关于虚拟数字货币核心加密算法的探究

课程名称： 信息安全

院系： 计算机学院

专业： 软件工程

年级： 大二

姓名： 任思远

学号： 2016302580320

摘要：

随着区块链与数字货币的兴起，人们希望有一种更加强力的加密算法来保证信息传输的安全性。而当前广泛使用的是RSA加密算法，但为了保障数据的安全，RSA 的密钥需要不断增加，但是，密钥长度的增加导致了其加解密的速度大为降低，硬件实现也变得越来越难以承担，因此ECC加密技术逐渐被人们锁应用。

Abstract:

With the rise of block chain and digital currency, there is a desire need of a stronger encryption algorithm to ensure the security of information transmission. RSA encryption algorithms are widely used today. However, in order to ensure the security of data, RSA keys need to be continuously increased. However, the increase in the key length leads to a significant decrease in the speed of encryption and decryption, and implementations on hardware have also become increasingly harder to afford, so ECC encryption technology is gradually being used.

关键字：

虚拟货币，区块链，加密算法，椭圆曲线。

引言：

加密货币基于区块链技术，是一种基于密码学原理来确保交易安全及控制交易单位创造的交易媒介。密货币基于去中心化的共识机制，与依赖中心化监管体系的银行金融系统相对[1]。

比特币（BTC）被部分观点认为是一种去中心化，非普遍全球可支付的电子加密货币，而多数国家则认为比特币属于虚拟商品，并非货币。比特币由中本聪于2009年1月3日，基于无国界的对等网络，用共识主动性开源软件发明创立[2]。

本文主要对比特币所使用的的椭圆曲线加密算法(ECC)进行探究并进行简要的实验。

首先引入一个简单的例子[3]：

假设Alice与Bob进行通信，并且在现有代数系统下，计算‘+’，‘-’，‘\*’消耗较小，但是计算 ‘/’（除法）消耗极大，以至于几乎不可能进行除法运算。

Alice持有一个密钥k = ‘456’，然后挑出一个数字 ‘123’，进行乘法运算123 \* 456 = 56088，告知Bob ‘123 \* k = 56088’。

现在Bob想告诉Alice一个秘密 x = ‘67’，随机选取一个加密数字 M = ‘222’，并进行两次计算得到123 \* 222 = 27306，56088 \* 222 + 67 = 12451603。告知Alice ‘123 \* M = 27306，56088 \* M + x = 12451603’。

Alice欲想知道‘x’，用手中的密钥‘k’进行解密：

123 \* M \* 456 = 27306 \* 456 = 12451536， 然后将两式相减，即得 ‘x = 67’ 为秘密信息。在此过程中，若信息被敌人截获，但由于不知道密钥 ‘k’，只能做除法先得到 ‘M’，然后解出 ‘x’，但除法消耗代价极大，所以这就保证了信息的安全性。

然后问题就落在如何寻找这样一个（计算除法代价极大的）代数系统的问题上了，下面介绍国内外相关工作介绍和研究现状，然后对ECC的具体细节进行探讨。

密码学的历史可以分成两个时代：传统时代和现代。两个时代的转折点发生在1977年，即RSA算法和Diffie-Hellman密钥交换被提出。这些新的算法具有革命性是因为他们代表第一代安全基于数论的切实可行的密码学方案；这也是首次提出的使没有共享秘钥的双方能够安全通讯的方案。密码学从全世界传输密码电报到已证明能够在不用担心某人窃听密钥交换的双方安全通讯[4]。

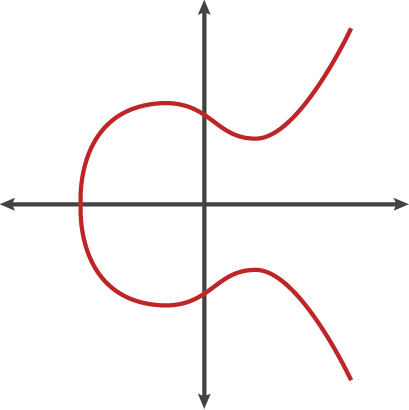
现代密码学被建立在一个理念之上，这个理念就是把公钥（把数据译成密码的钥匙）公之于众，并且自己持有密钥（解译数据的钥匙）。这些系统被称为公钥密码学系统，这些系统里最广泛被应用的是RSA。RSA的困难在于两个大数的乘法相对简单，但是大数的分解质因数是非常困难的。一个公钥加密系统有两部分，一个是公钥，一个是私钥。加密通过信息传递和应用一个数学运算来得到一串随机号码。解码采用这组随机号码并且进行一个困难的运算来解出原来的号码。有公钥的加密只能被有私钥的解码解开。

