

基于 HTTP 代理的模糊测试技术研究

孙歆¹, 姚一杨², 卢新岱¹, 刘雪娇³, 吴永涵³

(1. 国网浙江省电力公司电力科学研究院, 浙江 杭州 310014;

2. 国网浙江省电力公司信息通信分公司, 浙江 杭州 310014;

3. 杭州师范大学信息科学与工程学院, 浙江 杭州 311121)

摘要: 常用的 Web 应用测试工具普遍存在着测试功能欠优化、可配置程度较低、测试结果不够智能等问题, 无法较好地辅助测试。进行了基于 HTTP 代理的模糊测试技术研究, 设计了相应的测试工具。该工具应用模糊测试理论, 异步监听实现了 HTTP 代理与服务器、浏览器间的高性能通信; 采用基于伪码的可配置测试用例生成策略, 灵活进行自动化测试; 通过解析网络数据分组, 为用户提供全面的测试分析结果。实验结果表明, 该工具支持主流 Web 应用漏洞检测、测试策略配置, 可检测出目录遍历、SQL 注入、跨站脚本等漏洞。

关键词: 模糊测试; HTTP; 代理; Web 应用; 安全漏洞; 策略

中图分类号: TP391

文献标识码: A

doi: 10.11959/j.issn.2096-109x.2016.00022

Research and implementation of fuzzing testing based on HTTP proxy

SUN Xin¹, YAO Yi-yang², LU Xin-dai¹, LIU Xue-jiao³, WU Yong-han³

(1. Electric Power Research Institute of State Grid, Zhejiang Electric Power Company, Hangzhou 310014, China;

2. Information and Communication Branch, Zhejiang Electric Power Company, Hangzhou 310014, China;

3. School of Information Science and Engineering, Hangzhou Normal University, Hangzhou 311121, China)

Abstract: Most of the security testing tools lack of optimization of testing, configured strategy and intelligent analysis of testing results. These problems lead to the status that these tools can't be used in Web application testing well. A fuzzing testing method towards Web application security based on HTTP proxy was proposed. The high-performance communication between HTTP proxy server and browser through the mechanism of asynchronous monitoring was realized. Configured strategy of testing cases based on pseudo code could help to do flexible and automatic tests. By using multi-dimensional ways to parse the packet, intelligent analysis of testing results was achieved. Experiments show that the tool supports mainstream Web application vulnerabilities detection and configured strategy of testing. It can detect the vulnerabilities such as directory traversal, SQL injection, cross-site scripting.

Key words: fuzzing, HTTP, proxy, Web application, security vulnerability, strategy

收稿日期: 2015-10-23; 修回日期: 2016-01-25。通信作者: 刘雪娇, liuxuejiao0406@163.com

基金项目: 国网浙江省电力公司科技基金资助项目 (No.5211XT14009G); 国家自然科学基金资助项目 (No.61502134); 浙江省重点科技创新团队基金资助项目 (No.2013TD03)

Foundation Items: Science and Technology Program of State Grid Zhejiang Electric Power Company(No.5211XT14009G), The National Natural Science Foundation of China(No.61502134), Zhejiang Provincial Science and Technology Innovation Program (No.2013TD03)

1 引言

国际权威调研机构 Gartner 的最新调查显示, 互联网安全攻击事件 75% 发生在 Web 应用层, 而且这一数字呈快速上升趋势^[1]。Web 应用是黑客攻击的主要目标, 而且超过 $\frac{2}{3}$ 的 Web 站点都存在明

显的安全漏洞, 容易受到来自内部或外部的恶意攻击。2013 年, 由 Symantec 的网站漏洞评估服务对公共网站进行漏洞扫描后发现, 77% 的站点都包含漏洞。其中, 16% 的漏洞被划分为主要漏洞, 可允许攻击者访问敏感数据、改变网站内容或攻击访问者的计算机。这意味着, 如果攻击者要查找攻击的站点, $\frac{1}{8}$ 的站点可让其相对轻松地进行访问。图 1 显示了 2011 年~2013 年每日拦截的 Web 攻击数量, 其中, 2013 年比 2012 年增加了 23%。

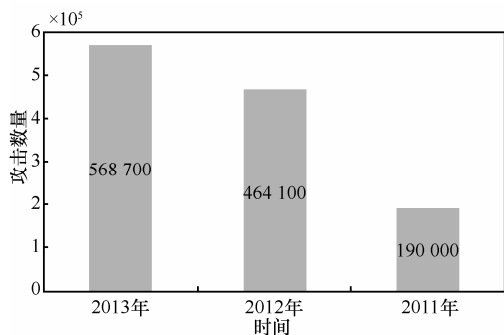


图1 2011 年~2013 年每日拦截的 Web 攻击

Web 应用存在的漏洞和安全隐患很大程度上是由于对用户的某些输入数据缺乏相应的校验或异常处理机制造成的^[2]。然而, 许多企业将大量的资金花费在网络和服务器的安全防护上, 没有从真正意义上关注 Web 应用安全。定期通过模拟黑客攻击的方式对 Web 应用进行安全性测试, 检查网站或 Web 应用的安全程度, 是提升 Web 应用安全状况的重要手段。因此, 对 Web 应用进行全面彻底的安全性测试, 发掘 Web 应用中存在的安全漏洞和潜在的安全风险尤为重要。

模糊测试^[3]是一种自动化或半自动化的漏洞挖掘技术, 通过模糊器构造大量的模糊测试数据发送给目标应用, 并监测异常来发现可能存在的漏洞。模糊测试技术作为软件测试研究中一种新的思路和方法, 近年来开始被应用到 Web 应用测

试领域。利用模糊测试构造大量随机或者经过变异的输入值给 Web 应用, 同时, 通过各种方式监测 Web 程序的异常情况, 查找并分析引起程序异常的原因, 进而发现程序中存在的薄弱环节和安全漏洞。模糊测试的优势在于成本较低、效率较高、可复用性强、能够简化测试设计并捕捉利用其他方法捕捉不到的缺陷。

