Complexity, Cryptography, and Financial Technologies

Lecture 7 – Elliptic Curve Cryptography

Chan Nam Ngo

Why ECC -RSA key length

- **The combination of**
  - algorithmic improvements
  - and more powerful computers
- **have pushed up the length of moduli that can be factored.**
  - 1999: The 512-bit RSA modulus from the RSA Factoring Challenge was factored.
  - 2003: A 576-bit RSA modulus was factored.
  - Regulierungsbehörde für Telekommunikation und Post (RegTP, 2004): recommends at least 1024-bit moduli, preferably 2048-bit.
- **It is similar for DLOG and DH, etc.**

Why ECC – Key size matters ... (2)

There is a concern that RSA keys are getting too large for applications with limited storage or communications facilities (e.g. smart cards).

Symmetric Encryption	RSA	ECC
80	1024	160
112	2048	224
128	3072	256
192	7680	384
256	15360	512

Elliptic Curve

- **An elliptic curve E is a set of points (x,y) s.t.**

  $$y^2 = x^3 + ax^2 + bx + c$$

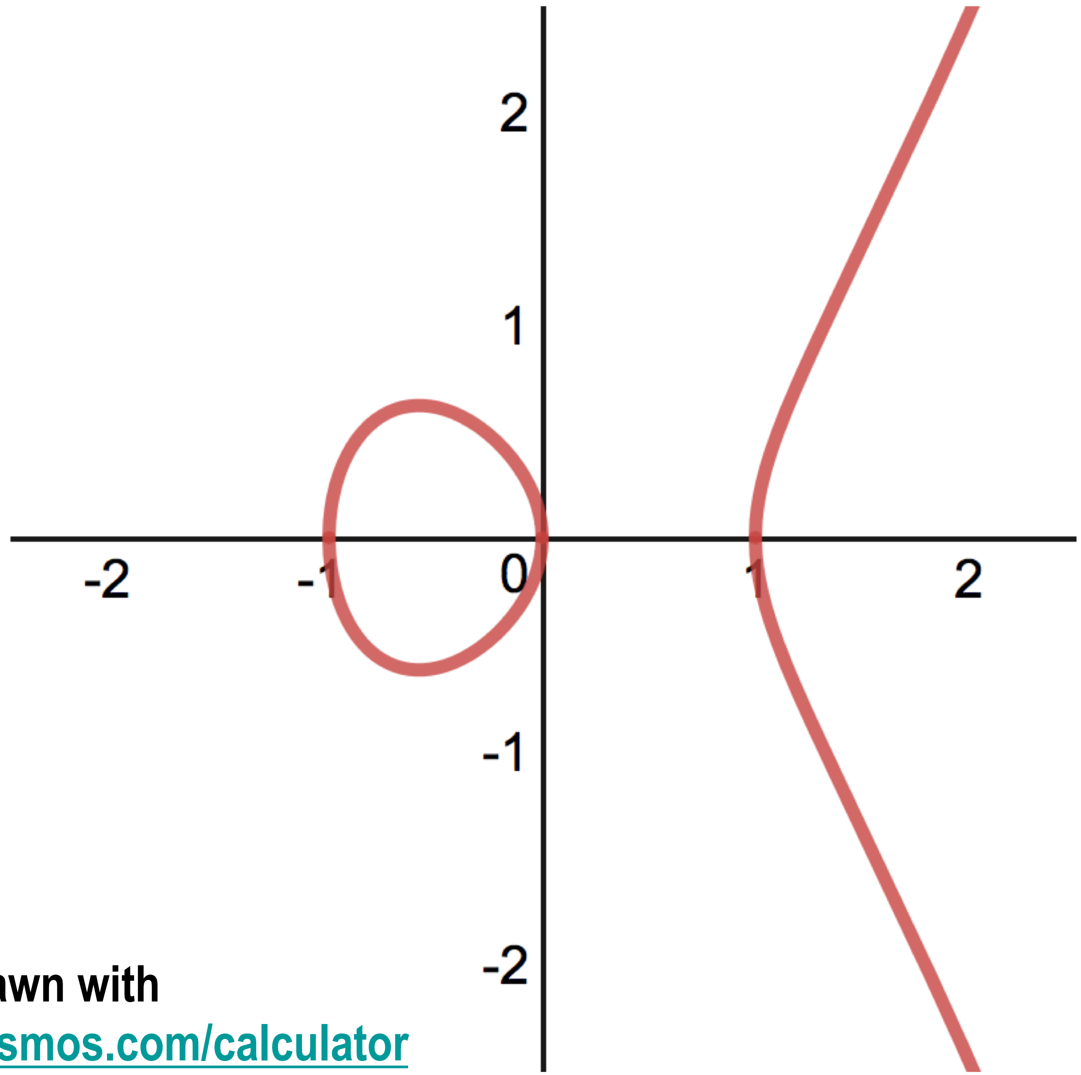
- **where x, y, a, b and c are in a set K,**
- **e.g. K is the set of**
  - Q: rational numbers
  - R: real numbers
  - Z_p: integers mod p
- **we can write**

