Genus 2 curves in cryptography: successes and challenges

Craig Costello

AMS Special Session on Arithmetic Geometry
Las Vegas, Nevada
April 18, 2015
Mostly based on joint works with J. Bos, H. Hisil and K. Lauter

$\mathcal{K} = J_{\mathcal{C}}/\{\pm 1\}$

Traditional cryptographic groups

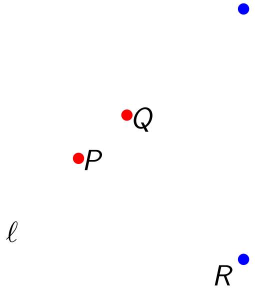
- To achieve public-key cryptography (e.g. secure the internet), we need **groups that facilitate computationally hard problems**, e.g:
    - the finite field DLP: given $g, g^\alpha \in \mathbb{F}_q^\times$, find α
    - the RSA problem: given $g^\alpha \in \mathbb{Z}_n$ and α, find g (where $n = pq$)
- these **traditional groups have problems**:
    - subexponential attacks against these problems have got better and better (index calculus, NFS: $L_{1/2}$, $L_{1/3}$, *quasi-polynomial*)
    - today, we want problems that take $\approx 2^{128}$ steps to solve $\implies q, n \approx 2^{3072}$
    - they're dead boring...

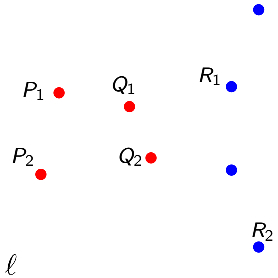
Better, "generic" cryptographic groups

- Jacobians of genus 1 and genus 2 curves both resist index calculus (as far as we know!)
- This talk: both will be defined over large prime fields $\mathbb{F}_p$
- **(H)ECDLP:** given P and $Q = [\alpha]P$ in $J_{\mathcal{C}}(\mathbb{F}_p)[N]$, find α
- Computing $\alpha, P \mapsto [\alpha]P$ needs $\approx \log_2(N)$ "double-and-adds"
- To attackers, they're as stubborn as a generic group. Pollard's (random walk) algorithm best: $O(\sqrt{N})$ steps (N large prime)
- But to cryptographers, they're far from generic: endomorphisms, Kummer varieties, torsion structure, etc

So what's better: genus 1 or genus 2?

Genus 1 versus Genus 2: points and Jacobian groups





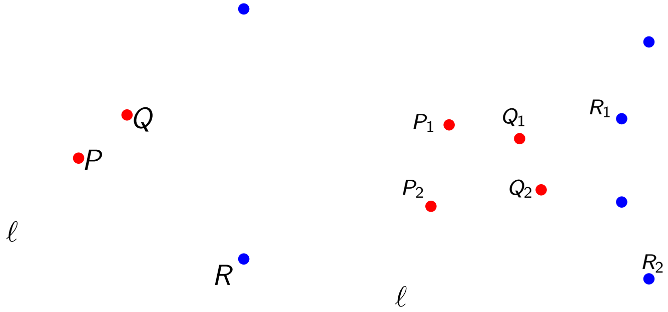
$J_{\mathcal{E}}(\mathbb{F}_p) \cong \mathcal{E}(\mathbb{F}_p)$ $\qquad$ $J_{\mathcal{C}}(\mathbb{F}_p) \ncong \mathcal{C}(\mathbb{F}_p)$

Mapping $P \in \mathcal{E}$ to $(P) - (\mathcal{O})$ in $\mathrm{Pic}^0(\mathcal{C})$ is a group homomorphism.

Challenge: you don't necessarily need to know what $\mathrm{Pic}^0(\mathcal{C})$ is to do ECC. This is not the case in genus 2.

Challenge: for a fixed group elt. P, there are $o(1)$ special points in $\mathcal{E}$, there are $O(p)$ special points in $J_{\mathcal{C}}$ ($\approx$ point of the talk!)

Genus 1 versus Genus 2: sizes of fields



$\#J_{\mathcal{E}}(\mathbb{F}_p) \approx p$ $\qquad$ $\#J_{\mathcal{C}}(\mathbb{F}_p) \approx p^2$

Challenge: Computing group law (additions/doublings) much more complicated in genus 2

Success: $p \approx 2^{256}$ for elliptic versus $p \approx 2^{128}$ for genus 2
(bonus: by far the fastest prime in software is $p = 2^{127} - 1$)

Jacobian coordinates in projective space

On elliptic curve $\mathcal{E}: \; y^2 = x^3 + ax + b$

Don't add (x_1, y_1) and (x_2, y_2) in $\mathbb{A}^2(K)$.

