习模型包括隐马尔可夫链(HMM)^[34]、单分类支持向量机(OCSVM)^[35]、深度长短时记忆网络(DLSTM)^[36]、深度卷积神经网络-门控循环单元(DCNN-GRU)^[38]等。实验所需的数据集以 XSS 漏洞专门网站(xssed.com)居多,也有一些论文基于 DVWA 或其他 Web 漏洞权威网站的数据。由于 XSS 攻击样本的多样性和易变性,传统方法在提升 XSS 漏洞的查全率上遇到瓶颈。即便是机器学习模型,由于在学习训练时的样本覆盖性有限,XSS 漏洞检测的漏报率仍然有待改善。

5.2 SQL 注入

表 7 汇总了一些关于 SQL 注入漏洞的研究论文。

从检测方法看,SQL 注入漏洞检测以静态、动态或动静相结合的传统方法为主。如文献[39]采用黑盒式模糊测试方法来检测 JSP SQL 注入漏洞;文献[40]以数据流依赖关系的 Context Free Grammar(CFG)可达性分析来开展 JSP SQL 注入漏洞的静态分析;文献[42]以 Seed URL、Crawling、Attacking 和 Analysis 四个 Component 来开展动态测试,以提高 PHP SQL 注入漏洞的查准率;文献[43]先进行基于污点的静态分析,再通过动态验证来提高 SQL 注入漏洞的查准率;文献[44]以扩展有限状态机(EFSM)开展动态测试,通过相应的状态转换关系来分析识别 PHP SQL 注入漏洞。机器学习方法也逐渐用于 SQL 注入漏洞检测,文献[41]采用支持向量机(SVM)模型进行了 PHP SQL 注入漏洞检测。一方面由于动态网页开发语言的多样性以及代码量的增大,另一方面由于检测软件或检测方法的不完善,SQL 注入漏洞的检测仍然难以达到 100% 的准确率,提高查全率仍然是 SQL 注入漏洞检测的关键问题。

表 7 SQL 注入漏洞检测的有关论文

论文	年份	Web 开发语言	检测方法	实验数据集
文献[39]	2017	JSP	Fuzzing	BookStore 等
文献[40]	2018	JSP	CFG	—
文献[41]	2019	PHP	SVM	STIVALET
文献[42]	2020	PHP	component-based	HROAD 等
文献[43]	2021	Java EE	动静结合	bodgeit 等
文献[44]	2021	PHP	EFSM	Wavsep 等

从跨站脚本、SQL 注入检测的有关论文来看,基于机器学习、深度学习的检测方法越来越多见,Web 安全漏洞的智能化检测成为发展趋势。

6 当前研究的不足与展望

6.1 不足之处

(1) 未能有效对接工业界需求。当前 Web 安全漏洞检测研究的覆盖面和前沿面未能有效对接工业界的发展状况,有关研究覆盖的漏洞类型不够全面,研究热点的更新也未能赶上工业界的节奏。譬如,安全配置错误、使用含有已知漏洞的组件等漏洞,早已位列 OWASP Top 10 的 2013 版和 2017 版本中,但还没有文献对这些漏洞进行系统和深入的研究。又如,工业界已高度关注加密机制失效、服务器端请求伪造和不安安全设计等漏洞,并将它们列入 OWASP Top 10 的 2021

版,但当前的研究热点仍然停留在 XSS、注入等为数不多的常见漏洞类型上。

(2) 检测的标准化程度不高。Web 安全漏洞的类型众多,同一种漏洞也可能因为软件环境的不同还需要进一步细分,如 SQL 注入漏洞可再划分为 PHP SQL 注入、JSP SQL 注入等。各种文献往往自行设计测试方法,自行选用测试数据,导致难以对不同文献检测方法的效果进行横向对比,孰优孰劣一时难以分辨。测试方法不统一、测试的基准数据未能有效公开和共享,制约了检测的标准化。

