A software implementation of NIST P-224

D. J. Bernstein
University of Illinois at Chicago
NSF CCR–9983950

`cr.yp.to/nistp224.html`

NIST P-224 is the elliptic curve $y^2 = x^3 - 3x + c_6$ over $\mathbf{Z}/p$.

Here $c_6 = $ 18958286285566608
000408668544493926
41550468096867932
1075787234672564

and $p = 2^{224} - 2^{96} + 1$.

Multiply $(10(2^{224}-1)/(2^8-1), \ldots)$ by n on the curve to get $(K_n, \ldots)$, for $n \in (\mathbf{Z}/\#\text{curve}(\mathbf{Z}/p))^*$.

Compressed Diffie-Hellman

$$\begin{tikzcd}
\text{Secret } K_{ab} & \text{Brian's public key } K_b \arrow[l] \\
\text{Alice's secret } a \arrow[u] \arrow[d] & \text{Brian's secret } b \arrow[u] \arrow[d] \\
\text{Alice's public key } K_a \arrow[r] & \text{Secret } K_{ab}
\end{tikzcd}$$

What `nistp224` does

`nistp224` is a new program to compute K_{ab} given a, K_b.

Alice puts 28 random bytes into `A`, 28 newlines into `K1`.

```
cat A K1 | nistp224 > KA
cat A KB | nistp224 > KAB
```

Also a C-language library:

```
unsigned char a[28];
unsigned char kb[28];
unsigned char kab[28];
nistp224(kab,kb,a);
```

58612 bytes for library on PIII.

Speed of version 0.76

Typical cycle counts, typical a's:

x	x, y	
595683	522639	Athlon
785900	668566	UltraSPARC
835530	734731	Pentium II
943244	827360	Pentium 4
1120824	985097	Pentium
1166080	1019027	RS64-III

x, y time does not depend on K_b.
Depends on a, i-cache state, etc.

923556, 864600, 864340, 864564, 864336, 864536, 864336, 864540, 864340, 864340, 881720, 879356, 864340, 864340, 864544, 864340, 864552, 864348, 864340, 864340, 864340, 864552, 864340, 864544, 878656, 864340, 884640, 864312, 864340, 864140, 864140, 864140

Floating-point arithmetic

A 64-**bit fp number**
is a real number $2^e f$
with $e, f \in \mathbf{Z}$ and $|f| < 2^{64}$.

Round each real number z to
closest 64-bit fp number, $\mathrm{fp}_{64}\, z$.
Round halves to even.

Given 64-bit fp numbers r, s (subject to limits on e), x86 chips can quickly compute $\text{fp}_{64}(r+s)$, $\text{fp}_{64}(r-s)$, $\text{fp}_{64}\, rs$.

If $r_0, s_0, r_1, s_1 \in \mathbf{Z}$, $|r_i| \le 2^{31}$, $|s_i| \le 2^{31}$, then
$r_0 s_1 = \text{fp}_{64}\, r_0 s_1$,
$r_1 s_0 = \text{fp}_{64}\, r_1 s_0$,
$r_0 s_1 + r_1 s_0 = \text{fp}_{64}(r_0 s_1 + r_1 s_0)$.

Carrying

Say $r = 31415926 \cdot 2^{28} + 53589793$.

Define $\alpha = 3 \cdot 2^{90}$,
$r_1 = \mathrm{fp}_{64}(\mathrm{fp}_{64}(r + \alpha) - \alpha)$.

Then $r_1 = 31415926 \cdot 2^{28}$
and $\mathrm{fp}_{64}(r - r_1) = 53589793$.

(Kahan 1965, et al.)

Arithmetic mod p

Can build big-integer arithmetic using floating-point operations. (Veltkamp 1968; Dekker 1971)

`nistp224` uses $\mathbf{Z}[2^{28}t] = \{\sum_{i\geq 0} g_i t^i : g_i \in 2^{28i}\mathbf{Z}\}$.

$\mathbf{Z}[2^{28}t] \to \mathbf{Z}/p$ by $g \mapsto g(1)$.

Normally use small polynomials:
$r = r_0 + r_1 t + r_2 t^2 + \cdots + r_7 t^7$
with $|r_i| \le 2^{28i} 2^{27} \cdot 1.01$.

If r and s are small:
Using fp can compute rs and
reduce mod $\mathrm{Ker}(\mathbf{Z}[2^{28}t] \to \mathbf{Z}/p)$
to a small polynomial.

Also $r^2 - 8s$, $r(4s - u) - 8v^2$, etc.

$\mathbf{Z}[100t] \to \mathbf{Z}/(10^6 - 4 \cdot 10^2 - 1)$.

$310000t^2 + 4100t + 51 \mapsto 314151$,

$140000t^2 - 1500t + 45 \mapsto 138545$.

