Writeup

1. 拿到题目附件，分析源码

```
# These three are constants
p =
12039102490128509125925019010000012423515617235219127649182470182570195018265927223
g =
10729072579307052184848302322451332192456229619044181105063011741516558110216720725

# random generation
m1 = "test1"
m2 = "test2"

# Initialization
r1, s1 = sign(m1)
# r1 R1 will be provided to player

def int2str(data, mode="big"):
    if mode == "little":
        return sum([ord(data[_]) * 2 ** (8 * _) for _ in range(len(data))])
    elif mode == "big":
        return sum([ord(data[::-1][_]) * 2 ** (8 * _) for _ in range(len(data))])

def get_parameter(m):
    x = int2str(m, 'little')
    y = powmod(g, x, p)
    a = bytes_to_long(hashlib.sha256(long_to_bytes(y).rjust(128, "\0")).digest())
    b = powmod(a, a, p - 1)
    h = powmod(g, b, p)

    return y, h, b

def sign(m):
    y, h, b = get_parameter(m)
    r = getStrongPrime(512)
    s = (y * powmod(h, r, p)) % p

    return str(r),str(s)

def verify(m, r, s):
    y, h, b = get_parameter(m)
    if s == ((y * powmod(h, r, p)) % p):
        return True
    else:
```

```
        return False

# Give me the (r2,s2)
if r2 != r1 and s2 == s1 and verify(m2, r2, s2):
    print("Congratulation!Here is your flag: %s" % flag)
```

题目实现了一个类似于DSA签名的算法，其中p、g是固定不变的（**这里设置为常量值，是因为只有满足特定关系的p、g才能够实现题目的攻击效果，具有局限性；本题制作的大部分时间，都消耗在生成p、g上**）。我们输入消息m，程序会计算出该消息的签名`(r,s)`，其中`r`是随机生成的，因此`s`除非拥有`r`，否则无法进行验证。

题目要求很简单：随机生成两个消息`m1`和`m2`，并提供`m1`的`r1`值，需要我们计算`m2`的`r2`值，满足`r1!=r2`且`s1==s2`，而且该签名必须有效的。将`(r2,s2)`提交给服务端进行验证，验证成功即可得到flag。

2. 我们简单分析一下：题目要求 `s1 == s2`，也就有

```
y1*(h1^r1 mod p) == y2^(h2^r2 mod p)
↓↓↓
(g^x1 mod p)*(g^b1*r1 mod p) == (g^x2 mod p)*(g^b2*r2 mod p)
↓↓↓
(g^(x1+b1*r1) mod p) == (g^(x2+b2*r2) mod p)
```

可以看到，该算法的本质是`g`的莫幂运算，则有

```
x1+b1*r1 = x2+b2*r2 mod φ(p)
↓↓↓
r2 = (x1 - x2 + b1*r1)*b2^(-1) mod φ(p)
```

但是，问题在于`gcd(b2,φ(p))!=1`。那么，我们可以转换思路，去寻找`p`的大质数因子，假设这个因子为`q`，利用这个质数大因子进行求解，那么我们的求解就变成

```
r2 = (x1 - x2 + b1*r1)*b2^(-1) mod q
```

利用这个质数大因子`q`，即可求解`r2`。

当然，由于`p`值本身并不大，我们也可以对`p`进行分解，然后再求欧拉函数值。

3. 依据上述的思路，编写`exp.py`即可。