10/01/22

Lecture 8:

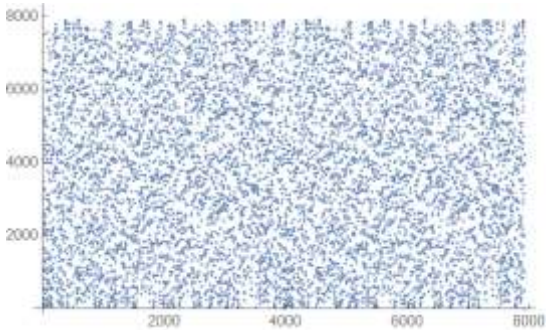
Elliptic Curve Crypto

A (minimal) introduction

Why bothering with ECC?

- **Computational security = Hard problems**
  - ⇨Easy in one way
  - ⇨Hard in the other way
- **Example: discrete logarithm**
  - ⇨Z*(p), p>2 prime, group order q
    - → $x \rightarrow g^x \bmod p$ easy
    - → $g^x \rightarrow x \bmod p$ hard (inverso hard, basta vedere il plot, sembra random!)

2^x mod 7919




```
In[31]:= p = 59;
         l = PowerMod[2, Range[1, p - 1], p]
         ListPlot[l]
```

```
Out[32]= {2, 4, 8, 16, 32, 5, 10, 20, 40, 21, 42, 25, 50, 41, 23, 46, 33, 7, 14, 28, 56, 53, 47, 35, 11, 22, 44, 29, 58,
          57, 55, 51, 43, 27, 54, 49, 39, 19, 38, 17, 34, 9, 18, 36, 13, 26, 52, 45, 31, 3, 6, 12, 24, 48, 37, 15, 30, 1}
```

tale problema è più esteso

- ⇨Not restricted to Z*(p), can be **any** cyclic group Gp

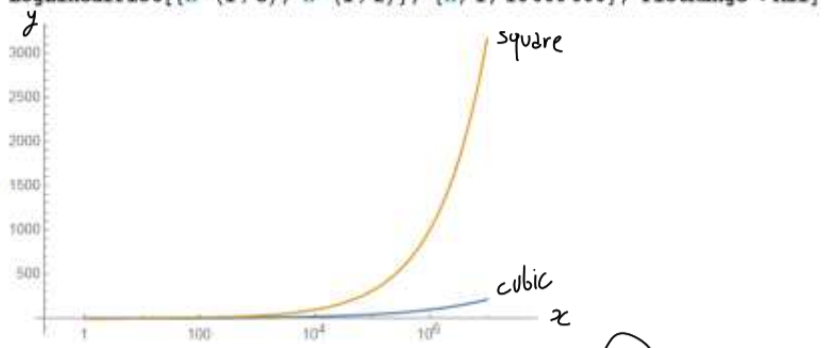
Why bothering with ECC?

➔**Same problem (e.g. Dlog, factoring), different "hardness" in different groups!**

⇨Dlog in Z*(p): $\exp(O(n^{1/3}))$

⇨Dlog in EC: $\exp(O(n^{1/2}))$

⇨*Dlog in (Z(p),+)): poly!*



Symm key equiv	modulus size	Elliptic Curve
80 bits	1024 bits	163 bits
128 bits	3072 bits	283 bits
256 bits	**15360** bits	571 bits

EC scales well (so far) with increased security parameter!!

Obvious long term deployment choice (now clear why so many new EC ciphers in TLS)!

Semplice aritmetica, NO groups!

Elliptic curves, cosa sono?

➔They are NOT ellipses!

- ⇨They are special cubic curves
- ⇨Named from elliptic integrals

➔General (Weierstrass) expression

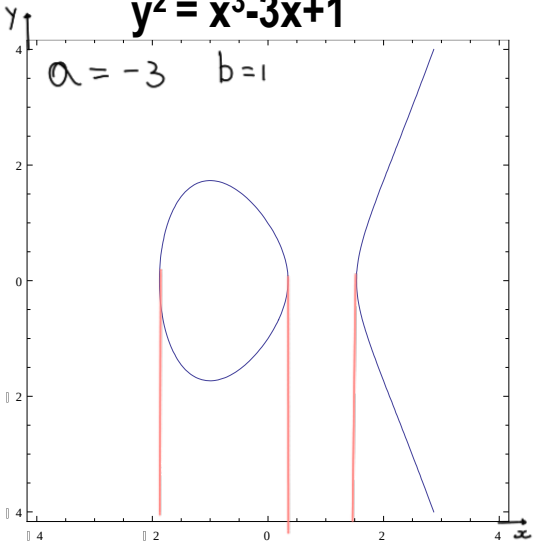
esclude punti di tangenza, tipo:

parametri variabili

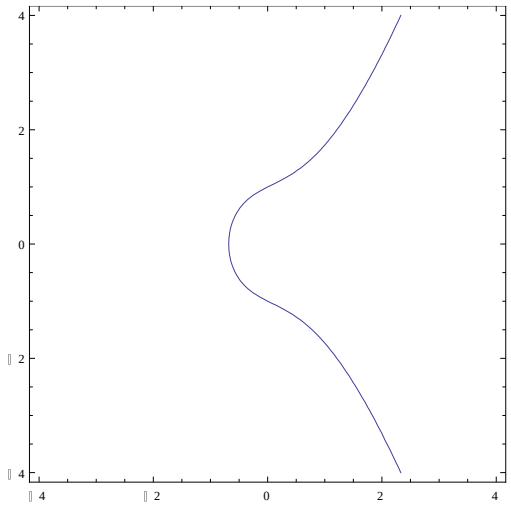
$$y^2 = x^3 + ax + b \quad \textit{where} \quad 4a^3 + 27b^2 \neq 0$$

fissati

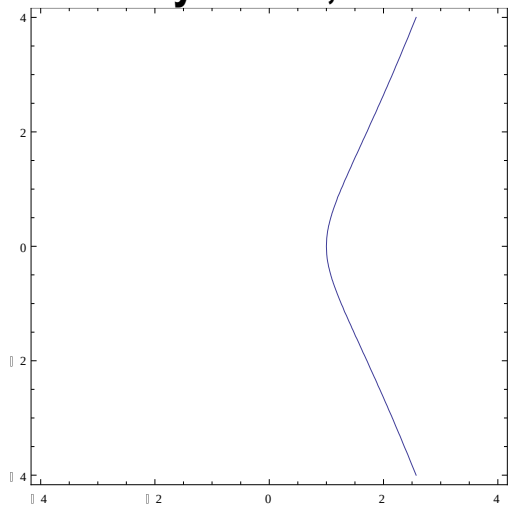
$y^2 = x^3 - 3x + 1$



$y^2 = x^3 + x + 1$

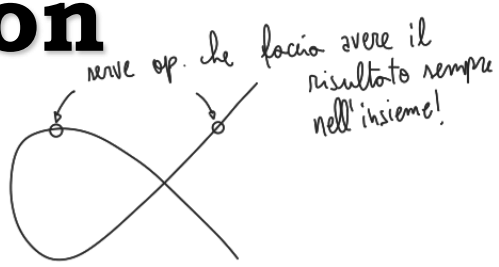


$y^2 = x^3 - 1$, $a = 0$



che posso farci?

