椭圆曲线加密算法

射影平面

假设两条平行线相交于一个无限远点，通过这个无线远点，就可以了解到所有的直线是可以相交的。

两条或者更多条直线的共点，意思是以下两种情况之一：或者存在一个点，所有的直线都通过它，或者它们两两平行。

以下是无穷远点的几个性质。

▲直线L上的无穷远点只能有一个。（从定义可直接得出） ▲平面上一组相互平行的直线有公共的无穷远点。（从定义可直接得出） ▲平面上任何相交的两直线L1,L2有不同的无穷远点。（否则L1和L2有公共的无穷远点P，则L1和L2有两个交点A、P，故假设错误。） ▲平面上全体无穷远点构成一条无穷远直线。（自己想象一下这条直线吧） ▲平面上全体无穷远点与全体平常点构成射影平面。

在初等数学中，平面直角坐标系可以表示所有的 直线相交的Ê平常点ê，在高等数学中我们建设一个坐标系可以表示这个这个无穷远点，这个坐标系被称作为**射影平面坐标系。**

射影直角坐标系

我们对普通平面直角坐标系上的点A的坐标（x,y）做如下改造： 令x=X/Z，y=Y/Z（Zử0）；则A点可以表示为（X:Y:Z）。 变成了有三个参量的坐标点，这就对平面上的点建立了一个新的坐标体系。

例2.1：求点（1,2）在新的坐标体系下的坐标。

解：∵X/Z=1，Y/Z=2（Zử0）∴X=Z，Y=2Z ∴坐标为（Z:2Z:Z），Zử0。即（1:2:1） （2:4:2） （1.2:2.4:1.2）等形如（Z:2Z:Z），Zử0的坐标，都是（1,2）在新的坐标体系下的坐标。

我们也可以得到直线的方程aX+bY+cZ=0（想想为什么？提示：普通平面直角坐标系下直线一般方程是ax+by+c=0）。新的坐标体系能够表示无穷远点么？那要让我们先想想无穷远点在哪里。根据上一节的知识，我们知道无穷远点是两条平行直线的交点。那么，如何求两条直线的交点坐标？这是初中的知识，就是将两条直线对应的方程联立求解。平行直线的方程是：aX+bY+c1Z =0； aX+bY+c2Z =0 (c1ửc2)；（为什么？提示：可以从斜率考虑，因为平行线斜率相同）；

将二方程联立，求解。有c2Z= c1Z= -（aX+bY），∵c1ửc2 ∴Z=0 ∴aX+bY=0； 所以无穷远点就是这种形式（X：Y：0）表示。注意，平常点Zử0，无穷远点Z=0，因此无穷远直线对应的方程是Z=0。

例2.2：求平行线L1：X+2Y+3Z=0 与L2：X+2Y+Z=0 相交的无穷远点。 **解：**因为L1∥L2 所以有Z=0，X+2Y=0；所以坐标为（-2Y:Y:0），Yử0。即（-2:1:0） （-4:2:0） （-2.4:1.2:0）等形如（-2Y:Y:0），Yử0的坐标，都表示这个无穷远点。

看来这个新的坐标体系能够表示射影平面上所有的点，我们就把这个能够表示射影平面上所有点的坐标体系叫做射影平面坐标系。

模运算

不错的在线教程

模运算的基本概念

模运算 $a \bmod b$的一般式 $r = a - b * int(a/b)$，换算成数学描述则是 **给定一个正整数p ，任意一个整数 n（正数,负数,0都行），一定一个整数k满足等式n = k * p+r （0≤r<n）**

模运算的几个性质

1. 同余式：正整数a，b对p取模，它们的余数相同，记做 a ≡ b % p或者a ≡ b (mod p)。
2. 若p|(a-b)，则a≡b (mod p) 备注： (|是整除性符号，也就是（a- b）是p的因子）

证明：令a-b=kp;那么a=kp+b1;所以a mod p = kp+b mod p =b mod ;所以a≡b (mod p)

3. 结合率（(x+y)mod p +c）mod p =（x+ (y +c mod) p）mod p ;

证明：存在x= k1p +r1,y = k2p+r2;c=k3p+r3；则代入等式左边，等于r1+r2+r3,等式右边也等于r1+r2+r3 。

4. 非常重要的运算：加法逆元和乘法逆元

定义：加法逆元 （x+y）mod p =0

乘法逆元（x*y）mod p =1

举例：模8加法和乘法逆元

w	-w	W-1
0	0	不存在
1	7	1
2	6	不存在
3	5	3
4	4	不存在
5	3	5
6	2	不存在
7	1	7

费马小定理

假如p是质数，a 是整数，那么 $a^(p-1)$ 是p的倍数，可以表示为$a^p \equiv a \bmod p$,如果a不是p的倍数，即不存在整数k使得 a = kp 成立，那么这个式子也可以写成 $a^{p-1} \equiv 1 \bmod p$

