基于 w-NNAF 的快速 Edwards 曲线标量乘法

贺毅朝[1] 寇应展[2] 曲文龙[1]

(石家庄经济学院信息工程学院 石家庄 050031)[1] (军械工程学院计算机工程系 石家庄 050003)[2]

摘 要 在分析利用 Edwards 曲线上三倍点公式计算 $3^nP(n=1,2,\cdots)$ 的基础上,根据各 3^nP 的坐标具有统一表示形式的特性,提出了一种通过减少求逆运算而快速计算 $3^nP(n=2,3\cdots)$ 的新算法 Tripling_Algorithm,并将此算法与标量 k 的 w-NNAF 表示方法相结合,给出了一种计算标量乘法 kP 的高效算法 ImprovedSM-3-NNAF。通过对 ImprovedSM-3-NNAF的计算复杂性与安全性分析表明,利用该算法计算 kP 不仅是安全的,而且至少可节约 20.78%的计算量,大大改进了 Edwards 曲线上标量乘法的计算效率。

关键词 椭圆曲线密码,标量乘法,Edwards 曲线,三倍点公式,w-NNAF

中图法分类号 TP309.7 **文献标识码** A

Fast Scalar Multiplication on Edwards Curves Based on w-NNAF

HE Yi-chao[1] KOU Ying-zhan[2] QU Wen-long[1]

(Information Engineering School, Shijiazhuang University of Economics, Shijiazhuang 050031, China)[1]

(Computer Engineering Department, Ordnance Engineering College, Shijiazhuang 050003, China)[2]

Abstract Based on analysing and using the triple formula of Edwards curve to calculate $3^nP(n=1,2,\cdots)$, according to circumstances that the coordinate of every 3^nP have the unitive representation, we put forward a new algorithm Tripling_Algorithm which can quickly compute $3^nP(n=2,3\cdots)$ by reducing inverse operation. And combining this algorithm with the w-NNAF representation with scalar k, we gave a kind of highly efficient algorithm ImprovedSM-3-NNAF to calculate scalar multiplication kP(for short ImprovedSM-3-NNAF). The analysis of the complexity and security of the Improved SM-3-NNAF shows that not only calculating kP by using this algorithm is security, but also it can save the amount of calculation by at least 20.78%, and has greatly improved the computational efficiency of the scalar multiplication in Edwards curve.

Keywords Elliptic curve cryptosystems, Scalar multiplication, Edwards curves, Triple formula, w-NNAF

自 Miller 和 Koblitz 提出椭圆曲线密码系统(Elliptic Curve Cryptosystems, ECC)[1,2]以来,ECC 一直得到密码专家和信息安全者的关注,并被应用于许多重要领域。ECC 以其不可比拟的优势取代了传统的 RSA 系统[3],其中主要原因在于:(1)在相同的安全性下 ECC 的密钥更短,更有利于系统的实现;(2)ECC 的单比特安全性更高,其复杂的代数结构能够提供更高的安全系数。随着 ECC 的广泛应用,有关其软、硬件实现方面的研究越来越重要,已为密码学的一个重要研究方向。在 ECC 实现中,标量乘法 kP 是最基本也是最耗时的运算,其运算过程分为两个层次:一是上层运算,即椭圆曲线上有理点之间的运算;另一个是底层运算,主要指在有限域上的求逆、乘法和乘方运算(以下分别简记为 I,M,S)。目前有关底层运算的研究主要集中在 3 个方面:一是改进基域中的运算,减少底层域运算的运算量;二是研究标量 k 的更有效表示,减少上层运算的运算量,主要有非相邻表示法(NAF)、滑动窗口法[4]和双基数系统法(DBNS)[5,6];三是将椭圆曲线的运算等价转换为其他曲线,如转换为 Hessian 曲线[7]和 Edwards 曲线[8,9]等。

本文第 1 节介绍了 Edwards 曲线及其上的点加、倍点与 3 倍点计算公式,简述了表示标量 k 的 w-NNAF 方法;第 2 节,首先分析并设计了一种计算 3^nP 的快算法(Tripling_ Algorithm),然后基于 3-NNAF 给出了加速标量乘法 kP 的一种新的计算方法(ImprovedSM-3-NNAF);第 3 节中,对所提出的两种算法进行了计算复杂性分析与比较,并进行了安全性分析;最后指出今后需要进一步探讨和研究的问题。

1 数学背景知识

1.1 Edwards 曲线及其运算

2007 年,Edwards[8]指出在特征不为 2 的域 K 上椭圆曲线双有理等价于形如 $x^2+y^2=c^2(1+dx^2y^2)$ 的曲线,其中 c, $d\in K, cd\neq 0$ 且 $cd^4\neq 1$,该曲线被称为 Edwards 形式的椭圆曲线(Elliptic curve in Edwards form),简称 Edwards 曲线(以