本文的主要工作和成果如下。

1) 对 Web 安全性测试技术、模糊测试技术以及模糊测试技术在 Web 应用安全性测试技术中的发展与应用进行了研究, 详细分析了目前主流安全测试工具的优缺点, 为基于 HTTP 协议的模糊安全测试工具的实现确立了主导方向。

2) 设计并实现了一个可配置的、智能化程度较高的基于 HTTP 协议的模糊测试工具, 工具包含抓包模块、模糊测试模块、策略配置模块、智能分析模块, 实现了 HTTP 分组实时抓取、模糊测试和测试结果智能化分析等功能。

3) 采用 DVWA (damn vulnerable Web application) 渗透测试平台, 测试了目录遍历、SQL 注入、跨站脚本、请求方式、穷举这 5 种主流的漏洞, 测试表明该工具可以很好地挖掘出 Web 应用的安全漏洞, 且通过策略配置使得安全测试更加简便与智能, 提高了测试效率。

2 相关工作

在 Web 应用安全领域, 发现漏洞的主要技术有静态分析技术和动态分析技术。静态分析技术是一种典型的白盒测试方法, 指在不执行程序的情况下对程序进行分析, 主要包括词法语法分析、模式匹配分析、数据流分析、补丁比较分析和模型化分析。静态分析技术虽然高效、高自动化且程序内部代码具有高度的可视性, 但是通用性较差、误报率较高且依赖被测程序的源代码。动态分析技术从外围或者客户端测试执行程序, 是一种黑盒测试技术, 具有较高的准确性和灵活性。模糊测试作为软件测试研究中一种新的思路和方法, 近年来开始被应用到 Web 应用测试领域。

模糊测试技术源于软件测试技术中的黑盒测试技术, 其基本思想是自动产生和发送大量随机或经过变异的输入值给软件, 一旦发生异常或失

效便可挖掘出软件系统中存在的薄弱环节和安全漏洞^[4-6]。模糊测试最早是由美国 Wisconsin 大学的 Barton Miller 教授于 1989 年提出的, 用来测试 UNIX 应用程序的顽健性和可靠性^[7]。经过近 20 年的发展, 现在的模糊测试技术更接近于灰盒测试。

2.1 Web 安全综合测试工具

目前, 信息安全界用于渗透测试和安全检测的工具种类繁多、功能杂乱、稳定性不高、安全性更得不到保障。不少的安全渗透工具本身藏有后门和木马, 一旦应用会对企业本身的信息安全造成巨大的影响。国外某些厂商或者安全团队开发了较为成熟的渗透测试框架, 例如开源 backtrack、metasploit 和商业的 CANVAS 等, 这些工具功能强大而全面, 但上手较难, 功能多但欠优化。

目前, 较为流行的商业 Web 应用安全测试工具主要有以下几款: SPIDynamics 公司开发的 Web 应用测试工具 WebInspect——以构件的形式提供了图形化的 SPI 模糊器^[8], 并向用户提供了对模糊测试所使用的原始 HTTP 请求的完整控制, 它要求用户具备一定的 HTTP 知识以开发测试用例并解释和识别生成结果, 对测试人员的专业性有较高的要求; BeyondSecurity 开发的 beSTORM 工具^[9]——可以为包括 HTTP 在内的多种协议产生模糊测试集, 支持包括网络层、协议层、文件层、硬件层、DLL 和 API 层在内的所有层面的检测, 在经过用户配置后可实现自动化测试; HP 推出的 Web 应用安全性测试工具 WebInspect^[10]——是准确和自动化的 Web 应用程序和 Web 服务漏洞评估解决方案, 其中附带的一款具有图形化界面的模糊测试工具, 能够对 Web 应用以及 Web 服务器进行模糊测试, 实用性较强^[11]。

以上所提工具均为通用检测工具, 这些工具大多功能集成度高, 适用于批量应用系统安全检测; 但可配置和定制化程度不高, 且缺乏有针对性的模糊测试模块, 无法适应特殊情况下的安全检测。对于熟练的渗透测试人员来说, 若要进行深度渗透, 这些产品就不适用了。

2.2 基于模糊测试方法的 Web 应用安全性研究

近年来, 国际上面向 Web 应用的模糊测试研究中涌现出了许多研究成果。Hammersland 等主

要关注 Web 表单这种类型, 提出了一种半自动化的模糊测试方法来测试 Web 应用处理各种类型用户输入的能力^[12]; Graff 在此基础上提出了一种更加智能化的模糊测试方法, 针对特定 Web 框架的 Web 应用生成合法的模糊数据以避免其被 Web 框架所带的输入验证模块过滤或拒绝^[13]; Stoeckel 等提出了一种通过捕获 Web 浏览器请求, 然后允许用户为一个目标 Web 应用运行一系列预定义的审核, 以识别多种漏洞的方法, 并开发了相应的安全测试工具^[14]。但是, 目前, 国内关于面向 Web 应用的模糊测试技术研究开展的还比较少, 主要研究成果包括: 研究了基于模糊测试方法的 Web 应用的安全性测试技术, 并开发了图形化的 Web 应用模糊测试工具 WFT^[15]; 针对跨站点脚本漏洞, 提出了一种基于服务器过滤删除机制的上下文无关文法模糊数据动态生成算法, 能够对未知漏洞进行有效检测^[16]。

尽管模糊测试技术在 Web 应用安全测试技术中的有效性已经得到了验证, 但是, 不得不承认, 目前的研究工作还存在许多不足。主要问题包括: 测试的自动化程度低, 模糊测试数据生成和结果分析等过程都需要人工参与; 支持的漏洞类型少, 大部分工具只针对几种特定类型的漏洞, 如 Web Fuzz、WFT 等工具——只能够发掘目标 Web 应用中 SQL 注入和跨站点脚本类型的漏洞; 漏洞检测的漏报、误报率较高, 例如 Web Fuzz、SPIKE Proxy 等工具——只是通过在原始请求中简单地插入攻击载荷的方式来生成模糊数据, 漏洞检测上也只是简单地通过查找返回的 Web 页面中是否存在特定的内容; 工具的可扩展性较差, 不能方便地增加对新漏洞类型的检测; 测试结果的展示不够直观, 甚至需要测试人员手动地对数百条记录进行分析来获取哪些测试数据引发了被测对象的安全漏洞。

