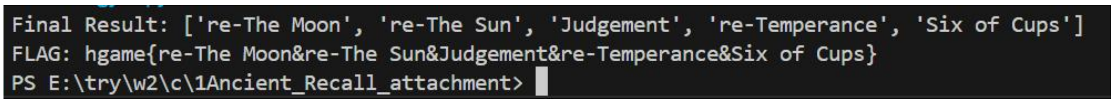
队伍名称：**siyah**

队伍 **ID：00007f**

Ancient Recall

```python
import random

Major_Arcana = ["The Fool", "The Magician", "The High Priestess","The Empress", "The Emperor", "The
Hierophant","The Lovers", "The Chariot", "Strength","The Hermit", "Wheel of Fortune", "Justice","The Hanged Man",
"Death", "Temperance","The Devil", "The Tower", "The Star","The Moon", "The Sun", "Judgement","The World"]
wands = ["Ace of Wands", "Two of Wands", "Three of Wands", "Four of Wands", "Five of Wands", "Six of Wands",
"Seven of Wands", "Eight of Wands", "Nine of Wands", "Ten of Wands", "Page of Wands", "Knight of Wands", "Queen
of Wands", "King of Wands"]
cups = ["Ace of Cups", "Two of Cups", "Three of Cups", "Four of Cups", "Five of Cups", "Six of Cups", "Seven
of Cups", "Eight of Cups", "Nine of Cups", "Ten of Cups", "Page of Cups", "Knight of Cups", "Queen of Cups",
"King of Cups"]
swords = ["Ace of Swords", "Two of Swords", "Three of Swords", "Four of Swords", "Five of Swords", "Six of Swords",
"Seven of Swords", "Eight of Swords", "Nine of Swords", "Ten of Swords", "Page of Swords", "Knight of Swords",
"Queen of Swords", "King of Swords"]
pentacles = ["Ace of Pentacles", "Two of Pentacles", "Three of Pentacles", "Four of Pentacles", "Five of Pentacles",
"Six of Pentacles", "Seven of Pentacles", "Eight of Pentacles", "Nine of Pentacles", "Ten of Pentacles", "Page
of Pentacles", "Knight of Pentacles", "Queen of Pentacles", "King of Pentacles"]
Minor_Arcana = wands + cups + swords + pentacles
tarot = Major_Arcana + Minor_Arcana
reversals = [0,-1]
YOUR_final_Value = [
    2532951952066291774890498369114195917240794704918210520571067085311474675019,
    2532951952066291774890327666074100357898023013105443178881294700381509795270,
    2532951952066291774890554459287276604903130315859258544173068376967072335730,
    2532951952066291774890865328241532885391510162611534514014409174284299139015,
    2532951952066291774890830662608134156017946376309989934175833913921142609334
]

def restore_FATE(FATEd):
    S = sum(FATEd) // 2  # 计算 S = (FATE0 + FATE1 + ... + FATE4)
    FATE = [0] * 5
    FATE[0] = S - FATEd[1] - FATEd[3]
    FATE[1] = FATEd[0] - FATE[0]
    FATE[2] = FATEd[1] - FATE[1]
    FATE[3] = FATEd[2] - FATE[2]
    FATE[4] = FATEd[3] - FATE[3]
    return FATE
for i in range(250):
    YOUR_final_Value = restore_FATE(YOUR_final_Value)
result_cards = []
for value in YOUR_final_Value:
    if value < 0:
        card = tarot[value ^ -1]
        result_cards.append("re-" + card)
    else:
        card = tarot[value]
        result_cards.append(card)
print("Final Result:", result_cards)
# 生成 FLAG
FLAG = "hgame{" + "&".join(result_cards) + "}".replace(" ", "_")
print("FLAG:", FLAG)
```

运行结果：



```
Final Result: ['re-The Moon', 're-The Sun', 'Judgement', 're-Temperance', 'Six of Cups']
FLAG: hgame{re-The Moon&re-The Sun&Judgement&re-Temperance&Six of Cups}
PS E:\try\w2\c\1Ancient_Recall_attachment>
```

