УДК 681.3.06

Е.Г.КАЧКО, канд. техн. наук, А.В.СВИНАРЕВ, канд. техн. наук, С.А.ГОЛОВАШИЧ

МЕТОДЫ И АЛГОРИТМЫ УСКОРЕНИЯ ВЫЧИСЛЕНИЙ В НЕСИММЕТРИЧНЫХ ПРЕОБРАЗОВАНИЯХ НА ЭЛЛИПТИЧЕСКИХ КРИВЫХ

1. Введение

В настоящее время предложено множество алгоритмов несимметричных криптографических преобразований. Их стойкость основана на сложности решения некоторой математической задачи. Широко известны два класса задач криптоанализа: факторизация и нахождение дискретного логарифма в поле GF(p). В течение последних 5 лет проводятся исследования третьего класса несимметричных преобразований – преобразования на эллиптических кривых [1,2].

Эллиптическая кривая E над полем $\boldsymbol{Z}_p$, где p – простое, $p>3$, определяется уравнением

$$y^2 = x^3+ax+b, \qquad (1)$$

где $a, b \in Z_p$, $4a^3+27b^2 \not\equiv 0 \pmod p$. Множество точек на кривой $E(\boldsymbol{Z}_p)$ состоит из всех точек (x,y); $x, y \in \boldsymbol{Z}_p$, удовлетворяющих (1). В множество $E(\boldsymbol{Z}_p)$ также включается нулевая точка, обозначаемая как 0.

Для точек эллиптической кривой определим операцию сложения, удовлетворяющую следующим свойствам:

1. $P+0 = 0+P = P$ для $\forall P \in E(\mathrm{GF}(p))$

2. Для любой точки $P = (x,y)$, $P \in E(\mathrm{GF}(p))$, существует точка $Q = (x,-y)$, $Q \in E(\mathrm{GF}(p))$, такая, что $P+Q = 0$. Точка Q называется обратным элементом по отношению к P и обозначается как $(-P)$.

3. Для любых точек $P,Q \in E(\mathrm{GF}(p))$ существует точка $R=P+Q$; $R \in E(\mathrm{GF}(p))$.

Определим также операцию скалярного умножения точки на число. Пусть $P \in E(\mathrm{GF}(p))$, $c \in N$, тогда $c \times P = \underbrace{P + P + \ldots + P}_{c \text{ раз}}$.

Таким образом, множество $E(\mathrm{GF}(p))$, на котором определена операция сложения, образует абелеву группу.

2. Алгоритмы сложения на эллиптической кривой

Существует 2 способа внутреннего (машинного) представления точек эллиптической кривой – аффинные координаты и проективные координаты.

Аффинные координаты – пара чисел (x, y), удовлетворяющая уравнению кривой (1). Точка 0 не имеет аффинных координат, но для вычислений может быть представлена как пара чисел, не удовлетворяющих (1): (0, 0) при $b \neq 0$, (0,1) при $b=0$. Заметим, что при известном значении x можно из (1) вычислить y, поэтому точку можно однозначно представить как (x, y'), где $y' = y \bmod 2$.

Алгоритм 1. Сложение точек в аффинном представлении.
Вход: $P = (X_1,Y_1)$, $Q = (X_2,Y_2)$, $P \neq Q$.
Выход: $R = P+Q = (X_3,Y_3)$.
1. $L := \dfrac{X_2 - X_1}{Y_2 - Y_1}$.
2. $X_3 := L^2 - X_1 - X_2$.
3. $Y_3 := L(X_1 - X_3) - Y_1$.

Алгоритм 2. Удвоение точек в аффинном представлении.
Вход: $P = (X_1,Y_1)$.
Выход: $R = P+P = (X_3,Y_3)$.
1. $L := \dfrac{3X_1^2 + a_1}{2Y_1}$.
2. $X_3 := L^2 - 2X_1$.

3. $Y_3 := L(X_1 - X_3) - Y_1$.