到稿日期:2009-12-04 返修日期:2010-03-05 本文受 863 国家基金项目(2007AA01Z454),河北省科学技术研究与发展指导计划项目(07216926)资助。

贺毅朝(1969-),男,副教授,CCF 会员,主要研究方向为计算机密码学与智能计算,E-mail:heyichao@sjzue.edu.cn;寇应展(1962-),男,教授,硕士生导师,主要研究方向为网络安全与软件工程;曲文龙(1971-),男,副教授,硕士生导师,主要研究方向为并行计算与数据挖掘。

下记为 $EdC(K)$)。D. J. Bernstein 和 Tanja Lange 在文献[9]中对 Edwards 曲线进行了大量研究，指出域 $K(char(K)\neq 2$ 且 $2\neq 0)$上任意 Edwards 曲线都非常容易转换为与其同构且 $c=1$ 的形式 $x^2+y^2=1+dx^2y^2$。本文以下有关 Edwards 曲线的讨论均针对此形式，且所有结果容易推广到 $x^2+y^2=c^2(1+dx^2y^2)(c\neq 1)$的形式。

在 $EdC(K)$上，其有理点关于加法群的单位元为$(0,1)$，有理点 $P=(x_1,y_1)$的负元为$-P=(-x_1,y_1)$。令 $P_1=(x_1,y_1)$和 $P_2=(x_2,y_2)$为 $EdC(K)$上两有理点，则有点加公式

$$(x_1,y_1),(x_2,y_2)\mapsto\left(\frac{x_1y_2+y_1x_2}{1+dx_1x_2y_1y_2},\frac{y_1y_2-x_1x_2}{1-dx_1x_2y_1y_2}\right) \tag{1}$$

当 $P_1=P_2$ 时，得到倍点公式：

$$2(x_1,y_1)\mapsto\left(\frac{2x_1y_1}{1+dx_1^2y_1^2},\frac{y_1^2-x_1^2}{1-dx_1^2y_1^2}\right) \tag{2}$$

对式(2)，由 $x_1^2+y_1^2=1+dx_1^2y_1^2$和 $2-(x_1^2+y_1^2)=1-dx_1^2y_1^2$可消去坐标中分母的常数 d，将倍点公式等价变形为

$$2(x_1,y_1)\mapsto\left(\frac{2x_1y_1}{x_1^2+y_1^2},\frac{y_1^2-x_1^2}{2-(x_1^2+y_1^2)}\right) \tag{3}$$

这样式(3)不仅消去了常数 d，而且比式(2)减少了一次乘方运算和一次乘 d 运算。利用点加式(1)和倍点式(3)，并由 $d=(x_1^2+y_1^2-1)/x_1^2y_1^2$替换式(1)中的 d，即可得三倍点公式

$$3(x_1,y_1)\mapsto\left(\frac{(x_1^2+y_1^2)^2-4y_1^2}{4(x_1^2-1)^2x_1^2-(x_1^2-y_1^2)^2}x_1,\ \frac{(x_1^2+y_1^2)^2-4x_1^2}{-4(y_1^2-1)^2y_1^2+(x_1^2-y_1^2)^2}y_1\right) \tag{4}$$

利用式(1)、式(3)和式(4)，即可实现 EdC(K)上的各种标量运算。

1.2 w-NNAF 方法简介

1979 年 M. E. Hellman 首次提出 NAF 方法(non-adjacent form)[4,10]，此后人们对 NAF 方法进行改进与扩充，又相继提出了 NAF_w 方法和窗口 NAF 方法，丰富了对标量 k 的有效表示方法。2005 年丁勇[11]在其博士论文中给出了 w-NNAF 方法，该方法以任意正整数 $w(w\geqslant 2)$而不仅仅是 2 为基来分解标量 k。w-NNAF 的形式近似于 NAF，且满足下述引理。

引理 1[11] 令 $l(n)$表示整数 n 的 w-NNAF 表示的长度，则有：(1) $|\log_w^n|\leqslant l(n)\leqslant|\log_w^n|+1$；(2) w-NNAF($n$)的平均汉明重量约为 $|\log_w^n|(w-1)/(w+1)$。

给定标量 n，若其 w-NNAF 表示为 w-NNAF(n)$=((s_t,k_t),(s_{t-1},k_{t-1}),\cdots,(s_1,k_1)$，则有 $n=\sum_{i=1}^{t}s_iw^{k_i}$，其中 $s_i\neq 0$ 且 $-(w-1)\leqslant s_i\leqslant w-1$。对于正整数 n，很容易求出其 w-NNAF 表示。有关 w-NNAF(n)的一般求解算法请参考文献[11]，下面仅给出 $w=3$ 时求解 3-NNAF(n)算法的伪 C 代码描述，其中 $s[\ ]$与 $k[\ ]$分别为两个存放 s_i 和 k_i 的数组，$s[j]\in\{-2,-1,1,2\}$，$k[j]\geqslant 0$，$1\leqslant j\leqslant i-1$。