We need an operation



➔**We want an operation**

⇨call it + or ○ , ∃, ma non è op. "classica"

→ NON è il "+" che conosciamo!

➔historically called +, though someone today may prefer mult. notation

➔**So that P+Q=R → R BELONGS to the curve** (op. chiusa!)

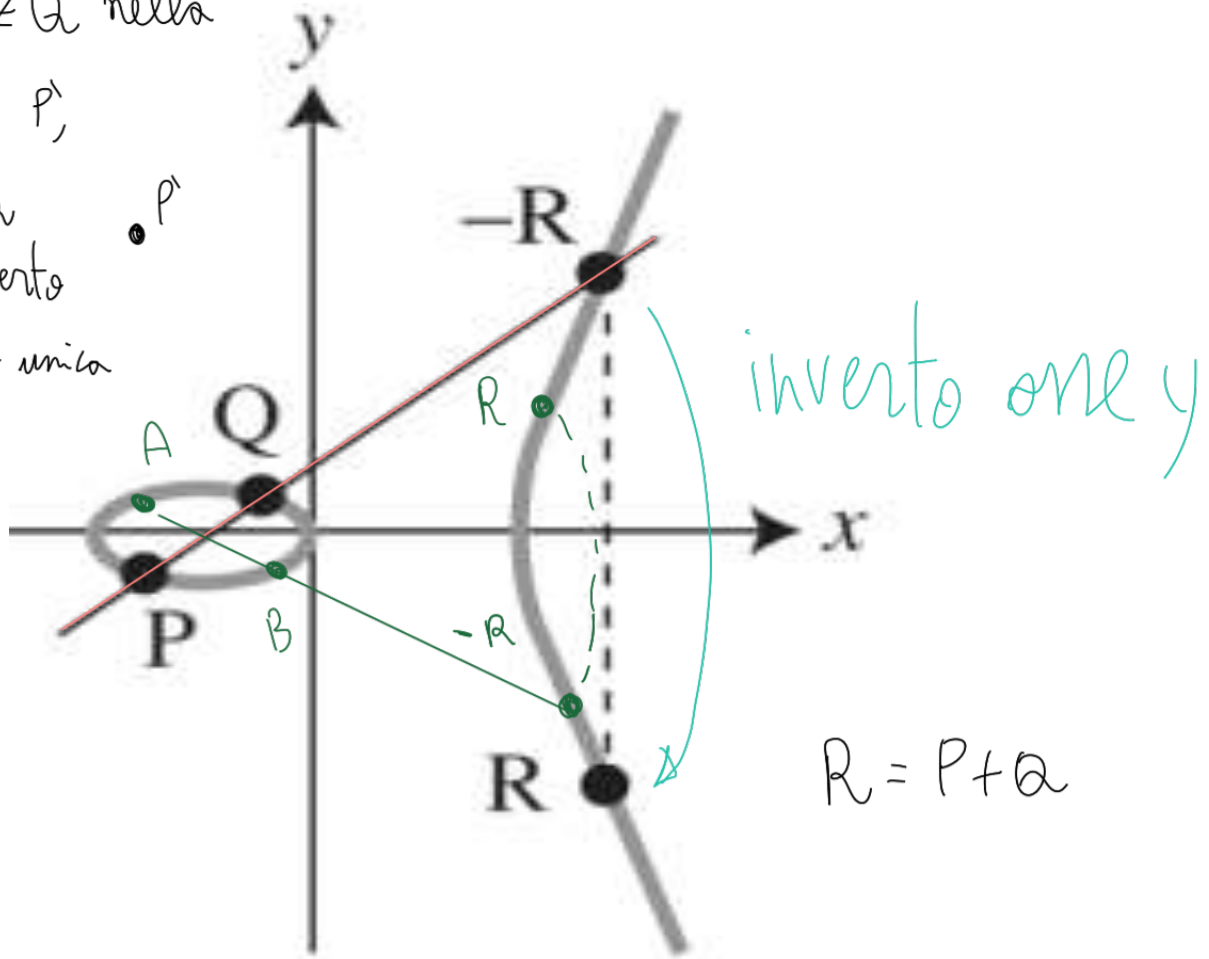
➔**AND the operation gives algebraic group structure**

L'op è : # Elliptic points addition P+Q

(geometric)

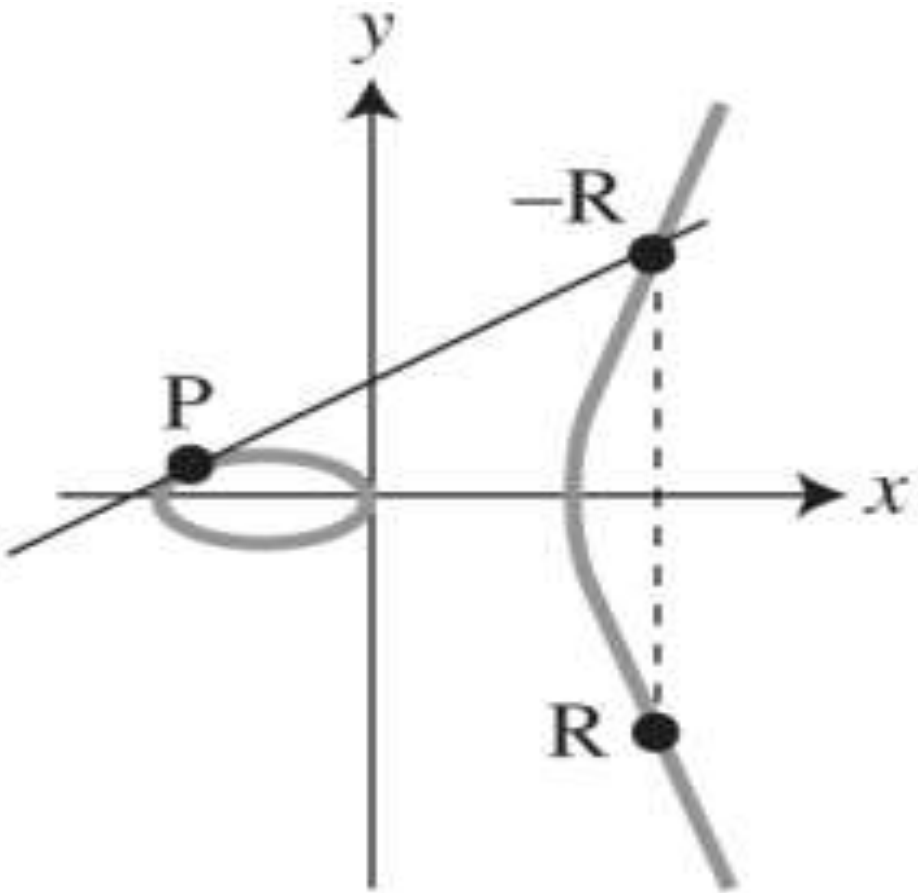
Prendo P, Q con $P \neq Q$ nella curva, non fuori come P', la linea che passa per P, Q tocca 3° punto "-R", e inverto one y. Se $P = Q$ non ho unica retta!