用两个小例子说明即可:

$$
\begin{aligned}
2^{100} &\equiv 2^{12\times 8+4} \pmod{13} \\
&\equiv (2^{12})^8 \cdot 2^4 \pmod{13} \\
&\equiv 1^8 \cdot 16 \pmod{13} \\
&\equiv 16 \pmod{13}
\end{aligned}
$$

$$\equiv 3 \pmod{13}$$

```
>>> math.pow(5,20) % 17
13.0
>>> math.pow(5,4) % 17
13.0
```

不同编程语言模运算的结果

在通常的编程运算中对于取模运算 a % b ，当 a，b符号相同时，它们的结果通常是一样的。但是当它们的符号相异时结果会不相同。

首先我们来看 a %b 的一般运算方式 $r = a - b * int(a/b)$

当 ab 符号相异时， a%b结果会不同，对于java ， golang ，js， 他们的结果都是取余运算的结果，而python的结果是取模运算的结果。其实按数学解释两者都是对的。

例如， -10 % 23,python的结果是 13 ，而js，golang，java的结果就是10，原因就在于 $r = a - b * int(a/b)$ 这式子里$int(a/b)$ 的计算的。 例如 -10 / 23 的浮点型结果为 `-0.43478260869565216` 。python对其的求整会向其负无穷小方向取证，最后的结果就是 -1 ， java,golang ， js等语言会向其0的方向取整，所以最后结果为0。这就是为什么不同语言中 a % b 模运算不同的原因!

群，环，域，向量空间

> 先来总结的一张图

群

群的概念可以理解为：一个集合以及定义在这个集合上的二目运算，满足群的四条公理，**封闭性，结合性，单位元，反元素。**

1. 封闭性： 假定集合G已经经过完美的拓扑，那么在集合上面进行二目运算，不会诞生超出集合范畴的新元素。**满足这一点的即可称之为原群，整数和 +，-，*中任一预算符都满足以上性质**
2. 结合性： 组合一个二元操作链，之间没有先后运算的区别，这种操作是平坦的。 **满足这一点的称之为可交换群，也称之为半群**
3. 单位元： 具有单位的属性，单位元和任何一个元素操作等于那个元素的本身。如果一个半群拥有单位元，则称之为幺半群，例如整数集合和二目运算
4. 反元素（逆元）： 集合中任何一个元素，存在一个被称为反元素的元素与那个元素进行操作后，最后的结果为单位元。
5. 交换律： 例如$a+b=b+a$，如果满足交换律一个群，则称之为阿贝尔群。

环

环在阿贝尔群的基础上，添加一个二目运算，添加后的集合**满足环公理**，则称之为环。 例如(R,+,*)组合

域

域(Field)在交换环的基础上，还增加了二元运算除法，要求元素(除零以外)可以作除法运算，即**每个非零的元素都要有乘法逆元。**由此可见，域是一种可以进行**加减乘除**(除0以外)的代数结构，是数域与四则运算的推广。整数集合，不存在乘法逆元(1/3不是整数)，所以整数集合不是域。有理数、实数、复数可以形成域，分别叫有理数域、实数域、复数域。域的几种定义，直接看维基百科英文吧：

向量空间

向量空间(vector space)是一些向量的集合。最熟悉的例子是几何向量或矢量(Euclidean vectors, geometric vector, spatial vector)，表示具有大小和方向的对象，如$\overrightarrow{AB}$，矢量可以做加法(addition)和乘法(scalar multiplication)运算，举例如下：

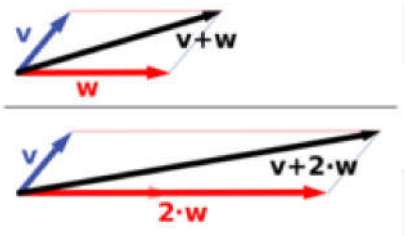


图5 Vector addition and scalar multiplication (source from here)

其他例子，还包括坐标空间(Coordinate spaces)、复数、函数空间(Function spaces)、线性方程组(linear equations)。详情可查阅维基百科词条：Examples of vector spaces.

