babyrsa

November 2, 2021

```
In [45]: pk = (52,
         47782489586021221729935382562238217213800826152327617948974334325027793433971508995399
         4332296195603747350220606126604871393115472775644073162694668557793309308041609733257
         594494499417894774789048451572687632660208181953899084791149805130704950465573064455
         261181509476485410165375351178418657075805043256124045823778754055035546150122155335
         78198056729486649750112380592340915292157198579677858031071483250010642559916812974570
         c0,c1 = (9443988406787586692651048082648080521808695684874354749919800682045855739679
         g, h, A, B, p, q = pk
         R = IntegerModRing (p)
         Rn = IntegerModRing (p-1)
         k0 = c1 * R(c0)^-1 * (R(h)^Rn(A))^-1
         k1 = Rn(B-1)^-1
         y = R(c0)^Rn(B)*R(h)^Rn(A)*R(c1)^-1
         ee = Rn(0xfaab*B-1)
         # ee^-1 is not exist
         littlee = gcd(Integer(ee),p-1)
         #factor ee in factordb.com
         # ee = 167*3*335451867841431314236066144147986873666325923543861873476567924296958850
         d0=Rn(3)^-1
         d1=Rn(3354518678414313142360661441479868736663259235438618734765679242969588504681524
         yy = (y^d0)^d1

In [46]: def AMM(o, r, q):
             import time
             import random
             start = time.time()
             print('\n--------------------------------------------------------------------------------
             print('Start to run Adleman-Manders-Miller Root Extraction Method')
             print('Try to find one {:#x}th root of {} modulo {}'.format(r, o, q))
             g = GF(q)
             o = g(o)
             p = g(random.randint(1, q))
             while p ^ ((q-1) // r) == 1:
                 p = g(random.randint(1, q))
             print('[+] Find p:{}'.format(p))
             t = 0
             s = q - 1
```

```
            while s % r == 0:
                t += 1
                s = s // r
            print('[+] Find s:{}, t:{}'.format(s, t))
            k = 1
            while (k * s + 1) % r != 0:
                k += 1
            alp = (k * s + 1) // r
            print('[+] Find alp:{}'.format(alp))
            a = p ^ (r**(t-1) * s)
            b = o ^ (r*alp - 1)
            c = p ^ s
            h = 1
            for i in range(1, t):
                d = b ^ (r^(t-1-i))
                if d == 1:
                    j = 0
                else:
                    print('[+] Calculating DLP...')
                    j = - discrete_log(d, a)
                    print('[+] Finish DLP...')
                b = b * (c^r)^j
                h = h * c^j
                c = c^r
            result = o^alp * h
            end = time.time()
            print("Finished in {} seconds.".format(end - start))
            print('Find one solution: {}'.format(result))
            return result
```

```
In [47]: def findAllPRoot(p, e):
             import time
             import random
             print("Start to find all the Primitive {:#x}th root of 1 modulo {}.".format(e, p)
             start = time.time()
             proot = set()
             while len(proot) < e:
                 proot.add(pow(random.randint(2, p-1), (p-1)//e, p))
             end = time.time()
             print("Finished in {} seconds.".format(end - start))
             return proot
```

```
In [48]: # yy=1797107407905827711236235645086245621631815671133817819503312723791932387457787
         # ee=167
         # p=26118150947648541016537535117841865707580504325612404582377875405503554615012215
```

```
In [49]: mp = AMM(yy, littlee, p)
```

```
Start to run Adleman-Manders-Miller Root Extraction Method
Try to find one 0xa7th root of 179710740790582771123623564508624562163181567113381781950331272
[+] Find p:978958012857851394872761445574993325459688521675044629113315409529739250678032339045
[+] Find s:15639611345897329950022476118468183058431439715935571606214296650002128511983362594
[+] Find alp:131110514276983604371445907579972792106610871870118564363473145568880119262135973
Finished in 680.180348873 seconds.
Find one solution: 19751913164810400425450486484783784205227141879021217573170053241792457020
```

In [50]:
```python
p_proot = findAllPRoot(p, littlee)
```

```
Start to find all the Primitive 0xa7th root of 1 modulo 26118150947648541016537535117841865707
Finished in 26.5517168045 seconds.
```

In [51]:
```python
def findAllSolutions(mp, proot, cp, p,e):
    import time
    print("Start to find all the {:#x}th root of {} modulo {}.".format(e, cp, p))
    start = time.time()
    all_mp = set()
    for root in proot:
        mp2 = (mp * root) % p
        if (pow(mp2, e, p) == cp):
            all_mp.add(mp2)
    end = time.time()
    print("Finished in {} seconds.".format(end - start))
    return all_mp

mps = findAllSolutions(mp, p_proot, yy, p,littlee)
```

```
Start to find all the 0xa7th root of 17971074079058277112362356450862456216318156711338178195
Finished in 0.0204038619995 seconds.
```

In [52]:
```python
for i in mps:
    if('HIT' in hex(Integer(i)).decode('hex;')):
        print(hex(Integer(i)).decode('hex;'))
        break
```

```
HITCTF2021{Numb3r_Th30ry_1s_Funny!}
```