RSA算法的大致运行过程是：选择两个质数的乘积，作为一个大数，选择一个公钥，然后根据“扩展欧几里得算法”计算出私钥：，选择使得。接着选择要加密的信息，向持有私钥的一方传递信息为，持有私钥的一方将信息进行次乘方(并运用初等数论中的欧拉定理)，得到，即为解出信息。

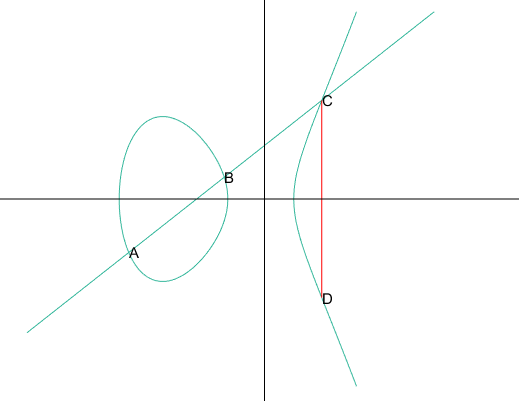
RSA基于严格的证明并且有着强大的加密能力，要想破解这个密码系统，相当于解决了一个极其困难的数学问题。即便如此，因式分解也不是最难的问题，一些算法比如两次筛法和普通数域筛法，它们在分解质因子方面有较强的能力。

这些因式分解算法随着被因式分解的数字变得越大而变得越有效率。因式分解大批数字和乘以大量数字的难度的差距随着数字（即秘钥的字节长度）变大而缩小。随着有效解码数字的资源增加，秘钥的大小必须更快增长。对限制计算能力的手机和低功率设备来说，这不是一个乐观的情况。从长期来看，因式分解和乘法的差距是不可持续的，因为密钥长度呈指数级别增长。这意味着对当代密码学来说，RSA不是理想的密码系统。在一个表现良好的trapdoor函数，对于密钥长度大小的问题，简单方法和困难方法所消耗的代价都以同样的速率增长。所以我们需要一个基于更好的trapdoor的公钥系统。综上：想要构建一个安全的密码系统，就需要找到一个好的trapdoor函数，而椭圆曲线正是基于此被注意到并被引入非对称加密领域。

在引入RSA和Diffie-Hellman后，研究人员探寻另外的密码学解决方案用来充当好trapdoor函数的算法。在1985年，加密算法被提议以一个叫椭圆曲线的数学奥秘分支为基础。椭圆曲线的定义是：，并且曲线没有尖点或自相交。如下图所示：



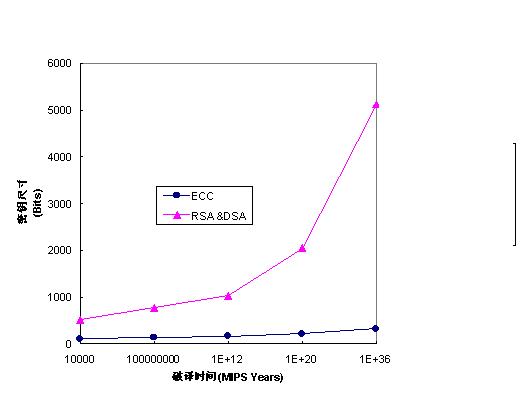
在此之上建立代数系统，首先定义加法‘+’，如下图所示：



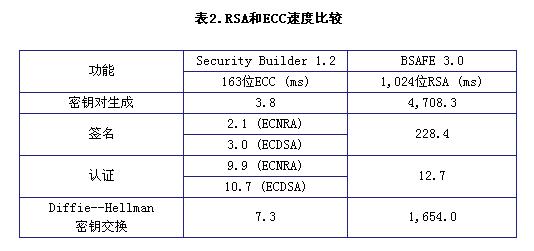
A+B表示弦AB所在的直线与曲线交点关于x轴的镜像点，即点D；若AB所在的弦和y轴平行，则定义A+B=0（无穷远点，0位单位元）。数学家们证明了加法对于该曲线的点集合构成Abel群。此外定义减法‘-’，A-B为A+(-B)，其中-B为点B关于x轴的对称点。之后定义乘法‘\*’，Z \* E(q) -> E(q)，例如2P = P + P，3P = 2P + P，椭圆曲线离散对数问题（ECDLP）就是给定点P和Q，确定整数k使 k \* P = Q，这样就得到了文章首段所希望的代数系统，即计算加法，减法，乘法都没有较大的困难，然而计算除法却要付出极大的消耗代价。事实上，一般认为在一个有限域乘法群上的离散对数问题（DLP）和椭圆曲线上的离散对数问题（ECDLP）并不等价；ECDLP比DLP要困难的多。