Add $(X_1 : Y_1 : Z_1)$ and $(X_2 : Y_2 : Z_2)$ in $\mathbb{P}(2,3,1)(K)$.

double-and-add uses 18 muls (compared to 7 muls + 2 invs)

[HC'14] On genus 2 curve $\mathcal{C}: \; y^2 = x^5 + a_4 x^4 + \cdots + a_0$

Don't work with points $(x, y) \in \mathbb{A}^2(K)$.

Work with $(X : Y : Z) \in \mathbb{P}(2,5,1)(K)$, i.e. $(x, y) = (X/Z^2, Y/Z^5)$

Translate into Mumford coords $(x^2 + u_1 x + u_0, v_1 x + v_0) \in J_{\mathcal{C}}$

$$(u_1, u_0, v_1, v_0) \leftrightarrow \left(\frac{U_1}{Z^2}, \frac{U_0}{Z^4}, \frac{V_1}{Z^3}, \frac{V_0}{Z^5} \right)$$

double-and-add uses 63 muls (compared to 46 muls + 2 invs, or 82 muls for homogeneous projection)

Speed comparison for general Weierstrass curves

Success: Nowadays genus 2 CM method and point counting make it possible to find cryptographically strong genus 2 curves, e.g. [GS'12] counted points to give curve below.

g	curve	work	coords	prime p	cycles/scalar mult.
1	generic $\mathcal{E}$	[BCLN'15]	Jacobian	$2^{256} - 189$	**278,000** (SB)
2	generic $J_{\mathcal{C}}$	[BCHL'13]	homog.	$2^{127} - 1$	**243,000** (IB)
	generic $J_{\mathcal{C}}$	[HC'14]	Jacobian	$2^{127} - 1$	**195,000** (IB)

Timings on Intel Core i7 3.4GHz (Sandy Bridge (SB) and Ivy Bridge (IB))

Success: Fair to say generic genus 2 at least as fast (if not faster) than generic genus 1 (at 128-bit level)

In fact, w.r.t speed, the story only gets better for genus 2

Speed comparisons cont.

g	curve	work	coords	prime p	cycles
1	generic $\mathcal{E}$	[BCLN'15]	Jacobian	$2^{256}-189$	**278,000** (SB)
	Kummer	[B'06→14]	Mont-ladder	$2^{255}-19$	**194,000** (SB)
2	generic $J_{\mathcal{C}}$	[BCHL'13]	homog.	$2^{127}-1$	**243,000** (IB)
	generic $J_{\mathcal{C}}$	[HC'14]	Jacobian	$2^{127}-1$	**195,000** (IB)
	Kummer $\mathcal{K}_{\mathcal{C}}$	[BCLS'14]	theta	$2^{127}-1$	**89,000** (SB)

- Montgomery: work with $x \in \mathcal{E}/\{\pm 1\}$, not $(x, y) \in \mathcal{E}$
- Genus 2 Kummer: work on $\mathcal{K}_{\mathcal{C}} \cong J_{\mathcal{C}}/\{\pm 1\}$, not $J_{\mathcal{C}}$
- $J_{\mathcal{C}}$ described 72 quadratic forms in $\mathbb{P}^{15}$ (written down by Flynn)
- Kummer (Gaudry in crypto): $\mathcal{K}_{\mathcal{C}}$ described by one quartic in $\mathbb{P}^4$, i.e. projective points $P = (X : Y : Z : T)$ on

$$E \cdot XYZT = ((X^2+Y^2+Z^2+T^2) - F(XT+YZ) - G(XZ+YT) - H(XY+ZT))^2$$

Success: Over prime fields, and at the 128-bit level, it should be fair to say that genus 2 is MUCH faster than genus 1 ...

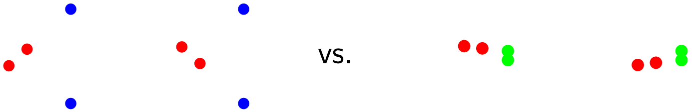
So why aren't they in the current debate?

x–coordinate only arithmetic

Montgomery's arithmetic: $By^2 = x^3 + Ax^2 + x$

$$x_{[2]T} = \mathtt{DBL}(x_T, A)$$

$$x_{T+P} = \mathtt{PSEUDOADD}(x_T, x_P, x_{T-P})$$



- opposite y's give different x-coordinate than same-sign y's
- decide between them with difference x_{T-P}
- "Differential" additions: $x_{T+P} = \mathtt{PSEUDOADD}(x_T, x_P, x_{T-P})$
- **Can exponentiate:** intermediate points $[n]P$ and $[n+1]P$ (difference P invariant)
- **Can't add generically:** Kummers are restricted ($\approx$ to DH) in crypto, can't do traditional signatures or complex protocols