Вычислительную сложность сложения точек можно оценить как

$$I_1(l) = 2I_{mul}(l) + I_{sqr}(l) + I_{inv}(l) + 6I_{add}(l),$$

где l – длина числа p в словах;
$I_{mul}(l)$ – вычислительная сложность умножения в поле GF(p);
$I_{sqr}(l)$ – вычислительная сложность возведения в квадрат в GF(p);
$I_{inv}(l)$ – вычислительная сложность нахождения обратного элемента в GF(p);
$I_{add}(l)$ – вычислительная сложность сложения и вычитания в GF(p).
Вычислительная сложность удвоения точки (т.е. сложения $P+P$) оценивается как

$$I_2(l) = 2I_{mul}(l) + 2I_{sqr}(l) + I_{inv}(l) + 5I_{add}(l).$$

Проективные координаты – числа (X, Y, Z) такие, что $x = X/Z^2$, $y = Y/Z^3$. Точка 0 имеет координаты $(\lambda^2, \lambda^3, 0)$, где λ – произвольное ненулевое число. Проективные координаты не являются однозначными, т.к. $(X, Y, Z) = (\lambda^2 X, \lambda^3 Y, \lambda Z)$.

Алгоритм 3. Сложение точек в проективном представлении.
Вход: $P = (X_0, Y_0, Z_0)$, $Q = (X_1, Y_1, Z_1)$, $P \neq Q$.
Выход: $R = P+Q = (X_2, Y_2, Z_2)$.
Если $P = Q$, алгоритм дает результат (0, 0, 0).

1. $L_1 := X_0 Z_1^2 - X_1 Z_0^2$
2. $L_2 := Y_0 Z_1^3 - Y_1 Z_0^3$
3. $L_3 := X_0 Z_1^2 + X_1 Z_0^2$
4. $L_4 := Y_0 Z_1^3 + Y_1 Z_0^3$
5. $Z_2 := Z_0 Z_1 L_1$
6. $X_2 := L_2^2 - L_3 L_1^2$
7. $Y_2 := ((L_3 L_1^2 - 2X_2)L_2 - L_4 L_1^3)/2$

Вычислительную сложность сложения точек можно оценить как

$$I_3(l) = 12I_{mul}(l) + 4I_{sqr}(l) + 10{,}5I_{add}(l).$$

Алгоритм 4. Удвоение точки в проективном представлении.
Вход: $P = (X_1, Y_1, Z_1)$.
Выход: $R = P+P = (X_2, Y_2, Z_2)$.

1. $L_1 := 3X_1^2 + aZ_1^4$
2. $Z_2 := 2Y_1 Z_1$
3. $L_2 := 4X_1 Y_1^2$
4. $X_2 := L_1^2 - 2L_2$
5. $Y_2 := L_1(L_2 - X_2) - 8Y_1^4$

Вычислительная сложность удвоения точки оценивается как

$$I_4(l) = 4I_{mul}(l) + 6I_{sqr}(l) + 12I_{add}(l).$$

Отметим, что для ускорения выполнения описанных алгоритмов целесообразно применить арифметику Монтгомери [3]. Используя численные значения сложности операций арифметики многократной разрядности, полученные в [4], определим оценки сложности операций на эллиптической кривой при программной реализации:

$I_1(l) = 5{,}3l^3 + 74l^2 + 124l$
$I_2(l) = 5{,}3l^3 + 88l^2 + 144l$
$I_3(l) = 260l^2 + 379l$
$I_4(l) = 152l^2 + 316l$

Для сравнения построим графики полученных зависимостей (рис.1, 2).

