Math 640:348 Prof. Kontorovich Spring 2015, 3/31 lecture

We will use the index calculus method to solve a Discrete Log Problem

First lets get a “safe” prime

```
In[1]:= p = Prime[11113]
Out[1]= 117779
```

```
In[2]:= FactorInteger[(p - 1) / 2]
Out[2]= {{58889, 1}}
```

```
In[3]:= q = (p - 1) / 2
Out[3]= 58889
```

Good, so q and p are a “Sophie Germain prime pair” - in the literature q is often called a Sophie Germain prime. This means the Discrete Log Problem (DLP) mod p is not susceptible to Pohlig-Hellman attacks.

Next we need a way of testing whether a number is smooth, say 5-smooth (meaning all its prime factors are 2, 3, or 5). Here is a number called “smooth” with the property that: any n<117000 which is 5-smooth has gcd(n,smooth)=n. That is, all prime power factors (r^e for prime r) of any 5-smooth number less than p are also factors of “smooth”:

```
In[4]:= smooth = 2^Floor[Log[2, p]]  3^Floor[Log[3, p]]  5^Floor[Log[5, p]]
Out[4]= 302330880000000
```

Then “is5smooth” returns “True” if n is 5-smooth, and “False” otherwise. (There are surely better ways to implement this...)

```
In[5]:= is5smooth[n_] := (GCD[n, smooth] == n);
```

Now let's get a random primitive root, say,

```
In[6]:= g = 7;
        MultiplicativeOrder[g, p] == p - 1
Out[7]= True
```

In class, we chose a random value of a, say

```
In[8]:= a = 1919
Out[8]= 1919
```

Ok, now we're ready to solve the DLP. We need to find x (remember that x is only determined mod p-1, which is the same as mod 2q) so that g^x = a (mod p).

Step 1: solve the DLP for small primes.

Let's randomly test some values of j, hoping to find a smooth value for g^j(mod p):

```
In[9]:= For[j = 5000, j < 6000, j++,
          If[
            is5smooth[PowerMod[g, j, p]]
            ,
            Print["g^" <> ToString[j] <> " mod p is smooth"]
          ];
        ]

        g^5189 mod p is smooth

        g^5664 mod p is smooth

        g^5838 mod p is smooth
```

We found three values,

```
In[10]:= j1 = 5189;
         j2 = 5664;
         j3 = 5838;
```

How do the values of g^j(mod p) factor?

```
In[13]:= FactorInteger[PowerMod[g, j1, p]]
Out[13]= {{2, 3}, {3, 1}, {5, 2}}

In[14]:= FactorInteger[PowerMod[g, j2, p]]
Out[14]= {{2, 2}, {3, 4}, {5, 1}}
```

In[15]:=
```
FactorInteger[PowerMod[g, j3, p]]
```
Out[15]=
```
{{2, 8}, {3, 2}, {5, 2}}
```

Writing ell2 for the exponent of g which gives 2, and similarly ell3, ell5, we have the system of equations (mod 2q):

3 ell2 + ell3 + 2 ell5 = j1
2 ell2 + 4 ell3 + ell5 = j2
8 ell2 + 2 ell3 + 2 ell5 = j3,

or in "augmented" matrix form:
(in *Mathematica*, use "MatrixForm" to make it look like a matrix, instead of a sequence)

In[16]:=
```
mat = {
    {3, 1, 2, j1},
    {2, 4, 1, j2},
    {8, 2, 2, j3}
   };
MatrixForm[mat]
```
Out[17]//MatrixForm=

$$\begin{pmatrix} 3 & 1 & 2 & 5189 \\ 2 & 4 & 1 & 5664 \\ 8 & 2 & 2 & 5838 \end{pmatrix}$$

Now we just need to Gaussian Eliminate this matrix to solve for ell2, ell3, and ell5. But there's a catch! The coefficients will in general not be invertible mod 2q!

So you should first solve the equations mod 2, then mod q, and then "Chinese Remainder Theorem" them back together.