  $$E = \{(x,y) \mid x,y \in K,\ y^2 = x^3 + ax^2 + bx + c\}$$

- **In most cases we use elliptic curves of the form**

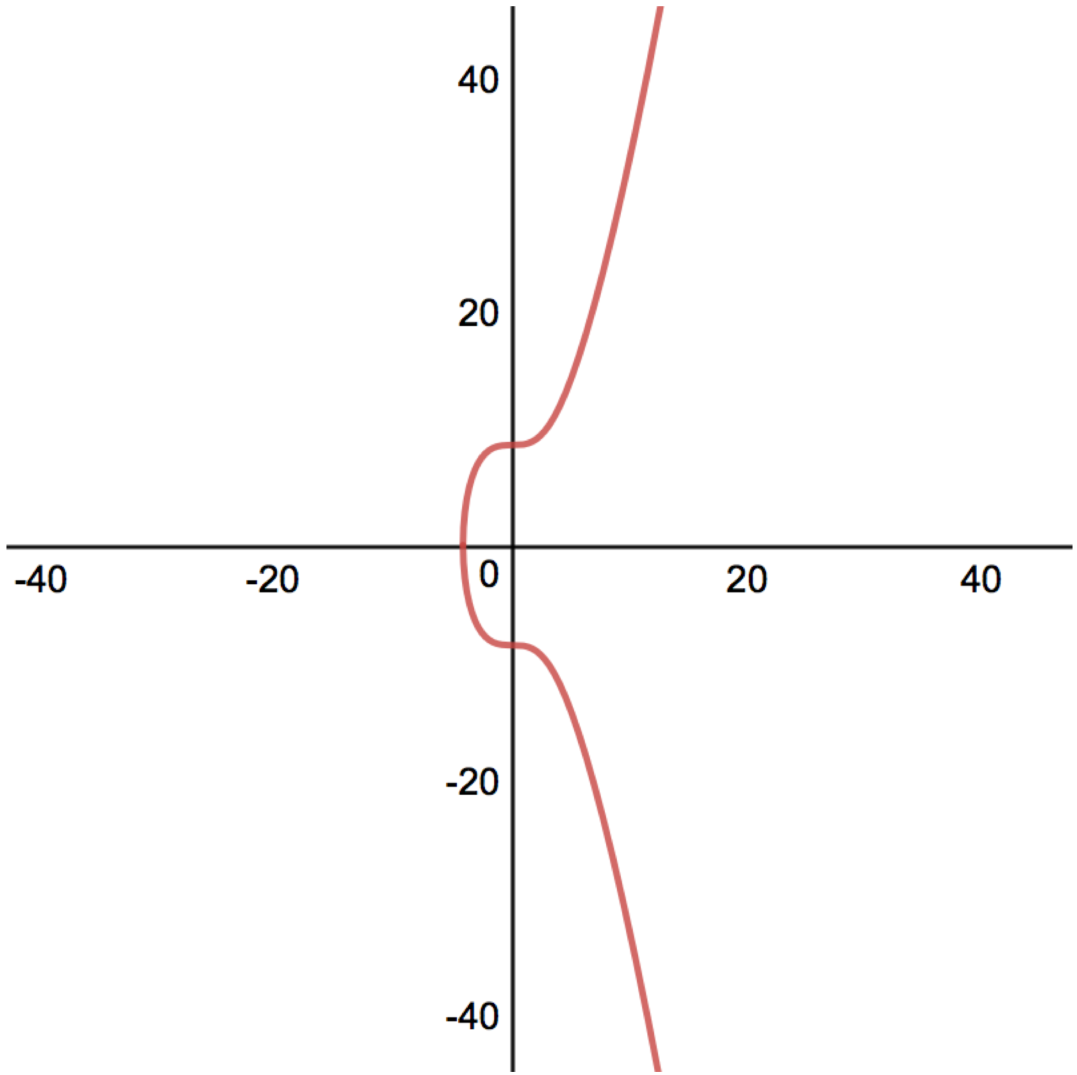
  $$E = \{(x,y) \mid x,y \in K,\ y^2 = x^3 + bx + c\}$$