(3) 检测的智能化程度不高。有关漏洞的训练数据不多,机器学习方法的应用受到限制。Web 安全漏洞的研究常常是攻击方驱动的,漏洞信息往往掌握在黑客、白帽子、安全测试人员手中。目前关于漏洞的有关数据属于专家知识,工业界和学术界还没有形成通用化、标准化的漏洞数据。机器学习、深度学习恰恰需要使用巨量的、有效的数据对学习模型进行训练,训练数据的缺乏导致机器学习方法难以充分施展。

6.2 研究展望

(1) 加强与工业界的对接。未来,Web 安全漏洞检测的学术研究将加强与工业界发展状况的对接。预计 Web 安全漏洞的研究将会覆盖更多的漏洞类型,研究的热点也会很快转到加密机制失效、服务器端请求伪造和不安全设计等漏洞上。

(2) 实现检测的智能化。国外已有关于将 ChatGPT 用于制造零日漏洞的报道^[45],该攻击例程甚至可躲过专业安全软件的检测,人工智能的强大可见一斑。实际上,如能提供足够的训练样本,诸如 chatGPT 这样的模型可取代动态测试技术实现对 Web 漏洞的智能检测。在新一代人工智能技术的引领下,基于学习的智能化方法将独立或融合应用于 Web 安全漏洞的检测,并逐步实现智能化。

(3) 实现检测的标准化。国内(如 CNVD、CNNVD 等)、国外(如 CVE)的漏洞平台将加强漏洞训练数据生成机制、共享机制的建设,为 Web 安全漏洞的智能化检测创造更有利的条件。根据 Web 开发架构多样化的特点,Web 安全漏洞检测将会从单语言到多语言、单漏洞到多漏洞逐步走向成熟和统一。

7 结 语

Web 应用安全状况堪忧,做好 Web 安全漏洞检测是应对 Web 应用安全的“执牛耳”之举。实际上,Web 安全漏洞不是检测环节产生的,而是 Web 应用开发过

程产生的。期望工业界和学术界相互借鉴,取长补短,为 Web 安全漏洞检测提供更为完善的技术方案。也期望在 Web 应用开发的过程中,对 Web 安全测试环节(人员)与 Web 应用开发环节(人员)进行更好的协调,共同保障 Web 应用的安全。

参 考 文 献

- [1] 2021 年上半年我国互联网网络安全监测数据分析报告[EB/OL]. [2023-03-03]. <https://www.cert.org.cn/publish/main/upload/File/first-half%20%20year%20cybersecurity%20report%202021.pdf>.
- [2] 2021 OWASP TOP 10[EB/OL]. [2023-03-03]. <http://www.owasp.org.cn/OWASP-CHINA/owasp-project/OWASP-TOP10-2021%E4%B8%AD%E6%96%87%E7%89%88V1.0%E5%8F%91%E5%B8%83.pdf>.
- [3] OWASP 2013 Top 10[EB/OL]. [2023-03-03]. <http://www.owasp.org.cn/OWASP-CHINA/owasp-project/download/mobile-top-10-2013-2.pdf>.
- [4] Alsaleh M, Alomar N, Alshreef M, et al. Performance-based comparative assessment of open source web vulnerability scanners[J]. Security and Communication Networks, 2017,2017:1-14.
- [5] Amankwah R, Chen J F, Kudjo P K, et al. An empirical comparison of commercial and open-source web vulnerability scanners[J]. Software-Practice & Experience,2020,50(9):1842-1857.
- [6] Nunes P, Medeiros I, Fonseca J, et al. An empirical study on combining diverse static analysis tools for web security vulnerabilities based on development scenarios[J]. Computing,2019,101(2):161-185.
- [7] Tudela F M, Higuera J R B, Higuera J B, et al. On combining static, dynamic and interactive analysis security testing tools to improve OWASP Top Ten security vulnerability detection in web applications[J]. Applied Sciences-Basel, 2020,10(24):9119.
- [8] Dessiatnikoff A, Akrouf R, Alata E, et al. A clustering approach for web vulnerabilities detection[C]//17th Pacific Rim International Symposium on Dependable Computing, 2011:194-203.
- [9] Hou X Y, Zhao X L, Wu M J, et al. A dynamic detection technique for XSS vulnerabilities[C]//4th Annual International Conference on Network and Information Systems for Computers,2018:34-43.
- [10] 谷家腾,辛阳. 基于动态分析的 XSS 漏洞检测模型[J]. 计算机工程,2018,44(10):34-41.
- [11] 陈亮,李永刚,刘磊,等. 基于特征的电力信息系统注入漏洞检测方法[J]. 计算机工程与设计,2021,42(8):2115-2123.

