ECC

Elliptic curve cryptography

Algorithm Family	CryptoSystem	SecurityLevel:80	128	192	256
Integer Factorization	RSA	1024 bits	3072	7680	15360
Discrete Logarithm	DH,DSA,Elagaml...	1024 bits	3072	7680	15360
Elliptic Curve	ECDH,ECDSA	160 bits	256	384	512

As we can see, EC need fewer bits to achieve the same security level, which is more computational efficient

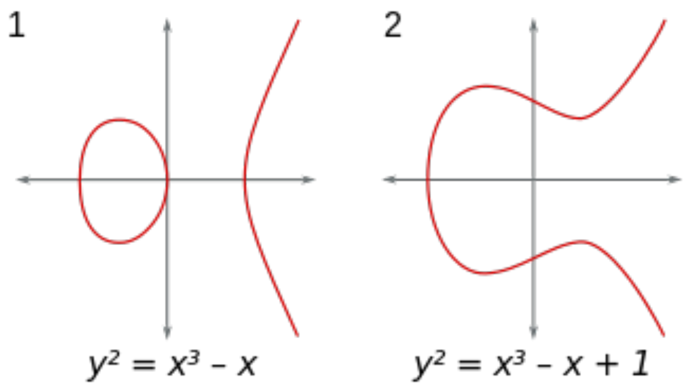
Motivation of ECC:Find PK family with shorter operands

Idea: Can we find another cyclic group with hard log problem

Elliptic Curve

> ### Definition:
>
> The Elliptic Curve over $Z_p, p > 3$, is the set of all pairs $(x, y) \in Z_p$:
>
> $$y^2 \equiv x^3 + ax + b \mod p$$
>
> together with an imaginary point at infinity Θ,
>
> Where $a, b \in Z_p$ and $4a^3 + 27b^2 \not\equiv 0 \mod p$

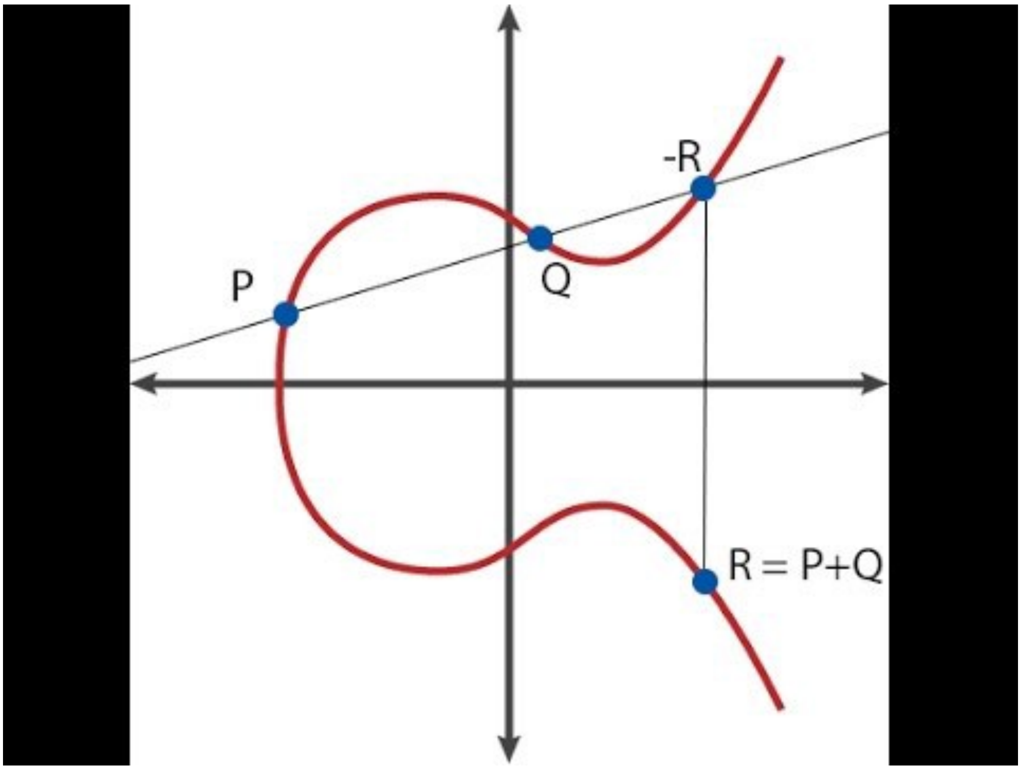
It may be hard to understand it from algebra perspective, but geometry can give a perfect interpretation



Step 1 : we want to find a cyclic group, we need the group elements and group operation.