关于RSA与ECC在加密方面性能的对比：最突出也最明显的就是随着加密强度的上升RSA密钥长度上升显著高于ECC密钥长度。[5]此外，RSA算法的特点之一是数学原理简单、在工程应用中比较易于实现，但它的单位安全强度相对较低。目前用国际上公认的对于RSA算法最有效的攻击方法--一般数域筛(NFS)方法去破译和攻击RSA算法，它的破译或求解难度是亚指数级的。ECC算法的数学理论非常深奥和复杂，在工程应用中比较难于实现，但它的单位安全强度相对较高。用国际上公认的对于ECC算法最有效的攻击方法--Pollard rho方法去破译和攻击ECC算法，它的破译或求解难度基本上是指数级的。正是由于RSA算法和ECC算法这一明显不同，使得ECC算法的单位安全强度高于RSA算法，也就是说，要达到同样的安全强度，ECC算法所需的密钥长度远比RSA算法低（密钥长度比较如下图所示）。





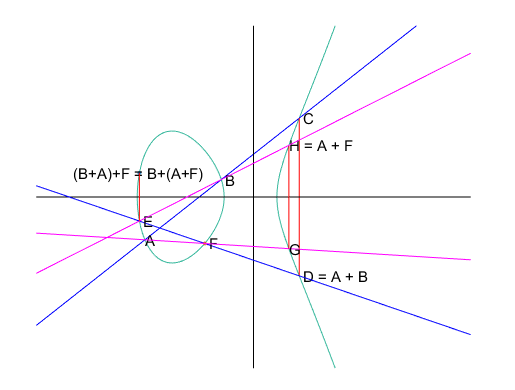
这就有效地解决了为了提高安全强度必须增加密钥长度所带来的工程实现难度的问题。（工程性能比较如下图所示）



总的来说，ECC 和 RSA 相比，在许多方面都有对绝对的优势，主要体现在以下方面：1）抗攻击性强 2）CPU 占用少 3）密钥长度短 4）网络消耗低 5）加密速度快。

下面对ECC算法的作用原理进行具体的分析。

首先研究ECC的一些数学性质，椭圆曲线对其加法构成一个Abel群。交换律是自然成立的，结合律的成立见下图（实验代码[1]见文末），给出3个原始点A、B、F。A+B相交于C反射得到D，D+F相交于E反射得到(B+A)+F；另一方面A+F相交于G反射得到H，H+B相交于E反射得到B+(A+F)。从而验证了结合律。



椭圆曲线的阶定义为：

[6]首先将曲线离散化，如果椭圆曲线上一点P，存在最小的正整数n使得nP = O(mod p) ,则将n称为P的阶。若n不存在，则P是无限阶的。

为了方便，下文将形如 的方程记为 ，下面给出一些计算椭圆曲线的核心函数。

1. 给定x值，计算区间[0,p]内的y的取值集合。

过程：计算出方程右边的值(取模)，然后遍历y，若相等，则加入集合。

%file:ECCCal.m

function [ y ] = ECCCal( a,b,p,x )

y = [];

mm = mod(x^3+a\*x+b,p);

index = 1;

for yy = 0:1:p

if mod(yy^2,p) == mm

y(index)=yy;

index=index+1;

end

end

end

1. 画出区间的在曲线上的点的图像。

过程：遍历x，用函数ECCAL计算点集并加入集合。

%file:ECCPlot.m

function [x,y] = ECCPlot(a, b, p)

x = [];

y = [];

for xr = 0:1:p

y1 = ECCCal(a, b, p, xr);

if(size(y1, 2) > 0)

x1 = zeros(1, size(y1, 2));

x1(1:size(y1, 2)) = xr;

x = [x,x1];

y = [y,y1];