- [12] Alsaedi A, Alhuzali A, Bamasag O. Effective and scalable black-box fuzzing approach for modern web applications[J]. Journal of King Saud University-Computer and Information Sciences, 2022, 34(10): 10068 – 10078.
- [13] Fang Z J, Zhang Y Q, Kong Y, et al. Static detection of logic vulnerabilities in Java web applications[J]. Security and Communication Networks, 2014, 7(3): 519 – 531.
- [14] Medeiros I, Neves N F, Correia M, et al. Automatic detection and correction of web application vulnerabilities using data mining to predict false positives[C]//23rd International Conference on World Wide Web, 2014: 63 – 73.
- [15] Chen J, Olivo O, Dillig I, et al. Static detection of asymptotic resource side-channel vulnerabilities in web applications[C]//32nd IEEE International Conference on Automated Software Engineering, 2017: 229 – 239.
- [16] Medeiros I, Neves N. Effect of coding styles in detection of web application vulnerabilities[C]//16th European Dependable Computing Conference, 2020: 111 – 118.
- [17] 刘行波, 李源林, 余明俊, 等. 基于污点分析与符号执行的Web漏洞检测[J]. 计算机应用与软件, 2022, 39(11): 297 – 303.
- [18] Marashdih A W, Zaaba Z F, Suwais K. An enhanced static taint analysis approach to detect input validation vulnerability[J]. Journal of King Saud University-Computer and Information Sciences, 2023, 35(2): 682 – 701.
- [19] Liu M, Zhang B Y, Chen W B, et al. A survey of exploitation and detection methods of XSS vulnerabilities[J]. IEEE Access, 2019, 7: 182004 – 182016.
- [20] Zhao J L, Gong R L. A new framework of security vulnerabilities detection in PHP web application[C]//9th International Conference on Innovative Mobile and Internet Services in Ubiquitous Computing, 2015: 271 – 276.
- [21] Le V G, Nguyen H T, Lu D N, et al. A solution for automatically malicious web shell and web application VD[C]//International Conference on Computational Collective Intelligence, 2016: 367 – 378.
- [22] 李鑫, 张维伟, 郑力新. 动静结合的二阶SQL注入漏洞检测技术[J]. 华侨大学学报(自然科学版), 2018, 39(4): 600 – 605.
- [23] Saleh A Z M, Rozali N A, Buja A G, et al. A method for web application vulnerabilities detection by using Boyer-Moore string matching algorithm[J]. Procedia Computer Science, 2015, 72: 112 – 121.
- [24] Antunes N, Vieira M. Designing vulnerability testing tools for web services: approach, components, and tools[J]. International Journal of Information Security, 2017, 16(4): 435 – 457.