给定域F，向量空间V记为F-向量空间。其二元运算：

- 向量加法：+ : V × V → V 记作 v + w, ∃ v, w ∈ V
- 标量乘法：·: F × V → V 记作 a v, ∃a ∈ F 且 v ∈ V

并且满足如下8条公理[10]：

- 向量加法结合律：u + (v + w) = (u + v) + w

- 向量加法的单位元：V存在零向量的0，∀ v ∈ V , v + 0 = v
- 向量加法的逆元素：∀v∈V, ∃w∈V，使得 v + w = 0
- 向量加法交换律：v + w = w + v
- 标量乘法与域乘法兼容性(compatibility): a(b v) = (ab)v
- 标量乘法有单位元: 1 v = v, 1指域F的乘法单位元
- 标量乘法对于向量加法满足分配律：a(v + w) = a v + a w
- 标量乘法对于域加法满足分配律: (a + b)v = a v + b v

另，若F是实数域$\mathbb{R}$，则V称为实数向量空间；若F是复数域$\mathbb{C}$，则V称为复数向量空间；若F是有限域，则V称为有限域向量空间。

椭圆曲线

椭圆曲线的一般方程 $y^2 + axy + by = x^3 + cx^2 + dx + e$,其他简单的形式为 $y^2 = x(x-1)(x-a)$

将一般方程的低阶去掉，可得到简化形式 $y^2 + axy = x^3$ 。也可以写为 $y^2 = P(x)$ P(x) 为任一没有重根的三次或者四次多项式。

椭圆曲线的是无奇点的，也就是说不存在一个点使得偏导数 $\frac{dF(x,y)}{dx}$,$\frac{dF(x,y)}{dy}$ 同时为0。也可以称之为在曲线的任何一点都是存在切线的

既然椭圆曲线上面没处都存在切线，那么可以求式（1） 的斜率,

解：令$F(x,y) = y^2 + axy + by = x^3 + cx^2 + dx + e$,求斜率k

$F(x,y) = y^2 + axy + by - x^3 - cx^2 - dx - e$

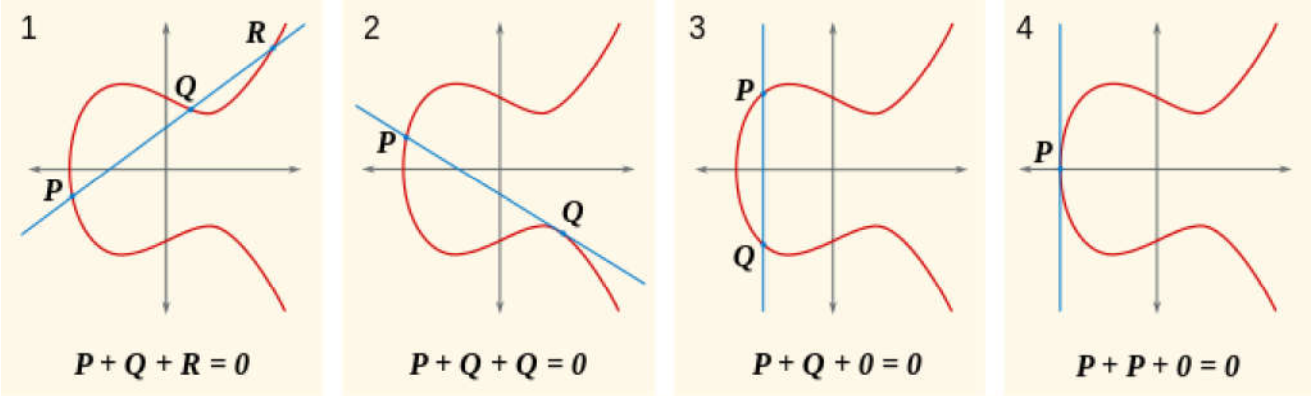
Fx(x,y)= $ay - 3x^2 - 2cx - d$

Fy(x,y)= $2y + ax + b$　　则导数为：f'(x)=$\frac{-Fx(x,y)}{Fy(x,y)}$= $\frac{3x^2-ay+2cx+d}{2y+ax+b}$

椭圆曲线上的运算

我们来以椭圆曲线为集合*E*，在上面定义 + 运算。 **不要混淆 此 + 和普通代数 +的概念，此处的 + 为抽象代数的+**

取*E*上的两点*P*,*Q*，若两者相异，*P* + *Q*表示穿过*P*和*Q*的弦和椭圆曲线相交的第三点，再经x轴反射的镜像点；若两者是同一点，*P*+*P*=2*P*表示以*P*为切点和椭圆曲线相交的点再经x轴反射的镜像点。若P和Q的弦与y轴平行（不存在和椭圆曲线相交的平常点），P+Q=0（无限远点）。+定义了一个*E*上的交换群，这个群以0为单位元。