$x, y \in R, y^2 = x^3 - x$



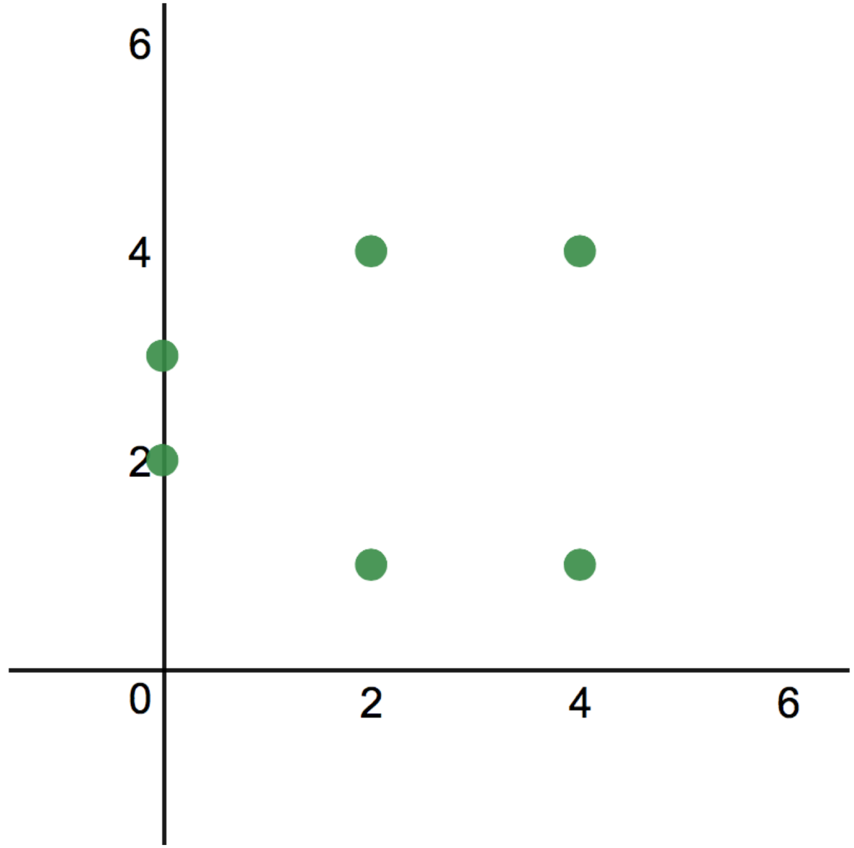
Drawn with
https://www.desmos.com/calculator

