**基于NGINX\_LUA\_MOUDLE模块的服务器安全软件**

**的设计与实现**

李政，高明哲，董本浩，徐淑琳，马冲

（山东理工大学，淄博，255000）

摘要:本文主要分析了当前背景下的服务器所处的安全环境形势。通过我们的这款软件有效的来解决服务器的非法参数过滤，上传文件内容审查，常见的OWASP攻击语句，资源防盗，以及自身系统的漏洞扫描，暴力破解日志分析和最新公开的漏洞exp拦截等功能保护服务器安全的措施。

关键词：安全；WAF；资源防盗；后门查杀；漏洞扫描

1. **引言及概述**

随着互联网的迅速发展以及大众创新万众创业的提出，使得许多企业如雨后春笋般出现，网站便成为各大企业的门面与业务办理的主要阵地，许多业务的办理变得简单迅速。但随之信息安全问题也成为大众瞩目的焦点。

根据360互联网安全中心的数据显示，全网各类针对网站的攻击呈上升趋势，其中一天中就有共计75185个网站遭到攻击，漏洞攻击达218.4万次，DDOS攻击次数达到3.6亿次。医疗，学校，政府，企业官网是高危漏洞比例存在较高的网站类型，此类网站存有大量的有价值的信息，因此格外受到黑客青睐。从补天平台的漏洞利用分布中可以看到SQL注入占据了半壁江山，说明黑客通过SQL注入攻击方式对网站进行暴库、脱库以获取存放在网站数据库中的敏感业务数据信息和网站管理员账户信息是攻击者对网站进行攻击的重要目的和主要攻击方式。这也和SQL注入工具更加自动化、智能化并且非常容易获取有关。

根据360网站安全监测数据发现，国内56%的中小学网站存在安全漏洞，其中32%的网站存在高危安全漏洞。存在高危安全漏洞的网站，很容易被入侵、拖库和篡改。检测结果还显示，7.2%的中小学网站已经被篡改。所以网络服务器亟需一款迅速有效服务器防火墙来阻挡来自外界的各种安全攻击。我们开发的安全软件（SerSafer）就应运而生了。

安全狗是基于nginx网络服务器的一款网站防火墙，由于采用Nginx自带的LUA扩展模块可以使所有请求数据在到达服务器之前先有安全狗进行数据分析与清洗，找出其中的所有对于服务器安全构成威胁的数据与注入语句，尽可能消除安全隐患。淘宝的agentzh和chaoslawful开发的OPENRESTY通过将lua解释器集成进Nginx，可以采用lua脚本实现业务逻辑，由于lua的紧凑、快速以及内建协程，所以在保证高并发服务能力的同时极大地降低了业务逻辑实现成本。故本产品默认安装在OPENRESTY上，如用户需求不同可自行下载LUA解释器以及编译安装带有LUA模块的nginx服务器来使用本软件。