Real world problems facing $J_{\mathcal{C}}$

- **Success:** Genus 2 Kummer is by far the best prime field option out there!
- Why not use $\mathcal{K}_{\mathcal{C}}$ for key agreement and $J_{\mathcal{C}}$ for everything else?
- **Reason/challenge:** There isn't one addition formula that handles all points in $J_{\mathcal{C}}$ – this makes writing "constant-time" code extremely difficult/cumbersome/slow for $J_{\mathcal{C}}$ (Cantor's algorithm variable time and very "branchy")
- See disclaimers: Assumption 1 and Section 7.3 in [HC'14]
- So many special cases: e.g. $O(p)$ "degenerate" divisors with one rational element in support (just the beginning)

Two questions

Question 1: does this happen with the Kummers too?

Answer: nope, no exceptions to differential additions. $J_{\mathcal{C}} \longrightarrow \mathcal{K}$ kills two-torsion and all divisors work in the addition formula (Riemann relations)

Question 2: does this happen in genus 1?

Answer: yes and no. For generic Weierstrass curves, group law has exceptional points (and different cases). But genus 1 has one advantage here: non-generic, non-Weierstrass models ...

e.g. (Twisted) Edwards curves and complete formulas

$\mathcal{O}$ P_2

P_1 P_3 $-P_3$

$\mathcal{O}'$

On $\mathcal{E}/K\colon -x^2+y^2=1+dx^2y^2$, if d is non-square in K, then

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

works for all (x_1,y_1) and (x_2,y_2) in $E(K)$, including $x_1=x_2$, the neutral point $(0,1)$, etc.

No special cases means easier constant-time code

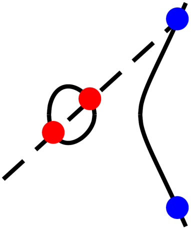
$\leftrightarrow$ **real-world adoption**

The (sad) situation for general elliptic curves

Algorithm 19 Complete (projective) addition using masking and Jacobian coordinates on prime-order Weierstrass curves E_b.

Input: $P, Q \in E_b(\mathbf{F}_p)$ such that $P = (X_1, Y_1, Z_1)$ and $Q = (X_2, Y_2, Z_2)$ are in Jacobian coordinates.
Output: $R = P + Q \in E_b(\mathbf{F}_p)$ in Jacobian coordinates. Computations marked with [*] are implemented in constant time using masking.

$T[0] = \mathcal{O}$ $\quad \{T[i] = (\tilde{X}_i, \tilde{Y}_i, \tilde{Z}_i)$ for $0 \leq i < 5\}$
$T[1] = Q$
$T[4] = P$
$t_2 = Z_1^2$
$t_3 = Z_1 \times t_2$
$t_1 = X_2 \times t_2$
$t_4 = Y_2 \times t_3$
$t_3 = Z_2^2$
$t_5 = Z_2 \times t_3$
$t_7 = X_1 \times t_3$
$t_8 = Y_1 \times t_5$
$t_1 = t_1 - t_7$
$t_4 = t_4 - t_8$
index $= 3$
if $t_1 = 0$ **then** [*]
  index $= 0$ $\quad \{R = \mathcal{O}\}$
  if $t_4 = 0$ **then** index $= 2$ $\quad \{R = 2P\}$ [*]
if $P = \mathcal{O}$ **then** index $= 1$ $\quad \{R = Q\}$ [*]
if $Q = \mathcal{O}$ **then** index $= 4$ $\quad \{R = P\}$ [*]
mask $= 0$
if index $= 3$ **then** mask $= 1$
{case $P + Q$, else any other case} [*]
$t_3 = X_1 + t_2$
$t_6 = X_1 - t_2$
if mask $= 0$ **then** $t_2 = Y_1$ **else** $t_2 = t_1$ [*]
$t_5 = t_2^2$
if mask $= 0$ **then** $t_7 = X_1$ [*]
$t_1 = t_5 \times t_7$
$\tilde{Z}_2 = Z_1 \times t_2$
$\tilde{Z}_3 = Z_2 \times \tilde{Z}_2$
if mask $\neq 0$ **then** $t_3 = t_2$ [*]
if mask $\neq 0$ **then** $t_6 = t_5$ [*]
$t_2 = t_3 \times t_6$
$t_3 = t_2/2$
$t_3 = t_2 + t_3$
if mask $\neq 0$ **then** $t_3 = t_4$ [*]
$t_4 = t_3^2$
$t_4 = t_4 - t_1$
$\tilde{X}_2 = t_4 - t_1$
$\tilde{X}_3 = \tilde{X}_2 - t_2$
if mask $= 0$ **then** $t_4 = \tilde{X}_2$ **else** $t_4 = \tilde{X}_3$ [*]
$t_1 = t_1 - t_4$
$t_4 = t_3 \times t_1$
if mask $= 0$ **then** $t_1 = t_5$ **else** $t_1 = t_8$ [*]
if mask $= 0$ **then** $t_2 = t_5$ [*]
$t_3 = t_1 \times t_2$
$\tilde{Y}_2 = t_4 - t_3$
$\tilde{Y}_3 = \tilde{Y}_2$
$R = T[\text{index}]$ $\quad (= (\tilde{X}_{\text{index}}, \tilde{Y}_{\text{index}}, \tilde{Z}_{\text{index}}))$ [*]
return R