算法 1 3-NNAF_Algorithm (n)

```
1 { m=n;i=1;j=0;
2   while(m>0) {
3     T=m (mod 9);
4     if(T=0 or T=3 or T=6) s[i]=0;
5     if(T=1 or T=2) s[i]=T;
6     if(T=7 or T=8) s[i]=T-9;
7     if(T=4 or T=5) s[i]=T-3;
8     m=(m-s[i])/3;
9     if(s[i]≠0) { k[i]= j;i++;}
10     j++;
11   }
12 return ((s[i-1],k[i-1] ),…,(s[1],k[1]));
13 }
```

例如当 $n=721302$ 时，由算法 3-NNAF_Algorithm(n)可得 3-NNAF(721302)$=((1,12),(1,11),(2,8),(-2,5),(1,4),(-1,1))$，也即 $721302=3^{12}+3^{11}+2\times 3^8-2\times 3^5+3^4-3$。注意到 $721302=2^{20}-2^{18}-2^{16}+2^9-2^7+2^5-2^3-2$，显然 3-NNAF(721302)比 NAF(721302)的长度更短。事实上，由引理 1 可知 w-NNAF(n)总比 NAF(n)的长度更短，而且算法 1 的计算复杂性也仅为 $O(\log_w^n)$。

2 基于 w-NNAF 的标量乘法

2.1 计算 3^nP 的快速算法

为了减少非常耗时的求逆运算次数，降低标量乘法运算量，下面基于 k 的有效表示，研究在应用三倍点式(4)计算 $3^nP(n=1,2,\cdots)$时，如何将 $n=1,2,\cdots$的各倍点坐标表示成统一的形式；在此基础上，利用归纳方法导出计算 $3^nP(n=2,3,\cdots)$的有效递推公式，并据此给出相应的算法描述。

设 $P=(x_1,y_1)\in EdC(K)$，令 $A_0=x_1$，$B_0=1$，$C_0=y_1$，$D_0=1$，则计算 $3P$ 的式(4)可重写为

$$3(x_1,y_1)\mapsto\left(\frac{(A_0D_0)^4+2(A_0D_0)^2(B_0C_0)^2+(B_0C_0)^4-4(B_0C_0)^2(B_0D_0)^2}{3(A_0D_0)^4-4(A_0D_0)^2(B_0D_0)^2+2(A_0D_0)^2(B_0C_0)^2-(B_0C_0)^4}\frac{A_0}{B_0},\ \frac{(A_0D_0)^4+2(A_0D_0)^2(B_0C_0)^2+(B_0C_0)^4-4(A_0D_0)^2(B_0D_0)^2}{-3(B_0C_0)^4+4(B_0C_0)^2(B_0D_0)^2-2(A_0D_0)^2(B_0C_0)^2+(A_0D_0)^4}\frac{C_0}{D_0}\right) \tag{5}$$

若令

$$\begin{aligned}A_1&=[(A_0D_0)^4+2(A_0D_0)^2(B_0C_0)^2+(B_0C_0)^4-4(B_0C_0)^2(B_0D_0)^2]A_0\\B_1&=[3(A_0D_0)^4-4(A_0D_0)^2(B_0D_0)^2+2(A_0D_0)^2(B_0C_0)^2-(B_0C_0)^4]B_0\\C_1&=[(A_0D_0)^4+2(A_0D_0)^2(B_0C_0)^2+(B_0C_0)^4-4(A_0D_0)^2(B_0D_0)^2]C_0\\D_1&=[-3(B_0C_0)^4+4(B_0C_0)^2(B_0D_0)^2-2(A_0D_0)^2(B_0C_0)^2+(A_0D_0)^4]D_0\end{aligned}$$

则式(5)即为 $3(x_1,y_1)=(A_1/B_1,C_1/D_1)$，再对其利用式(4)可得 3^2P 的计算式

$$3^2(x_1,y_1)\mapsto\left(\frac{(A_1D_1)^4+2(A_1D_1)^2(B_1C_1)^2+(B_1C_1)^4-4(B_1C_1)^2(B_1D_1)^2}{3(A_1D_1)^4-4(A_1D_1)^2(B_1D_1)^2+2(A_1D_1)^2(B_1C_1)^2-(B_1C_1)^4}\frac{A_1}{B_1},\ \frac{(A_1D_1)^4+2(A_1D_1)^2(B_1C_1)^2+(B_1C_1)^4-4(A_1D_1)^2(B_1D_1)^2}{-3(B_1C_1)^4+4(B_1C_1)^2(B_1D_1)^2-2(A_1D_1)^2(B_1C_1)^2+(A_1D_1)^4}\frac{C_1}{D_1}\right) \tag{6}$$

