分类号_____

UDC _____ 编号_____

中南大學

CENTRAL SOUTH UNIVERSITY

硕士学位论文

论 文 题 目 椭圆曲线密码系统的实现及安全性分析

学 科、专 业 软件工程

研 究 生 姓 名 赵云辉

导师姓名及专业技术职务 蒋外文 教授 尹双云高级工程师

原 创 性 声 明

本人声明，所呈交的学位论文是本人在导师指导下进行的研究工作及取得的研究成果。尽我所知，除了论文中特别加以标注和致谢的地方外，论文中不包含其他人已经发表或撰写过的研究成果，也不包含为获得中南大学或其他单位的学位或证书而使用过的材料。与我共同工作的同志对本研究所作的贡献均已在论文中作了明确的说明。

作者签名：
日期：2012年 2月 28日

关于学位论文使用授权说明

本人了解中南大学有关保留、使用学位论文的规定，即：学校有权保留学位论文，允许学位论文被查阅和借阅；学校可以公布学位论文的全部或部分内容，可以采用复印、缩印或其它手段保存学位论文；学校可根据国家或湖南省有关部门规定送交学位论文。

作者签名：
导师签名：
日　　期：2012年 5月 30日

摘 要

椭圆曲线密码体制以其特殊的优越性已引起信息安全及密码学界的高度重视，从安全性及有效性看，这种密码体制有着广阔的前景，普遍认为它将代替目前通用的 RSA 密码体制而成为新的通用公钥密码。椭圆曲线密码体制已成为公钥密码研究的主流，现已逐渐成为公钥密码中的研究和开发的热点。而椭圆曲线作为公钥密码体制的基础是由其定义在有限域上的椭圆曲线上的点的集合构成的 Abelian 加法群，由此可定义其上的离散对数，即椭圆曲线离散对数。

本文系统地论述了椭圆曲线密码理论的研究现状，描述了椭圆曲线上的基本算法及椭圆曲线的基本密码学性质，重点讨论了有限域的确定；安全椭圆曲线的选取算法和基点的选取算法；ECC 快速算法的设计及实现；椭圆曲线密码体制的安全性。比较了 ECC 与经典公钥密码体系间的优劣；安全的椭圆曲线构造和基点的选取，是椭圆曲线密码体制实现的的关键。标量乘法运算是影响 ECC 实现的关键步骤，加速标量乘，提高 ECC 的实现速度，是 ECC 实现中一个重要的研究课题。对椭圆曲线的离散对数问题算法、攻击情况、椭圆曲线挑战情况、安全椭圆曲线选取、其它公钥密码系统比较等给出了详细的论述，体现了该领域的最新成就，并对今后 ECC 的可能发展方向及研究内容提出了看法。说明 ECC 比其它公钥密码能提供更好的加密强度，更快的速度和更小的密钥长度，因此可用较小的开销和时延实现较高的安全性。

关键字 椭圆曲线，椭圆曲线离散对数，ECC，算法，安全性

ABSTRACT

The elliptic curve cryptosystem is more and more be high notice by its special superiority. The main advantage of elliptic curve cryptosystems over other public-key cryptosystems is the fact that they are based on a different intractable problem, and that their keys have smaller sizes for the same level of security. For information security and cryptography, Cryptography universally think that the elliptic curve cryptosystem is going to stead the current RSA cryptosystem at present and become a new current public key cryptosystem. In recent years, several studies have been conducted on implementation of elliptic curve cryptosystem.

The ECC, as an optimal scheme of advanced public key cryptosystem become a research focus. The base that using elliptic curve is public-key cryptosystem is become of the points set of elliptic curve on finite field can construct Abelian group. by this, the discrete logarithm on public curve can be define. The discrete logarithm on elliptic curve is very difficult to be solved.

In this paper, The Basic contents of the cryptology on elliptic curves are summarized. A description is given to the basic arithmetic and cryptography property on elliptic curves. The major part of this paper is dedicated to select a suitable finite field, select logarithms of secure elliptic curve and basic point, design and implement a fast arithmetic of ECC based over finite prime field, discuss security of elliptic curve cryptography. The ECC and classical public key cryptogram system are compared. Point out that the secure elliptic curve is the master key of constructing the elliptic curve cryptosystem. The scalar multiplication of elliptic curve is one of important problem in the practice of efficiently implementing an elliptic curve cryptosystems. The speed of the scalar multiplication is the key steps of the implement elliptic curve cryptosystem. We discussed the discrete logarithm

problem of elliptic curve and the available attack in the theory, then analyzed some attack methods and for elliptic curve cryptosystems. The newest achievements in this area are covered and views on the development of the cryptosystems are proposed. It is shown that ECC can provide greater strongest , highest speed and smaller keys than other systems, Thus ECC can relatively high security with less cost and time delay.

KEY WORDS elliptic curves, elliptic discrete logarithm, Elliptic curves cryptosystem, logarithms, Security

目录

第一章　绪论

1.1　课题研究的背景

在计算机网络、数据通讯及存储系统中存在着许多潜在的不安全因素，密码技术是保护信息系统的机密性、完整性的有效手段，是解决信息网络数据安全和应用安全的核心问题，它由密码编码技术和密码分析技术两个分支组成。密码编码技术的主要任务是寻求产生安全性高的有效算法和协议，以满足消息认证、数字签名和电子商务等系统的要求；而密码分析技术的主要任务是破译密码或伪造认证信息，实现窃取机密信息或进行诈骗破坏活动。

自 1976 年 Diffie 和 Hellman 提出公钥密码的思想以来，国际上大量的公钥密码体制被相继提出，所有这些公钥密码体制的安全性均依赖于数学问题的难解性。

近年，一些公钥密码系统被攻破；另一些则被证明是难以实现的。迄今只有三类密码体制被认为是安全和有效的。根据其依赖的数学问题，将这些系统分为：

基于因数分解问题的因数分解密码系统，典型代表为 RSA；

基于离散对数问题的离散对数密码系统，典型代表为 DSA；

基于椭圆曲线离散对数问题的椭圆曲线密码系统(ECC)。

寻找新的数学难题作为密码资源早就是人们努力的一个方向。椭圆曲线理论是线性代数、数论等多个数学分支的一个交叉点，一直被认为是纯理论学科内容，直到 1985 年 Koblitz 和 Miller 将椭圆曲线理论引入到密码学中，提出了基于椭圆曲线离散对数问题的椭圆曲线密码系统(ECC)[1,2]。

经典的 RSA、Diffie-Hellman 等公钥密码体制的密钥长度一般为 512 字节，但是，随着密码理论研究的深入，计算机技术的飞速发展，这一密码长度已变得越来越不安全了。1999 年 8 月 RSA-512 被攻破，因而密钥不得不被加长，要达到对称密钥 128 字节的安全水平，NIST 推荐使用 3072 字节的 RSA 密钥。很显然，密钥长度的增加，对本来计算速度就很慢的 RSA 来说，更是不堪负重，且这种现象还将继续下去。而 ECC 相对于经典公钥在算法效率上取得了突破性进展。RSA 的危机在于其存在亚指数算法，而椭圆曲线密码体制不存在亚指数攻击算法，且其密钥长度大大减小了，256 字节的 ECC 密钥就可达到对称密钥 128 字节的安全水平，这样就为 ECC 的使用提供了充分的空间，也为 ECC 的广泛应用提供了条件。

1．2 课题研究的目的和意义

基于椭圆曲线上的有理点构成的 Abel 群实现的椭圆曲线密码体制，是目前公钥密码学研究的一个热点。椭圆曲线密码体制作为新一代公钥密码的最佳方案，既可以用于信息加密，又可用于数字签名等；在计算机网络安全与信息安全领域应用前景十分广阔。

椭圆曲线作为密码系统的诱人之处在于：

1、在安全性相当的情况下，可使用较短的密钥，一般认为 q 元域上的椭圆曲线密码体制，当 q 长度为 160bit 时，其安全性相当于 RSA 使用的 1024bit 长密码。密码短意味着小的带宽和存贮要求，这在某些应用中可能是决定性因素。

2、对椭圆曲线密码系统的资源而言，同一个有限域上存在着大量的椭圆曲线，这为其安全性增加了额外的保证，也为软、硬件的实现带来了方便。

3、分析 RSA 和 DSA 密码系统等均存在亚指数算法，而目前分析椭圆曲线密码系统不存在亚指数算法[3]。

因此椭圆曲线密码体制将逐步取代RSA等公钥密码而成为一个十分令人感兴趣的密码学分支。1997 年以来，椭圆曲线密码系统在国际上形成了一个研究热点。国内外不少专家和学者对椭圆曲线密码体制进行了研究，并取得了许多成果[4,5,6,7]。目前，国内外有大量的厂商已经使用或计划使用椭圆曲线密码体制的产品。例如加拿大的 Certicom 公司将所有人力、物力、财力都投在椭圆曲线密码上，专门生产 ECC 产品，为其他公司提供服务；著名的 Motorola 公司则将 ECC 用于它的 CipherNet，将安全特性加入到应用软件中；SET(Secure Electronic Transaction)协议的创立者 Vis 和 Matercard 公司都计划使用 ECC 等。ECC 对 RSA 已经形成强劲的挑战。

近年来，椭圆曲线密码体制越来越受到重视，它在 Internet 协议安全、国际电子商务、远程通讯、移动电话、智能卡等方面得到了广泛的应用。

1．3 论文研究的工作及创新之处

本文系统地论述了椭圆曲线密码的基本理论知识及研究现状，重点讨论了椭圆曲线有限域的确定、安全椭圆曲线的选取算法、基点的选取算法、ECC 快速算法的设计和实现、椭圆曲线密码系统的安全性等。给出了有限域上椭圆曲线密码系统的构造方法及算法，比较了 ECC 与经典公钥密码体系的优劣，阐述了安全椭圆曲线的构造和基点的选取，是椭圆曲线密码体制实现的关键。标量乘法运算是影响 ECC 实现速度的重

要内容，设计和改进了 ECC 的快速算法，提高 ECC 的实现速度；其次对椭圆曲线的离散对数问题的相关算法、椭圆曲线密码的攻击算法、硬件攻击、椭圆曲线的挑战等做了较详细的探讨，分析了椭圆曲线密码的明文编码，并提出了今后 ECC 的可能发展方向。

文中对安全椭圆曲线的构造给出了选择原则，根据选择定则对有限域的确定、安全椭圆曲线的选取算法及基点的选取算法进行了设计，并通过模拟得出相关结论；同时探讨并设计了 ECC 的快速算法；对其安全性给出了详细的研究。

椭圆曲线密码的研究方兴未艾，许多问题还有待解决，将逐步走向成熟；对它的进一步研究，一方面可以为 ECC 的应用和安全性提供基础和保障；另一方面，也将为密码理论展示更加丰富的数学背景。

第二章　椭圆曲线基本理论简介

由于koblitz 和miller的开创性工作，使被数学家们研究了一百多年的椭圆曲线在密码学领域中发挥着重要作用。基于椭圆曲线的密码体制是最近发展起来的且安全性能较好的一种体制，在一定程度上代表着公钥密码体制的发展方向。而构造椭圆曲线密码体制的数学基础是利用椭圆曲线上的点构成的Abelian加法群中的离散对数的计算困难性。已有的攻击算法表明：基于有限域椭圆曲线上的离散对数问题的困难性要高于一般乘法群上的离散对数问题的困难性，且椭圆曲线所基于的域的运算位数远小于传统离散对数的运算位数，且很容易在计算机的软件和硬件上实现。为构造和实现安全椭圆曲线密码体制，下面简单介绍椭圆曲线基本理论。

2.1　椭圆曲线的定义

椭圆曲线的研究来源于椭圆积分[8]：

$$\int \frac{dx}{\sqrt{E(x)}} \tag{2-1}$$

此处的E（x）是x的三次多项式或四次多项式。这里的积分不能用初等函数来表达，因而引入椭圆曲线函数。

在射影平面 p_k^2 上的齐次表达式

$$E: Y^2Z + a_1XYZ + a_3YZ^2 = X^3 + a_2X^2Z + a_4XZ^2 + a_6Z^3 \tag{2-2}$$

称为weirstrass等式。若F是一个有限域，$a_i \in F_q$，$i=1,2,3,4,5,6$。令

$$F(X,Y,Z) = Y^2Z + a_1XYZ + a_3YZ^2 - X^3 - a_2X^2Z - a_4XZ^2 - a_6Z^3 \tag{2-3}$$

称F（X，Y，Z）为weirstrass多项式。当 $F(X,Y,Z)\left(\frac{\partial f}{\partial X},\frac{\partial f}{\partial Y},\frac{\partial f}{\partial Z}\right) \neq (0,0,0)$ 时，由weirstrass等式所确定的曲线即为椭圆曲线。

由于 p_k^2 模型由仿射平面 q_k^2 添加一条虚直线而成，所以射影平面 p_k^2 上椭圆曲线有其仿射平面 q_k^2 上的等价形式，若 $x = X/Z, y = Y/Z$，将其代入到weirstrass等式中，经过变换可得到仿射平面 q_k^2 上等价的椭圆曲线方程：

$$y^2 + a_1xy + a_3y = x^3 + a_2x^2 + a_4x + a_6 \quad (2\text{-}4)$$

若 F 是一个域，$a_i \in F_q$，i=1, 2, 3, 4, 5, 6。满足式(2-4)的数偶(x, y)及无穷远点的特殊点 O 称为 F 域上的椭圆曲线 E 上的点，椭圆曲线 E 上的所有点加上无穷远点 O 构成有限阿贝尔群，记为 $E(F_q)$。

设 P 为一素数，n 是正整数，记 $q = P^n$，记 F_q 是特征值为 P 的有限域。当其特征值不等于 2 时，对方程(2－2) 作如下坐标变换：

$$(x,y) \to (X, Y - (a_1/2)X - a_3/2) \quad (2\text{-}5)$$

则可将曲线 E 变为：

$$E_1 : y^2 = x^3 + b_2x^2 + b_4x + b_6 \quad (2\text{-}6)$$

则在有限域 F_q 上 E 与 E_1 是同构的。

当其特征值不等于 2，3 时，对方程(2－6) 作如下坐标变换：

$$(x,y) \to ((x - 3b_2)/36, y/216) \quad (2\text{-}7)$$

可以将 E_1 变换为：

$$E_2 : y^2 = x^3 + ax + b \quad (2\text{-}8)$$

由于在有限域 F_q 上 E_1 与 E_2 上同构，因而 E 与 E_2 也是同构的。

当其特征值等于 2 时，对方程(2－2) 作如下坐标变换：

$$(x,y) \to (a_1^2X + a_3/a_1, a_1^3Y + (a_1^2 + a_4 + a_1^2)/a_1^3) \quad (2\text{-}9)$$

可将曲线 E 化为 E_3：

$$y^2 + xy = x^3 + a_2x^2 + a_6 \quad (2\text{-}10)$$

对曲线 E_3 再作变换：

$$(x,y) \to (x + a_2, y) \quad (2\text{-}11)$$

则曲线 E_3 化为 E_4：

$$y^2 + a_3y = x^3 + a_4x + a_6 \quad (2\text{-}12)$$

因此，在不同特征值的域上对方程进行分析，可分为如下三种情况：

特征值 P=2 时，$E_1 : y^2 + xy = x^3 + ax^2 + b$ 或 $E_2 : y^2 + ay = x^3 + bx + c$

特征值 P=3 时，$E_3 : y^2 = x^3 + b_2x^2 + b_4x + b_6$

特征值 $P \neq 2,3$ 时，$E_4 : y^2 = x^3 + ax + b$

2.2　GF(P)上的椭圆曲线

设 F_q 是特征值大于 3 的有限素整数域，则在仿射坐标平面上，定义在有限域 F_q 上的椭圆曲线 $E(F_q)$ 是由满足下式

$$y^2 = x^3 + ax + b \text{ 其中 } a,b \in F_q \quad (2\text{-}13)$$

及判别式$4a^3+27b^3\neq 0$的点$(x,y)\in GF(q)\times GF(q)$及无穷远点O所组成的在$F_q$上的非空集合。这些点在下面定义的加法运算下构成一个阿贝尔群，群运算的恒等元是O。利用“正切和弦”的几何法则，如图2-1所示，可定义一个阿贝尔加法群，其运算规则如下：