通过以上定义可以多出，所有在椭圆曲线上集合E上面的有理数可以定义为一个（E，+）的阿贝尔群。

下面我们来求一下P,Q,R点

例子： 求 $F(x,y)=y^2+axy+by=x^3+cx^2+dx+e$ 上，求有平常点 $P(x_1,y_1),Q(x_2,y_2)$求 $R(x_3,y_3)$

我们先求R'（$-x_3,y_3$), 根据假设， P,Q,R三点共线，我们可以假设直线方程式为$y=kx+b$，将这个带入到曲线方程中得到 $(kx+b)^2+ax(kx+b)+b(kx+b)=x^3+cx^2+dx+c$

可以得到一般式 : R的横坐标为： $k^2+ka+c+x1+x2$ ，简单的带入纵坐标为 $y_3=y_1-k(x_1-x_3)$

点 R'和 R 横坐标显然一样 我们将其带入$F(x,y)$ 即可得到一二次方程：然后求其解可得到R点坐标(x_4,y_4)：

$x_4=k^2+ka+b+x_1+x_2$,$y_4=k(x_1-x_4)-y_1-ax_4-c$

密码学中的椭圆曲线

我们现在基本上对椭圆曲线有了初步的认识，这是值得高兴的。但请大家注意，前面学到的椭圆曲线是连续的，并不适合用于加密；所以，我们必须把椭圆曲线变成离散的点。

让我们想一想，为什么椭圆曲线为什么连续？是因为椭圆曲线上点的坐标，是实数的（也就是说前面讲到的椭圆曲线是定义在实数域上的），实数是连续的，导致了曲线的连续。因此，我们要把椭圆曲线定义在有限域上（顾名思义，有限域是一种只有由有限个元素组成的域）。

假设我们有一个椭圆曲线方程 E : $y^2=x^3+ax+b$ 我们在这个椭圆曲线上面假定一个有限域F_p (p 为 一个质数)，我们来定义其上面的四则运算：

- $a+b\equiv c(mod)p$
- $a*b\equiv c(mod)p$
- $a/b\equiv c(mod)p$ 即： $a*b^{-1}\equiv c(modp)$, b^{-1} 也满足存在与有限域内，实际在有限域内的点我们暂且称之为b_1 那么b_1 但是必须满足 $b*b_1\equiv 1(mod)p$ 。也就是说 $a*b^{-1}$ 和 $a*b_1$ 是一个同余式！ 模运算
- $a+b\equiv c(modp)$
- 单位元为1 ， 零元为0
- F_p 有限域满足结合律，交换律，分配律

将上面的定义结合到椭圆曲线中则有

$y^2=x^3+ax+b \bmod p$

结合椭圆曲线的性质

- 无限远点 + p = p
- 椭圆曲线上某点 p(x,y)的负元是 (x,y mod p) = (x,p-y) 则有 p + (-p) = 无限远点

另外，对于曲线方程： $y^2 = x^3 + ax + b \bmod p$ ，我们结合上一节的方式，已知P点(x_p, y_p) Q点(x_Q, y_Q) 求Q点坐标，当X

为$x_R = (k^2 - x_P - x_Q) \mod p$,$y_R = (y_P + m(x_R - x_P)) \bmod p$

当P,Q两点相交时，其斜率 $k = \frac{3x_P^2 + a}{2y_P} \bmod p$

Q = 2P

以上便是最重要的公式了

举个例子： 我们这里假设a= b = 1,则有限域$E_{23}(1,1)$上有两个点 P(3，10)，Q(9,7) ，求-P，2P。这里有个小H5

- $-P = (3, 10 \mod 23) = (3, 13)$
- P,Q形成的直线斜率为k，那么$k = (7-10)/(9-3) = -2^{-1} \bmod 23$ 。根据上面的定义 我们要找到一个有限域F_{23}范围内的数 x满足 $2 * 2^{-1} \equiv 2 * x \bmod 23$ 。显然这个数是12，所以 $2^{-1} = 12$ 将其带入到上面的$k = (7-10)/(9-3) = -2^{-1} \bmod 23 = -12 \bmod 23 = 11$ 。