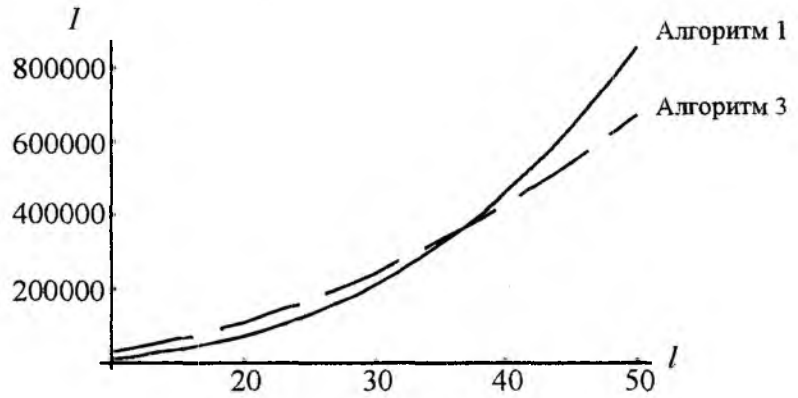

Рис.1

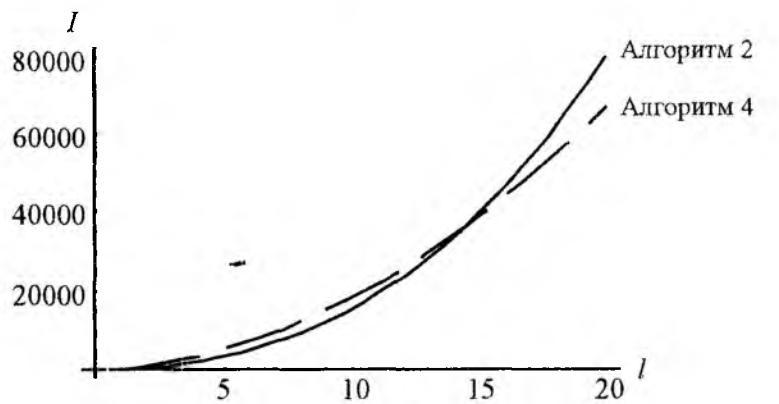

Рис.2

Очевидно, что для достаточно больших значений l $I_3<I_1$, $I_4<I_2$, т.е. в проективном представлении операции выполняются быстрее.

3. Алгоритмы скалярного умножения на эллиптической кривой

Для выполнения скалярного умножения $a\times P$ можно применить алгоритмы, аналогичные алгоритмам модульного возведения в степень – бинарного [4] и блочного [4,5].

Алгоритм 5. Бинарное скалярное умножение на эллиптической кривой.
Исходные данные: число $c\neq 0$, точка P, эллиптическая кривая $E = <a,b,p>$.
Результат: точка $Q = c\times P$.
1. Если $c=1$, то $Q:=P$; закончить работу алгоритма.
2. $k:=l_c-2$; $Q:=P$.
3. Для i, принимающего значения от k до 0, выполнить шаги 4-5.
4. $Q := Q+Q$.
5. Если i-й бит c равен 1, то $Q := Q+P$.
6. Закончить работу алгоритма.
Вычислительная сложность бинарного алгоритма составляет

$$I_{bin}(l_p, L_c) \approx L_c(I_{edbl}(l_p) + \rho I_{eadd}(l_p)), \qquad (2)$$

где L_c – длина числа c в битах;
ρ – доля единичных бит в числе c.
При $L_c = bl_p$, где b – размер слова в битах, выражение (2) примет вид

$$I_{bin} = (260\rho+152)bl_p^3 + (379\rho+316)bl_p^2 \qquad (3)$$

Алгоритм 6. Блочное скалярное умножение на эллиптической кривой.

Исходные данные: число $c \neq 0$, точка P, эллиптическая кривая $E = \langle a,b,p \rangle$.

Результат: точка $Q = c \times P$.

В алгоритме используется вспомогательная таблица $R[\,2^{d_{max}}\,]$.

В тексте алгоритма будут использоваться следующие обозначения [5]:

getblock(c, j) – операция выделения очередного блока из числа c, начиная с j-го бита;

end(C) – номер бита в числе c, на котором заканчивается блок C.