Se $P // Q$ (stesso x, diverso y) NON TOCCO MAI, va all'∞, definisco punto "∅"



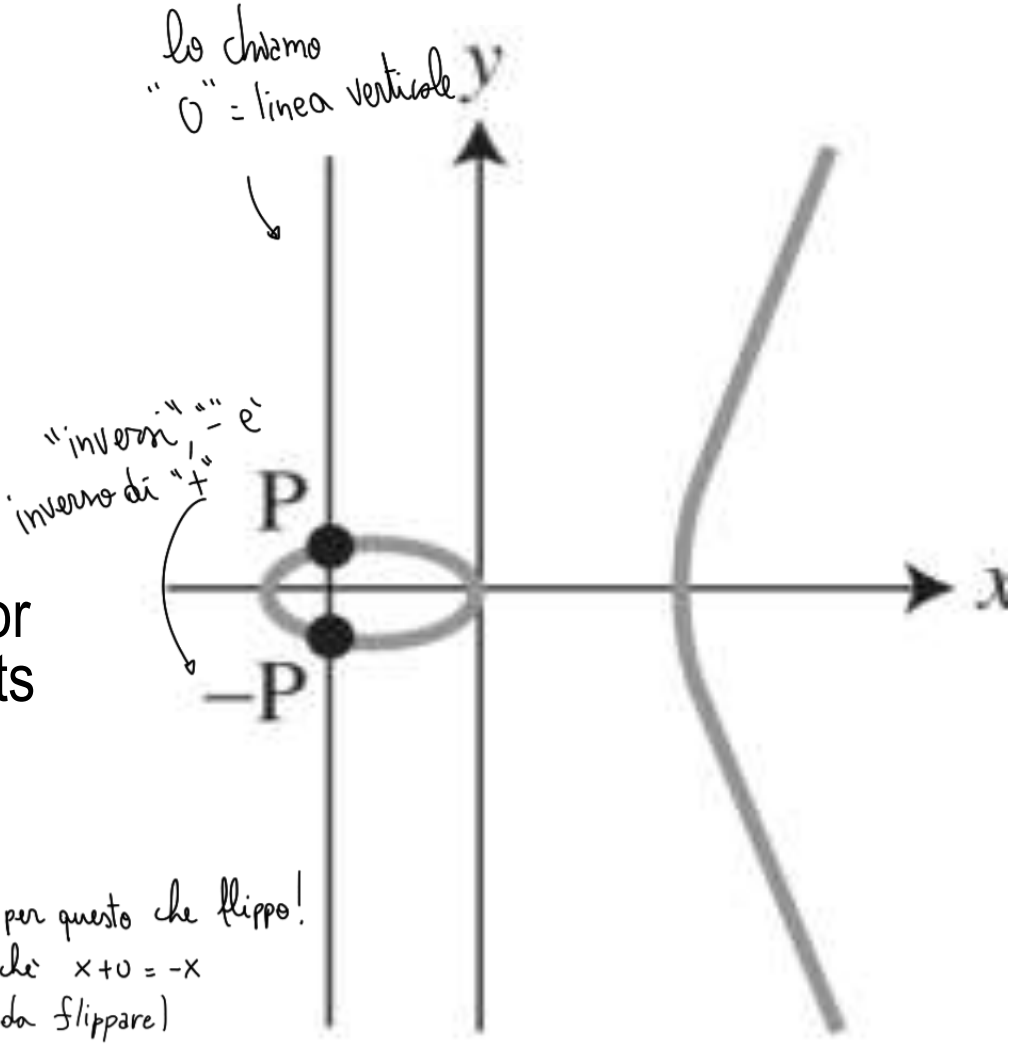
Elliptic points addition P+P

(geometric)

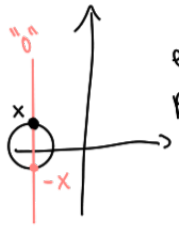


Elliptic points addition P-P
(geometric)

- **"complete" the curve by adding point O=-O to infinity**
- **P+O=P**
  - O is (additive) zero for the group of EC points
  - Now clear why the "inversion" ☺