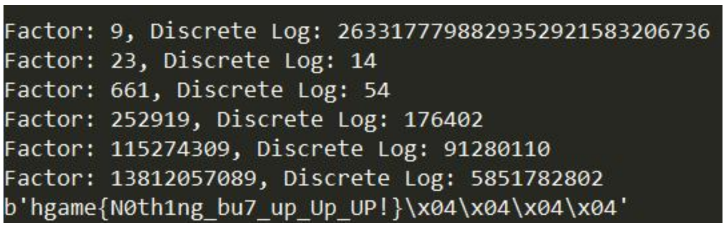
flag:hgame{re-The_Moon&re-The_Sun&Judgement&re-Temperance&Six_of_Cups}

Intergalactic Bound

```python
from Crypto.Util.number import *
from Crypto.Cipher import AES
import hashlib

p = 55099055368053948610276786301
G = (19663446762962927633037926740, 35074412430915656071777015320)
Q = (26805137673536635825884330180, 26376833112609309475951186883)
R.<a> = Zmod(p)[]
# 构造多项式 f 并求解根
f = a * Q[0] * Q[1] * G[0]**3 + G[1]**3 * Q[0] * Q[1] + Q[0] * Q[1] - a * G[0] * G[1] * Q[0]**3 - Q[1]**3 * G[0]
* G[1] - G[0] * G[1]
roots = f.roots()
if roots:
    a_value = int(roots[0][0])
else:
    raise ValueError("No roots found for the polynomial f")
# 私钥 d
d = (a_value * G[0]**3 + G[1]**3 + 1) * inverse(G[0] * G[1], p) % p
# 验证
```

```python
assert (a_value * G[0]**3 + G[1]**3 + 1) % p == (d * G[0] * G[1]) % p

assert (a_value * Q[0]**3 + Q[1]**3 + 1) % p == (d * Q[0] * Q[1]) % p

# 椭圆曲线的系数

a0 = 1

a1 = -3 * (d / 3) / (a_value - (d / 3)**3)

a3 = -9 / ((a_value - (d / 3)**3)**2)

a2 = -9 * (d / 3)**2 / ((a_value - (d / 3)**3)**2)

a4 = -27 * (d / 3) / ((a_value - (d / 3)**3)**3)

a6 = -27 / ((a_value - (d / 3)**3)**4)

# 定义椭圆曲线

E = EllipticCurve(GF(p), [a1, a2, a3, a4, a6])

# 计算 G 和 Q 的坐标

gx = (-3 / (a_value - d**3 / 27)) * G[0] / (d * G[0] / 3 - (-G[1]) + 1)

gy = (-9 / ((a_value - d**3 / 27)**2)) * (-G[1]) / (d * G[0] / 3 - (-G[1]) + 1)

px = (-3 / (a_value - d**3 / 27)) * Q[0] / (d * Q[0] / 3 - (-Q[1]) + 1)

py = (-9 / ((a_value - d**3 / 27)**2)) * (-Q[1]) / (d * Q[0] / 3 - (-Q[1]) + 1)

G_point = E(gx, gy)

Q_point = E(px, py)

factors, exponents = zip(*factor(E.order()))

primes = [factor**exponent for factor, exponent in zip(factors, exponents)]

# 计算离散对数

dlogs = []

for prime in primes:

    t = int(G_point.order()) // int(prime)

    try:

        dlog = discrete_log(t * Q_point, t * G_point, operation="+")

        dlogs.append(dlog)

        print(f"Factor: {prime}, Discrete Log: {dlog}")

    except Exception as e:

        print(f"Error computing discrete log for factor {prime}: {e}")

# 中国剩余定理

try:

    l = crt(dlogs, primes)

except Exception as e:

    raise ValueError("Failed to compute CRT") from e

# 解密

ciphertext = b"k\xe8\xbe\x94\x9e\xfc\xe2\x9e\x97\xe5\xf3\x04'\x8f\xb2\x01T\x06\x88\x04\xeb3Jl\xdd
Pk$\x00:\xf5"

key = hashlib.sha256(str(l).encode()).digest()

cipher = AES.new(key, AES.MODE_ECB)

decrypted = cipher.decrypt(ciphertext)

print(decrypted)
```

