2016-05-27

William A. Stein

5/27/2016

Contents

1 Math 480: Open Source Mathematical Software

1.0.1 2016-05-27

1.0.2 William Stein

1.1 *Lectures 27: Number Theoretic Public key crypto (part 3/3) *

Notes:

- 1: Today is the last day of this class.
  - Your peer grading and final assignment are **due at 6pm tonight**.
  - We will tell you what we think your grade should be and why soon, and you can double check that this agrees with what you think (or not respond).
- 2: 20 min Elliptic curve Diffie-Hellman, which is perhaps the best possible system for agreeing on a shared secret.
- 3: Finish homework, etc.

width=450 class=

1.2 Elliptic Curve Diffie-Hellman

Neal Koblitz, a UW professor, co-introduced something called **elliptic curve cryptography**, which is possibly the best possible way to do public-key cryptography. (It's used heavily in bitcoin and web browsers!)

Recall, DH in the group $H = (/p)^*$, where p is a prime number.

- 1: Chose element $g \in H$.
- 2: A and B generate random a and b, and send each other g^a and g^b.
- 3: The shared secret is g^{ab}.

DH in any multiplicative group H.

- 1: Chose element $g \in H$.
- 2: A and B generate random a and b, and send each other g^a and g^b.
- 3: The shared secret is g^{ab}.

DH in an additive group E.

- 1: Chose element $P \in E$.
- 2: A and B generate random a and b, and send each other $aP = P + \cdots + P$ (a times), and bP.
- 3: The shared secret is abP.

A bad example of an additive group: $E = /p$.

```
p = next_prime(10^50)
a = ZZ.random_element(p)
b = ZZ.random_element(p)
P = Mod(1, p)
aP = a*P; aP
bP = b*P; bP
a*b*P
```

```
16310698904644024209690286239093230059318574650977
779363879454453275349383521507611751712464450006
93270582869945164299836578988914495124050707151780
```

But easy to crack, since given aP and P, can instantly compute a!

```
aP / P
a
%timeit aP / P
```

```
16310698904644024209690286239093230059318574650977
16310698904644024209690286239093230059318574650977
625 loops, best of 3: 683 ns per loop
```

A much better additive abelian group the set of points over $/p$ on an elliptic curve. E.g.,

$$E = \{(x, y) : y^2 \equiv x^3 - 3x + b \pmod{p}\} \cup \{\infty\},$$

where $p = 6277101735386680763835789423207666416083908700390324961279$ and $b = 245515554600894381$
(This is NIST curve Curve P-192.)

That E can be given the structure of (interesting) additive abelian group is fairly deep, and I won't explain it to you.

```
p = 6277101735386680763835789423207666416083908700390324961279
b = int("64210519E59C80E70FA7E9AB72243049FEB8DEECC146B9B1", base=16)
b
E = EllipticCurve(GF(p), [-3, b])
show(E)
```

```
2455155546008943817740293915197451784769108058161191238065L
```

$y^2 = x^3 + 6277101735386680763835789423207666416083908700390324961276x + 245515554600894381774$

height=300 class=

That the cardinality $\#E$ can be computed so quickly is **AMAZING** and deep, and critical to elliptic curve crypto being practical. I won't explain this either. The algorithm was discovered by the guy (Rene Schoof) in the middle in the picture on the right.

```
%time r = E.cardinality()
print(r)
is_prime(r)
```

```
CPU time: 0.00 s, Wall time: 0.00 s
6277101735386680763835789423176059013767194773182842284081
True
```

```
P = E.lift_x(int("188DA80EB03090F67CBF20EB43A18800F4FF0AFD82FF1012",\
    base=16))
P
```

```
(602046282375688656758213480587526111916698976636884684818 :
174050332293622031404857552280219410364023488927386650641 : 1)
```

```
P.order()
```

```
6277101735386680763835789423176059013767194773182842284081
```

```
Q = P + P; Q
```

```
(5369744403678710563432458361254544170966096384586764429448 :
5429234379789071039750654906915254128254326554272718558123 : 1)
```

```
Q + P
```

```
(2915109630280678890720206779706963455590627465886103135194 :
2946626711558792003980654088990112021985937607003425539581 : 1)
```

```
Q[0] + P[0]
```

```
5971790686054399220190671841842070282882795361223649114266
```

```
a = ZZ.random_element(r)
aP = a*P; aP
b = ZZ.random_element(r)
bP = b*P; bP
```

```
(1010112466383532588414336248664916157523291276178422843136 :
90439193584477906557216196686946163581516859214286736817 : 1)
(3658450568571674634730726759471282568571143029108933055159 :
5045525967726998051020071617599851419165873752343087175548 : 1)
```

```
secret = a*bP
secret
```

```
(96241905756100882823771114885612712126104299717649570085 :
968127913256723831389111723898882476183306459135854559781 : 1)
```

```
b*aP
```

```
(96241905756100882823771114885612712126104299717649570085 :
968127913256723831389111723898882476183306459135854559781 : 1)
```

```
(a*b)*P
```

```
(96241905756100882823771114885612712126104299717649570085 :
968127913256723831389111723898882476183306459135854559781 : 1)
```