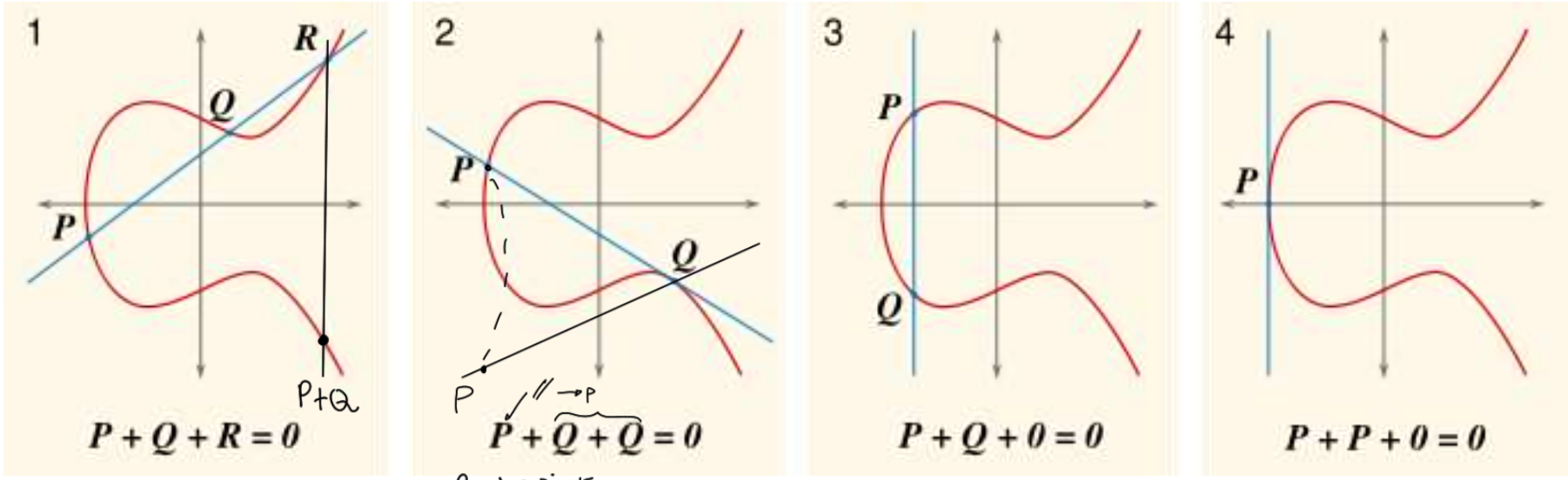
- $P+(-P)=[\emptyset]$
- se faccio $P+\emptyset =$



è per questo che flippo!
Perché $x+0=-x$
(da flippare)

General rule

➔ **If three points of an elliptic curve lay on a same line, their sum is 0**



Se fosse $P+P+P+P = K \cdot P$ (è solo NOTAZIONE), $\underbrace{P \cdot P \cdot P \cdot P}_{K} = P^K$

Algebraic expressions, posso trovare 3° punto?

➔ **Can be derived from geometric interpretation**

$$P = (x_1, y_1)$$
$$Q = (x_2, y_2)$$

coordinate (standard arithmetic), $y_1 = x_1^3 + a x_1 + b$, SODDISFA WEIERSTRASS!

$$R = P + Q = (x_3, y_3)$$

ALGEBRIC EXPRESSION

$$x_3 = \lambda^2 - x_1 - x_2$$
$$y_3 = \lambda(x_1 - x_3) - y_1$$

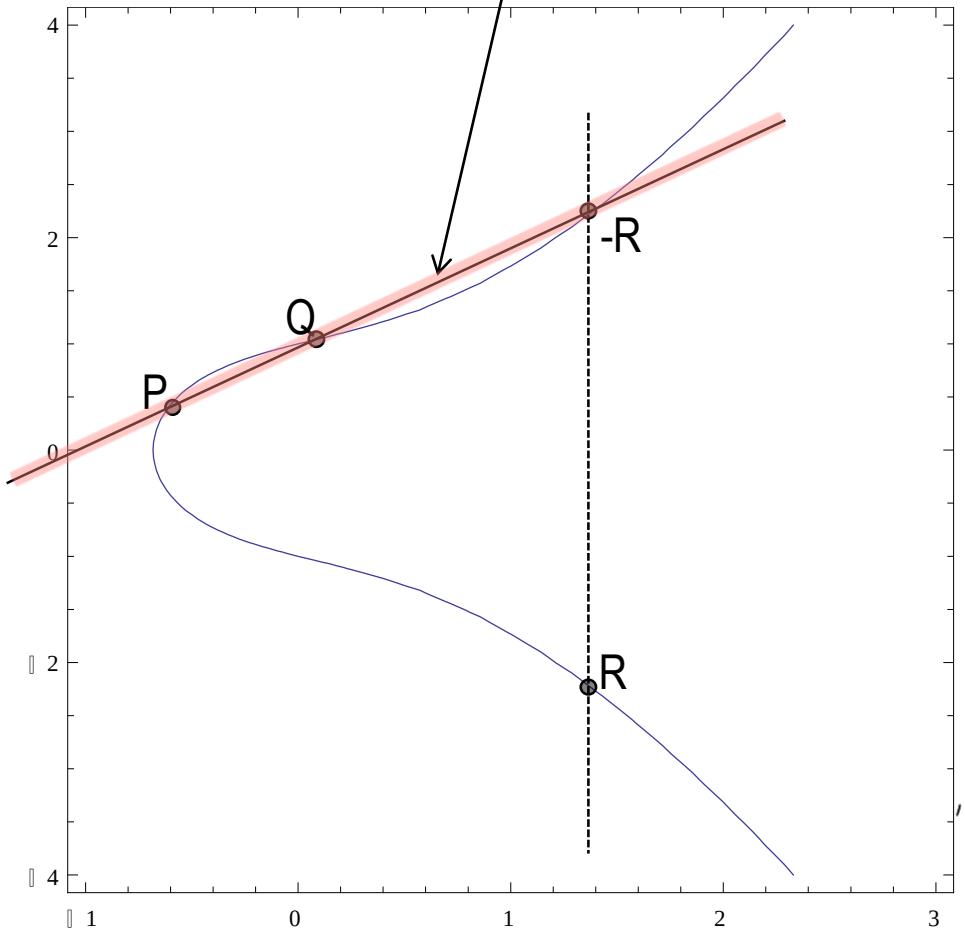
, dove:

$$\lambda = \begin{cases} \dfrac{y_2 - y_1}{x_2 - x_1} & P \neq Q \quad \text{(come coeff. angolare)} \\ \dfrac{3x_1^2 + a}{2y_1} & P = Q \quad \text{(tangente)} \end{cases}$$

P≠Q sketch

equazione linea passante per (P,Q)

$$y = y_1 + \frac{y_2 - y_1}{x_2 - x_1}(x - x_1) = \lambda(x - x_1) + y_1$$



$y^2 = x^3 + ax + b$ ≔ eq. curva

$(\lambda(x - x_1) + y_1)^2 = x^3 + ax + b$ (Porto a destra)

$0 = x^3 - \lambda^2 x^2 + \ldots^{o(x^2)}$

x^3: ho 3 punti: X_P, X_Q, X_R, mi manca solo X_R

recall:

$(x - x_1)(x - x_2)(x - x_3) =$ ($x_1 = x_P$, $x_2 = x_Q$, miss $x_3 = x_R$)

$= x^3 - (x_1 + x_2 + x_3)x^2 + \ldots^{o(x^2)}$

hence (lineare!)

