HGAME 2022 Week3writeup by 给爷点一杯奶茶

笔记本:	My Notebook		
创建时间:	2022/2/11 0:14	**更新时间:**	2022/2/11 0:29
作者:	cjx		
URL:	https://hgame.vidar.club/#/challenge/list		

HGAME 2022 Week3writeup by 给爷点一杯奶茶

CRYPTO

Multi Prime RSA

位现象，建议100%大小显示。

定理2：取值为素数方幂的欧拉函数

设p是素数，则对于任一正整数r，有

$$\varphi(p^r) = p^{r-1}(p-1).$$

证 明

由于互素的整数个数不易计算，下面从不互素的整数出发进行证明。

设集合$\Omega_{p^r} = \{1, 2, \cdots, p^r\}$，其中与$p^r$不互素的整数的个数即所求。对$\forall a \in \Omega_{p^r}$，利用互素整数的性质3推广，有$(a, p) = 1 = (a, p^r) = 1$。若$(a, p) \neq 1$，则$p \mid a$，从而$(a, p^r) \neq 1$，因此

$$\begin{aligned}(a, p^r) \neq 1 &\iff (a, p) \neq 1 \iff p \mid a \\ &\iff a = p, 2p, \cdots, p^{r-1}p,\end{aligned}$$

从而Ω_{p^r}中与p^r中不互素的整数的个数为p^{r-1}，于是得到

$$\varphi(p^r) = p^r - p^{r-1} = p^{r-1}(p-1). \qquad \square$$

由图中定理可求出phi（n）

知道phi（n）d便可轻易求出

```
from libnum import n2s,s2n
def egcd(a,b):          #扩展欧几里得算法
    if a==0:
        return  (b,0,1)
    else:
        g,y,x=egcd(b%a,a)
        return (g,x-(b//a)*y,y)

def modinv(a,m):
    g,x,y=egcd(a,m)
    if g!=1:
        raise Exception('modular inverse does not
    else:
        return x%m

p = 617899321487194773840274583333805689780562861
q = 912079693533557636856332843788335063197947145
r = 105471299607375388622347272479207944509670502
s = 831532387489037724481383075055797992771626521
n = 103934437216508710000106392059815181232415106
e = 65537
c = 844677395496466411520394190869787261209960246
phi = p*(p-1)*q**2*(q-1)*r**4*(r-1)*s**6*(s-1)
d = modinv(e,phi)
m = pow(c,d,n)
print(n2s(m))
```

```
b'hgame{EulEr:fUNcTIon;iS.So*IMpORTaNt*In&RsA}'
```

RSA Attack 2

考察低解密指数攻击
在网上下载脚本
git clone https://github.com/pablocelayes/rsa-wiener-attack
改一下关键脚本放入数据即可求出d

```
import ContinuedFractions, Arithmetic, RSAvulnerableKeyGenerator
import sys
sys.setrecursionlimit(10000000)
def hack_RSA(e, n):
    '''
    Finds d knowing (e,n)
    applying the Wiener continued fraction attack
    '''
    frac = ContinuedFractions.rational_to_contfrac(e, n)
    convergents = ContinuedFractions.convergents_from_contfrac(frac)
    for (k, d) in convergents:

                        # check if d is actually the key
        if k != 0 and (e * d - 1) % k == 0:
            phi = (e * d - 1) // k
            s = n - phi + 1
                                    # check if the equation x^2 - s*x + n = 0
                                    # has integer roots
            discr = s * s - 4 * n
            if (discr >= 0):
                t = Arithmetic.is_perfect_square(discr)
                if t != -1 and (s + t) % 2 == 0:
                    return d

n = 507419170088344932990702256911694788408493968749527614421614
26116727701474818301284444060338498964766419007485308669340852
e = 773101998674486777820815721093434727837811356417125976435971
34447212248648869514585047871442060412622164276894766238383894
print (hack_RSA(e, n))
```

后即可求出flag

```
from libnum import n2s,s2n

n = 507419170088344932990702256911694788408493968749527614421614568
c = 165251729917394529793163344300848992394021337429474789711805041
d = 13094612077654083919
m=pow(c,d,n)
print(n2s(m))
```

```
b'hgame{dO|YOU:kNOw!tHE*PRINcIplE*bEhInd%WInNEr#aTTacK}'
```