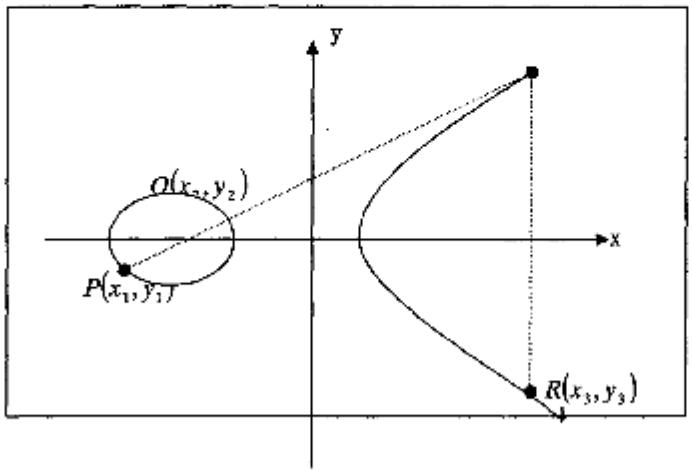

图2-1　椭圆曲线上阿贝尔加法群的几何意义

设P和Q是椭圆曲线$E(GF(q))$上的两个点。若P是椭圆曲线E上的无穷远点O，即P=O，则-P=O，且P+Q=Q+P=Q，Q可看成加法幺元。

若P、Q均不为无穷远点，令$P=(x_1,y_1)$，$Q=(x_2,y_2)$，则$-P=(x_1,-y_1)$，且$P+(-P)=O$；如果P、Q的X坐标不同，则直线L=PQ截曲线有且仅有一个点R（若L为曲线的切线，则设P=R或Q=R为切点），定义P+Q=-R，即所截的第三个点的镜像，如图2-2所示。由此可计算2P，而P+Q称为点加，对于M个相同的点相加，即P=P+…+P，可表示为M*P，称为点乘或数乘。

1、射坐标的点加公式

设$P=(x_1,y_1)$，$Q=(x_2,y_2)$，则$P+Q=(x_3,y_3)$，它们都是椭圆曲线E上的点。当$Q\neq -P$，其点加公式为：

$$x_3=\lambda^2-x_1-x_2 \tag{2-14}$$

$$y_3=\lambda(x_1-x_3)-y_1 \tag{2-15}$$

$$\lambda=(y_2-y_1)/(x_2-x_1) \tag{2-16}$$

当P=Q时，其点加公式为：

$$x_3=\lambda^2-2\lambda \tag{2-17}$$

$$y_3 = \lambda(x_1 - x_3) - y_1 \tag{2-18}$$

$$\lambda = (3x_1^2 + a)/2y_1 \tag{2-19}$$

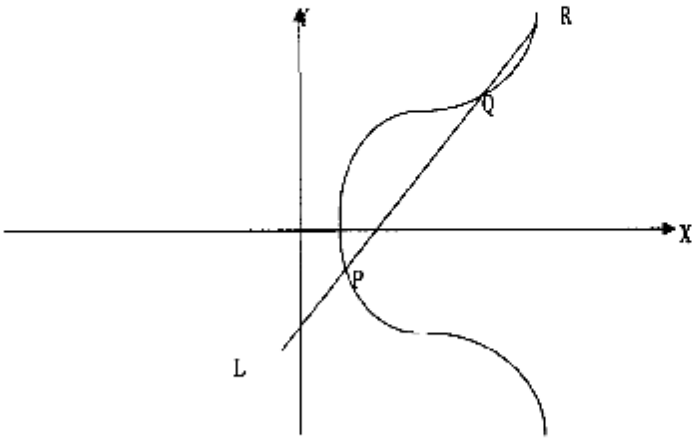

图 2-2　$y^2 = x^3 + ax + b$ 椭圆曲线

2、射影坐标的点加公式

设 $P=(x_1, y_1, z_1)$，$Q=(x_2, y_2, z_2)$，$P+Q=(x_3, y_3, z_3)$

当 $Q \neq -P$ 时，其点加公式为：

$$x_3 = uw \tag{2-20}$$

$$y_3 = u(v^2 x_1 z_2 - w) - v^3 y_1 z_2 \tag{2-21}$$

$$z_3 = v^3 z_1 z_2 \tag{2-22}$$

其中 $u = y_2 z_1 - y_1 z_2, v = x_2 z_1 - x_1 z_2, w = u^2 z_1 z_2 - v^2(x_2 z_1 + x_1 z_2)$

当 P=Q 时，其点加公式为：

$$x_3 = 2vw \tag{2-23}$$

$$y_3 = u(4x_1 y_1 v - w) - 8y_1^2 v^2 \tag{2-24}$$

$$z_3 = 8v^3 \tag{2-25}$$

其中 $u = az_1^2 + 3x_1^2, v = y_1 z_1, w = u^2 - 8x_1 y_1 v z_1$

仿射坐标与射影坐标下的点加公式是等价的，但用于不同目的，效果不同。当仅计算点加时，仿射坐标点加公式只需 2 次或 3 次乘法，1 次除法；而射影坐标点加公式用于单次点加时，不能省去除法，且乘法运算比仿射坐标点加运算次数多，因此，这种情况下用射影坐标点加公式比用仿射坐标点加公式慢。而当用于连续点加以求数

乘时，影射坐标点加公式可避免除法，它比仿射坐标点加公式要快得多。

2.3 $GF(2^m)$上的椭圆曲线

设E为特征值为2的有限域$GF(2^m)$上的非奇异椭圆曲线，则在仿射坐标中它定义为满足

$$E_1: y^2+xy=x^3+ax^2+b \text{ 或}$$
$$E_2: y^2+ay=x^3+bx+c \qquad (2\text{-}26)$$

的点$(x,y)\in GF(2^m)\times GF(2^m)$和曲线上无穷远点O所组成的集合。这里$a,b,c\in GF(2^m)$且$b,c\neq 0$，一般将$E(GF(2^m))$简记为E，这些点在下面定义的加法运算下构成一个阿贝尔加法群。

1、设P和Q是椭圆曲线E_1上的两个点，P=(x_1,y_1)，Q=(x_2,y_2)。若P是椭圆曲线E_1上的无穷远点O，即P=O，则-P=O，且P+Q=Q+P=Q，Q可看成加法幺元。若P=(x_1,y_1)，则$-P=(x_1,y_1+x_1)$，且P+(-P)=(-P)+P=O。

下面给出仿射坐标下的点加公式：

设P=(x_1,y_1)，Q=(x_2,y_2)，则P+Q=(x_3,y_3)，它们都是椭圆曲线E_1上的点。

若$Q\neq -P$，其点加公式：

$$x_3=\lambda^2+\lambda+x_1+x_2+b, y_3=\lambda(x_1+x_3)+x_3+y_1 \qquad (2\text{-}27)$$
$$y_3=\lambda(x_1+x_3)+x_3+y_1 \qquad (2\text{-}28)$$

其中$\lambda=(y_1+y_2)/(x_1+x_2)$。

若P=Q，其倍点公式为：

$$x_3=x_1^2+b/x_1 \qquad (2\text{-}29)$$
$$y_3=x_1^2+(x_1+y_1/x_1)x_3+x_3 \qquad (2\text{-}30)$$

2、设P和Q是椭圆曲线E_2上的两个点，P=(x_1,y_1)，Q=(x_2,y_2)，则P+Q=(x_3,y_3)。仿射坐标下的点加公式为：

若$Q\neq -P$时，

$$x_3=\lambda^2+x_1+x_2 \qquad (2\text{-}31)$$
$$y_3=\lambda(x_1+x_3)+y_1+a \qquad (2\text{-}32)$$

其中$\lambda=(y_1+y_2)/(x_1+x_2)$。

若P=Q时，

$$x_3=(x_1^4+b^2)/a^2 \qquad (2\text{-}33)$$
$$y_3=((x_1^2+b)/a)(x_1+x_3)+y_1+a \qquad (2\text{-}34)$$

3、射影平面点加及倍点公式

设 x=X/Z，y=Y/Z²，将其代入到到方程（2-26），有

$$E_5: Y^2+XYZ=X^3Z+aX^2Z^2+bZ^4 \tag{2-35}$$

设 P=(x_1,y_1,z_1)，Q=(x_2,y_2,z_2)，P+Q=(x_3,y_3,z_3)，P、Q 是 E_5 上的两个点，则其点加公式为：

$$\begin{aligned}
&(X_1,Y_1,Z_1)+(X_2,Y_2,Z_2)=(X_3,Y_3,Z_3)\\
&A_1=Y_2Z_1^2, A_2=Y_1Z_2^2, B_1=X_2Z_1, B_2=X_1Z_2, C_1=A_1+A_2, C_2=B_1+B_2\\
&D_1=Z_1Z_2, D_2=C_2D_1, E_1=C_2^2(D_2+aD_1^2), E_2=C_1D_2\\
&F_1=C_2^2B_1D_1+X_3, F_2=C_2^2A_1+X_3\\
&X_3=C_1^2+E_2+E_1, Y_3=E_2F_1+Z_3F_2, Z_3=D_2^2
\end{aligned} \tag{2-36}$$

其倍点公式为：

$$\begin{aligned}
&2(X_1,y_1,Z_1)=(X_2,Y_2,Z_2)\\
&Z_2=X_2^2Z_1^2, X_2=X_1^4+bZ_2^4, Y_2=bZ_1^4Z_2+X_2(aZ_2+Y_1^2+bZ_1^4)
\end{aligned} \tag{2-37}$$

射影平面中点加的计算量为：有限域中 6 次加法、9 次乘法，4 次平方。而倍点的计算量为：有限域中 4 次加法、4 次乘法、5 次平方。

2.4　椭圆曲线的阶

为保证椭圆曲线密码系统的安全性，椭圆曲线的阶不能被小素数整除。因而在椭圆曲线密码系统中，计算椭圆曲线的阶是一个非常重要的内容。设椭圆曲线 E 在 F_q 上的阶用 # $E(GF(q))$ 表示，# $E(GF(q))=1+\sum_{x\in F_q}(1+l)=1-p+\sum_{x\in F_q}l$，而公式中 $l=\left(\frac{x^3+ax+b}{p}\right)$是勒让德符号。

根据仿射坐标中椭圆曲线 E 的定义：$y^2=x^3+ax+b$，故当 x^3+ax+b 是一个模 P 的平方剩余时 $E(GF(q))$将增加两个点$(x,|y|)$和$(x,-|y|)$；若 P|（x^3+ax+b）将增加一个点（X，0）；若 x^3+ax+b 既不是模 P 的平方剩余也不能被 P 整除，$E(GF(q))$将不增加任何点。由 Hasse 定理，如下不等式成立：

$$p+1-2\sqrt{p}\le \#E(GF(q))\le p+1+2\sqrt{p} \tag{2-38}$$

椭圆曲线 E 上点 P 的阶 M 是使 MP=0 的最小整数，M 称为 P 的阶，记为 ord(P)=M。让 $\#(E_q)$表示在椭圆曲线 E 上的点的数量，显然 E 至多有 2q+1 个点，即无穷远点与 $2q$ 个点对(x, y)。 $\#(E_q)$较理想的值是接近于 q 值。要计算有限域一条椭圆曲线 E 上的点群的阶，一般是比较困难的。Schoof 算法[9]是一个在多项式时间内确定阶的方法，但当 q 很大时，该算法并不适用。下面给出一种计算 $E(GF(q))$的方法[10]。

先选择一个 E 中的点 P，计算 M，$p+1-2\sqrt{p}\le M\le p+1+2\sqrt{p}$ 使得 MP=0，若 M 在此间隔内是唯一的，根据 Hasse 定理，M 就是椭圆曲线 E 的阶。

1、计算$u=p^{1/4}$,计算列表$l_1=\{0,\pm p,\pm 2p,\pm 3p\cdots\pm up\}$;

2、计算$Q=(2u+1)P, R=(p+1)P, v=2\sqrt{p}/(2u+1)$;

3、计算列表$l_2=\{R,R\pm Q,R\pm 2Q,R\pm 3Q\cdots R\pm vQ\}$;

4、列表l_1中含2u+1个点，列表l_2中含有2v+1个点，由v的定义有$u=v$，则由Hasse定理，必然存在A=B（$A\in l_1, B\in l_2$），故R+iQ=jP，于是（p+1）P+i（2u+1）P=jP，最后结果为（（p+1）+i（2u+1）-j）P=0;

5、M=（p+1）+i（2u+1）-j，若$p+1-2\sqrt{p}\le M\le p+1+2\sqrt{p}$，且M是唯一的，M就是椭圆曲线E的阶。若M有两个解，$M_1P=M_2P$，此时应重新选择P点。

2.5　椭圆曲线离散对数

根据有限域上椭圆曲线点群运算规则，可知椭圆曲线上的点加运算类似于有限域$GF(q^*)$上的两个元素相乘，故椭圆曲线上的点与有限域上的整数的倍乘（点积）相当于$GF(q^*)$上元素的幂运算，由上述运算公式，可以给出点积mP的运算，即

$$mP=\underbrace{P+P+P+\cdots+P}_{共m个} \tag{2-39}$$

这里$m\in N$。对于2P的形式，也称为P的平方。

椭圆曲线上的离散对数问题是指：给定一条有限域$GF(q)$的椭圆曲线E_q及二个点P和B，$P,B\in E_q$。寻找一个整数K，使得$P=KB$，如果这样的数存在，这就是椭圆曲线离散对数。即选取椭圆曲线上的一个点P(x, y)，作为一个基点，那么给定一个整数M，计算MP=K是容易的，但要从P点推导出整数M，则是非常困难的。既没有多项式时间内的求解算法，这就是ECDLP，也是构造ECC的数学基础。

2.6　椭圆曲线域参数及有效验证

基于椭圆曲线的每一种公钥密码体制的操作，均包含由一些椭圆曲线域参数所确定的有限域椭圆曲线上的算术运算。在SECI及IEEE P1363工作草案中，所定义的椭圆曲线参数域由下面的一个六元偶所组成：

$$T=(P,a,b,g,n,h) \tag{2-40}$$

其中整数P表示一个有限域E_q，m表示F_{2^m}，二元素$a,b\in F_q$，指出了由方程(2-3)定义的椭圆曲线，g表示一个基点，n为素数且等于点g的阶，$h=\#E(F_q)/n$，h是一个小整数，称为余因子。根据具有不同的性质，则有不同的基点$G(x_G,y_G)$选择计算算法。基于椭圆曲线的密码系统的椭圆曲线系统参数中，在密码系统中使用的有限域的种类主要有

GF(p)、$GF(2^m)$，其中p是大于3的素数。FR是有限域上的元素的表示方法。例如:在特征(characteristic)为2的有限域上，有限域上的元素可以使用正规基(normal basis)或者多项式基(polynomial basis)等表示方法，在素数域上的元素一般可以直接使用整数表示。a和b表示椭圆曲线的系数，在$GF(p)$上特征大于3的曲线的方程为$y^2 = x^3 - ax + b$，而其系数必须满足椭圆曲线的判别式不为0，即$\Delta = 4a^2 + 27b^2 \neq 0$。在$GF(2^m)$上曲线的方程为$E_1 : y^2 + xy = x^3 + ax^2 + b$或$E_2 : y^2 + ay = x^3 + bx + c$，系数必须满足b≠0。异常曲线(Anomalous curves)和超奇异曲线(Supersingular curves)已经被证明是不安全的曲线，一定要避免使用它。n是该基点的阶(order)，n必须是一个足够大的素数，否则Pohlig-Hellman攻击将会威胁到系统安全。而h是n的余因子(cofactor)，即#(E)=nh(其中N(E)为椭圆曲线的阶，。