$\lambda^2 = x_1 + x_2 + x_3 \rightarrow x_3 = \lambda^2 - x_1 - x_2$

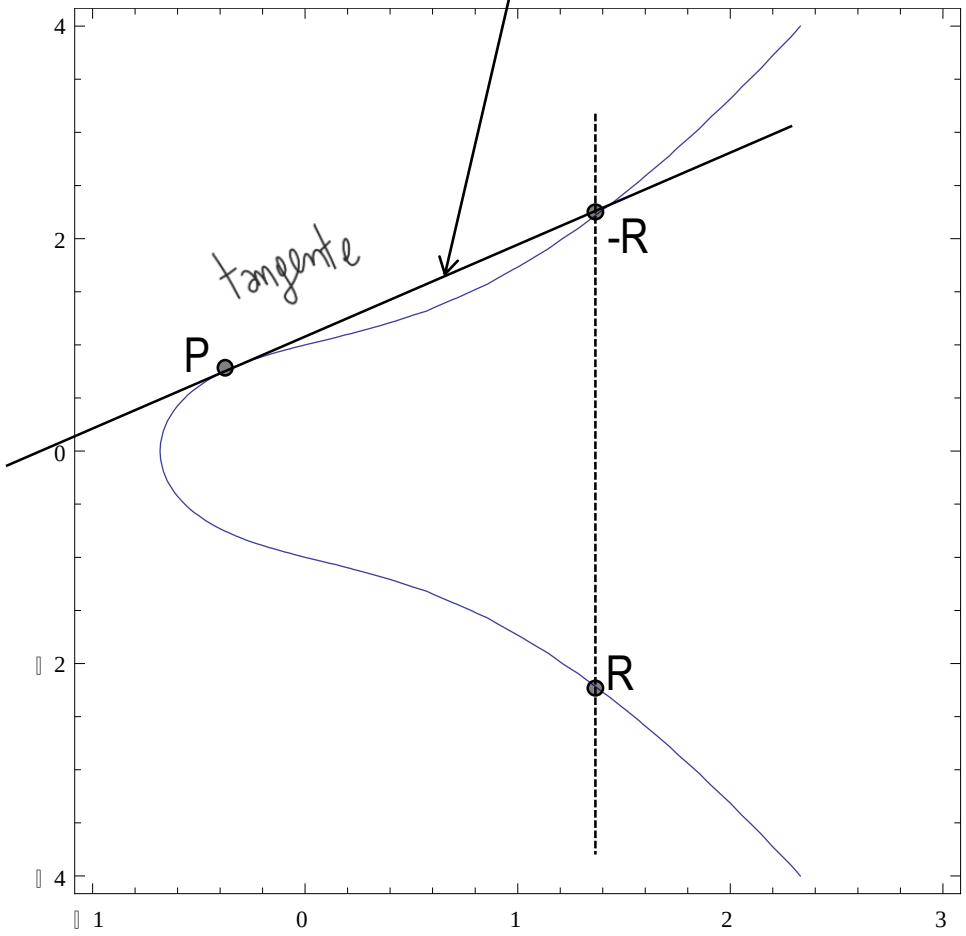
y_3 now trivial from line eq.

$y_3 = \lambda(x_1 - x_3) - y_1$

P=Q sketch

derivata

$$y = y_1 + y'_1 (x - x_1) = \lambda (x - x_1) + y_1$$



$$y^2 = x^3 + ax + b \text{ , derivo !}$$

$$2y\frac{dy}{dx} = 3x^2 + a \rightarrow \frac{dy}{dx} = \frac{3x^2 + a}{2y}$$

$$\lambda = y'_1 = \frac{3x_1^2 + a}{2y_1}$$

Follow up as before...

il resto è come prima!

Abbiamo lavorato su real numbers, con tutti i punti della curva.
Posso cambiare set di numeri? Se passassi a modular p integers!?

Elliptic curve group
(over Zp)

EC over modular integers

- **Geometric interpretation** → algebric formulations (cioè lavoro con formule)
  - Over real numbers
- **Our applications**
  - Over modular integers
- **Define "curve" as** (ora è set punti, niente più PLOT)
  - $E_p(x,y)$ such that
    - x,y integers in Zp (integer mod p), coordinate INTERE!
    - $y^2 \bmod p = x^3 + a\,x + b \bmod p$ (da soddisfare!)
- **Finite field → finite # of points**

(sono p^2, cioè se ho p 'x' → p 'y')

Example: $(x^3+x+1) \bmod 5$

→ non ho a che fare con l'ordine del gruppo che genera (che ha 9 elementi)

↳ PLOT " < " ma ora ho interi e mod p

➔ All points:

perché MOD 5

⇨ pairs (i,j), i in [0,4], j in [0,4] s.t. $j^2=i^3+i+1 \bmod 5$

(se $x \in [0,4]$ e $y \in [0,4]$ → 5·5 punti totali)

⇨ And O (∅ artificiale)

$$E(Z_5)=\{O,(0,1),(0,4),(2,1),(2,4),(3,1),(3,4),(4,2),(4,3)\}$$

(RIMANE UN GRUPPO, MA È FINITO)

⇨ Result: 9 points

⇨ Let P=(0,1)

➔ k P=P+P+…+P
generates all the points

$P=(0,1)$ → devo fare tutta la computazione di prima!

$2P=(0,1)+(0,1)=(4,2)$

$3P=(4,2)+(0,1)=(2,1)$

$4P=(2,1)+(0,1)=(3,4)$

$5P=(3,4)+(0,1)=(3,1)$

$6P=(3,1)+(0,1)=(2,4)$

$7P=(2,4)+(0,1)=(4,3)$

$8P=(4,3)+(0,1)=(0,4)$

$9P=(0,4)+(0,1)=O$

tutti i valori $\in E(Z_5)$

$10P = O + (0,1) = (0,1)$

example computation

$(0,1)+(0,1)$

step1: compute λ

$\lambda = \frac{3x_1^2 + a}{2y_1} = 1 \cdot 2^{-1} = 1 \cdot 3 = 3 \mod 5$

$\frac{3 \cdot 0 + 1}{2} = 1 \cdot 2^{-1} \mod 5$; 2^{-1}: 3 inverso

$2y_1$: modular inverse

step2: compute x_3

$x_3 = \lambda^2 - \cancel{x_1} - \cancel{x_1} = 3^2 = 4 \mod 5$

step3: compute y_3

$y_3 = \lambda(x_1 - x_3) - y_1 = 3(0-4) - 1 = 2 \mod 5$

result: $(4,2)$

$(4,2)+(0,1)$

step1: compute λ

$\lambda = \frac{y_2 - y_1}{x_2 - x_1} = 1 \cdot 4^{-1} = 4 \mod 5$

step2: compute x_3

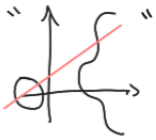
$x_3 = \lambda^2 - x_1 - x_1 = 2 \mod 5$