Bosma-Lenstra: in general, need at least two sets of formulas, e.g.

$$\begin{aligned}
X_3 &= (X_1Y_2 - X_2Y_1)(Y_1Z_2 + Y_2Z_1) - (X_1Z_2 - X_2Z_1)(a(X_1Z_2 + X_2Z_1) + 3bZ_1Z_2 - Y_1Y_2);\\
Y_3 &= -(3X_1X_2 + aZ_1Z_2)(X_1Y_2 - X_2Y_1) + (Y_1Z_2 - Y_2Z_1)(a(X_1Z_2 + X_2Z_1) + 3bZ_1Z_2 - Y_1Y_2);\\
Z_3 &= (3X_1X_2 + aZ_1Z_2)(X_1Z_2 - X_2Z_1) - (Y_1Z_2 + Y_2Z_1)(Y_1Z_2 - Y_2Z_1);\\
X_3' &= -(X_1Y_2 + X_2Y_1)(a(X_1Z_2 + X_2Z_1) + 3bZ_1Z_2 - Y_1Y_2) - (Y_1Z_2 + Y_2Z_1)(3b(X_1Z_2 + X_2Z_1) + a(X_1X_2 - aZ_1Z_2));\\
Y_3' &= Y_1^2Y_2^2 + 3aX_1^2X_2^2 - 2a^2X_1X_2Z_1Z_2 - (a^3 + 9b^2)Z_1Z_2^2 + (X_1Z_2 + X_2Z_1)(3b(3X_1X_2 - aZ_1Z_2) - a^2(X_2Z_1 + X_1Z_2));\\
Z_3' &= (3X_1X_2 + aZ_1Z_2)(X_1Y_2 + X_2Y_1) + (Y_1Z_2 + Y_2Z_1)(Y_1Y_2 + 3bZ_1Z_2 + a(X_1Z_2 + X_2Z_1)).
\end{aligned} \tag{1}$$

. . . see [BCLN'14] for more discussion. . .

Real-world status

g	curve	work	formulas	prime p	cycles
1	Montgom. Edwards	[B'06→14] [BDLSY'11]	ladder complete	$2^{255}-19$ $2^{255}-19$	**194,000** (SB) **230,000** (??)
2	Kummer ??	[BCLS'14] ??	theta complete	$2^{127}-1$??	**89,000** (SB) ??

- **Challenge:** fill in the ??'s
- Highly desirable to find a non-Weierstrass model to mimic genus 1 non-Weierstrass completeness
- Or, highly desirable to achieve completeness via other means

References

[B'06] Berstein. *Curve25519: new Diffie-Hellman speed records*. http://cr.yp.to/ecdh/curve25519-20060209.pdf

[BDLSY'11] Bernstein-Duif-Lange-Schwabe-Yang. *High-speed high-security signatures*. http://ed25519.cr.yp.to/ed25519-20110926.pdf

[BCHL'13] Bos-C-Hisil-Lauter. *Fast cryptography in genus 2*. http://eprint.iacr.org/2012/670.pdf

[HC'14] Hisil-C. *Jacobian coordinates on genus 2 curves*. http://eprint.iacr.org/2014/385.pdf

[BCLS'14] Bernstein-Chuengsatiansup-Lange-Schwabe. *Kummer strikes back: new DH speed records.* https://eprint.iacr.org/2014/134.pdf

[BCLN'15] Bos-C-Longa-Naehrig. *Selecting elliptic curves for cryptography: an efficiency and security analysis.* http://eprint.iacr.org/2014/130.pdf