在给出了椭圆曲线域参数 T 之后，还要验证 T 是否有效。也就是说，一方面指是否符合 T 的算法需求，另一方面指防止不安全参数的恶意插入或检测传输错误。在 SECI[11]的 ECC 工作草案中，给出的有效验证过程如下：

1、检验 p 是一个奇素数，使得 lbp=2t。这里$t \in \{56,64,80,96,112,128\cdots\}$是以比特表示的近似的安全级，例 t=96 时，p 则是一个 192 比特长的奇素数。

2、检验a, b, x_G, y_G是区间$[1, p-1]$内的整数。

3、检验$4a^3 + 27b^3 \neq 0 (\mathrm{mod}\, p)$。

4、检验$y_G^2 = x_G^3 + ax_G + b (\mathrm{mod}\, p)$。

5、检验 n 是一个素数，且 nG=0。这时的 O 是一个点群的幺元。

6、检验$h \leq 4$，且$h = \left\lfloor \left((p)^{1/2} + 1\right)^2 / n \right\rfloor$，防止大步小步法攻击。

7、$\#E \neq E(GF(q)) \neq p$；防止防止异常椭圆曲线。

2.7　椭圆曲线的 j 不变量

设椭圆曲线 E 由(2-2)式给出，可定义如下特征量：

$$d_2 = a_1^2 + 4a_2$$
$$d_4 = 2a_4 - a_1 a_3$$
$$d_6 = a_3^2 + 4a_6$$
$$d_8 = a_1^2 a_6 + 4a_2 a_6 - a_1 a_3 a_4 + a_2 a_3^2 - a_4^2$$
$$c_4 = d_2^2 - 24d_4$$
$$\Delta = -d_2^2 d_8 - 8d_4^3 - 27d_6^2 + 9d_2 d_4 d_6 \qquad (2\text{-}41)$$
$$j(E) = c_4^3 / \Delta \qquad (2\text{-}42)$$

其中△被称为 Weisterass 等式的判别式，j(E)当$\Delta \neq 0$时称为 E 的 j 不变量。

设 $E(F_q)$ 为椭圆曲线，E 的 j 不变量就是由 a_1,a_2,a_3,a_4,a_6 决定的 F_q 上的一个元素，记为 j(E)，具有如下的重要性质：

1、两个椭圆曲线在 $\overline{F_q}$ 上是同构的，当且仅当它们具有相同的 j 不变量。

2、对任何 $j_0 \in F_q$，存在一条定义在 F_q 上的椭圆曲线，它具有等于 j_0 的 j 变量。如 $j_0 \neq 0,1728$；并且特征值大于等于 5，j(E) 就等于 j_0，此处椭圆曲线 E 为：

$$y^2 = x^3 + \frac{3j_0}{1728 - j_0}x + \frac{2j_0}{1728 - j_0} \qquad (2\text{-}43)$$

第三章　椭圆曲线密码系统的构造

利用椭圆曲线构造椭圆曲线密码体制中，问题之一是先要构造安全的椭圆曲线。所谓安全，是指建立在所构造的椭圆曲线上的密码体制能够抗击目前存在的所有攻击。目前对椭圆曲线离散对数的主要攻击方法有：

1、穷举搜索法；

2、大步小步法；

3、Pohlig—算法；

4、Pollard—rho 算法；

5、并行 Pollard—rho 算法；

6、pollard—Lameda 算法；

7、倍对数算法；

8、Xedni 计算法；

9、MOV 攻击等。

对于一般的 ECDLP，目前有两种较好的求解法：pollard—Hellman 算法和并行 Pollard—rho 算法。但对于两类特殊的椭圆曲线：一类是超奇异椭圆曲线，利用 MOV 算法和 FR 算法可以解决 ECDLP；另一类是异常椭圆曲线，利用 SSAS 算法可以解决 ECDLP。在基于椭圆曲线的加密算法之前，必须完成下列工作：

有限域的选择：在实际应用中，通常在$GF(p)$、$GF(2^m)$二者之中选择其一。

基点的确定：在有限域选定之后，一旦给定椭圆曲线 E，其基点也就相应地确定了，故基点的选择对椭圆曲线密码体制起着至关重要的作用。

3.1　有限域的确定

要构造安全的椭圆曲线，首先遇到的问题是确定有限域。椭圆曲线密码的原理是基于有限域上椭圆曲线离散对数问题(ECDLP)，ECDLP 的计算复杂度目前已知仅与点群的阶有关，有限域的代数结构在此不起作用。因而在实际应用中，通常在$GF(p)$和$GF(2^m)$二者之间选择其一，至于哪个域上的安全性更好一些，理论界还没有明确的结论，在一定意义上，可近似认为两者等价，在一般的软件环境下，使用$GF(2^m)$能提供比$GF(p)$更大的执行优越性，这对于工作站、HP 服务器之类的平台，尤其是对低费用、8 比特的智能卡效果更明显。要获得基于$GF(p)$的 ECC 快速执行需要一个密码协处理

器做模幂运算，这额外的协处理器增加了智能卡的费用；而使用$GF(2^m)$则不需要协处理器来分担计算执行。在使用 Pentium 处理器的平台上或带有做模幂运算的密码协处理器的智能卡上，$GF(p)$的执行要优于$GF(2^m)$的执行。

一般地说，若采用现成的处理器，由于它们具有通常算术运算指令而缺少特征 2 的域的运算功能，可能选择奇特征域要方便些；若采用硬件集成实现，由于乘法器是主要的功能部件，显然采用特征 2 的域实现要方便些。

1、若选择奇特征的域，一般采用素域$GF(p)$。而$GF(p)$中的运算是整数模 p 的运算，故选用 $p=b^m-c$，其中 b 是字长，c 较小且其汉明重量 W（c）也小，这能加快整数模的运算。

2、当域是$GF(2^m)$时，是采用多项式基还是正规基会影响计算效率；若是用硬件实现，用多项式基工程实现要简单一些；然而用正规基实现可提高运算效率，因为正规基下有限域元素的平方是循环移位，故在实现椭圆曲线密码时，要选择具有高斯优化正规基的域，下面给出 160〈m〈200 具有 1 型和 2 型高斯正规基的域$GF(2^m)$[12]：

1 型：m=162，172，178，180，196

2 型：m=165，173，174，179，183，186，189，191，194，198

设 $a=(a_0,a_1,a_2\cdots a_{m-1}),b=(b_0,b_1,b_2\cdots b_{m-1})\in GF(2^m)$，$a^2=(a_{m-1},a_0\cdots a_{m-2})$；设 T=1，u=1；对 1 型 GNB，T=2，u=-1；对 2 型 GNB，p=Tm+1

先计算$F(1),F(2),\cdots F(p-1),F(2^iu^j \bmod p)=i\ for\ 0\le i\le m-1,0\le j\le T-1$，

$$ab=c=(c_0,c_1,c_2\cdots c_{m-1}),\text{其中}c_i=\sum_{k=1}^{p-2}a_{F(k+1)+i}b_{F(p-k)+i},\ forT=2;$$

$$c_i=\sum_{k=1}^{m/2}(a_{k+i-1}b_{m/2+k-1}+a_{m/2+k-1}b_{k+i-1})+\sum_{k=2}^{p-2}a_{F(k+1)+i}b_{F(p-k)+i},\ forT=1$$

然后求$a^{-1},a\neq 0$时可使用下式进行计算：

$a^{-1}=a^{2^m-2}=a^{(2^{m-1}-1)2}$，当 m 为偶数时$a^{2^{m-1}-1}=aa^{2^{m-1}-2}=aa^{(2^{m-2}-1)2}$；

当 m 为奇数时，$a^{2^{m-1}-1}=a^{(2^{m-1/2}-1)(2^{m-1/2}+1)}=a^{(2^{m-1/2}-1)2^{m-1/2}}a^{2^{m-1/2}-1}$

当有限域确定后，选择安全的椭圆曲线是个相当重要的问题。

3.2　$GF(p)$上安全椭圆曲线的选取

1、要保证椭圆曲线密码的安全性，就是要使所选取的椭圆曲线能抵抗上述的关于

则

ECDLP 解决的算法和方法，故选取一条安全的椭圆曲线是一个很难的数学难题，下面提供几点选取安全椭圆曲线的规则：

- 为了抗击 Pollard—rho 算法攻击，所选取的 EC 的阶#E(GF(q))的分解式中应包含一个大的素因子，目前应不小于 160bit；
- 为了抗击 Weil 对和 Tate 对的攻击，对于 $1 \le k \le 30$，n 不能除 $q^k - 1$；
- 为了抗击 SSAS 攻击，所选椭圆曲线的阶不能完全等于该曲线所定义的有限域的阶，即 $\#E(F_q) \neq q$；
- 对于二进制域 $GF(2^m)$ 的度 m 不能为合数，否则有比 Pollard—rho 算法更快求解 ECDLP 的方法。
- 椭圆曲线域参数 h <=4，防止大小步法攻击。

2、为了构造出安全的椭圆曲线，下面给出三类适宜构造安全椭圆曲线的选取方法：

- 仅限于有限域为 F_2^m 上构造椭圆曲线：其原理是将定义在特征值较小的有限域的椭圆曲线提升到该域的扩域上，且 m 能被小整数 k>=1 整除。该方法简单，但获得的椭圆曲线较少。
- CM 法：根据给定的阶来选取符合此阶的椭圆曲线，其实现速度较快，但会产生具有附带的结构特征，从安全性方面考虑，这是一个潜在的威胁。
- 随机选取曲线：随机选取曲线的参数 $a,b \in F_q$，然后计算 u=#E(F_q)和大素数因子 n，直到所选的曲线符合安全要求为止。这种方法选取的曲线安全级别高，它完全取决于椭圆曲线阶的计算，这种计算有一定难度。

基于上述安全规则说明：在有限域 $GF(p)$ 上找到安全椭圆曲线 $E_{(a,b)}: y^2 = x^3 + ax + b$，应满足下列要求：$\#E_{(a,b)}$ 具有最大的素因子 n，$n > 2^{160}$ 且 $n > 4\sqrt{p}$，$E_{(a,b)}$ 是非超奇异的，$E_{(a,b)}$ 不是非正规的。

构造有限域 $GF(p)$ 上的安全椭圆曲线的方法一般有二种：随机曲线法(SEA 法)和 CM 法。随机曲线法是通用的方法，它可以构造出许多安全的椭圆曲线，只是实现起来比较复杂；CM 法相对比较简单，实现速度也较快，但用这种方法找出的椭圆曲线具有复乘，且和虚二次域的某个阶有内在的联系，故建立在它上成的椭圆曲线密码体制可能存在潜在的不安全因素。因此从 ECC 的安全考虑，利用随机曲线方法构造安全的椭圆曲线是非常必要的。

3.2.1　在 $GF(p)$ 上构造安全椭圆曲线的算法：

1、根据上述构造安全椭圆曲线的规则，在有限域 $GF(p)$ 上构造安全椭圆曲线的

算法之一[14]如下：

输入：有限域的大小 P，这些系数都具有形如 $P=2^n-c$（c 是小整数）的形式。由于 $GF(p)$ 上 ECC 实现的最基本的运算是 $GF(p)$ 中的域元素运算，而 $GF(p)$ 上域元素运算中得最多的是两个数的模乘运算。当 $P=2^n-c$ 为素数时，A、B 是 $GF(p)$ 中的两个元素，L=AB 是一个 2^n 比特的数，则 L 可写成 $L=2^nH+Q$ $H(Q<2^n)$，则 L=H+Q mod p。

输出：椭圆曲线 $E_{(a,b)}$，$\#E(a,b)=nh$，n 是大的素因子，$n>2^{160}$ 且 $n>4\sqrt{p}$。

1) 随机产生 $GF(p)$ 上一条椭圆曲线 $E_{(a,b)}: y^2=x^3+ax+b$；

2) 利用 SEA 算法计算出 $\#E_{(a,b)}$；

3) 利用大整数分解算法分解 $\#E_{(a,b)}$；并检测 $\#E_{(a,b)}$ 的分解中有无 $n>2^{160}$ 的素因子，如果没有，返回第 1 步；

4) 检测 n 是否等于 p，不等于就返回第 1 步；

5) 检测 $p^k\neq 1 \bmod n, k=1,2,3,\ldots\ldots 20$，返回第 1 步；

6) 输出 a，b，$\#E_{(a,b)}$；n，h=$\#E_{(a,b)}$；

实现上面算法的最关键一步是计算椭圆曲线点的个数 $\#E_{(a,b)}$；即 SEA 算法的实现。设 $p>3$，有限域 $GF(p)$ 上给定的椭圆曲线：$y^2=x^3+ax+b$，$a,b\in GF(p)$，且 $4a^3+27b^2\neq 0(\bmod p)$，计算它在这个域上的点的个数是一个非常重要和非常有意义的工作，这不仅是数论中的重要问题，而且也是密码学，特别是现代公钥密码学中的一个急需解决的问题。

3.2.2　计算椭圆曲线阶的 SEA 算法

要使椭圆曲线安全，则曲线的阶要充分的大。在构造安全椭圆曲线时，其核心步骤是寻找素数阶的椭圆曲线。目前处理此问题有二种方法：

1、复乘方法，即构造给定阶的椭圆曲线；

2、随机选取椭圆曲线参数，计算它的阶，直到找到素数阶椭圆曲线。

复乘方法找到的椭圆曲线具有附带的结构特征，从安全性方面考虑，这是一个潜在的威胁；而第二种随机选取的方法是比较理想的，它完全依赖于椭圆曲线阶的计算。在这一方面，R. Schoof 做了开创性的工作，1985 年提出了一个计算复杂度是多项式时间的算法，即著名的 Schoof 算法；但由于整个算法需要的存储量相当大，所以在具体实现算法时仍然不太可行。后经 Elkies 和 Atkind 的改进，该算法有了实用价值，因而被称为 SEA 算法[15]。后来 Morain Lercier 等人又对它进行优化，现今 SEA 算法已成为计算椭圆曲线阶的有效算法。

由 Hasse 定理知：$\#E(FG(p))=p+1-t$，其中 $[t]\leq 2\sqrt{p}$，并满足

$\phi^2 - t\phi + p = 0$,这里ϕ是 Frobenius 变换,即$\phi(x,y) = (x^p, y^p)$。若能确定出 t,也就确定了 $\#E_{(a,b)}$。SEA 算法的基本思想是先对一些小素数 l_i 计算出 $t_i = t \bmod l_i, i = 1,2,3\cdots s$,且$\prod_{i=1}^{s} l_i > 4\sqrt{p}$;然后利用中国剩余定理计算出 t。其中$l_i$可以从 2, 3, 5, 7, …取起。

输入:有限域的大小 P,椭圆曲线$E_{(a,b)}$的参数$a,b \in GF(p)$,

$j(E) = 12^3 \times 4a^3/(4a^3 + 27b^2), F(x,y) = y^2 - x^3 - ax - b$。

输出:$\#E_{(a,b)}$。

1、令 l=1; M=1; l=1;

2、While (M ($4\sqrt{p}$) do

(1)l=nextpyime (l);

(2)计算第 l 个模多项式$\phi_l(x,y) \in GF(p)[x,y]$;

(3)计算$\phi_l(x, j(E)) \in GF(p)[x]$;

(4)检验$GCD(x^p - x, \phi_l(x, j(E))) = G_l(x) \neq 1$,否则 goto(1);

(5)计算除多项式$d_l(x)$的一个$(l-1)/2$次因式$g_l(x)$;

(6)计算λ_l,使得$(x^p, y^p) = \lambda_l(x,y) \bmod (F(x,y), g_l(x))$;

(7)计算$t_l = (\lambda_l^2 + p)/(\lambda_l \bmod l)$,即得$t_k = t \bmod l$;

(8)$M = M \times l, i++$.