step3: compute y_3

$y_3 = \lambda(x_1 - x_3) - y_1 = 4(4-2) - 2 = 1 \mod 5$

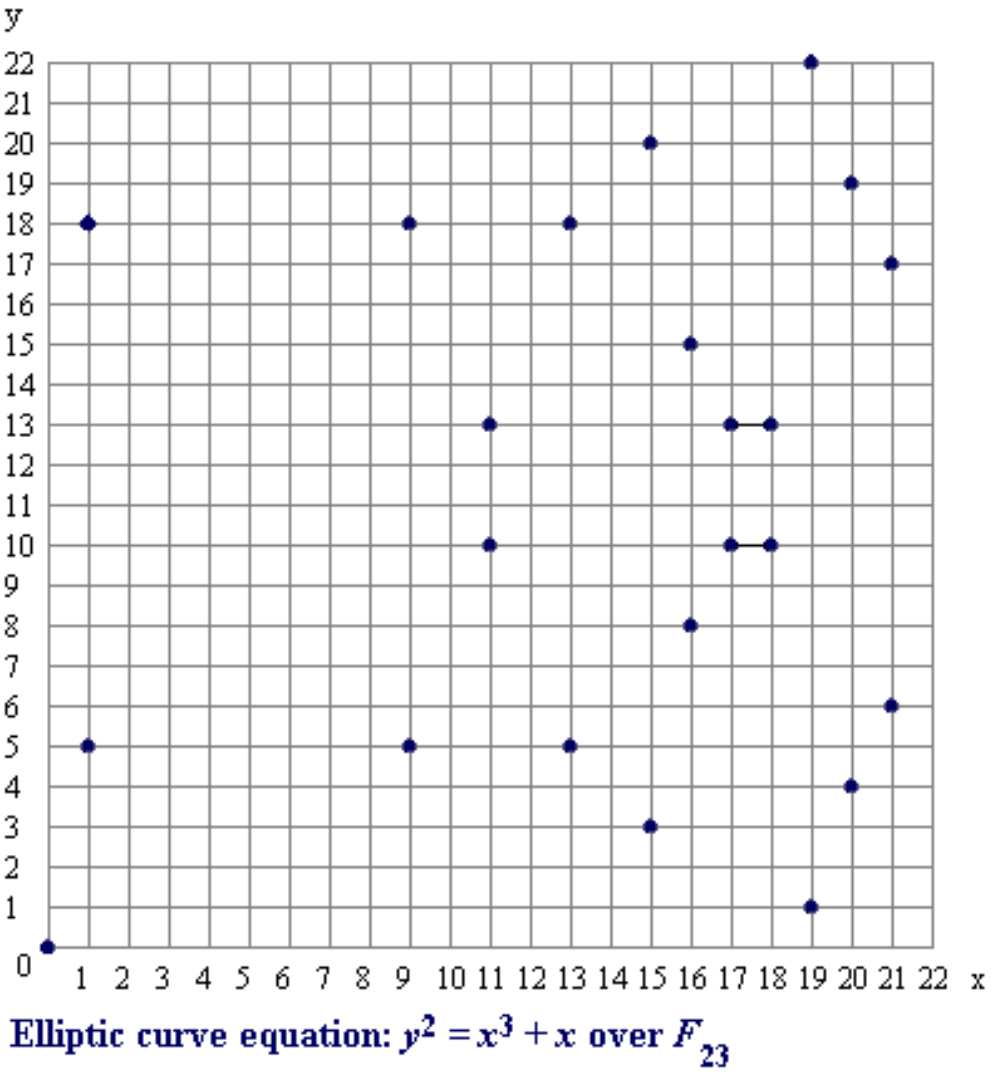
result: $(2,1)$

Which points in “larger” group?

Da Weierstrass  , poi ho ristretto il campo e sono passato a int mod p

No geometric “look&feel” anymore…
no problem!

sembrano "RANDOM", c'è solo simmetria su punti aventi stessa "x".
è molto ristretto rispetto y^x mod p



Elliptic curve equation: $y^2 = x^3 + x$ over F_{23}

EC group

Weierstrass curve — interi mod p — "+" oper.

➔ **$(E(Z_p),+)$ is a group**

1. Addition **closed** on set E
2. Addition is **commutative** (abeliano)
3. O is **identity** wrt addition
4. Every point P in E has inverse –P wrt addition
5. Associative property holds

Harder to prove in generality, but holds

Abelian group

EC Group properties for crypto

quante volte ripeto l'op. — P on T, e' elliptic curve!!

➔ **Easy/hard operation: [k]P** = P+P+P+P

- ⇨ Given P and k, easy to compute [k]P (complex: LOG con 'k')
- ⇨ Given P and [k]P, hard to compute k (complex: exponential!)

] DLOG PROBLEM

  - →P = elliptic curve point on group
  - →k = integer mod group order
- ⇨ "Equivalent" to Dlog problem on Zp
  - →If we had used multiplicative notation: k → P^k

kP mod p errore!
il mod si applica per trovare i punti, ma quando sommo (es (0,1)+(2,4)) NO MODULO!!

➔ **CRUCIAL: DEPEND ON THE CHOSEN CURVE!!!**

- ⇨ Use curves recommended by standards!!!! Parametri non corretti cambiano la complessità
- ⇨ Curves not only in Zp, but also in GF(2^m) – Binary curves: convenient!

➔ **Different EC curves satisfy different assumptions!**

- ⇨ We'll see (and exploit!) later on: e.g. CDH hard but DDH easy

Multiplication (exponentiation)

- **Group order 160 bits → $q=2^{160}$**
- **Major cost: multiplication**
  - ⇨analogous to exp
  - ⇨Uses very large factors k (160 bits)
- **Efficiency: approx 1.5 Log_2 q**
  - ⇨i.e. linear with the number of bits
    - →Basic algorithm: double and sum
      - » (analogous to square & multiply)

NB: NON applico modulo ai punti, solo ai valori!

Example

- **Goal: compute [1437]P**
- **Express in bits**
  - $1437_{10}=10110011101_2$
- **Initialize first line**
  - Differs if 0 or 1
- **Start from 2° lsb to msb**
  - For every bit
    - Double;
    - If 1: add to result
- **Complexity:**
  - O(nbit) x2
  - O(nbit/2) additions

$lsb->msb$	*double*	*result*
1	$1P$	$+1P=P$
0	$2P$	.
1	$4P$	$+4P=5P$
1	$8P$	$+8P=13P$
1	$16P$	$+16P=29P$
0	$32P$	.
0	$64P$	.
1	$128P$	$+128P=157P$
1	$256P$	$+256P=413P$
0	$512P$	.
1	$1024P$	$+1024P=1437P$

No algorithm such as this for the opposite problem!! That's why it is hard!