3 基于 HTTP 代理的模糊测试方法

为了解决目前安全测试工具普遍存在的功能欠优化、可配置程度不高、分析结果不够智能等问题, 本文研究了基于 HTTP 代理的模糊测试方法, 该方法能够支持测试功能扩展, 且攻击策略可配置, 适用于辅助安全测试人员进行深度测试。

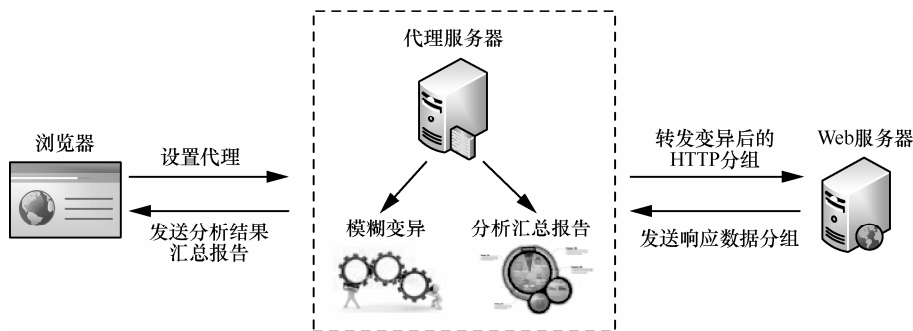


图2 基于 HTTP 代理的 Web 应用安全测试架构

3.1 Web 应用安全测试架构

如图2所示，基于 HTTP 协议的代理服务器构造模糊测试引擎，进行 Web 应用安全性测试。该测试架构包含客户端浏览器、代理服务器及 Web 服务器，通过在客户端浏览器中设置代理服务器地址，默认端口号为 8888，完成对 HTTP 数据分组的分析和转发。

代理服务器是网络的中间实体，位于客户端和服务器之间，扮演“中间人”的角色，透明获取用户的 HTTP 请求数据，支持安全检测人员通过监控、拦截以及修改数据信息的方式对分组进行处理，完成对客户端浏览器 HTTP 请求的转发；同时在模糊测试引擎中，结合策略配置，对 HTTP

请求数据进行模糊变异，生成多样化的测试数据，发送至 Web 服务器；进而接收目标服务器的响应，返回给客户端，分析基于多样测试数据的服务器返回值的变化，进行智能化分析，形成测试结果，以发现 Web 应用中存在的安全漏洞。

3.2 Web 应用安全测试流程

Web 应用模糊测试主要包括 4 个模块，分别是抓包、模糊测试、策略管理、结果分析。抓包主要通过监听网络获取原始测试数据；模糊测试是整个工具的核心，对测试目标进行模糊测试；策略管理提供模糊原始测试数据的数据集；结果分析直观展示模糊测试结果。模糊测试方法的测试流程如图3所示。

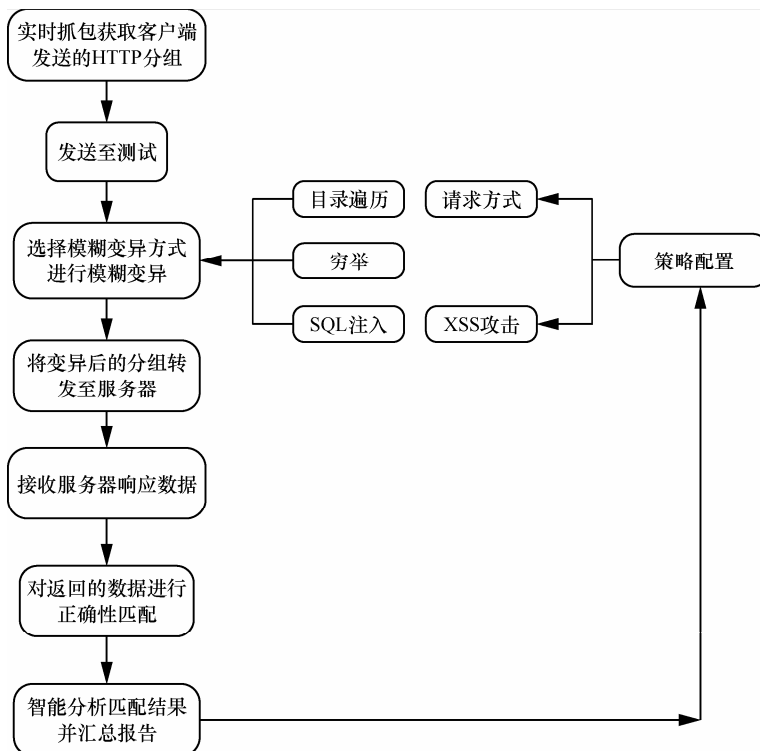


图3 模糊测试方法的测试流程

基于 HTTP 代理服务器，实时抓包获取客户端浏览器发送的 HTTP 请求分组并转发至模糊测试引擎进行测试。在测试引擎中找到需要测试的请求数据，将其发送至测试引擎。然后根据需要查找的特定漏洞类型，选择相应的模糊方式，实施加载策略。测试引擎就会根据策略变异数据分组并发送至目标服务器进行测试，并不停地抓取响应数据分组，基于策略文件中的测试数据生成方法，并结合返回信息代码、返回结果长度等信息对响应数据信息进行智能化分析。最后，将分析结果汇总并显示，同时将发现的新漏洞特征进一步配置到相应的策略文件中。整个过程中，模糊测试工具透明地获取所有交互内容，并将交互产生的数据分组实时显示在页面中供用户查看、分析。

3.3 模糊测试引擎

模糊测试作为一种基于缺陷注入的自动软件测试技术，正在发展成为一种有效的软件安全性测试方法。模糊测试引擎是整个测试工具的核心，结合 Web 安全漏洞攻击原理，基于模糊测试方法实现 Web 应用安全性测试工具。它主要对捕获的数据分组进行模糊测试并结合策略配置和响应结果进行智能化分析。