对于2P的计算。可类比：这里也有个H5

- $$(3)\, k = \frac{3\times 3^2+1}{2\times 10} \bmod 23 = 7 \cdot 5^{-1} \bmod 23$$
  $$5 \cdot 5^{-1} = 1 \bmod 23 \Rightarrow 5^{-1} = 14$$
  $$k = 7 \cdot 14 \bmod 23 = 6$$
  $$2P = \left(6^2 - 3 - 3 \bmod 23, 6 \times (3-7) \bmod 23\right) = (7, 12)$$

ECC椭圆曲线加密算法

对于以上，如果椭圆曲线上一点P，存在最小的正整数n使得数乘nP=OÙ,则将n称为P的阶 若n不存在，则P是无限阶的。

现在我们考虑 K = kG 。K,G 皆为椭圆曲线上面的点，n以G为基点的阶，k <n。 我们会将 K 作为公钥公布出去，k作为我们的私钥。G作为一个基点其实相当与算法的一个参数。

对于椭圆曲线加密。算法的一般步奏

1. 随机产生一个32byte (256bit)的Big Integer作为私钥
2. 通过基准点G计算公钥kG。算法如下，**根据定义在椭圆曲线上的加法定理可知：kG = G+G+G……+G 一共k个G相加。我们举个例子**$151G = (2^7 + 2^4 + 2^2 + 2^1 + 2^0)G$ **所以我们可以很容易的通过随机产生私钥的256bit上面那些位是1就可以很容易的得到kG,其算法复杂度为O(256)，甚至在大部分实现就已经做好，因为基准G是提供好的、**
3. 那为什么公布了公钥kG，逆推私钥k就很难哪？因为需要一步步的去试！算法复杂度为$O(2^{256})$ 这个数字已经不能使用天文数字来形容了 2^256到底有多大 2^100次方有多大

椭圆加密算法的应用

综上所述，现在我们已经得到了私钥`k`和公钥`kG`。 现在我们要加密信息`m`，然后发送出去。

如果我们进行应用的话还需要解决的问题包括：

1. Base Point的选择
2. 信息的加解密

Base Point的选择

在了解选择Base Point之前，需要先了解循环子组的概念 。有一个椭圆曲线方程定义的有限域：
$y^2 = x^3 + 2x + 3 \bmod 97$ 在椭圆曲线上选择一点P(3,6)

1. 1P=(3,6)1
2. 2P=(80,10)
3. 3P=(80,87)
4. 4P=(3,91)
5. 5P=0
6. 6P=(3,6)
7. 7P=(80,10)
8. 8P=(80,87)
9. 9P=(3,91)

可以很容易的看出，P点的subgroup 长度为5，即5P = 0，另外，椭圆曲线上面所有的Point的 subgroup的长度均为椭圆曲线定义的有限域上点个数的倍数，例如有限域$F_9 7(2,3)$ 上面一共100个点，point(3,6) 的subgroup的长度为5。

对应的，我们可以知道选择P(3,6)肯定不合适。

所以我们先要选择一个长度足够大的Base Point。具体的方式是确定一个我们长度足够的subgroup，然后找这个Base Point。具体方式如下:

1. 计算有限域上面的的点数N，利用Schoof's algorithm
2. 得到一个素数n，这个素数必须可以被N整除
3. 得到 $h = N/n$
4. 随机选择一个点P，令G = hP
5. 如果G = 0，重复上一步，直到G != 0 这样我们就得到了 长度n和h和Base Point G

椭圆曲线的加解密

通过以上步骤我们得到了有限域的Base Point G，那么算法就可以进行下去了。随机生成一个超大数k(256 bit) 作为私钥，计算kG得到公钥。

现在A要向B发送信息m，首先A将m映射到椭圆曲线上形成P_m ,例入encode一个Hash，其实它就是big integer,利用它作为横坐标，很容得到P_m 。

A将信息加密： $C_m = \{k_A G, P_m + k_A P_B\}$ 这里`k`为A的私钥,P_B 为B的公钥 C_m 为信息m 加密后的结果,C_m的前半部分为A的公钥P_A，后半部分为加密的信息

B拿到了信息C_m 后由于解密的方程如下： $P_m + k_A P_B - k_B(k_A G) = P_m + K_A K_B G - K_B K_A G = P_M$

A通过将$k_A P_B$ 与P_M 相加来伪装信息P_M ,因为只有A知道k,所以即使P_B 是公钥，任何人无法伪装这个信息，通过解密的公式可以知道只有拥有私钥K_B 的人才能解密P_M