1. **软件功能分析**

SerSafer主要分为三大功能，WAF，网站后门及病毒查杀以及漏洞扫描，如图1-1所示。

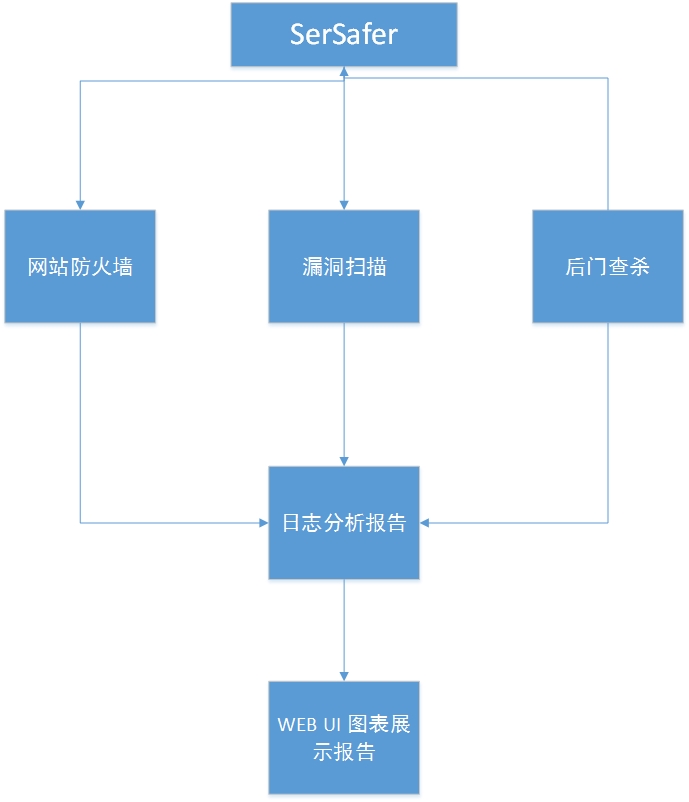
****

图2-1

**2.1 网站防火墙**

网站防火墙是SerSafer的核心组件，其具体功能如图2-2所示

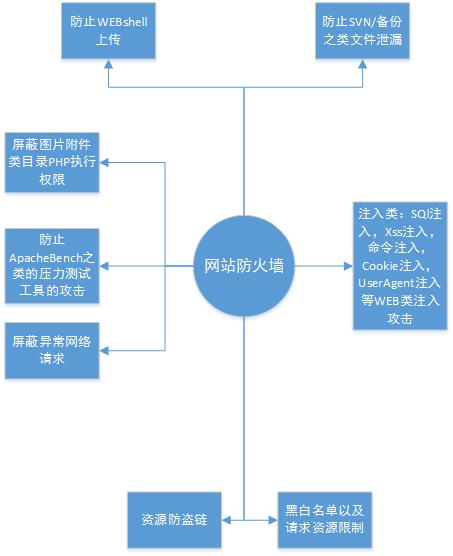


图2-2

**网站防火墙构成：**由解析HTTP请求（协议解析模块），规则检测（规则模块），做不同的防御动作（动作模块），并将防御过程（日志模块）记录下来。所以本文中的WAF的实现由五个模块(配置模块、协议解析模块、规则模块、动作模块、错误处理模块）组成。

**网站防火墙参数过滤方式：**基于正则表达式匹配匹配非法及有害参数从而选择是否过滤，**根据网络上出现的绕过编码等多种绕过方式制定相应解码方法，**从而保证能够拦截绝大多数非法以及利用绕过脚本将非法字符编码的有害请求。

**功能列表：**

1. 支持IP白名单和黑名单功能，直接将黑名单的IP访问拒绝。
2. 支持URL白名单，将不需要过滤的URL进行定义。
3. 支持User-Agent的过滤，匹配自定义规则中的条目，然后进行处理（返回403）。
4. 支持CC攻击防护，单个URL指定时间的访问次数，超过设定值，直接返回403。
5. 支持Cookie过滤，匹配自定义规则中的条目，然后进行处理（返回403）。
6. 支持URL过滤，匹配自定义规则中的条目，如果用户请求的URL包含这些，返回403。
7. 支持URL参数过滤，原理同上。
8. 支持日志记录，将所有拒绝的操作，记录到日志中去。
9. 日志记录为JSON格式，便于日志分析，例如使用ELKStack进行攻击日志收集、存储、搜索和展示。
10. 资源防盗链，支持添加资源正则规则与可访问Refer头中的域名，默认关闭。
11. 支持对上传的文件进行文件类型匹配与彻底的文件内容审查，发现非法语句记入日志并返回403。
12. 屏蔽除GET，OPTIONS之外的HTTP请求，可在设置中关闭此项功能。
13. 支持网段反向匹配，只允许在网段内的IP访问。