11/01/2022

Elliptic curve crypto
(examples)

skip

EC crypto

- **Invented in 1985**
  - Neal Koblitz
  - Victor Miller
  - (independently)
- **Security: based on hardness of:**
  - Given P and [k]P, hard to compute k
  - Elliptic Curve Discrete Log Problem – ECDLP
- **Much shorter keys than RSA/DLP**
  - And better scaling!

Symm key equiv	modulus size	Elliptic Curve
80 bits	1024 bits	163 bits
128 bits	3072 bits	283 bits
256 bits	**15360** bits	571 bits

ECDH/ECDSA

- **Topmost used, since included in US National Security Agency – suite B ciphers**
- **Suite B: NIST-approved, promoted in 2005, adopted in TLS, IPsec, S/MIME, etc**
- **Algorithms:**
  - Encryption: AES 128/256
  - Key Exchange: ECDH
    - 256 and 384 bit prime curves
  - Digital Signature: Elliptic Curve Digital Signature Algorithm (ECDSA)
    - 256 and 384-bit prime moduli
  - Hashing: SHA-256 and SHA-384

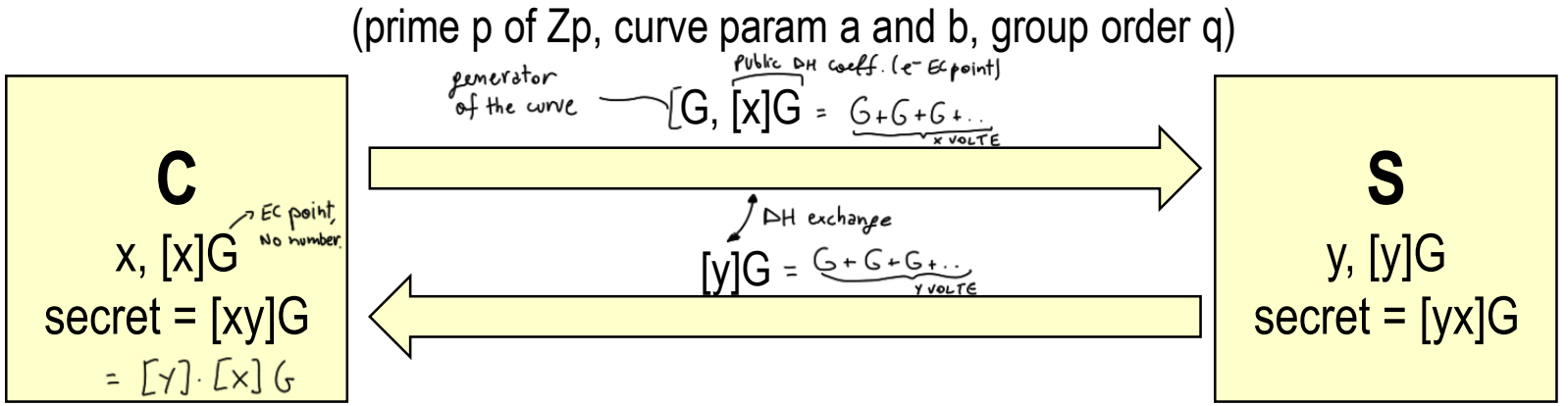
In DH $g^x \bmod p$ ≡ faccio x volte prodotto punto g con se stesso in $Z^*(p)$,

tale gruppo è stato sostituito dai gruppi EC

ECDH

➔Exactly as DH

⇨Just use as group an EC group!



C ha x, e riceve [y]G, allora computa [xy]G= G + G + G … xy volte.
Infatti [x] * [y]G = [y]G + [y]G + [y]G …. x volte

Esempio:
x = 3, y = 2 -> [xy] G = 6G = G + G + G +G +G +G
[3]* [2]G = ([2]G + [2]G +[2]G)* [3] = [2]*[3] G = [6]G

ECDSA

- **Elliptic Curve Digital Signature Algorithm**
  - EC version of DSA
    - Invented by Vanstone
    - Proposed by NIST in august 1991
  - adopted by NSA Suite B
- **Variant of ElGamal**
  - Dlog based → $P_K = g^s \bmod p$, $S_K = s$
- **Famous also for other reasons ☺**
  - Broken in Playstation 3, 2010
    - Not because ECDSA is vulnerable…
    - Because of incorrect usage, see later

Let's first remember DSA (Senza EC)

(simplified)

mod. arith.

➔Multiplicative group

⇨Z*(p), p>2 prime

⇨Large prime q in group order

➔Usually p = 2q+1 (strong prime)

➔Generator

⇨g s.t. g^x mod p spans all group elements

➔Private/Public keys

⇨d, $y = g^d \bmod p$

Let's first remember DSA
(simplified)