Multiply and reduce:

	434	109	1494	1080	2295
4	34	109	1494	1080	2295
	34	125	1498	1080	2295
	35	25	1498	1080	2295
		25	1638	1115	2295
		25	1649	15	2295
		41	49	15	2295
			49	179	2336
			49	202	36
			51	2	36

Elliptic-curve arithmetic

Use Jacobian coordinates.
(Miller 1985, et al.)

$(x, y, z) \in (\mathbf{Z}/p)^3$, with $z \neq 0$
and with $y^2 = x^3 - 3xz^4 + c_6 z^6$,
represents $(x/z^2, y/z^3)$ on curve.

Use small polynomials q, r, s
to represent x, y, z.

Elliptic-curve doubling

Given (x_1, y_1, z_1) with $z_1 \neq 0$:

$2(x_1/z_1^2, y_1/z_1^3) = (x_2/z_2^2, y_2/z_2^3)$
where $\delta = z_1^2$, $\gamma = y_1^2$, $\beta = x_1\gamma$,
$\alpha = 3(x_1 - \delta)(x_1 + \delta)$,
$x_2 = \alpha^2 - 8\beta$, $z_2 = 2y_1z_1$,
$y_2 = \alpha(4\beta - x_2) - 8\gamma^2$.

4 squares, 4 mults, 8 reduces.

`nistp224` computes

$\delta = \text{reduce}\, s_1^2$,

$\gamma = \text{reduce}\, r_1^2$,

$\beta = \text{reduce}\, q_1\gamma$,

$\alpha = \text{reduce}\, 3(q_1 - \delta)(q_1 + \delta)$,

$q_2 = \text{reduce}(\alpha^2 - 8\beta)$,

$s_2 = \text{reduce}((r_1 + s_1)^2 - \gamma - \delta)$,

$r_2 = \text{reduce}(\alpha(4\beta - q_2) - 8\gamma^2)$.

5 squares, 3 mults, 7 reduces.

Elliptic-curve addition

Given (x_1, y_1, z_1) and (x_2, y_2, z_2) with $z_1 \neq 0$, $z_2 \neq 0$, and $(x_1/z_1^2, y_1/z_1^3) \neq (x_2/z_2^2, y_2/z_2^3)$: Use 4 squares and 12 mults to obtain sum (x_3, y_3, z_3).

Again eliminate one reduction. Could again trade mult for square.

Some of the intermediate results are z_1^2, z_1^3, z_2^2, z_2^3.

When reusing (x_1, y_1, z_1), also reuse z_1^2, z_1^3.

(Chudnovsky, Chudnovsky 1987; Cohen, Miyaji, Ono 1998)

Elliptic-curve multiplication

$a_0, \ldots, a_{27} \in \{0, 1, \ldots, 255\}$.
Define $a = 2^{216}(a_0 + 120) + 2^{208}(a_1 - 136) + \cdots + (a_{27} - 136)$.

`nistp224` uses simplest base-16 chain for a, coeffs $\{-8, -7, \ldots, 7\}$.
225 doubles, ≤ 59 adds.
Could eliminate a few adds.
Could exploit initial $z = 1$.

Reciprocals mod p

`nistp224` computes $p - 2$ power with obvious addition chain: 223 squares, 11 mults.

Simpler than Euclid, and time independent of input. However, Euclid is faster. Could use randomized Euclid.

Plans: better primes

Use prime in $3+4\mathbf{Z}$
for easier square root.

Use prime near power of 2
to chop carries in half.

Example of good prime: $2^{226}-5$.
Can use radix $2^{28.25}$.

Plans: better curves

Shape $y^2 = x^3 + c_2x^2 + x$ allows fast compressed multiplication. (Montgomery 1987)

x-coords of $2R, Q + 2R, 2Q + 2R$ are very simple functions of x-coords of $Q, R, Q + R$, when none of these points are ∞.

$y^2 = x^3 + 7530x^2 + x$, $p = 2^{226} - 5$.
Curve order $p + 1 - 1200040326\cdots$
is $16 \cdot$ prime. Use a's in $16\mathbf{Z}$.
Base $(53(2^{224} - 1)/(2^8 - 1), \ldots)$.
Can force $0 \le K_a < 2^{224}$.

Twist has order $8 \cdot$ prime,
so don't need to check whether
compressed input K_b is on curve.

Given K_b: For various n find x_n, z_n with $K_{nb} = x_n / z_n$.

From $K_b, x_n, z_n, x_{n+1}, z_{n+1}$ obtain $K_b, x_{2n}, z_{2n}, x_{2n+1}, z_{2n+1}$ or $K_b, x_{2n+1}, z_{2n+1}, x_{2n+2}, z_{2n+2}$ using 4 squares, 5 mults, and one easy mult by 1883.

No need for square roots.

Perhaps better to choose curve with another fast endomorphism. (Gallant, Lambert, Vanstone 2000)

In some cases can still use fast x-coordinate addition.