**过滤规则介绍：**过滤方式由可扩展的规则模块构成，具体路径在waf目录下的rule-config文件夹中。

Args.rule：对get参数进行过滤。

Blackip.rule：匹配黑名单规则。

Cookie.rule：过滤请求中cookie头中的数据。

url.rule：url黑名单规则。

Useragent.rule：过滤请求中cookie头中的数据。

Whitetip.rule：IP白名单。

Whiteurl.rule：url白名单。

Exp.rule：匹配最新漏洞的exp攻击语句。

**2.2漏洞扫描器**

漏洞扫描器分析来自nmap扫描器的扫描结果，并对其中扫描到的服务版本进行CVE版本号匹配，枚举出所有可能存在本机服务上的已公开的漏洞信息。

原理图如图2-3所示。

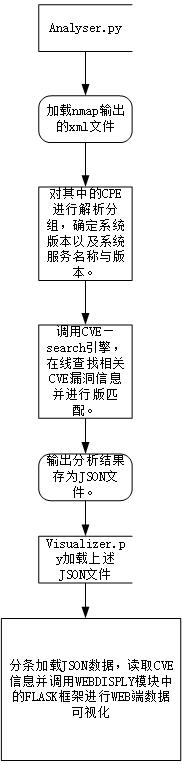


图2-3

注意：为了完全卫士的迅速与精悍，若将所有已公开的CVE信息连同数据安装到本地可能会造成运行速度缓慢，因此本模块并不提供单机安装，所有CVE信息均来自互联网，因此本模块需在有正常网络环境中使用，CVE搜索引擎地址在项目主目录etc文件下的配置文件中，可根据相应环境进行配置。

**2.3 后门扫描器**

网站后门查杀模块是基于静态扫描，在本地搭建了三个特征库，一个是可疑文件名的特征库，比如muma.php，houmen.php,一个是一句话木马的特征库，在扫描特定目录时，生成该目录下所有文件的绝对路径，之后判断每个文件的大小，根据目录下文件的具体大小选择是将文件内容全部读入内存或者是逐行读入内存，然后根据读入内存的内容，在特征库中逐行查找，如果存在匹配的webshell，则判定其为木马程序。至于这个一句话木马的特征库是我这么多年来上传shell总结的一个小型的个人库以及在github上搜索到的一些开源的木马库。另外一个哈希库是我搜集的所有大马生成的md5值的汇总，大约在上千个左右，这个匹配，是在拿到指定目录下所有文件绝对路径的基础上，调取系统命令生成哈希值，然后拿指定的哈希值匹配本地的特征库，适用于防备一些手里头有特定大马的攻击者。除此之外，还有一种匹配正则表达方式，使用python库中的正则表达式库，构建一些webshell的常用语法格式，进行读取文件之后，拿读入在内存中的数据匹配正则，这样能加大匹配的准确率，从而大大减小的编写程序的误报率。为了提高检测木马程序的速度，我引入了python的多线程编程，如果本地服务器为多核，则或许可提高检测后门的速度。但在匹配特征库的时候会存在竞争问题，所以对多线程进行加锁，以避免出现恶性竞争问题。在这个小模块中，还有一些获取查找文件最后修改时间，获取本机ip地址，以及生成特定格式的一些小函数，这些函数帮助了本模块更好的融入到了整体waf的环境中去，为用户的web服务器提供了更多的安全保障同时提升了用户的体验效果。模块结构如图2-3所示。

