Courbe elliptique

Réalisé par : Aicha BOUICHOU

Plan:

I. Introduction

En cryptographie, les courbes elliptiques, des objets mathématiques, peuvent être utilisées pour des opérations asymétriques comme des échanges de clés sur un canal non sécurisé ou un chiffrement asymétrique, on parle alors de cryptographie sur les courbes elliptiques ou ECC .

II. Courbe elliptique

1. Définition

Une courbe elliptique est un cas particulier de courbe algébrique, munie entre autres propriétés d'une addition géométrique sur ses points .

L'équation d'une courbe elliptique définie sur le corps des nombres réels peut être mise sous la forme plus simple (dite équation de Weierstrass)

$$y^2 = x^3 + ax + b$$

Avec delta non nul :

$$\Delta = -16(4a^3 + 27b^2),$$

2. Addition

L'addition de deux points sur une courbe elliptique *E* est rendue possible par la propriété suivante, qui est un cas particulier du théorème de Bézout sur l'intersection de courbes algébriques sur les réels : une droite sécante passant par deux points de la courbe recoupe la courbe en un troisième point (distinct ou non).
Elle est définie comme suite :

Soient P(x , y) et Q(x' , y') appartenant a la Courbe , et R(X , Y) = P+Q.

- Si x=x' : « Cas d' addition des points verticaux »
  R est un point à l'infinie ,
- Si P !=Q :
  S= (y-y')/(x-x') alors X= s²-(x + x') et Y= s*(x - x') - y
- Si P=Q :
  R= 2*P
  S=(3 x^3+a)/(2*y) alors X = s² -2*x et Y=s*(x-X)-y

3. Multiplication par un scalaire

La multiplication d'un point d'une courbe par un scalaire k , c'est faire la somme de ce point k-fois , ce qui va nous amener à procéder on se basant sur l'addition a cette algorithme :

- Faire la somme en deux ; on va avoir un R(X , Y).
- Faire la somme de R avec le P ; pour avoir un nouveau R ;
- Refaire l'étape précédente tant que nous avons pas atteint l'infinie .
- Sortir de cette boucle et récupérer le n , qui est l'ordre de générateur de l'infinie .

4. Les paramètres

Les paramètres que nous avons dans ce protocole sont :
-p : qui est l'enivrement de calcule .
-a et b : paramètres données par l'équation de werstarass « la courbe »
-G : c'est un point de la courbe qui est le générateur de l'infinie
-n : l'ordre qui représente algébriquement le nombre d'addition faite sur G pour atteindre l'infinie
-h : facteur abs(E(Z/Pz))/n ; si il est égal a 1 c'est le cas idéal

5. Diffie Hellman

L'échange de clef Diffie-Hellman basé sur les courbes elliptiques (de l'anglais *Elliptic curve Diffie–Hellman*, abrégé ECDH) est un protocole d'échange de clés anonyme qui permet à deux pairs, chacun ayant un couple de clé privée/publique basé sur les courbes elliptiques, d'établir un secret partagé à travers un canal de communication non sécurisé. Ce secret partagé peut être employé directement comme une clef de chiffrement ou être utilisé pour dériver une autre clef qui, à son tour, peut être utilisée pour chiffrer les communications.

Il s'agit d'une variante du protocole d'échange de clefs Diffie-Hellman utilisant les courbes elliptiques.

Le process et le suivant :

-Bob choisit une clef privée beta tel que : 0<beta<n

-Il calcule : B= beta *G

- Il envoi B (x', y')a Alice

De l'autre part :

-Alice choisit une clef aussi privé alfa tel que : 0<alfa<n

-Elle calcule : A= alfa *G

- Elle envoi A(x , y) a Bob

Les deux auront le même en multipliant , A ou B avec leurs clefs privées :

-Bob calcule P= beta *A= beta* alfa *G

- Alice calcule le même P avec sa clef privée : P= alfa* B= alfa*beta * G

6. Problème de logarithme discret

Le problème de Diffie-Hellman est énoncé informellement comme tel:

Soit un élément g et les valeurs g^x et g^y, quel est la valeur de g^{xy}?

Formellement, g est un générateur d'un groupe et x et y sont des nombres entiers choisis aléatoirement.

Par exemple, dans l'échange de clef Diffie-Hellman, un observateur indiscret observe g^x et g^y échangés, appartenant au protocole, et les deux tiers calculent la clé partagée g^{xy}. Un méthode rapide pour résoudre le problème de Diffie-Hellman serait un moyen pour l'observateur indiscret de violer le secret de l'échange de la clef de Diffie-Hellman ainsi que ses variantes, incluant le Cryptosystème de ElGamal.