该测试引擎能够实现多样化、自动化、智能化的模糊测试，并支持多字段模糊，是一种启发式模糊测试的方法，启发测试人员根据现有策略配置新的策略用以发现新的漏洞特征。在测试过程中，用户可以选择一种或多种模糊测试方法，加载相应的策略进行模糊测试。模糊测试引擎会根据策略变异数据分组发送至代理服务器进行全自动化测试，最后，将返回的响应数据分组和测试结果显示于页面。下面重点就策略配置和智能化结果分析模块进行详细介绍。

3.3.1 策略配置

为了使模糊测试工具具有自主判断测试结果的能力，以提高漏洞分析的准确性，本文提出基于伪码可配置的测试用例生成策略。伪码采用 XML 文件进行描述和存储，通过 DOM4J 对 XML 文件进行解析。策略具有相应的解析程序，能够进行灵活的自动化测试，支持 if、for 等普通逻辑词，支持操作符和自定义变量。最后，将 XML 策略转化为 C# 逻辑生成测试用例进行测试。此

外，伪码语言编写规则提供了请求分组数据与返回分组数据的接口。具体策略配置流程如图 4 所示。

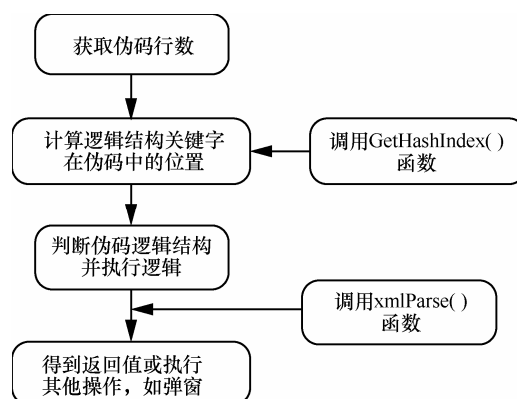


图4 策略配置流程

首先，获取 XML 策略文件的根节点，遍历整个 RuleRow 节点，计算属性 ID 不为空的节点，得到策略文件中的伪码行数；然后获取每个子节点的内容，用 Hashtable 键值对其进行存储；接着获取逻辑结构关键字如 if、else、foreach 等在伪码中的位置，从而判断伪码包含的逻辑结构类型：for 循环逻辑结构中嵌套 if 判断逻辑结构、if 判断逻辑结构中嵌套 for 循环逻辑结构、for 循环逻辑结构、if 判断逻辑结构等；最后根据所属逻辑结构，递归地调用 xmlParse 函数计算伪码各行或者执行其他操作，如弹窗显示信息等。

具体伪码语言编写规则如下。

- 1) 配置语言为遵循自定义语法规则的伪码，且伪码编写一律用英文小写字母或数字。
- 2) 自定义变量必须标记&，具有特定含义的变量如表 1 所示。
- 3) 可用运算符、操作符、关键字如表 2 所示。
- 4) 配置规则只能包含 2 种特殊逻辑结构：if 判断和 for 循环。2 种逻辑结构可以单独使用，也可以单层嵌套使用。

本文主要针对系统中默认配置的目录遍历、SQL 注入、跨站脚本、请求方式、穷举这 5 种漏洞，并以 XML 策略文件的形式存储其相对应的模糊方式，其他漏洞可根据需求自定义配置实现。测试人员可在测试前对需要插入的模糊内容及结果的判断进行配置，测试时只需加载相应文件。模糊测试工具会自动读取选定的 XML 文件中的内容，据此进行模糊变异并将模糊后的数据分组

表 1 特定含义的变量

变量	含义	变量	含义
<i>request.host</i>	请求主机	<i>attack.request.host</i>	攻击请求主机
<i>request.url</i>	请求地址	<i>attack.request.url</i>	攻击请求地址
<i>request.method</i>	请求方式	<i>attack.request.method</i>	攻击请求方式
<i>request.length</i>	请求数据长度	<i>attack.request.length</i>	攻击请求数据长度
<i>request.headers</i>	请求头	<i>attack.request.headers</i>	攻击请求头
<i>request.post</i>	请求数据	<i>attack.request.post</i>	攻击请求数据
<i>request.body</i>	请求体	<i>attack.request.body</i>	攻击请求体
<i>response.post</i>	响应数据	<i>attack.response.post</i>	攻击响应数据
<i>response.length</i>	响应数据长度	<i>attack.response.length</i>	攻击响应数据长度
<i>response.headers</i>	响应头	<i>attack.response.headers</i>	攻击响应头
<i>response.body</i>	响应体	<i>attack.response.body</i>	攻击响应体

表 2 可用运算符、操作符、关键字

运算符	释义	操作符	释义	关键字	释义
=	赋值	==	等于	show()	弹窗
+=	先加再赋值	!=	不等于	.contains()	包含
		&&	和	HIT	命中，表示找到漏洞
			或	UNHIT	未命中，表示未找到漏洞
		>	大于		
		<	小于		
		>=	大于等于		
		<=	小于等于		

转发至目标服务器。自动解析并执行策略中的对测试结果的判断语句获取响应数据分组后，工具自动解析并执行策略中对测试结果的判断语句，给出漏洞测试结果。

3.3.2 结果分析

本模块对模糊测试的结果进行智能化分析。工具通过抓取返回的 session 值，根据返回信息代码、返回结果长度等对返回信息的正确性进行匹配，对每类返回码出现次数进行统计，并将返回码不同、唯一或命中标志为 true 的数据设置成高亮显示。同时将结果进行统计汇总，最终生成分析汇总报告并展现给用户。

工具自动完成对测试结果的判断，节省了测试人员对漏洞的判断，提高了测试的效率，体现了工具的智能性。此外，当根据旧的漏洞特征发现新的漏洞时，可查看返回的响应数据信息，查找是否有

新的漏洞特征，并将新的漏洞特征配置到策略文件中。随着漏洞特征逐步地被发现，策略文件中对漏洞特征的判断也会相应增多，即同一个漏洞可能存在几个不同的漏洞特征。那么当漏洞只显示其中一个特征时，工具仍能检测到该漏洞。漏洞匹配特征可添加这一功能大大提高了工具查找漏洞的准确度，同时也体现了工具的智能性。