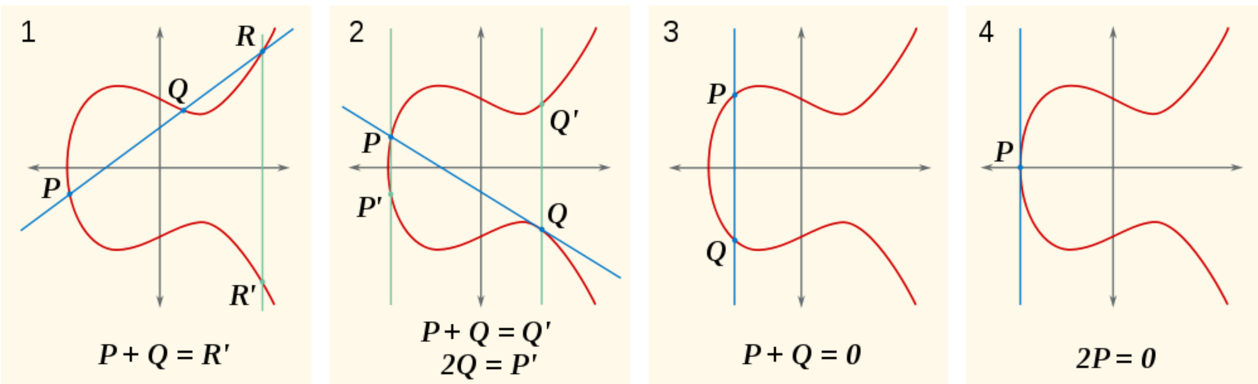
- Group elements : set of points on the curve
- Group operation(between two points A, B, denoted by $+$)

- connect the two points, then we will get a third point on the curve
- mirror the third point against x axis



- what if $A = B$? What is $Q + Q = 2Q$?

  we make the tangent of point P, and the rest of the operation is the same($2Q = P'$)



Step 2 : We need to do calculate instead of drawing graphs, so how to express the group operation in formula?

Given:

$$\begin{gathered} E : y^2 = x^3 + ax + b \\ P = (x_1, y_1) \\ Q = (x_2, y_2) \end{gathered}$$

Process:

$$\text{find } l_{PQ} = sx + m$$

$$\begin{gathered} \text{find } x_R : (l_{PQ}^2) = x^3 + ax + b \\ (sx+m)^2 = x^3 + ax + b \text{ (we already know } x_P, x_Q) \end{gathered}$$

Result:

$$\begin{cases} x_R = s^2 - x_P - x_Q \mod p \\ y_R = s(x_P - x_R) - y_P \mod p \end{cases}$$

Where

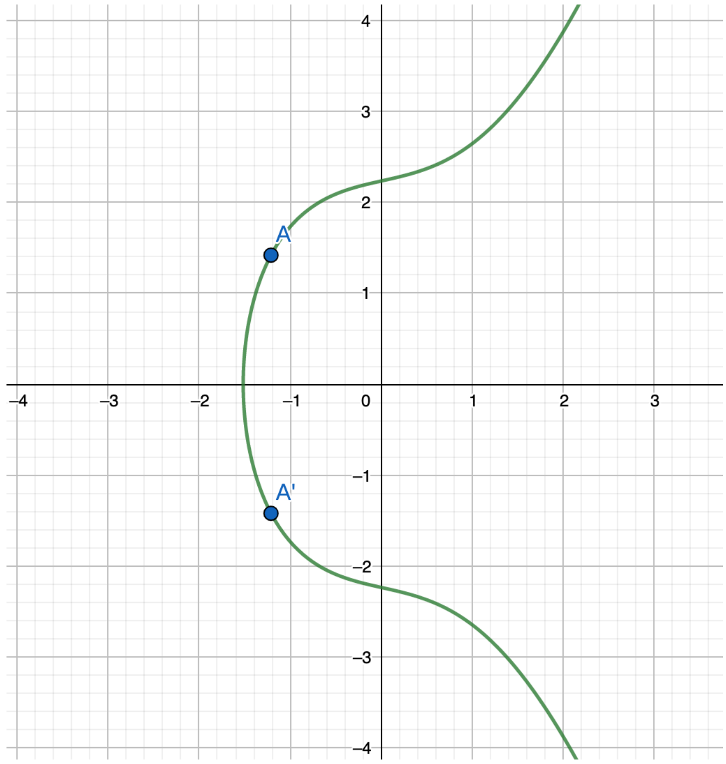
$$\begin{cases} \frac{y_Q - y_P}{x_Q - x_P} \mod p, & \text{if } P \neq Q \\ \frac{3(x_P)^2 + a}{2y_P} \mod p, & \text{if P} = \text{Q} \end{cases}$$