3、利用中国剩余定理计算 t

(1)for$(j = 1, j <= i, j++)$

$L_j = M/l_j; y_j = L_j^{-1} \bmod l_j$.

(2)计算$t = t_1 L_1 y_1 + t_2 L_2 y_2 + \cdots + t_i L_i y_i$

4、输出$\#E(FG(p)) = p + 1 - t$.

上述算法中,比较困难的是计算第 l 个模多项式$\phi_l(x,y) \in GF(p)[x,y]$和计算除多项式$d_l(x)$的一个$(l-1)/2$次因式$g_l(x)$。对于模多项式的计算,可以采用预计算的方法,可先预计算小于200, 180, 160等的所有素数的模多项式,并以文件的形式保存,在上述 SEA 算法中可直接调用。

经验说明,对于奇特征的域,当$2^{160} < q < 2^{200}$时,5~6 百条椭圆曲线中有一条较好的椭圆曲线,可用于构造安全的椭圆曲线密码系统;使用 Pentium4—2.8GHz 的计算机运行约需时间 4~7 小时。

3.3　GF(2^m)上安全椭圆曲线的选取

对域是$GF(2^m)$的椭圆曲线的选取,上述给出的安全椭圆曲线的选取规则也适应,

当域是$GF(2^m)$时，此时任一椭圆曲线E同构于$E_1: y^2+xy=x^3+ax^2+b$，b不等于0；由于有Order（E_1）=0 mod 2，Trace（a）=1；Order（E_1）=0 mod 4，Trace（a）=0。因而选择的椭圆曲线应满足如下条件：

Order（E_1）=2n，Trace（a）=1，n是素数；

Order（E_1）=4n，Trace（a）=0，n是素数。

对应的选择步骤为：

1、随机地选择整数a、b，$b \neq 0$；

2、计算阶n=Order（E_1），使得n为素数，且有

Order（E_1）=2n，Trace（a）=1，n是素数，

Order（E_1）=4n，Trace（a）=0，n是素数；

3、检验n满足$n \neq q, n \neq q^k-1, 1 \leq k \leq 20$；

4、随机选择曲线上一点I'，基点I取为

$I=2I', Trace(a)=1; I=4I', Trace(a)=0$。

选择符合上述条件的椭圆曲线并不困难，经验说明，当$2^{160}<q<2^{200}$时，对特征值为2的域，约2百多条椭圆曲线中有一条较好的椭圆曲线，可用于构造安全的椭圆曲线密码系统，用Pentium4-2.8GHz大约需用时一个小时。

3.4　基点的选取及计算

要构造基于椭圆曲线离散对数密码体制，就必须找出椭圆曲线加法群的大素因子子群的一个生成元，即基点。基点的选择中需要处理的主要问题是：

在曲线上随机选择一个有效点，在GF（p）上求解平方根；

如何通过随机选择的点得到阶是n的有效基点。

1、当所选取的整数x_i是方程（2-13）的平方剩余时，根据椭圆曲线域参数T中的p值，可计算基点的纵坐标y_i的值。

下面根据p的性质，利用逐次降幂法给出计算$y_i^2=A(\mathrm{mod}\ p)$的解的有效方法[15]，此处$A=x_i^3+ax_i+b(\mathrm{mod}\ p)$，并为模p的平方剩余，其算法的流程框图如图3-1所示。

(1)当$p=3(\mathrm{mod}\ 4)$或$p=7(\mathrm{mod}\ 8)$时，即p具有2k+1（k为奇数）的形式时，因$A^p \equiv A(\mathrm{mod}\ p), A^{(p-1)/2} \equiv x^{p-1}=1(\mathrm{mod}\ p)$；则有$1=(A/p) \equiv A^{(p-1)/2}(\mathrm{mod}\ p)$，此式两边同乘A，得$A=A^{(p+1)/2}=\left(A^{(p+1)/4}\right)^2$；由于$(p+1)/4$是整数，因此可求得其解为：

$y_i=y=\pm A^{(p+1)/4}(\mathrm{mod}\ p)$.

(2)当$p=1(\mathrm{mod}\ 4)$或$p=5(\mathrm{mod}\ 8)$时，即p具有4k-1（k为奇数）的形式时，

可得到$A^{(p-1)/2}-1=0(\mathrm{mod}\,p)$，对其分解可得：

$(A^{(p-1)/4}-1)(A^{(p-1)/4}+1)\equiv 0(\mathrm{mod}\,p)$

当$A^{(p-1)/4}=1(\mathrm{mod}\,p)$时，利用上面介绍方法可求解为$y=A^{(p+3)/8}(\mathrm{mod}\,p)$；

当$A^{(p-1)/4}=-1(\mathrm{mod}\,p)$时，利用平方非剩余乘平方非剩余得平方剩余的性质及上面介绍的方法可求得其解为：$y_1=y\equiv 2^{(p-1)/2}\cdot A^{(p+3)/8}(\mathrm{mod}\,p)$

(3)判别平方剩余的算法流程图如图 3-1 所示，其中 A 为基点 I 的横坐标x_1，Q 为椭圆曲线域参数 T 中的 p 值。

(4)下面给出的是寻找 GF（p）上的给定椭圆曲线 E 上的一个随机点（无穷远点除外）的算法。

输入：一个素数 P>3, GF(p)上椭圆曲线 E 的参数 a, b;

输出:E 上一个随机点(无穷远除外)。

a、随机选取 x, 0, <=x<p;

b、令$\partial\leftarrow x^3+ax+b\,\mathrm{mod}\,p$;

c、如果∂=0, 输出(x, 0)并停止;

d、利用模 p 的平方根方法求∂的一个平方根或判断它不存在;

e、如果(4)中的结果没有平方根存在，则返回(1)；否则(4)输出一个整数$\beta, 0<\beta<p$使得$\beta^2=\partial(\mathrm{mod}\,p)$;

f、生成一个随机比特μ，令$y\leftarrow(-1)^{\mu}\beta$；

g、输出（x，y)。

2、若#E(GF(p))=n=hk, k 是椭圆曲线 E 的阶中的大素因子(不小于 160 比特), 基点选择思想是：先在曲线上随机选择一个有效点，然后根据选择的点得到阶是 k 的有效基点。下面算法给出查找 E 上一个具有 k 阶的点, 即所用椭圆曲线群的基点。

输入:h, I, GF(p) 上椭圆曲线 E, #E(GF(p))=hk;

输出:E 上阶为 k 的点 G。

(1)在域 GF(p)上随机选择一个元素作为候选点的 X 坐标，用上面算法生成 E 上一个随机点 P(不是 0)；

(2)根据公式$y^2=x^3+ax+b$计算y^2。

(3)利用下面算法 3 计算y^2/p，若$y^2/p=1$继续该算法，否则转到(1)；

(4)利用下面算法 2 计算 y，标记 G=(x,y)，计算 kG。若 kG=0，可使用算法[25]，则 G 即为所求基点；否则继续使用算法 2；

(5)计算hG，如果 G=0，则返回(1)；否则$G\leftarrow hG$，G 即为所求基点。

隔

3、在 GF（p）上求解平方根

在 GF（p）上求解平方根问就是求解方程 $x^2 = a \bmod p$，其中 a 为 GF（p）上的二次剩余，目前有三种方法可以求解该方程。

(1)上面 2 中介绍的方法，其算法称为算法 1；

(2)利用分解多项式的方法求解方程；

(3)Schoof 利用椭圆曲线的求解方法。

算法 2：有限域 GF（p）上平方根的求解方法：

输入：有限域 GF（p），GF（p）上的二次剩余 a；

输出：x 满足 $x^2 = a \bmod p$。

(1)如果 $p \equiv 3 \pmod 4$，返回 $x = a^{(p+1)/4} \pmod p$；

(2)如果 $p \equiv 5 \pmod 8$，则

a、如果 $a^{(p-1)/4} \equiv 1 \pmod p$，返回 $x = a^{(p+3)/8} \pmod p$；

b、如果 $a^{(p-1)/4} \equiv -1 \pmod p$，返回 $x = a^{(p+1)/8} 2^{(p-1)/4} \pmod p$；

(3)若 $p \equiv 1 \pmod 8$，则

a、令 $p - 1 = 2^s t$，t 是奇数，$s > 2$；

b、在 F（p）上随机选择元素 u，使得 $u/p = -1$；

c、$3r \leftarrow u^t \pmod p$，$x_1 \leftarrow a^{(t+1)/2} \pmod p$，$i \leftarrow 0$，$s \leftarrow s - 2$，$k \leftarrow 0$，$z \leftarrow a^{-1} \pmod p$

c、对 k=0 到 k=s-2 循环做如下操作：

计算 $(x_1 z)^{2^{s-k}} \pmod p$；如果 $(x_1 z)^{2^{s-k}} \pmod p$=1，则 $x_1 \leftarrow x_1 r^{2^k} \pmod p$，否则下一步，$k \leftarrow k + 1$，

e、返回 $x = x_1$。

算法 3：

输入：奇数 $n \geq 3$，整数 a，$0 \leq a < n$；

输出：jacobi 符号 (a/n)，当 n 是素数时，该算法给出 Legendre 符号 (a/n) 的值。

(1)若 a=0，返回 0；

(2)若 a=1，返回 1；

(3)将 a 写成 $a = 2^c a_1$ 的形式，其中 a_1 为奇数；

(4)若 c 是偶数，$s \leftarrow 1$；

(5)若 c 是奇数，当 $n \equiv 1,7 \pmod 8$ 时，$s \leftarrow 1$；当 $n \equiv 3,5 \pmod 8$ 时，$s \leftarrow -1$；

(6)若 $n \equiv 3 \pmod 4$，并且 $a_1 \equiv 3 \pmod 4$ 时，$s \leftarrow -s$；

(7)$n_1 \leftarrow n \bmod a_1$；

(8)若 $a_1 = 1$，返回 s；否则返回 s-jacobi (n_1, a_1)。

在这些算法中，算法 1 是最基本被广泛使用的方法，它是专门用于求解此类方程的。

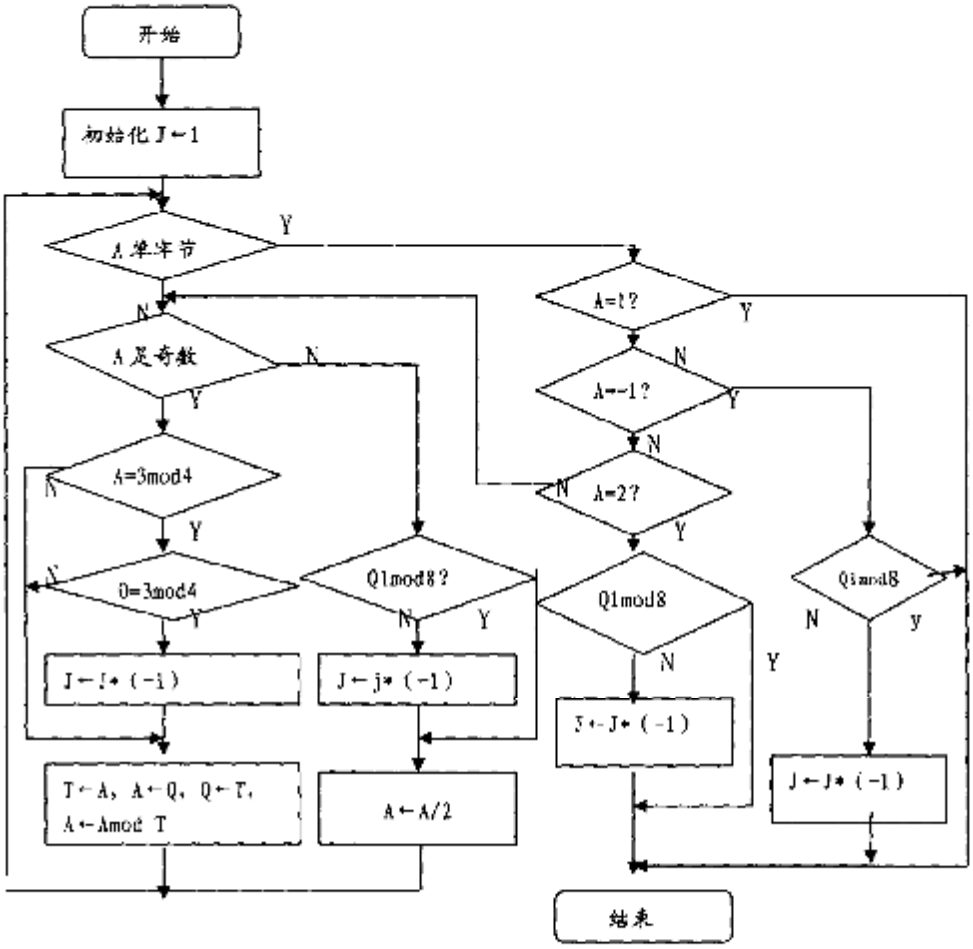

图 3-1 判别平方剩余的算法流程框图

第四章　ECC 快速算法设计与 ECC 实现

椭圆曲线密码系统的问题之二是计算速度慢。虽然 ECC 密钥短，比 RSA 等公钥密码体制速度要快，但实际上 ECC 的运行速度仍然很慢，与对称密码体制相差很远。因此，设计 ECC 的快速算法，提高 ECC 的运行速度，成为 ECC 研究的重要内容之一。

而加速 ECC 的算法可分为两个独立的步骤：一是加速底层域运算，寻找适当的基，提高域中乘积运算、逆运算的速度是加速底层运算速度的有效方法。目前一般采用的是多项式基和正规基，但速度的改变有限。对于具体的不同的域，可选基的自由度更大，可进一步优化计算，如优化正规基、高斯优化正规基等。二是加速群运算速度，寻找一般的或针对某些域和椭圆曲线的特殊方法，优化椭圆曲线的加法、倍乘运算等。常用的有：具有复乘的椭圆曲线、引进 Frobenius 同种及椭圆曲线间的同种方法，用于加速椭圆曲线的计算速度。

目前，加速椭圆曲线的各种算法中，不可能有一个共同的好的方案，且这种现象将会持续下去，而 $GF(2^m)$，$GF(p)$ 上快速算法的研究发展很不平衡，$GF(p)$ 上快速椭圆曲线的算法研究要更难实现一些。