4 模糊测试工具

基于前面的设计方法，实现了基于 HTTP 代理的模糊测试工具，并进行了详细的测试。

4.1 测试工具实现

该工具采用 C/S 架构，使用 C#语言开发，实现了基于 HTTP 代理的模糊测试工具。基于 HTTP 代理的模糊测试工作界面如图 5 所示，该工具获取客户端的 HTTP 分组，对 HTTP 分组进行模糊

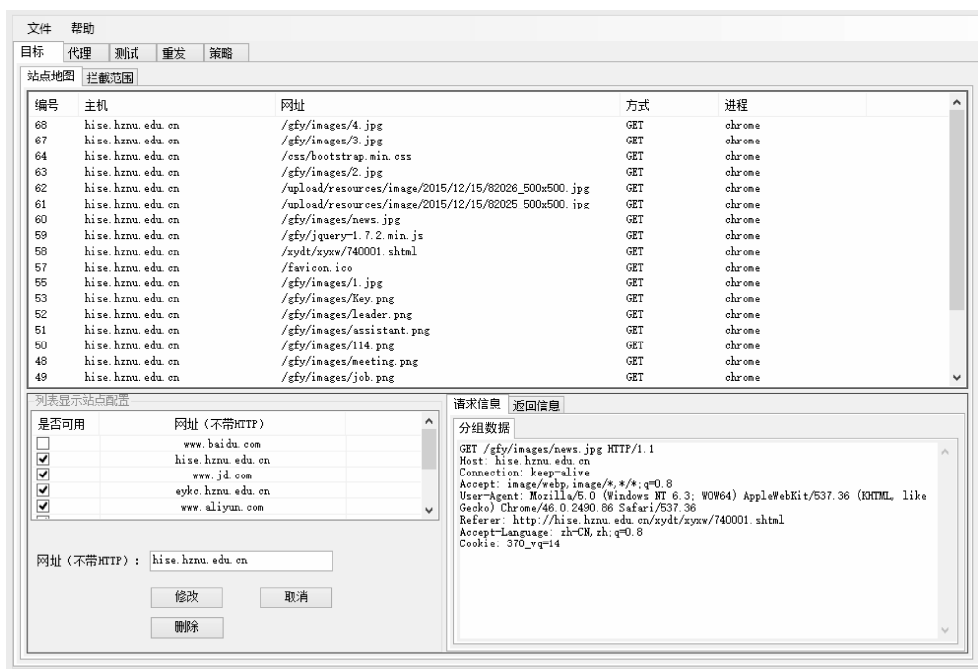


图5 基于 HTTP 代理的模糊测试工具界面

变异并转发变异后的分组至服务器，对服务器返回的响应数据分组进行智能分析并形成汇总报告。该工具可供用户选择测试类型，生成 Web 应用模糊测试数据，执行相应的测试并捕捉 Web 应用模糊测试结果。

在底层通信方面，工具采用异步 socket、多线程、端口监听、session 等技术实现了高性能、高并发的 HTTP 通信。工具开启代理时，首先实例化一个代理（proxy）类，然后调用这个实例的 start 方法来启动 HTTP 代理服务，start 方法会不停地异步监听本机或者服务器的端口（该端口可由用户自行设置），具体原理是创建一个 socket 对象并绑定监听的端口，一旦监听到，系统就会从线程池中取出一个线程，并在这个线程里构造一个 session 对象。session 对象代表客户端与服务器的一次会话过程，在基于 HTTP 代理的通信中，一次 session 会话主要有以下几个步骤：

- 1) 从客户端读请求；
- 2) 拦截请求进行模糊变异或者直接通过，转发至目标服务器；
- 3) 从目标服务器读取响应信息；
- 4) 包装接受到的响应信息并返回给客户端。

4.2 实验测试

为了证明 Web 应用安全性测试工具的有效

性，选取并部署了 DVWA^[17]，一个预置 Web 常见漏洞的渗透测试演练平台作为测试对象进行实验。DVWA 作为待测试的 Web 应用服务器，地址为 <http://192.168.1.125:8082/dvwa/>。

测试中，首先在客户端浏览器中配置代理服务器。例如在 Google Chrome 浏览器上配置代理服务器，设置 IP 地址为 127.0.0.1，端口为 8888。

1) 代理获取 HTTP 请求数据

设置站点信息如图 6 所示，在代理服务器的“站点配置”栏中，添加目标测试网站的地址，这里为 192.168.1.125。这样，配置 HTTP 代理服务器后，所有客户端浏览器访问的 www 站点请求都会先发送至代理服务器，再由代理服务器转发至目标服务器，目标服务器的响应结果也由代理服务器转发至客户端浏览器。



图6 设置站点信息

本测试工具既支持代理方式获取 HTTP 请求数据,也支持用户手动构造 HTTP 请求,只要手动构造的数据分组符合标准的 HTTP 请求头格式。工具的 GET 方法请求分组头的基本格式如图 7 所示。

```

分组数据
GET / HTTP/1.1
Accept: text/html, application/xhtml+xml, */*
Accept-Language: zh-CN
User-Agent: Mozilla/5.0 (compatible; MSIE 10.0; Windows NT 6.1; WOW64; Trident/6.0)
Host: www.baidu.com
DNT: 1

```

图7 方法请求头

2) 执行测试

选择数据分组中某部分数据,加载策略文件,采用随机法和变异法生成模糊测试数据,然后添加策略,加载请求至模糊测试,进行模糊测试。测试中,主要对目录遍历、SQL 注入、XSS 攻击、请求方法、穷举攻击 5 种主流的 Web 漏洞进行测试。每一种类型的漏洞测试都包含了策略配置、抓包、模糊测试、结果分析等步骤。

4.3 有效性测试

下面以目录遍历漏洞测试为例,给出具体测试流程。

1) 目录遍历测试

为了进行目录遍历,编写伪码文件,如图 8 所示。

```

{ foreach &.../etc/passwd in 10 }<
{/response}<
{ if attack.response.body.contains(utf-8.root) }<
HIT<
{/if}<