$x, y \in R, y^2 = x^3 + 73$



$x, y \in Z_5, y^2 = x^3 + 2x - 1$

x	0	1	2	3	4
y	2	3	1	4	4



Adding two different points on an Elliptic Curve

UNIVERSITY OF TRENTO

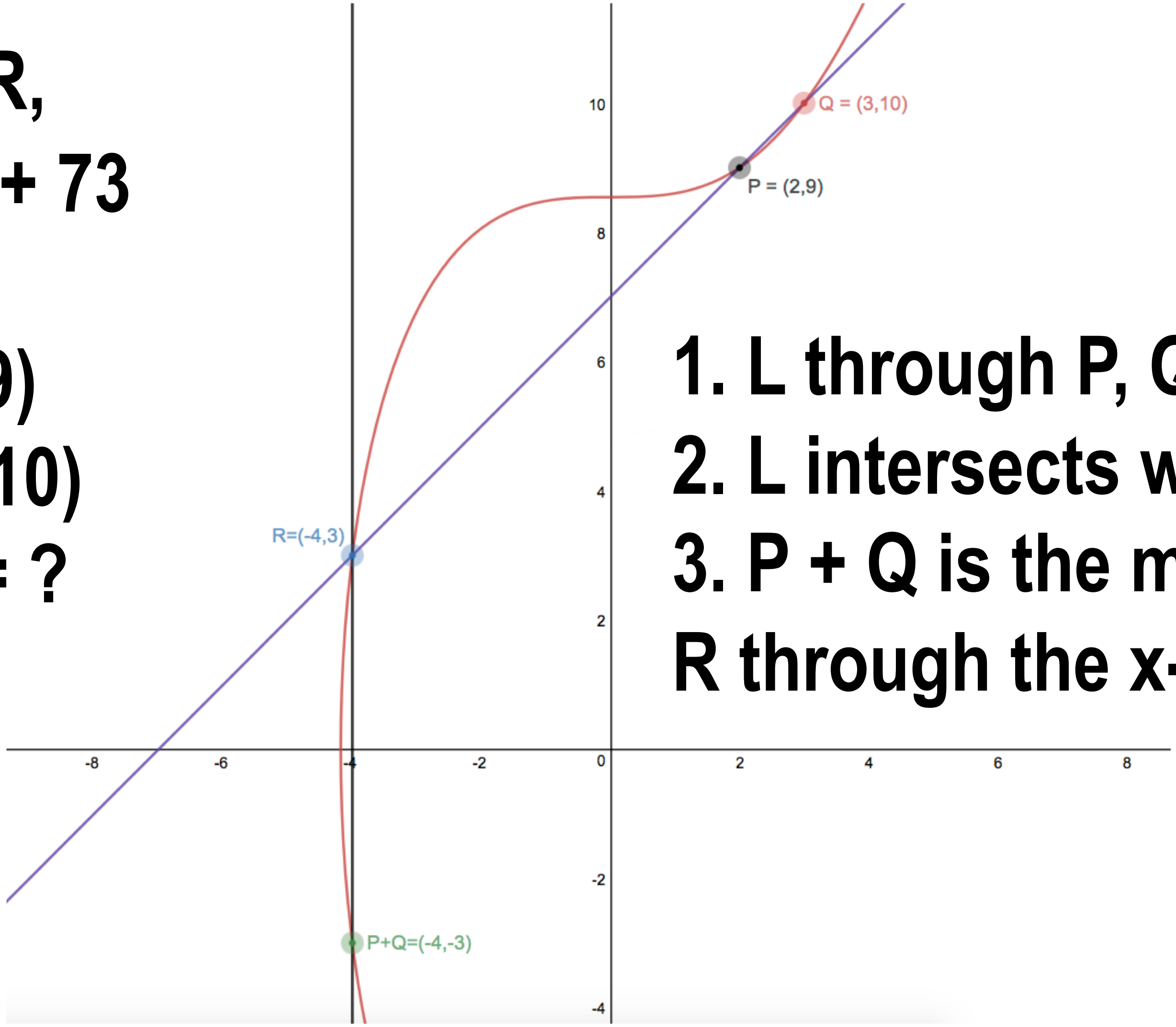
$x, y \in R,$

$y^2 = x^3 + 73$

$P = (2,9)$

$Q = (3,10)$

$P + Q = ?$



1. L through P, Q
2. L intersects with E at R
3. P + Q is the mirror of R through the x-axis