又令

$$\begin{aligned}A_2&=[(A_1D_1)^4+2(A_1D_1)^2(B_1C_1)^2+(B_1C_1)^4-4(B_1C_1)^2(B_1D_1)^2]A_1\\B_2&=[3(A_1D_1)^4-4(A_1D_1)^2(B_1D_1)^2+2(A_1D_1)^2(B_1C_1)^2-(B_1C_1)^4]B_1\\C_2&=[(A_1D_1)^4+2(A_1D_1)^2(B_1C_1)^2+(B_1C_1)^4-\end{aligned}$$

$$4(A_1D_1)^2(B_1D_1)^2]C_1$$

$$D_2=[-3(B_1C_1)^4+4(B_1C_1)^2(B_1D_1)^2-2(A_1D_1)^2(B_1C_1)^2+(A_1D_1)^4]D_1$$

则式(6)即为 $3^2(x_1,y_1)=(A_2/B_2,C_2/D_2)$。再对其利用式(4)又可得 3^3P 的计算公式,…,如此根据所得结果反复利用式(4)及上述形式的变量代换,即可得到 $3^nP(n=1,2,\cdots)$ 的计算式。注意到无论是各计算公式还是变量代换形式都具有形式上的统一性,于是根据其具有的递推性质,由数学归纳法易证下述定理成立。

定理 1 设 $P=(x_1,y_1)$ 为 $EdC(K)$ 上一有理点,令 $A_0=x_1,B_0=1,C_0=y_1,D_0=1$,则当 $n\geqslant 1$ 时,连续三倍点 $3^nP=3^n(x_1,y_1)$ 的计算公式为 $3^n(x_1,y_1)=(A_n/B_n,C_n/D_n)$,其中 $n\geqslant 1$,

$$A_n=[(A_{n-1}D_{n-1})^4+2(A_{n-1}D_{n-1})^2(B_{n-1}C_{n-1})^2+(B_{n-1}C_{n-1})^4-4(B_{n-1}C_{n-1})^2(B_{n-1}D_{n-1})^2]*A_{n-1}$$

$$B_n=[3(A_{n-1}D_{n-1})^4-4(A_{n-1}D_{n-1})^2(B_{n-1}D_{n-1})^2+2(A_{n-1}D_{n-1})^2(B_{n-1}C_{n-1})^2-(B_{n-1}C_{n-1})^4]*B_{n-1}$$

$$C_n=[(A_{n-1}D_{n-1})^4+2(A_{n-1}D_{n-1})^2(B_{n-1}C_{n-1})^2+(B_{n-1}C_{n-1})^4-4(A_{n-1}D_{n-1})^2(B_{n-1}D_{n-1})^2]*C_{n-1}$$

$$D_n=[-3(B_{n-1}C_{n-1})^4+4(B_{n-1}C_{n-1})^2(B_{n-1}D_{n-1})^2-2(A_{n-1}D_{n-1})^2(B_{n-1}C_{n-1})^2+(A_{n-1}D_{n-1})^4]*D_{n-1}$$

根据定理 1,若已知 $EdC(K)$ 上有理点 $P=(x_1,y_1)$,则计算 $3^nP(n\geqslant 2)$ 的有效算法(记为 Tripling_Algorithm)的伪 C 代码描述如下:

算法 2 Tripling_Algorithm($(x_1,y_1),n$)

```
1 { A=x1;B=1;C=y1;D=1;
2   for(i=1;i≤n;i++)
3   { T1=A*D;T2=B*C;T3=B*D;
4     T1=(T1)^2;T2=(T2)^2;T3=(T3)^2;
5     W1=T1*T2;W2=T2*T3;W3=T1*T3;
6     T1=(T1)^2;T2=(T2)^2;
7     A=(T1+2*W1+T2-4*W2)*A;
8     B=(3*T1-4*W3+2*W1-T2)*B;
9     C=(T1+2*W1+T2-4*W3)*C;
10    D=(-3*T2+4*W2-2*W1+T1)*D;
11  }
12 return(A/B,C/D);
13 }
```

2.2 基于 3-NNAF 的标量乘法

显然,利用 3-NNAF 表示标量 k ($k>3$)时,其中含有一系列不同的 3^{k_i} ($1\leqslant i\leqslant t$ 且 $1\leqslant k_1\leqslant k_2\leqslant\cdots\leqslant k_t$),则计算 kP 需要计算一系列的 $3^{k_i}P(k_1\leqslant k_2\leqslant\cdots\leqslant k_t)$,这恰恰可以连续应用算法 2 来完成。首先应用算法 2 求得 $3^{k_1}P$,然后在将 $3^{k_1}P$ 作为 (x_1,y_1) 的基础上,取 $n=k_2-k_1$ 并调用算法 2 即可求得 $3^{k_2}P$,…。一般地,在计算出 $3^{k_i}P$ 后将其作为 (x_1,y_1),然后取 $n=k_{i+1}-k_i$ 并应用算法 2 即可求得 $3^{k_{i+1}}P$。在计算这些具有不连续指数的标量乘法 $3^{k_i}P$ 的同时,根据 $kP=\sum_{i=1}^{t}s_i3^{k_i}P$(其中 $s_i\in\{-2,-1,1,2\}$)对所求结果依次进行累加,即可计算出 kP 的坐标。不妨设 $P=(x_1,y_1)\in EdC(K)$,上述计算 $kP(k>3)$ 的算法(记为 ImprovedSM-3-NNAF)伪 C 代码描述为:

算法 3 ImprovedSM-3-NNAF($(x_1,y_1),k$)

```
1 { if (k≤3) return("k is smaller!");
2 ((s[m],k[m]),…,(s[1],k[1]))=3-NNAF_Algorithm (k);
3   if(k[1]=0){
4     if(s[1]=1) (x,y)=(x1,y1);
5     else if(s[1]=-1) (x,y)=(-x,y);
6     else{ (x,y)=DoublePoint (x1,y1);
7         if(s[1]=-2) (x,y)=(-x,y);
8     }
9     i=2;
10  } else{ (x,y)=(0,1);i=1;k[0]=0;}
11  While (i≤m){
12  (a,b)=Tripling_Algorithm(P(x1,y1),k[i+1]-k[i]);
13    (x1,y1)=(a,b);
14    if(s[i]=-1) (a,b)=(-a,b);
15    else if(s[i]≠0 and s[i]≠1) {
16        (a,b)=DoublePoint((a,b));
17        if(s[i]=-2) (a,b)=(-a,b);
18    }
19    if(s[i]≠0) (x,y)=AddPoint((x,y),(a,b));
20    i++;
21  }
22  return ((x,y));
23 }
```

在算法 3 中,m 元组 $((s[m],k[m]),\cdots,(s[1],k[1]))$ 为 k 的 3-NNAF 表示;步骤 10 中 $k[0]\notin$ 3-NNAF(k),仅仅是为了简化算法描述而虚设的,与实现无关;AddPoint($(x_1,y_1),(x_2,y_2)$)表示利用式(1)计算点加 $(x_1,y_1)+(x_2,y_2)$ 的子函数,DoublePoint((x_1,y_1))表示利用式(3)计算倍点 $2(x_1,y_1)$ 的子函数,它们的算法描述由式(1)和式(3)很容易写出,在此略去。

例如,对于 $P=(x_1,y_1)\in EdC(K)$,利用算法 3 计算 $721302P$ 的过程如下:

首先由步骤 2 求得 $((s[m],k[m]),(s[m-1],k[m-1]),\cdots,(s[1],k[1]))=((1,12),(1,11),(2,8),(-2,5),(1,4),(-1,1))$。因为 $k[1]\neq 0,P_0=(x,y)=(0,1),i=1$ 且 $k[0]=0$,于是再由步骤 11—步骤 21,计算过程为:

$$\begin{pmatrix}P_1\leftarrow 3P\\P_0\leftarrow P_0-P_1\end{pmatrix}\Rightarrow\begin{pmatrix}P_1\leftarrow 3^3P_1\\P_0\leftarrow P_0+P_1\end{pmatrix}\Rightarrow\begin{pmatrix}P_1\leftarrow 3P_1\\P_0\leftarrow P_0-2P_1\end{pmatrix}\Rightarrow$$

$$\begin{pmatrix}P_1\leftarrow 3^3P_1\\P_0\leftarrow P_0+2P_1\end{pmatrix}\Rightarrow\begin{pmatrix}P_1\leftarrow 3^3P_1\\P_0\leftarrow P_0+P_1\end{pmatrix}\Rightarrow\begin{pmatrix}P_1\leftarrow 3P_1\\P_0\leftarrow P_0+P_1\end{pmatrix}$$

最后输出的 P_0 即为所求。

3 算法的复杂性与安全性分析

3.1 算法复杂性分析与比较

令 D 表示乘 d 运算,显然其计算复杂性不超过 $1M$,故不妨令 $1D=1M$;此外加法、减法、乘 2 运算和乘 4 运算均比较简单,忽略不计。于是根据文献[12]中 $I=10M,S=0.8M$ 的结论,点加式(1)的计算复杂性为 $2I+5M+1D=26M$,倍点式(3)的计算复杂性为 $2I+1M+2S=22.6M$,三倍点式(4)的计算复杂性为 $2I+4M+6S28.8M$。