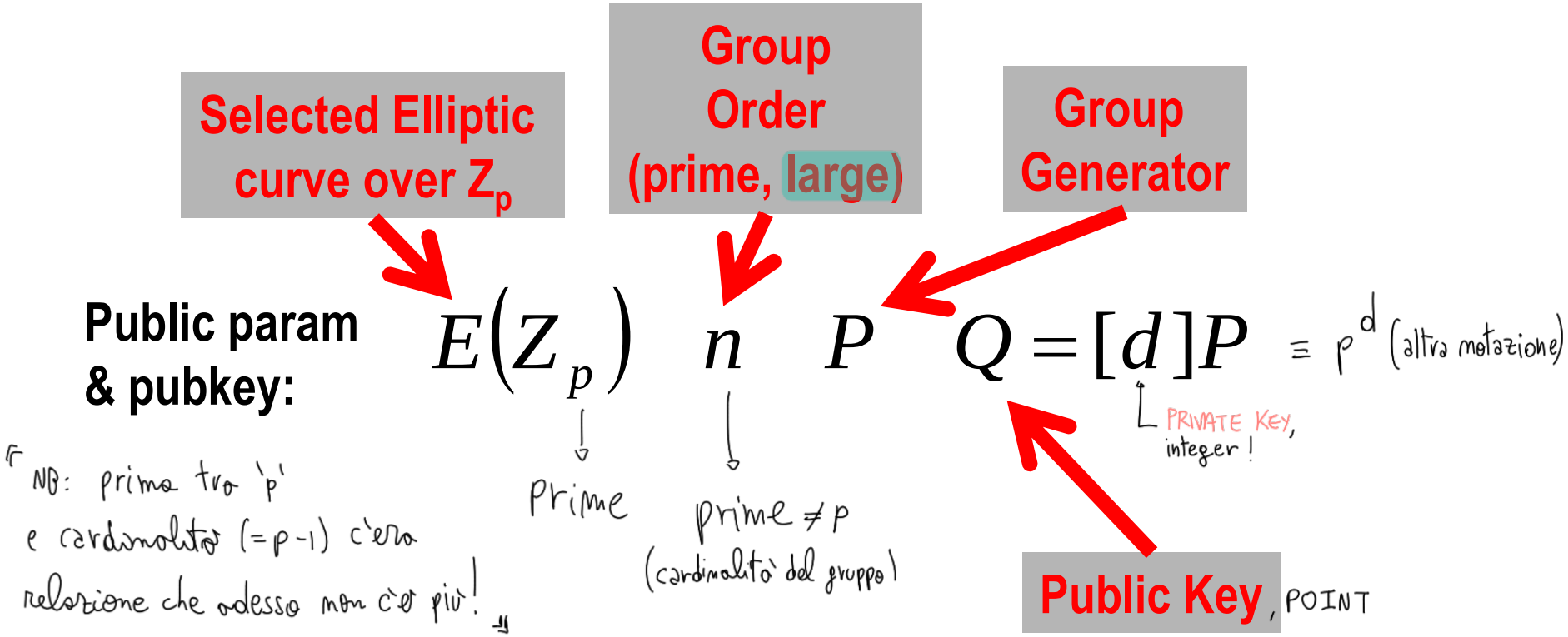
- **Signature for message m** (∀ msg ho NONCE)
  - Random k in (1,q) (nonce), poiché q prime, e k<q ⇝ ∃ inverso!
  - Compute r = $(g^k \bmod p) \bmod q$, r = f(k)
  - Compute s = k^{-1} (H(m)+ d r) mod q ⇝ $f(H(m), d, K^{-1})$ (in RSA nonc'è!)
  - Signature: (r,s) (in RSA avevo $H(m)^d$, cioè f(H(m), d)) → PUBLIC VERSION of the nonce, NON è nonce in clear!)
- **Verification**
  - Compute $u_1 = s^{-1} H(m) \bmod q$ → $\frac{K}{H(m)+rd}$ (H(m) manca "d" che non posso sapere!)
  - Compute $u_2 = s^{-1} r \bmod q$ → $\frac{K}{H(m)+rd} \cdot r$
  - Verify $((g^{u1} y^{u2}) \bmod p) \bmod q = r$ (lavoro all'esponente dove ho g^d, g^K, che non avrei altrimenti!)

public key $g^d \bmod p$

$$g^{u_1} y^{u_2} = g^{u_1} g^{d \cdot u_2}$$

$$u_1 + d \cdot u_2 = \frac{H(m)k}{H(m)+rd} + \frac{d \cdot r \cdot k}{H(m)+rd} = k$$

ECDSA: setup



➔Private key:

- ⇨d = random integer in [1,n-1]
- ⇨Q = dP

ECDSA: signature generation

- **k in [1,n-1]: random integer**
  - ESSENTIAL: MUST BE
    - unique per each signature
    - unpredictable

$r = g^k \approx$ → gen. EC group, è la parte più computazionale

- **Compute kP=(x_1,y_1) EC point**
  - x_1 is integer, then

coordinate, range $[0, p-1]$ interi mod P → group generator

- **Compute r = x_1 mod n**

INT, NO POINT — posso NON considerarlo

(se $P=11$, $n=9$, $x_1=10$, devo riadattarlo, ma NON capita quasi MAI!)

- **Compute k^{-1} mod n** (k numero < n prime → inverse)

- **Hash function H(.) e.g. SHA**

Signature:

(r,s)

$$s = k^{-1}(H(m) + d\,r)$$

tutti numeri!

ORA è MOD n, order group

NON INVIO PUNTI, MA NUMERI!

ECDSA: verification

- **From message m**
  - compute H(m)
- **From s**
  - compute $w=s^{-1} \bmod n$
- **Compute integers:**
  - $u_1 = H[m]\, w \bmod n$
  - $u_2 = r\, w$
- **Compute elliptic point**
  - $u_1 P + u_2 Q = (x_0, y_0) = (u_1 + u_2 d) P$
- **Compute**
  - $v = x_0 \bmod n$

Verification:

$x_0 \bmod n = v \overset{?}{=} r = x_1 \bmod n$

$$w = s^{-1} = k\left[H(m) + rd\right]^{-1}$$

$$u_1 P + u_2 Q =$$

$$= \frac{H(m)k}{H(m)+rd} P + \frac{r \cdot k}{H(m)+rd} Q =$$

$$= \frac{H(m)k}{H(m)+rd} P + \frac{rd \cdot k}{H(m)+rd} P = kP$$

Note: k never disclosed!

What if k predicted?? GAME OVER

➔Private key would be disclosed!

↳ qualcuno può fare sign col mio nome

from (r, s) and knowledge of k:

$$s = k^{-1}\left[H(m) + rd\right] \mod n$$

$$sk = H(m) + rd \mod n$$

$$\left[sk - H(m)\right]r^{-1} = d \mod n$$

d readily computed!!

⇒ K must be UNPREDICTABLE!!

What if k repeated??

Potrei "riusarla" per motivi energetici e di efficienza

➔Private key would be disclosed!

from $(r, s_1), (r, s_2)$ - same unknown k = same r

$$\begin{cases} s_1 = k^{-1}\left[H(m_1) + rd\right] \\ s_2 = k^{-1}\left[H(m_2) + rd\right] \end{cases} \mod n \quad \Rightarrow \quad \begin{cases} k = s_1^{-1}\left[H(m_1) + rd\right] \\ k = s_2^{-1}\left[H(m_2) + rd\right] \end{cases} \mod n$$

$$s_1^{-1}\left[H(m_1) + rd\right] = s_2^{-1}\left[H(m_2) + rd\right]$$

$$s_2 H(m_1) + s_2 rd = s_1 H(m_2) + s_1 rd$$

$$d(s_2 - s_1)r = s_1 H(m_2) - s_2 H(m_1)$$

$$d = \left[s_1 H(m_2) - s_2 H(m_1)\right] r^{-1}(s_2 - s_1)^{-1} \mod n$$

d readily computed!!

This is what happened with Sony's Playstation 3