4.1　点加的快速算法设计

设椭圆曲线 E 上两点 $P=(x_1,y_1),Q=(x_2,y_2)$，这两点之和为 $R=(x_3,y_3)$，即 $R=P+Q$，则计算 $R=P+Q$ 的算法如下：

if　$P=o$　then　output　$R\leftarrow Q$

if　$Q=o$　then　output　$R\leftarrow P$

if　$x_1=x_2$

then

if　$y_1+y_2=x_2$　then　output　$R\leftarrow o$

else

$\lambda\leftarrow x_2+y_2/x_2$

$x_3=\lambda^2+\lambda+a$

$y3\leftarrow x_2^2+(\lambda+1)x_3$

else

$\lambda\leftarrow(y_1+y_2)/(x_2+x_1)$

$x_3 \leftarrow \lambda^2 + \lambda + x_2 + x_1 + a$

$y3 \leftarrow (x_1 + x_2)\lambda + x_3 + y_2$

output　$Q \leftarrow (x_3, y_3)$

4.2　点的数乘快速算法设计

ECC 密码实现所面临的最大挑战是算法运算量大，而椭圆曲线的点积运算又是一个主要的运算，也是最耗时的一种运算。

1、点的数乘的基本算法

对椭圆曲线 $E: y^2 + xy = x^3 + ax^2 + b$，其中 $a, b, y \in F_q$，P、Q 为曲线上不为无穷远的两个点，P=(x_1, y_1)，Q=(x_2, y_2)，P 的逆元为 $-p = (x_1, y_1 + x_1)$。若 Q≠-P，Q≠P，则 P+Q=(x_3, y_3)，其中 $x_3 = \lambda^2 + \lambda + x_1 + x_2 + a, y_3 = \lambda(x_1 + x_3) + x_3 + y_1$，$\lambda = (y_2 + y_1)/(x_2 + x_1)$。当 Q=P=$(x_1, y_1)$，P+Q=2P=$(x_3, y_3)$，$x_3 = \lambda^2 + \lambda + a$，$y_3 = x_1^2 + (\lambda + 1)x_3, \lambda = x_1 + y_1/x_1$。设 $p \in E$ 为椭圆曲线 E 上的一个 t 阶点，K 为任意正整数，且 $0 \le K < t$，要提高椭圆曲线加解密速度，必须提高点的数乘 KP 的效率，因而先分析一般情况。

对 K，可唯一表示为 $K = a_{n-1}m^{n-1} + a_{n-2}m^{n-2} + \cdots + a_1 m + a_0, 0 \le a_j < m$，$a_j$ 是整数（j=0，1，2…，n-1），m 是大于 1 的正整数。

当 m=2 时，上式可表示为：

$K = a_{n-1}2^{n-1} + a_{n-2}2^{n-2} + \cdots + a_1 2^1 + a_0 2^0, 0 \le a_j < 2, j = 0,1,2,\cdots n-1$。因而有：

$$KP = (a_{n-1}2^{n-1} + a_{n-2}2^{n-2} + \cdots + a_1 2^1 + a_0 2^0)P$$
$$= a_{n-1}2^{n-1}P + a_{n-2}2^{n-2}P + \cdots + a_1 2P + a_0 P$$
$$= 2(2\cdots(2(O_k + a_{n-1}P) + a_{n-2}P)\cdots + a_1 p) + a_0 P$$

由椭圆曲线上倍点运算的定义可知：要计算 $2^i P$，只需计算 2P、2（2P）、…、2（2（…2P…）），共需 i 次倍点计算。且有 $a_i 2^i P = 2^i P, a_i = 1; a_i 2^i P = O_k, a_i = 0$，i=0，1，2，…，n-1。由此可得到 KP 的基本的点的数乘算法：

算法 1

```
Begin
T=O_k
For i=n-1 to 0 by -1
Begin
If a[i]≠0  then  T=2(T+P)
Else  T=2T
```

```
End
End
```

根据算法 1 计算点的数乘 KP，需要 n-1 次倍点运算及至多 n-1 次加法运算，故在 $GF(2^m)$ 域上要运行 7（n-1）次乘法及 2（n-1）次求逆。

2、 点的数乘快速算法

根据上面的基本算法[17]分析可知：KP 的运算速度取决于 n 的大小，如果能选取适当的正整数 S>1，且满足 n=SN，其中 N 为某一正整数，用 2^s 代替 2 构成整数 K 的基，这样 KP 式的长度将大大缩短，也就减小了加法运算次数。因此对上面式中取 $m=2^i, i>1$，则有

$K=a_{N-1}(2^s)^{N-1}+a_{N-2}(2^s)^{N-2}+\cdots+a_1 2^s+a_0, 0\le a_i<2^s, i=0,1,2,\cdots,N-1$。从而得到：

$KP=a_{n-1}(2^s)^{N-1}P+a_{N-2}(2^s)^{N-2}P+\cdots+a_1 2^s P+a_0 P$

$=2^s(2^s(\cdots(2^s a_{N-1}p+a_{n-2}P)\cdots)+a_1 P)+a_0 P, a_i\in\{0,1,\cdots 2^s-1\}$。

因而只需计算 2P，3P，…，（2^s-1）P。由椭圆曲线上加法运算定义可得：

2P=P+P，3P=2P+P，…（2^s-1）P=（2^s-2）P+P。由分析可知其共需 2^s-2 次加法运算，下面构造点的数乘 KP 的快速算法。

算法 2

```
1、begin
T[0]= O_y;
For i=1 to 2^s-1
T[i]=T[i-1]+P
End

2、begin
T=O_y;
For i=N-1 to 0 by -1
Begin
For u=1 to s
T=2T;
For j=1 to 2^s-1
If j=a[i] then T=T+T[j]
End
End
```

对算法 2 分析可知：对第 1 步需要 2^s-1 次点的加法运算，即要在 $GF(2^m)$ 域上做 3（2^s-1）次乘法及 2^s-1 次求逆运算。对第 2 步要做 N 次点的加法运算及 S×N 次倍点运算，即需在 $GF(2^m)$ 域上做（3+4S）×N 次乘法及（S+1）×N 次求逆运算。

通过对算法 1 和算法 2 的分析，可得到两种算法的运算步数如表 4-1 所示。

表 4－1　两种算法的运算步数比较

	加法运算（次数）	倍点运算（次数）
算法 1	n-1	n-1
算法 2	$2^s-1+n/s$	n

由表 4—1 可知，两种算法的倍点运算次数相差很小，当 n 较大时，可认为二者相等；区别仅在于加法运算中。下面分析在给定 K 后，s 取何值时，算法 2 的效率可达到最佳？为此可令 $f(s)=2^s-1+n/s$，对 f（s）求导，则可得：$f(s)'=2^s\ln 2-n/s^2$，令 $f(s)'=0$，由极值理论或得到 s 的最佳选取与 K 的二进制序列长度 n 满足的关系式：$n=2^s s^2\ln 2$。当 s=3，4，5，6 时，n 分别为 50，177，555，1597。即当 $40\le n\le 2000$ 时，s 可以取值此，4，5，6，使算法 2 达到最佳效果。

下面随机选取 6 个大整数进行计算，将结果列于表 4—2 中。从表中可知：所得结果与理论分析一致，算法 2 的效率比算法 1 的平均效率提高了 60%以上[ISI]。

表 4－2　大整数的运算

序号	K 的十进制序列长度	K 的二进制序列长度	S 值	K 的 2^s 进制序列长度	算法 1 加法运算次数	算法 2 加法运算次数	运算次数减少量
1	23	76	4	19	75	34	55%
2	49	160	4	40	159	55	65%
3	106	352	5	71	351	102	71%
4	120	394	5	79	393	110	72%
5	154	512	5	103	511	134	74%
6	180	594	5	119	593	150	75%

3、下面给出利用大数模乘思想设计的更有效的算法，概括如下：

输入：一个整数 n，一个椭圆曲线上的点 p

输出：椭圆曲线上的点 np

(1)　如果 n=0，则输出 0，停止。

(2)　如果 n<0，则让 Q=-P, k=-n，否则让 Q=?, k=n。

(3)　让 $h_m h_{m-1}\cdots h_1 h_0$ 表示满意 3k 的二进制形式，这里最高有效位 h_m 为 1。

(4)　让 $k_m k_{m-1}\cdots k_1 k_0$ 表示 3 的二进制形式。

(5)设 s=Q

(6)for I=m-1　down　to　1 do

s=2s;

if $h_i = 1$ and $k_i = 0$ then S=S+Q

if $h_i = 0$ and $k_i = 1$ then S=S-Q

(7) 输出 s。

简单地说，这是一个二进制加法运算算法，也就是说，先完成 2P、4P、8P……然后把这些积进行求和运算，使用这些运算，将大大减少所需要的运算时间。

4.3 椭圆曲线的标量乘法快速算法设计

椭圆曲线中最基本最重要的运算之一是标量乘法，即求点 P 的 K 倍。在椭圆曲线密码体制实现中，它的运算速度直接影响到整体速度。椭圆曲线上的点加运算和倍点运算均是为标量乘法服务的。

目前标量乘的算法有很多种，主要有：二进制方法、n 进制方法、带符号的二进制方法、带符号的 m 进制方法、改进的 m 进制方法、Montgomery 方法等。

1、带符号的二进制方法

在基于离散对数问题的公钥密码系统中，当底数 a 公开时，加解密实施的计算量集中于 a 的指数运算，即计算 $a^x \bmod p$ 的值，模 p 是一个大数。一般的计算方法是把整数 x 化为二进制形式，然后进行乘法和平方运算。

2、Montgomery 方法算法

输入：$k = (k_{m-1}, k_{m-2}, L, k_1, k_0)$，基点 $G = (x, y)$；

输出：Q=KG；

(1). $(X_1, Z_1, X_2, Z_2) = con\_projective(x, y)$，

其中 $X_1 = x, Z_1 = 1, X_2 = x^4 - b, Z_2 = x^2$；

(2). for I=m-2　down to 0 do

$$\begin{cases} Z_1 \leftarrow (X_1 Z_2 + X_2 Z_1)^2 \\ X_1 \leftarrow X_1 Z_2 X_2 Z_1 + x(X_1 Z_2 + X_2 Z_1)^2 \end{cases}$$

(X, Z)=mont_double(X, Z)

$$X = X^4 + bZ^4$$

$$Z = X^2 Z^2$$

(3). $Q = (x_Q, y_Q) = project\_con(X_1, Z_1, X_2, Z_2, x, y)$

$$x_Q = X_1 / Z_1$$

$$y_Q = ((X_1 / Z_1 + x)(X_2 / Z_2 - x) + x + y) \times (X_1 / Z_1 + x) / x - y$$

3、$GF(2^m)$域元素的乘法算法[19]

$GF(2^m)$域上的每一个元素都可以唯一表示为一个小于 m 阶的系数是 0 或 1 的多项式，设多项式 $a(x) = \sum_{i=0}^{m-1} a_i x^i$ 和 $b(x) = \sum_{j=0}^{m-1} b_j x^j$，可以通过移位_加的方法计算出 a(x)*b(x)的乘积。设不可约多项式为：

$f(x) = x^m + f_{m-1}x^{m-1} + \cdots + f_2 x^2 + f_1 x^1 + f_0$，$f_i \in \{0,1\}$。

下面是域元素的乘法算法：

输入：小于 m 二进制多项式 a(x)和 b(x)

输出：c(x)=a(x)*b(x) mod f(x)，f(x)为不可约多项式。

(1) if $a_0 == 1$ then c=b else c=0

(2) for I=1 to m-1 do

a: b=b*xmodf(x)

b: if $a_i == 1$ then c=c+b

(3) return c

很显然上述算法的计算时间依赖于 a(x)二进制位串中 1 的个数。

4、几种算法的比较[20]

表 4-3　模为 512 的不同算法运行数比较

采用的算法	平均运行数	最大运行数
常规二进制方法	766.5	1022
带符号二进制算法	681.7	768
基于数据压缩的分段算法	635.1	
常规分段窗口法(W=5)	609.3	630
带符号的二进制窗口法	602	629

4.4 快速模算法设计

为保证有限域 F_p 上运算的封闭性，所有基本运算的结果均要对 p 取模，故设计高效的取模算法对提高所有基本运算的速度至关重要。而提高取模运算速度的常用算法是除余数法和累加法。但由于有限域中大整数的模运算的运算量很大，故所用方法

运算速度太慢而不可行。而模余数法是目前比较有效的取模算法，其基本方法是利用有限域F_q确定时取模运算 x mod p 在模相同的性质，预计算$2^i \bmod p$的结果造表备用。

取模运算时先将被模数分解为$\sum_{i=0}^{2n-1} c_i 2^i$，只要对不为 0 的各$c_i$，在模结果中累加上$2^i \bmod p$所预计算值即可。本算法的基本思想是预计算，将预计算的结果造表待用，以空间换取时间，因而不再需要在调用时临时计算。

设 A=$\sum_{i=0}^{n-1} a_i B^i, a_i \in [0, B]$，则$A = \left[\sum_{i=0}^{n-1} a_i B^i + \left(a_{n-1} B^{n-1} \bmod M\right)\right] \bmod M$；

令 M=$\sum_{i=0}^{n-1} m_i B^i, m_i \in [0, B]$，则：$R_1^{(j)} = j \cdot B^d \bmod M$，也可定义为$\sum_{i=0}^{d-1} r_{1,i}^{(j)} B^i$，其中 j=1，2，3，…，B-1；$R_1^{(j)}$为 M 的 d+1 阶模表，记为$R_1$，如表 4—4 所示。

表 4－4　M 的 d+1 阶模表

j	R_1	$\sum_{i=0}^{d-1} r_{1,i}^{(j)} B^i$
1	$B^d \bmod M$	$\sum_{i=0}^{d-1} r_{1,i}^{(1)} B^i$
2	$2B^d \bmod M$	$\sum_{i=0}^{d-1} r_{1,i}^{(2)} B^i$
…	…	…
B-1	$(B-1)B^d \bmod M$	$\sum_{i=0}^{d-1} r_{1,i}^{(B-1)} B^i$

设 M=$\sum_{i=0}^{n-1} a_i B^i, a_i \in [0, B]$，则：$R_0^{(j)} = j \cdot B^{d-1} \bmod M$，也可定义为$\sum_{i=0}^{d-1} r_{0,i}^{(j)} B^i$，其中 j=1，2，3，…，B-1；$R_0^{(j)}$为 M 的 d 阶模表，记为$R_0$，如表 4－5 所示。

表 4－5 M 的 d 阶模表

j	R_1	$\sum_{i=0}^{d-1} r_{0,i}^{(j)} B^i$
1	$B^{d-1} \bmod M$	$\sum_{i=0}^{d-1} r_{0,i}^{(1)} B^i$
2	$2B^{d-1} \bmod M$	$\sum_{i=0}^{d-1} r_{0,i}^{(2)} B^i$
…	…	…
B-1	$(B-1)B^{d-1} \bmod M$	$\sum_{i=0}^{d-1} r_{0,i}^{(B-1)} B^i$

对上面定义的 A 和 M，

(1) 若$a_{d-2} < m_{d-1}, A \bmod M = A$；

(2) 若$a_{d-1} = m_{d-1}$，则当 M>A，则 A mod M=A；当 M=A，则 A mod M=0；当 M<A，则 A mod M=A-M。

(3) 若 $a_{d-1} > m_{d-1}$，则当 M>T 时，A mod M=T；当 M<T 时，A mod M=T-M。

其中 $T=\sum_{i=0}^{d-2} a_i B^i + R_0^{(a_{d-1})}$。

1、快速模算法设计[21]

设计快速模算法，它由建表、初步模和同阶模三步完成。

(1)建表

先构造 M 的 d+1 阶和 d 阶模表，可采用递推法，可在每次执行加、解密任务时临时进行，效率很高。而建表必须在任何一次取模之前完成，同时要考虑到表所占内存的大小，可依下列顺序进行建表：

① $(A\times 2^n + B)(\mathrm{mod}\,N) = [A\times 2^n(\mathrm{mod}\,N) + B](\mathrm{mod}\,N)$

② $(A_1\times 2^n + A_2)(\mathrm{mod}\,N) = [A_1\times 2^n(\mathrm{mod}\,N) + A_2(\mathrm{mod}\,N)](\mathrm{mod}\,N)$

③ $(A\times 2^n + B)(\mathrm{mod}\,N) = [A\times (2^n \bmod N) + B](\mathrm{mod}\,N)$