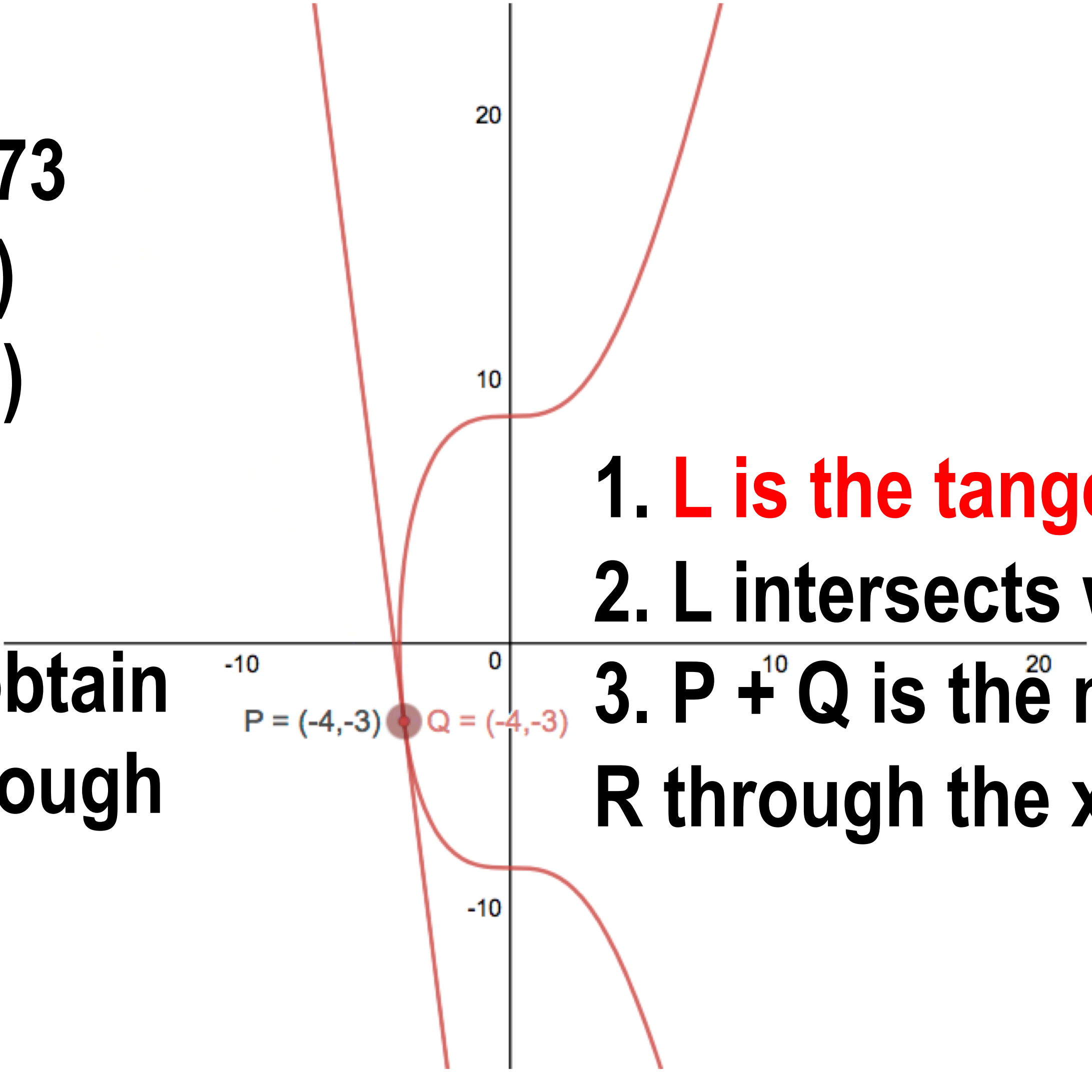
Adding two same points on an Elliptic Curve

$x, y \in R$,

$y^2 = x^3 + 73$

$P = (-4,-3)$

$Q = (-4,-3)$



Cannot obtain line L through P and Q

$P + Q = ?$

1. L is the tangent line to E
2. L intersects with E at R
3. P + Q is the mirror of R through the x-axis

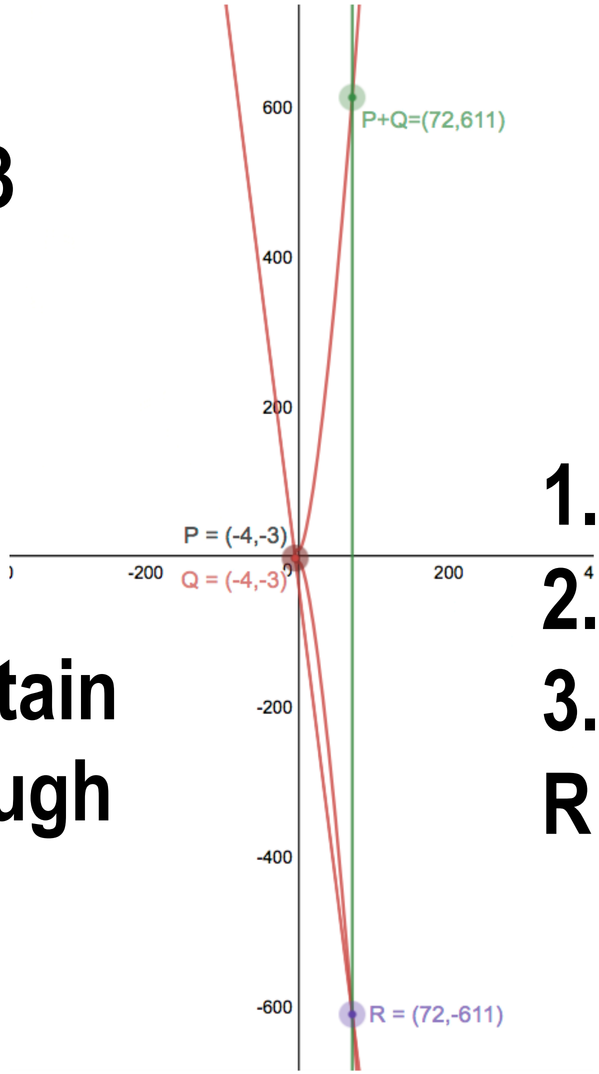
Adding two same points on an Elliptic Curve (2)