运行结果：

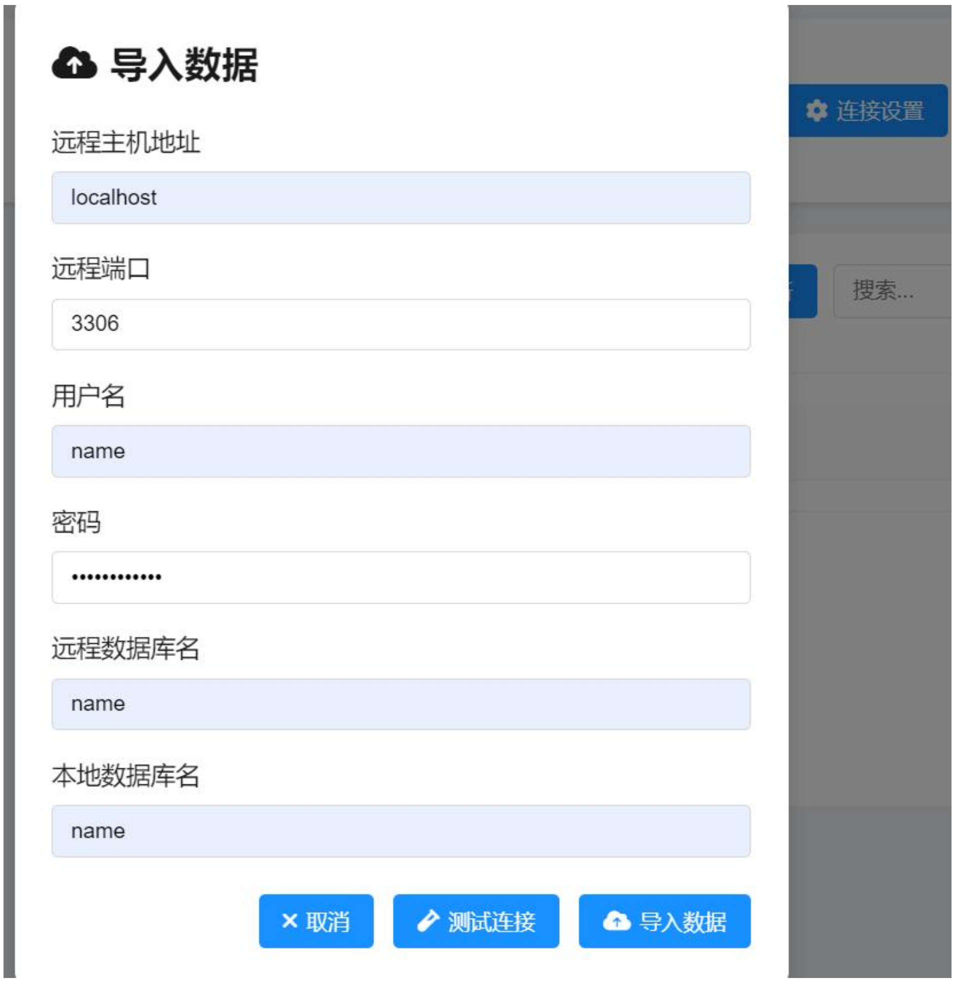


```
Factor: 9, Discrete Log: 26331777988293529215832067З6
Factor: 23, Discrete Log: 14
Factor: 661, Discrete Log: 54
Factor: 252919, Discrete Log: 176402
Factor: 115274309, Discrete Log: 91280110
Factor: 13812057089, Discrete Log: 5851782802
b'hgame{N0th1ng_bu7_up_Up_UP!}\x04\x04\x04\x04'
```

flag:hgame{N0th1ng_bu7_up_Up_UP!}

Level 21096 HoneyPot

密码输入：;/writeflag;



导入数据

远程主机地址: localhost

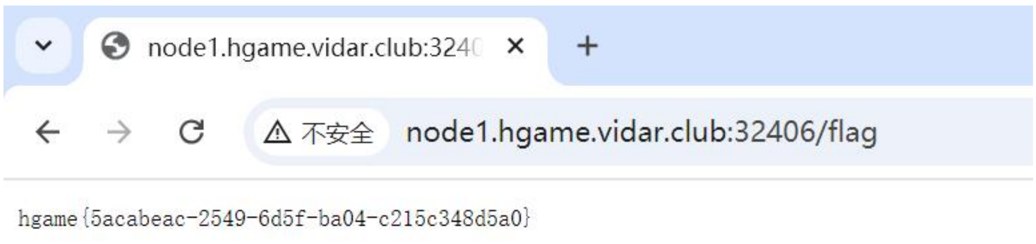
远程端口: 3306

用户名: name

密码: ••••••••••••

远程数据库名: name

本地数据库名: name



node1.hgame.vidar.club:32406/flag

hgame{5acabeac-2549-6d5f-ba04-c215c348d5a0} flag:hgame{5acabeac-2549-6d5f-ba04-c215c348d5a0}