III. Implémentation Python

1. Vérification de courbe

```
def Verifier_courbe(a,b):
        v= 4*pow(a,3)+27 * pow(b,2)
        if v == 0 :
              print ("la courbe  est  mal definie ")
              a=int(input("la valeur de a"))
              b=int(input("la valeur de b"))
              Verifier_courbe(a,b)
        else:
               print ("la courbe  est  bien definie ")
```

2. Addition

- ***P+Q***

```
def addpointdis(x,y,c,d) :

        s= (y-d)/(x-c)

        xres= pow(s,2)-(x+c)

        yres= s *(x-c)-y

        print (" Le point résultant R a comme coordonnées  : ", xres,yres)

        return s,xres,yres
```

- ***P+P***

```
def addition_2_fois(x,y,a) :

                s=(3*pow(x,2)+a)/(2*y)
                xres=pow(s ,2)-2*x
                yres= s*(x-xres)-y
                print (" Le point résultant R a comme coordonnées  : ", xres,yres)
                return s,xres,yres
```

- ***Point vertical :***

c'est le test qu'en doit effectuer avant toute addition :

```
def verifie_Verc(x,y,c,d) :

    if x==c & d==y:
        print (" le double est ")
        addition_2_fois(x,y,a)
    elif x==c & d!=y:
        print (" l'infinie est le resultat")
    elif x!=c & y!=d:
        print (" l'addition de deux point #")
        addpointdis(x,y,c,d)
```

3. Multiplication par scalaire

Un scalaire paire :

```
def multiplierparpaire(xp,yp,nbr,a):
    while nbr/2!=1:
```

```
        S,X,Y= addition_2_fois(xp,yp,a)
        nbr=nbr/2
        xp=X
        yp=Y
    addition_2_fois(xp,yp,a)
```

Un scalaire impaire :

```
def multiplier(xp,yp,nbr,a):
    par,exe,ey=multiplierparpaire(xp,yp,nbr-1,a)
    addpointdis(xp,yp,exe,ey)
```

IV. Résumé

```
# -*- coding: utf-8 -*-
"""
Created on Fri Jan  5 14:19:59 2018

@author: aicha
"""

from math import sqrt



def Verifier_courbe(a,b):
        v= 4*pow(a,3)+27 * pow(b,2)
        if v == 0 :
            print ("la courbe  est  mal definie ")
            a=int(input("la valeur de a"))
            b=int(input("la valeur de b"))
            Verifier_courbe(a,b)
        else:
            print ("la courbe  est  bien definie ")
def addpointdis(x,y,c,d) :
        s= (y-d)/(x-c)
        xres= pow(s,2)-(x+c)
        yres= s *(x-c)-y
        print (" Le point résultant R a comme coordonnées  : ", xres,yres)
        return s,xres,yres

def addition_2_fois(x,y,a) :
```

```
            s=(3*pow(x,2)+a)/(2*y)
            xres=pow(s ,2)-2*x
            yres= s*(x-xres)-y
            print (" Le point résultant R a comme coordonnées  : ", xres,yres)
            return s,xres,yres

def verifie_Verc(x,y,c,d) :

        if x==c & d==y:
            print (" le double est ")
            addition_2_fois(x,y,a)
        elif x==c & d!=y:
            print (" l'infinie est le resultat")
        elif x!=c & y!=d:
            print (" l'addition de deux point #")
            addpointdis(x,y,c,d)
def multiplierparpaire(xp,yp,nbr,a):
      while nbr/2!=1:
        S,X,Y= addition_2_fois(xp,yp,a)
        nbr=nbr/2
        xp=X
        yp=Y
      addition_2_fois(xp,yp,a)

def multiplier(xp,yp,nbr,a):
      par,exe,ey=multiplierparpaire(xp,yp,nbr-1,a)
      addpointdis(xp,yp,exe,ey)




if __name__ == '__main__':


     print(" veuillez etablir l'equation de weirstrass ( y²=x^3+ a*x+b) est suivre le
   traitement :")
     a=int(input("la valeur de a"))
     b=int(input("la valeur de b"))

     Verifier_courbe(a,b)

     choix=input("veuillez taper la nature de traitment que vou voulez faire :A/M")
     if choix=='A':
         print(" traitemnt d'addition :")
         xp=int(input("la valeur de x de premier point"))
         yp=int(sqrt(pow(xp,3)+a*xp+b))
         print("la valeur de yp est  :",yp)
```

```
        xq=int(input("la valeur de x de deuxieme point"))
        yq=int(sqrt(pow(xq,3)+a*xq+b))
        print("la valeur de yq est  :",yq)
        verifie_Verc(xp,yp,xq,yq)
    elif choix =='M':
        print(" traitmnt de multiplication par un scalaire  :")
        nbr=int(input("tapez le nombre scalaire "))
        xp=int(input("la valeur de x du point"))
        yp=int(sqrt(pow(xp,3)+a*xp+b))
        print("la valeur de yp est  :",yp)
        if nbr%2==0:
            multiplierparpaire(xp,yp,nbr,a)
        else :
            multiplier(xp,yp,nbr,a)
```