end

end

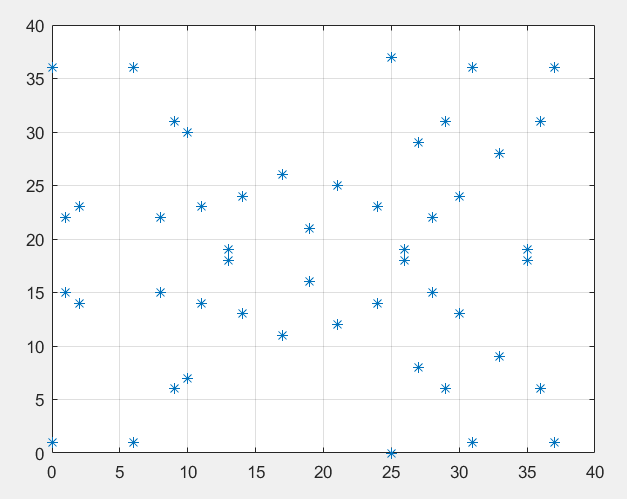
plot(x, y, '\*')

hold on

grid on

end

在命令行调用 ECCPlot(1, 1, 37)，结果如下图：



1. 计算数论分数乘法逆元

过程：先遍历求出分母d的乘法逆元，然后与分子相乘。

%file:modfrac.m

% n numerator d denominator m module

function y = modfrac(n, d, m)

n = mod(n, m);

d = mod(d, m);

i = 1;

while mod(d\*i, m) ~=1

i = i+1;

end

y = mod(n\*i, m);

end

1. 计算两点加法

过程：首先给出 的计算公式。

与方程 联立，

因为知道 和 都是方程的根，故因式分解为：

故有

1)当P，Q在x轴上投影不重合时，

2)当P，Q对称时

3)当P，Q重合时，直线PQ为切线

%file:Add.m

%calculate P(x1, y1) + Q(x2, y2) in E\_p(a, b)

function [resx, resy] = Add(a, b, p, x1, y1, x2, y2)

% Inf

if x1==inf

resx = x2;

resy = y2;

return;

end

if x2 == inf

resx = x1;

resy = y1;

return;

end

% Tangency

if x1==x2 && y1==y2

k = modfrac(3 \* x1^2 + a, 2\*y1, p);

resx = mod(k^2 - x1 - x2, p);

resy = mod(k \* (x1-resx) - y1, p);

return;

end

% P + Q = 0

if x1==x2 && y1~=y2

resx = inf;

resy = inf;

return;

end

% Normal

if x1 ~= x2

k = modfrac(y2-y1, x2-x1, p);

resx = mod(k^2 - x1 - x2,p);

resy = mod(k \* (x1-resx) - y1, p);

return;

end

end

1. 计算标量乘法

过程：使用分治乘法，例如 ，从最低位开始，如果有1，就把当前坐标加到答案中，然后当前坐标自加；若当前位为0，就只是当前坐标自加。最后的结果相当于：，一共只做了8次加法(3次答案加法，5次P倍增加法)。

分治乘法的复杂度是。

%file:NP.m

%calculate n \* P(x, y)

function [resx, resy] = NP(a, b, p, n, x, y)

resx = inf;

resy = inf;

while(n >= 1)

if(mod(n, 2) == 1)

[resx, resy] = Add(a, b, p, resx, resy, x, y);

end

n = floor(n/2);

[x, y] = Add(a, b, p, x, y, x, y);

end

end

下面是具体的ECC加密解密通信流程：

1. Alice选取一条椭圆曲线，并且选定一个基点G。
2. 然后选择一个密钥k，并计算。
3. Alice公布E，K，G。
4. Bob选取一个随机数r，用来加密信息x。先求出E上点，然后计算
5. Bob将C1、C2传给Alice。
6. Alice计算，从而得到加密信息x。

用C++模拟：详见实验代码[2]。

参考文献：

[1]. https://zh.wikipedia.org/wiki/%E5%AF%86%E7%A2%BC%E8%B2%A8%E5%B9%A3

[2]. https://zh.wikipedia.org/wiki/%E6%AF%94%E7%89%B9%E5%B8%81

[3]. https://www.zhihu.com/question/22399196

[4]. https://arstechnica.com/information-technology/2013/10/a-relatively-easy-to-understand-primer-on-elliptic-curve-cryptography/

[5]. https://blog.csdn.net/zzstack/article/details/7528213

[6]. https://blog.csdn.net/alphags/article/details/79660819