$x, y \in R$,

$y^2 = x^3 + 73$

$P = (-4,-3)$

$Q = (-4,-3)$



1. L is the tangent line to E
2. L intersects with E at R
3. P + Q is the mirror of R through the x-axis

Cannot obtain line L through P and Q

P + Q = ?

The point at infinity O (∞,∞)

- **O (∞,∞) sits at both**
  - the top
  - and the bottom
  - of the y-axis
- **We have**
  - P + O = P
  - P – P = O
- **P – Q = P + (-Q)**

The Addition Law

- Let E be given by $y^2 = x^3 + bx + c$
- and $P = (x, y)$ and $Q = (u,v)$
- then $P + Q = (s,t)$ such that

$$s = m^2 - x - u$$

$$t = m(x - s) - y$$

If $P \neq Q$ then $m = (v - y)/(u - x)$

If $P = Q$ then $m = (3x^2 + b)/(2y)$

If m is ∞, then $P = O = (s,t) = (\infty, \infty)$

- Additionally we have $P + O = P$

The Addition Law (2)

- **E forms an abelian group {E,+}**
  - Associativity

    $$(P + Q) + R = P + (Q + R)$$

  - Commutativity

    $$P + Q = Q + P$$

- **Identity element of {E,+} is O**

  $$P + O = P$$

Elliptic Curves in Z_p

- $x, y \in Z_5,\ y^2 = x^3 + 4x + 4$

x	0		1		2	4	
y	2	3	2	3	0	2	3

- $P = (1,2),\ Q = (4,3)$
- $P + Q = (s,t)$

  $m = (3-2)/(4-1) = 1*3^{-1} = 2$

  $s = 2^2 - 1 - 4 = 4$

  $t = 2(1-4) - 2 = 2$

- $P + Q = (4,2)$

ECDLOG

- Let E be an elliptic curve
- Let P, Q be two points on E
- and $Q = \underbrace{P + P + \ldots + P}_{\text{k times}}$ (we can write $kP = Q$)
- Find k → very hard
- Attacks on DLOG in Finite Fields, e.g.
  - Pohlig-Hellman
  - Index Calculus
- don't work well on DLOG on Elliptic Curves

Elliptic Curve Cryptosystems

- **EC versions for DLOG-based public key cryptosystems exist, e.g.**
  - Diffie-Hellman Key Exchange (ECDH)
  - El-Gamal Public Key Encryption
  - Digital Signature (ECDSA)
    - We go into details of ECDSA in the next lecture
- **Given our present knowledge about ECDLOG, EC schemes can be used with much shorter keys than RSA.**
- **Popular in applications where cryptographic operations are performed on smart cards.**

Diffie-Hellman (DH) Key Exchange - EC version

UNIVERSITY OF TRENTO

- **Alice and Bob wants to obtain a shared secret key for secure communication**
- **but Eve can see every information exchanged between Alice and Bob**
- **Can we construct a protocol such that Eve cannot derive the secret key from the public transcript?**
- **Based on problems related to EC DLOG**
  - Computational DH, EC version
    - Given $A = xG$, $B = yG$, find $C = xyG$
  - Decisional DH, EC version
    - Given $A = xG$, $B = yG$ and $C = zG$, determine if $z = xy$

