Cado-nfs 计算 DLP 举例

p=2234567890123456783012345678901234567890123456789012345678071

g=17311125480404630 1125

p-1=2 * 5 * 223456789012345678301234567890123456789012345678901234567807

h=g^x(modp)=4934187330375128509560317493098121016496489415597804987492 0

k=g^y(modp)=1147010785503565676377667024223788608331996333817020535033 9

(1) 计算 logh

注意这个时候的基不是我们所选的 g,是算法运算过程中的另外一个基。

./cado-nfs.py -dlp -ell 223456789012345678301234567890123456789012345678901234567807 target=4934187330375128509560317493098121016496489415597804987492 0 2234567890123456783012345678901234567890123456789012345678071

其中 ell 选择的是 p-1 的最大的素因子

得到结果：

```
Info:root: If you want to compute a new target, run ./cado-nfs.py /tmp/cado.503gn9v7/p60.parameters_snapshot.0 target=
arget>
p = 2234567890123456783012345678901234567890123456789012345678071
ell = 223456789012345678301234567890123456789012345678901234567807
log2 = 203023307460328734249664525420948469291051159158258865182 86
log3 = 64297174295078900949049315077285378190087514398659510643 14
The other logarithms of the factor base elements are in /tmp/cado.503gn9v7/p60.dlog
target = 4934187330375128509560317493098121016496489415597804987492 0
log(target) = 110684396376717129430541782167564603955980126575326270520 40
```

也就是：

logh=110684396376717129430541782167564603955980126575326270520 40(mod ell)

(2) 计算 logg

根据提示，只需要执行:

./cado-nfs.py /tmp/cado.503gn9v7/p60.parameters_snapshot.0

target=17311125480404630 1125

得到结果:

```
Info:root: If you want to compute a new target, run ./cado-nfs.py /tmp/cado.503gn9v7/p60.parameters_snapshot.1 target=<t
arget>
p = 2234567890123456783012345678901234567890123456789012345678071
ell = 223456789012345678301234567890123456789012345678901234567807
log2 = 203023307460328734249664525420948469291051159158258865182 86
log3 = 64297174295078900949049315077285378190087514398659510643 14
The other logarithms of the factor base elements are in /tmp/cado.503gn9v7/p60.dlog
target = 17311125480404630 1125
log(target) = 353051940241047920010586424126888471542192079897415989093 4
```

也就是:

logg=353051940241047920010586424126888471542192079897415989093 4(mod ell)

(3) 计算$x = log_g h$

换底公示计算：logh*logg^{-1}(mod ell)并验算

```
p=2234567890123456783012345678901234567890123456789012345678071
R=GF(p)
g=R(17311125480404630 1125)
gx=R(4934187330375128509560317493098121016496489415597804987492 0)
gy=R(1147010785503565676377667024223788608331996333817020535033 9)
ell=223456789012345678301234567890123456789012345678901234567807
log_h = 110684396376717129430541782167564603955980126575326270520 40
log_g = 353051940241047920010586424126888471542192079897415989093 4
temp=log_h * inverse_mod(log_g, ell) % ell
temp;g^temp;gx
```

```
8480023
4934187330375128509560317493098121016496489415597804987492 0
4934187330375128509560317493098121016496489415597804987492 0
```

已经得到 x=8480023

（4）计算$g^{xy}(mod p)$

得到 3333333333333333333333333333333333333333

即为口令。

以下作进一步讨论：

（5）计算 logk

运行：./cado-nfs.py /tmp/cado.503gn9v7/p60.parameters_snapshot.1

target=11470107855035656763776670242237886083319963338170205350339

得到 logk=2104706469553386779074488314562927800929700338655854189195 1（mod ell）

(6) 计算$log_g k$

换底公式计算$\text{logk}*\text{logg}^{-1}$(mod ell)

```
p=223456789012345678301234567890123456789012345678901234568071
R=GF(p)
g=R(173111254804046301125)
gx=R(49341873303751285095603174930981210164964894155978049874920)
gy=R(11470107855035656763776670242237886083319963338170205350339)
ell=22345678901234567830123456789012345678901234567890123456807
log_k = 21047064695533867790744883145629278009297003386558541891951
log_g = 35305194024104792001058642412688847154219207989741598909 34
temp=log_k * inverse_mod(log_g, ell) % ell
temp;g^temp;gy
```

```
8554194652334066494527973542492042121974827626609579
11470107855035656763776670242237886083319963338170205350339
11470107855035656763776670242237886083319963338170205350339
```

经验算 y= $\text{logk}*\text{logg}^{-1}$(mod ell)=

8554194652334066494527973542492042121974827626609579

以上 x,y 均比 ell 小，现在我们已知了

$$\boldsymbol{s = g^{xy}(mod p) = 3333333333333333333333333333333333333333}$$

试着求一下3333333333333333333333333333333333333333的离散对数

(7) 计算 logs

运行：./cado-nfs.py /tmp/cado.503gn9v7/p60.parameters_snapshot.2

target=3333333333333333333333333333333333333333

得到 logs= 7415243246095171081154394907486947430598565788895011470017（mod ell）

(8) 计算$log_g s$

换底公式计算$\text{logs}*\text{logg}^{-1}$(mod ell)

```
p=2234567890123456783012345678901234567890123456789012345680 71
R=GF(p)
g=R(17311125480404630 1125)
s=3333333333333333333333333333333333333
gx=R(4934187330375128509560317493098121016496489415597804987 4920)
gy=R(1147010785503565676377667024223788608331996333817020535 0339)
ell=22345678901234567830123456789012345678901234567890123456807
log_s = 741524324609517108115439490748694743059856578889501147 0017
log_g = 353051940241047920010586424126888471542192079897415989 0934
temp=log_s * inverse_mod(log_g, ell) % ell
temp;g^temp;s
```

```
55027306945661840667562194166869574746116399910142715698 96
11037027483447480485513365194265379456990408489685069653 7992
3333333333333333333333333333333333333
```

此时得到的是

$log_g s$ =55027306945661840667562194166869574746116399910142715698 96(mod ell)

而且验算也不能通过，说明 temp 值并不是最终的$log_g s$

那么如何得到$log_g s$呢?

根据小费马定理，我们只需要得到$log_g s(mod\ p-1)$即可

因为 p-1=10*ell，

已知$log_g s$ =55027306945661840667562194166869574746116399910142715698 96(mod ell)

那么$log_g s(mod\ p-1)$必然是下列之一（剩余类改写）

55027306945661840667562194166869574746116399910142715698 96+i*ell,i=0,1,2,⋯,9

于是我们穷举一下:

```
p=2234567890123456783012345678901234567890123456789012345680 71
R=GF(p)
g=R(17311125480404630 1125)
s=3333333333333333333333333333333333333
gx=R(4934187330375128509560317493098121016496489415597804987 4920)
gy=R(1147010785503565676377667024223788608331996333817020535 0339)
ell=22345678901234567830123456789012345678901234567890123456807
log_s = 741524324609517108115439490748694743059856578889501147 0017
log_g = 353051940241047920010586424126888471542192079897415989 0934
temp=log_s * inverse_mod(log_g, ell) % ell
for i in range(0,10):
    if g^(temp+i*ell)==s:
        print(i,temp+i*ell)
```

```
(3, 72539767398269887557126589783723994511315343694684641940317)
```

说明:

$log_g s$ =72539767398269887557126589783723994511315343694684641940317(mod p-1)

这个就是 3333333333333333333333333333333333333 以 g 为底 modp 的离散对数。