```

图8 目录遍历测试策略文件

将选中的请求发送至测试模块,并添加目录遍历模糊测试策略,策略内容为 {if attack.response.body.contains(utf-8.root)},即判断测试结果中的返回主体中是否包含“root”字符串,并指定结果的编码是 utf-8,同时进行 10 次目录遍历。如果存在目录遍历漏洞,则将返回结果高亮显示,如图 9 所示。

目录遍历测试的结果如图 10 所示,所有的高亮数据的返回主体中都包含了“root”字符串,获取到了 Linux 下的系统文件 etc/passwd,表明该 Web 应用存在目录遍历漏洞。

2) SQL 注入

将测试包抓取并添加 SQL 注入模糊测试,这是一个用户登录的数据分组,用户名为空,密码部分添加了注入策略 {if attack.response.headers.contains(200)},如图 11 所示。

SQL 注入测试的结果如图 12 所示,根据返回信息,得知登录成功,判断该测试数据分组利用 SQL 注入漏洞绕过了登录验证。

3) XSS 跨站

抓取包发送至测试模块并添加 XSS 跨站策略 xssinjection.xml,策略的内容是 alert (document.cookie),即输出 cookie。XSS 测试结果如图 13 所示,无过滤输出了脚本,说明存在 XSS 跨站脚本,为反射型。

4) 字典攻击

抓取登录的数据分组进行字典攻击,分别在用户名和密码处添加字典文件。字典攻击测试结果如图 14 所示,结果显示成功匹配了用户名和密码。

文件 帮助

目标 代理 测试 重发 策略

站点地图 拦截范围

编号	主机	网址	方式	进程
63	192.168.1.122	/dwa/dwa/css/main.css	GET	ieexplore
62	192.168.1.122	/dwa/dwa/images/logo.png	GET	ieexplore
61	192.168.1.122	/dwa/dwa/js/dwaPage.js	GET	ieexplore
60	192.168.1.122	/dwa/vulnerabilities/Fi/?page=include.php	GET	ieexplore
59	192.168.1.122	/dwa/vulnerabilities/Fi/?page=include.php	GET	ieexplore
55	192.168.1.122	/dwa/dwa/js/dwaPage.js	GET	ieexplore
57	192.168.1.122	/dwa/dwa/images/logo.png	GET	ieexplore
56	192.168.1.122	/dwa/dwa/css/main.css	GET	ieexplore
54	192.168.1.122	/dwa/vulnerabilities/capcha/	GET	ieexplore
53	192.168.1.122	/dwa/vulnerabilities/capcha/	GET	ieexplore
52	192.168.1.122	/dwa/dwa/css/main.css	GET	ieexplore
51	192.168.1.122	/dwa/dwa/js/dwaPage.js	GET	ieexplore
50	192.168.1.122	/dwa/dwa/images/logo.png	GET	ieexplore
49	192.168.1.122	/dwa/vulnerabilities/csr/	GET	ieexplore
48	192.168.1.122	/dwa/vulnerabilities/csr/	GET	ieexplore
46	192.168.1.122	/dwa/dwa/js/dwaPage.js	GET	ieexplore
47	192.168.1.122	/dwa/dwa/images/logo.png	GET	ieexplore

图9 数据分组捕获界面

编号	主机	请求信息	返回码	返回长度
0	192.168.1.122	GET /dvwa/vulnerabilities/fi/?page=../../../../../../../../ 200 OK	200 OK	5594
1	192.168.1.122	GET /dvwa/vulnerabilities/fi/?page=../../../../../../../../ 200 OK	200 OK	5594
2	192.168.1.122	GET /dvwa/vulnerabilities/fi/?page=../../../../../../../../etc/ 200 OK	200 OK	5594
3	192.168.1.122	GET /dvwa/vulnerabilities/fi/?page=../../../../../../../../etc/pa 200 OK	200 OK	5594
4	192.168.1.122	GET /dvwa/vulnerabilities/fi/?page=../../../../../../../../etc/passwd 200 OK	200 OK	4079
5	192.168.1.122	GET /dvwa/vulnerabilities/fi/?page=../../../../../../../../etc/passwd 200 OK	200 OK	4073
6	192.168.1.122	GET /dvwa/vulnerabilities/fi/?page=../../../../../../../../etc/passwd 200 OK	200 OK	4067
7	192.168.1.122	GET /dvwa/vulnerabilities/fi/?page=../../../../../../../../etc/passwd 200 OK	200 OK	4055
8	192.168.1.122	GET /dvwa/vulnerabilities/fi/?page=../../../../../../../../etc/passwd 200 OK	200 OK	4061
9	192.168.1.122	GET /dvwa/vulnerabilities/fi/?page=etc/passwd 200 OK	200 OK	4049

请求信息	返回信息	网页信息	分析结果
------	------	------	------

```

HTTP/1.1 200 OK
Date: Tue, 02 Sep 2014 09:21:12 GMT
Server: Apache/2.4.9 (Unix) OpenSSL/1.0.1g PHP/5.5.11 mod_perl/2.0.8-dev Perl/v5.16.3
X-Powered-By: PHP/5.5.11
Expires: Tue, 23 Jun 2009 12:00:00 GMT
Cache-Control: no-cache, must-revalidate
Pragma: no-cache
Content-Length: 5594
Keep-Alive: timeout=5, max=100
Connection: Keep-Alive
Content-Type: text/html; charset=utf-8

root:x:0:0:root:/root:/bin/bash

```

图 10 目录遍历测试结果

模糊测试参数设置

选择测试类型:

```

POST /sgfuzz/login.php HTTP/1.1
Accept: text/html, application/xhtml+xml, */*
Referer: http://192.168.1.132/sgfuzz/
Accept-Language: zh-Hans-CN, zh-Hans;q=0.5
User-Agent: Mozilla/5.0 (Windows NT 6.3; WOW64; Trident/7.0; rv:11.0) like Gecko
Content-Type: application/x-www-form-urlencoded
Connection: Keep-Alive
Content-Length: 50
DNT: 1
Host: 192.168.1.132
Pragma: no-cache

username=@password=[sqlinjection.xml]&submit=++%E7%99%BB+%E5%BD%A5++

```

图 11 SQL 注入内容

编号	主机	请求信息	返回码	返回长度
0	hise.hznu.edu.cn	GET /gfy/jquery-1.7.2.min.js	返回码还...	长度...