- the second slope $\frac{3(x_P)^2+a}{2y_P}$ is the derivative of $y_P = \pm\sqrt{x_P^3 + ax_P + b}$
- $\frac{y_Q - y_P}{x_Q - x_P}$ is actually $(y_Q - y_P)(x_Q - x_P)^{-1} \mod p$(same with the second equation)

Step 3 : check the group laws

- the neutral element : Θ
  - $$\forall P \in E, P + \Theta = P$$
  - For the reason that we cannot find an element Θ fulfil the requirement above, we artificially defined one, "an imaginary point at infinity".

    when we add a point with Θ, the straight line will go perpendicular with x axis through the point
- The inverse element
  - $$P^{-1} + P = \Theta$$
  -



- $$P = (x, y)$$
  $$P^{-1} = (x, -y)$$

Step 4 : is the group cyclic ?

> ### Theorem:
>
> The points on EC, including Θ, have cyclic subgroups.
>
> Under certain conditions all points on a EC form a cyclic group.

Example

$$E : y^2 \equiv x^3 + 2x + 2 \mod 17$$

$$\text{generator} : P = (5, 1)$$

$$\therefore 2P = P + P = (6, 3)$$

$$\therefore 3P = 2P + P = (10, 6)$$

$$\cdots$$

$$18P = (5, 16) \equiv (5, -1) \mod 17 = P^{-1}$$

$$\therefore 19P = 18P + P \equiv P + P^{-1} \equiv \Theta \mod 17$$

Elliptic curve discrete logarithm problem

> **Definition**
>
> Given an elliptic curve E. We consider a primitive element P and another element T. The discrete logarithm problem is finding the integer d, where $1 \leq d \leq |E|$, such that
>
> $$\underbrace{P + P + \ldots + P}_{\text{d times}} = dP \equiv T$$

Note that $d = K_{\text{Pr}}, T = (x, y) = K_{\text{Pub}}$

> ### Hasse's Theorem
>
> Given an elliptic curve E modulo p, the number of points on the curve is denoted by |E| and is bounded by
>
> $$p+1-2\sqrt{p} \leq |E| \leq p+1+2\sqrt{p}$$
>
> This tell us, in the long run, $|E| = O(p)$.
>
> However, to perform a real attack we need to know exactly what p is, this is computational hard.

If the EC is chosen carefully, the best known attack algorithm need $O(\sqrt{p})$

Elliptic Curve Diffie-Hellman Keyexchange

straight forward adoption of DH on Z_p^*

Phase 1 : set up the parameters

$$E : y^2 \equiv x^3 + ax^2 + b \mod p$$

$$\text{generator} : P = (x, y)$$

Phase 2 : the protocol

E,P

Alice

K_prA = select([2,3,..,|E|-1])

K_pubA="pow"(P,K_prA)=(xa,ya)

Bob

K_prB = select([2,3,..,|E|-1])

K_pubB="pow"(P,K_prB)=(xb,yb)

$\xleftarrow{\text{K\_pubB}}$

$\xrightarrow[\text{K\_pubA}]{}$

K_shared="pow"(K_pubB,K_prA)=(xab,yab)

K_shared="pow"(K_pubA,K_prB)=(xab,yab)

"Fast exponential" in EC

Aka "doubling and addition"

How to compute $\underbrace{P + P + .. + P}_{\text{a times}}$ quickly?

Example: P^{26}

```
class ECPoint():
    def __init__(self,x,y) -> None:
        self.x , self.y = x,y

    def __add__(self , other):
        pass

def add(P,Q):
```

```
    pass

def double(P):
    pass

res = ECPoint()
P = ECPoint()
for i in bin(26)[3:]:
    res = double(res)
    if i == '1':
        add(res , P)
```

$$\text{binary}(26) = 0b11010$$

Step	
$\emptyset$	$P = 1_2 P$
1a	$P + P = 2P = 10_2 \cdot P$
1b	$2P + P = 3P = \underline{11}_2 P$
2a	$3P + 3P = 6P = \underline{110}_2 \cdot P$
3a	$6P + 6P = 12P = 1100_2 \cdot P$
3b	$12P + P = 13P = \underline{1101}_2 P$
4a	$13P + 13P = 26P = \underline{11010}_2 P$