1. Если $c = 1$, то $Q:=P$; закончить работу алгоритма.
2. Вычислить $R[i]=i \times P$ для всех нечетных $i<2^m$.
3. $k := 1$; $Q := 1$.
4. Выделить блок: C:=**getblock**(c, 1).
5. Для k, принимающего значения от 1 до l_c, выполнить шаги 6-8.
6. $Q := Q+Q$
7. Если k = **end**(C) и $C \neq 0$, то: $Q := Q+R[C]$;
8. Если k = **end**(C), то C := **getblock**(c,k+1).
9. Закончить работу алгоритма.

Вычислительная сложность блочного алгоритма оценивается как:

$$I_{bl}(l_p, L_c) \approx I_{eadd}(l_p)(\,2^{d_{\max}-1} + \mu(L_c) + 1) + L_c I_{edbl}(l_p), \qquad (4)$$

где d_{max} – максимальная длина блока;

$\mu(L_c)$ – среднее количество блоков в числе c.

В [4] показано, что $\mu(L_c) = \dfrac{L_c}{2^{1-d_{\max}} + d_{\max}}$. Определим, при каком значении d_{max} функция (4) достигает минимума. Функцию (4) можно представить в виде суперпозиции двух функций: $I_{bl}=f(g(d_{max}))$, где $f(x)= x\, I_{eadd}(l_p) + (L_c-1) I_{edbl}(l_p)$; $g(d_{max})= 2^{d_{\max}-1} + \dfrac{L_c}{2^{1-d_{\max}} + d_{\max}} + 1$.

Т.к. $f(x)$ – монотонно возрастающая функция, то значение d_{max}, при котором функция $g(x)$ достигнет минимума, будет являться искомым значением d_0. Построив таблицу значений $g(x)$ для некоторых часто используемых длин показателя, найдем d_0 (см. табл. 1).

Таблица 1

x	L_c			
	160	192	224	256
3	54,2	64,1	73,9	83,8
4	47,8	55,5	63,3	71,1
5	48,6	54,9	61,2	67,6
6	59,5	64,8	70,1	75,4
7	87,8	92,4	96,9	101
d_0	4	5	5	5

Далее определим, при какой длине показателя степени блочный алгоритм скалярного умножения получает преимущество по быстродействию перед бинарным.

Используя (2) и (4), оценим выигрыш алгоритма 6 по сравнению с алгоритмом 5:

$$\Delta I = I_{bin}(l_p, L_c) - I_{bl}(l_p, L_c) =$$

$$= I_{eadd}(l_p)(\rho(L_c-1) - 2^{d_{\max}-1} - \frac{L_c}{2^{1-d_{\max}} + d_{\max}} - 1). \qquad (5)$$

Требуется найти значение L_c, при котором (5) обращается в 0.

Пусть $f(x)=\rho(x-1) - 2^{d_{\max}-1} - \dfrac{x}{2^{1-d_{\max}} + d_{\max}} - 1$. Тогда искомое значение $L(Y)$ совпадает с корнем уравнения $f(x) = 0$.

При ρ = 0,5 и d_{max} = 5 (см. табл. 1) это уравнение имеет корень $x_0 \approx 57{,}8$, следовательно, при $L_c \geq 58$ бит блочный алгоритм быстрее бинарного.

В алгоритме проверки цифровой подписи Эль-Гамаля на эллиптических кривых (например, ECDSA), встречается выражение вида $Q = a_1 \times P_1 + a_2 \times P_2$. Для ускоренного вычисления таких выражений можно использовать параллельный блочный алгоритм [4, 5].

Алгоритм 7. Параллельное блочное умножение на эллиптической кривой.

Исходные данные: числа c_i, точки P_i; $i=\overline{1,k}$; эллиптическая кривая E = <*a*,*b*,*p*>.

Результат: точка $Q = c_1 \times P_1 + c_2 \times P_2 + \ldots + c_k \times P_k$.

В алгоритме используется вспомогательная таблица $R[2^{d_{max}}, k]$.