请求信息	返回信息	网页信息	分析结果
------	------	------	------

```

Server: Apache-Coyote/1.1
Content-Type: text/html; charset=utf-8
Content-Length: 1024
Date: Wed, 30 Dec 2015 06:23:03 GMT

<html><head><title>Apache Tomcat/6.0.32 - Error report</title><style><!--H1 {font-family:Tahoma,Arial,sans-serif;color:white;background-color:#525D76;font-size:22px;} H2 {font-family:Tahoma,Arial,sans-serif;color:white;background-color:#525D76;font-size:16px;} H3 {font-family:Tahoma,Arial,sans-serif;color:white;background-color:#525D76;font-size:14px;} BODY {font-family:Tahoma,Arial,sans-serif;color:black;background-color:white;} B {font-family:Tahoma,Arial,sans-serif;color:white;background-color:#525D76;} P {font-family:Tahoma,Arial,sans-serif;background:white;color:black;font-size:12px;} A {color : black;} A.name {color : black;}HR {color : #525D76;}--></style> </head><body><h1>HTTP Status 404 - /gfy/jquery-1.7.2.min.js</h1><HR size="1" noshade="noshade"><p><b>Status report</b></p><p><b>message</b></p><p><b>description</b></p><p><b>The requested resource (/gfy/jquery-1.7.2.min.js) is not available.</p></p><HR size="1" noshade="noshade"><h3>Apache Tomcat/6.0.32</h3></body></html>

```

图 12 SQL 注入测试结果

编号	主机	请求信息	返回码	返回长度
0	192.168.1.122	GET /dwa/vulnerabilities/xss_r/?name=<script>alert(document.cookie)</script>	200 OK	4509

请求信息	返回信息	网页信息	分析结果
<pre> <div class="vulnerable_code_area"> <form name="XSS" action="#" method="GET"> <p>What's your name?</p> <input type="text" name="name"> <input type="submit" value="Submit"> </form> <pre>Hello <script>alert(document.cookie)</script></pre> </pre>			

图13 XSS 测试结果内容

编号	主机	请求信息	返回码	返回长度
0	192.168.1.132	POST /sgfuzz/login.php	200 OK	258
1	192.168.1.132	POST /sgfuzz/login.php	200 OK	87
2	192.168.1.132	POST /sgfuzz/login.php	200 OK	87
3	192.168.1.132	POST /sgfuzz/login.php	200 OK	87
4	192.168.1.132	POST /sgfuzz/login.php	200 OK	87
5	192.168.1.132	POST /sgfuzz/login.php	200 OK	87
6	192.168.1.132	POST /sgfuzz/login.php	200 OK	87
7	192.168.1.132	POST /sgfuzz/login.php	200 OK	87
8	192.168.1.132	POST /sgfuzz/login.php	200 OK	87
9	192.168.1.132	POST /sgfuzz/login.php	200 OK	87
10	192.168.1.132	POST /sgfuzz/login.php	200 OK	87

请求信息	返回信息	网页信息	分析结果
<pre> HTTP/1.1 200 OK Date: Tue, 02 Sep 2014 12:43:36 GMT Server: Apache/2.2.25 (Win32) PHP/5.4.27 X-Powered-By: PHP/5.4.27 Expires: Thu, 19 Nov 1981 08:52:00 GMT Cache-Control: no-store, no-cache, must-revalidate, post-check=0, pre-check=0 Pragma: no-cache Content-Length: 258 Keep-Alive: timeout=5, max=100 Connection: Keep-Alive Content-Type: text/html zbfzbf 欢迎你! 进入 用户中心
用户信息:
用户ID: 1
用户名: zbf
密码: 123456
返回登录
select * from user where username='zbf' and password='123456' limit 1 </pre>			

图14 字典攻击测试结果

经过对目录遍历、SQL 注入、XSS 攻击、请求方式、穷举这 5 种主流漏洞的测试，该测试工具能够精确地找出各种漏洞并给出相应提示，帮助测试人员节省判断漏洞的时间，从而在一定程度上提高测试效率。

4.4 结果分析与比较

本测试工具根据测试需求，生成多样化的模糊测试数据，发送至目标服务器，记录目标服务器的返回信息，对其进行智能化分析，自动为用户生成分析结果报告。本测试工具主要根据以下几个方面进行分析：

- 1) 根据请求返回的信息代码对返回信息的正确性进行匹配；
- 2) 根据请求返回结果的数据长度对返回信息的正确性进行猜测匹配；
- 3) 根据请求返回的详细结果内容进行对比，从而对返回信息的正确性进行猜测匹配；

- 4) 自动智能判断检测结果的有效性，支持正则表达式等匹配方式。

SQL 注入和目录遍历模糊测试的分析结果如图 15 所示。

请求信息	返回信息	网页信息	分析结果
<pre> 本次使用了SQL注入模糊测试方法，分析结果如下： 测试结果的返回码统计如下： 出现200 OK共计1次 测试结果的长度统计如下： 出现无返回长度共计1次 </pre>			

请求信息	返回信息	网页信息	分析结果
<pre> 本次使用了目录遍历模糊测试方法，分析结果如下： 测试结果的返回码统计如下： 出现404 Not Found共计10次 测试结果的长度统计如下： 出现1024共计10次 </pre>			

图15 测试分析结果

选取 2 个常见的 Web 应用测试工具 WebFuzz、BurpSuite，作为比较对象，在策略配置、分组重发、测试结果分析等方面进行了比较，如表 3 所示。

表3 与 WebFuzz、BurpSuite 测试工具比较

工具名	策略配置	测试对象	分组重发	智能结果分析
本文测试工具	基于伪码的可配置的测试策略	能根据需求拦截列表中的若干请求	工具自动可对有疑问的数据分组进行重发	工具维护一个过滤列表, 显示筛选后的数据分组
WebFuzz ^[18]	只能对特定参数进行设置	拦截单一网页请求	无	显示所有 HTTP 反馈信息
BurpSuite ^[19]	采用 Intruder 选择特定参数进行策略配置	拦截若干请求	手动重新发送单个 HTTP 请求	显示所有 HTTP 反馈信息