First let's look mod 2:

In[18]:=
```
MatrixForm[Mod[mat, 2]]
```
Out[18]//MatrixForm=

$$\begin{pmatrix} 1 & 1 & 0 & 1 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 \end{pmatrix}$$

So our equations are:

ell2+ell3 = 1 (mod 2)
ell5 = 0 (mod 2)

We do not have a full-rank matrix, so can't solve exactly; that's ok, in the end we'll have to guess whether ell2=0 or 1(mod 2), from which everything else will be determined...

Now let's look mod q, and Gaussian Eliminate:
Again, the matrix is:

In[19]:=
```
MatrixForm[mat]
```

Out[19]//MatrixForm=

$$\begin{pmatrix} 3 & 1 & 2 & 5189 \\ 2 & 4 & 1 & 5664 \\ 8 & 2 & 2 & 5838 \end{pmatrix}$$

So if we want to use the first row to eliminate coefficients below the "3", we first need to invert the top row. First we need to know the inverse of 3 mod q:

In[20]:=
```
inverse3modq = PowerMod[3, -1, q]
```

Out[20]= 19 630

And now we multiply the top row by 3^(-1), that is, multiply the matrix "mat" on the left by a 3x3 diagonal matrix with diagonal entries 3^(-1), 1, and 1. And of course reduce everything mod q:

In[21]:=
```
mat1 = Mod[
    DiagonalMatrix[{inverse3modq, 1, 1}].mat
    , q];
MatrixForm[mat1]
```

Out[22]//MatrixForm=

$$\begin{pmatrix} 1 & 19\,630 & 39\,260 & 40\,989 \\ 2 & 4 & 1 & 5664 \\ 8 & 2 & 2 & 5838 \end{pmatrix}$$

Next subtract off 2x(top row) from the second row, and 8x(top row) from the third row, as always, reducing mod q:

In[23]:=
```
mat2 = Mod[{{1, 0, 0}, {-2, 1, 0}, {-8, 0, 1}}.mat1, q];
MatrixForm[mat2]
```

Out[24]//MatrixForm=

$$\begin{pmatrix} 1 & 19\,630 & 39\,260 & 40\,989 \\ 0 & 19\,633 & 39\,259 & 41\,464 \\ 0 & 19\,629 & 39\,256 & 31\,260 \end{pmatrix}$$

We continue Gaussian Elimination; now we need to turn that "19633" in the middle into a "1", so we need its inverse mod q

In[25]:=
```
inverse19633 = PowerMod[19 633, -1, q]
```

Out[25]= 17 667

Multiply the second row by this number, and reduce mod q

```
In[26]:= mat3 = Mod[DiagonalMatrix[{1, inverse19633, 1}].mat2, q];
         MatrixForm[mat3]
```

Out[27]//MatrixForm=

$$\begin{pmatrix} 1 & 19\,630 & 39\,260 & 40\,989 \\ 0 & 1 & 53\,000 & 24\,217 \\ 0 & 19\,629 & 39\,256 & 31\,260 \end{pmatrix}$$

Now subtract 19630x(second row) from the first row, and 19629x(second row) from the third row:

```
In[28]:= mat4 = Mod[{{1, -19 630, 0}, {0, 1, 0}, {0, -19 629, 1}}.mat3, q];
         MatrixForm[mat4]
```

Out[29]//MatrixForm=

$$\begin{pmatrix} 1 & 0 & 41\,223 & 13\,287 \\ 0 & 1 & 53\,000 & 24\,217 \\ 0 & 0 & 35\,330 & 27\,775 \end{pmatrix}$$

Again invert the last diagonal “35330”:

```
In[30]:= inverse35330 = PowerMod[35 330, -1, q]
Out[30]= 17 320
```

Multiply through

```
In[31]:= mat5 = Mod[DiagonalMatrix[{1, 1, inverse35330}].mat4, q];
         MatrixForm[mat5]
```

