Examples in pari for BSGS, rho, and index calculus

The following is the Pari history with annotations (after the //). I'm skipping lines with typos.

Setting up the group with large prime subgroup:

l= 53 // we want a "big" subgroup

? isprime(2*l+1)

%7 = 1

? p=2*l+1

%8 = 107 // so we'll work in F_107, in the subgroup of order 53

? Mod(23^2,p) // find a genrator by picking a random elt, raising to the power of the cofactor, here just 2

%9 = Mod(101, 107)

? g=% // the result was not 1, so g has order 53 and will be our generator

%10 = Mod(101, 107)

? znorder(g) // just double checking the order (only for demo puposes, not necessary)

%11 = 53

? znorder(Mod(42,p)) //42 is a good target, but is it in the subgroup?

%12 = 53 // we could also check this by computing 42^53 and getting 1

? h=Mod(42,p)   // yes, it is, so we're taking it as our target

%14 = Mod(42, 107)

? m = floor(sqrt(53)) // set up the boundary for BSGS

%16 = 7

? g0 =g^0 // begin of baby steps

%17 = Mod(1, 107)

? g1 = g^1

%18 = Mod(101, 107)

? g2=g^2

%19 = Mod(36, 107)

? g3=g2*g

%20 = Mod(105, 107)

? g4=g3*g

%21 = Mod(12, 107)

? g5=g4*g

%22 = Mod(35, 107)

? g6=g5*g // last baby step

```
%23 = Mod(4, 107)

? gminv=g^(-7) // compute g^{-m}; normally this would be (g6*g)^{-1}.

%24 = Mod(49, 107)

? h  // first giant step, this checks a1 = 0

%25 = Mod(42, 107)

? h*gminv // second giant step, checks a1 = 1

%26 = Mod(25, 107)
? %*gminv // third giant step, checks a1 = 2

%27 = Mod(48, 107)

? % * gminv //fourth giant step, checks a1= 3

%28 = Mod(105, 107) //success, we know this one from the table of BS!

? a1 = 3

%29 = 3

? a0 =3

%30 = 3

? a = a0 + m*a1 // puttinng a together

%31 = 24

? g^24 // test

%32 = Mod(42, 107) // Hurray! This matches the target.
```

Pollard rho, schoolbook method.

```
% g % 3

%33 = Mod(0, 1) // this does not do what we want: 3 and 107 are coprime, so we get a result mod 1.

? g

%34 = Mod(101, 107) // here we see that g carries the Mod 107

? lift(g) % 3 // lifting is the opposite of reduction, so lift(g) is an integer, so we can reduce that mod 3.

%35 = 2

? s = g // start of the slow walk at g

%36 = Mod(101, 107)

? f = g // start of the fast walk at g

%37 = Mod(101, 107)

? s= s^2 // 101 is 2 mod 3, so we square

%38 = Mod(36, 107)

? bs = 1 // we need to set the counters

%39 = 1

? bs = 2*bs  // we need to update the counters
```

```
%40 = 2
? cs = 0 // setting counters for power of h
%41 = 0
? cs =2*cs //updating
%42 = 2
? bf = 1 //same for fast walk
%43 = 1
? cf =0
%44 = 0
? f = f^2 // fast walk starts; we know this step already, but we need to get all variables right
%47 = Mod(36, 107)
? bf = 2*bf //updates
%48 = 2
? cf = 2*cf
%49 = 0
? lift(f) % 3 // second part of the fast step is *g
%50 = 0
? f = f*g
%51 = Mod(105, 107)
? bf = bf +1
%52 = 3
? cf = cf
%53 = 0 // done with 1 fast step; will skip updates to c until they are non-zero
? lift(s) % 3 // begin of the 2nd slow step
%54 = 0
? s = s*g
%55 = Mod(105, 107)
? bs = bs +1
%56 = 3
? lift(f) %3 // begin of the 2nd fast step
%57 = 0
? f = f*g
%58 = Mod(12, 107)
? bf = bf +1
```

```
%59 = 4
? lift(f) %3 // and the second half of it
%60 = 0
? f = f*g
%61 = Mod(35, 107)
? bf = bf +1
%66 = 5
? lift(s) % 3 // 3rd slow step
%67 = 0
? s = g*s
%68 = Mod(12, 107)
? bs = bs +1
%69 = 4
? lift(f) % 3 // 3rd fast step
%70 = 2
? f = f^2
%71 = Mod(48, 107)
? bf = 2*bf
%72 = 10
? f=f*g //I can read off that 48 %3 = 0
%73 = Mod(33, 107)
? bf = bf+1
%74 = 11
? s= s*g //begin of 4th step
%75 = Mod(35, 107)
?  /// here my laptop was going crazy and I didn't see the result, so I retyped it;
? s = s*g // this is not the next step, as 35 %3 = 2 and not 0; so I left the path, oops!
%76 = Mod(4, 107) // this is not there, I should still be at 35
? bs = bs+1
%77 = 5
? f = f*g // 4th fast step
%78 = Mod(16, 107)
? bf = bf+1
%79 = 12
```