对于算法 Tripling_Algorithm($(x_1,y_1),n$),步骤 3—步骤 10 的一次迭代需要 10 次乘法和 5 次乘方运算,而求逆运

算仅在算法结束为输出 3^nP 的坐标时才执行 2 次，因此其计算复杂性为 $n(10M+5S)+2I=(14n+20)M$。

目前，计算 $3^nP(n\geqslant 2)$ 的主要方法是应用 Horner 规则[12]，即 $3^nP=\underbrace{3\times(3\times(3\times(\cdots 3\times(3P)\cdots)))}_{n}$，其算法描述请参考文献[13]。易知，利用 Horner 规则计算 3^nP 的计算复杂性为 $28.8nM$，由于 $n\geqslant 2$，因此 $28.8nM>(14n+20)M$，即利用算法 2 计算 3^nP 相比 Horner 规则更节省运算量。实际上，在实际应用中 n 的值往往都比较大(注意到：当 $n=10$ 时 $3^{10}=59049$ 仅仅是一个 5 位整数；而当 $n=20$ 时 $3^{20}=3486784401$ 也仅仅是一个 10 位整数)。令 $W(n)=[28.8nM-(14n+20)M]/28.8nM=0.5139-0.6944/n$，则当 $n\geqslant 10$ 时，由 $W(n)\geqslant 0.4444$ 可知算法 2 比 Horner 规则至少节省 44.44%的运算量；当 $n\geqslant 20$ 时，由 $W(n)\geqslant 0.4792$ 可知算法 2 比 Horner 规则至少节省 47.92%的运算量；而当 $n\geqslant 30$ 时，由 $W(n)\geqslant 0.4907$ 可知算法 2 比 Horner 规则至少节省 49.07%的运算量…。显然，随着 n 值的逐渐增大，算法 2 比 Horner 规则节省的运算量也越来越多，但最多不可能超过 51.39%。于是，有下述结论：

结论 1 在计算 $3^nP(n\geqslant 2)$ 时，算法 2 比 Horner 规则更优；而且当 $n\geqslant 10$ 时，算法 2 计算 $3^nP(n\geqslant 2)$ 比 Horner 规则至少节省 44.44%的运算量；同时，随着 n 值的增大，算法 2 所节省的运算量也随之增大，并且最大不超过 51.39%。

对于算法 ImprovedSM-3-NNAF$((x_1,y_1),k)$，由于 3-NNAF_Algorithm(k)的复杂性为 $O(\log_w^n)$，相对于算法 Tripling_Algorithm()可忽略不计，因此算法 3 的计算复杂性主要取决于 While 循环(即步骤 11－步骤 21)。根据 3-NNAF(k)的表示，$kP=\sum_{i=1}^{t}s_i(3^{k_i}P)$，因此在 While 循环中共需要调用 t 次 Tripling_Algorithm 算法。又当 $s_i=2$ 或 $s_i=-2$ 时，需要调用 DoublePoint()计算倍点 $2(a,b)$，于是由引理 1 与算法 1 可知，调用 DoublePoint()不超过 $2t/9$ 次。此外，对所求得的各项进行累加时还需要调用 t 次 AddPoint()以计算两个不同有理点的和。于是，t 次调用 Tripling_Algorithm()的计算量分别为 $(14k_i+20)M(1\leqslant k_1\leqslant k_2\leqslant\cdots\leqslant k_t)$，而 DoublePoint()的计算量为 $22.6M$，AddPoint()的计算量为 $26M$，故而 While 循环的总计算量为

$$(14k_1+20)M+\sum_{i=2}^{t}[14(k_i-k_{i-1})+20]M+(2t/9)\times 22.6M+t\times 26M=(14k_t+51t)M$$

从而算法 3 的计算复杂性为 $(14k_t+51t)M$。

当前，在确定标量 $k(k>3)$ 的有效表示后，计算 kP 通常是基于 Horner 规则进行的。这样对于 3-NNAF(k)，在 $k_1\neq 0$ 时(对 $k_1=0$ 只需在展开式的末尾加上一项 s_0P，$s_0\in\{-2,-1,1,2\}$ 即可，由于其计算量很小，忽略不计)，kP 的计算是按照下面展开式进行的。

$$kP=(\sum_{i=1}^{t}s_i3^{k_i})P=\underbrace{3(\cdots}_{k_1}\underbrace{3(\cdots}_{k_{t-1}-k_{t-2}}\underbrace{3(\cdots}_{k_t-k_{t-1}}3(3(\cdots 3(s_tP))\cdots)+s_{t-1}P)\cdots)+\cdots)+s_1P)$$

根据上面展开式，很容易给出这种方法(以下称为 HSM3-NNAF 算法)的计算复杂性为

$$k_t(28.8M)+(2t/9)\times 22.6M+26tM=(28.8k_t+31t)M$$

比较算法 3 与 HSM3-NNAF 算法易知，只要 $(28.8k_t+31t)>(14k_t+51t)\Leftrightarrow k_t/t>1.352$，则算法 3 优于 HSM3-NNAF 算法。根据引理 1，标量 k 的 w-NNAF(k)表示长度为 $l(k)$，注意到在 3-NNAF(k)中相邻的幂 3^x 的指数并不总是连续的，那么由算法 1 必然有 $k_t\geqslant(3/2)l(k)$ 并且 $t\leqslant l(k)-1$，而 $l(k)\geqslant 2$ 显然，因此总有 $k_t/t\geqslant 1.5+3/2(l(k)-1)\geqslant 3>1.352$，即算法 3 总是优于 HSM3-NNAF 算法。为了比较其优劣程度，令

$$T(k_t/t)=\frac{(28.8k_t+31t)M-(14k_t+51t)M}{(28.8k_t+31t)M}=\frac{37}{72}\left(1-\frac{A}{k_t/t-B}\right)$$