Out[32]//MatrixForm=

$$\begin{pmatrix} 1 & 0 & 41\,223 & 13\,287 \\ 0 & 1 & 53\,000 & 24\,217 \\ 0 & 0 & 1 & 57\,648 \end{pmatrix}$$

And subtract

```
In[33]:= mat6 = Mod[{{1, 0, -41 223}, {0, 1, -53 000}, {0, 0, 1}}.mat5, q];
         MatrixForm[mat6]
```

Out[34]//MatrixForm=

$$\begin{pmatrix} 1 & 0 & 0 & 55\,378 \\ 0 & 1 & 0 & 18\,204 \\ 0 & 0 & 1 & 57\,648 \end{pmatrix}$$

Yay! Now we know that

```
In[35]:= ell2q = 55 378;
         ell3q = 18 204;
         ell5q = 57 648;
```

I called these ell2“q”, etc., with q’s at the end are because these are the values mod q, not mod 2q=p-1. Here comes the Chinese Remainder Theorem step:

We already know that ell5=0(mod 2), so also knowing its value mod q determines its value mod 2q:

```
In[38]:= ell5 = ell5q
Out[38]= 57 648
```

(I trust you can figure out why I did that.)

Let's check to make sure we didn't make any mistakes - does raising g to this power mod p give us 5?

```
In[39]:= PowerMod[g, ell5, p]
Out[39]= 5
```

Great!

For ell2 and ell3, we do not know their values mod 2, but we do know that ell2+ell3=1(mod 2), which means they're different (one odd, one even). Let's guess that ell2 is even. If that really was the case, then ell2 would be the same as ell2q (which is already even). So we test:

```
In[40]:= PowerMod[g, ell2q, p]
Out[40]= 117 777
```

Aha! This is *not* 2, so ell2q is not ell2! That means that ell2 is in fact odd. So:

```
In[41]:= ell2 = ell2q + q
Out[41]= 114 267
```

(Again I trust you can figure out why I did that!...)

Let's test if we got it right:

```
In[42]:= PowerMod[g, ell2, p]
Out[42]= 2
```

Two down, one to go! Since ell2 is odd, we know that ell3 is even, so

```
In[43]:= ell3 = ell3q
Out[43]= 18 204
```

As always, check your work:

```
In[44]:= PowerMod[g, ell3, p]
Out[44]= 3
```

Now Step 1 is done - we have solved the DLP for the small primes 2, 3, and 5.

Extra credit question: what would happen if we just row reduced the original matrix, and interpreted fractions as inverses mod 2q? Would we find the values of ell2, ell3, and ell5 directly? Why or why not?...

Step 2: Find smooth values of a g^-j

The next step is to find some random j so that a g^(-j) (mod p) is also smooth; then using the DLP solution above, we should be able to solve DLP for "a". Again, we loop over possible j values:

```
In[45]:= For[j = 5000, j < 6000, j++,
          If[
            is5smooth[Mod[ a PowerMod[g, -j, p], p]]
            ,
            Print["a g^-" <> ToString[j] <> " mod p is smooth"]
           ];
         ]
         a g^-5057 mod p is smooth
         a g^-5453 mod p is smooth
         a g^-5532 mod p is smooth
```

Great! We found three, but should only need one j. Let's play with the value

```
In[46]:= j0 = 5057
Out[46]= 5057
```

Then a g^(-j0) (mod p) is

```
In[47]:= Mod[ a PowerMod[g, -j0, p], p]
Out[47]= 59 049
```

which factors as

```
In[48]:= FactorInteger[59 049]
Out[48]= {{3, 10}}
```

Solving for “a” in the equation

a g^(-j0) = 3^10 = (g^ell3)^10 = g^(10 ell3)

gives

a = g^(j0 + 10 ell3),

so

```
In[49]:= x = Mod[ j0 + 10 ell3, p - 1]
Out[49]= 69 319
```

should be our desired exponent! (Note that we reduced mod p-1.) Did it work?

```
In[50]:= PowerMod[g, x, p]
Out[50]= 1919
```

```
In[51]:= a
Out[51]= 1919
```

The index calculus wins again...