- [25] Wang Z. Design and implementation of core modules of Web application vulnerability detection model[C]//11th International Conference on Measuring Technology and Mechatronics Automation, 2019: 10 – 14.
- [26] Wang C, Liu X, Zhou X K, et al. FalconEye: A high-performance distributed security scanning system[C]//17th International Conference on Dependable, 2019: 282 – 288.
- [27] Xu D W, Chen T X, Tan Z H, et al. Web vulnerability detection analyzer based on Python[J]. International Journal of Digital Crime and Forensics, 2022, 14(2): 1 – 17.
- [28] 顾绵雪, 孙鸿宇, 韩丹, 等. 基于深度学习的软件安全漏洞挖掘[J]. 计算机研究与发展, 2021, 58(10): 2140 – 2162.
- [29] Jana I, Oprea A. AppMine: Behavioral analytics for web application VD[C]//ACM SIGSAC Conference on Cloud Computing Security Workshop, 2019: 69 – 80.
- [30] Calzavara S, Conti M, Focardi R, et al. Machine learning for web vulnerability detection: the case of cross-site request forgery[J]. IEEE Security & Privacy, 2020, 18(3): 8 – 16.
- [31] Fidalgo A, Medeiros I, Antunes P, et al. Towards a deep learning model for vulnerability detection on web application variants[C]//13th International Conference on Software Testing, Verification and Validation Workshops, 2020: 465 – 476.
- [32] Alaoui R L, Nfaoui E H. Web attacks detection using stacked generalization ensemble for LSTMs and word embedding[J]. Procedia Computer Science, 2022, 215: 687 – 696.
- [33] Mohammadi M, Chu B, Lipford H R, et al. Automatic web security unit testing[C]//11th International Workshop in Automation of Software Test, 2016: 78 – 84.
- [34] 赵澄, 陈君新. 基于隐马尔可夫模型的反射型XSS检测技术[J]. 浙江工业大学学报, 2019, 47(4): 442 – 447.
- [35] 顾兆军, 李志平, 张礼哲. 基于单分类支持向量机的XSS攻击检测研究[J]. 计算机应用与软件, 2021, 38(6): 299 – 305.
- [36] 张海军, 陈映辉. 语义分析及向量化大数据跨站脚本攻击智检[J]. 山东大学学报(工学版), 2020, 50(2): 118 – 128.
- [37] 郭晓军, 闫宇辰, 吴志浩. 基于AST分析与Fuzzing的反射型XSS漏洞识别模型[J]. 实验室研究与探索, 2022, 41(10): 49 – 53.
- [38] 许丹丹, 徐洋, 张思聪, 等. 基于DCNN-GRU模型的XSS攻击检测方法[J]. 计算机应用与软件, 2022, 39(2): 324 – 329.
- [39] Deepa G, Thilagam P S, Khan F A, et al. Black-box detection of XQuery injection and parameter tampering vulnerabilities in web applications[J]. International Journal of Information Security, 2018, 17(1): 105 – 120.
- [40] Umar K, Sultan A B, Zulzalil H, et al. Formulation of SQL injection VD as grammar reachability problem[C]//International Conference on Information and Communication Technology for the Muslim World, 2018: 179 – 184.

