# fast\_attack

题目是求解椭圆曲线上离散对数的问题，发现曲线的阶是和模数相同的，因此使用SmartAttcak方法

EXP：

from pwn import \*

def SmartAttack(P,Q,p):

    E = P.curve()

    Eqp = EllipticCurve(Qp(p, 2), [ ZZ(t) + randint(0,p)\*p for t in E.a\_invariants() ])

    P\_Qps = Eqp.lift\_x(ZZ(P.xy()[0]), all=True)

    for P\_Qp in P\_Qps:

        if GF(p)(P\_Qp.xy()[1]) == P.xy()[1]:

            break

    Q\_Qps = Eqp.lift\_x(ZZ(Q.xy()[0]), all=True)

    for Q\_Qp in Q\_Qps:

        if GF(p)(Q\_Qp.xy()[1]) == Q.xy()[1]:

            break

    p\_times\_P = p\*P\_Qp

    p\_times\_Q = p\*Q\_Qp

    x\_P,y\_P = p\_times\_P.xy()

    x\_Q,y\_Q = p\_times\_Q.xy()

    phi\_P = -(x\_P/y\_P)

    phi\_Q = -(x\_Q/y\_Q)

    k = phi\_Q/phi\_P

    return ZZ(k)

pp=remote("139.155.139.109",10000)

for i in range(10):

    x=pp.recvline()

    y=pp.recvline()

    x\_str = x.decode('utf-8').strip(">")

    y\_str = y.decode('utf-8').strip()

# strip() 去除两端的空白字符，包括换行符

# 使用冒号分割字符串，并转换为列表

    parts = x\_str[1:-1].split(':')  # [1:-1] 去除字符串两端的括号

    parts\_y = y\_str[1:-1].split(':')

    print(x\_str)

    print(parts)

    print(y\_str)

    print(parts\_y)

# 提取前两个元素，并转换为整数

    first\_number\_x = int(parts[0].strip())

    second\_number\_x = int(parts[1].strip())

    first\_number\_y = int(parts\_y[0].strip())

    second\_number\_y = int(parts\_y[1].strip())

    print(first\_number\_x)

    P=(int(first\_number\_x),int(second\_number\_x))

    Q=(int(first\_number\_y),int(second\_number\_y))

    a = 0

    b = -3045286915816228928193649683228046896481316

    p = 235322474717419

    P1 = GF(p)

    E = EllipticCurve(P1, [a, b])

    P=E(P)

    Q=E(Q)

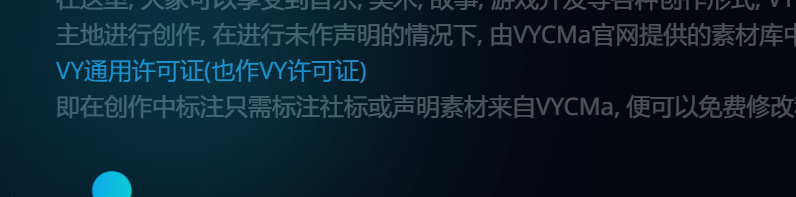
    result=SmartAttack(P,Q,p)

    pp.sendline(str(result).encode())

flag=pp.recvline()

print(flag)

# 签到



进入查看源代码