WebFuzz 需要用户对 HTTP 协议有较深的理解, 且只能拦截单一网页请求, 对特定参数进行测试, 同时不能对分组进行重发, 最重要的是该工具只能显示所有 HTTP 反馈信息, 智能化分析程度不够高。BurpSuite 虽然能够拦截若干请求, 但是也只能选择特定的参数(例如 attack type)等, 把 post 请求分组发送到 Intruder 进行破解, 只显示所有 HTTP 反馈信息, 需要手动重新发送分组。综合分析, 可知本测试工具能够具有可配置策略、全面的结果分析、自动分组重发的特点。

5 结束语

本文按照基于 HTTP 代理的 Web 应用安全性测试技术的研究思路, 分析了目前常用的 Web 应用安全测试工具的不足, 研究并实现了面向 Web 应用的模糊测试方法, 开发了一个可配置的、智能化程度较高的模糊测试工具。该工具的策略配置和智能分析模块, 大大提高了工具对漏洞检测的准确性和针对性, 并使其具备良好的学习能力, 方便用户对测试结果的理解和分析。实验表明, 该工具能有效地检测出目录遍历、SQL 注入、跨站脚本、请求方式等漏洞, 并能对测试结果进行智能化分析, 帮助测试人员节省判断漏洞的时间, 从一定程度上提高了测试效率。当然, 该工具还存在需要改进之处: 在策略配置中伪码并不能够完全支持所有的逻辑词、操作符, 即在使用伪码配置策略的过程中可能会受到一定的限制。在后续的研究和应用中应考虑加强策略管理模块, 使其更加灵活易用。

参考文献:

- [1] 崔光耀. 启明星辰 IPS 突破 Web 防御僵局[J]. 信息安全与通信保密, 2008(6):33.
- [2] CUI G Y. Venus IPS breakthrough Web defense deadlock[J]. Information Security and Communications Privacy, 2008(6):33.
- [3] HUANG Y W, HUANG S K, LIN T P, et al. Web application security assessment by fault injection and behavior monitoring[C]//The 12th International World Wide Web Conference. New York. c2003:148-159.
- [4] MILLER B. Project: operating system utility program reliability-the fuzz generator[EB/OL]. <http://pages.cs.wisc.edu/~bart/fuzz/fuzz.html>.
- [5] OEHLERT P. Violating assumptions with fuzzing[J]. IEEE Security and Privacy, 2005, 3(2): 58-62.
- [6] SUTTON M, GREENE A, AMINI P. Fuzzing: brute force vulnerability discovery[M]. 2007.
- [7] 吴志勇, 王红川, 孙乐昌, 等. Fuzzing 技术综述[J]. 计算机应用研究, 2007, 27(3):829-832.
- [8] WU Z Y, WANG H C, SUN L C, et al. Survey on fuzzing[J]. Application Research of Computers, 2007, 27(3):829-832.
- [9] MILLER B P, FREDRIKSEN L, SO B. An empirical study of the reliability of UNIX utilities[J]. Communications of the ACM, 1990, 33(12):32-44.
- [10] LIN J C, CHEN J M. An automated mechanism for secure input handling[J]. Journal of Computers, 2009, 4(9):837-844.
- [11] ERIKSSON B I, DAHL O E, BÜLLER R, et al. A new oral direct thrombin inhibitor, dabigatran etexilate, compared with enoxaparin for prevention of thromboembolic events following total hip or knee replacement: the BISTRO II randomized trial[J]. Journal of Thrombosis and Haemostasis, 2005, 3(1):103-111.
- [12] SPI Dynamics. Web application security assessment[R]. 2003.
- [13] 于莉莉, 杜蒙杉, 张平, 等. Web 安全性测试技术综述[J]. 计算机应用研究, 2012, 29(11): 4000-4002.
- [14] YU L L, DU M S, ZHANG P, et al. Survey on Web security testing technologies[J]. Application Research of Computers, 2012, 29(11): 4000-4002.
- [15] HAMMERSLAND R, SNEKKENES E. Fuzz testing of Web applications[EB/OL]. <http://rune.hammersland.net/tekst/fuzzing-article.pdf>.
- [16] GRAAF M D. Intelligent fuzzing of Web applications[D]. Amsterdam: University of Amsterdam, 2009.
- [17] STOECKEL D M, STELZER E A, DICK L K. Evaluation of two spike-and-recovery controls for assessment of extraction efficiency in microbial source tracking studies[J]. Water Research, 2009, 43(19): 4820-4827.
- [18] 都娟. 基于模糊测试方法的 Web 应用安全性测试技术的研究及其工具实现[D]. 上海: 华东师范大学, 2011.
- [19] DU J. Web application security testing technique based on fuzzing and implementation of the tool[D]. Shanghai: East China Normal University, 2011.
- [20] 刘为. 基于模糊测试的 XSS 漏洞检测系统研究与实现[D]. 长沙: 湖南大学, 2010.
- [21] LIU W. Research on and implementation of XSS vulnerability

mining based on fuzzing technology[D]. Changsha:Hunan University, 2010.

[17] Damn vulnerable Web application[EB/OL]. <http://www.dvwa.co.uk/>.

[18] Fuzzing: brute force vulnerability discovery[EB/OL]. <http://www.fuzzing.org/>.

[19] Web security[EB/OL]. <https://portswigger.net/burp/>.



卢新岱(1988-), 男, 浙江奉化人, 国网浙江省电力公司工程师, 主要研究方向为信息安全。

作者简介:



孙歆(1981-), 男, 浙江杭州人, 国网浙江省电力公司高级工程师, 主要研究方向为 Web 安全。



刘雪娇(1984-), 女, 河南安阳人, 杭州师范大学讲师, 主要研究方向为网络安全评估、应用安全。



姚一杨(1984-), 男, 浙江杭州人, 国网浙江省电力公司工程师, 主要研究方向为网络安全。



吴永涵(1996-), 男, 浙江温州人, 杭州师范大学本科生, 主要研究方向为信息安全。