机、平板等接入无线局域网,均可实现对系统的访问。

以上实验证实了保密知识测试系统完全可行,并可支持多设备访问系统。

为进一步验证系统实际运行表现,选择一次保密培训活动作为试点。该培训活动现场共有 36 人参加,培训形式为讲座。结合讲座内容,预先准备相关试题并导入数据库,设定一次测试题量为 10 题。在培训现场搭建好服务器和相关网络,讲座结束后进入测试环节,考察参训人员对讲座内容的理解程度。参训人员通过手机连接现场搭建的无线局域网,通过服务器地址进入系统,操作过程较为流畅。部分参训人员对手机的操作不熟练,在键入服务器地址时输入错误导致无法连接系统,重新输入正确地址后可连接系统。

培训后给参训人员发放问卷 36 份,回收 36 份。问卷结果显示参训人员普遍认为,保密测试过程流畅,测试页面友好,经过测试后对保密要求的理解进一步加深。由此可见,保密测试系统的应用能提升保密培训的效果,且得到参训人员的普遍认可。系统在实际运行中未发生卡顿、死机等情况,表现出较好的稳定性。

4 结 语

本文针对国家教育考试涉密环节多、保密责任重的特点,以提升所有涉密人员保密能力水平为目标,基于 Flask 框架设计实现保密知识测试系统。测试系统的应用是保密培训的有效补充,也是对保密要求的查漏补缺,能较为全面地反映涉密工作人员对保密要求的理解程度。通过单机实验、局域网实验和培训活动试点,保密知识测试系统的可行性和有效性得到验证,符合设计目标。

从实际应用角度出发,测试系统的应用必须结合国家教育考试的具体保密要求和涉密岗位设置,编制足够数量的试题形成题库,避免多次测试后试题大量重复的情况。测试系统未来可以更进一步运用项目反应理论、自适应测试等现代测评理论设计并建设保密题库、优化测试形式,借助认知诊断理论使成绩分析更加科学有效、多维度地评价涉密工作人员在保密方面的相关特质。

从系统架构的角度考虑,系统前后端可以进一步分离,后端服务器提供 API 供前端调用,前端的表现形式可不局限于浏览器,也可以是手机 App 等形式,系统将更加灵活,应用场景更为丰富。

国家教育考试保密知识测试系统是贯彻习近平总书记关于保密工作的指示精神、响应教育部对国家教育考试保密工作要求的一次生动实践,也是学习贯彻

党的二十大精神、统筹推进新时代国家教育考试发展和安全的一次积极探索,系统的使用与推广将有效转变传统保密培训模式,提升专业化考试机构精细化管理水平,促进国家教育考试的保密事业持续健康发展。

参 考 文 献

[1] 路程伊,杨诏旭,谢百治. 本科生信息安全保密素养多元培养模式的探索与实践[J]. 中国医学教育技术,2012,26(6):629-632.

[2] 梁海堂,胡旭千. 检察机关安全保密制度管理系统的构建与设计[J]. 信息安全与通信保密,2011,206(2):43-45.

[3] 郑炯,李鸥,严静. 涉密单位安全保密综合管理平台的设计[C]//中国核工业勘察设计协会. 中国核工业勘察设计协会 2014 年优秀论文集,2014.

[4] 王泽庆. Flask 框架下成品油销售系统设计与实现[D]. 西安:西安电子科技大学,2015.

[5] 张荣,杨树例,李炳武. “互联网+”时代应用系统数据库开发关键技术[J]. 信息与电脑(理论版),2022,34(11):164-166.

[6] 宋永生,黄蓉美. 基于 Python 的学习成绩分析及可视化分析[J]. 信息技术与信息化,2021,261(12):100-102.

[7] 张浩洋,顾丹鹏,陈肖勇. 基于 Vue 的数据管理平台实践与应用[J]. 计算机时代,2022,361(7):66-67,72.

[8] Matplotlib documentation. Backends[EB/OL]. [2023-04-17]. <https://matplotlib.org/stable/users/explain/backends.html>.

[9] Python documentation. Core tools for working with streams[EB/OL]. [2023-04-17]. <https://docs.python.org/3/library/io.html>.

[10] 徐海伟. 局域网服务器搭建探究[J]. 电子制作,2013,240(15):135.

(上接第 9 页)

[41] 胡建伟,赵伟,闫峥,等. 基于机器学习的 SQL 注入漏洞挖掘技术的分析与实现[J]. 信息网络安全,2019(11):36-42.

[42] Aliero M S, Qureshi K N, Pasha M F, et al. Detection of structure query language injection vulnerability in web driven database application[J]. Concurrency and Computation-Practice & Experience,2022,34(13):5936-5941.

[43] 郭帆,范威威. 面向 Java EE 程序的 SQLIA 漏洞分析和验证方法[J]. 计算机科学与探索,2021,15(2):270-283.

[44] 殷博,刘磊,朱静雯,等. 状态驱动的电力信息系统注入漏洞检测模型[J]. 计算机工程与设计,2021,42(3):614-621.

[45] Mulgrew A. I built a zero day virus with undetectable exfiltration using only ChatGPT prompts[EB/OL]. [2023-03-03]. <https://www.forcepoint.com/blog/x-labs/zero-day-exfiltration-using-chatgpt-prompts>.