式中，$A=B+(50/37)$ 与 $B=155/144$ 是常数，显然 $T(k_t/t)$ 是关于 k_t/t 的单调递增函数，这样由 $k_t/t\geqslant 3$，必有 $T(k_t/t)\geqslant(37/72)[1-A/(3-B)]\geqslant 0.207836$。从而算法 3 比 HSM3-NNAF 至少节省 20.78%的计算量。于是，可得下述结论：

结论 2 当利用 3-NNAF 表示标量 $k(k>3)$ 时，应用算法 3 计算标量乘法 kP 比基于 Horner 规则的 HSM3-NNAF 方法至少节省 20.78%的计算量。

3.2 安全性分析

目前，对 ECC 的常见攻击有简单能量分析(simple power analysis，SPA)和差分能量分析(Differential power analysis，DPA)[14]。一般而言，抵抗 SPA 的防御措施均可有效转变为抵抗 DPA 的措施[15]。为此，下面有关安全性的讨论主要针对 SPA 攻击。

ImprovedSM-3-NNAF 算法的关键是 While 循环(步骤 11－步骤 21)，其中的算法 Tripling_Algorithm 不依赖于秘密信息 k 的具体比特位的值，而 While 循环(步骤 11－步骤 21)中其它运算也不依赖于秘密信息 k 的具体比特位的值，因此算法 ImprovedSM-3-NNAF 对 SPA 等攻击是安全的。于是，下述结论成立。

结论 3 ImprovedSM-3-NNAF 算法对于常见的能量分析攻击是安全的。

结束语 目前，有关 Edwards 曲线的研究还相对较少，诸如多标量乘法 $kP+sQ(k,s\in N)$ 等问题还有待深入研究。本文基于 Edwards 曲线的三倍点公式和坐标的统一表示形式，设计了一种计算 $3^nP(m=2,3,4,\cdots)$ 的安全且有效的新算法，并基于 k 的 3-NNAF 表示提出了计算标量乘法 kP 的有效算法 ImprovedSM-3-NNAF，对其计算复杂性和安全性分析表明：新算法不仅可以抵御 SPA 等能量分析攻击，而且大大减少了求逆运算的次数，使标量乘法 kP 的计算量至少减少 20.78%，有效改善了标量乘法运算的效率。今后，将进一步研究计算 Edwards 曲线上多标量乘法 $kP+sQ$ 的安全有效方法，以更好地提高 ECC 的实现速度。

参 考 文 献

[1] Miller V S. Use of elliptic curve in cryptography[C]// Williams H C, ed. Advances in Cryptology-CRYPTO'85. Lecture Notes in Computer Science. vol. 218, Springer-Verlag, 1986: 417-426

[2] Koblitz N. Elliptic curve cryptosystem[J]. Mathematics of Computation, 1987, 48(177): 203- 209

[3] 冯登国，裴定一. 密码学导引[M]. 北京：科学出版社，2001

[4] Hankerson D, Menezes A, Vanstone S. 椭圆曲线密码学导论[M]. 张焕国,等译. 北京:电子工业出版社,2005

[5] Dimitrov V S, Jullien G A. A new nymber representation with application[J]. IEEE Circuits and Systems Magazine, 2003, 3(2):6-23

[6] Avanzi R, Dimitrov V S, et al. Extending scalar multiplication using double bases[C]// Proceedings of Advances in Cryptology-ASIACRYPT2006. Springer Berlin: Heidelberg Press, 2006

[7] Joye M, Quisquater J J. Hessian elliptic curves and side-channel attack[C] // Proceedings of the Workshop on Cryptographic Hardware and Embedded Systems-CHES 2001. LNCS2162. Berlin: Springer, 2001: 402-410

[8] Edwards H M. A normal form for elliptic curves[J]. Bulletin of the American Mathematical Society, 2007, 44(3): 393-422

[9] Bernstein D J, Lange T. Faster addition and doubling on elliptic curves[C] // Kurosawa K, ed. Advances in cryptology-ASIACRYPT2007. volume 4833 of Lecture Notes in Computer Science. Berlin Heidelberg: Springer-Verlag, 2007: 29-50

[10] Hellman M E. The mathematics of public-key cryptography[J]. Scientific American, 1979, 16(7): 32-39

[11] 丁勇. 椭圆曲线密码体系中标量乘的快速算法研究[D]. 西安:西安电子科技大学,2005

[12] Fong K, Hankerson D, López J, et al. Field inversion and point halving revisited [J]. IEEE Transactions on Computers, 2004, 53(8): 1047-1059