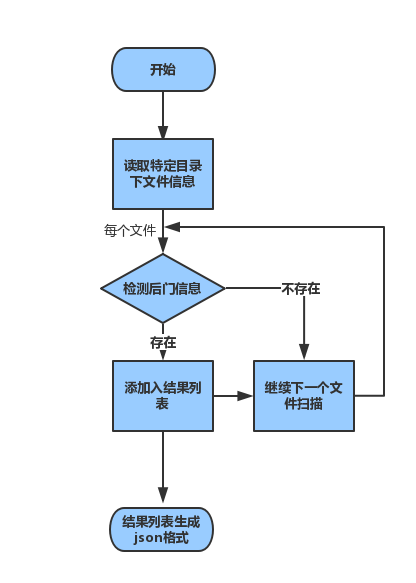


图2-4

模块原理图如图2-4所示。

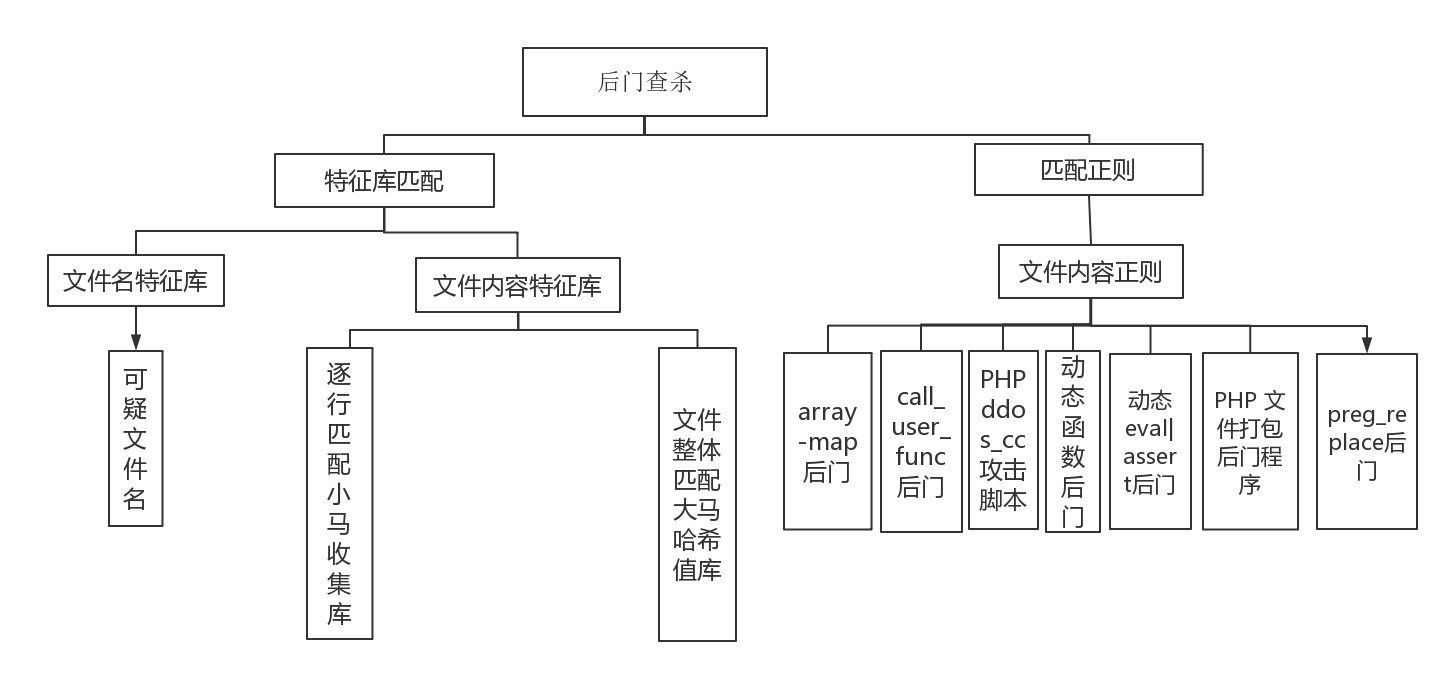


图2-5

**2.4 网站前端设计**

为了方便用户使用，我们直接将功能展示于web网页上。前端使用bootstrap、semantic、jquery、datatables和echarts等框架，后端主要是使用python2.7开发，使用flask框架，对json文件进行处理。

Flask是一个使用pyhton编写的轻量级 Web 应用框架。其 WSGI工具箱采用 Werkzeug ，模板引擎则使用 Jinja2 。相对于django框架，falsk更加灵活。本项目中主要使用了flask的@app.route()利用装饰器的方式实现了路由函数，方便读取json数据和传输json串给前端的ajax。

Datatables是jquery的表格插件，是一个高度灵活的工具，可以将任何HTML表格添加高级的交互功能。本项目中，利用Datatables实现威胁分析中的网站应用攻击（web容器、目录遍历、文件包含、文件上传、非法参数、黑客工具利用攻击、非正常HTTP请求、命令注入、SQL注入攻击、跨站脚本攻击（XSS））、系统层攻击（页面后门、系统漏洞、恶意病毒）、网络层攻击（暴力破解、CC攻击、防盗链）的数据展示，而且提供了按时间、类型等排序方式，还有搜索功能，可以轻松找到需要查看的数据，让用户一目了然。