1. $m := (\max L(Y_i)) - 1$; $Q := 0$.
2. Дополнить все c_i слева нулями до длины m бит.
3. Заполнение таблицы R:

$$R[i,j] := \begin{cases} 0; & j \bmod 2 = 0 \\ j \times P_i; & j \bmod 2 = 1 \end{cases}; \; i=\overline{1,k}; \; j=\overline{1, 2^{d_{max}} - 1}$$

4. Выделить блоки: C_i:=**getblock**(c_i, 1), $i=\overline{1,k}$.
5. Для j, принимающего значения от 1 до m, выполнить шаги 6-7.
6. Q := $Q+Q$.
7. Для i, принимающего значения от 1 до k, выполнить шаги 8-9.
8. Если j=end(C_i) и $C_i \neq 0$, то: Q:=Q+R[C_i,i].
9. Если j=end(C_i), то C_i:=**getblock**(c_i,j+1).
10. Закончить работу алгоритма. ▪

Вычислительная сложность алгоритма 7 оценивается как

$$I_{pm}(l_p, m, k) = mI_{edbl}(l_p) + \left(k\left(2^{d_{\max}-1}\right) + \sum_{i=1}^{k} \mu(c_i) \right) I_{eadd}(l_p),$$

а выигрыш по сравнению с алгоритмом 6 –

$$\Delta I = \left(\sum_{i=1}^{k} l_{c_i} - m \right) I_{edbl}(l_p).$$

Полученные оценки вычислительной сложности показывают, что арифметические операции на эллиптической кривой выполняются медленнее, чем соответствующие операции на GF(p) при одинаковой длине p. В то же время алгоритмы на эллиптических кривых обладают большей стойкостью: 160-битовое простое число в ECDSA соответствует по стойкости 1024-битовому числу в DSA, а 256-битовое - 2048-битовому [2].

4. Результаты экспериментальных измерений вычислительной сложности

В табл. 2 приведены временные характеристики несимметричных преобразований на эллиптических кривых, измеренные на Pentium-200 MMX.

Таблица 2

Длина ключа, бит	Время, мс		
	ECDSA, формирование	ECDSA, проверка	ECDH
160	22,7	28,0	22,7
192	31,5	40,3	31,5
224	42,6	55,0	42,6
256	56,5	73,0	56,5

Для сравнения приведем характеристики несимметричных преобразований в поле GF(p) (табл.3).

Таблица 3

Длина ключа и модуля, бит	Время, мс		
	DSA, формирование	DSA, проверка	DH
160 / 512	9,7	11,1	9,7
160 / 1024	31,2	37,2	31,2
256 / 2048	170,3	210,4	170,3

Приведенные результаты показывают, что с возрастанием длины ключевых параметров применение эллиптических алгоритмов является все более эффективным с точки зрения производительности: для ECDSA со 160-битовым ключом выигрыш составляет 37%, а для 256-битового ключа – 3 раза.

5. Заключение

Полученные результаты показывают, что применение цифровых подписей на эллиптических кривых над полем GF(p) может дать значительное повышение производительности систем защиты информации. В то же время интерес представляют исследования вычислительной сложности операций на эллиптических кривых над полем GF(2^m).

Список литературы: **1.** *ANSI X9.62.* Public Key Cryptography for the Financial Services Industry: The Elliptic Curve Digital Signature Algorithm (ECDSA). Draft. ANSI, 1998. **2.** *IEEE P1363.* Standard Specifications for Public-Key Cryptography. Draft. IEEE, 1999. **3.** *Горбенко И.Д., Качко Е.Г., Свинарев А.В.* Повышение быстродействия алгоритмов арифметики многократной точности // Безопасность информации, 1997. №1. С.6-12. **4.** *Свинарев А.В.* Методы и средства комбинированных несимметричных криптографических преобразований информации с уменьшенной вычислительной сложностью. Диссертация на соискание ученой степени кандидата технических наук по специальности 05.13.06. ХТУРЭ, Харьков, 1998. **5.** *Свинарев А.В.* Методы ускорения процедур цифровой подписи класса Эль-Гамаля // Радиотехника. Всеукр. межвед. науч.-техн. сб. 1997. Вып.104. С.173-178.

Харьковский государственный технический университет радиоэлектроники

Поступила в редколлегию 15.03.2000