规则①指建表过程中可忽略掉 X 中比 N 小的部分；规则②指将表分为多个部分；规则③指能使各种待处理的数重复使用该表，但要结合前面二个规则使用，以减小表的大小。

(2)初步取模

设 A 为 N 位的 B 进制数，M 为 D 位 B 进制数，则初步取模的结果为一个 D 位的 B 进制数，可能大于 M，也可能小于 M。算法描述如下：

```
Modular1 (A, M, R)
 { for i=n-1 to d    do
 {a: -a_i ;
 a_i=0;
 A=A+B^(n-1-d) R_1^(a)
 }
 }
```

最终取模的结果放在 A 中，为一个 d 位的 B 进制。

(3)求同阶模

对上一步求得的结果进一步求模，由上面讨论可得到最终结果，其算法如下：

```
Begin
Modular2(A,M,T)
For j=0 to d-1 do
If a_(d-1) < m_(d-i)
Then A mod M =A
```

If $a_{d-1}=m_{d-1}$

Then

If M>A

then A mod M =A

If M=A

Then A mod M =0

If M<A

Then A mod M =A-M

If $a_{d-1}>m_{d-1}$

Then if M>T

Then A mod M =T

Else A mod M =T-M

End

2、模加算法

设 A、B、C∈F_p，C=A+B mod p，则模加算法如下：

{if A+B<=p

then C=A+B;

else C=A+B-p}

3、模乘算法

模乘可分为求积和取模两步求解：设 A、B∈F_p，$C\equiv A*B(\mathrm{mod}\,p)$，则先求出 A 和 B 的乘积；再利用快速求模算法对乘积取模即可得到模乘的结果。

4、模逆算法

已知$a\in F_p$，gcd(a,p)=1，，满足等式 aX-1 mod p 的 X∈F_p称为有限域中的逆，可表示为$a^{-1}\,\mathrm{mod}\,p$。求模逆的算法有很多种，以 Euclid 算法最为简捷，算法如下：

begin

b:=p;$u_0=0;u_1=1$;

while (b<>0) do

begin

s:=a div b; r:=a mod b;u:=u_0-s*u_1;

a:=b; $u_0=u_1;u_1=u$;

end

if ($u_0 >= 0$)　then return　u_0

else　return　$u_0 + p$

5、模幂算法[22]

当讨论算法的复杂度时，一般是分析算法执行各种操作的次数，可以得到有用的结果；然而有时还需考虑更多的因素。如对于大整数模幂，它的程序如下：

功能：进行大整数的模幂乘

输入：一个大整数的模 m，一个指数 e，一个底数 x

输出：$x^e \bmod m$

```
Vlong modexp (const vlong &x,const vlong &e.const vlong &m)
{
Unsigned I,n=e.bits();
Vlong xt,result=1;
Xt=x;
For (in=0;i<n;i++)
{if (e.bit(i))
Result=(result*xt)%m;
Xt=(xt*xt)%m;
}
Return result
}
```

从该程序来看，若把模 m 的操作放在最后返回函数的语句里，而把循环体内的模运算去掉，这样做就减少了整个算法做除法运算的次数，好像能提高算法的效率。事实上，由于长整数是可以任意长的，xt 不断地做平方，规模越来越大，结果便得做乘法运算耗费更多的时间。相反，每做一次乘法，就做一次模运算，能控制下一次乘法运算的规模，结果是节约的时间远大于模运算所用的时间。下面给出其测试程序：

```
Void disp (vlong &x)
{
Char buf[255]
Unsigned n=x.store(buf,255);
For(unsigned i=0;i<n;i++)
{printf("%C",buf[i]);}
Printf("\n");
}
```

```
Int main(int argc,char**argv)
{
Unsigned ut1,ut2;
Time_tt;
 t= time(null);
 ut1= (unsigned)t;
 char*chv=" 14859933",
 *che=" 3948",
 *chm=" 19999997" ;
 Vlong v, e, m, rz;
 v.load(chv,8);
 e.load(che,4);
 m.load(chm,8);
 rz= modexp(v,e,m);
 disp(rz);
 t= time(Null);
 ut2= (unsigned)t;
 printf( "seconds:%u\n",ut2-ut1);
 getch();
 return 0;
}
```

对该测试程序，在同一执行环境下，当 modexp 函数中的“xt=（xt*xt）%m”改为“xt=xt*xt”后所用的执行时间为 148 秒，而没改变时所用时间还不到 1 秒。

下面给出一个计算模幂的算法：

```
{Y=X;
 for(I=n-1;I>=0;I=I-1)
 {Y = Y^2 * X^m mod p
  }
  return    y;
  }
```

4.5 椭圆曲线密码系统的实现

非椭圆曲线域上的密码体制，明文只需稍加变换就可应用到加密和签名中；而椭圆曲线密码体制必须把明文转换到椭圆曲线上的点才能用于加密和签名过程，而这种转换可以通过可以通过编码来完成。

曲线密码系统的基本原理是：设选定的椭圆曲线为E（F_q），明文消息为m，编码方法为$\pi: m \to p_m \in E(F_q), N = \#E(F_q)$，选取公、私密钥，对$p_m$应用E（$F_q$）上的算法进行加密等操作。在另一端，得到$p_m$，再进行反编码：$\pi^{-1}: p_m \to m$。理论上，ECC与经典公钥密码仅明、密文空间（基域）之差；实际上，ECC的明文编码却成为一个难题，也就是说，还没有一种一般的、有效的编码方法，将明文嵌入到椭圆曲线上。N.koblitz应用二次扩域的性质，提出了一种明文编码方案，但此方案有很明显的局限性。后来，N.koblitz提出了一种比较一般的编码方法—概率嵌入法。

1、 私钥和公钥以及共享密钥的产生

若A和B要进行通讯，首先得到椭圆曲线E、点P及素数N。然后用户A将[1,N-1]中随机选取的整数a作为私钥，A将KpubA=a×P作为自己的公钥传送给用户B，与此同时B将[1,N-1]中随机选取的整数b作为私钥，并将KpubB=b×P作为自己的公钥传送给A。A、B各自将自己的私钥点乘于对方传过来的公钥得到K_{AB}，这样就完成了密钥的交换过程。当用户A需要将待传数据m传送给用户B时，A利用m和K_{AB}生成E_m，当用户B得到E_m后，利用密钥交换过程自己生成的K_{AB}和从用户A处得到的加密数据E_m生成数据m。目前大多使用该方法实现ECC。

设有用户A 和B，则它们的私钥和公钥以及共享密钥按如下步骤产生：

(1)、用户A随机地选取一整数s_A作为自己的私钥，计算$P_A = s_A G$（点积运算）作为自己的公钥

(2)、用户B 用类似的方法产生自己的私钥和公钥s_B和P_B；

(3)用户A 产生共享密钥$K_A = s_A P_B$，用户B产生共享密钥$K_B = s_B P_A$，显然有

$$K_A = s_A P_B = s_A(s_B G) = s_B(s_A G) = s_B P_A = K_B$$

2、 明文映射到椭圆曲线上

在对明文M加密之前，需将M映射到椭圆曲线的有限域F_q上的一个点上；若M较长，可分段处理。值得注意的是不能简单地将M映射到椭圆曲线的任意点上，而必须是F_q域中的点，即椭圆曲线上满足平方剩余的点。映射的方法有多种，映射方法并非唯一，本文采用的映射方法[20]如下：

首先对M进行分段处理，设m 为M的一个分段，m 满足条件：

$$0 \le m \le [p/256] - 1$$

将m 映射到点$P_m(x,y)$上，使得：$256m \le x \le 256(m+1)$，$p_m(x,y) \in F_q$，找点$P_m(x,y)$并不困难，因为当$256m \le x \le 256(m+1)$时，$y^2 = x^3 + ax + b \pmod p$是一个非平方剩余的概率很小。

3、加密

明文分段m 映射到点$P_m(x,y)$不是加密，仅仅是一种编码，任何用户都可以通过解码将$P_m(x,y)$恢复成m。因此，$P_m(x,y)$实际上可视为明文m，在发送$P_m(x,y)$前，需对其进行如下加密处理得到密文$c_m = p_m + s_A p_B$。

4、解密

接收端收到密文c_m后，进行如下处理恢复明文p_m：

$p_m = (p_m + s_A p_B) - s_B p_A$。

5、解码

接收端得到$P_m(x,y)$后，取出p_m的X 坐标值x，再用下式得到明文m:

$m = [x/256]$

5、椭圆加密体制实现

迄今所投入使用的椭圆加密系统中，绝大部分的密钥长度都比较短，一般集中在30—60位，这是因为在软件实现时，由于软件执行速率所限，密钥长度比较大(=160)的椭圆加密系统的速率将达不到使用要求。在硬件实现时，密钥长度比较大的椭圆加密系统将耗费大量的硬件资源。根据以上椭圆加密体制的要求，设计出图4-1的加密系统结构图，其中椭圆加密系统参数接口获取与加密有关的椭圆的基本参数，如私钥、椭圆曲线、椭圆曲线上的给定点等。软件实现的结构如图4-1所示。在软件验证系统实现的过程中，有限域上的加法是异或操作，有限域上的乘法和求逆是关键点。在此系统中使用了复合域带来的特殊性，可以高效、快速的实现乘法和求逆运算。

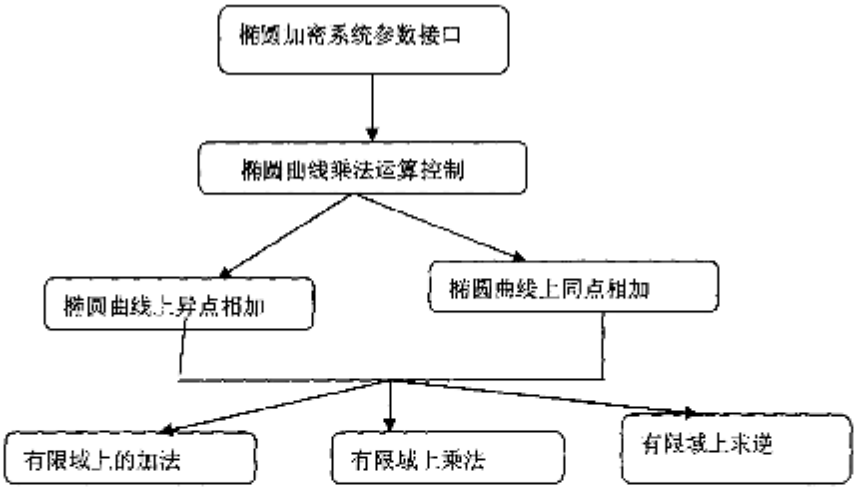

图 4-1 椭圆曲线加密系统结构图

第五章　椭圆曲线密码系统的安全性

5.1　ECC 的安全性分析

椭圆曲线密码系统是建立在求椭圆曲线离散对数问题的困难性上。据说，其难度要大于因数分解和一般离散对数问题。因此，ECC 是目前已知安全性最强的公钥密码系统。

虽然椭圆曲线点运算的概念易于理解，但要产生合适的符合安全性条件的椭圆曲线和有效点乘运算的方法却是很复杂的。符合安全性条件的合适的椭圆曲线一旦产生，即可形成一椭圆曲线群，并可被众多的用户所公用。用户可基于此群生成其公/私密钥对。根据 Hasse 定理：$p+1-2\sqrt{p} \le \#E(F_q) \le p+1+2\sqrt{p}$，其中 $\#E(F_q)$ 表示椭圆曲线群上点的总数，即可求出椭圆曲线上点的总数范围。虽然可用 Schoof 算法计算出 $\#E(F_q)$ 的精确值，但其过程相当复杂。由于椭圆曲线上符合安全性条件的点的有限性和点的数目的难确定性，而这些曲线包含了一些离散的点，使攻击者难以将这些点连成曲线，也就无法确定其几何关系。正是由于这种属性，使椭圆曲线密码系统的安全性更高。

由表 5－1 可知：当素数 P 按位数增加时，椭圆曲线 $E(F_q)$ 可取的条数大大增加，增加的速度远远超过素数 P 的增加速度。当 P=997 时，F_q 上的椭圆曲线为 992020 条，而在实际应用中选取的素数非常大，位数不会低于 30 位(十进制)。亦即在一个确定的有限域群上，椭圆曲线非常多，则可取的椭圆曲线非常丰富，完全有条件选取既利于加密计算，又不利于攻击者反算的椭圆曲线，这对椭圆曲线密码系统的安全性非常有利[23]。

表 5－1　椭圆曲线在有限域上的数据

素数位数/位	最大素数 P	E 的条数/条
1	7	30
2	97	9181
3	997	992020

一般而言，椭圆曲线上的点构成的群 G 上的离散对数算法可分为二类：即指数计算法和碰撞搜索法。目前最好的指数计算法是线性筛法和数域筛法；最有效的碰撞法是 Pllord--rho 方法[24]、Pohlig--hellman 方法[25]和 Xedni 计算法[26]。这些也是目

前对 ECC 最有影响的方法。

实现 ECC 时主要要解决两个方面的问题：一是如何选取合适的符合安全条件的随机椭圆曲线；二是如何快速实现椭圆曲线密码。而加密算法的安全性一般是通过该算法的抗攻击强度来反映。

5.2　对椭圆曲线的攻击

椭圆曲线上密码系统的安全核心是椭圆曲线上离散对数问题（ECDLP）。对一般的子群 DLP 问题，已有亚指数算法；对于 ECDLP，到现在的研究表明：一般不存在亚指数算法。但是，对不同的情况，也存在不同的攻击算法。ECC 是目前最流行的公钥密码，所以对它的攻击也成了当前密码学研究中的一个热点。

设 $E(F_q)$ 是定义在有限域 F_q 上的椭圆曲线，P、Q 是 $E(F_q)$ 上的两个点，如果存在 l，满足 $0 \le l \le n-1, lP = Q$，那么 l 称为以 P 为基的 Q 的离散对数。

5.2.1　对任意椭圆曲线的攻击[27]

1、穷举搜索法

该方法就是简单地连续计算 P 的整数倍：P，2P，3P…直到得到 Q 为止，最坏情况下需要计算 n 步。这里 n 为 p 的秩。

2、大步小步法

设 P 为椭圆曲线 E 上的点，P 的阶为 n，已知点 $S \in \langle p \rangle$（P 生成的循环群），求正整数 l，使得 $S = lP (0 \le l \le n)$，将 l 表示为 $l = c\left\lceil n^{1/2} \right\rceil + d$，令 $R_d = S - dP$，存储关于 R_d 的表，对于 c=0，1，2…，$\left\lceil n^{1/2} \right\rceil - 1$，依次计算 $R_c = c\left\lceil n^{1/2} \right\rceil P$，将 R_c 与 R_d 表中的点比较，若某个 R_c' 与 R_d' 相同，则有 $l = c'\left\lceil n^{1/2} \right\rceil + d'$。

该算法的复杂度为 $o\left(n^{1/2}\right)$，是已知计算 ECDLP 的最好的复杂度，但该算法需存储约 $n^{1/2}$ 个点，是一种以空间换时间的方法。

3、Pohlig—Hellman 算法

该算法是由 Pohlig 和 Hellman 提出的。

设 $n = p^c$ 为素数幂，令 $s_0 = p^{c-1}s = l\left(p^{c-1}p\right) = lp_0, p_0$ 的阶为 p，计算 n=p 的 ECDLP 就可得到 $l \equiv l_0 \bmod p$。设 $l = l_0 + l_1 p$，则 $s_1' = s - l_0 p = l_1(pp) = l_1 p_1'$，$p_1'$ 的阶为 p^{c-1}，采用类似的方法，令 $s_1 = p^{c-2}s_1' = l_1\left(p^{c-2}p_1'\right) = l_1 p_1$，$p_1$ 的阶为 p，计算该 ECDLP 得到