? f = f*h // finally a step involving h!

%80 = Mod(30, 107)

? cf =cf +1

%81 = 1 // this is when I noticed that I had done an extra step, but

Try to finish this yourself; you should find a=24 again

Index calculus attacker

Trying to solve the same DLP, this time with index calculus.

107 is small, so F={2,3,5}.

The running time depends on the smoothness probability of numbers in [0,p-1].

? g^12 // we pick some random exponent of g

%89 = Mod(16, 107)

? factor(lift(%)) // then factor the integer representation of the result

%90 =

[2 4] // OK, this is quite unusual! I expect a mix of primes, but hey, we get the DL of 2 with base g

? g // reminder of what g is

%91 = Mod(101, 107)

// this equation means that 12 = 4 * log_g(2), put lg2 = log_g(2)

? lg2=3

%92 = 3

? g^17 // another random exponent

%93 = Mod(25, 107) // again, too nice

? lg5=Mod(17/2,53) // note that we're working with exponents, so those are mod 17

? g^23

%95 = Mod(100, 107) // again too nice and we know lg2 and lg5 already

? g^27

%96 = Mod(23, 107) // more normal bad luck

? g^42

%97 = Mod(13, 107) // more normal bad luck

? g^18

%98 = Mod(64, 107) // too nice, but nothing new

? g^21

%99 = Mod(86, 107)

? factor(lift(%)) // this is how you would implement this

```
%100 =
[ 2 1]
[43 1]
? g^29
%101 = Mod(79, 107) // nope

? factor(lift(%))
%102 =
[79 1]
? g^37
%103 = Mod(34, 107)
? factor(lift(%))
%104 =
[ 2 1]
[17 1]
? g^random(53) //  somebody was concerned about me depleting my brain of random numbers
%105 = Mod(49, 107)
? factor(lift(%))
%106 =
[7 2] // oh, nice, but 7 is not in our factor base, so ignore this
? g^random(53)
%107 = Mod(14, 107) // same
[[snip]]
? g^random(53)
%114 = Mod(81, 107)
? factor(lift(%)) // oh, nice! We would know all DLs for the factor basis
// alas I don't know what random exponent that was, oops!
%115 =
[3 4]
? b= random(53) // let's do it the safe way
%116 = 44
? g^b
%118 = Mod(40, 107)
? factor(lift(%))
```

```
%119 =  // nothing new here
[2 3]
[5 1]
? b= random(53) // let’s just move to the second phase; maybe 3 isn’t even in the right subgroup
%121 = 36
? h*g^b
%122 = Mod(83, 107)
? factor(lift(%)) //nope
%123 =
[83 1]
? factor(lift(h)) // also should try h itself, but no
%124 =
[2 1]
[3 1]
[7 1]
? b= random(53)
%125 = 23 //great randomness!
? h*g^b
%126 = Mod(27, 107) // argh, now we would need dg3
? b= random(53)
%128 = 7
? h*g^b
%129 = Mod(62, 107) // nope
 ? b= random(53)
%130 = 3
? h*g^b
%131 = Mod(23, 107) //nope
? b= random(53)
%132 = 14
? h*g^b
%133 = Mod(10, 107) // yeah! This factors of the factor base and we know both pieces
? factor(lift(%))
%134 =
[2 1]
```

[5 1]

A = Mod(-14 + 1*lg2 + 1*lg5,53)

%136 = Mod(24, 53)

g^A *g^14 - g^(lg2*1) * g^(lift(lg5)*1) // this is how we found it, lift(lg5) because it was mod 53

%139 = Mod(0, 107)