Choosing curves

D. J. Bernstein

University of Illinois at Chicago

Elliptic-curve Diffie-Hellman could use any elliptic curve E over any finite field $\mathbf{F}_q$.

Some choices of $E, \mathbf{F}_q$ are better than others.
Higher speed: easier to compute nth multiples in $E(\mathbf{F}_q)$.
Higher security: harder to find n given an nth multiple, i.e., to solve ECDLP.
Lower bandwidth. Etc.

How do we choose $E, \mathbf{F}_q$?
Which curves are best?

Occasionally an application has different criteria for $E, \mathbf{F}_q$.

e.g. Some cryptographic protocols need specific embedding degrees for pairings.

For simplicity I'll focus on fast, secure Diffie-Hellman.

Can also consider, e.g., genus-2 hyperelliptic curves. Better than elliptic curves? Active research area.

For simplicity I'll focus on the elliptic-curve case.

Field size?

The group $E(\mathbf{F}_q)$ has
$\approx q$ elements.

"Generic" algorithms such as
"Pollard's rho method"
solve ECDLP using
$\approx q^{1/2}$ simple operations.
Highly parallelizable.

e.g. $\approx 2^{40}$ simple operations
to solve ECDLP if $q \approx 2^{80}$.
Reject q: too small.

$q \approx 2^{256}$ is clearly safe
against these ECDLP algorithms.
$\approx 2^{128}$ simple operations
would need massive advances
in computer technology.

These algorithms can finish early,
but almost never do: e.g., chance
$\approx 2^{-56}$ of finishing after 2^{100}
simple operations. No serious risk.

Popular today: $q \approx 2^{160}$.
Somewhat faster arithmetic.
I don't recommend this; I can
imagine 2^{80} simple operations.

Field degree?

Field size q is a power of field characteristic p. Many possibilities for field degree $(\lg q)/(\lg p)$.

e.g. $q = 2^{255} - 19$; prime;
$p = 2^{255} - 19$; degree 1.

e.g. $q = (2^{61} - 1)^5$;
$p = 2^{61} - 1$; degree 5.

e.g. $q = 2^{255}$;
$p = 2$; degree 255.

What's the best degree?

Degree > 1 has a possible security problem: "Weil descent."

e.g. Degree divisible by 4 allows ECDLP to be solved with only about $q^{0.375}$ simple operations. Need to increase q, outweighing all known benefits. (Gaudry, Diem)

Other degrees are at risk too. Exactly which curves are broken by Weil descent? Very complicated answer; active research area.

Maybe we can be comfortable with degree > 1 despite Weil descent.

Standard argument for using
small characteristic, large degree:

Arithmetic on polynomials mod 2
is just like integer arithmetic
but faster: skip the carries.

Also have fast squarings.
Use fast curve endomorphisms.

Fewer bit operations
for scalar multiplication
in characteristic 2,
compared to large characteristic.
Speculation: ≈ 4 times fewer?

Counterargument:

Typical CPU includes circuits
for integer multiplication,
not for poly mult mod 2.

Large char is slower in hardware
than char 2, but
char 2 is *much* slower in software
than large char.

It seems to me that
the counterargument is winning:
char-2 standards are dying.

Medium char? Similar problems.

e.g. $q = (2^{31} - 1)^8$, $p = 2^{31} - 1$, degree 8, polys with coefficients in $\{0, 1, \ldots, 2^{31} - 2\}$:

Coefficient products fit comfortably into 64 bits. Also have fast inversion.

But hard to take advantage of 128-bit products; and hard to fit into 53-bit floating-point products. Big speed loss on many CPUs, outweighing all known benefits.

Prime shape?

Assume prime field from now on;
$\mathbf{F}_q = \mathbf{F}_p = \mathbf{Z}/p$.

How to choose prime p? Three common choices in literature.

"Binomial":
e.g., $2^{255} - 19$.

"Radix 2^{32}":
e.g., NIST prime $2^{224} - 2^{96} + 1$.

"Random":
no special shape for p.

Classic Diffie-Hellman had an
argument for random primes.

Here's the argument:
Best attack so far, namely
modern "NFS" index calculus,
is faster for special primes,
requiring larger primes,
outweighing any possible speedup.

Argument disappears for
elliptic curves over prime fields.
Attacker doesn't seem to
benefit from special primes;
don't have anything like NFS.

So choose prime
very close to power of 2,
saving time in field operations.

Binomial primes allow very fast
reduction, as we've seen.

Radix-2^{32} primes also allow
very fast reduction *if*
integer arithmetic uses radix 2^{32}.
Otherwise not quite as fast.
Different CPUs want
different choices of radix,
so binomial primes are better.

Which power of 2?

Primes not far below 2^{32w} allow field elements to fit in $4w$ bytes, minimal waste.

Comfortable security, $w = 8$: $2^{253} + 39$, $2^{253} + 51$, $2^{254} + 79$, $2^{255} - 31$, $2^{255} - 19$, $2^{255} + 95$. I recommend $2^{255} - 19$.

Subgroup shape?

Elliptic-curve Diffie-Hellman
uses standard base point B.
Bob's secret key is n;
Bob's public key is nB.

Order of B in group
should be a prime $\ell \approx q$.
Otherwise ECDLP is accelerated
by "Pohlig-Hellman algorithm."

This constrains curve choice:
number of elements of $E(\mathbf{F}_q)$
must have large prime divisor ℓ.

Quickly compute $\#E(\mathbf{F}_q)$, number of elements of $E(\mathbf{F}_q)$, using "Schoof's algorithm."

Enforce other constraints: $\gcd\{\#E(\mathbf{F}_q), q\} = 1$ to stop "anomalous curve attack"; large prime divisor of "twist order" $2q + 2 - \#E(\mathbf{F}_q)$ to stop "twist attacks"; large embedding degree to eliminate pairings.

Curve shape?

How to choose a_1, a_2, a_3, a_4, a_6
defining elliptic curve
$y^2 + a_1 xy + a_3 y =$
$x^3 + a_2 x^2 + a_4 x + a_6$?

See some coefficients
in explicit formulas
for curve operations.

e.g. Derivative $3x^2 + 2a_2 x + a_4$
usually creates mult by a_2.

But formulas vary: e.g.,
can see mult by $(a_2 - 2)/4$.

Save time in these formulas
by specializing coefficients.

e.g. $y^2 = x^3 - 3x + a_6$.

e.g. $y^2 = x^3 + a_2 x^2 + x$.

Many other interesting choices.

Warning: some specializations can
force low embedding degree or
otherwise create security problems.
Remember to check
all the security conditions.

Note on comparing curves
and comparing explicit formulas:
Count CPU cycles, not field ops!
Otherwise you make bad choices.

Reality: mult by small constant
is as expensive as several adds.

Reality: square-to-multiply ratio
is $2/3$ for a typical field,
not the often-presumed $4/5$.

Reality: $a^2 + b^2 + c^2$ is
faster than (a^2, b^2, c^2).

Current speed records use curve $y^2 = x^3 + a_2 x^2 + x$ with small $(a_2 - 2)/4$. Additional advantages: easily resist timing attacks; easily eliminate y.

$a_2 = 486662$ has near-prime curve order and twist order.

"Curve25519":
http://cr.yp.to/ecdh.html