$l_i \bmod p$,从而得到$l \bmod p^2$。重复上述方法,可依次得到$l \bmod p^i (i=1,2,\ldots c)$。

它使用了椭圆点P的阶n的因数分解,将求解l的问题简化为求解l模除n的每个素因子的问题,然后通过中国剩余定理同余方程组得出要求的l。当n全是小素数因子的乘积时,这种算法是非常有效的。故为了抗击这种算法的攻击,n必须有大素因子。

4、Pollard—rho算法

该算法是由Pollard提出,是大步小步算法的随机版。运行时间与小步大步法差不多,但它比小步大步法优越之处是它的预存储空间的需求可忽略。

算法的基本思想是:将阶为素数n的椭圆群(循环群)G通过一些简单的方法分为大致相等的三部分:A_1, A_2, A_3。随机生成两个整数a_0和b_0,其中$1 \le a_0, b_0 \le n-1$。

Pollard rho-方法是一个随机算法,无法给出运算时间上限,只能给出其期望运算次数。它的计算需要$\sqrt{\pi(n/2)}$步,每一步是一个椭圆曲线加运算。表5—3给出了对于有效值n使用Pollard—rho方法计算一个离散对数时的计算能力。

表5－3　有效值n在使用Pollard—rho方法时的计算能力

域的大小/位	n的大小	$\sqrt{\pi(n/2)}$	MIPS年
163	160	2^{80}	9.6×10^{11}
191	186	2^{93}	7.9×10^{15}
239	234	2^{117}	1.6×10^{23}
359	354	2^{177}	1.5×10^{41}
431	426	2^{213}	1.0×10^{52}

Pohlig和Hellman发现,Abel群的阶的平滑性对于G中求解离散对数问题是有帮助的。借助中国剩余定理和相关变换,算法的时间复杂度为$O\left(\sqrt{e_p}\max(e_p, p)\log(p_{\min}(e_p, p))\right)$。

5、并行Pollard—rho算法

Van Oorschot和Wiener提出Pollard－rho算法的并行算法,当该算法并行运行在s个处理器时运行时间约为$\sqrt{\pi n/2s}$步,也就加速了$\sqrt{s}$倍。该算法是目前已知的求解ECDLP的最快的算法,最新的求解成功的ECDLP问题几乎都是采用该算法。

6、pollard—Lameda 算法

这是 Pollard—rho 算法的另一个随机性算法。它是 Pollard—rho 提出的，同 Pollard—rho 算法一样，该算法可以并行执行。但并行的 pollard—Lameda 算法比并行化的 Pollard—rho 算法稍慢，但如果已知对数 d 的取值在区间[0,]中时，pollard—Lameda 算法要更快上些，而 b<0.39n。

7、倍对数算法

R.Silverman 和 Stapleton 发现：如果一个 ECDLP 的实例被 Pollard rho 算法解决，那么对同一 EC 和基点，其它实例的求解可以被加速。如第一个实例需要时间 t，第二个实例需要时间 $\left(\sqrt{2}-1\right)t \approx 0.41t$，第三个实例需要时间 $\left(\sqrt{3}-\sqrt{2}\right)t = 0.32t$，…这样 ECDLP 的求解将变得越来越容易。因此，对 ECDLP 的应用中应适当选择 EC 的参数，使第一个实例求解是困难的。

8、Xedni 计算法

Xedni 计算法是 Silverman 在 1998 年提出的一种新的攻击方法，对于椭圆曲线上的离散对数问题，指数计算法是不能用的，但将指数计算方法的顺序倒过来考虑，于是发现了这种攻击方法。据 Koblitz 等人的讨论，Xedni 计算法成功成功的概率非常小，且它的计算量是指数时间的。

上述多种算法关于 n_A 或其最大素数因子的长度是纯指数复杂度的。当 n_A 含较大素因子时(如长度大于等于 160 位)时失效，这对椭圆曲线密码系统构不成真正的威胁。根据分析，椭圆曲线密码系统的安全性依赖于椭圆曲线离散对数问题的难解性，为保证密码体制的安全性，所使用的基本阶应含长度不小于 160 位的素因子。事实上，自 1997 年以来，椭圆曲线密码体制及安全性分析引起了密码学界及各界的极大关注，并成为研究热点。

5.2.2　对特殊椭圆曲线的攻击

1、MOV 攻击

Menezes、Okamoto 和 Vanstone[28]在 1991 年发表了将 ECDLP 归约到有限域上离散对数的有效解法，即为 MOV 归约。

设 E 是定义在有限域 F_q 上的椭圆曲线，F_q 的特征标是 p，$\overline{F}_q$ 是 F_q 的代数闭包，p 是曲线 E 上阶为 t 的点，设 t 是不同于 p 的素数，曲线 E 的 t 挠点的集合(即 t 倍点为无穷远点的点)记为 $E[t]=\{P\in E|tP=0\}$，weil 对是一个函数 $e_t: E[t]\times E[t]\to \overline{F}_q$，并具

有多个性质。

归约算法为：

输入：E 上阶为 t 的点 P，另一个点 Q；

输出：一个整数 l，$0 \le l \le t-1$，使得 $Q = lP$。

①确定 k，使得 $E[t] \subseteq E(F_q^k)$；

②找 $S \in E[t]$，使得 $\alpha = e_t(P,S)$ 是 t 阶元素；

③计算 $\beta = e_t(Q,S)$；

④计算 β 关于 α 在 F_q^k 中的离散对数 l；

⑤输出 l。

该算法主要用于超奇异椭圆曲线攻击，对其它椭圆曲线不适应。而超奇异椭圆曲线是指 F_q 的特征标是 p 整除 q+1 的（F_q）的曲线。当 E 是超奇异椭圆曲线时，扩域次数 $k \le 6$。

2、FR 攻击

Frey，Muller 和 Ruck[28]利用 Tate 对，类似于 MOV 攻击给出了椭圆曲线离散对数的一个攻击算法。利用 MOV 攻击时，首先要找到 k，使得 $E[t] \subseteq E(F_q^k)$；在 FR 攻击中，仅要求 $t \mid q-1$，并不要求 k，使得 $E[t] \subseteq E(F_q)$，故 FR 攻击一般要比 MOV 攻击的适应范围广，且 FR 攻击效率也较 MOV 攻击高。

3、SSAS 攻击

1997 年 Smart 和 Satoh 及 Araki 同时独立地提出了对素域上某一类称为非正规椭圆曲线的离散对数的攻击法。不久，Semaev 也得到了同样的结果，故简称为 SSAS 攻击。

若有限域 F_q 上的一条椭圆曲线的阶恰好为 q，则称该椭圆曲线为非正规椭圆曲线。Semaev 有超越函数域的方法给出了 ECDLP 的攻击算法；Smart 用椭圆对数给出了另一种求非正规 ECDLP 的攻击算法；Satoh 及 Araki 也得到了类似的结果，即 ECDLP 在非正规椭圆曲线上是容易解的；而对于其它类型的椭圆曲线 SSAS 算法是失效的。故在构造安全椭圆曲线时避开椭圆曲线上的点的个数等于有限域中元素个数的情况就可避免 SSAS 算法的攻击。

4、Weil 下降攻击

Gaudry，Hess 和 Smart[30]基于 Frey 的 Weil 下降理论，对 $GF(q^n)$ 上的椭圆曲线提

出了一种新的攻击算法，此处$q=2^r$。通过Weil下降的思想将扩域上椭圆曲线$E(F_q^n)$离散对数问题归约到子域代数簇$J_\infty(c)(F_q)$上的离散对数问题，然后用指标计算法解$J_\infty(c)(F_q)$上的离散对数问题，该算法的基本思想为：

设r是一个有限域，R是它的扩域。椭圆曲线E(R)定义在R上，先构造一个定义在r上的Abel簇A(r)，然后找到一条定义在k上的曲线C(有可能奇异)，由Jacobians的性质得到映射$\varphi: J_\infty(C') \to A$，其中$C'$是C的正规化曲线。

设P_1, P_2是E(R)的两个点，且有$P_2 = lP_1$；则P_1, P_2就对应A(r)上的两个点，由映射φ对应到$P_{r_1}^{m}(C)(r)$的两个除子D_1, D_2，且$\varphi(D_i)=P_i$，最后使用指标指标计算法解$P_{r_1}^{m}(C)(r)$上的离散对数问题$D_2 = lD_1$。

用Weil下降算法必须解决二个问题：(1)必须找出A上只有小亏格的曲线；(2)必须找出A上的点所对应的C的除子；(3)必须找出解决广义除子类群离散对数问题的指标计算方法。

Weil下降方法只适应于特征2的复合域上，并且当亏格比较大时，解决上述三个问题几乎是不可能的；而当亏格较小时，如$g<4$，对这样的超椭圆曲线离散对数问题的攻击，目前最好的是Pollard并行攻击算法。Weil下降方法可以将椭圆曲线离散对数问题规约成超椭圆曲线离散对数问题，而对于高亏格的超椭圆曲线离散对数问题是有亚指数时间攻击算法的[34]。

在实际中所用的特征为2的ECC大都是$GF(2^p)$上的，且p为素数，Weil下降方法是不能用于这类域上的椭圆曲线的，故这一攻击对当前实现的ECC并未造成太大的威胁。

5.2.3 硬件攻击

一个充满前景的对椭圆曲线的攻击是建立一个使用Pollard—rho算法进行并行搜索的专用硬件。Van Oorschot和Wiener对这种攻击的可能性做了详细说明，预测如果有r=325000个处理器，则解决一个$n \approx 10^{36} \approx 2^{120}$的ECDLP问题大约需要35天。而ANSI X9.62要求$n > 2^{160}$，以目前的技术进行硬件也是不可能成功的。

若一台每分钟执行1百万条指令的计算机每秒能执行4×10^4次椭圆曲线加法运算，则一年内可执行约2^{40}次椭圆曲线点加运算，一个MIPS表示一台每分钟执行一百万条指令的计算机工作一个。表5－4给出了对不同的n，使用Pollard—rho算法计算一个单一的离散对数所需要的能力[32]。

表 5-4　计算离散对数所需的计算能力

域长/bit	N 长/bit	$\sqrt{\pi n/2}$	MIPS 年
163	160	2^{80}	9.6×10^{11}
191	186	2^{93}	7.9×10^{15}
239	234	2^{117}	1.6×10^{23}
359	354	2^{177}	1.5×10^{41}
431	426	2^{213}	1.0×10^{52}

5.3 对椭圆曲线的挑战

自 Neal koblitz 提出椭圆曲线在密码学中的应用以来，马上引起了密码学界和企业界的极大关注，同 RSA 的安全性验证一样，ECC 的安全性验证也很快开展起来。

5.3.1 Carticom ECC 挑战

Certicom 是一家专门从事安全解决方案的加密技术公司，继发起 RSA 挑战以后，1997 年 11 月发起了一项对椭圆曲线密码的挑战，其定义如下：设 F_q 为有限域，ECCp-m 表示 F_q 上的 ECC，ECC2-m 表示 F_{2^m} 上 Koblitz 椭圆曲线上的密码系统，其中 m 表示密钥的比特数。假定 $n=\#E(F_q)$。

Certicom ECC 挑战的内容是[33]：给定 EC 及其参数、公钥，寻找 ECC 的私钥。挑战分三个部分：

1. ECCp-79，ECCp-89，ECCp-97，ECCp-109，ECCp-131，ECCp-163，ECCp-191，ECCp-239，ECCp-359；

2. ECC2-79，ECC2-89，ECC2-97，ECC2-109，ECC2-131，ECC2-163，ECC2-191，ECC2-238，ECC2-359；

3. ECC2k-95，ECC2k-108，ECC2k-130，ECC2k-163，ECC2k-238，ECC2k-358。

ECC 挑战分为两个水平：一级水平为 ECC-109，ECC-131；二级水平为 ECC-163，ECC-191，ECC-239，ECC-359 等。Certicom 认为：ECC-109 可以在几个月内解得，ECC-131 将要更多的资源。二级水平被认为是困难的。

下面是 ECC 挑战的情况：

ECCp 79，1997 年 12 月 6 日解决；ECC2-79，1997 年 12 月 16 日解决。ECCp-89，1998 年 1 月 12 日解决；ECC2-89，1998 年 2 月 7 日解决。ECCp-97，1998 年 3 月 16 日解决；A. Escott 等应用 Pollard rho 算法完成了 Certicom 的 ECCp 97，大约进行了 2×10^{14} 次 EC 加法，这个数字与期望时间很接近。由此 Escott 估计：求解 ECCp-109

大约需要 5 万台 Pentium Pro200MHz 计算机 3 个月的计算时间。ECC2k-95，1998 年 5 月 21 日解决；ECC2-97，1999 年 9 月 22 日解决。

2000 年 4 月，爱尔兰数学家 Robert Harry 和法国的研究人员一起成功地破译了键长 109bit 的椭圆曲线密码（ECC）。被破译的密码是由密码大户加拿大的 Certicom Corp 开发的名为“ECC2K-108”。上述研究人员利用该公司的公共键清单及系统参数，参加推导秘密键的“ECCChallenge”活动，最终取得了成功。推导出秘密键需要进行庞大的运算，为此研究人员使用联网的 9500 台电脑进行了长达 4 个月的运算。有 40 个国家的 1300 名志愿者参加了这一活动。其运算量相当于配置 450MHz 微处理器的个人电脑运算 500 年的运算量。由于 ECC 公开键密码键长比事实标准 RSA 密码更短而且又能确保安全特性因而备受注目。这次需要的运算量相当于解开 600bit 的 RSA 密码所需要的运算量。这一结论支持了 Certicom 关于 ECC 密钥至少 160bit 的观点。据估计，求解 ECC-163 的工程大约是 ECC-108 的 10 亿倍。

5.3.2 因子分解竞赛

EC 在因子分解算法上的表现不俗，1985 年 Richard Brent 预言：应用椭圆曲线的方法可以找到 50 位十进制的因子。1995 年 Peter Montgomery 分解了 $5^{256}+1$，得到了一个 47 位十进制数因子；1997 年 10 月，Richard Brent 分解了 $24^{121}+1$，得到了一个 48 位的十进制数因子；1998 年 9 月 14 日，Conrad Curry 应用 mprime 程序找到了 $2^{997}+1$ 的一个 53 位的十进制数因子。实现了 Richard Brent1985 年的预言。

新的记录是 1999 年 12 月 Nik Lygeros 的 Michel Mizony 创造的，他们应用 Paul Zimmermann 的 GMP-ECM 程序找到了一个 54 位的素因子。近几年的记录如表 5－5。其次还有对椭圆曲线上点的计算的挑战。

表 5－5　　用 EC 进行素因子分解

位数	大数	日期	完成人
46	$115\times9^{115}-1$	2001-07-23	P. Leyland
46	$10^{1227}+1$	2001-02-28	A. Kruppa
46	$12^{291}-1$	2001-08-20	D. Doligez
47	$2^{1522}+1$	2001-01-15	P. Zimmermann
50	$5^{193}-2^{193}$	2001-01-19	B. Silverman

5.4　三种有效密码系统的比较[34]

基于因式分解问题的因式分解密码系统，典型代表为 RSA；基于离散结数问题的

离散对数密码系统，典型代表为 DSA；基于椭圆曲线上离散对数问题的椭圆曲线密码系统(ECC)，一般认为这三类密码系统是安全且有效的。

现在分析 RSA 系统，ELGamal 密码和 ECC 密码所所需的计算量做个比较。设 $L_p(v,c)=\exp\left(c(\log p)^v(\log\log p)^{1-v}\right)$，当 v=0 时，$L_p(v,c)$ 相对于 logp 是多项式关系；当 v=1 时，是指数关系；而 $0<v<1$ 称为亚指数关系。对模数 n 的 RSA 系统，采用数域筛法分解 n 的计算复杂性为 $L_p(1/3,c)$；同样利用数域筛法求有限域 GF(p) 上的离散对数的计算复杂性，它们具有相同的数量级 $L_p(1/3,c')$；求 ECDLP 的计算复杂性为 $L_p(1,c)$。也就是说，分析 RSA 和 ELGamal 系统均存在亚指数算法，而目前分析 ECC 不存在亚指数算法。因此，对于相同规模的参数，ECC 每一比特密钥的强度比 RSA 和 ELGamal 系统要大得多；而要得到相同强度的密码，ECC 的参数规模要小得多。