[13] Knuth D E. The Art of Computer Programming: Seminumerical Algorithms(Third edition)[M]. Vol. 2, Addtion- Wesley, 1998

[14] Messerges T S, Dabbish E A, Sloan R H. Power analysis attacks of modular exponentiation in smartcards[A] // Proceedings of the Workshop on Cryptographic Hardware and Embedded System(CHES1999)[C]. Berlin: Springer, 2000: 144-157

[15] 张宝华,殷新春,张海灵. Edwards 曲线安全快速标量乘法运算算法—EDSM[J]. 通信学报,2008,29(10):76-81

(上接第 58 页)

列号 N_A 进行比较判断,从而保证了 TEARAN 协议一定是无环的。

结束语 本文结合目前信息安全中的研究热点即可信计算,以及当前 Ad hoc 网络中典型安全路由协议 ARAN 协议。提出了一种新的安全路由协议 TEARAN 协议。该协议采用了 TPM 中密钥对产生、密钥安全存储以及数字签名、非对称密钥和对称密钥算法的引擎,从而加快了端到端认证、鉴别和加解密的过程,同时,结合邻居节点可信阈值软安全的保证,避开网络中的恶意节点加入到网络中,实现了路由建立过程中双安全的保证,最后对该协议的有效性进行了理论的分析和证明。在未来的工作中,我们将进一步通过仿真实验与其他经典的安全路由协议进行性能对比,来评估该协议的有效性,以及与 QoS 要求进行结合考虑,实现一种满足 QoS 的安全路由协议。

参 考 文 献

[1] Li dong Z, Haas Z J. Securing ad hoc networks [J]. IEEE Network Magazine, 1999, 13(6): 24-30

[2] Zhang C, Zhou M C, Yu M. Ad hoc network security: a review [J]. Int. J. Commun. Syst., 2007, 20(8): 909-925

[3] Hu Y C, Perring A, Johnson D B. Ariadne: A Secure On-Demand Routing Protocol for Ad Hoc Networks[S]. IEEE WMCSA, 2002: 23-28

[4] Hu Y C, Johnson D, Perring A. SEAD: secure efficient distance vector routing for mobile wireless ad hoc networks[C]// Fourth IEEE Workshop on Mobile Computing Systems and Applications. June 2002: 3-13

[5] Sanzgiri K, Dahill B, Levine B N, et al. Authenticated routing for ad hoc networks[J]. IEEE J. Select. Areas Commun., 2005, 2(1)

[6] Zapata M G. Secure Ad hoc on-demand distance vector(SAODV) routing[J]. ACM SIGMOBILE Mobile Computing and Communications Review, 2002, 8(3): 106-107

[7] Papadimitratos P, Haas Z J. Secure routing for mobile ad hoc networks[C] // Proceedings of the SCS Communication Networks and Distributed Systems Modeling and Simulation Conference (CNDS2002)

[8] Hu Yih-chun, Perrig A. A survey of secure wireless ad hoc routing[J]. IEEE Security& Privacy, 2004, 2(3): 28-39

[9] 张焕国,罗捷,金刚,等. 可信计算研究进展[J]. 武汉大学学报:理学版,2006,5(52):513-518

[10] TCG. Trusted Platform Module(TPM) Summary[EB/OL]. https://www.trusted computing group.org/. 2008,5

[11] TCG 1.2 Architecture Overview. Trusted Computing Group. 2004

[12] Brickell E, Camenisch J, Chen L. Direct anonymous attestation [C]// Proceedings of the 11th ACM Conference on Computer and Communications Security. Washington, DC, USA, 2004: 132-145

[13] Theodorakopoulos G, Baras J S. Trust evaluation in ad-hoc networks[C] // Proc. 2004 ACM Workshop on Wirel. Security, Philadelphia, U.S., 2004: 1-10

[14] Ren K, Li T, Wan Z, et al. Highly reliable trust establishment scheme in ad hoc networks[J]. Comput. Netw., Apr. 2004: 687-699

[15] Reed M G, Syverson P F, Goldschlag D M. Anonymous Connections and Onion Routing[J]. IEEE Journal on Selected Areas in Communications, 1998, 16(4)

[16] Zhang Y, Liu W, Lou W. Anonymous Communications in Mobile Ad Hoc Networks[C]// IEEE INFOCOM. 2005

[17] Zhu B, Wan Z, Kankanhalli M S, et al. Anonymous Secure Routing in Mobile Ad-Hoc Networks[C]// 29th IEEE International Conference on Local Computer Networks (LCN'04). 2004: 102-108

[18] Pfitzmann A, Kohntopp M. Anonymity, Unobservability, and Pseudonymity-A Proposal for Terminology[C] // H. Federrath, ed. DIAU'00, Lecture Notes in Computer Science 2009. 2000: 1-9