Echarts是一个纯Javascript的图表库，可以流畅的运行在PC和移动设备上，底层依赖轻量级的Canvas类库ZRender，提供直观，生动，可交互，可高度个性化的数据可视化图表。本项目中的攻击总量、服务器攻击次数和攻击类型的图表，都是在echarts的基础上个性化出来的。将鼠标移动至每一个坐标点，将会显示详细信息，让用户能进一步了解到攻击状况。

Bootstrap和Semantic是HTML、CSS和JS框架，本项目中，Bootstrap提供页面的整体布局（如图），使页面简约而且兼容各种平台。使用Semantic实现页面一些模块的凸显风格，使页面看上去浑然一体。



在攻击事件详情菜单栏，使用jquery生成堆叠式菜单栏，不占空间，方便操作，同时，我们引入了iframe内联框架，嵌入了另一个页面，以弹窗的形式显示出来，附带动画过渡效果，给用户更好的体验。

对于数据处理，本项目中并没有使用数据库，因为我们在衡量数据库和json文件的性能时，经过测试结果显示，直接读取json文件的性能会更好，所以我们采用直接读取json文件的方式。将所有的json文件放在static/temp下，然后使用flask和jquery调用读取json文件显示在指定的标签中。Flask中先使用os.path.realpath读取json文件的真实路径，os.path.join对路径进行拼接，json.load将json文件加载到变量中，然后对变量值进行操作，最后通过jsonify方法将文件转换为json串的形式传递到ajax和前端标签中。

**网站监听端口可在etc目录下的configuration.ini进行配置。**

1. **安装步骤**

**3.1 网站防火墙安装步骤**

**3.1.1 通过install.sh进行安装：**

本安装脚本只适用于使用APT包管理器的Linux发行版本，例如Ubuntu，其他版本请参照其他安装方法。

# 给予instal.sh运行权限:

*Chmod + x install.sh*

# 运行安装脚本

./install.sh

等待安装完成后会在*/usr/local/*中发现openresty文件夹，此为带有nginx服务器与LUA脚本解释器的服务端，用户可自行选择是否使用此服务作为主要服务器或者使用此应用程序的反向代理保护其他服务器。

# 编辑*/usr/local/openresty/nginx/conf/nginx.conf*文件，在http段添加以下内容:

*lua\_shared\_dict limit 50m;*

*lua\_package\_path "/usr/local/openresty/nginx/waf/?.lua";*

*init\_by\_lua\_file "/usr/local/openresty/nginx/waf/init.lua";*

*access\_by\_lua\_file "/usr/local/openresty/nginx/waf/access.lua";*

#进入*/usr/local/openresty/nginx/sbin*目录，启动nginx:

*./nginx*

每次进行匹配规则修改或者设置更改都需要重启nginx服务器：

.*/nginx -s reload*

安装结束后访问本机80端口，若出现Openresty欢迎界面则说明安装成功。

**查看日志分析报表：**

#进入项目主目录，在bin文夹下执行启动脚本

*Python3 run.py*

由于漏洞扫描脚本也被集成于此，所以在每次启动时都会执行漏洞扫描，因此启动时间可能会比较长。

启动完毕后，请访问本机81端口，可看到日志分析报告。

我们强烈建议在完成日志分析之后关闭81端口，因为此网页依载与Flask的简单WEB服务，若在实际生产环境中可能存在重大安全隐患。

用户也可选择分步进行，bin目录下的脚本都可单独执行，也可选择利用Linux定时任务每天执行，考虑到项目的简易性不再提供相关工具。

**Bin目录介绍如下：**

Analyzer.py 将nmap输出结果json化并在CVE库中寻找相匹配的系统漏洞。

Visualizer.py 将analyzer输出的结果加载到网页端进行数据可视化。

Log\_analyser.py 分析来自网站防火墙的日志，并生成对应统计文件供WEB端调用。

Ssh\_analyser.py 分析来自SSH的日志，归纳分析是否存在暴力破解等攻击。

run.py 一键加载执行上述脚本。

**现介绍编译安装：**

环境准备

首先，现在Nginx安装必备的Nginx和PCRE软件包。

*wget http://nginx.org/download/nginx-1.9.4.tar.gz*

*wget* ftp://ftp.csx.cam.ac.uk/pub/software/programming/pcre/pcre-8.37.tar.gz

其次，下载当前最新的luajit和ngx\_devel\_kit (NDK)，以及春哥（章）编写的lua-nginx-module。

*wget http://luajit.org/download/LuaJIT-2.0.4.tar.gz*

*wget https://github.com/simpl/ngx\_devel\_kit/archive/v0.2.19.tar.gz*

*wget https://github.com/openresty/lua-nginx-module/archive/v0.9.16.tar.gz*

最后，创建Nginx运行的普通用户。

*useradd -s /sbin/nologin -M www*

解压NDK和lua-nginx-module。

*tar zxvf v0.2.19 #解压后为ngx\_devel\_kit-0.2.19*

*tar zxvf v0.9.10 #解压后为lua-nginx-module-0.9.16*

安装LuaJIT Luajit是Lua即时编译器。

*tar zxvf LuaJIT-2.0.3.tar.gz*

*cd LuaJIT-2.0.3*

*make && make install*

安装Nginx并加载模块。

*tar zxvf nginx-1.9.4.tar.gz*

*cd nginx-1.9.4*

*export LUAJIT\_LIB=/usr/local/lib*

*export LUAJIT\_INC=/usr/local/include/luajit-2.0*

*./configure --prefix=/usr/local/nginx --user=www --group=www --with-http\_ssl\_module --with-http\_stub\_status\_module --with-file-aio --with-http\_dav\_module --add-module=../ngx\_devel\_kit-0.2.19/ --add-module=../lua-nginx-module-0.9.16/ --with-pcre=/usr/local/src/pcre-8.37*

*make -j2 && make install*

*ln -s /usr/local/lib/libluajit-5.1.so.2 /lib64/libluajit-5.1.so.2*

**如果不创建符号链接，可能出现以下异常： error while loading shared libraries: libluajit-5.1.so.2: cannot open shared object file: No such file or directory**

测试安装 安装完毕后，下面可以测试安装了，修改nginx.conf 增加第一个配置。

*location /hello {*

*default\_type 'text/plain';*

*content\_by\_lua 'ngx.say("hello,lua")';*

*}*

/usr/local/nginx-1.9.4/sbin/nginx –t

/usr/local/nginx-1.9.4/sbin/nginx

然后访问http://xxx.xxx.xxx.xxx/hello 如果出现hello,lua。表示安装完成,然后就可以。

注意：也可以直接部署春哥的开源项目：https://github.com/openresty。

**网站防火墙配置请参考安装脚本，安装时切记目录与配置文件目录需一致，无法创建日志时请检查当前NGINX服务器运行用户对配置文件中的日志目录是否具有可写权限。**

参考文献：

[1] Miguel Grinberg,Flask Web开发：基于Python的Web应用开发实战, 人民邮电出版社,2015:8-45

[2] Lua设计与实现， codedump， 人民邮电出版社

[3] Lua程序设计（第2版）, 莱鲁萨利姆斯奇（Ierusalimschy，R.）, 电子工业出版社

[4] Python 黑帽子：黑客与渗透测试编程之道, Justin Seitz（贾斯汀·塞茨） 著，孙松柏 李聪 润秋 译, 电子工业出版社

[5] 恶意代码分析实战, Michael Sikorski（迈克尔.斯科尔斯基）, Andrew Honig（安德鲁.哈尼克）, 电子工业出版社

[6] 反入侵的艺术——黑客入侵背后的真实故事, （美）米特尼克，（美）西蒙　著，袁月杨，谢衡　译, 清华大学出版社

[7] linux服务器安全攻防, [美]Chris Binnie著 田洪 译, 清华大学出版社

作者联络方式：

（1）通讯地址：山东省淄博市张店区山东理工大学

（2）电话：18369975220

（3）Ｅ－ｍａｉｌ：738686337@qq.com

（4）QQ号：738686337

命题老师评语：