下面对这三类密码在安全性方面、计算负载能力方面、密钥大小和带宽等方面进行比较。

1、安全性

由于当前国际上主流的公钥加密技术仍然是采用 RSA 系统，但 ECC 也得到了世界范围的广泛认可。RSA 的优点主要是：原理简单，易于使用；但随着整数因子分解方法与技术的不断完善，计算机运行速度的提高及计算机网络的发展，作为 RSA 加解密安全保障的大整数要求越来越大，而要保证 RSA 的安全性，就要不断增加其密钥的长度，目前一般认为 RSA 需用 1024 位及以上的字长才是安全的。但密钥长度的增加将导致其加解密速度的降低，硬件实现也越来越困难，这对使用 RSA 系统的用户带来了很重的负担，从而使其应用受到限制。ECC 具有较短的密钥，160 位的 ECC 与 1024 位的 RSA、DSA 具有相同的安全强度。ECC、RSA、DSA 的安全性分析如表三所示。表中的 MIPS 年是指以每秒执行 100 万条指令的计算机运行一年。当前一般认为破译时间为 10^{12} MIPS 年代表安全。从表 5—6 中可以知道：要保证相同的安全性，ECC 只需 160 位就已足够，RSA

表 5－6、ECC、RSA、DSA 的安全性分析

破译所需时间/MIPS 年	RSA/DSA 密钥长度	ECC 密钥长度	RSA/ECC 密钥长度之比
10^4	512	106	5:1
10^8	768	132	6:1
10^{12}	1024	160	7:1
10^{20}	2048	210	10:1
10^{78}	21000	600	35:1

和 DSA 却需 1024 位模长。ECC 的安全性比 RSA 和 DSA 增加快得多。如果是 220 位密钥

长的 ECC，它比 2048 位的 RSA 和 DSA 安全性更高。攻击有限域上离散对数问题有指数积分法，而它对椭圆曲线上的离散对数问题并不有效。对 RSA 和 DSA 而言，均存在指数时间算法，对 ECC 至今还没有发现指数时间算法。故 ECC 比 RSA、DSA 的抗攻击性强，安全性高。

2、计算负载

计算负载是指完成公钥密码和私钥密码交换所需的计算量。表 5—7 中的数据表示完成给定操作所需的时间单元数，只是进行大致的比较。

表 5－7，各密码系统的计算负载分析表

	ECDSA 或 ECES	RSA (n=1024bit)	离散对数系统/1024bit
加密	120	17	480
解密	60	384	240
签名	60	384	240
验签	120	17	480

从上表分析可知：与 ECC 相比，其它公钥密码体制由于产生密钥的计算很复杂，故在计算能力受限时很难产生合适的密钥，而 ECC 可在很短的时间里产生符合条件的密钥。用其 160 位的 ECC 进行加解密或数字签名要比用 1024 位的 RSA 密码体制快 10 倍。

3、密钥大小

密钥大小是指存贮密钥对和系统参数所需比特。CERticom 公司对 ECC、RSA、DSA 三种系统的各种参量密钥大小进行比较，如表 5—8 所示。

表 5－8，系统参量和密钥大小的比较

	系统参数	公开密码 (比特)	私有密钥 (比特)
RSA		1088	2048
DSA	2208	1024	160
ECC	481	161	160

根据表中数据分析，ECC 在系统参量和密钥大小方面比其它密码系统都要短得多。同样可由攻击它们的算法复杂度可知，在实现相同的安全性能条件下，ECC 所需密钥量远远小于 RSA 和 DSA 的密钥量，这意味着 ECC 所要求的存储空间要小得多，这就有效

解决了为增加安全强度而必须提高密钥长度所带来的实现上的难度问题。

4、带宽

带宽是指传输信息和数字签名表和数字签名的通讯量。表 5－9 显示签名长度和加密后的的密文长度。

表 5－9、加密 100bit 消息和长消息签名

	密文大小		签名长度
RSA	1024	RSA	1024
DSA	2048	DSA	320
ECC	321	ECC	320

通过比较，可以看出，当对长消息进行加解密时，ECC 与 RSA、DSA 具有相同的带宽要求；但在系统中传输短消息，ECC 比其它公钥密码系统节省带宽。

ECC 中的椭圆曲线可以通过改变曲线参数，得到不同的曲线，形成不同的循环群，即椭圆曲线具有丰富的群结构和多选择性；而 RSA 和 DSA 灵活性没有 ECC 好。

通过以上分析和比较，ECC 与其它公钥密码体制相比，能提供更好的加密强度、更高的安全性、更快的速度、更小的密钥长度和更低的带宽。

第六章　结束语

ECC 作为一种密码系统，它比 RSA、DSA 具有更高的安全强度，因 RSA、DSA 均存在指数时间复杂度的通用算法；ECC 只有亚指数时间算法，而亚指数时间算法没有指数时间处处算法准确；ECC 具有合理的安全强度，且没有很大的附加开支，由于椭圆曲线密码体制具有所需要的密钥较短、算法快速实现及对内存资源占有较少等优点，从而将成为新一代信息安全技术，为鼓励开发可适应于广泛的计算机平台及最大的互操作性的 ECC 产品，几个国际标准化组织已经同意把 ECC 作为新的信息安全标准，如 IEEP1363、ANS1X9.62 等草案。可以相信，ECC 将会逐步替代 RSA，在信息安全方面发挥更好的作用。

6.1　本文的主要研究工作

首先介绍了椭圆曲线密码系统的实现及其安全性分析课题的研究背景，该课题研究的目的和意义，对用于构造公钥密码系统的椭圆曲线基本理论及其研究现状进行了分析，重点阐述了射影平面和仿射平面中特征值不同时对应域的椭圆曲线构成的点加公式和倍点公式、椭圆曲线的阶的计算方法，并给出了椭圆曲线参数的验证过程及椭圆曲线的特征量。

其次分析了构造安全椭圆曲线密码体制的步骤以及需完成的工作，根据不同情况不同作用可选定有限域，详细论述了不同域上安全曲线的选取方法及其相关算法，同时对基点的选取算法及计算做了详细的分析。

再次 ECC 的各种算法速度的快慢是影响 ECC 实现的重要因素，尽管 ECC 密钥短，比 RSA 等公钥密码体制速度要快，但实际上 ECC 的运行速度仍然很慢，与对称密码体制相差很大，故设计 ECC 的快速算法，提高 ECC 的运行速度，是目前 ECC 研究的重要内容之一。本文介绍了加速 ECC 的算法的二个步骤，一是加速底层域运算，另一是加速群运算速度。文中设计及给出了 ECC 的快速算法，也说明了椭圆曲线密码体制的实现过程。

最后详细研究椭圆曲线密码体制的安全性，对椭圆曲线可能受到的攻击进行了分析，介绍了椭圆曲线多种攻击算法及特殊椭圆曲线的攻击方法，硬件攻击方法，对椭圆曲线的挑战等，比较了三类有效密码体制的各种特性，通过一些图表能更好地说明 ECC 的优势所在。

6.2 进一步的研究工作

ECC 的理论中还有很多的的问题没有解决，有些还需做更深入的研究；ECC 不仅在高端设备中得到重要应用，而且在低端设备中也得到广泛应用。今后 ECC 将主要研究如下内容：

1、ECC 快速算法研究；

2、EC 上的离散对数问题；

3、EC 的编码方法的研究；

4、超椭圆曲线的密码学研究及应用。

5、EC 的并行运算实现、硬件实现、软件实现。

ECC 正在以其更短的密钥和理论上更高的强度引起业界的重视，而椭圆曲线密码体制(ECC)的硬件实现也将是公钥密码体制中的一个聚焦点。椭圆曲线密码体制本身的改进；有限域数学运算的硬件实现算法的进一步改良，将能提供更长密钥和更快的数据速率的软、硬件实现。

6.3 发展方向

ECC 实现的最大困难是编码问题，到目前还没有特别好的编码方式，因而研究通用的椭圆曲线密码体制的编码，实现椭圆曲线密码体制的快速实现，有其特别重要的作用。其次，怎样快速实现 ECC，设计相关的快速算法，也是今后一段时间内所研究的重要内容之一。

致 谢

本文是在导师蒋外文教授悉心指导下完成的。从选题到具体论文写作工作，导师都倾注了大量的心血，给我多次改稿，耐心讲解。导师的知识结构、研究方法、思维模式和学术风格均使我获益匪浅；导师高尚的品德、忘我的工作热情、严谨求实的治学风范都给我以极大的教育；导师渊博的知识、学术上不断创新的思想方法、敏锐的洞察力和深邃、独到的见解给了我莫大的启迪。这些将激励我在今后的工作中不懈努力、不断进取。同时，在论文写作期间，还得到我的企业指导教师尹双云高级工程师的指教，在此，谨向两位导师表示衷心的感谢！

在攻读硕士学位期间，受到中南大学信息科学与工程学院蔡自兴博导、陈松乔博导、陈志刚博导、杨路明博导等许多知名教授、专家在学业上的教诲，使作者奠定了软件工程领域的知识基础。在此向中南大学信息科学与工程学院的领导、专家和老师们表示诚挚的谢意！同时还要感谢各位同窗好友们三年来的帮助和关心。

在这里还要感谢邵阳学院电气信息工程学院院长彭解华教授及其它领导和各位同仁，为我提供了学习和科研的环境，才使我的硕士学位论文研究得以顺利进行。

在这里，更不应该忘记的是，在三年的求学过程中，我的爱妻谢海屏女士作出了巨大的牺牲和默默的奉献，正是她给予我生活和学习上无微不至的关心和照顾，才使我的学业和硕士论文圆满完成。

攻读硕士学位期间的主要研究成果

发表的论文如下：

赵云辉，蒋外文. 椭圆曲线密码系统的安全性分析[J]. 计算技术与自动化，2004，3（23）：335-337

曾爱华，赵云辉，匡振华. 纠缠相干态的压缩特性[J]. 量子光学学报，2004，10（2）：67-72

赵云辉. 关于极矢量和轴矢量的探讨[J]. 邵阳学院学报（自然科学版），2003，2（5）：37-39

参考文献

[1]Muller V. Use of Elliptic curve in cryptography[A]. In:advances in cryptology Proceedings of CRYPTO' 85 LNCS218[C]. springer - Verlag, 1985, 417—426

[2]Koblitz, N. Elliptic curve cryptosysyems[J]. Machematics of Compution. 1987, 48:203-209

[3]Balasubramanian R, koblitz N. the improbability that an elliptic curve has subexponential discrete log problem under to Menezes — Okamotovanstane a logoritum[J]. J of cryptography. 1998, 11:141-145

[4]王晖，张方国，王育民. 大素数域上椭圆曲线密码体制的软件实现. 西安电子科技大学学报（自然科学版）. 2002, 3(29):426-428

[5]张龙军，赵霖，沈均毅. 基于有限域椭圆曲线密码体制的建立研究[J]. 小型微型计算机. 2002，10（21）：1039-1041

[6]卢凯，卞正中，葛卫丽. 椭圆曲线密码体制在电子商务中的安全应用[J]. 计算机工程. 2002, 10（25）：31-34

[7]陈建华，汪朝晖. 椭圆曲线密码的研究与发展. 信息安全技术--产业发展论坛论文集，2002，11：95-104

[8]卢开澄. 计算机密码学（第二版）. 清华大学出版社：北京. 1998，240-245

[9]杨文锋. Schoof 算法及其在椭圆曲线密码体制中的应用[J]. 2001, 6(117):10-12

[10] 张龙军，赵霖，沈均毅. 适于建立椭圆曲线体制的椭圆曲线的研究[J] . 计算机工程. 2000. 26（8）

[11]SECI. Elliptic curve cryptography:standards for efficient cryptography group:workirg draft[EB/OL], http://www.secg.org/. 2004-12

[12]杨宗辉，戴宗锋，杨栋毅等. 关于椭圆曲线密码的实现[J]. 通讯技术. 2001，6（11）：1—3.

[14]蒲利群，杨刚. 有限域 F_{2^n} 上椭圆曲线密码体制的快速实现[J]. 北京理工大学学报. 2002. 22(4):416—417

[15]顾纯祥，祝跃飞. SEA 算法及安全椭圆曲线的有效选取[J]. 信息工程大学学报. 2000，1（4）：1—4

[16]秦晓东，辛运帏，卢桂章. 椭圆曲线密码系统在 GF（p）上的基点选择方法[J].

计算机工程.2003.29（8）：64—133

[17]卢悦，严立平，卞正中.有限素整数域椭圆曲线密码体制基于代数几何快速算法设计[J].计算机工程与应用.2002，15：63—66

[18]郝林，罗平.椭圆曲线密码体制中点的数乘的一种快速算法[J].电子与信息学报。2003，25（2）：275—278

[19]马允平，马文平.圆曲线密码机制的硬件实现.技术论坛.003，2：46—48

[20]郑志彬.吴关.张乃通.基于离散对数问题的椭圆曲线公钥系统的一种快速算法[J].遥测遥控.1999，6:42-48

[22]张龙军，赵霖，沈钧毅.安全的椭圆曲线密码体制并行实现方法[J].计算机工程.2000，26（5）：1—14

[23] 张龙军，赵霖，沈钧毅.椭圆曲线密码体制快速算法研究[J].计算机工程与应用.2000，12：12—14

[24]Pollard J. Monte Carlo Methods for Indax Computation mod p[j]. Mathematics of Computation, 1978,32(6):918-924.

[25]Pohlig S, Hellman M. An Improved Algorithm for Computing Logarithms Over GF(p) and Its Cryptographic Significance[j]. IEEE Trans on Information Theory, 1978,24(1):106-110.

[26]Silverman J. The Xedni Calculus and the Elliptic Curve Discrete Logarithm Problem[j]. Design Codes and Cryptogrphy,2000,20(1):5-40.

Van Oorschot P, Wiener M. Parallel Collision Search with Cryptanalytic Applications[J].Journal of Cryptology,1999,12(1):1-28.

[27]韩明华，韩晓东.基于椭圆曲线加密的系统安全性分析[J].浙江万里学院学报.2003，16（4）：75—78.

[28]Satoh T, Araki K, Fermat Quotients and the Polynomial Time Discrete Log Algorithm for Anomalous Elliptic Curve[J]. Comm Math Pauli, 1998,47(1):81-92.

[29]Frey G, Muller M, Ruck H. The Tate Pairing and the Discrete Logarithm Applied to Elliptic Curve Cryptosystems[j]. IEEE Trans on Information Theory, 1999,45(5):1717-1719.

[30]Gaudry P. Hess F. Sman N. Construccttion and Destructive Facets of weil Descent on elliptic curves [R]. HP Laboratories, 2000.

[31] Gaudry P, An Algorithm for Solving the discrete Log problem on hyperelliptic curves [A]. Eurocrypt 2000(C). Berlin: Springer Verlag, 2000,

③

19 34.

[32]张方国，陈骁峰，王育民. 椭圆曲线离散对数的攻击现状[J]. 西安电子科技大学学报，2002，29(3)：398- 403.

[33]王衍波. 椭圆曲线上密码研究研究现状与展望. 解放军理工大学学报（自然科学版）. 2002，6（3）：18-25

[34]赵云辉，蒋外文. 椭圆曲线密码系统的安全性分析[J]. 计算技术和自动化. 2004，23(3)：335-337.