- **Alice**
  - Pick random x
  - Send xG to Bob
  - Receive yG
  - Compute x(yG)
- **Bob**
  - Pick random y
  - Send yG to Alice
  - Receive xG
  - Compute y(xG)

Computing Q = kP

- **The Double and Add algorithm in $E(Z_p)$**
  - $k = k_0 + 2k_1 + 4k_2 + \ldots + 2^m k_m$
  - Similar to Square and Multiply algorithm in Z_p
- **The algorithm**
  - N = P
  - Q = O
  - for i from 0 to m do
    - if $k_i = 1$ then
      - Q = Q + N
    - N = N + N
  - return Q

Diffie-Hellman (DH) Key Exchange - EC version - Example

- **Given $x,y \in Z_{7211}$, $y^2 = x^3 + 7206x + c$**
- **and $G = (3,5)$**

- **Alice**
  - $x = 12$
  - $xG = (1794, 6375)$
  - Receive yG
  - $x(yG)$

    $= 12(1794, 6375)$

    $= (1472, 2098)$

- **Bob**
  - $y = 23$
  - $yG = (3861,1242)$
  - Receive xG
  - $y(xG)$

    $= 23(3861, 1242)$

    $= (1472, 2098)$

El-Gamal Cryptosystem – Public Key Encryption - EC version

- **(pk,sk) ← KeyGen()**
  - Fix a large prime p,
  - Generate an EC

    $x, y \in Z_p, y^2 = x^3 + bx + c$
  - Fix a point G on E
  - Randomly pick k in Z_p
  - Compute Y = kG
  - Return pk = (p,G,Y) and sk = (k)
- **c ← Enc(pk,m)**
  - Randomly pick r in Z_p
  - Compute R = rG and M = m + rY = m + rkG
  - Return c = (R,M)
- **m = Dec(sk,c)**
  - Return m = M - kR = m + rkG - krG

El-Gamal Cryptosystem – Public Key Encryption - EC version - Example

UNIVERSITY OF TRENTO

- **(pk,sk) ← KeyGen()**
  - p = 8831
  - Generate an EC

    $x, y \in Z_{8831}$, $y^2 = x^3 + 45x + c$
  - G = (4,11)
  - k = 3
  - Y = xG = (413,1808)
  - Return pk = (8831,(4,11),(413,1808)) and sk = (3)
- **c ← Enc(pk,(5,1743))**
  - r = 8
  - R = rG = (5415,6321)
  - M = m + rY = (6626,3576)
  - Return c = ((5415,6321),(6626,3576))
- **m = Dec(sk,c)**
  - Return m = M - xR = (6626,3576) – (673,146)

    = (6626,3576) + (673,-146) = (5, 1743)

Suggested Readings

- **Introduction to Cryptography with Coding Theory, book by W. Trappe and L. C. Washington.**
  - Chapter 16 is on Elliptic Curves
- **NIST standard FIPS 186-2**
  - https://csrc.nist.gov/csrc/media/publications/fips/186/2/archive/2000-01-27/documents/fips186-2.pdf
  - see Appendix 6 for a list of recommended elliptic curves

UNIVERSITY OF TRENTO

Complexity, Cryptography, and Financial Technologies

Lab 2 – Elliptic Curve Cryptography

Chan Nam Ngo

libff

- **libff is a dependency of libsnark**
- **libff provides operations for elliptic curves over a prime Finite Field**
  - edwards: based on Edwards curve, 80 bits of security
  - bn128: based on Barreto-Naehrig curve, 127 bits of security
  - alt_bn128: alternative to bn128
- **by default, the curve choice is bn128**
- **to choose a curve, use compilation flag**

  -DCURVE=choice

  where choice is one of: ALT_BN128, BN128, EDWARDS

libff (2)

- **libff examples for EC**
  - libff/libff/algebra/curves/tests/test_groups.cpp
- **An important operation on EC is scalar multiplication**
  - e.g. xG where x is a scalar and G is a based point
  - see libff/libff/algebra/curves/curve_utils.hpp

Exercise

- **Implement the Double and Add algorithm in $E(Z_p)$**
  - check the output against the libff built-in function
- **Implement the EC version (using libff) of**
  - Diffie-Hellman Key Exchange (ECDH)
  - El-Gamal Public Key Encryption